UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
CURSO DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Desenvolvimento e análise de desempenho de um 'Packet/Session Filter'

por

MARCO AURÉLIO SPOHN

Dissertação submetida à avaliação, como requisito parcial para a obtenção do grau de Mestre em Ciência da Computação

Prof. Dr. Raul Fernando Weber
Orientador

Porto Alegre, novembro de 1997.

CIP - CATALOGAÇÃO NA PUBLICAÇÃO

Spohn, Marco Aurélio

Desenvolvimento e análise de desempenho de um 'Packet/Session Filter' / por Marco Aurélio Spohn. - Porto Alegre: CPGCC da UFRGS, 1997.

67 f. : il.

Dissertação (mestrado) - Universidade Federal do Rio Grande do Sul. Curso de Pós-Graduação em Ciência da Computação, Porto Alegre, BR-RS, 1997. Orientador: Weber, Raul Fernando.

1. Filtro de pacotes. 2 Session Filter. 3. Internet Firewall. 4. Segurança na Internet. I. Weber, Raul Fernando. II. Título.

UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
Reitora: Profa. Dra. Wrana Panizzi
Pró-Reitor Pós-Graduação: Prof. Dr. José Carlos Ferraz Hennemann
Diretor do Instituto de Informática: Prof. Dr. Roberto Tom Price
Coordenador do CPGCC: Prof. Dr. Flávio Rech Wagner
Bibliotecária-Chefe do Instituto de Informática: Zita Prates de Oliveira

Para minha querida Tânia.

Agradecimentos

Agradeço sobretudo ao apoio e orientação do Professor Raul Fernando Weber, ao CNPq pelo apoio financeiro sem o qual a realização desse trabalho teria sido muito mais difícil, ao Instituto de Informática e ao Grupo de Tolerância a Falhas pelos recursos disponibilizados.

Não poderia deixar de registrar os meus agradecimentos a minha querida e amada esposa Tânia, cuja presença e amor tem sido fundamental em minha vida. Meus agradecimentos ao meu ilustre irmão Marcelo que, apesar da distância física, tem demonstrado seu apoio e energia. Agradeço também aos meus pais Verno e Remi, que de uma forma ou outra sempre tem me apoiado e motivado.

Para não ser injusto ao citar nomes, agradeço de uma forma geral a todos os demais amigos que me ajudaram direta ou indiretamente na realização desse trabalho.

Sumário

Lista de Figuras

Lista deTabelas

Lista de Abreviaturas

ACK (bit)	Acknowledgment bit
BSD	Berkeley Software Distribution
DNS	Domain Name Service
DOS	Disc Operating System
FTP	File Transfer Protocol
HTTP	Hypertext Transfer Protocol
ICMP	Internet Control Message Protocol
IP	Internet Protocol
NFS	Network File System
NNTP	Network News Transfer Protocol
OSI	Open Systems Interconnect
PC	Personal Computer
SMTP	Simple Mail Transfer Protocol
SNMP	Simple Network Management Protocol
SYN	Synchronize sequence number
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
WWW	World Wide Web

Resumo

A segurança em sistemas de computação é uma das áreas mais críticas nas ciências da informação. Frequentemente se ouve falar em alguma "brecha" descoberta em algum sistema. Não importando quão grave seja o problema, já existe uma certa paranóia em relação a questões de segurança. Esses furos na segurança podem ser explorados de diversas maneiras: para obter benefícios financeiros indevidos, personificação maliciosa a custos de uma pretensa "brincadeira", perda de informações sigilosas, danos em arquivos do sistema, etc. Esses são alguns dos motivos porque se investe cada vez mais em segurança. Como se não bastasse, ainda existem os problemas relativos a legislação: por um lado, o governo pode vir a exigir que a segurança em sistemas de informação seja encarada como um problema federal e impor controle sobre tudo que seja utilizado nos mecanismos de proteção; por outro lado, outros podem reclamar a necessidade de privacidade e liberdade de escolha em problemas relativos a segurança.

Os sistemas são, a medida que agregam mais recursos, muito complexos e extensos, propiciando um meio fértil para acúmulo de erros e conseqüentes problemas de segurança.

A conectividade alterou a realidade da computação: as redes de computadores estão aí para provar. Entretanto, os sistemas em rede a medida que facilitam a realização de uma série de tarefas também apresentam vulnerabilidades. As redes públicas são um exemplo crítico, pois todas as informações passam por meios não confiáveis sujeitos a alteração ou utilização por terceiros.

A Internet talvez represente um dos ambientes mais críticos a nível de segurança. Devido a sua abrangência e velocidade com que cresce, ela é um dos ambientes mais propícios a disseminação e exploração de "furos" de segurança. Existem, entretanto, um esforço constante em desenvolvimento de mecanismos para protegê-la.

"Internet Firewalls", são um conjunto de mecanismos utilizados para formar uma barreira de segurança entre a rede interna e a Internet, isolando-a das principais ameaças. Os componentes básicos de um "firewall" são o "packet filter" e o "proxy server". Basicamente, o que esses componentes fazem é uma filtragem dos pacotes. O "packet filter", como o nome sugere, faz uma filtragem dos pacotes até o nível de transporte (UDP e TCP). O "proxy server" atua como um procurador entre o cliente e o servidor real realizando uma filtragem a nível de aplicação; portanto, é uma tarefa mais apurada. Cada um desses componentes básicos apresentam os seus benefícios à segurança da rede interna.

Um dos problemas desses mecanismos, é que eles acarretam em um "overhead" ao sistema, degradando o desempenho. Um "packet filter" tem de ser suficientemente rápido para que não seja o gargalo do sistema. De fato, como é apresentado nesse trabalho, pode-se conseguir um filtro tão rápido quanto necessário.

A filtragem dos pacotes é baseada nas regras de filtragem que são elaboradas segundo a política de segurança, a qual estabelece qual o tipo de tráfego que pode existir entre a rede interna e a Internet. Cada pacote que passa pelo filtro é comparado com as regras de filtragem na ordem em que elas foram especificadas pelo administrador. Dependendo do número de regras, e este é um fator relevante porque são em grande número os serviços utilizados na rede, o tempo médio de filtragem aumenta.

Uma solução para melhorar o desempenho na filtragem, é realizar a filtragem dos pacotes por sessão ("session filter"). Nesse caso somente se consegue atender aos serviços baseados no TCP (que é orientado a conexão); entretanto, considerando que os serviços mais utilizados na rede são baseados no TCP esse é um mecanismo viável.

Na filtragem por sessão apenas o pacote de solicitação de conexão é comparado com a lista de regras de filtragem. Os pacotes subsequentes são verificados junto a uma "cache" de sessões ativas. Caso o pacote pertença a alguma sessão válida ele é passado adiante; caso contrário, ele é descartado. O acesso a "cache" deve ser suficientemente rápido para justificar esse procedimento. Além do ganho em desempenho, o filtro de sessões apresenta a vantagem de monitoramento: todas as sessões ativas entre a rede interna e a Internet estão registradas na "cache".

Esta dissertação apresenta o projeto e a implementação de um "Packet/session filter". O filtro foi implementado como um módulo do "kernel" do sistema operacional "FreeBSD". O filtro "ip_fw", disponível como "freeware" na referida plataforma, serviu como referência básica para o desenvolvimento do filtro proposto. O utilitário "ipfw", disponível para gerenciar o filtro de pacotes "ip_fw", foi adaptado para interagir com o filtro desenvolvido.

Os testes de desempenho apresentam resultados esperados. Ou seja, o filtro de sessão melhora consideravelmente o processo de filtragem em relação a um filtro convencional. O recurso de monitoramento das sessões ativas também representa uma facilidade a mais no controle e obtenção de estatísticas para o "firewall".

Palavras-chave: Filtro de Pacotes, Session Filter, Internet Firewalls, Segurança na Internet.

TITLE: "Development and performance analysis of a packet/session Filter"

Abstract

Security in computer systems is one of the most critical areas in "information sciences". Often it is heard something about a new "hole" discovered in some system. No matter how important is such a problem, there is a "paranoia" regarding questions about security. These security holes could be explored in a diversity of ways: to obtain financial benefits, to impersonate somebody to spoofing, to obtain secret informations, to damage to file systems, etc. Those are some of the reasons because so much time and money are spent in security. There are also some problems concerning legislation: government can demand that security in information systems is a federal problem and it can lead to impose control over everything that is used in protection mechanisms; on the other hand, there is the question of privacy and freedom of choice concerning to security.

By adding new resources to the systems, the complexity is increase and new bugs and security problems can arise.

Connectivity has changed the computer reality: computer networks show that. However, network systems also present weak points. Public networks are a critical example because all information flow by untrusted medias subject to modification or misuse by outsiders.

The Internet may be one of the most critical environments concerning security. Because the internet covers almost all the world and grows so fast, it's able to dissiminate "holes" in security. There is, however, a constant effort to develop new mechanisms to protect the Internet.

Internet Firewalls are a set of mechanisms used to build a security barrier between the internal network and the internet, maintaining the internal network far away from the main threats. The basic components of a firewall are the packet filter and the proxy server. These two components basically do a packet screening. The "packet filter", as the name suggests, makes a packet filtering up to the transport layer (UDP and TCP). The Proxy Server acts like a proxy between the client and the real server, the proxy does a packet filtering at the application level; therefore, it's a refined task. Each one of these components present their own benefits to the internal network security.

A problem of those mechanisms is that they bring an overhead to the system, slowing the performance. A packet filter must be fast enough, otherwise it'll be the bottleneck in the system. The present work slows that, it's possible to obtain a filter as fast as necessary.

The packet filtering is based on the filter rules which are prepared following the security policy that establishes what kind of traffic can flow between the internal network and the Internet. Each packet that passes through the filter is compared with the filtering rules to check if it is in the sequence specified by the administrator. Depending on the number of rules, and this is an important issue because there is a great number of services available in the network, the filtering mean time increases.

A solution to improve the filtering process is to make the packet filtering by session (session filter). In this case it's only possible to attend to TCP, based services (connection oriented); however, considering that the most used services in the internet are based on TCP this is a feasible mechanism.

In the session filtering only the first packet is compared against the filtering rules. The next packets are verified against a session cache. If the packet refers to a valid session it's sent forward; otherwise, it's dropped. The access time to the cache must be fast enough in order to allow applying this procedure. Beyond the performance improvement the session filter presents the advantage of monitoring: all the active sessions are recorded in the "cache".

This work presents the project and development of a "packet/session filter". The filter was developed as a kernel module of the operating system "FreeBSD". The filter "ip_fw" available as freeware served as a basic reference to the implementation of the filter here presented. The "ipfw" utility available to manage the "ipfw" packet filter was adapted to interact to the developed filter .

The performance benchmark presents expected results. The session filter enhances the filtering process when compared to the conventional packet filter. The monitoring facility also represents an enhancement the control and statistics gathering.

Keywords: Packet Filter, Session Filter, Internet Firewalls, Internet Security.

1 Introdução

A segurança em sistemas de computação tem se apresentado como um requisito indispensável, sobretudo em se tratando de sistemas interconectados através de redes de computadores públicas. Existem ameaças de segurança que comprometem desde máquinas isoladas (por exemplo: vírus) até toda uma rede (ex: "vermes"). Sistemas em rede incrementam ainda mais as possibilidades de se explorar vulnerabilidades existentes.

A medida que novas vulnerabilidades de segurança são exploradas surgem mecanismos para tentar diminuir as possibilidades de ataque. Os anti-vírus, por exemplo, são constantemente atualizados porque outros vírus são desenvolvidos a todo momento.

Os mecanismos de segurança abrangem desde proteção de uma única máquina a também toda uma rede. Entretanto, diversos níveis de "falhas" na segurança são apontados em diferentes sistemas; ou seja, muitas vezes mecanismos genéricos não são possíveis de serem aplicados porque as diversas plataformas envolvidas apresentam características intrínsecas. A grande variedade de dispositivos e sistemas dificulta ainda mais o processo de desenvolvimento de ferramentas de segurança. Por exemplo, o *Unix* apresenta uma série de "bugs" de segurança; entretanto, diferentes plataformas *Unix* existem no mercado, cada uma com suas potencialidades e fraquezas.

Pela inexistência de uma nomenclatura estabelecida em português, e pelo fato da maioria das referências utilizadas serem em inglês, serão adotados neste trabalho, tanto nesta seção como nas demais, diretamente os nomes em inglês.

1.1 Estratégias de Segurança

As estratégias de segurança são empregadas para tornar mais robusto o sistema de segurança. Segundo [CHA 95] algumas estratégias adotadas são as seguintes:

- "Least privilege": segue o princípio do privilégio mínimo, onde qualquer objeto (usuário, programa, sistema, etc) possui apenas os privilégios estritamente necessários para executar as suas tarefas;
- "Defense in depth": utilizar redundância em mecanismos de segurança, caso um componente falhe há outro(s) garantindo a segurança;
- "Choke point": é o ponto de acesso por onde tudo que transita entre a rede interna e externa deve necessariamente passar; portanto é um ponto que possibilita um alto controle e monitoramento;
- "Weakest link": consiste em determinar qual o ponto mais fraco no sistema de segurança;
- "Fail safe": em caso de falha o sistema se mantém em um estado seguro (indisponível mas seguro);
- "Universal participation": a participação de todos os usuários, funcionários, administradores, etc, é indispensável para que um bom sistema de segurança seja eficiente. A educação dos usuários é fundamental para o sucesso na

segurança ("do que adianta um forte sistema de segurança quando os usuários utilizam senhas facilmente adivinháveis?");

- "Diversity of Defense": investir em diferentes sistemas de segurança dificulta o trabalho do atacante porque ele precisa explorar as vulnerabilidades de sistemas diferentes;
- Simplicidade: sistemas simples são mais seguros, a complexidade esconde "furos" difíceis de serem percebidos.

1.2 Postura

É importante que a postura do sistema de segurança seja bem definida. As duas principais posturas são as seguintes:

- Postura padrão de negação: especificar apenas o que é permitido e proibir o resto;
- Postura padrão de permissão: especificar somente o que é proibido e permitir o resto.

A primeira postura é "prudente", a segunda é "permissiva". Ou seja, a prudente estabelece que "tudo aquilo que não é expressamente permitido é proibido" e a permissiva diz que "tudo aquilo que não é expressamente proibido é permitido".

Certamente a postura mais segura é a prudente. Deve-se permitir somente aquilo que se considera seguro; do contrário, "furos" em serviços não seguros e legais podem ser facilmente aproveitados caso sejam descobertos. A experiência vivenciada na "Texas A&M University" [SAF 92] reforça a importância na determinação da postura. Vítimas de uma série de ataques, eles foram forçados a desenvolver mecanismos de proteção; entretanto, a postura adotada incialmente foi a permissiva. Os intrusos começaram a aplicar ataques contra outros serviços ainda não protegidos mas permitidos. A partir do momento que foi adotada a postura prudente os ataques cessaram.

1.3 Segurança na Internet

A Internet, pela sua própria natureza, apresenta uma série de furos na segurança. Na própria rede se obtém recursos para realizar ataques sem precisar ser um "expert" no assunto. Essas e outras ameaças tornam a Internet um ambiente inseguro. Existem mecanismos, tais como "Internet firewalls", que podem ser utilizados para minimizar essas ameaças.

1.3.1 "Internet Firewalls"

"Internet Firewalls" são mecanismos utilizados para proteger uma rede conectada à Internet [SPO 96][CHA 95]. Operam como uma barreira de segurança entre a rede interna e a Internet, tudo o que sai e entra na rede interna passa pelo "firewall" [SIY 95](Fig. 1.1).

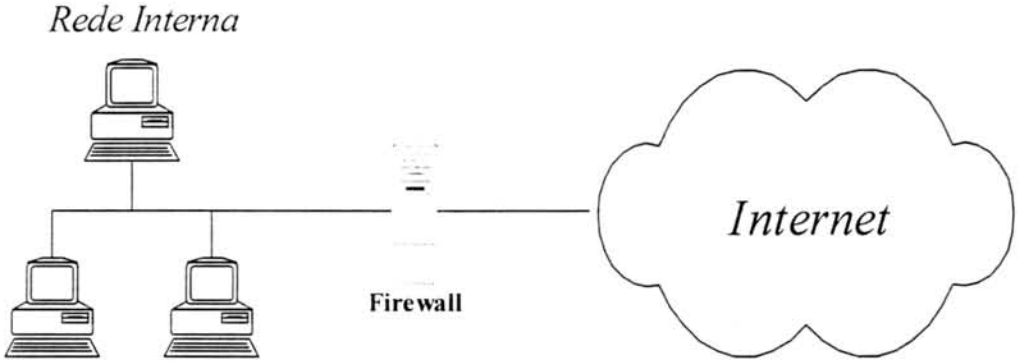


FIGURA 1.1 - "Internet Firewall"

Quando se conecta à Internet, põe-se em risco três coisas:

- Os dados: as informações armazenadas nos computadores;
- Os recursos: os próprios computadores;
- A reputação da instituição.

Com relação aos dados, tem-se três características que precisam ser protegidas:

- Segredo (privacidade);
- Integridade;
- Disponibilidade.

Há três principais riscos associados com os serviços fornecidos pela rede privada (serviços "inbound") aos usuários externos na Internet:

- "Hijacking": situação em que alguém rouba uma conexão depois que o usuário legítimo já realizou a sua autenticação;
- "Packet Sniffing": situação em que alguém lê dados confidenciais enquanto eles transitam pela rede, sem interferir com a conexão;
- "False authentication": situação em que alguém "engana" o sistema de autenticação.

Algumas tarefas cabíveis a um "firewall" são as seguintes:

- Atuar como um "checkpoint"; ou seja, ser o foco para as decisões referentes à segurança, o ponto de conexão com o mundo externo, tudo o que chega à rede interna deve passar por ele;
- Aplicar a política de segurança;
- Registrar ("logging") eficientemente as atividades na Internet;
- Limitar a exposição da rede interna ao mundo externo.

Algumas tarefas que um *firewall* não pode realizar (pelo menos atualmente) são as seguintes:

- Proteger a rede interna contra usuários internos mal intencionados: "se o inimigo mora dentro da própria casa, certamente não será esta uma morada segura";
- Proteger de conexões que não passam pelo "firewall": "do que adianta colocar uma porta da frente em aço maciço e uma dúzia de fechaduras se alguém deixou a porta dos fundos aberta?";

- Proteger contra ameaças completamente novas: "qual será o próximo furo a ser descoberto?"
- Proteger contra vírus.

Segundo [RAN 93] é importante examinar como o "firewall" atende a cada um dos seguintes quesitos:

Controle do dano - Caso ele seja comprometido, a que tipos de ameaças a rede privada fica exposta? E caso seja destruído?

Zonas de risco - Qual é o tamanho da zona de risco durante operação normal? Uma medida é o número de "hosts" ou roteadores que podem ser sondados da rede exterior;

Modo de falha - Se ele é "quebrado", quão facilmente isto pode ser detectado? E caso ele seja destruído? E "*post mortem*", quanta informação é retida e que pode ser utilizada para diagnosticar o ataque?

Facilidade de uso - Quão inconveniente ele é?

Postura - A filosofia básica do projeto é "Tudo que não é expressamente permitido é proibido" ou "Tudo que não é expressamente proibido é permitido"?

1.3.1.1 Componentes de um "firewall"

A barreira criada pelo "firewall" é construída a partir de alguns componentes básicos. Estes componentes atuam do nível de rede ao nível de aplicação (modelo OSI [TAN 89]). De fato, esses componentes examinam o tráfego entre a rede interna e a Internet; ou seja, realizam uma filtragem dos pacotes que trafegam entre as duas redes. Essa filtragem é menos ou mais apurada dependendo dos componentes, os quais são os seguintes:

- **"Packet filter" (filtro de pacotes)**: filtra os pacotes utilizando as informações dos cabeçalhos dos protocolos IP, ICMP, TCP e UDP (Fig. 1.2). A filtragem é baseada em regras especificadas em uma lista de acessos; o pacote é barrado ou liberado dependendo das regras de filtragem;

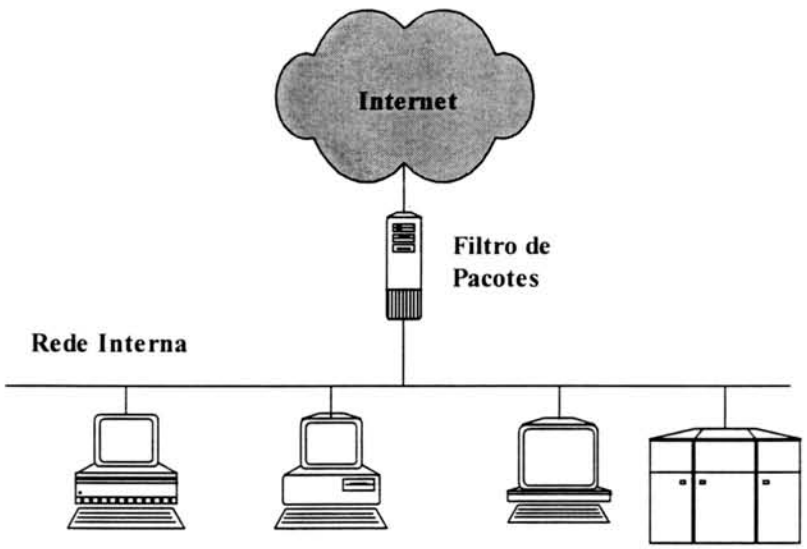


FIGURA 1.2 - "Packet Filter"

- **"Proxy server"**: atua como um procurador entre o cliente e o servidor. Realiza uma filtragem dos pacotes até o nível de aplicação. Todos os pacotes que trafegam entre o cliente e o servidor real passam, necessariamente, primeiro pelo "proxy server" e este, após o devido tratamento de filtragem, o encaminha adiante (Fig. 1.3). O servidor procurador pode ser dedicado (atende a uma única aplicação) ou genérico (atende a um conjunto de aplicações). É imprescindível que o servidor procurador seja mais robusto que um servidor convencional; caso contrário, de nada adianta utilizá-lo como uma "barreira".

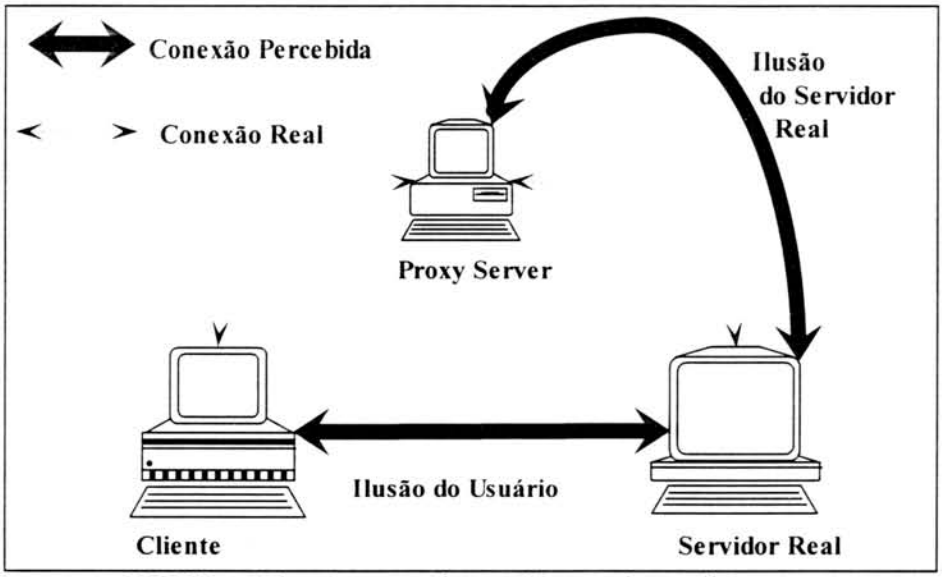


FIGURA 1.3 - Atuação de um "Proxy Server"

1.3.1.2 "Bastion Host"

As máquinas presentes no "firewall" configuradas para serem responsáveis por algum mecanismo da segurança (com *proxy servers*, por exemplo) são denominadas de **"bastion hosts"**. Ou seja, "bastion host" é qualquer máquina configurada para desempenhar algum papel crítico na segurança da rede interna; constituindo-se na presença pública na Internet, provendo os serviços permitidos segundo a política de segurança.

Dependendo da localização do "bastion host" dentro do "firewall", tem-se alguns tipos de máquinas com funções diferenciadas na segurança [CHA 95], as quais são:

- "Dual homed host": trata-se de um computador com duas interfaces de rede conectadas cada uma a segmentos diferentes de rede. Uma das características fundamentais dessa configuração é que o roteamento direto (IP forwarding) é desabilitado e, portanto, todo o roteamento é realizado a nível de aplicação. Neste caso, todos os serviços segurados podem ser fornecidos via procuração ("proxy servers") e somente o tráfego referente aos serviços habilitados via "proxy" e aqueles especificados pelas regras de filtragem circulam entre os dois segmentos de rede conectados ao "bastion host" (FIG. 1.4);

- "Victim machines": estas máquinas abrigam serviços que não são considerados fáceis de serem segurados. A máquina é configurada basicamente somente com os serviços fornecidos para garantir que nada mais significativo esteja a disposição do atacante caso a máquina seja comprometida;
- "Internal bastion hosts": são aquelas máquinas com maior interação com as máquinas internas (por exemplo, uma máquina que recebe o correio eletrônico e o reenvia a um servidor de correio eletrônico residente na rede interna).

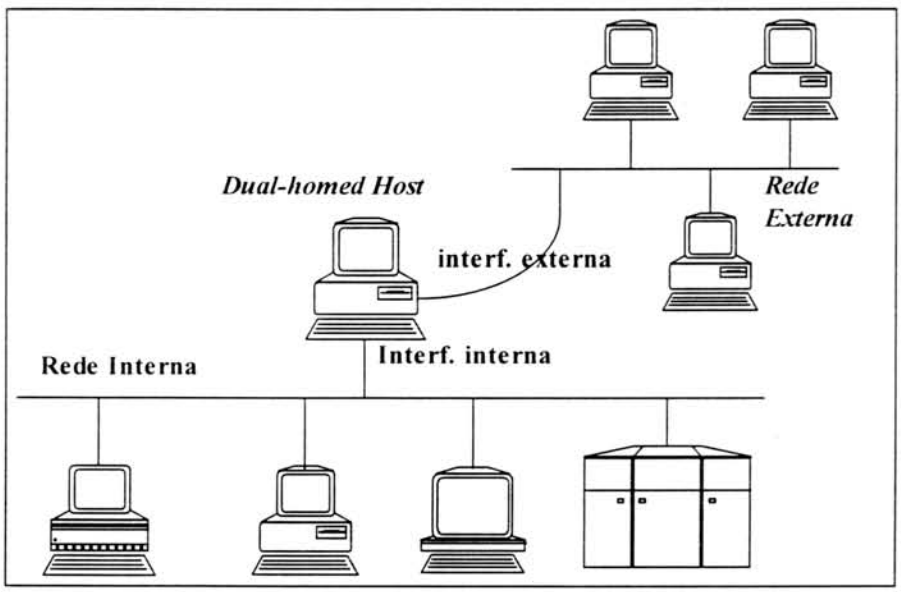


FIGURA 1.4 - Dual-homed Host

1.3.2 Arquiteturas de "Firewall"

O *Firewall* consiste em um conjunto de componentes agrupados de uma forma a garantir certos requisitos de segurança. Determinadas arquiteturas recebem denominações especiais e uma infinidade de variantes podem ser obtidas a partir destas.

As arquiteturas "screened subnet" e "screened host" merecem destaque porque constituem estruturas básicas montadas a partir dos componentes "packet filter" e "bastion host", permitindo que a partir dessas configurações possam ser elaboradas uma série de variantes de acordo com a necessidade de proteção desejada para a rede interna.

1.3.2.1 "Screened Subnet"

Esta é uma arquitetura que apresenta múltiplos níveis de redudância e provê um bom esquema de segurança, sendo um exemplo clássico em arquiteturas de "firewall".

Os componentes que a compõe são os seguintes (FIG. 1.5):

- "Perimenter Network": é uma sub-rede isolada situada entre a rede interna e a rede externa (Internet);

- Filtro de pacotes externo: diretamente conectado à Internet e à "perimeter network";
- Filtro de pacotes interno: diretamente conectado à rede interna e à "perimeter network";
- "Bastion hosts" presentes na "perimeter network".

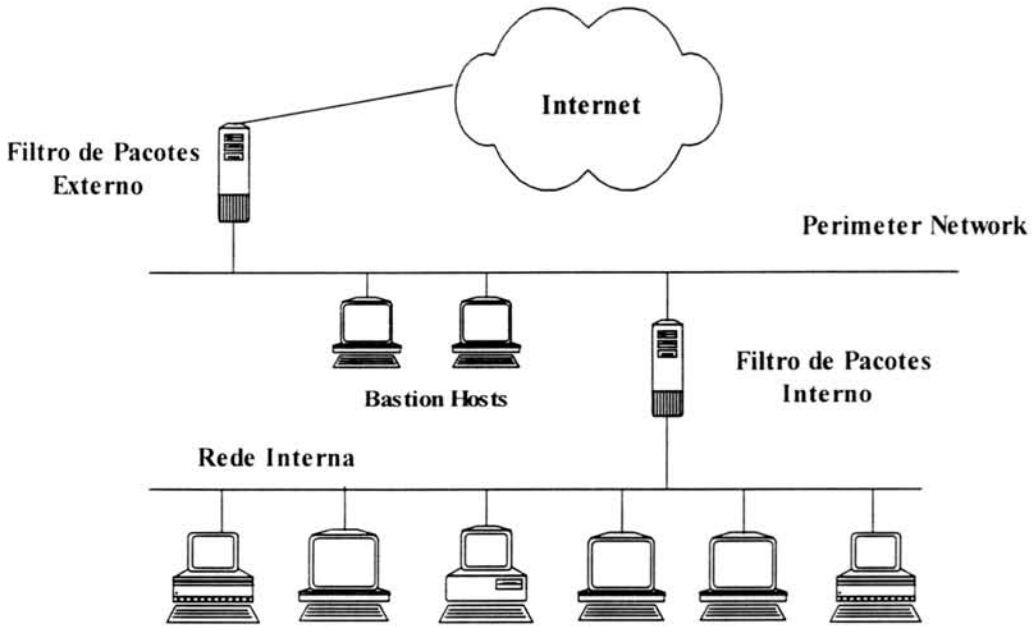


FIGURA 1.5 - "Sreened Subnet"

Definidos os serviços a serem providos, pode-se elaborar as regras de filtragem para os filtros externo e interno. Uma alternativa consiste em repetir algumas das regras de filtragem adotadas no filtro externo também no filtro interno, de forma que o interno ainda se comporte como uma barreira caso o externo seja comprometido.

Nessa arquitetura os "proxy servers" devem estar localizados nos "bastion hosts", sobretudo os procuradores de serviços "inbound". Dessa forma, caso um servidor procurador seja comprometido o atacante ainda tem de passar pelo filtro de pacotes interno.

Para forçar o acesso aos serviços por procuração (residentes nos "bastion hosts"), o servidor externo e interno devem estar apropriadamente configurados de forma que os procuradores não sejam desviados. Os clientes da rede interna acessam os serviços que possuem "proxy" por intermédio dos "bastion hosts" e estes, por sua vez, atuam como clientes dos servidores na rede externa.

1.3.2.2 "Screened Host"

Nesta arquitetura não há uma sub-rede de segurança ("perimeter network") entre a Internet e a rede Interna. Existem apenas um filtro de pacotes e um "bastion host" situado junto à rede interna (FIG. 1.6).

Esta é uma arquitetura de custo mais baixo que a "screened subnet"; entretanto, também é menos segura, basta que o filtro seja comprometido para que toda a rede

interna fique exposta para o ataque, simplesmente porque o "bastion host" pode ser desviado já que é o filtro quem força o tráfego atráves do "bastion host". As regras de filtragem devem estabelecer que somente o tráfego proveniente do "bastion host" (proxy server) possa passar diretamente pelo filtro de pacotes. Os usuários da rede interna interagem com os procuradores residentes no "bastion host" e este, por sua vez, atua em nome do cliente com o mundo externo (internet).

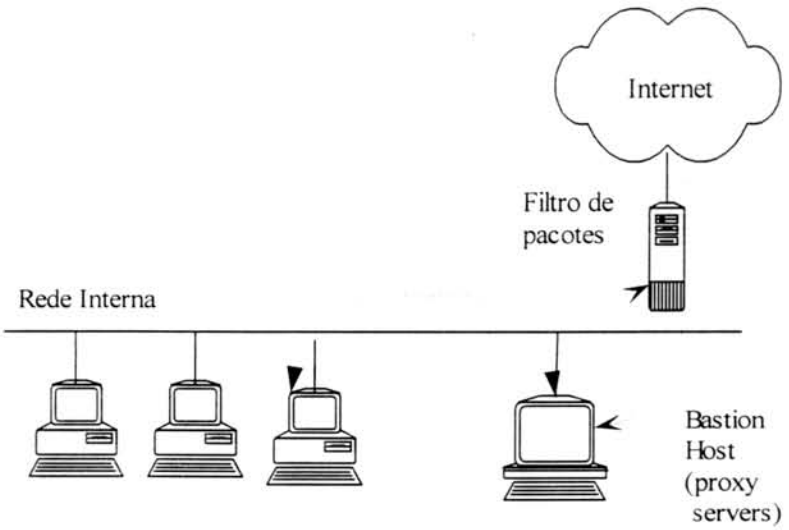


FIGURA 1.6 - "Screened Host"

1.3.2 "Firewall": O futuro?

Existem especialistas que demonstram dúvidas quanto ao futuro dos "firewalls". Marcus Ranum[RAN 96-1] [RAN 96-2], um especialista muito conhecido na área, acredita que os "firewalls" são um mecanismo que protegem tentando tapar furos que deveriam ter sido resolvidos na própria fonte do problema (no projeto dos sistemas deveria se dar mais ênfase a aspectos de segurança). Entretanto, ele salienta que os "firewalls" ainda serão necessários, da forma como o são, porque esta estratégia parece não estar sendo adotada pelos desenvolvedores de sistemas. Todavia, as barreiras de segurança sempre terão de existir mas não necessariamente nos moldes atuais. A medida que as aplicações evoluem na prática de mecanismos de criptografia e autenticidade (assinatura digital, "one time passwords", etc) ficará mais fácil construir e manter uma barreira de segurança.

2 Filtragem de Pacotes

Como um primeiro passo ao se implementar uma barreira de segurança em uma rede de computadores, é fundamental que se conheça os detalhes dos protocolos de comunicação utilizados. Na Internet, a atenção deve ser voltada aos protocolos IP, TCP, ICMP e UDP. Estes são os principais protocolos a nível de rede e transporte que são considerados e examinados ao se estabelecer regras de filtragem para um filtro de pacotes na Internet.

A filtragem de pacotes permite controlar o tipo de tráfego em qualquer segmento de rede; conseqüentemente, é possível controlar os serviços que trafegam pela rede. Serviços que comprometem a segurança da rede podem ser restringidos.

Deve-se ressaltar que o processo de filtragem de pacotes causa um "overhead" ao sistema; portanto, para uma situação de tráfego intenso é necessário que se coloque o filtro em uma máquina com uma velocidade de processamento compatível com as necessidades.

2.1 Procedimento básico de filtragem

A filtragem convencional de pacotes na Internet atua até o nível de transporte (UDP e TCP), ficando os níveis superiores a cargo dos servidores procuradores. Portanto, qualquer problema de segurança instrínseco aos protocolos superiores (aplicação, por exemplo) não pode ser resolvido utilizando um filtro de pacotes convencional.

A filtragem dos pacotes é realizada utilizando as seguintes informações presentes nos cabeçalhos dos pacotes:

- Endereço IP fonte;
- Endereço IP destino;
- Protocolo (TCP, UDP ou ICMP);
- "Port" TCP ou UDP fontes;
- "Port" TCP ou UDP destino;
- Tipo de mensagem ICMP (se for o caso).

No protocolo TCP existe um "flag" denominado ACK que é utilizado para confirmação de pacotes e também pode ser utilizado para detectar se o pacote é o primeiro de uma solicitação de conexão. Quando o "flag" não estiver marcado ("bit" com valor 1) significa que o pacote se refere a uma solicitação de conexão e, caso contrário, o pacote corresponde a alguma conexão já existente. Desta forma, o "packet filter" pode bloquear um serviço *inbound* (de fora para dentro; ou seja, o servidor está na rede interna) apenas não permitindo o fluxo de pacotes com o ACK marcado destinado a um servidor interno associado a "port" (por exemplo, a port 23 do telnet) do serviço bloqueado. Em protocolos não orientados a conexão, como é o caso do protocolo UDP, não é possível tomar nenhuma decisão deste tipo; ou seja, nestes

protocolos, nunca se sabe se o pacote que está chegando é o primeiro que o servidor está recebendo.

Para fazer uma filtragem correta dos pacotes, é importante saber se o protocolo é bidirecional (pacotes fluem nos dois sentidos, cliente para servidor e vice-versa) ou unidirecional. Não se pode confundir serviços "inbound" (a rede interna provendo algum serviço) e serviços "outbound" (o cliente está na rede interna e o servidor na Internet) com pacotes "inbound" (pacotes que chegam na rede interna) e pacotes "outbound" (pacotes que saem da rede interna); ou seja, ambos os serviços apresentam pacotes "inbound" e "outbound" caso o protocolo seja bidirecional

É importante que o "packet filter" tenha facilidades de filtragem por interfaces de rede. Ou seja, todas as interfaces disponíveis são submetidas às regras de filtragem, possibilitando que as regras sejam aplicadas considerando as seguintes informações:

- A interface na qual o pacote chega;
- A interface pela qual o pacote sai.

2.2 Regras de filtragem

No filtro de pacotes existe uma lista de regras de filtragem aplicadas para cada pacote que trafega pelo filtro. Esta lista de regras é elaborada segundo a política de segurança adotada, onde é estabelecido o que pode e o que não pode passar pelo filtro. A estrutura básica de uma regra de filtragem é apresentada na FIGURA 2.1.

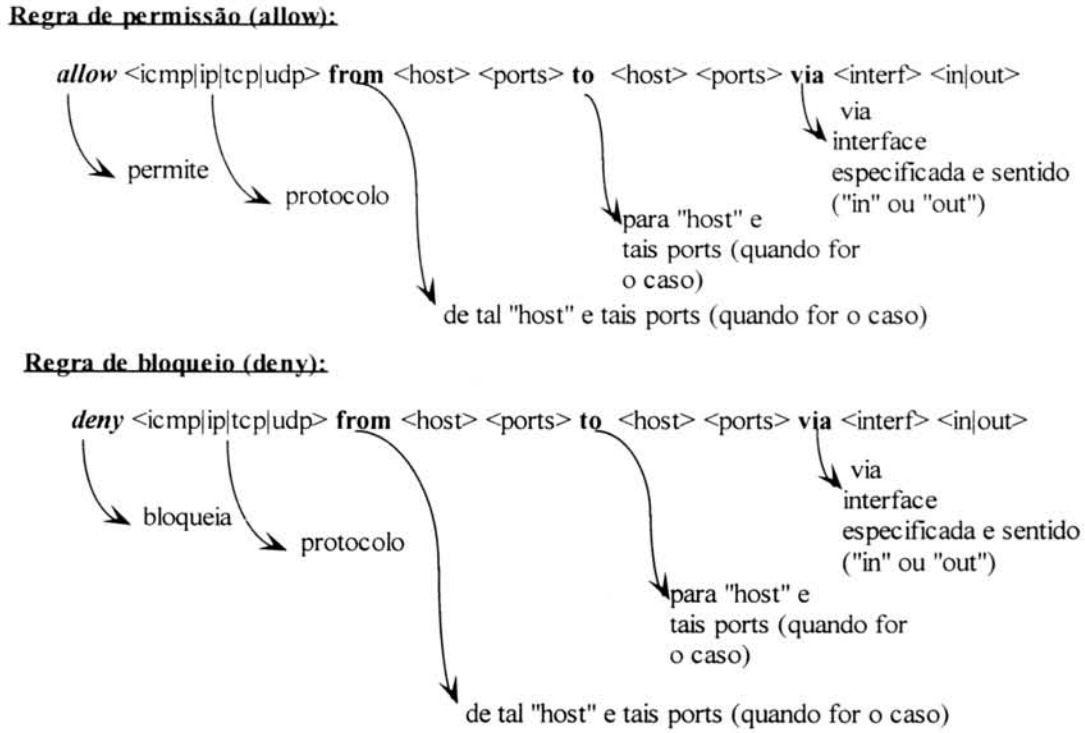


FIGURA 2.1 - Estrutura de uma regra de filtragem

Não existe um padrão de sintaxe de regras de filtragem. É aconselhável montar uma tabela com as possibilidades de pacotes para cada serviço a ser provido, e a partir dessa tabela escrever as regras de filtragem. Veja um exemplo na TABELA 2.1.

Descrição dos campos da tabela: "Direção", é a direção do pacote (entrando no filtro ou saindo do filtro); "Endereço Fonte", é o endereço IP fonte; "Endereço Destino", é o endereço IP destino; "protocolo", é o protocolo utilizado (udp ou tcp); "Port fonte", é a porta fonte; "Port Destino", é a porta destino; ACK, indica se o flag ACK está marcado (valor 1); "comentário" apresenta um breve comentário sobre o pacote.

TABELA 2.1 - Possibilidade de pacotes no protocolo DNS

Direção	Endereço Fonte	Endereço Destino	Protocolo	Port Fonte	Port Destino	ACK	Comentário
Entrando	Externo	Interno	UDP	> 1023	53	a	Cliente externo para servidor interno
Saindo	Interno	Externo	UDP	53	> 1023	a	Servidor interno para cliente externo
Entrando	Externo	Interno	TCP	> 1023	53	b	Cliente externo para servidor interno
Saindo	Interno	Externo	TCP	53	> 1023	Sim	Servidor interno para cliente externo
Saindo	Interno	Externo	UDP	> 1023	53	a	Cliente interno para servidor externo
Entrando	Externo	Interno	UDP	53	> 1023	a	Servidor externo para cliente interno
Saindo	Interno	Externo	TCP	> 1023	53	b	Cliente interno para servidor externo
Entrando	Externo	Interno	TCP	53	> 1023	Sim	Servidor externo para cliente interno
Entrando	Externo	Interno	UDP	53	53	a	Comunicação entre dois servidores via UDP
Saindo	Interno	Externo	UDP	53	53	a	Comunicação entre dois servidores via UDP
Entrando	Externo	Interno	TCP	> 1023	53	b	Resposta do servidor externo para servidor interno via TCP
Saindo	Interno	Externo	TCP	53	> 1023	Sim	Resposta do servidor interno para servidor externo via TCP
Saindo	Interno	Externo	TCP	> 1023	53	b	Solicitação de servidor interno para serv. Ext.
Entrando	Externo	Interno	TCP	53	> 1023	Sim	Resposta de servidor externo para servidor interno via TCP

a Pacotes UDP não possuem o ACK bit

b O ACK bit não está marcado no primeiro pacote deste tipo (estabelecimento de conexão) mas estará marcado no resto

2.3 Operações de um "Packet filter"

O comportamento básico de um filtro de pacotes convencional é o seguinte[SIY 95] [SPO 96] [CHA 94]:

1. As regras de filtragem são armazenadas em uma lista na ordem em que foram especificadas.
2. Quando um pacote chega em uma determinada interface, os cabeçalhos do pacote são analizados.
3. Cada regra é aplicada ao pacote na ordem em que as regras estão armazenadas.
4. Se uma regra bloqueia a transmissão ou recepção do pacote, o pacote é bloqueado.

5. Se uma regra permite a transmissão ou recepção do pacote, o pacote é aceito para prosseguir.
6. Se um pacote não satisfaz qualquer regra ele é bloqueado (postura prudente).

Pelas regras 4 e 5 fica evidente que a ordem das regras de filtragem é de fundamental importância. Uma ordenação incorreta das regras pode acarretar em bloqueio de serviços válidos e em permissão de serviços que deveriam ser negados [CHA 95]. Portanto, nenhuma otimização deve ser feita na ordem de aplicação das regras de filtragem. Da regra 6 segue a filosofia "O que não é expressamente permitido é proibido".

2.4 Vantagens e Desvantagens

Algumas vantagens dos "packet filters" são [CHA 95]:

- O filtro pode ajudar a proteger toda uma rede, principalmente caso ele seja o único ponto de conexão da rede interna à Internet;
- A filtragem de pacotes é transparente e não requer conhecimento nem cooperação dos usuários;
- Há uma série de produtos comerciais e "freeware" disponíveis;
- A maioria dos roteadores (CISCO, por exemplo) apresentam recursos de filtragem de pacotes.

Algumas desvantagens são:

- Alguns protocolos não são bem adaptados para a filtragem;
- Algumas políticas não podem ser aplicadas somente com a filtragem de pacotes.

Quando se aplica alguma restrição em algum protocolo de mais alto nível, através de números de "ports", espera-se que nada além do próprio serviço esteja associado àquela "port"; entretanto, um usuário interno mal intencionados pode subverter este tipo de controle substituindo o servidor por outro desenvolvido por ele. Como citado anteriormente, um "firewall" não é apropriado para se defender de ameaças internas.

2.5 Ações do Filtro

A máquina encarregada da filtragem dos pacotes pode executar uma série de atividades que servem, entre outras coisas, para monitorar o sistema. Algumas atividades são:

- Realizar "logs" de acordo com a configuração especificada pelo administrador. Dessa forma, é possível analisar eventuais tentativas de ataque, bem como verificar a correta operação do sistema;
- Retorno de mensagens de erros ICMP: caso um pacote seja barrado existe a possibilidade de se enviar ao endereço fonte alguma mensagem com o código de erro ICMP do tipo "host unreachable" ou "host administratively unreachable". Entretanto, tais mensagens, além de causar um "overhead",

podem fornecer algumas informações sobre o filtro ao atacante, pois dessa forma ele poderia descobrir quais os protocolos que são barrados e quais estão disponíveis; portanto, recomenda-se que não se retorne nenhum código ICMP de erro para hosts na rede externa.

2.6 Características desejáveis em um filtro

Eis algumas características altamente desejáveis a fim de que se possa realizar uma filtragem de pacotes bem apurada [CHA 95]:

- Ter uma boa performance na filtragem dos pacotes: um "overhead" aceitável de acordo com as necessidades;
- Pode ser um roteador dedicado ou um computador de propósito geral;
- Permitir uma especificação de regras de forma simples;
- Permitir regras baseadas em qualquer cabeçalho ou critério "meta packet" (por exemplo, em qual interface o pacote chegou ou está saindo);
- Aplicar as regras na ordem especificada;
- Aplicar as regras separadamente para pacotes que chegam e partem em e de cada interface de rede;
- Registrar informações sobre pacotes aceitos e rejeitados;
- Ter capacidade de teste e validação.

2.7 Filtragem versus Ataques

Há uma série de ataques que se consegue evitar com a filtragem de pacotes. De uma forma geral, se evitam todos os ataques aos serviços que não são permitidos pelo filtro. Ou seja, o intruso não consegue sondar "ports" e "hosts" a revelia.

O "IP spoofing" (Fig. 2.2) é um ataque que pode ser evitado com a filtragem de pacotes. Neste ataque o intruso tenta se passar por host confiável utilizando como endereço fonte o endereço IP de alguma máquina permitida pelo filtro. Se a filtragem é realizada por interface em ambos os sentidos e o atacante estiver utilizando o endereço de alguma máquina da rede interna, este ataque não funciona porque jamais um pacote pode chegar da rede externa (Internet) tendo como endereço fonte o endereço de uma máquina que está na rede interna; ou seja, só poderia chegar naquela interface no outro sentido.

Para evitar que o "IP spoofing" seja utilizado por usuários da rede interna, basta que o filtro barre todo pacote enviado para a Internet que tenha como endereço IP fonte algum endereço diferente dos endereços utilizados na rede interna.

Geralmente o ataque de "IP spoofing" é utilizado conjuntamente com a opção "source route" do protocolo IP. Essa opção permite que o pacote seja roteado segundo a rota especificada no próprio pacote. Dessa forma, mesmo que o atacante utilize um endereço IP fonte qualquer, os roteadores pelos quais o pacote passar o enviarão segundo a rota constante no pacote, sem levar em consideração os recursos de roteamento do roteador. Ou seja, o atacante garante dessa forma que a resposta também

retornará para ele (pacotes de retorno tem como endereço destino o endereço fonte forjado pelo intruso). É fundamental, portanto, que o filtro descarte qualquer pacote IP que utilize a opção de "source routing".

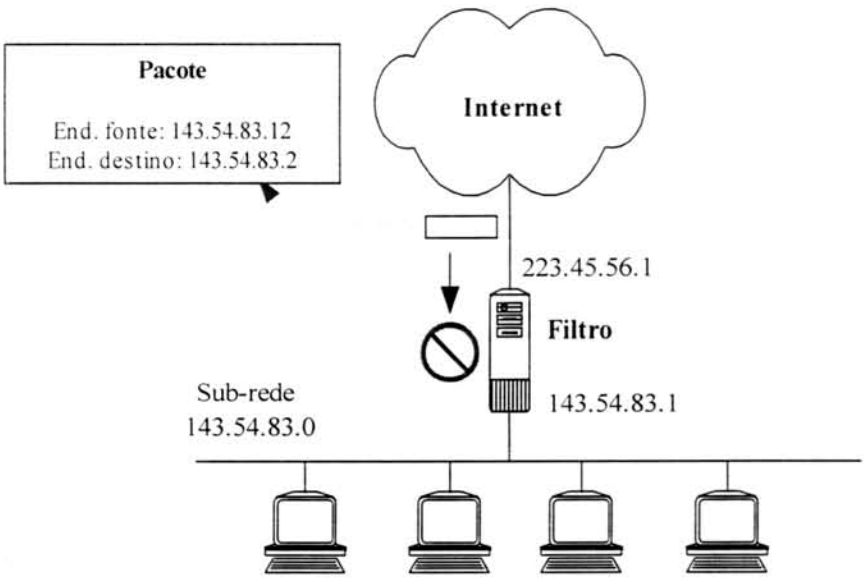


FIGURA 2.2 - Exemplo de "IP spoofing"

O "syn flooding" é outro ataque que também utiliza "IP spoofing". Esse ataque explora a deficiência no tratamento de conexões pendentes. A tabela de conexões pendentes é limitada e a liberação de uma entrada é realizada após um certo tempo ("timeout"). O ataque acontece da seguinte forma:

1. O atacante utiliza como endereço fonte o de alguma máquina que esteja inoperante;
2. Monta um pacote de solicitação de conexão e o envia à máquina alvo; repete o processo até que a tabela de conexões pendentes da vítima esteja cheia (as tabelas são relativamente pequenas, por exemplo: NT, 7 entradas; SunOS, 10 entradas);
3. Periodicamente o atacante repete o processo para novamente encher a tabela.

Esse ataque somente funciona utilizando o endereço fonte de alguma máquina inoperante porque, caso contrário, ela enviaria um pacote "reset" (com o bit RST com valor 1) e a entrada na tabela de conexões pendentes seria liberada no servidor. Todavia, caso o atacante utilize um PC com algum utilitário desenvolvido para o DOS, por exemplo, com a única tarefa de aplicar o ataque, o sistema não enviará pacotes de "reset" porque ele simplesmente descartaria os pacotes de retorno.

O "syn flooding" é um ataque do tipo "denial of service" porque o serviço atacado fica inacessível. Por exemplo, se o procedimento for executado para o serviço de "telnet", é pequena a chance de um usuário conseguir acessar via "telnet" a máquina alvo porque quase sempre a tabela de conexões pendentes está cheia.

O protocolo ICMP também apresenta as suas vulnerabilidades. Pacotes ICMP com a opção "redirect" (código 5) devem ser bloqueados pelo filtro porque ela pode ser utilizada para alterar uma rota válida. Caso não se queira que as máquinas da rede

interna sejam sondadas (via "ping", por exemplo) por usuários da Internet, deve-se bloquear também pacotes ICMP com a opção "request" (código 8).

2.8 Riscos na Filtragem

A filtragem por endereço fonte apresenta alguns riscos, os quais são os seguintes:

- "Source address": basicamente é o problema do "IP spoofing", só que a resposta ao ataque poderia ser, por exemplo, o envio de alguma informação (ex.: o arquivo "/etc/passwd") via correio eletrônico diretamente ao atacante. Ou seja, o atacante não precisa estar no caminho entre a vítima e a máquina cujo endereço foi forjado porque o retorno ocorre através de outros meios.
- "Man in the middle": além de forjar o endereço, nesse ataque o atacante deve estar no caminho entre a vítima e o host confiável (endereço forjado) porque ele tem de capturar os pacotes que são, na realidade, enviados ao host confiável (daí a denominação do ataque).

Muitos desses ataques só funcionam quando o host confiável (aquele cujo endereço é utilizado pelo atacante) não receber os pacotes de retorno, porque assim que ele receber algum pacote que não esteja relacionado com nenhuma conexão que ele tenha iniciado ele solicitará que a conexão forjada seja encerrada ("reset"). Existem várias formas de se evitar que o host confiável tome conhecimento da conexão forjada pelo atacante, eis alguns métodos:

- Confundindo o roteamento entre a máquina real (host confiável) e a máquina alvo;
- Utilizando um ataque onde somente a primeira resposta é requerida, de tal forma que o "reset" solicitado pela máquina real não importará;
- Inundando a máquina real com pacotes lixo (por exemplo, pacotes ICMP) enquanto o ataque ocorre, de forma que a máquina real ficará ocupada tentando processar os pacotes que ela recebe;
- Como foi citado anteriormente, utilizando "source routing".

A filtragem baseada na "port" fonte apresenta um problema semelhante àquele da filtragem pelo endereço fonte. Assume-se que a uma determinada "port" um determinado serviço esteja associado, mas nada impede que alguém com os devidos direitos (por exemplo, "root" no Unix) substitua o servidor por outro. Para evitar este tipo de ataque, deve-se garantir que o servidor seja confiável e execute somente o permitido; impedindo, de outra forma, que o cliente modificado posso solicitar alguma facilidade que comprometa o servidor. Portanto, é fundamental que tanto o servidor como também os clientes utilizados não sejam passiveis de serem alterados por pessoal não autorizado.

2.9 Filtragem de fragmentos de pacotes IP

Atacantes cada vez mais sofisticados tem começado a explorar os mais sutis aspectos do protocolo IP. A fragmentação de pacotes IP, uma característica importante

em interconexão de redes heterogêneas, apresenta vários problemas de segurança [RFC 1858].

Os filtros de pacotes são projetados com uma interface de usuário que esconde a fragmentação de pacotes do administrador. Conceitualmente, a filtragem é aplicada a cada pacote IP como uma entidade completa.

No datagrama IP todos os fragmentos exceto o último devem ter um tamanho da área de dados múltiplo de 8 bytes, a unidade elementar de fragmentação [TAN 96]. O campo de identificação do pacote IP ("identification") é utilizado para determinar a que datagrama um determinado fragmento pertence. Há também dois "bits" de controle de fragmentação: DF ("Don't fragment"), quando marcado (DF=1) significa que o datagrama jamais deve ser fragmentado; MF ("More fragment"), todos os fragmentos exceto o último tem esse bit marcado, isto é necessário para saber quando todos os fragmentos de um datagrama chegaram.

Uma proposta para a filtragem de fragmentos descrita por Mogul [MOG 89], envolve manter informações dos resultados da aplicação das regras de filtragem ao primeiro fragmento (com "offset" igual a 0) e aplicando eles aos fragmentos subsequentes do mesmo pacote. O módulo de filtragem deveria manter uma lista de pacotes indexados pelo endereço fonte, endereço destino, protocolo e identificador IP. Quando o fragmento inicial é visto ("offset" igual a zero: OFF = 0), caso o bit MF esteja marcado, uma entrada na lista é alocada para guardar o resultado da checagem de acesso ao filtro. Quando chegam pacotes com um "offset" diferente de zero, verifica-se na lista se há uma referência a fragmentos anteriores e o filtro aplica o resultado armazenado (passa ou bloqueia o pacote). Quando um fragmento com o bit MF zerado é visto, a entrada correspondente na lista é liberada.

Esse método apresenta algumas dificuldades. Fragmentos que chegam fora de ordem, possivelmente porque eles percorreram caminhos diferentes, violam uma das suposições do projeto, e fragmentos não desejados podem escapar.

Felizmente não é necessário remover todos os fragmentos de um pacote ofensivo. Como as informações "interessantes" para o filtro estão contidas nos cabeçalhos no início do pacote, a filtragem geralmente é aplicada somente ao primeiro fragmento. Os demais fragmentos do datagrama são passados sem serem filtrados porque é impossível para o destinatário completar a montagem do pacote se o primeiro pacote está faltando e, portanto, todo o pacote será descartado.

O protocolo IP permite fragmentação de pacotes em pedaços tão pequenos que se tornam impraticáveis porque o "overhead" é muito grande. Os atacantes podem às vezes explorar o comportamento típico do filtro e a habilidade de criar sequências de fragmentos peculiares que conseguem passar furtivamente pelo filtro. Na prática, tais fragmentos nunca são utilizados, assim é seguro bloqueá-los sem causar danos à operação normal do sistema.

Em muitas implementações do protocolo IP é possível impedir que se envie um fragmento muito pequeno. Se o tamanho do fragmento é tal que força que alguns

campos do cabeçalho TCP sejam enviados em um segundo fragmento, as regras de filtragem que precisam de tais campos não podem ser aplicadas. Caso o filtro não exija um tamanho mínimo para os fragmentos, um pacote que não deveria ser aceito poderia passar porque não há uma regra que satisfaz no filtro.

Qualquer módulo na Internet deve ser capaz de enviar um datagrama de 68 octetos sem precisar fragmentá-lo [RFC 1858]. Isto é porque um cabeçalho IP pode ter até 60 octetos e o tamanho mínimo de um fragmento é de 8 bytes. Entretanto, para o propósito de segurança, não é suficiente meramente garantir que um fragmento contenha no mínimo 8 octetos de dados além do cabeçalho IP porque informações importantes do cabeçalho de transporte (por exemplo: o campo CODE no cabeçalho TCP) poderiam estar além do oitavo octeto de dados.

Em um dos ataques utilizados, o primeiro fragmento contém somente oito octetos de dados (o tamanho mínimo). No caso do TCP, isto é suficiente para conter as ports fonte e destino, mas o restante do cabeçalho somente será enviado no segundo fragmento. Filtros que tentam barrar solicitações de conexão (SYN=1 e ACK=0) não conseguirão testar estes "flags" no primeiro octeto e tipicamente ignorarão eles nos fragmentos subsequentes.

Esse tipo de ataque pode ser evitado forçando certos limites no tamanho do fragmento; ou seja, que ele tenha um tamanho suficiente para conter todas as informações do cabeçalho necessárias. Há duas maneiras para garantir que o primeiro fragmento de um pacote inclua todos os campos necessários, as quais são:

- Método direto: envolve o cálculo do comprimento do cabeçalho do protocolo de transporte em cada fragmento com "offset" zero e a comparação com o tamanho mínimo estipulado. Caso o tamanho seja menor que o mínimo, o fragmento é descartado. Fragmentos com "offset" não zerado não precisam ser checados porque se o primeiro fragmento é descartado o destinatário não será capaz de completar a remontagem do pacote.
- Método indireto: este método confia na observação que quando um pacote TCP é fragmentado de forma a forçar que campos "interessantes" do cabeçalho não sejam enviados no primeiro fragmento, haverá um fragmento com "offset" igual a 1. Se um pacote com "offset" igual a 1 for visto, pelo contrário, ele poderia indicar a presença, no conjunto de fragmentos, de um fragmento com "offset" zero com o comprimento do cabeçalho de transporte igual a 8 octetos. Descartando esse fragmento bloqueará a remontagem do pacote no destinatário, sendo tão efetivo como no método direto.

Entretanto, há ainda uma outra possibilidade de ataque. Como não existe um padrão que especifique de que forma os fragmentos devem ser montados no seu destinatário, a maioria das implementações permitem que um fragmento mais recente substitua um fragmento mais antigo. Dessa forma, considerando que o filtro bloqueia estabelecimento de conexões baseado nas informações constantes nos "CODE bits" (ACK, SYN), um atacante poderia enviar um fragmento de pacote com o SYN (solicitação de conexão) não marcado e o filtro permitiria a passagem dele. Em seguida, o atacante montaria um fragmento de pacote com "offset" igual a 1 com os campos do cabeçalho do protocolo TCP devidamente alterados para compor um pacote de

solicitação de conexão. Considerando que este pacote não tem "offset" igual a zero ele passará livremente pelo filtro. No destinatário este fragmento irá substituir os devidos campos dentro do primeiro fragmento enviado garantindo ao atacante acesso a um serviço que deveria ser negado pelo filtro.

Uma solução genérica para todos esses ataques é a mesma apontada pelo método indireto no primeiro tipo de ataque; ou seja, bloquear todo fragmento que tenha "offset" igual a 1. Como não existe a necessidade normal de ocorrer um fragmento deste tipo, não haverá possibilidade de se causar qualquer problema ao andamento normal dos serviços providos na Internet.

3 Configuração do Filtro

A configuração do filtro é uma atividade que deve ser realizada cuidadosamente. Caso contrário, pacotes que deveriam ser barrados não o são e pacotes que poderiam passar pelo filtro não o fazem. Cada novo serviço disponível na rede requer que novas regras de filtragem sejam adicionadas.

Este capítulo apresenta as características de filtragem de pacotes dos principais serviços na Internet, quais suas implicações na segurança com respeito ao "firewall" e como fazê-los funcionar com um "firewall".

Para se elaborar as regras de filtragem é necessário que se saiba quais as informações relevantes que estão presentes em cada um dos pacotes de uma sessão. Para cada um dos serviços abordados a seguir é apresentada uma tabela com os pacotes do protocolo. Definida a política de segurança, basta escrever as regras de filtragem de acordo com a sintaxe do filtro e ajustar os endereços dos "hosts" envolvidos.

Para fins de exemplificação é utilizado o endereço "143.54.83.0" como endereço da sub-rede interna nos exemplos de regras de filtragem apresentados em alguns dos itens a seguir. Para representar qualquer endereço IP é utilizada a palavra "any" (qualquer) e a palavra "ACK" significa que o "flag" correspondente está marcado.

3.1 SMTP (serviço de correio eletrônico)

O SMTP é um serviço baseado no TCP, o servidor atende na "port" 25 e os clientes utilizam valores de "port" acima de 1023. As características dos pacotes do serviço smtp são as seguintes:

TABELA 3.1 - Pacotes no protocolo SMTP

Direção	End. Fonte	End. Destino	Protocolo	Port fonte	Port dest.	flag ACK	Observação
Entrando	Externo	Interno	TCP	> 1023	25	*	Mail chegando (sender to recipient)
Saindo	Interno	Externo	TCP	25	> 1023	Sim	Mail chegando (recipient to sender)
Saindo	Interno	Externo	TCP	> 1023	25	*	Mail saindo (sender to recipient)
Entrando	Externo	Interno	TCP	25	> 1023	Sim	Mail saindo (recipient to sender)

* ACK não está marcado no primeiro pacote deste tipo (estabelecimento de conexão) mas estará marcado nos demais.

Recomendações na configuração do "firewall" para o SMTP [CHA 95]:

- Enviar todo o "mail" que sai e que entra na rede interna através de um "bastion host";

- Configurar o filtro de pacotes para restringir conexões externas somente com o "bastion host";
- Configurar o filtro de pacotes para restringir conexões do "bastion host" para um servidor interno específico;
- Permitir qualquer máquina da rede interna enviar "mail" para o "bastion host";
- Utilizar um servidor de correio eletrônico tão robusto quanto possível;
- Estar atualizado com "patches" para os agentes de entrega e de usuário;
- Educar os usuários quanto a ataques praticados via "mail", como por exemplo aqueles que contém instruções para executar um determinado programa ou alterar a senha do próprio usuário para um determinado valor.

As possíveis regras de filtragem seriam as seguintes (considerando que o endereço do servidor de correio eletrônico é 143.54.83.2):

1. Allow tcp from any >1023 to 143.54.83.2 25
2. Allow tcp from 143.54.83.2 25 to any > 1023 ACK
3. Allow tcp from 143.54.83.2 > 1023 to any 25
4. Allow tcp from any 25 to 143.54.83.2 > 1023 ACK

3.2 POP (Post Office Protocol)

Este é um serviço baseado no protocolo TCP, o servidor utiliza as "ports" 110 (POP versão 3) e 109 (versão 2), e os clientes utilizam valores acima de 1023. As características dos pacotes do serviço Pop são as seguintes:

TABELA 3.2 - Pacotes no protocolo POP

Direção	End. Fonte	End. Destino	Protocolo	Port fonte	Port dest.	flag ACK	Observação
Entrando	Externo	Interno	TCP	> 1023	110 ou 109	*	Conexão com servidor POP na rede interna
Saindo	Interno	Externo	TCP	110 ou 109	> 1023	Sim	Servidor POP interno para cliente
Saindo	Interno	Externo	TCP	> 1023	110 ou 109	*	Cliente interno acessando servidor POP externo
Entrando	Externo	Interno	TCP	110ou 109	> 1023	Sim	Servidor POP externo para cliente

* ACK não está marcado no primeiro pacote deste tipo (estabelecimento de conexão) mas estará marcado nos demais.

Recomendações na configuração do "firewall" para o POP [CHA 95]:

- Não permitir que os usuários da rede interna transfiram mail (do "site" para a Internet) via POP, ao menos que isso possa ser feito sem revelar as senhas reutilizáveis, o conteúdo do "mail" não seja confidencial ou então o canal sendo utilizado seja criptografado;
- Utilizar um "proxy server" para o POP;

- Se for necessário transferir "mail" de algum outro local na Internet para a rede interna, restringir via filtragem de pacotes de e para somente os "hosts" envolvidos.

As regras de filtragem para o caso hipotético de um servidor com endereço 143.54.83.3 são as seguintes:

1. Allow tcp from any >1023 to 143.54.83.3 109,110
2. Allow tcp from 143.54.83.3 109,110 to any > 1023 ACK
3. Allow tcp from 143.54.83.3 > 1023 to any 109,110
4. Allow tcp from any 109,110 to 143.54.83.3 > 1023 ACK

3.3 FTP (File Transfer Protocol)

O protocolo FTP abrange duas conexõs, uma de comandos e outra de dados. O servidor do canal de comandos atende na "port" 21 e o de dados na "port" 20. No modo normal de uma conexão FTP é o servidor do canal de dados quem inicia a conexão com o cliente para a transferência de dados(Fig. 3.1). Os clientes FTP utilizam valores de "port" acima de 1023. As características dos pacotes do serviço ftp são apresentadas na Tabela 3.3.

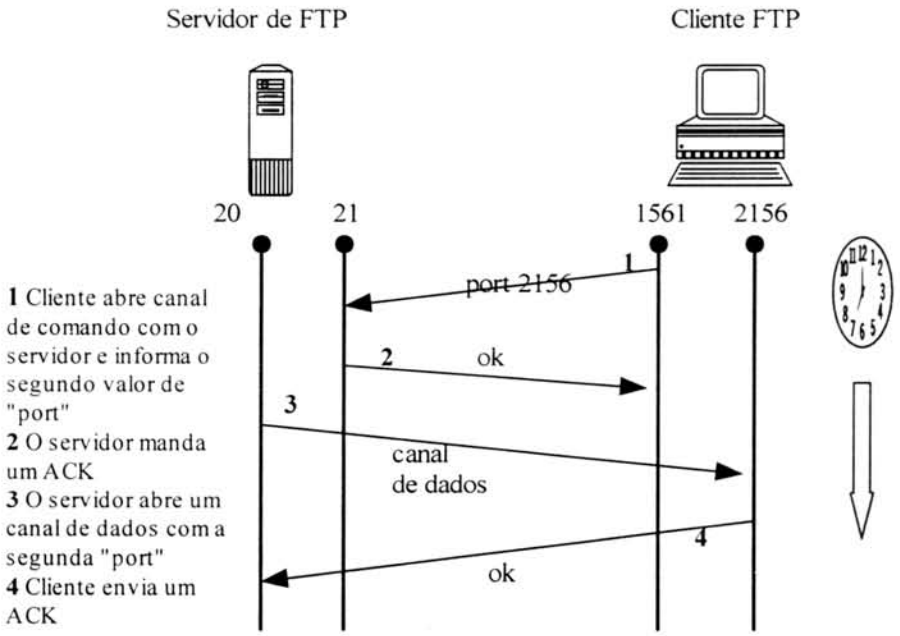


FIGURA 3.1 - Uma conexão FTP no modo normal

O FTP também permite a conexão no modo passivo ("PASV mode") , sendo que nesta situação é o cliente que inicia a conexão com o servidor do canal de dados. Nesse modo, o servidor de comandos comunica em qual "port" (valores acima de 1023, e não mais a "port" 20) o servidor do canal de dados aceitará conexão; ou seja, nesse modo de conexão o servidor é passivo.

TABELA 3.3 - Pacotes no protocolo FTP

Direção	End. Fonte	End. Destino	Protocolo	Port fonte	Port dest.	flag ACK	Observação
Entrando	Externo	Interno	TCP	> 1023	21	*	Conexão com serv. interno
Saindo	Interno	Externo	TCP	21	> 1023	Sim	Serv. int. para cliente externo
Saindo	Interno	Externo	TCP	20	> 1023	*	Serv. canal dados para cliente ext.
Entrando	Externo	Interno	TCP	> 1023	20	Sim	Cliente ext. para serv. interno
Entrando	Externo	Interno	TCP	> 1023	> 1023	*	Serv. int. (PASV)
Saindo	Interno	Externo	TCP	>1023	> 1023	Sim	Serv. int (PASV) para cliente ext.
Saindo	Interno	Externo	TCP	> 1023	21	*	Serv. externo
Entrando	Externo	Interno	TCP	21	> 1023	Sim	Serv. ext. para cliente interno
Entrando	Externo	Interno	TCP	20	> 1023	*	Serv. ext. dados
Saindo	Interno	Externo	TCP	>1023	20	Sim	Cliente int. para serv. ext
Saindo	Interno	Externo	TCP	> 1023	>1023	*	Serv. externo modo PASV
Entrando	Externo	Interno	TCP	> 1023	> 1023	Sim	Serv. ext. para cliente interno

* ACK não está marcado no primeiro pacote deste tipo (estabelecimento de conexão) mas estará marcado nos demais.

Recomendações na configuração do "firewall" para o serviço de FTP [CHA 95]:

- Se os clientes internos permitem conexão no modo passivo (PASV), configurar o filtro de pacotes de forma que apenas conexões de dentro para fora (outbound) sejam permitidas de e para valores de "ports" acima de 1023.
- Se os clientes internos não permitem o modo PASV, é recomendável utilizar um servidor "proxy" para o ftp.
- É também possível permitir conexões no modo PASV (o filtro deve estar apropriadamente configurado, veja tabela acima) e via um servidor "proxy"; dessa forma se atende aos dois tipos de clientes.
- Se houver um servidor ftp na rede interna, este deve estar no "bastion host" e o filtro deve ser configurado para aceitar conexões para apenas esta máquina;
- Utilizar sempre um servidor de ftp atualizado;
- Não é recomendável que os usuários acessem de fora máquinas internas via ftp não anônimo (anonymous) porque a senha passa de forma clara (ou seja, basta que alguem capture o tráfego da sessão para obter a senha do usuário).

Considerando a utilização de um "proxy server" para o serviço de ftp com endereço hipotético 143.54.83.4 ter-se-ia as seguintes regras de filtragem:

1. Allow tcp from any > 1023 to 143.54.83.4 21
2. Allow tcp from 143.54.83.4 21 to any > 1023 ACK
3. Allow tcp from 143.54.83.4 20 to any > 1023
4. Allow tcp from any > 1023 to 143.54.83.4 20 ACK
5. Allow tcp from any > 1023 to 143.54.83.4 > 1023
6. Allow tcp from 143.54.83.4 > 1023 to any > 1023 ACK

7. Allow tcp from 143.54.83.4 > 1023 to any 21
8. Allow tcp from any 21 to 143.54.83.4 > 1023 ACK
9. Allow tcp from any 20 to 143.54.83.4 > 1023
10. Allow tcp from 143.54.83.4 >1023 to any 20 ACK
11. Allow tcp from 143.54.83.4 > 1023 to any > 1023
12. Allow tcp from any > 1023 to 143.54.83.4 > 1023 ACK

3.4 Telnet

O Telnet é um serviço baseado no TCP, os servidores utilizam a "port" 23 e os clientes utilizam valores acima de 1023. As características dos pacotes do serviço Telnet são apresentadas abaixo (Tab. 3.4).

TABELA 3.4 - Pacotes no protocolo Telnet

Direção	End. Fonte	End. Destino	Protocolo	Port fonte	Port dest.	flag ACK	Observação
Entrando	Externo	Interno	TCP	> 1023	23	*	Cliente ext. para servidor interno
Saindo	Interno	Externo	TCP	23	> 1023	Sim	Serv. interno para cliente externo
Saindo	Interno	Externo	TCP	> 1023	23	*	Cliente interno para serv. externo
Entrando	Externo	Interno	TCP	23	> 1023	Sim	Servidor externo para cliente int.

* ACK não está marcado no primeiro pacote deste tipo (estabelecimento de conexão) mas estará marcado nos demais.

Recomendações na configuração do "firewall" para o Telnet [CHA 95]:

- Não permitir que os usuários acessem de fora a rede interna via telnet (a senha passa de forma clara), a não ser que se utilize algum mecanismo de autenticação mais robusto com senhas não reutilizáveis ("one time passwords");
- Utilizar um servidor "proxy" no "bastion host";
- Configurar o filtro para permitir apenas conexões telnet "outgoing" via "bastion host";
- É recomendável utilizar uma versão criptografada de telnet (dessa forma nada passa de forma clara).

Tendo-se um servidor "proxy" para o telnet na máquina cujo endereço IP é 143.54.83.5, as regras de filtragem seriam as seguintes:

1. Allow tcp from any >1023 to 143.54.83.5 23
2. Allow tcp from 143.54.83.5 23 to any > 1023 ACK
3. Allow tcp from 143.54.83.5 > 1023 to any 23
4. Allow tcp from any 23 to 143.54.83.5 > 1023 ACK

3.5 NNTP (Network News Transfer Protocol)

O serviço NNTP é baseado no TCP, os servidores utilizam a "port" 119 e os clientes (incluindo servidores transferindo "news" para outros servidores) utilizam valores acima de 1023. As características dos pacotes do serviço NNTP são apresentadas abaixo (Tab. 3.5):

TABELA 3.5 - Pacotes no protocolo NNTP

Direção	End. Fonte	End. Destino	Protocolo	Port fonte	Port dest.	flag ACK	Observação
Entrando	Externo	Interno	TCP	> 1023	119	*	News chegando
Saindo	Interno	Externo	TCP	119	>1023	Sim	Resposta a news que está chegando
Saindo	Interno	Externo	TCP	> 1023	119	*	Envio de news
Entrando	Externo	Interno	TCP	119	> 1023	Sim	Resposta a news enviado
**	Interno	Serv. de news	TCP	> 1023	119	*	Cliente int. lendo news no serv. int.
**	Serv. de news	Interno	TCP	119	> 1023	Sim	Serv. int. enviando artigos para cliente interno

* ACK não está marcado no primeiro pacote deste tipo (estabelecimento de conexão) mas estará marcado nos demais.
** Ambas máquinas envolvidas são internas, geralmente.

Recomendações na configuração do "firewall" para o News [CHA 95]:

- Não utilizar um "bastion host" como um servidor de "news";
- Não permitir criação automática de grupos;
- Configurar o filtro de pacotes para aceitar conexão somente entre o servidor de news interno e os servidores externos considerados confiáveis e vice versa.

As regras de filtragem correspondentes a troca de news do servidor interno (endereço IP 143.54.83.6) com servidores news na Internet são:

1. Allow tcp from any >1023 to 143.54.83.6 119
2. Allow tcp from 143.54.83.6 119 to any > 1023 ACK
3. Allow tcp from 143.54.83.6 > 1023 to any 119
4. Allow tcp from any 119 to 143.54.83.6 > 1023 ACK

3.6 HTTP (Hypertext transfer protocol)

Este é um serviço baseado no TCP, a maioria dos servidores utilizam a "port" 80 e os clientes utilizam valores acima de 1023. . As características dos pacotes do serviço Http são as seguintes:

TABELA 3.6 - Pacotes no protocolo HTTP

Direção	End. Fonte	End. Destino	Protocolo	Port fonte	Port dest.	flag ACK	Observação
Entrando	Externo	Interno	TCP	> 1023	80	*	Cliente p/ serv. int.
Saindo	Interno	Externo	TCP	80	>1023	Sim	Serv. int. p/ cliente
Saindo	Interno	Externo	TCP	> 1023	80	*	Cliente int. p/ sev. exteno
Entrando	Externo	Interno	TCP	80	> 1023	Sim	Serv. externo para cliente interno

* ACK não está marcado no primeiro pacote deste tipo (estabelecimento de conexão) mas estará marcado nos demais.

Recomendações na configuração do "firewall" para o http [CHA 95]:

- Se houver um servidor http interno, utilizar um "bastion host" dedicado para ele;
- Configurar cuidadosamente o servidor para controlar ao que ele tem acesso;
- Configurar o filtro para permitir apenas conexões externas com a "port" padrão (80);
- Utilizar um servidor "proxy" (muitos servidores procuradores apresentam a facilidade de uma "cache", o que aumenta o desempenho do servidor);
- Configurar os "browsers" apropriadamente e alertar os usuários para não reconfigurá-los baseados em conselho externo.

Tendo como hospedeiro do "proxy server" http a máquina 143.54.83.7, as regras de filtragem são as seguintes:

1. Allow tcp from any >1023 to 143.54.83.7 80
2. Allow tcp from 143.54.83.7 80 to any > 1023 ACK
3. Allow tcp from 143.54.83.7 > 1023 to any 80
4. Allow tcp from any 80 to 143.54.83.7 > 1023 ACK

3.7 Finger

O finger é um serviço baseado no TCP, os servidores atendem na "port" 79 e os clientes utilizam valores acima de 1023. As características dos pacotes deste serviço são as seguintes:

TABELA 3.7 - Pacotes no protocolo finger

Direção	End. Fonte	End. Destino	Protocolo	Port fonte	Port dest.	flag ACK	Observação
Entrando	Externo	Interno	TCP	> 1023	79	*	Cliente p/ serv. int.
Saindo	Interno	Externo	TCP	80	>1023	Sim	Serv. int. p/ cliente
Saindo	Interno	Externo	TCP	> 1023	79	*	Cliente int. p/ sev. exteno
Entrando	Externo	Interno	TCP	80	> 1023	Sim	Serv. externo para cliente interno

* ACK não está marcado no primeiro pacote deste tipo (estabelecimento de conexão) mas estará marcado nos demais.

Recomendações na configuração do "firewall" para o finger [CHA 95]:

- Limitar solicitações de fora para dentro somente para o "bastion host";
- Permitir conexões de dentro para fora da rede interna.

Considerando que se adote a política de somente permitir conexões finger da rede interna para a Internet as regras de filtragem seriam as seguintes:

1. Allow tcp from 143.54.83.0 > 1023 to any 79
2. Allow tcp from any 79 to 143.54.83.0 > 1023

3.8 DNS (Domain Name System)

O serviço DNS utiliza os protocolos UDP ou TCP. Em ambos os casos, o servidor utiliza a "port" 53 e os clientes utilizam valores acima de 1023. Entretanto, quando um servidor está conectado a outro servidor, ambos utilizam a "port" 53 quando o protocolo for UDP e valores 1023 para o servidor-cliente quando o protocolo for TCP. As características dos pacotes desse serviço foram apresentadas em um exemplo anterior (Tabela 2.1).

Recomendações na configuração do "firewall" para o DNS [CHA 95]:

- Colocar um servidor DNS em um "bastion host" para acesso dos usuários na Internet;
- Não permitir que os registros HINFO (que fornecem informações do hardware e software de um determinado "host") sejam visíveis do mundo externo;
- Utilizar uma versão atual do BIND e utilizar verificação reversa para evitar "spoofing";
- Configurar o filtro para permitir "zone transfers" apenas com "hosts" confiáveis.

3.9 RIP (Routing Information Protocol)

O RIP é um serviço baseado no protocolo UDP, os servidores atendem na "port" 520 para difusão de outros servidores e solicitações de clientes. Os clientes geralmente utilizam valores acima de 1023. As características dos pacotes deste serviço são apresentadas na TABELA 3.8.

Não se deve permitir protocolos de roteamento através do "firewall" de ou para a Internet porque o roteamento em um "firewall" geralmente é muito simples e funciona bem com rotas estáticas. Simplesmente se pode configurar o "firewall" para direcionar os pacotes que são para alguma máquina interna para um roteador interno e direcionar todos os outros pacotes para a conexão com a Internet.

TABELA 3.8 - Pacotes no protocolo RIP

Direção	End. Fonte	End. Destino	Protocolo	Port fonte	Port dest.	flag ACK	Observação
Entrando	Externo	Interno	UDP	> 1023	520	*	Cliente p/ serv. int.
Saindo	Interno	Externo	UDP	520	>1023	*	Serv. int. p/ cliente
Saindo	Interno	Externo	UDP	> 1023	520	*	Cliente int. p/ sev. exteno
Entrando	Externo	Interno	UDP	520	> 1023	*	Serv. externo para cliente interno
Entrando	Externo	Interno	UDP	520	520	*	"Broadcast" p/ servidor int.
Saindo	Interno	Externo	UDP	520	520	*	Serv. interno para servidor externo

* Pacotes UDP não tem o flag ACK

3.10 Outros serviços

Existem uma série de outros serviços na Internet, e muitos deles não apresentam o mínimo de segurança para permitir acesso através do "firewall". Alguns serviços que se inserem neste grupo são os seguintes:

- "talk": baseado no protocolo UDP e TCP; "ports" 517 e 518 no servidor da parte UDP do protocolo e clientes com valores acima de 1023, e "ports" acima de 1023 tanto para o cliente e servidor na parte TCP do protocolo. Esse é um protocolo inseguro justamente porque deveria se permitir conexão com qualquer "port" acima de 1023 nos "hosts" da rede interna.
- "syslog": serviço baseado no protocolo UDP, os servidores utilizam a "port" 514 e os clientes utilizam valores acima de 1023. Quando os servidores estão passando mensagens entre si ambos utilizam a "port" 514. Este serviço é inseguro porque permite ataque de enchurrada ("flooding") contra o servidor.
- SNMP (Simple network management protocol): esse serviço não é seguro através do "firewall" porque permitiria ao atacante gerenciar e/ou obter informações sobre recursos da rede interna;
- NFS (Network Filte System): não é seguro porque a autenticação é baseada apenas no endereço IP fonte da máquina que solicita o serviço, como foi visto anteriormente é fácil forjar o endereço fonte ("ip spoofing");
- NIS/YP (Yellow Pages): não é seguro porque permite que se obtenha uma série de informações úteis para explorar vulnerabilidades no sistema (ex.: arquivo de senhas).

4 Desenvolvimento do Filtro

Este capítulo aborda aspectos do "session filter" e apresenta os pontos mais relevantes no projeto e implementação do filtro de sessões proposto. Essa implementação é o produto de uma coleta de informações sobre filtragem de pacotes, resumidamente apresentadas nos capítulos anteriores, e a escolha de uma plataforma viável ao desenvolvimento do filtro.

4.1 "Session Filter"

Na filtragem convencional cada pacote que transita pelo filtro deve ser verificado junto às regras de filtragem. A filtragem por sessão permite que se valide um pacote baseado na sessão a qual ele pertence. Dessa forma, apenas o pacote de inicialização da conexão precisa ser verificado com as regras de filtragem, os pacotes seguintes são verificados junto a uma "cache" de sessões ativas [AMO 96]: caso exista uma sessão correspondente o pacote é liberado; caso contrário, o pacote é descartado (Fig. 4.1).

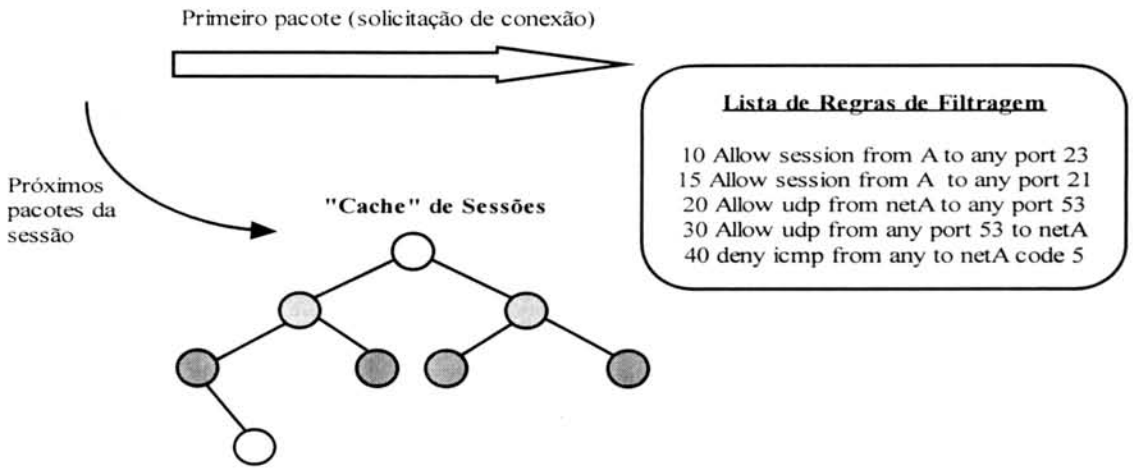


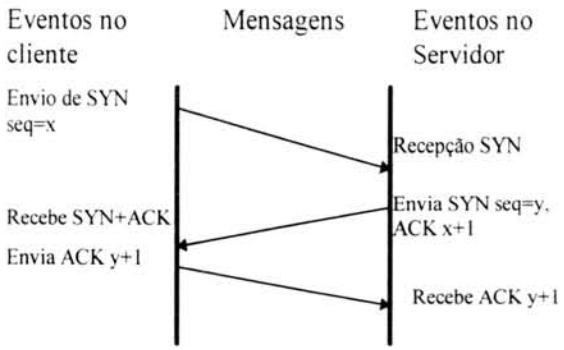
FIGURA 4.1 - Estrutura básica de um "Session Filter"

4.1.1 Controle das Sessões

Para que se mantenha controle sobre as sessões ativas é necessário que o protocolo seja orientado a conexão; ou seja, deve ser possível determinar exatamente quando uma nova sessão começa e termina.

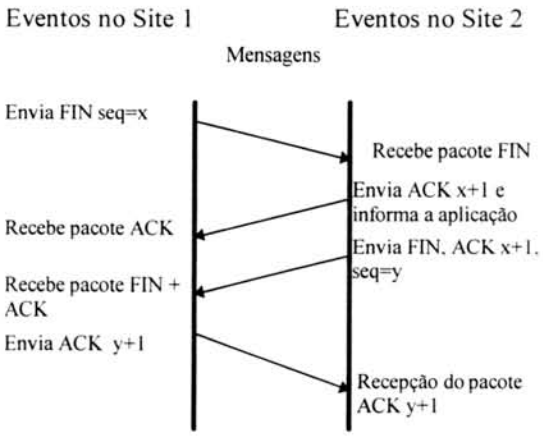
No ambiente TCP/IP o protocolo TCP é orientado a conexão; portanto, existe um procedimento de estabelecimento de conexão. No TCP esse procedimento é conhecido por "three-way handshake" [COM 91](FIGURA 4.2). O pacote de solicitação de conexão apresenta apenas o "bit" SYN marcado. O servidor, após receber a solicitação, responde com um pacote que apresenta os "bits" SYN e ACK setados. O cliente, ao receber esta confirmação do servidor, envia um pacote de reconhecimento que apresenta apenas o ACK marcado. Está então estabelecida a conexão. Os demais pacotes, até o término da sessão, apresentam o ACK marcado.

O procedimento de encerramento de uma conexão TCP é o seguinte (Fig. 4.3): o "site" que deseja encerrar a conexão envia um pacote com o "bit" FIN marcado, o outro "site" após receber este pacote envia a confirmação (ACK marcado) de solicitação de encerramento de conexão e, após um certo atraso, envia também um pacote com o FIN marcado e fica aguardando a confirmação do outro "site", quando este confirma e o outro recebe é encerrada a conexão.



(tempo avança no sentido de cima para baixo, linhas diagonais representam pacotes enviados entre o cliente e o servidor)

FIGURA 4.2 - Sequência de mensagens no procedimento de estabelecimento de conexão no TCP



(O Site que recebe o primeiro FIN confirma ele imediatamente e então espera um tempo antes de enviar o segundo pacote FIN)

FIGURA 4.3 - Procedimento de encerramento de uma conexão TCP

4.1.2 Identificação

Uma conexão TCP é identificada pelos pares "endereço IP fonte e destino" e "port fonte e destino" (Fig. 4.4). Esta identificação é única em toda a rede; ou seja, não há outra conexão entre os dois hosts envolvidos com o mesmo par de "port fonte e destino".

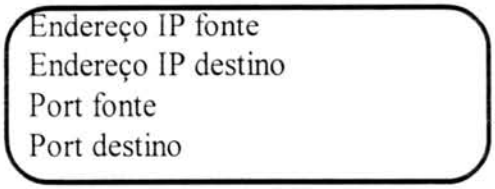
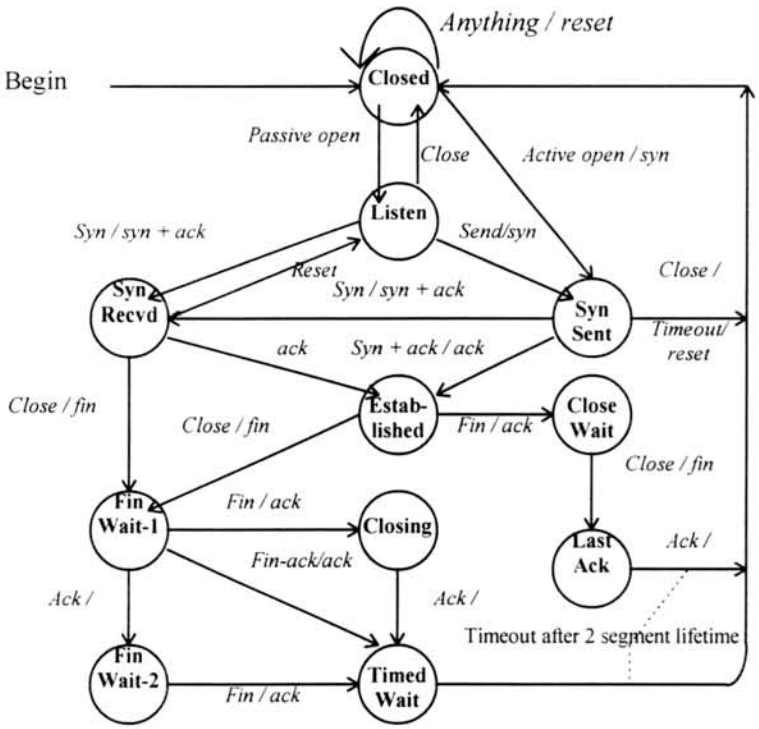


FIGURA 4.4 - Identificação de uma Sessão TCP

Além dessas informações, a "cache" deve também manter o estado da conexão dos dois "hosts" envolvidos a fim de que a sessão seja monitorada corretamente. A máquina de estados finitos do protocolo TCP [COM 91] é empregada para controlar o estado da conexão (Fig. 4.5). Para eliminar conexões terminadas de forma anormal, deve-se adotar um "timeout" para que estas conexões sejam eliminadas da "cache".



(Cada ponto final começa no estado *closed*. Rótulos nas transições mostram a entrada (*input*) que causou a transição seguido pela saída (*output*) se alguma.)

FIGURA 4.5 - A máquina de estados finitos do protocolo TCP

4.1.3 Desempenho

Uma das vantagens do filtro de sessões é o ganho em desempenho em relação a um filtro convencional. Pode-se afirmar isso em se considerando o fato de cada pacote ser analisado frente as regras de filtragem em um filtro convencional. Portanto, em média espera-se que o "overhead" acarretado pelo filtro de sessão seja consideravelmente menor. O ganho em desempenho depende do número de regras de filtragem e da quantidade de sessões presentes na "cache".

Num filtro convencional é necessário pelo menos duas regras de filtragem para cada serviço baseado no TCP (uma regra para o fluxo de pacotes em cada sentido), enquanto que é necessária apenas uma regra de filtragem para o filtro de sessão (Fig. 4.6). Portanto, também se diminui o tempo de pesquisa pelas regras de filtragem já que o número destas é reduzido.

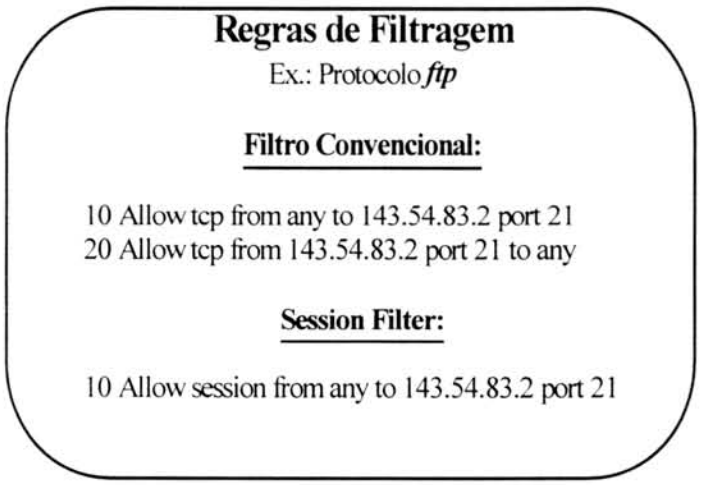


FIGURA 4.6 - Regras de filtragem (convencional versus session filter)

4.1.4 Monitoramento

Outra vantagem do filtro de sessões é a possibilidade de monitoramento das sessões ativas. Ou seja, todas as sessões ativas entre a rede interna e a rede externa estão registradas na "cache" de sessões.

Pode-se, dessa forma, obter estatísticas mais apuradas sobre as sessões que trafegam pelo "gateway": quantidade de pacotes, bytes, tempo médio de filtragem de cada pacote, "hosts" envolvidos, "ports" envolvidas, "timestamp" do último pacote.

4.2 Projeto e implementação do Filtro

O filtro foi desenvolvido na plataforma FreeBSD (Unix para a arquitetura Intel x86, compatível com o Unix BSD) [LEH 96]. Essa plataforma foi escolhida porque se trata de um sistema operacional "freeware" e todo o código fonte do sistema operacional está disponível.

O ponto inicial de partida foi o filtro de pacotes disponível no FreeBSD, o "ip_fw". Ele permetiu que se estudasse como o filtro se encaixa no "kernel" do sistema operacional. Para inserir, remover, listar, e realizar outras tarefas, é utilizado o utilitário "ipfw" também disponível como "freeware". Este utilitário foi adaptado ao filtro de sessão desenvolvido, desta forma se mantém compatibilidade com a estrutura das regras de filtragem já disponíveis sendo necessária apenas uma extensão à sintaxe das regras.

Outro fator importante na utilização do FreeBSD é a facilidade de se personalizar o "Kernel", bem como a sua compilação [LEH 96] . Isto é importante visto que, como o filtro é um acréscimo ao núcleo do sistema operacional, diversas compilações do "kernel" são necessárias durante o processo de implementação para a realização de testes.

Nos tópicos seguintes são apresentados os principais aspectos do projeto e implementação do filtro de pacotes desenvolvido. São descritos tanto os aspectos do componente de filtragem convencional (filtro convencional) como também o filtro de sessões.

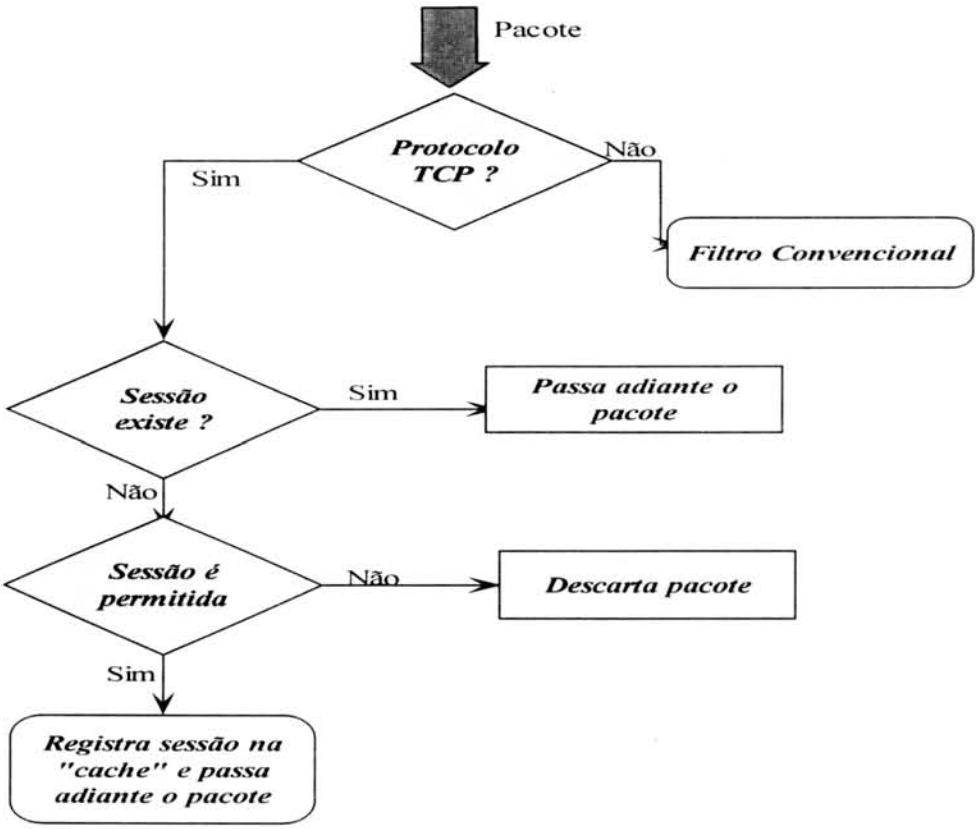


FIGURA 4.7 - Fluxo de pacotes TCP

4.2.1 Fluxo de Pacotes

A filtragem por sessão é realizada apenas para o protocolo TCP. Para os demais protocolos (icmp, ip, udp) a filtragem é convencional; ou seja, as regras de filtragem são aplicadas na ordem especificada. Todo pacote cujo protocolo é TCP é direcionado para o "Session Filter", os demais fluem pelo filtro convencional (Fig. 4.7).

4.2.2 Filtro convencional

As regras de filtragem são armazenadas numa lista duplamente encadeada respeitando a ordem que foi estabelecida. Cada regra tem um número. A estrutura de dados adotada para conter uma regra de filtragem é apresentada na figura abaixo (Fig. 4.8).

```
struct rule {
   u_short rule_number;
   u_long packets, bytes;       /* packets and byte counters */
   struct in_addr ip_src, ip_dst;  /* source and destination IP address */
   struct in_addr ip_smsk, ip_dmsk;  /* mask for source and destination address */
   union {
      struct in_addr ip_format;
      struct {
         char name[LENG_INT_NAME];
         short unit_format;
      } interface;
   } through;                        /* through a specified interface */
   u_char ip_options;                /* IP options */
   u_short rule_flags;               /* rule flags */
   struct port s_port, d_port;       /* array of source and destination ports  */
   unsigned icmptypes[ICMPTYPES_DIM];
   long timestamp;
};
```

FIGURA 4.8 - Estrutura de uma regra de filtragem

A estrutura acima apresentada contém os seguintes dados:

- Número da regra de filtragem: rule_number;
- Número de bytes e de pacotes que cumpriram a regra: bytes, packets;
- Endereço IP fonte e destino: ip_src, ip_dst;
- Máscara para o endereço fonte e destino: ip_smsk, ip_dmsk;
- Especificação da interface: through, pode ser o nome da interface (interface) ou o número ip correspondente a interface (ip_format);
- Opções do protocolo IP: ip_options;
- Flags da regra: rule_flags;
- "Ports" fonte e destino: s_port, d_port;
- Tipos ICMP: icmptypes;
- Timestamp: timestamp;

A máscara dos endereços IP fonte e destino permite que se aplique uma regra a uma rede (por exemplo: endereço IP 143.54.83.0 e máscara 255.255.255.0 significa que a regra deve ser aplicada a qualquer "host" da sub-rede especificada; ou seja, 143.54.83).

A interface para a qual a regra se aplica também pode ser especificada utilizando o nome simbólico (ex.: ed0) ou o endereço IP correspondente a interface.

O campo de opções do protocolo IP (Fig. 4.9) permite que se especifique quais as opções a serem consideradas. Uma opção que deveria ser negada é o "IP source routing" que representa uma grave ameaça a segurança, pois permite ao atacante descrever no próprio pacote qual a rota que o pacote deve seguir, permitindo desta forma que o atacante utilize um endereço fonte qualquer (geralmente o de uma máquina considerada confiável pela vítima) e garantindo que a resposta será enviada para a rota

especificada no pacote. As principais opções são: "Loose source routing" (utilizado para rotear um datagrama ao longo de um caminho especificado, "Strict source routing" (idem a opção anterior), "Record packet route" (utilizada para traçar uma rota) e "Internet timestamp" (utilizada para registrar os timestamps ao longo de uma rota).

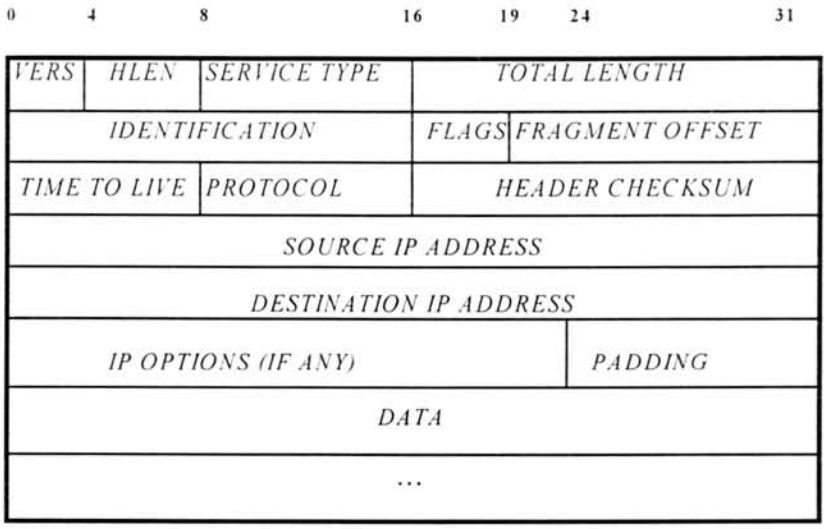


FIGURA 4.9 - Formato de um datagrama IP

Os "flags" da regra contém informações acerca de quais ações devem ser tomadas pela regra. Os "flags" adotados são os seguintes:

Flag	**valor** (em hexadecimal)	**descrição**
MATCH_ALL	0000	Aplicar a qualquer protocolo
SESSION_RULE	0001	Regra de Sessão
MATCH_UDP	0002	Protocolo UDP
MATCH_ICMP	0003	Protocolo ICMP
P_IN	0004	Pacote "inbound" (que chega)
P_OUT	0008	Pacote "outbound" (que sai)
ACCEPT_RULE	0010	Significa que esta é uma regra de aceitação (se não estiver marcado significa que é uma regra de rejeição)
USE_INTERF	0020	Utilizar a interface especificada ("through")
IP_FRAGMENT	0040	Fragmentos de pacotes IP
REPLY_ICMP	0080	Caso o pacote seja barrado enviar pacote ICMP "unreachable"
LOG	0200	Registrar no "log"
ICMP_IS_VALID	1000	O "bitmap" *icmptypes* é válido
MATCH_ALL_INTERF	2000	Qualquer interface

FIGURA 4.10 - Flags das regras de filtragem

Pode-se especificar até quatro "ports" tanto para a fonte como para o destino ou então até dois intervalos (ex: 1023-65535, significando valores de 1023 a 65535).

4.2.3 Postura

Foi adotada a postura prudente; ou seja, "tudo aquilo que não é expressamente permitido é proibido". Em outras palavras: "se não há uma regra de filtragem que permita a passagem do pacote este é descartado ('dropped')". O filtro "freeware" *ip_fw* garante isso colocando como última regra de filtragem (regra número 65535) uma regra que bloqueia todo e qualquer pacote. Na implementação aqui apresentada não se adotou esta estratégia; ao contrário, a postura "default" do filtro é bloquear o pacote caso não seja encontrada uma regra que o permita.

Em consideração ao tratamento de pacotes IP fragmentados foi adotada a recomendação apresentada no capítulo 2. Ou seja, fragmentos com "offset" igual a 1 são descartados.

4.2.4 Filtragem de Sessões

O filtro de sessão requer um grau de detalhamento maior. Em suma existe um registro para cada uma das sessões que trafegam através do filtro. Um pacote TCP que não pertença a uma sessão ativa ou cujo estabelecimento de conexão não seja autorizado é descartado e não é enviado adiante pelo filtro.

Para justificar o melhor desempenho desse esquema de filtragem, o acesso à "cache" de sessões deve ser mais rápido que, em média, o acesso ao filtro convencional para cada pacote da sessão. Portanto, a "cache" deve ter uma estrutura que permita a inserção, consulta e remoção de nodos com um número reduzido de operações. Para tanto, é utilizado uma estrutura em "Árvore AVL" (Fig. 4.11) que apresenta uma complexidade no tempo de ordem *O(log n)* na execução das três operações citadas. Essa ordem de complexidade se deve sobretudo ao fato desse tipo de árvore ser quase que totalmente equilibrada; ou seja, por exemplo, com 1024 nodos ela tem uma profundidade de no máximo 10 nodos. Esse equilíbrio é garantido porque não pode existir nenhum nodo que tenha uma diferença de "altura" maior do que 1 (um) entre os seu ramos esquerdo e direito. Toda a vez que é feita uma inserção ou remoção de um nodo isso é verificado. Existem 6 casos de excessões na inserção e 10 casos na remoção para os quais é necessário fazer um rearranjo (rotações) da árvore para restabelecer o equilíbrio [TER 91, STA 95].

Para a manipulação dos nodos da árvore é utilizada como chave primária os pares de endereço IP e valores de "port". Para formar uma chave única os dois pares são agrupados da seguinte forma: o menor endereço IP é a parte menos significativa do identificador relativo ao endereço e o outro endereço forma a parte mais significativa; da mesma forma para o par de "ports" (Fig. 4.12). Seguindo essa regra de formação, obtém-se um valor único.

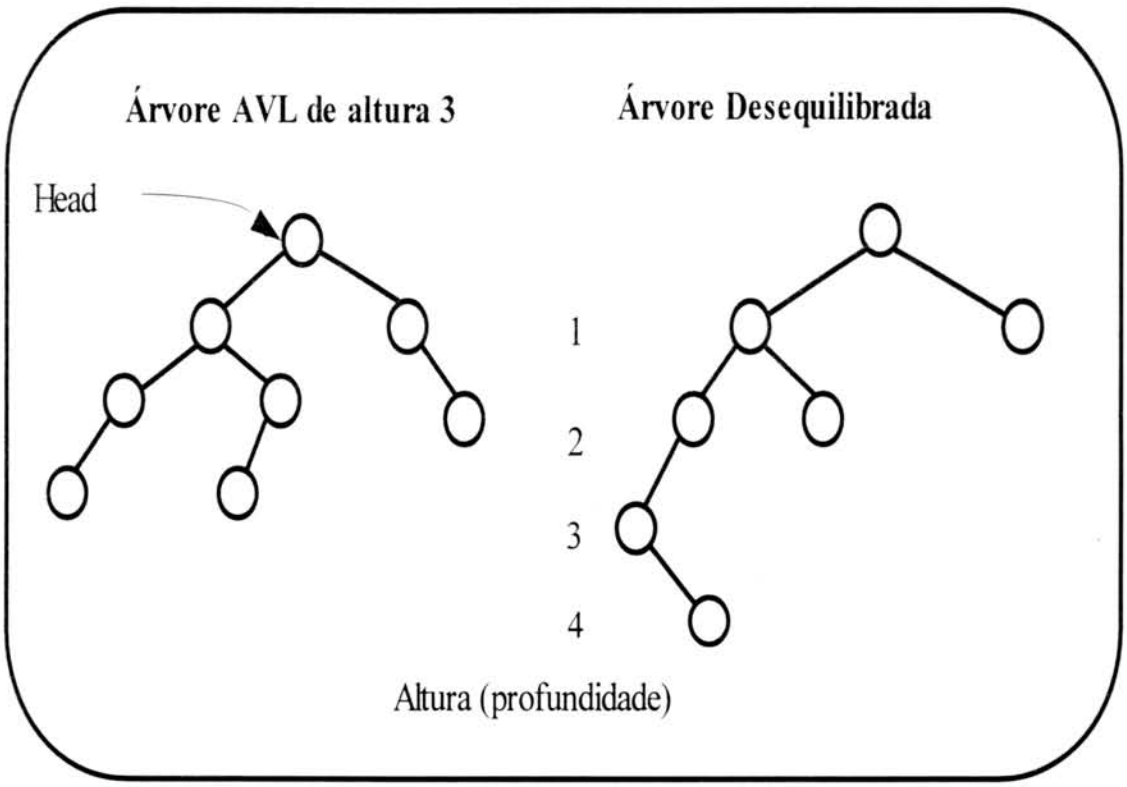


FIGURA 4.11 - Exemplo de "Árvore AVL"

Valores obtidos do pacote:

Endereço IP fonte: 143.54.83.11 (este é o menor IP)
Endereço IP destino: **200.23.45.7**
Port fonte: **1674**
Port destino: 21 (este é o menor valor de port)

Chave obtida:

Valor IP: **200.23.45.7** 143.54.83.11
Valor port: **1674** 21

FIGURA 4.12 - Obtenção da chave única para identificação da Sessão

A estrutura de um nodo da árvore é apresentada abaixo (Fig. 4.13). O valor da chave é armazenado na variável "value" (do tipo *struct content*). Há uma referência à fila de "timestamps" (variável *time*) que também referencia o respectivo nodo da árvore. Desta forma, para verificar se há alguma sessão em "timeout" basta verificar a fila de "timestamps" (que está ordenada) evitando que toda a árvore tenha de ser percorrida para encontrar prováveis sessões em "timeout". O estado da conexão nos dois sites envolvidos é mantido no vetor *state*. A contabilização de pacotes e bytes é armazenado

nas variáveis *packets* e *bytes*. Também é mantido o tempo médio (em microsegundos) de filtragem de cada pacote da sessão.

```
struct node {
        struct node *father;        /* father */
        struct node *left, *right;  /* left and right nodes  */
        int L, R;                   /* left and right heights */
        struct content *value;
        struct list *time;          /* timestamp */
        u_char state[2];            /* state of the connection */
        u_long packets,bytes;       /* packet e byte counter  */
        u_char process_time;        /* time spent by processing packet */
};
struct content {
        u_long ip_big;              /* biggest ip value  */
        u_long ip_small;            /* smaller ip value  */
        u_short port_big;           /* biggest port value */
        u_short port_small;         /* smaller port value */
};
```

FIGURA 4.13 - Estrutura adotada como elemento da "cache"

Mesmo após a detecção dos últimos pacotes de uma sessão, deve-se manter o registro da sessão na "cache" durante um certo tempo (Veja novamente o diagrama de estados na Fig. 4.5). Caso contrário, pode ocorrer o extravio de algum pacote e a sua consequente retransmissão, e se não houver mais registro da sessão na "cache" o pacote válido seria descartado.

4.2.5 O Filtro e o Kernel do FreeBSD

O filtro é incorporado ao módulo de rede do "kernel" do sistema operacional FreeBSD (Fig. 4.14). O filtro *ip_fw* distribuído junto com o sistema operacional foi a referência básica para o desenvolvimento do "session filter".

Quando o sistema de rede é inicializado (no "boot") é chamada uma rotina de inicialização do filtro. Cada pacote que flui pelo módulo de rede, após o processamento normal executado, é enviado ao filtro para verificar sua permissão. Caso o filtro responda negativamente (valor zero) o pacote é descartado; de outra forma, o pacote é encaminhado para o seu destino.

4.2.6 O utilitário "sipfw"

O utilitário "sipfw" é a versão alterada do "ipfw" desenvolvido para interagir com o filtro de pacotes. O "ipfw" possibilita as seguintes facilidades:

- Inserção e remoção de regras de filtragem;
- Listagem das regras de filtragem;
- Listagem de "accounting";

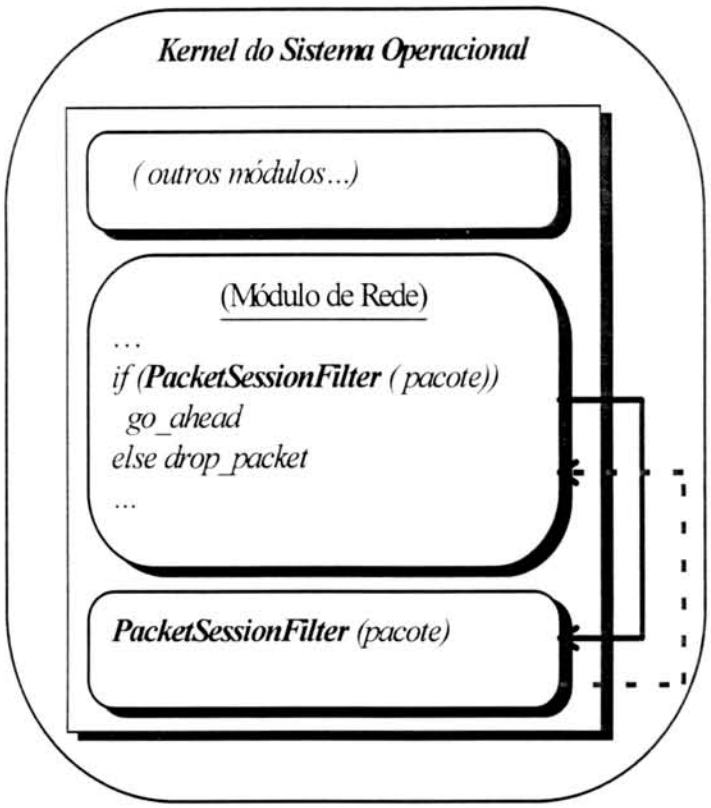


FIGURA 4.14 - Incorporação do Filtro ao Kernel do Sistema Operacional

Como fora citado anteriormente, o "ipfw" está disponível como freeware. Portanto, ele foi adaptado para suportar o filtro desenvolvido nesse trabalho. Para suportar regras que especifiquem uma sessão, a sintaxe das regras de filtragem do "ipfw" foi estendida no "sipfw". As alterações realizadas permitem o acréscimo dos seguintes recursos:

- Regras que descrevem sessões;
- Listagem das sessões presentes na "cache";
- Estatísticas (para cada sessão ativa): número de pacotes, número de bytes, tempo médio de processamento de cada pacote da sessão (em microsegundos) e "timestamp" do último pacote;
- Alteração no valor do "timeout" de exclusão de um elemento da "cache".

A sintaxe das regras de filtragem do "sipfw" é apresentada no Anexo 2.

4.2.6.1 Interação do "sipfw" com o filtro

O "sipfw" se comunica com o filtro de pacotes via "raw sockets". Este tipo de sockets só pode ser utilizado pelo superusuário ("root"), garantindo que as alterações são realizadas por pessoal autorizado.

A mensagem é enviada e armazenada em um "buffer". O código da mensagem é utilizado para saber qual o tipo de operação (inserção, remoção, listagem das regras, etc)

deve ser executada. A rotina de tratamento do filtro de pacotes é ativada e a operação é executada (Fig. 4.15).

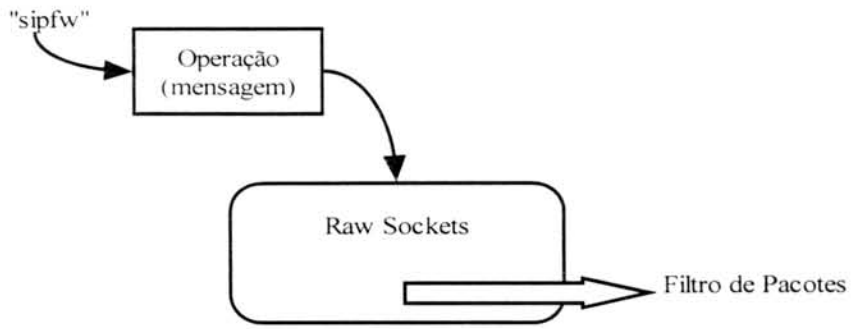


FIGURA 4.15 - Interação entre o "sipfw" e o filtro de pacotes

4.2.6.2 Informações de Monitoramento

Como fora citado anteriormente, o monitoramente permite ao administrador acompanhar as sessões existentes entre a rede interna e a Internet. Um exemplo dos resultados fornecidos pelo "sipfw" é apresentado na Figura 4.16. Cada entrada contém informações referentes as máquinas envolvidas, "ports", número de pacotes, número de bytes, tempo médio de processamento (em microssegundos) de filtragem dos pacotes da referida sessão e "timestamp" do último pacote registrado.

```
Information about sessions currently in the Session Cache:

ijui.inf.ufrgs.br  corvete.inf.ufrgs.br          1114  http       32 (packets)  2288 (bytes)  time = 8  -
- Wed Aug 13 13:31:24 1997
ijui.inf.ufrgs.br  corvete.inf.ufrgs.br          1113  http       30 (packets)  2216 (bytes)  time = 7  -
- Wed Aug 13 13:31:24 1997
ijui.inf.ufrgs.br  corvete.inf.ufrgs.br          1115  http       30 (packets)  2212 (bytes)  time = 7  -
- Wed Aug 13 13:31:24 1997
ijui.inf.ufrgs.br  corvete.inf.ufrgs.br          1116  http       30 (packets)  2212 (bytes)  time = 8  -
- Wed Aug 13 13:31:24 1997
ijui.inf.ufrgs.br  corvete.inf.ufrgs.br          1112  http       30 (packets)  2204 (bytes)  time = 7  -
- Wed Aug 13 13:31:24 1997
ijui.inf.ufrgs.br  corvete.inf.ufrgs.br          1111  http       120 (packets)  11234 (bytes)  time = 8
-- Wed Aug 13 13:31:24 1997
porsche-gw.inf.ufrgs.br  caracol-gw.inf.ufrgs.br      1027  ftp        20 (packets)  1286 (bytes)
time = 13  -- Wed Aug 13 13:29:14 1997
porsche-gw.inf.ufrgs.br  cepheus.inf.ufrgs.br         1028  telnet     65 (packets)  2999 (bytes)
time = 13  -- Wed Aug 13 13:29:57 1997
porsche-gw.inf.ufrgs.br  caracol-gw.inf.ufrgs.br      40001  ftp-data  8 (packets)  763 (bytes)  time
= 12  -- Wed Aug 13 13:31:59 1997
porsche-gw.inf.ufrgs.br  rigel.inf.ufrgs.br      1715  finger     10 (packets)  563 (bytes)  time = 14  -
- Wed Aug 13 13:32:39 1997
porsche-gw.inf.ufrgs.br  rigel.inf.ufrgs.br      2807  auth       2 (packets)  84 (bytes)  time = 25  --
Wed Aug 13 13:32:39 1997
```

FIGURA 4.16 - Informações fornecidas pelo "sipfw"

5 Avaliação de desempenho

O objetivo dos testes realizados é comparar o desempenho do filtro de sessão com relação ao filtro convencional. Isto é feito através da medida do tempo médio de filtragem dos pacotes em cada um dos filtros. O tempo é obtido utilizando a função "microtime", disponível na plataforma FreeBSD, que apresenta uma definição na ordem de microssegundos.

5.1 Plataforma de Testes

É importante descrever a plataforma de testes porque tem influência direta no desempenho do filtro. Os testes com os dois filtros foram realizados na mesma máquina, com a mesma configuração. As principais características do "gateway" onde o filtro estava ativo são as seguintes:

- Processador Pentium 133 MHz;
- 32 Mbytes de memória RAM;
- Interface de Rede Ethernet.

De fato, como o filtro faz parte do "Kernel" e está sempre na memória, os recursos mais importantes para um melhor desempenho são o processador e a memória principal.

5.2 Cenários

A filtragem se dá a nível do Kernel do sistema operacional e é realizada atomicamente; portanto, a filtragem dos pacotes referentes a diferentes serviços pode ser mensurada sem que ocorra interferência.

O fator mais importante na escolha do serviço para testes é que este seja baseado no protocolo TCP a fim de permitir comparações de desempenho entre o filtro convencional e o filtro de sessão. Na medida do "overhead" do filtro não influi o serviço em questão e sim a localização da regra na lista de regras. O serviço escolhido para a realização dos testes é o "finger".

A lista de regras de filtragem (Fig. 5.1) foi obtida considerando os serviços utilizados na sub-rede em teste. As quatro primeiras regras são relativas ao protocolo ICMP: a primeira permite que pacotes ICMP fluam da rede interna (endereço 143.54.83.0) para qualquer outra máquina, as duas regras seguintes impedem que ingressem na sub-rede pacotes ICMP do tipo "redirect" (código 5) e "echo request" (código 8) e a quarta permite que demais tipos de pacotes ICMP ingressem na rede. As demais regras se referem a serviços baseados nos protocolos UDP e TCP, os quais são os seguintes: Autenticação (port 113), telnet (23), rlogin (513), nntp (119), http (80), smtp (25), ftp (20 e 21), gopher (70), whois (43), talk (517,518, 1023-65535), finger (79), dns (53), rip (520), nfs (2049), lpr (515). Vale a pena salientar que o número de regras é ampliado quando o serviço é "inbound" (servidor está na sub-rede interna) e

"outbound" (servidor é externo, na Internet), como por exemplo o serviço de telnet (2 regras de filtragem para cada uma das situações).

```
1  allow icmp from 143.54.83.0/24 to any
2  deny icmp from any to 143.54.83.0/24 icmptypes 8
3  deny icmp from any to 143.54.83.0/24 icmptypes 5
4  allow icmp from any to 143.54.83.0/24
5  allow tcp from 143.54.83.0/24 1023-65535 to any 113
6  allow tcp from any 113 to 143.54.83.0/24 1023-65535
7  allow tcp from any 1023-65535 to 143.54.83.0/24 113
8  allow tcp from 143.54.83.0/24 113 to any 1023-65535
9  allow tcp from 143.54.83.0/24 1023-65535 to any 23
10 allow tcp from any 23 to 143.54.83.0/24 1023-65535
11 allow tcp from any 1023-65535 to 143.54.83.0/24 23
12 allow tcp from 143.54.83.0/24 23 to any 1023-65535
13 allow tcp from 143.54.83.0/24 1023-65535 to any 513
14 allow tcp from any 513 to 143.54.83.0/24 1023-65535
15 allow tcp from 143.54.83.0/24 1023-65535 to any 119
16 allow tcp from any 119 to 143.54.83.0/24 1023-65535
17 allow tcp from 143.54.83.0/24 1023-65535 to any 80
18 allow tcp from any 80 to 143.54.83.0/24 1023-65535
19 allow tcp from 143.54.83.0/24 1023-65535 to any 25
20 allow tcp from any 25 to 143.54.83.0/24 1023-65535
21 allow tcp from any 1023-65535 to 143.54.83.0/24 25
22 allow tcp from 143.54.83.0/24 25 to any 1023-65535
23 allow tcp from 143.54.83.0/24 1023-65535 to any 70
24 allow tcp from any 70 to 143.54.83.0/24 1023-65535
25 allow tcp from any 1023-65535 to 143.54.83.0/24 43
26 allow tcp from 143.54.83.0/24 43 to any 1023-65535
27 allow udp from 143.54.83.0/24 1023-65535 to any 517,518
28 allow udp from any 517,518 to 143.54.83.0/24 1023-65535
29 allow udp from any 1023-65535 to 143.54.83.0/24 517,518
30 allow udp from 143.54.83.0/24 517.518 to any 1023-65535
31 allow tcp from 143.54.83.0/24 1023-65535 to any 1023-65535
32 allow tcp from any 1023-65535 to 143.54.83.0/24 1023-65535
33 allow udp from 143.54.83.0/24 1023-65535 to any 53
34 allow udp from any 53 to 143.54.83.0/24 1023-65535
35 allow tcp from 143.54.83.0/24 1023-65535 to any 53
36 allow tcp from any 53 to 143.54.83.0/24 1023-65535
37 allow udp from 143.54.83.0/24 1023-65535 to any 520
38 allow udp from any 520 to 143.54.83.0/24 1023-65535
39 allow udp from any 1023-65535 to 143.54.83.0/24 520
40 allow udp from any 143.54.83.0/24 to any 1023-65535
41 allow udp from 143.54.83.0/24 520 to any 520
42  allow udp from any 520 to 143.54.83.0/24 520
43  allow tcp from 143.54.83.0/24 1023-65535 to 143.54.0.0/16 2049
44 allow tcp from 143.54.0.0/16 2049 to 143.54.83.0/24 1023-65535
45 allow tcp from 143.54.83.0/24 to any 515
46 allow tcp from any 515 to 143.54.83.0/24
47 allow tcp from 143.54.83.0/24 1023-65535 to any 21
48 allow tcp from any 21 to 143.54.83.0/24 1023-65535
49 allow tcp from any 20 to 143.54.83.0/24 1023-65535
50 allow tcp from 143.54.83.0/24 1023-65535 to any 20
51 allow tcp from any 1023-65535 to 143.54.83.0/24 21
52 allow tcp from 143.54.83.0/24 21 to any 1023-65535
53 allow tcp from any 1023-65535 to 143.54.83.0/24 20
54 allow tcp from 143.54.83.0/24 20 to any 1023-65535
55 allow tcp from 143.54.83.0/24 1023-65535 to any 79
56 allow tcp from any 79 to 143.54.83.0/24 1023-65535
```

FIGURA 5.1 - Regras de filtragem para o filtro convencional

No filtro de sessão as regras de filtragem são diferentes para os serviços baseados no protocolo TCP (Fig. 5.2). Apenas uma regra é necessária para habilitar um serviço, a qual é consultada uma única vez a cada nova conexão. O número de regras reduziu de 56 (filtro convencional) para 37.

O tempo médio de filtragem de cada pacote depende da posição da regra que permite o fluxo do pacote. Isto é relevante para o filtro convencional, já no filtro de sessão o mais relevante é o número de sessões registradas na "cache". O tempo médio

de acesso a "cache" é proporcional a *log n* onde *n* é o número de nodos ("sessões") na cache.

Em cada cenário de testes a variação na quantidade de sessões na "cache" é garantida através de múltiplos testes tendo cada um deles um número superior de sessões simultâneas. A regra de filtragem que permite o serviço testado também é deslocada para averiguar qual a alteração de comportamento. No pior caso, a regra é a última; no melhor caso, ela é a primeira e, no caso médio, ela é a regra intermediária. Ou seja, com relação as regras apresentadas na Figura 5.1, os três cenários apresentam as seguintes posições: melhor caso, regras 1 e 2; caso médio, regras 27 e 28 e, pior caso, regras 55 e 56.

```
1.  allow icmp from 14354830/24 to any
2.  deny icmp from any to 14354830/24 icmptypes 8
3.  deny icmp from any to 14354830/24 icmptypes 5
4.  allow icmp from any to 14354830/24
5.  allow session from 14354830/24 900-65535 to any 113
6.  allow session from any 900-65535 to 14354830/24 113
7.  allow session from 14354830/24 900-65535 to any 23
8.  allow session from any 900-65535 to 14354830/24 23
9.  allow session from 14354830/24 900-65535 to any 513
10. allow session from 14354830/24 900-65535 to any 119
11. allow session from 14354830/24 900-65535 to any 80
12. allow session from 14354830/24 900-65535 to any 25
13. allow session from any 900-65535 to 14354830/24 25
14. allow session from 14354830/24 900-65535 to any 70
15. allow session from 14354830/24 900-65535 to any 43
16. allow udp from 14354830/24 900-65535 to any 517,518
17. allow udp from any 517,518 to 14354830/24 900-65535
18. allow udp from any 900-65535 to 14354830/24 517,518
19. allow udp from 14354830/24 517,518 to any 900-65535
20. allow session from 14354830/24 900-65535 to any 900-65535
21. allow session from any 900-65535 to 14354830/24 900-65535
22. allow udp from 14354830/24 900-65535 to any 53
23. allow udp from any 53 to 14354830/24 900-65535
24. allow session from 14354830/24 900-65535 to any 53
25. allow udp from 14354830/24 900-65535 to any 520
26. allow udp from any 520 to 14354830/24 900-65535
27. allow udp from any 900-65535 to 14354830/24 520
28. allow udp from any 14354830/24 to any 900-65535
29. allow udp from 14354830/24 520 to any 520
30. allow udp from any 520 to 14354830/24 520
31. allow session from 14354830/24 900-65535 to 1435400/16 2049
32. allow session from 14354830/24 to any 515
33. allow session from 14354830/24 900-65535 to any 21
34. allow session from any 20 to 14354830/24 900-65535
35. allow session from any 900-65535 to 14354830/24 21
36. allow session from 14354830/24 20 to any 900-65535
37. allow session from 14354830/24 900-65535 to any 79
```

FIGURA 5.2 - Regras de filtragem para o filtro de sessão

A topologia da rede utilizada é apresentada na Figura 5.3. Um número variado de sessões são inicializadas a partir da sub-rede 143.54.83.0 tendo como servidoras máquinas localizadas na sub-rede externa. O filtro de pacotes está localizado junto ao "gateway" da sub-rede.

Foram realizadas um total de 59400 sessões finger (ver "script" utilizado no Anexo 1) . Para cada um dos filtros (convencional e sessão) se executou uma bateria de testes com 90, 360 e 540 sessões simultâneas. Os testes foram repetidos 10 vezes para cada um dos casos (melhor, médio e pior) nos dois filtros; ou seja, 10 vezes 90 sessões para cada uma das três diferentes listas de regras de filtragem em cada um dos filtros, idem para as outras quantidades de sessões.

Pode-se supor que no filtro de sessão a posição da regra de filtragem que habilita a passagem do pacote não tenha muita relevância visto que, apesar do primeiro pacote passar pelas regras de filtragem, o tempo dispendido na filtragem inicial disperse nos próximos tempos de acesso à "cache".

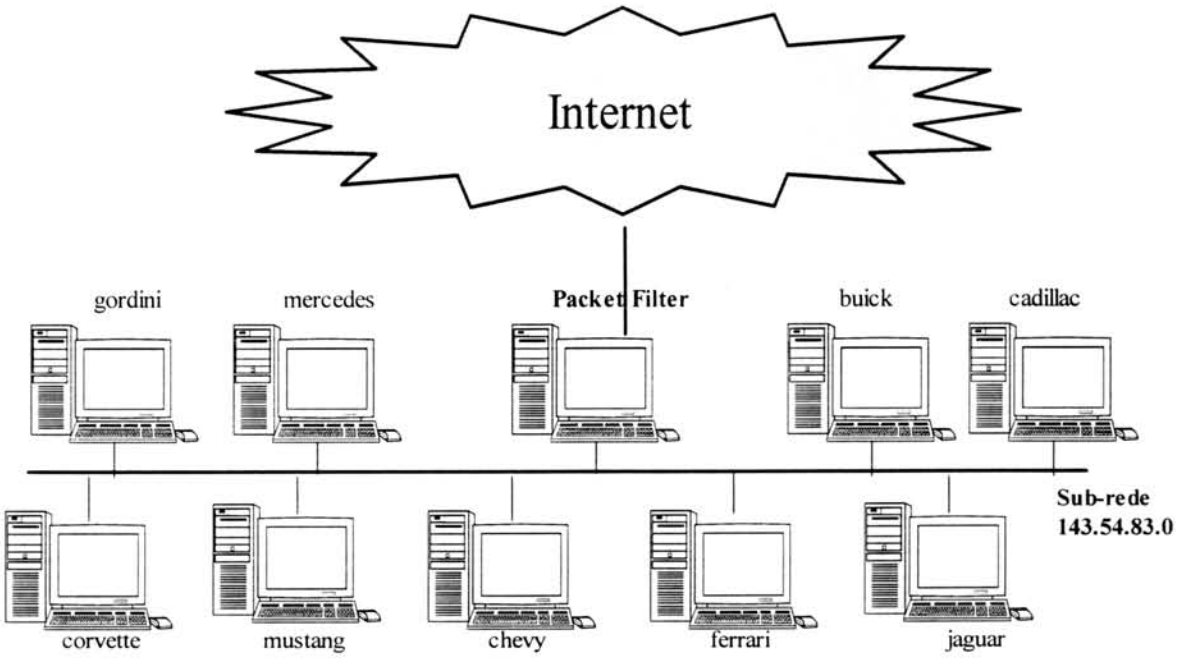


FIGURA 5.3: Topologia da rede em teste

5.3 Resultados

5.3.1 Cenário 1: 90 sessões simultâneas

O primeiro cenário corresponde a 90 sessões simultâneas representando um total de 2700 sessões (900 para cada um dos casos) em cada um dos filtros. O comportamento do filtro convencional se apresentou como esperado (Fig. 5.4), o pior caso teve um tempo médio de 38,4 μs, o caso médio 20,4 μs e o melhor caso 5,2 μs. Ou seja, o tempo médio de filtragem dos pacotes é proporcional a posição ocupada pela regra na lista de regras. O tempo no caso médio (regras números 27 e 28) corresponde aproximadamente a metade do tempo no pior caso (regras 55 e 56).

O filtro de sessão também apresentou um comportamento esperado (Fig. 5.5). Além das 90 sessões, 60 sessões de autenticação estiveram presentes na "cache". As sessões de autenticação foram inicializadas a partir das máquinas servidoras do "finger". A posição da regra não afetou significativamente o resultado. O tempo médio de filtragem apresentou um valor de 9,3 μs no pior caso, 8,9 μs no caso médio e 8,3 μs no melhor caso. Pode-se dizer que o tempo obtido representa praticamente o tempo médio de acesso à "cache" de sessões ativas.

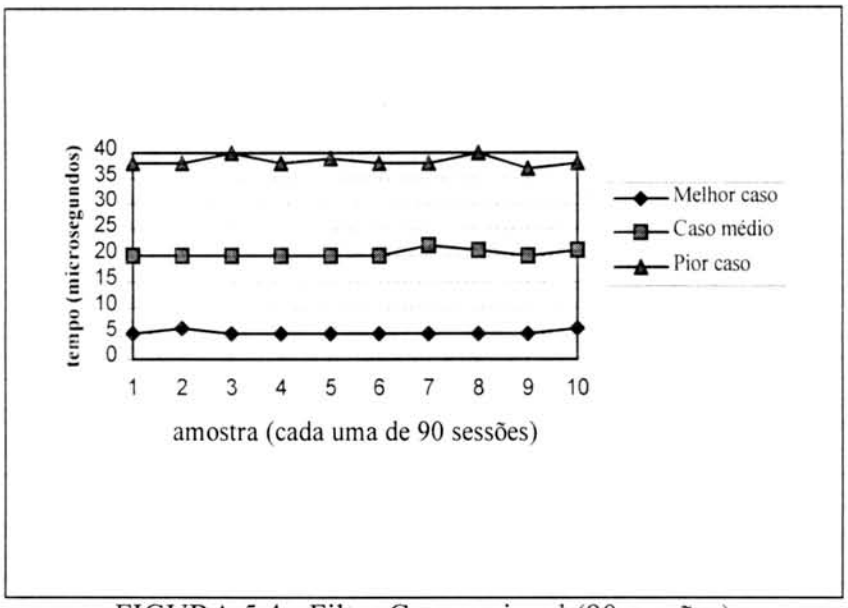


FIGURA 5.4 - Filtro Convencional (90 sessões)

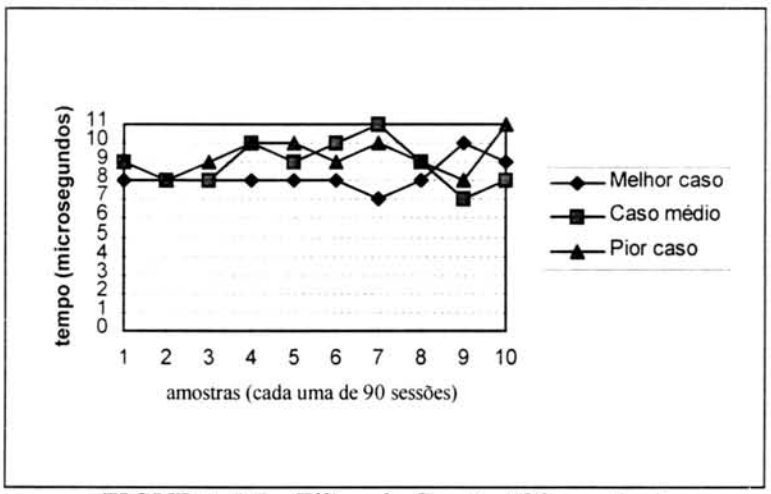


FIGURA 5.5 - Filtro de Sessão (90 sessões)

5.3.2 Cenário 2: 360 sessões simultâneas

Neste cenário 360 sessões simultâneas foram executadas 10 vezes em cada um dos casos, totalizando 10800 sessões em cada um dos filtros. O comportamento do filtro convencional (Fig. 5.6) foi praticamente idêntico ao registrado no primeiro cenário. Isto era esperado considerando que o número de sessões não tem influência porque o "overhead" de filtragem de cada pacote deve ser praticamente o mesmo para uma determinada posição da regra de filtragem. O pior caso apresentou um tempo médio de 39,2 μs, o caso médio 20,2 μs e o melhor caso 5,0 μs.

Além das 360 sessões finger estiveram presentes na "cache" 120 sessões de autenticação. O filtro de sessão também teve um comportamento semelhante ao apresentado no primeiro cenário (Fig. 5.7). O tempo médio de filtragem apresentou um valor de 8,9 μs no pior caso, 9,1 μs no caso médio e 10 μs no melhor caso. Novamente fica explícito que a ordem da regra não tem influência significativa, visto que o

"melhor" caso apresentou um valor maior que o "pior" caso. Atribui-se essa diferença entre os três casos a precisão na obtenção do tempo.

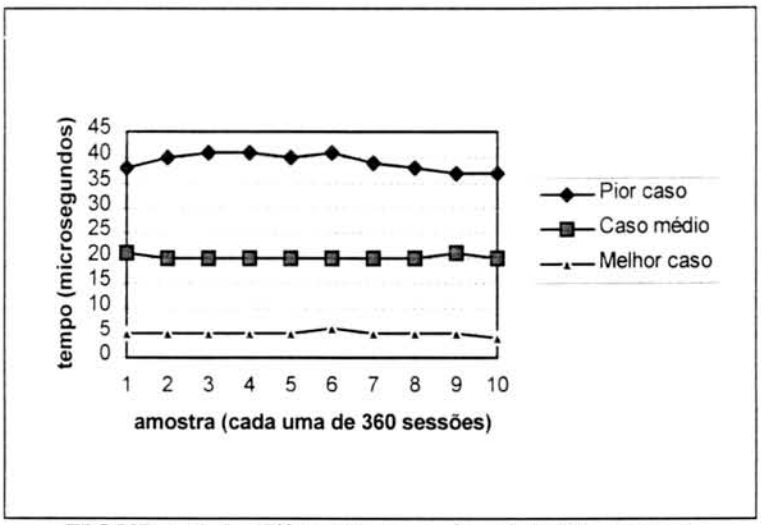


FIGURA 5.6 - Filtro Convencional (360 sessões)

Considerando que havia 480 (360 finger + 120 auth) sessões registradas na "cache" o tempo médio de acesso foi proporcional a 8,9 (*"log 480"*) operações sobre a árvore AVL, enquanto que no primeiro cenário havia 150 (90 finger + 60 auth) sessões e o tempo médio de acesso foi proporcional a 7,2 operações. Ou seja, este pode ser o motivo da proximidade dos resultados nos dois cenários, sobretudo porque há uma diferença média de apenas 1,7 operações sobre a árvore AVL.

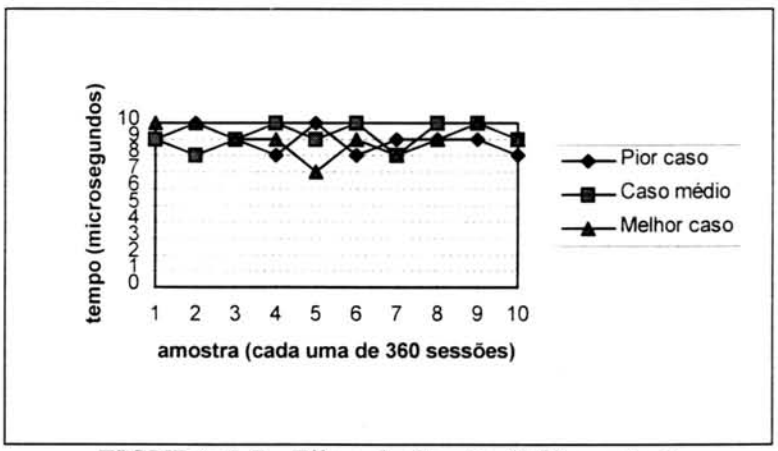


FIGURA 5.7 - Filtro de Sessão (360 sessões)

5.3.3 Cenário 3: 540 sessões simultâneas

Neste cenário foram executadas 540 sessões finger simultâneas, 10 vezes para cada um dos casos, totalizando 16200 sessões em cada um dos filtros. O tempo médio de filtragem apresentou um valor de 38,7 μs no pior caso, 20,5 μs no caso médio e 4,8 μs no melhor caso. O comportamento, esperado, foi semelhante ao ocorrido nos dois cenários anteriores (Fig. 5.8).

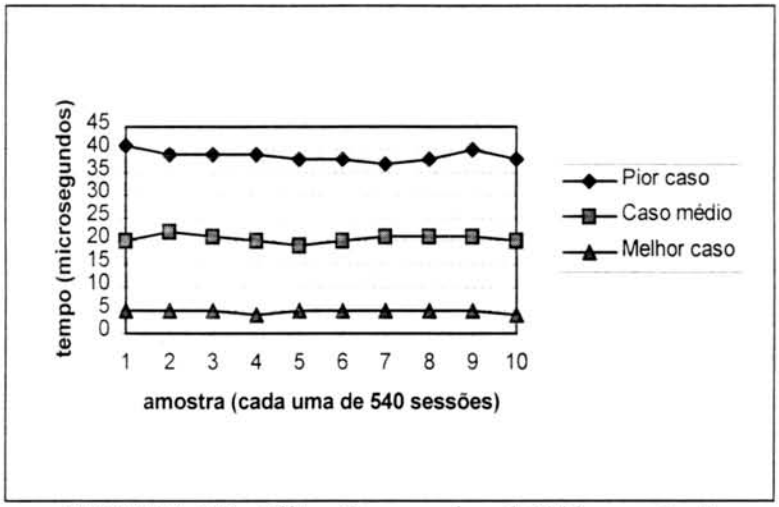


FIGURA 5.8 - Filtro Convencional (540 sesssões)

Além das 540 sessões finger estiveram presentes na "cache" 240 sessões de autenticação. O tempo médio de filtragem apresentou um valor de 9,5 μs no pior caso, 9,1 μs no caso médio e 9,2 μs no melhor caso (Fig. 5.9).

Considerando que havia 780 (540 finger + 240 auth) sessões registradas na "cache" o tempo médio de acesso foi proporcional a 9,6 (*"log 780"*) operações sobre a árvore AVL, enquanto que no segundo cenário havia 480 (360 finger + 120 auth) sessões e o tempo médio de acesso foi proporcional a 8,9 operações. Ou seja, este pode ser o motivo da proximidade dos resultados nos dois cenários, sobretudo porque há uma diferença média de apenas 0,7 operações sobre a árvore AVL. A diferença de operações entre o primeiro e o terceiro cenário é de 2,4.

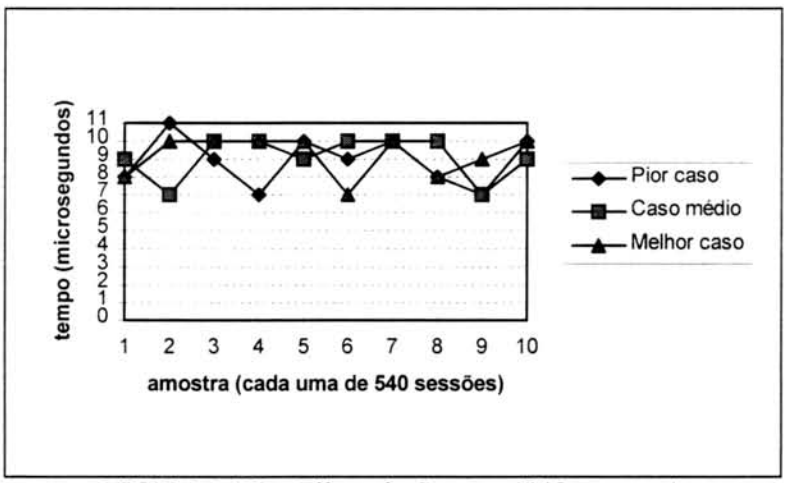


FIGURA 5.9 - Filtro de Sessão (540 sessões)

5.4 Análise Comparativa

Os três cenários de teste permitem concluir o seguinte:

- A ordem da regra na lista de filtragem afeta significativamente o tempo médio de filtragem apenas no filtro convencional;
- O número de sessões ativas não afeta o tempo médio de filtragem no filtro convencional porque cada pacote é analisado sem considerar pacotes anteriores (sessão);

- O filtro de sessão representa, considerando a média entre as três posições da regra de filtragem, um ganho significativo na filtragem dos pacotes;
- A estrutura em "Árvore AVL" é uma boa solução para a "cache" de sessões ativas.

Obtendo o valor médio entre os três casos em cada um dos cenários, pode-se supor o ganho real na filtragem dos pacotes de todos os serviços TCP se considerarmos que todas as regras de filtragem das respectivas sessões presentes na lista são acessadas com a mesma frequência. Os valores médios nos três cenários são apresentados na tabela abaixo.

TABELA 5.1 - Tempo médio de filtragem

Filtro	Cenário 1	Cenário 2	Cenário 3
Filtro de Sessão	**8,83μs**	**9,33μs**	**9,1μs**
Convencional	**21,33μs**	**21,46μs**	**21,33μs**

6 Conclusão

A filtragem de pacotes representa um recurso importante na segurança de redes conectadas à Internet. Mesmo que os serviços adquiram mecanismos mais robustos de segurança (criptografica, assinatura digital, senhas não reutilizáveis, etc) a filtragem pode se mostrar como uma solução indispensável à aplicação da política de segurança. Por exemplo, para limitar o tráfego entre dois segmentos de rede um filtro de pacotes já é suficiente. A postura prudente também pode ser aplicada facilmente com o recurso de filtragem. Tentativas de ataques conhecidos podem ser bloqueados e monitorados, bem como o filtro pode ser utilizado para barrar tentativas de ataques partindo da rede interna ("IP spoofing", por exemplo).

A plataforma escolhida, o sistema operacional FreeBSD, é uma ótima alternativa para se explorar os recursos disponíveis no módulo de rede. Todo o código fonte do sistema operacional é distribuído livremente, garantindo dessa forma livre acesso para alterações. O filtro de pacotes disponível nessa plataforma (*ip_fw*) serviu como uma referência básica ao filtro desenvolvido.

A compilação do Kernel foi realizada diversas vezes durante a fase de implementação do filtro. Novamente, a plataforma FreeBSD se mostrou fácil de compilar. Todavia, não se encontrou uma documentação específica para desenvolvimento a nível de Kernel. É necessário um esforço maior do que o desenvolvimento de uma aplicação, sobretudo porque qualquer "bug" a nível de kernel pode levar facilmente a um "crash" do sistema.

O utilitário "ipfw" foi alterado para suportar o filtro desenvolvido. A sintaxe das regras de filtragem foi mantida salvo algumas pequenas alterações, garantindo assim que quase todas as regras de filtragem utilizadas com o filtro convencional "ip_fw" possam ser reutilizadas. A possibilidade de monitoramento das sessões é um recurso valioso porque permite em qualquer instante listar quais as sessões ativas entre a rede interna e a Internet.

A configuração do filtro deve ser feita com o conhecimento dos serviços permitidos. É importante que se faça uma tabela com os possíveis pacotes de cada serviço, da mesma forma como foi apresentado no capítulo 3. As regras de filtragem são então elaboradas a partir da política de segurança adotada.

A avaliação de desempenho do filtro desenvolvido apresentou dados esperados. Os resultados obtidos após a execução de 59400 sessões "finger" permitem concluir que o filtro de sessões acarreta um "overhead" consideravelmente menor ao filtro convencional. A escolha de uma estrutura em "Árvore AVL" para a "cache" de sessões foi fundamental para o bom desempenho do filtro de sessões.

O número de regras de filtragem também é reduzido com o acréscimo da regra de sessão ("session"). No filtro convencional são necessárias duas regras de filtragem para habilitar um serviço baseado no protocolo TCP; enquanto que no filtro de sessões apenas uma regra é necessária, considerando-se sobretudo que os pacotes seguintes à solicitação de conexão são verificados na "cache" de sessões ativas.

A elaboração das regras de filtragem deve ser feita com muito cuidado a fim de se evitar que pacotes que deveriam ser barrados não o são e pacotes que deveriam passar livremente não o fazem.

O melhor desempenho, a possibilidade de monitoramento e o grande número de serviços baseados no protocolo TCP justificam a utilização de um filtro de sessão. Deve-se observar que outras estruturas de dados podem ser testadas como "cache" a fim de se obter um desempenho ainda melhor.

Anexo 1

Script utilizado nos testes de desempenho

O "script" apresentado abaixo foi utilizado para criar as sessões finger. As máquinas envolvidas (citadas nesse exemplo) não são as únicas utilizadas. Outras máquinas alvo foram utilizadas a medida que o número de sessões aumentou a fim de não sobrecarregar os servidores finger.

```
#!/bin/sh
#
# Maquinas envolvidas: arara, cepheus, tucano
#
#
PATH=/bin:/usr/bin:/etc:/usr/etc:/usr/ucb; export PATH
Echo=/bin/echo

if [ $# -eq 0 ]; then
        $Echo "usage: $0 #_of_tests"
        exit
fi


cat > out1 << END
Output dos testes do finger:
END


N=$1

while [ "$N" -ge 1 ]
do
        echo Test $N
        (finger @arara  >> out1)&
        (finger @cepheus >>  out1)&
        (finger @tucano  >> out1)&
        N=`expr $N - 1`
done
```

Anexo 2

Recursos do utilitário "sipfw"

O utilitário "sipfw" é uma adaptação do freeware "ipfw" utilizado no filtro de pacotes convencional "ip_fw". As opções disponíveis no "ipfw" e que continuam presentes no "sipfw" são as seguintes:

- "**flush**": limpa a lista de regras de filtragem;
- "**zero** [number]": zera os contadores de todas as regras de filtragem ou somente daquela cujo número é fornecido;
- "**delete** [number]": remove a regra de filtragem cujo número é fornecido;
- "**list**": lista as regras de filtragem;
- "**add** [number] **action** [log] **proto from src to dst** [via name|ipno] [options]": adiciona uma nova regra de filtagem.

As opções em "action" são as seguintes:

- "allow": permite a passagem do pacote;
- "pass": o mesmo que "allow";
- "accept": o mesmo que "allow";
- "count": somente atualiza a contabilização de pacotes e após continua a procura na lista de regras;
- "deny": descarta o pacote;
- "reject": descarta o pacote e tenta enviar uma mensagem ICMP de aviso.

As opções de protocolo ("proto") são as seguintes:

- "ip": todos os protocolos satisfazem;
- "all": o mesmo que "ip";
- "udp": somente pacotes udp;
- "icmp": somente pacotes icmp;

Com referência a fonte ("src") e destino ("dst") tem-se a seguinte estrutura: <address/mask> [ports]. O endereço IP e a máscara pode ser especificada como:

- Com o número IP: por exemplo 143.54.83.2, desta forma apenas pacotes com esse número IP satisfazem;
- Número IP/bits: por exemplo 143.54.83.0/24, desta forma todos os pacotes com endereço entre 143.54.83.0 e 143.54.83.255 satisfazem;
- Número IP:máscara: por exemplo 143.54.83.0:255.255.255.0, tendo o mesmo significado que o exemplo anterior;

Quando o protocolo for UDP ou TCP ("session"), os valores de "ports" são especificados da seguinte forma:

- Um único valor: por exemplo "port 25", neste caso somente o valor 25 satisfaz;
- Vários valores separados por vírgula: por exemplo "port 25, 26, 27", neste caso todos os valores citados satisfazem;

- Intervalo de valores: por exemplo "port 1023-6000", neste caso todos os valores entre 1023 e 6000 satisfazem.

Quando se especifica a interface a qual deve ser aplicada a regra se tem as seguintes possibilidades:

- "via name": nesse caso apenas os pacotes que são recebidos via ou encaminhados através da interface "name" satisfazem a regra; ex: "via ed1", onde "ed1" é o nome da interface;
- "via ipno": o mesmo caso anterior, com a única diferença que se está utilizando o endereço IP correspondente a interface; ex: "via 143.54.8.110", onde "143.54.8.110" é o endereço da interface.

As demais opções ("options") são as seguintes:

- "frag": satisfaz se é um fragmento (exceto o primeiro, com "offset" zero);
- "in" e "out": satisfaz se o pacote está entrando (in) ou saindo (out);
- "ipoptions **spec**": satisfaz se as opções IP descritas em "spec" estão presentes ou ausentes (quando antecedidas por "!"). As opções suportadas são: "ssrr" (strict source route), "lsrr" (loose source route), "rr" (record packet route), e "ts" (timestamp). Exemplo: "ipoptions !ssrr, !lsrr", neste caso satisfazem os pacotes que não utilizam as opções "ssrr" e "lsrr".
- "icmptypes **types**": permite a especificação do tipo de mensagens icmp. Pode ser uma lista de valores separados por vírgula ou intervalos de valores.

O "sipfw" apresenta as seguintes extensões em relação ao "ipfw":

- Regra de sessão ("session"): ao contrário do filtro convencional onde há a opção de protocolo TCP, no filtro de sessão se utiliza a identificação "session" quando se escreve uma regra para algum serviço baseado no TCP. Ou seja, no campo de protocolo (proto) se utiliza a palavra "session" para uma regra de sessão. Eis um exemplo de uma regra de filtragem de sessão: "**allow session from 143.54.83.0/24 1023-6000 to 143.54.11.7 21 via ed1 out**".
- A opção "list" também apresenta, além das regras de filtragem, as sessões ativas na "cache" (em capítulo anterior foi descrito este recurso);
- "sipfw timeout **valor**": este comando permite que se altere dinamicamente o valor do "timeout" utilizado pelo filtro de sessões para eliminar uma entrada na "cache" há muito tempo sem ser referenciada. O valor fornecido é em segundos; ex: "sipfw timeout 900", altera o valor do "timeout" para 900 segundos (15 minutos).

Eis alguns exemplos de regras de filtragem:

allow icmp from 143.54.8.0/24 to any (permite pacotes icmp da sub-rede 143.54.8.0 para qualquer outra máquina)

deny icmp from any to 143.54.8.0/24 icmptypes 5 (bloqueia pacotes icmp de código 5, "redirect", de qualquer endereço para qualquer máquina da sub-rede 143.54.8.0)

allow udp from 143.54.8.0/24 1023-6000 to 143.54.11.7 53

allow udp from 143.54.11.7 to 143.54.8.0/24 1023-6000

Bibliografia

[AMO 96] AMOROSO, E.; SHARP, R. **PCWeek Intranet and Internet Firewall Strategies**. Emeryville, CA: Ziff Davis Press. 1996. 218p.

[CHA 94] CHAPMAN, D. B. **Network (in)security through IP packet filtering**. Mountain View, CA: Great Circles Associates, 1994. Disponível por http em www.greatcircle.com. (julho 1996).

[CHA 95] CHAPMAN, D. B.; ZWICKY, E. D. **Building Internet Firewalls**. Sebastopol, CA: O'Reilly & Associates, 1995.

[COM 91] COMER, D. E. **Internetworking with TCP/IP**: principles, protocols, and architecture. 2. ed. Englewood Cliffs: Prentice Hall, 1991. 547p.

[LEH 96] LEHEY, G. **Installing and Running FreeBSD**. Walnut Creek, CA: Walnut Creek CDROM, 1996. 232 p.

[MOG 89] MOGUL, J. **Simple and Flexible Datagram Access Controls for Unix- based Gateways**. [S.l] Digital Equipment Corporation, 1989.

[RAN 93] RANUM, M. **Thinking About Firewalls**. Glenwood, Maryland: Trusted Information Systems, 1993.

[RAN 96-1] RANUM, M. **Firewalls: The future?** Rockwille, MD: V-One Corporation. 1996.

[RAN 96-2] RANUM, M. **Are Firewalls obsolete? The debate**. Rockwille, MD: V-One Corporation, 1996.

[RFC 1858] REQUEST FOR COMMENTS 1858. **Security Considerations for IP Fragment Filtering**. [S.l] 1995. Disponível por ftp em nic.mil.

[SAF 92] SAFFORD, D. R.; SCHALES, D. L.; HESS, D. K. **The Tamu security package:** an ongoing response to internet intruders in an academic environment. Austin, TX: Computing and Information Services - Network Group, Texas A&M University, 1992.

[SIY 95] SIYAN, K.; HARE, C. **Internet Firewalls and Network Security**. Indianapolis, Indiana: New Riders Publishing,1995.

[SPO 96] SPOHN, M. A. **Internet Firewalls**: trabalho Individual. Porto Alegre: CPGCC da UFRGS, 1996. 69 f. (TI-554).

[STA 95] STANDISH, T. A. **Data Structures, algorithms, and software principles**. Reading: Addison Wesley. 1995.

[TAN 89] TANENBAUM, A. S. **Computer Networks**. 2. ed. Englewood Cliffs: Prentice-Hall, 1989. 658 p.

[TAN 96] TANENBAUM, A. S. **Computer Networks**. 3. ed. Englewood Cliffs: Prentice-Hall, 1996. 813 p.

[TER 91] TERADA, R. **Desenvolvimento de Algoritmo e Estruturas de Dados**. São Paulo: Editora McGraw-Hill, 1991.

Informática UFRGS

CURSO DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Desenvolvimento e Análise de Desempenho de um 'Packet/Session Filter'

por

Marco Aurélio Spohn

Dissertação apresentada aos Senhores:

Prof. Dr. Orlando Gomes Loques Filho (UFF)

Profa. Dra. Taisy Silva Weber

Prof. Dr. João César Netto

Vista e permitida a impressão.
Porto Alegre, 28 / 11 / 97.

Prof. Dr. Raul Fernando Weber,
Orientador.

Prof. Flávio Rech Wagner
Coordenador do Curso de Pós-Graduação
em Ciência da Computação - CPGCC