UNIVERSIDADE FEDERAL DO RIO GRANDE DO NORTE
CENTRO DE TECNOLOGIA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

# SuRFE - Sub-Rede de Filtragens Específicas

**Ricardo Kléber Martins Galvão**

Orientador: Prof. Dr. Sergio Vianna Fialho

**Dissertação de Mestrado** apresentada ao Programa de Pós-Graduação em Engenharia Elétrica da UFRN (área de concentração: Engenharia de Computação) como parte dos requisitos para obtenção do título de Mestre em Ciências.

Número de ordem PPgEE: M173
Natal, RN, julho de 2006

Divisão de Serviços Técnicos

Catalogação da publicação na fonte. UFRN / Biblioteca Central Zila Mamede

Galvão, Ricardo Kléber Martins.
SurRFE - Sub-rede de filtragens específicas / Ricardo Kléber Martins Galvão. - Natal, RN, 2006.
81 f. : il.

Orientador: Sergio Vianna Fialho

Dissertação (mestrado) - Universidade Federal do Rio Grande do Norte. Centro de Tecnologia. Programa de Pós-Graduação em Engenharia Elétrica.

1. Segurança da informação - Dissertação. 2. Sistemas de detecção de intrusões - Dissertação. 3. Filtro de pacotes - Dissertação. 4. Firewalls - Dissertação. I. Fialho, Sergio Vianna. II. Título.

RN/UF/BCZM CDU 004.056(043.3)

# SuRFE - Sub-Rede de Filtragens Específicas

**Ricardo Kléber Martins Galvão**

Dissertação de Mestrado aprovada em 11 de julho de 2006 pela banca examinadora composta pelos seguintes membros:

---

Prof. Dr. Sergio Vianna Fialho (orientador) ..... DCA/UFRN

---

Prof. Dr. João Batista Bezerra ..... DCA/UFRN

---

Prof. Dr. José Ribamar Silva Oliveira ..... DCA/UFRN

---

Prof. Dr. Guido Lemos de Souza Filho ..... DI/UFPB

*Às minhas três mulheres, Samara (esposa), Bia e Carol (filhas) responsáveis pelo meu equilíbrio na conclusão deste trabalho.*

# Agradecimentos

A Deus, pelo dom da vida.

Ao meu orientador e amigo professor Fialho.

A minha esposa Samara pela força, compreensão e companheirismo.

A minhas filhas Ana Beatriz e Ana Carolina.

Aos meus pais pela persistência na minha condução moral, educacional e afetiva.

A todos aqueles que de alguma forma contribuíram para tornar este trabalho possível.

# Resumo

O aumento do número de ataques a redes de computadores tem sido combatido com o incremento dos recursos aplicados diretamente nos equipamentos ativos de roteamento destas redes. Nesse contexto, os *firewalls* consolidaram-se como elementos essenciais no processo de controle de entrada e saída de pacotes em uma rede. O surgimento dos sistemas detectores de intrusão (IDS) levou a esforços no sentido de incorporar a filtragem de pacotes baseada em padrões ao *firewall* tradicional, integrando as funções do IDS (como a filtragem baseada em assinaturas, até então um elemento passivo) às funções já existentes no *firewall*. Em contrapartida à eficiência obtida através desta incorporação no bloqueio de ataques com assinaturas conhecidas, a filtragem no nível de aplicação, além de provocar um retardo natural nos pacotes analisados, pode comprometer o desempenho da máquina na filtragem dos demais pacotes, pela natural demanda por recursos da máquina para este nível de filtragem. Essa tese apresenta modelos de tratamento deste problema, baseados no re-roteamento dos pacotes para análise por uma sub-rede de filtragens específicas. A sugestão de implementação deste modelo visa, além de amenizar o problema de desempenho supra-citado, abrir espaço para a consolidação de cenários em que outras soluções de filtragem não convencionais (como ferramentas de bloqueio de SPAM, controle/bloqueio de tráfego P2P, e outras) possam ser inseridas na sub-rede de filtragem, sem implicar em sobrecarga do *firewall* principal da rede corporativa.

**Palavras-chave**: Segurança, *Firewalls*, Sistemas de Detecção de Intrusões, Filtro de Pacotes, Re-roteamento.

# Abstract

The increasing of the number of attacks in the computer networks has been treated with the increment of the resources that are applied directly in the active routers equipments of these networks. In this context, the firewalls had been consolidated as essential elements in the input and output control process of packets in a network. With the advent of intrusion detectors systems (IDS), efforts have been done in the direction to incorporate packets filtering based in standards of traditional firewalls. This integration incorporates the IDS functions (as filtering based on signatures, until then a passive element) with the already existing functions in firewall. In opposite of the efficiency due this incorporation in the blockage of signature known attacks, the filtering in the application level provokes a natural retard in the analyzed packets, and it can reduce the machine performance to filter the others packets because of machine resources demand by this level of filtering. This work presents models of treatment for this problem based in the packets re-routing for analysis by a sub-network with specific filterings. The suggestion of implementation of this model aims reducing the performance problem and opening a space for the consolidation of scenes where others not conventional filtering solutions (spam blockage, P2P traffic control/blockage, etc.) can be inserted in the filtering sub-network, without inplying in overload of the main firewall in a corporative network.

**Keywords**: Security, Firewalls, Intrusion Detection Systems, Packet Filter, Re-routing.

# Sumário

# Lista de Figuras

# Lista de Tabelas

# Capítulo 1

# Introdução

A filtragem de pacotes em redes de computadores é uma necessidade fundamental para a garantia mínima de bloqueio de ataques contra as redes corporativas, bem como para possiblitar a implementação de restrições do tráfego de saída, ou seja, dos pacotes originados na rede interna e destinados à Internet. Diante desta constatação, a incorporação de *firewalls* como soluções específicas para a realização dos mais variados níveis de filtragem é prática comum em redes de qualquer porte, tendo como requisito mínimo a sua conexão à Internet, situação padrão na maioria das instituições.

A localização estratégica dos *firewalls* no perímetro das redes, isolando-as lógica e fisicamente e impondo regras para o repasse de pacotes entre elas, faz com que novas funcionalidades sejam-lhe adicionadas a cada dia, como a conversão de endereços (*NAT - Network Address Translation*), funcionamento como *proxy* e filtragem baseada nos protocolos de todas as camadas da pilha TCP/IP. A incorporação de recursos adicionais em um elemento de roteamento causa, naturalmente, perda de desempenho, podendo chegar a comprometer o seu papel fundamental, que é o de rotear pacotes entre as redes.

A filtragem de pacotes em nível de aplicação está disponível nos principais *firewalls* de rede, com recursos de configuração e manutenção cada vez mais simples e funcionais, como apresentado em [Galvão 2002].

A filtragem em nível de aplicação, modalidade de filtragem que causa mais retardo no encaminhamento de pacotes, já que atinge a camada mais alta da pilha TCP/IP, quando feita de forma isolada para poucos protocolos e/ou protocolos com baixo volume de tráfego, podem não chegar a representar uma degradação considerável no tráfego dos demais pacotes entre as redes roteadas pelo *firewall*. Porém, a exigência deste nível de filtragem em praticamente todos os principais protocolos de aplicação aponta para uma situação crítica de esgotamento de recursos de hardware do *firewall*, bem como comprometimento considerável no encaminhamento de pacotes cuja filtragem neste nível não seria necessária.

## 1.1 Objetivos

Diante desta situação, aparentemente irreversível, de compromisso entre necessidade de filtragem versus capacidade do *firewall*, esta dissertação apresenta um novo modelo de filtragem. De acordo com esse modelo, tarefas específicas de filtragem são distribuídas

a outros *firewalls* de apoio, localizados estrategicamente em uma rede à parte, conectada diretamente ao *firewall* principal, à qual batizou-se de SuRFE - Sub Rede de Filtragens Específicas.

A partir da utilização deste modelo deve ser possível garantir um menor retardo no repasse de pacotes caracterizados pelo *firewall* principal como não suspeitos, (que não necessitem de filtragens específicas), e realizar o re-roteamento de pacotes específicos, a partir do *firewall* principal, para a SuRFE, onde estes deverão ser submetidos a regras específicas de filtragem de acordo com o seu perfil. Assim, para avaliar a melhoria no desempenho do *firewall*, um conjunto de testes foi realizado a partir de um dos cenários possíveis de implementação da SuRFE.

## 1.2 Organização do texto

Esse documento está então organizado da seguinte forma: o capítulo 2 apresenta os conceitos básicos relacionados à segurança da informação em redes corporativas; no capítulo 3 é apresentada uma visão geral sobre o tema desta dissertação, além do detalhamento dos cenários implementados para a realização dos testes de desempenho, ferramentas utilizadas e métodos de implementação. No capítulo 4 são apresentados metodologia, objetivos, métricas e ferramentas utilizadas nos testes, bem como o ambiente de testes e resultados obtidos. No capítulo 5, é apresentada a análise de resultados e no capítulo 6 as conclusões e tópicos a serem explorados, no contexto desta dissertação, em trabalhos futuros.

# Capítulo 2

# Fundamentação Teórica

## 2.1 Aspectos de Segurança em Redes Corporativas

Nesse capítulo são apresentados os principais conceitos relacionados à segurança da informação e sua aplicação em redes corporativas, métodos de ataque comumente utilizados e procedimentos de segurança sugeridos para a manutenção de uma rede corporativa minimamente segura, incluindo os componentes físicos e lógicos considerados como imprescindíveis para a filtragem de pacotes e detecção de intrusões.

Por fim, são apresentados alguns modelos de filtragem (cenários), acompanhados de uma análise crítica quanto às vantagens e desvantagens de sua utilização, além de uma discussão sobre os impactos da concentração de elementos de filtragem em um único componente da rede.

### 2.1.1 Segurança da Informação

Esta seção apresenta um breve histórico sobre o tratamento dispensado à segurança das informações através dos tempos, até a consolidação atual dos objetivos básicos esperados de todo projeto relacionado à segurança da informação, incluindo disponibilidade, integridade, confidencialidade e não-repúdio.

#### Histórico

As informações, bem como os conhecimentos atrelados a elas, são motivo de preocupação desde os primórdios da civilização humana. Isto é fato ao se observar o processo de escrita de alguns povos, como a antiga civilização egípcia, onde somente poucos privilegiados da alta sociedade tinham acesso aos manuscritos da época, e um contingente menor ainda ao processo de escrita destes manuscritos. O conhecimento reportado nas informações escritas nos hieroglifos egípcios tinham em sua escrita um mecanismo de proteção e perpetuação da cultura daquele povo.

O armazenamento e acesso a informações em computadores na sociedade moderna recebeu, a partir do surgimento dos primeiros equipamentos, pouca ou nenhuma atenção sob o ponto de vista de segurança destas informações. Este panorama só começou a mudar com o surgimento dos sistemas operacionais com estrutura de tempo compartilhado (*time-sharing*), permitindo o uso do computador por mais de uma pessoa ao mesmo tempo e,

conseqüentemente, o acesso às mesmas informações por parte de mais de um usuário. A falta de gerenciamento do acesso a informações compartilhadas passou a causar efeitos imprevisíveis, comprometendo a integridade dos dados compartilhados e impossibilitando em muitos casos a identificação da autoria de procedimentos como criação, alteração e/ou exclusão destes dados, além de não garantir a privacidade de acesso a determinadas informações a um contingente limitado de usuários dos equipamentos.

O *time-sharing* trouxe, portanto, a necessidade da programação mais preocupada com mecanismos de controle de compartilhamento de recursos e informações, implementados de forma a garantir um mínimo de segurança no acesso a essas informações.

Em 1967, a Advanced Research Projects Agency - ARPA (atualmente denominada Defense Advanced Research Projects Agency - DARPA), nos Estados Unidos, organizou uma "força-tarefa" para estudar e recomendar procedimentos de segurança, de modo a proteger informações compartilhadas em computadores. O resultado deste esforço foi a elaboração do documento: Security Controls for Computer Systems - Report of Defense Science Board Task Force on Computer Security [Ware 1979], notadamente o primeiro registro literário com recomendações relacionadas à segurança da informação em computadores.

A partir de então, outras agências governamentais passaram a visualizar a segurança da informação como um problema crônico, carente de normatizações e implementações em *software*, que garantissem um mínimo de controle destas informações sob o ponto de vista da segurança. Esta preocupação fez com que o Departamento de Defesa dos Estados Unidos (United States Department of Defense - DoD), com o apoio da Agência Central de Inteligência (Central Inteligency Agency) iniciassem, a partir de então, o desenvolvimento do primeiro sistema operacional voltado à segurança da informação, seguindo as normas da política de segurança do DoD, o ADEPT-50, finalizado em 1960 [Landwehr 1981].

Também merece destaque o relatório técnico escrito por James P. Anderson em outubro de 1972 [Anderson 1972], denominado: Computer Security Technology Planning Study que, utilizado conjuntamente com [Bell & Lapadula 1973], [Bell 1974] e [Bell & Lapadula 1974] formou o conjunto literário de normas conhecido como "Doctrine", servindo de base para vários documentos posteriores na área de segurança da informação.

Em 1977, outra iniciativa do DoD, denominada DoD Computer Security Initiative, desenvolveu um centro específico para avaliação da segurança em soluções computacionais, instituindo a partir de 1978 um conjunto de regras para utilização neste processo de avaliação, conhecido informalmente como The Orange Book, ou DoD 5200.28-STD : Department of Defense Trusted Computer System Evaluation Criteria [DoD 1985]. Esse trabalho foi consolidado somente em 26 de dezembro de 1985 e constituiu-se no marco inicial de um processo mundial e contínuo de busca por um conjunto de medidas, que permitem a um ambiente computacional ser qualificado como minimamente seguro.

A primeira grande iniciativa de normatização na área de segurança da informação fora dos Estados Unidos se deu em 1987, quando o Departamento de Comércio e Indústria (DTI) do Reino Unido criou um centro de segurança de informações (CCSC - Commercial Computer Security Centre) encarregado, dentre outras tarefas, de criar uma norma de segurança da informação para o Reino Unido [Watkins & Calder 2003].

A partir de 1989, vários documentos preliminares foram publicados pelo CCSC, até

que, finalmente, em 1995, surgiu a BS7799 (British Standard 7799). Essa norma foi organizada em duas partes, sendo a primeira disponibilizada publicamente em 1995 e a segunda em 1998.

A repercussão deste documento e sua robustez serviram de base para uma discussão internacional em torno de uma padronização mundial e, com algumas alterações no texto original, foi publicada em 2000 a ISO/IEC 17799. Em setembro de 2001 foi homologada pela ABNT uma versão brasileira dessa norma, denominada NBR ISO/IEC 17799.

Este conjunto de normas abrange os mais diversos tópicos na área de segurança, exigindo o cumprimento de várias recomendações para a obtenção da certificação pelas instituições.

Porém, muitas corporações privadas e estatais têm buscado esta certificação como forma de referendar seus esforços em demonstrar preocupação com a segurança das informações de seus usuários/clientes, uma tendência observada em todo o mundo.

### Objetivos da Segurança da Informação

Todo projeto de Segurança de Informações procura abranger, pelo menos, os processos mais críticos do negócio em questão [Moreira 2001].

O resultado esperado de um trabalho como este é, sem dúvida, que no mínimo todos os investimentos efetuados devam conduzir para:

* Redução da probabilidade de ocorrência de incidentes de segurança;
* Redução dos danos/perdas causados por incidentes de segurança;
* Recuperação dos dados em caso de desastre/incidente.

O objetivo da segurança, no que tange à informação, é a busca da disponibilidade, confidencialidade e integridade dos seus recursos e da própria informação [Moreira 2001].

### Disponibilidade

Os esforços da instituição em proporcionar a disponibilidade dos seus recursos, sejam eles sistemas, informações ou processos, ocorrem quando estes necessitam de acesso contínuo e ininterrupto, ou seja, a informação deve estar disponível para a pessoa certa e no momento em que ela precisar [Moreira 2001].

Portanto, a instituição deve identificar as soluções existentes voltadas ao atendimento desta necessidade.

### Integridade

Consiste em proteger a informação contra qualquer tipo de alteração, sem a autorização explícita do autor da mesma [Moreira 2001].

Quando se fala em perda de integridade, refere-se à alteração ou modificação do conteúdo ou do *status*, remoção da informação, alteração do conteúdo de um e-mail, programas, etc [Moreira 2001].

A utilização de criptografia para a proteção de dados prevê a checagem de integridade dos dados transportados, através da verificação da sua assinatura (*hash*), comparando-a com o resultado do processo antes da transmissão.

A perda de integridade pode ser intencional ou não. Independente da forma ou motivo, o maior problema na perda de integridade é o montante que a instituição vai gastar para recuperar ou reconstituir os dados, quando possível.

#### Confidencialidade

A confidencialidade é a propriedade que visa manter o sigilo, o segredo ou a privacidade das informações, evitando que pessoas, entidades ou programas não-autorizados tenham acesso às mesmas [Moreira 2001].

A perda de confidencialidade ocorre quando pessoas não-autorizadas obtêm acesso a informações confidenciais e passam a revelá-las a terceiros. Nem todas as pessoas sabem que uma vez que seu computador esteja conectado à Internet, as informações nele contidas podem perder a confidencialidade desejada.

Por esta razão, muitas instituições têm proibido o armazenamento de informações confidenciais e de valor significativo em áreas públicas acessíveis pela Internet [Moreira 2001].

A criptografia é, também neste caso, uma garantia de confidencialidade no armazenamento e tráfego de informações.

#### Não-Repúdio

O não-repúdio compreende os esforços dispendidos para garantir a autoria de determinadas ações [Moreira 2001].

O uso de certificados digitais (cartórios virtuais) tem sido o principal componente na prova da autoria do conteúdo de documentos e informações trocadas em rede, impedindo, como no cartório tradicional, a negação de autoria de um emitente ou acusação de adulteração de dados.

### 2.1.2 Evolução das Redes de Computadores

O resumo apresentado a seguir, descrevendo a evolução das redes de computadores, foi baseado no levantamento apresentado em [Kurose & Ross 2003].

#### O Início: 1961 - 1972

Dada a importância cada vez maior dos computadores no início da década de 60 e, devido ao advento dos computadores com multiprogramação (*time-sharing*), era natural considerar a questão de como interligar esses computadores, de modo que os mesmos pudessem ser compartilhados entre usuários distribuídos em diversas localizações geográficas.

A partir de então, pesquisadores de diversas nacionalidades passaram a estudar e propor modelos de comunicação entre máquinas. Porém, somente em 1967, pesquisadores da ARPA, nos Estados Unidos, desenvolveram a primeira rede de computadores por comutação de pacotes, funcionamento padrão adotado até a consolidação da Internet pública.

Os primeiros comutadores de pacotes ficaram conhecidos como IMPs (Interface Message Processors - Processadores de Mensagens de Interface), instalados a partir de 1969 na Universidade da Califórnia (UCLA), na Stanford Research Institute (SRI), na UCLA de Santa Bárbara e na Universidade de Utah.

Em 1972, a ARPAnet, como então ficou conhecido o projeto de rede, foi apresentada publicamente na International Conference on Computer Communications (Conferência Internacional sobre Comunicação por Computadores). Nesta fase, a rede já contava com 15 nós e utilizou como primeiro protocolo fim-a-fim o NCP - Network-Control Protocol (protocolo de controle de rede), normatizado pela primeira RFC publicada (RFC 001), utilizado para os testes do primeiro programa de e-mail, elaborado por Ray Tomlinson em 1972.

**O Surgimento de Outras Redes: 1972 - 1980**

Em meados da década de 70, surgiram outras redes, além da ARPAnet, como a ALOHAnet, rede de microondas interligando universidades das ilhas do Havaí; a Telenet, rede de comutação de pacotes comercial baseada na ARPAnet; as redes francesas Tymnet e a Transpac, até que em 1973, Bob Metcalfe apresentou os princípios da Ethernet, que resultou no enorme crescimento das LANs de curta distância. Em 1975, a Digital Equipment Corporation (Digital) lançou a primeira versão de sua rede DECnet, interligando dois minicomputadores PDP-11. O avanço nas pesquisas da Digital serviram de base para parte dos conceitos do conjunto de protocolos padrões desenvolvidos mais tarde sob o nome de OSI (*Open Systems Interconnection* - Interconexão de Sistemas Abertos).

Visando a interconexão das várias redes em uso até então, a DARPA, criou em 1974 uma “rede de redes” cujo projeto chamava-se “internetting”.

Pesquisas isoladas de empresas como Xerox (arquitetura XNS) e IBM (arquitetura SNA) contribuíram, junto aos esforços de até então, para alavancar o desenvolvimento e a massificação do uso das redes de computadores.

**A Massificação do Uso da Internet: 1980 - 1990**

Contrastando com as aproximadamente 200 máquinas conectadas à ARPAnet no final da década de 70, no final da década de 80 o número de máquinas ligadas à Internet pública chegava a cem mil, crescimento justificado em parte pelos esforços de interligação de universidades.

Em 01 de janeiro de 1983, adotou-se oficialmente o TCP/IP como novo padrão de protocolo de máquinas para a ARPAnet (substituindo o NCP).

Na segunda metade da década de 80, a NSF (National Science Foundation) cria a NSFNet, conecta sua rede de supercomputadores à ARPANet e passa a financiar a implantação de redes regionais, dando origem a Internet global. [Comer 1995]

**Fim da ARPAnet e Surgimento do WWW: Década de 90**

O início dos anos 90 foi marcado pelo fim da ARPAnet com a criação, em 1992, da Sociedade Internet e de várias outras organizações que ainda hoje desempenham atividades

específicas na Internet, a exemplo do IAB (Internet Architecture Board), o INTERNIC (The Internet's Network Information Center) e outros. No mesmo período deu-se início ao funcionamento da *World Wide Web*, resultado de pesquisas do CERN (European Center for Nuclear Physics - Centro Europeu para Física Nuclear), que levou a Internet para os lares e as empresas de milhões de pessoas em todo o mundo, habilitando e disponibilizando centenas de novas aplicações.

**Novas Aplicações... Novos Problemas**

A comodidade proporcionada pela disponibilização e utilização destas novas aplicações trouxe problemas novos pelo valor das informações trafegadas a partir de então.

A fragilidade do TCP/IP sob o ponto de vista da integridade, privacidade e autenticidade não constituíam um problema sério no início da operação da Internet, já que a maioria dos dados trafegados eram públicos e sem valor comercial. Além disso, o perfil dos usuários era essencialmente técnico, formado por pesquisadores e estudantes, interessados muito mais em aproveitar as facilidades proporcionadas pelos recursos de infra-estrutura de transporte de informações, do que em observar e explorar as fragilidades destas transmissões diante de um usuário mal intencionado, interessado no conteúdo de pacotes não endereçados a ele.

*Homebanking*, compras via *Web*, leilões eletrônicos, ou mesmo arquivos de pessoas conhecidas armazenadas em servidores na *Web*, atraíram para as redes de computadores estelionatários, ladrões e outras pessoas de má índole, com intenção de explorar as fragilidades dos protocolos de comunicação, para acessar e tirar proveito de informações privilegiadas.

### 2.1.3 A (in)Segurança nas Redes Corporativas

Entende-se por rede corporativa, geralmente, uma rede de área local (*Local Area Network - LAN*), ou um conjunto de LANs interconectadas que pertencem a uma dada organização.

Nesses últimos anos, uma quantidade expressiva dessas redes tem adotado a arquitetura TCP/IP como sub-sistema de comunicação, constituindo as chamadas Intranets, que estão sujeitas às mesmas vulnerabilidades da Internet.

Em 2001, o FBI estimava que o crime por computador custava U$ 10 bilhões a cada ano e que as empresas gastavam U$ 4 bilhões em sistemas de proteção às suas redes [Cronkhite & Mccullough 2001]. Com o passar dos anos, estes números tendem a crescer. As ameaças a uma rede podem ser bastante óbvias, ou podem vir disfarçadas como atividades ou ações insuspeitas. As organizações possuem dados que são, em geral, de uso privado da empresa, e não para o consumo público. Em alguns ambientes, alguns dados podem se prestar ao acesso público, enquanto em outros não. Nessas situações, há a necessidade de se criar verdadeiras barreiras para impedir o acesso não-autorizado do lado público para o lado privado [Tittel 2003].

### Características Principais das Redes Corporativas

Redes corporativas tendem a refletir a complexidade das corporações que detêm a sua propriedade: a rede corporativa de pequenas empresas em geral se resume a uma única rede local (LAN), enquanto a rede corporativa de uma empresa de porte nacional pode chegar a ser constituída por várias LANs interconectadas através de redes metropolitanas (MANs) e redes de longa distância (WANs).

Tanto as redes locais quanto os enlaces de interconexão dessas redes podem usar tecnologias cabeadas ou comunicação sem fio.

A grande maioria das LANs atualmente se baseia em tecnologia Ethernet (e suas variantes), quando adota soluções cabeadas, e tecnologia Wi-Fi (e suas variantes), quando adota soluções sem fio.

Para implementação dos enlaces de comunicação entre essas LANs, quando a rede corporativa apresenta um maior nível de abrangência geográfica, existem na atualidade um grande número de soluções, tanto cabeadas quanto sem fio. Dentre as soluções cabeadas pode-se citar desde o uso das tradicionais linhas telefônicas com modems analógicos, às modernas técnicas de transmissão através de modems xDSL, ao uso de fibras ópticas, ou dos modems de cabo e as redes híbridas fibra-cabo coaxial das operadoras de distribuição de TV paga.

No mundo da comunicação sem fio, encontram-se soluções comerciais de enlaces de rádio, com várias taxas de transmissão, serviços de distribuição como LMDS (Local Multipoint Distribution System), até o uso de tecnologias emergentes como o Wi-MAX.

Entretanto, como solução de convergência para toda essa gama de alternativas, o mercado tem indicado a presença cada vez mais definida do IP e dos outros protocolos da arquitetura Internet.

A arquitetura TCP/IP surge não só incorporada aos sistemas operacionais de rede para os ambientes de rede local, caracterizando as Intranets, mas também na interconexão dessas LANs que pode ser feita através de enlaces proprietários, mas que em vários casos usa a Internet, através de seus provedores de acesso.

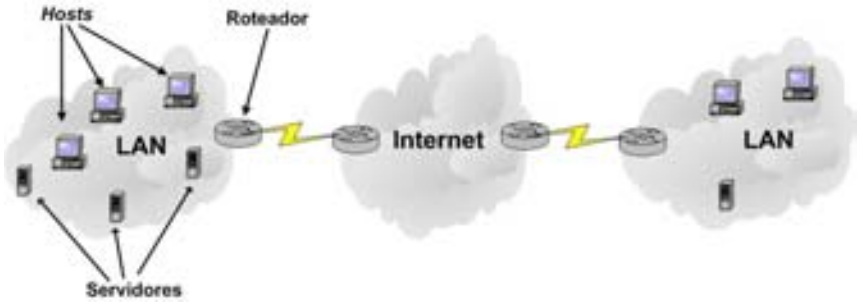


Figura 2.1: Arranjo Típico de uma Rede Corporativa (e elementos)

### Aspectos Envolvidos na Proteção de Sistemas

A segurança em uma rede envolve dois aspectos: segurança física e segurança de *software* (lógica). Os aspectos físicos incluem o controle do acesso físico ao equipamento. Uma organização precisa determinar quem pode andar em suas dependências e a quais áreas se pode ter acesso permitido. Ter acesso aos equipamentos físicos faz com que seja muito mais fácil a um indivíduo obter acesso não-autorizado a uma área, ou simplesmente roubar o equipamento para, mais tarde, fazer um escrutínio dele. Não importa quantos esquemas de proteção de *software* estejam ativos, se a segurança física for frágil ou se mostrar ausente, o *software* não pode remediar esta situação [Tittel 2003].

Quando os dados ou informações de uma instituição são eletronicamente furtados, perdidos ou modificados para sempre, existem outros fatores de custo que ocorrem depois do incidente. Se as informações necessitarem ser recriadas e/ou recolocadas no sistema, o custo associado às pessoas envolvidas nessa tarefa tem de ser considerado.

Haverá também o custo da perda ou degradação dos negócios durante o tempo em que o dados estiverem indisponíveis, principalmente se a imagem digital pública da instituição implica em parte (ou totalidade) de sua arrecadação [Tittel 2003].

O custo maior pode, ainda, estar relacionado à imagem da instituição diante do seu público alvo. A falta de credibilidade externada pela notícia de uma "invasão" e manipulação de dados de clientes, por exemplo, pode destruir completamente um negócio que até então parecia ter a solidez de uma empresa não-virtual. Todo o investimento de um banco ou um *site* de comércio eletrônico podem resultar em nada se, ao perder sua credibilidade, a instituição perder também os clientes que acessavam e adquiriam produtos ou manipulavam contas via rede.

### Perfil dos Atacantes

Existem muitos tipos diferentes de pessoas que podem ameaçar uma rede. O termo *hacker* é freqüentemente usado para descrever os indivíduos que tentam ganhar ou efetivamente chegam a obter acesso não-autorizado a uma rede ou sistema em geral. Como a denominação *hacker*, em sua origem, faz menção a um indivíduo que tenta decifrar algo para poder construir um código que funcione de acordo com suas necessidades [Tittel 2003], a mídia tentou por um curto período de tempo utilizar o termo *cracker*, como tentativa de distinguir um *hacker* que realiza atividades ilegais de um *hacker* fazendo tarefas legais. Porém, o termo *hacker* (ou ciberpirata, como também é grafado) se consolidou como referência a atividades ilegais de indivíduos que tentam ter acesso sem permissão a uma rede.

Alguns *hackers*, em suas atividades, alegam que estão experimentando ou testando sistemas, enquanto outros tentam de fato se apoderar de informações privadas, governamentais ou militares [Tittel 2003].

Outra ameaça em potencial à rede de uma instituição são os seus ex-funcionários, geralmente os que foram demitidos sob condições desfavoráveis. Tão cedo um empregado é notificado da sua demissão, todos os acessos às contas da instituição devem ser bloqueados, para que ele, se inconformado, não possa acessar o sistema e danificar ou furtar informações antes de deixar as dependências da instituição[Tittel 2003].

Porém, os principais agentes de furto de dados e vandalismo eletrônico são os próprios funcionários da instituição [Tittel 2003].

Existem várias razões para que pessoas mal intencionadas pratiquem atos de cunhos maldosos, algumas das principais são: ganhos financeiros, vingança, necessidade de aceitação ou respeito, idealismo, curiosidade ou busca de emoção, anarquia, aprendizado, ignorância e espionagem industrial [Moreira 2001].

**Passos Iniciais de um Processo Intrusivo**

Em geral todo ataque é precedido de uma verificação prévia do ambiente, seja por meio de análise de portas (serviços) em uso, sondagens usando engenharia social, identificação de versões dos programas ou busca por arquivos específicos utilizando máquinas de busca, como os disponíveis nos *sites* Yahoo!, Altavista, Cadê, Surf, dentre outros [Rufino 2002].

*Footprinting*

A coleta de informações sem o comprometimento legal, ou seja, sem infringir nenhuma lei, utilizando o resultado de procedimentos rotineiros e que não levantam suspeitas, como base para os passos seguintes de um ataque dá início ao conjunto de procedimentos que recebe o nome de *footprinting*.

A técnica mais utilizada e eficaz deste conjunto é a Engenharia Social, uma tentativa do *hacker* de conseguir que alguém o ajude em uma invasão, enganando ou confundindo essa pessoa. Geralmente isso é feito sem que as pessoas sequer saibam que estão comprometendo sua própria segurança [Hatch et al. 2002].

A Engenharia Social não requer "prática nem tão pouco habilidade", basta ter poder de convencimento e usar de psicologia comportamental. Quando bem executada é de uma eficiência surpreendente e normalmente não deixa rastros que permitam a identificação do autor [Rufino 2002].

Existem várias formas de obter informações que podem ser usadas nesse tipo de ataque: reportagens em jornais e revistas revelando o nome (e muitas vezes várias outras informações) dos responsáveis por determinada área da instituição, funcionários que revelam detalhes do ambiente computacional em listas públicas, por meio de encartes institucionais ou promocionais e apresentações em simpósios e eventos que podem conter informações sobre aspectos organizacionais, ou ainda através de perguntas bem feitas ao setor de atendimento ao cliente. Aliás, é interessante notar que quanto mais baixo o nível hierárquico do funcionário, mais ele tende a colaborar com ataques por engenharia social [Rufino 2002].

*Port Scanners*

Normalmente, a primeira ação efetuada por um atacante é a coleta de informações sobre determinado ambiente-alvo, tentando identificar a topologia, quantos e quais são os componentes da rede, tipo de sistemas operacionais dos servidores, serviços ativos

e várias outras características [Rufino 2002]. Esse procedimento inicial geralmente é efetuado utilizando-se os port scanners.

Executados contra um *host* específico, uma rede ou um conjunto de redes, os port scanners (vasculhadores de porta) identificam as portas abertas em cada máquina e as associam aos serviços normalmente utilizados nestas portas [Moreira 2001], oferecendo ao atacante informações para um possível ataque.

#### *Scanners* de Vulnerabilidades

Utilizado para a detecção de vulnerabilidades em programas que estão sendo executados em um sistema [Marcelo et al. 2000], com este tipo de scanner, além de informações sobre as portas/serviços, o atacante tem informações sobre a versão destes e pode escolher um *exploit* específico para o comprometimento do serviço.

Pode ser considerado como um sucessor do port scanner tradicional, porém com metodologia de ataque bem mais agressiva, funcionando como um verdadeiro kit de ferramentas de invasão, já que auxilia desde a descoberta dos serviços, versões de programas, vulnerabilidades conhecidas e ferramentas de ataque.

### Definição de Alvos

Depois de levantadas as informações com os scanners, o atacante parte para a análise destes resultados e o cruzamento de informações, explorando as possibilidades levantadas por cada ferramenta utilizada [Marcelo et al. 2000].

Desta forma, praticamente não mais existem ataques aleatórios voltados a serviços inexistentes em máquinas-alvo ou explorando vulnerabilidades de outras versões. A análise dos resultados dos *scanners* define para o atacante o que deve ser feito e como utilizar ferramentas específicas para uma possibilidade clara de intrusão contra máquinas vulneráveis da instituição-alvo.

### O Ataque (Explorando as Vulnerabilidades)

Com o alvo definido, o atacante escolhe a partir de então o que realmente deseja da máquina alvo, se apenas a inoperabilidade dos sistemas em execução, ou o seu comprometimento efetivo. Os serviços descobertos, as vulnerabilidades identificadas previamente e as ferramentas utilizadas determinam o grau de comprometimento da(s) máquina(s) alvo.

A seguir, são descritos os principais tipos de ataque e as principais ferramentas de *software* usadas.

#### *Exploits*

Os exploits são ferramentas de *software* prontas para a realização efetiva de um ataque. Geralmente após a descoberta de uma nova vulnerabilidade em uma versão específica de um serviço, vários *hackers* (alguns por ideologia outros com objetivo malicioso) passam a codificar programas específicos para a exploração daquela vulnerabilidade.

As linguagens preferidas para o desenvolvimento deste tipo de ferramenta são o C, o Assembler, além de *shell scripts* e *scripts* Perl.

a) *Exploits* baseados em Negação de Serviço (DoS/DDoS - *Denial of Service/Distributed Denial of Service*)

Os ataques de Negação de Serviços fazem com que recursos sejam explorados de maneira agressiva, de modo que usuários legítimos ficam impossibilitados de usá-los [Nakamura & Geus 2002]. Os *exploits* para ataques de DoS são geralmente utilizados para interromper o tráfego para *sites* de comércio eletrônico ou, simplesmente, para tornar os serviços de uma determinada máquina conectada à rede indisponíveis enquanto durar o ataque [Hatch et al. 2002].

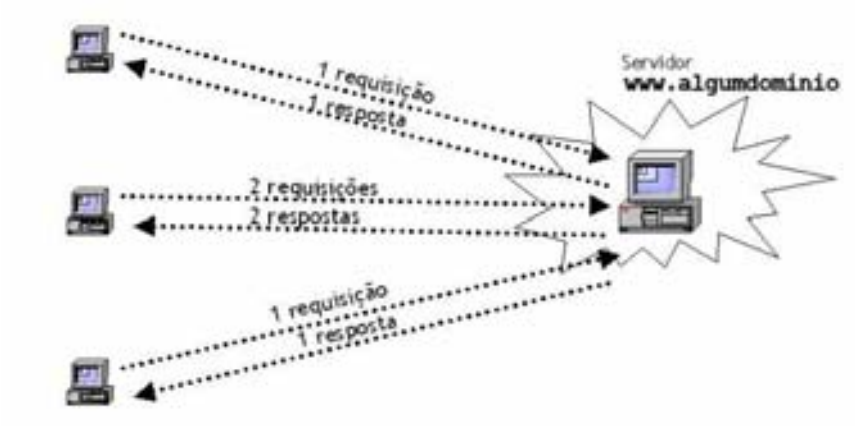


Figura 2.2: Tráfego Normal de Conexões Cliente/Servidor

A inundação (*flood*) é uma das formas mais antigas de ataque de DoS na Internet. Para criar uma inundação na rede, ao invés de um fluxo normal de pacotes (Figura 2.2), o invasor envia um fluxo rápido e excessivo de pacotes IP a um *host*, ocupando sua largura de banda na rede e atrapalhando o tráfego legítimo (Figura 2.3).

Esse tipo de ataque também pode causar um desempenho lento para os usuários locais da máquina-alvo, já que esta tenta processar todos os pacotes que recebe, consumindo tempo da CPU que seria destinado a outros aplicativos. As inundações são mais efetivas quando o *host* atacante tem mais largura de banda para a Internet que a máquina-alvo, assim, o atacante pode enviar mais dados do que a rede alvo pode manipular, não deixando recursos para outros tipos de tráfego [Hatch et al. 2002].

Em ataques de DoS distribuída (DDoS) ao invés de um *host* atacante, até centenas de milhares de *hosts* são utilizados simultaneamente, de forma sincronizada, gerando tráfego excessivo contra uma máquina-alvo (Figura 2.4).

Um ataque DDoS baseia-se na instalação de agentes remotos em muitas máquinas, localizadas em várias partes da Internet, e direcionamento de uma inundação amplificada

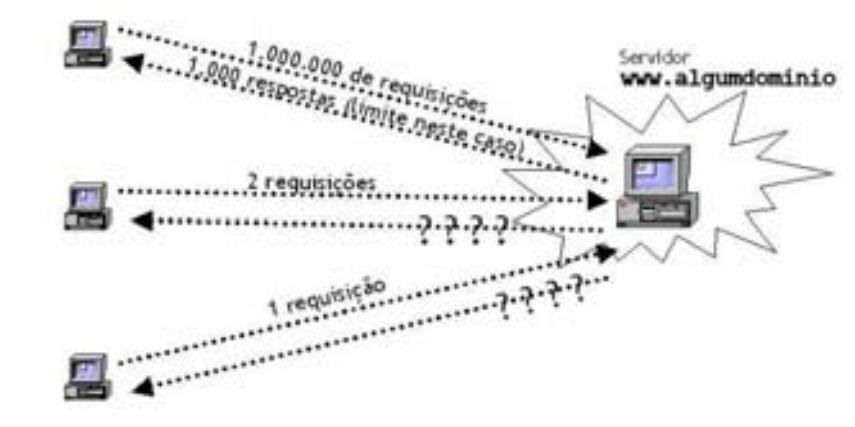


Figura 2.3: Tráfego de um Ataque DoS em Execução

contra o alvo, geralmente isolando não só a máquina-alvo, como seu provedor e até o *backbone* ao qual o provedor está conectado.

No primeiro trimestre de 2000, as principais empresas *on-line*, incluindo CNN Interactive, Amazon, Yahoo!, Excite e eBay, experimentaram ataques distribuídos de negação de serviço [Cronkhite & Mccullough 2001].

Para a instalação dos agentes, o atacante previamente invade vários sistemas e instala rotinas coordenadoras e agentes de inundação controlados remotamente. Em alguns casos, worms são utilizados para esta infecção, instalando-os em máquinas vulneráveis. Uma vez instalados os controladores e agentes de inundação, o atacante se conecta a um controlador e executa os comandos que farão com que os agentes de inundação direcionem um conjunto de ataques a uma máquina-alvo.

b) *Exploits* Baseados em *Buffer Overflow*

O *buffer overflow* é o método de ataque mais empregado, segundo os boletins do CERT, desde 1997. De acordo com o CERT, no período até 2002, mais da metade dos boletins eram relativos a ataques de *buffer overflow* [Nakamura & Geus 2002].

Os *exploits* de estouro (*overflow*) de *buffer* são criados quando os desenvolvedores usam técnicas impróprias de codificação para executar alguma operação em um programa. Funções padrão de strings do C como strcat(), strcpy(), sprintf(), vsprintf(), scanf() e gets() têm sido reportadas como as maiores responsáveis por problemas de *buffer overflow*. Nenhuma dessas funções verifica o tamanho de seus argumentos antes de executar as operações. Isso leva a uma vulnerabilidade que pode ser explorada para se atacar um sistema.

Os *buffer overflows* são causados por programas “errados”. Quando um *hacker* utiliza um *exploit* desse tipo, está simplesmente preenchendo uma variável ou um *buffer* do programa com excesso de informações. Nem todas as variáveis são boas opções para um ataque desse tipo - uma variável adequada deve ser local e estar armazenada na pilha do processador [Hatch et al. 2002].

Essa pilha controla a troca de dados entre os programas e também informa ao com-

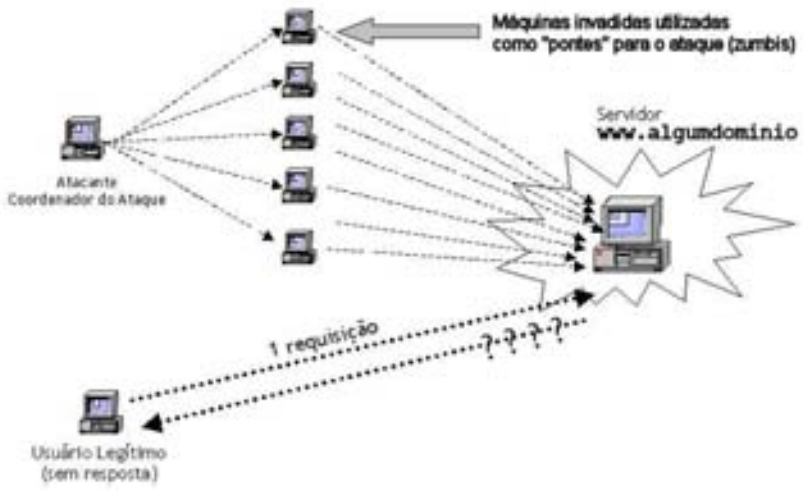


Figura 2.4: Ataque DDoS em Execução

putador que código executar, quando uma parte de um programa (ou função) completou sua tarefa. As variáveis locais também são armazenadas na pilha. Estourar o tamanho de uma pilha, corresponde a colocar em uma variável local instruções em quantidade suficientemente grande, para sobrescrever o endereço de retorno e para que o mesmo passe a apontar para uma nova instrução também inserida pelo atacante. O tipo de instrução definido na pilha controlará como o *buffer overflow* se comportará no sistema. Pode-se fazer com que um programa de *shell* seja executado, fornecendo, por exemplo, acesso interativo ao sistema ou fazendo com que um outro aplicativo seja iniciado. Pode-se até mesmo executar alterações em um arquivo de configuração de um serviço específico e fazer com que um novo serviço seja iniciado [Hatch et al. 2002].

*Defacements* **(Pixação de Páginas** *Web***)**

Para a realização deste tipo de ataque, a máquina-alvo tem que estar comprometida, ou seja, o atacante deve ter acesso efetivo aos arquivos da máquina. São ataques que, em sua grande maioria, exploram vulnerabilidades de servidores *Web*, embora o comprometimento possa se dar a partir de outros serviços.

Não existe aqui o desejo do atacante de utilizar a máquina como ponte para outros ataques ou de retornar em outra oportunidade à máquina, já que o resultado visual (a desfiguração do *site* da instituição) é geralmente notada pouco tempo depois de consolidado o ataque e as medidas de correção do problema e republicação das informações quase sempre são imediatas.

É um ataque que não demonstra perícia do atacante, que geralmente utiliza ferramentas prontas de exploração a servidores *Web*, sendo bastante realizado por adolescentes que deixam mensagens para “turmas” como conteúdo do *site* desfigurado.

É um dos ataques mais reportados tendo como autoria atacantes brasileiros.

*Sniffers*

Os *sniffers* (farejadores) são ferramentas comuns utilizadas pelos *hackers* para obter acesso a sistemas e captura de nomes de usuários e senhas de outros sistemas [Hatch et al. 2002].

Os principais alvos dos *sniffers* são os protocolos que não utilizam criptografia e necessitam de identificação (nome de usuário/senha) dos usuários para o acesso. Tanto o tráfego sem encriptação pode ser capturado e tratado de forma a exibir todo o conteúdo transportado, como, durante os processos de identificação, os nomes de usuários e suas senhas também podem ser capturados e poderão ser utilizados posteriormente para conexões aparentemente legítimas pelos atacantes.

Protocolos como Telnet (conexão remota em modo texto), POP3 e IMAP (ambos protocolos de recebimento de e-mails) e FTP (transferência de arquivos) exigem autenticação inicial, que pode ser capturada sem problemas por um sniffer implantado em uma intranet (LAN), em um roteador entre a vítima e o servidor ou mesmo na máquina da própria vítima caso esteja previamente instalado através de um cavalo-de-tróia, por exemplo.

Os *sniffers* funcionam na captura de dados quando esses trafegam pela rede. Sob condições normais de operação da rede, os dados são encapsulados em quadros para que a rede local (LAN) os conduza entre as máquinas.

Conforme já foi dito anteriormente, um grande número das redes locais atuais se utiliza, para transmissão de quadros, de protocolos como o Ethernet ou o Wi-Fi e suas variações. Esses protocolos utilizam um identificador, chamado de endereço MAC, para designar cada uma das máquinas da LAN.

Todas as placas de interface de rede (NIC) e os dispositivos de rede têm um endereço MAC exclusivo que é atribuído pelo fabricante da placa ou do equipamento. A maioria dessas NICs não permitem que o endereço MAC seja alterado [Hatch et al. 2002]. Numa transmissão *unicast*, cada quadro é destinado a um endereço MAC em particular. Durante a recepção de quadros, as NICs dos equipamentos nessa LAN examinarão os endereços MAC de destino dos quadros recebidos. Se o endereço MAC corresponder ao endereço da NIC, ela os lerá integralmente, os processará e passará o dado encapsulado no quadro (o pacote IP, muito provavelmente) para o protocolo superior da pilha de protocolos de comunicação, de modo a dar prosseguimento ao processo de recepção desse dado até o nível de aplicação. Além disso, se o endereço MAC de destino do quadro for o de broadcast, todos os equipamentos na LAN o lerão e processarão os dados. Caso contrário, o sistema lerá somente o endereço MAC de destino e ignorará a parte dos dados encapsulados do quadro.

Os *sniffers* funcionam pela colocação da NIC em um estado chamado de modo promíscuo. Quando a NIC estiver no modo promíscuo, ela lerá o conteúdo de todos os quadros independente do endereço MAC de destino. Portanto, um sniffer poderá, desse modo, examinar a parte de dados de qualquer quadro e selecionar as informações de interesse. Essas podem incluir as informações dos cabeçalhos dos protocolos, que revelam dados sobre as portas de serviço usadas e outros dados de controle ou informações como os nomes de usuário e senhas [Hatch et al. 2002].

*Backdoors*

Uma das maiores preocupações dos atacantes é a possibilidade de perda de controle de uma máquina já comprometida [Hatch et al. 2002]. Para evitar isso, geralmente, após o comprometimento da máquina, o atacante tenta instalar *backdoors* (portas dos fundos) nessa máquina, de forma a lhe assegurar o acesso permanente, mesmo se a vulnerabilidade explorada na primeira intrusão seja corrigida ou se o administrador bloquear o acesso à porta utilizada pela entrada original.

Comumente, as portas utilizadas para instalação de *backdoors* são portas não-privilegiadas, ou seja, portas maiores que 1024 (e menores que 65535, evidentemente, por ser esse o valor superior para um número de porta).

Outra forma de utilização das *backdoors* é o envio de programas camuflados para execução pela vítima, na forma de cavalos-de-tróia, ou automaticamente e aleatoriamente através de *worms*.

**Cavalos-de-Tróia (*trojans*)**

A lenda diz que os gregos derrotaram o exército de Tróia construindo um enorme cavalo de madeira, que encheram de soldados. Os habitantes de Tróia, acreditando que o cavalo tinha sido enviado pelos deuses, levaram-no para dentro da cidade e, naquela noite, os soldados gregos surgiram repentinamente de dentro dele, brandindo suas espadas. Agora, milhares de anos depois, esse truque foi ressuscitado [Hatch et al. 2002].

Os cavalos-de-tróia da era do computador são programas projetados para enganar a segurança das máquinas dos usuários, porém disfarçados como algo benigno. Como a invenção grega, um cavalo-de-tróia de computador não pode fazer nada por contra própria, mas precisa contar com a ajuda do usuário para cumprir seu destino [Hatch et al. 2002].

Não deve-se considerar os cavalos-de-tróia como vírus nativos, pois não se duplicam e tampouco contaminam outros programas. Trata-se de programas autônomos, camuflados em arquivos, que podem ficar inativos na máquina-alvo por muito tempo e só trabalharem em condições específicas [Moreira 2001].

Assim, a diferença entre um vírus e um cavalo-de-tróia é que o vírus se auto-reproduz, enquanto o cavalo-de-tróia não, além de precisar ser executado para se instalar na máquina-alvo [Moreira 2001].

Há três usos principais da expressão cavalo-de-tróia na linguagem da computação moderna [Hatch et al. 2002]:

a)Programa cavalo-de-tróia: Um programa pernicioso que se mascara de algo benigno, enganando secretamente sua segurança. Esse é o uso mais comum para a expressão.

b)Código-fonte com cavalo-de-tróia: Uma cópia do código-fonte do programa que foi alterado para conter alguma brecha na segurança ou uma porta dos fundos.

c)Arquivos binários com cavalo-de-tróia: Depois de uma invasão, um *hacker* pode substituir os arquivos binários do sistema, por versões que contenham portas dos fundos ou ocultem suas atividades.

Freqüentemente, os cavalos-de-tróia são oferecidos na forma de jogos, proteções de

tela e outros itens de interesse e são transferidos de uma pessoa para outra voluntariamente. Processar qualquer tipo de arquivo executável é sempre um risco.

*Keyloggers*

Geralmente instalados em máquinas-alvo por vírus/*worms* ou por cavalos-de-tróia, os *keyloggers* são ferramentas de ataque que capturam dados digitados na máquina e os enviam ao atacante, ou os mantém armazenados em arquivos ocultos, podendo ser recuperados pelo atacante posteriormente.

Os *keyloggers* podem capturar, inclusive, dados que são transmitidos utilizando criptografia, já que a coleta das teclas digitadas é feita antes da encriptação dos dados para envio.

Senhas, comandos, acessos a outras máquinas, tudo pode ser capturado pelos *keyloggers*, uma vez instalados na máquina.

Apesar de ser uma ferramenta tipicamente baseada em *software*, existem modelos físicos de keyloggers, ou seja, componentes de *hardware* específicos, colocados intencionalmente entre o teclado de máquinas-alvo e sua conexão à interface específica na máquina. Neste caso, o atacante tem que ter acesso físico à máquina-alvo para instalação e posterior remoção e coleta dos dados.

*Log Cleaners*

Dependendo do sistema operacional da máquina-alvo, muitas pistas sobre o atacante podem ser registradas, a exemplo de: origem do ataque, data/hora, duração, ferramentas instaladas e executadas na máquina-alvo, comandos utilizados, dentre outros. Essas informações ficam geralmente armazenadas em arquivos de *log*, mantidos pelo sistema operacional e pelos serviços instalados.

É uma preocupação para a maioria dos atacantes “encobrir os rastros” deixados na máquina atacada, de forma a não oferecer evidências que auxiliem na sua identificação como autor do ataque.

Com o conhecimento do sistema e dos seus métodos de registro, estes passos podem ser feitos após a consolidação do ataque, removendo as entradas nos *logs* referentes aos procedimentos executados pelo invasor. Porém, isto pode levar algum tempo e algo pode ser esquecido.

Para resolver este problema, geralmente, os atacantes utilizam ferramentas automatizadas de remoção de “rastros”, conhecidas como *log cleaners*.

Os *log cleaners* são escolhidos conforme o tipo e versão dos sistemas operacionais e mecanismos de registro de *logs* das máquinas-alvo.

*Rootkits*

Um atacante geralmente tem à sua disposição várias ferramentas para utilizar no procedimento intrusivo. A escolha de ferramentas de ataque específicas, arquivos a alterar, locais para a instalação de cavalos-de-tróia, dentre outras, geralmente leva tempo, o que

pode implicar em um tempo de permanência excessivo antes do comprometimento completo da máquina e, conseqüentemente, na possibilidade de descoberta do ataque antes da consumação do seu objetivo inicial.

Grande parte dos atacantes, principalmente os com menor experiência (*script kiddies*) normalmente não dedica a atenção necessária ou mesmo não possui habilidades suficientes de codificação de ferramentas, para a instalação correta de mecanismos de comprometimento do sistema-alvo. Para facilitar e agilizar esses procedimentos, existem os *rootkits*, conjuntos de ferramentas prontas para procedimentos intrusivos.

Um *rootkit* é simplesmente um conjunto de programas binários com cavalos-de-tróia, pré-empacotados, prontos para serem instalados rapidamente [Hatch et al. 2002]. Nem sempre incluem módulos carregáveis, já que isto dependeria mais do *kernel* do sistema operacional e precisariam ser compilados em cada máquina.

A maioria dos *rootkits* contém, ainda, um sniffer para vasculhar senhas na rede local, usadas em aplicativos sem criptografia. A alteração de binários para encoberta de passos, além de instalação de *backdoors* também são tarefas comuns à maioria dos *rootkits*.

*Password Crackers*

Este tipo de ataque baseia-se na tentativa de descobrir senhas para conseguir acesso a um computador (geralmente um servidor de aplicações). Trata-se da automatização de um ataque manual, chamado de “força bruta”, onde o atacante tenta exaustivamente descobrir senhas diante de um terminal do servidor remoto. Esta estratégia automatizada baseia-se na seleção de palavras comuns em um dicionário (por isso, também chamado de ’ataque do dicionário’), conforme mostrado na Figura 2.5, ou de padrões freqüentemente usados (senhas fáceis comumente utilizadas) [Hatch et al. 2002].


Figura 2.5: Ataque do Dicionário

A primeira etapa deste ataque é a captura do arquivo de senhas da máquina-alvo, disponível, geralmente, para todos os usuários legítimos do servidor, embora com o campo “senha” encriptado. De posse deste arquivo, o ataque pode ser feito na máquina do próprio atacante, baseando-se na comparação de sugestões do próprio atacante (ou uso do

dicionário), aplicação do algoritmo (público) utilizado normalmente pelo alvo para encriptação da senha e, finalmente, a comparação do resultado com a senha encriptada no arquivo de senhas.

Esse ataque geralmente consome muito tempo e, por isso, os programas automatizados (*password crackers*) são deixados em execução por horas e até dias, até conseguir uma resposta satisfatória ao atacante (nomes de usuário/senhas do sistema remoto).

Estas ferramentas têm níveis diferentes de "agressividade", ou seja, maneiras variadas de lançar tentativas, conseguindo "quebrar" senhas visivelmente complicadas, através de técnicas como utilização de palavras em ordem inversa, mistura aleatória de letras e números e uso de dicionários de várias línguas.

Embora as versões mais recentes dos sistemas operacionais, diante do grande número de ataques desse tipo, tenham tornado não-públicas as informações sobre os arquivos de senhas encriptadas, ainda é grande o número de servidores na Internet que utilizam o modo de funcionamento tradicional, ou seja, um arquivo de senhas público.

### Outros Procedimentos Utilizados em Ataques contra Redes Corporativas

Embora os procedimentos comuns da maioria dos ataques utilizem integral ou parcialmente as técnicas descritas anteriormente, existem ainda outros procedimentos utilizados em ataques, como complemento a estas técnicas ou de forma isolada.

#### *IP Spoofing*

A pilha de protocolos TCP/IP permite que a identificação do endereço de origem nos datagramas, ou seja, a informação de onde partiu a conexão, seja alterada para qualquer endereço válido (utilizando os 32 bits reservados para esta informação).

Assim, atacantes que desejam encobrir a origem de ataques podem trocar a informação de endereço de origem para um outro endereço IP qualquer. Esta técnica é conhecida como *IP Spoofing*.

A limitação, porém, para a utilização desta técnica é relativamente óbvia. Se o endereço de origem não é mais o da máquina atacante, a resposta ao ataque não vai ser enviada para esta máquina, e sim para o endereço informado como nova origem.

Por isso, este tipo de técnica geralmente é utilizada conjuntamente com ferramentas de ataque de negação de serviço, onde o objetivo é simplesmente inundar o destino com requisições de serviço, sem a necessidade de retorno à máquina atacante de respostas produzidas pela máquina-alvo.

#### *Trusted Hosts* **(Máquinas Confiáveis)**

Atualmente, é comum que as redes usem endereços de nós como uma prova de identificação totalmente confiável. Entretanto, os serviços que aceitam ou recusam as conexões com base no endereço IP do cliente não são eficazes, se um atacante puder usar um endereço confiável. O ataque de *trusted hosts* baseia-se na relação de confiança entre máquinas que utilizam, por exemplo, conexão sem senha entre si, desde que os endereços sejam reportados como confiáveis.

Sabendo disto, o atacante utiliza técnicas para "enganar" a máquina-alvo, fazendo-se passar por uma máquina com endereço IP de um *host* confiável e conseguindo, após isso, acesso privilegiado à máquina-alvo sem a necessidade de outros mecanismos de autenticação [Hatch et al. 2002].

**Vírus**/*Worms*

Os vírus são semelhantes aos cavalos-de-tróia no fato de que fazem algo a/ou na máquina infectada de um usuário, sem o seu conhecimento ou permissão. Um vírus, uma vez ativado, infectará outros programas ou arquivos no computador da vítima com cópias de si mesmo, enquanto um cavalo-de-tróia é simplesmente um programa autônomo que não pode propagar a si próprio. Tanto os vírus como os cavalos-de-tróia não podem infectar máquinas externas sem a ajuda do usuário da máquina [Hatch et al. 2002].

Um *worm* (verme) é um programa que pode infectar tanto a máquina local quanto as remotas. Geralmente se espalha de uma máquina para outra através de uma rede, atacando ou usando outros programas da vítima, ou empregando os recursos de compartilhamento de arquivos do computador remoto. Em outras palavras, o *worm* se propaga automaticamente, enquanto um cavalo-de-tróia precisa usar artifícios como sugestão de *downloads* e apelo para sua execução por parte da vítima. Assim, os worms têm um potencial muito maior para danificar as máquinas, já que não necessitam da ingenuidade dos usuários.

Existem, porém, programas híbridos (cavalo-de-tróia/vírus/*worm*) montados de forma a se disseminarem com o poder dos *worms*, infectarem a máquina da vítima com a abrangência de uma infecção por vírus e execução de cavalos-de-tróia para tarefas específicas.

Métodos de propagação dos vírus/*worms*:

a)Arquivos infectados: Um vírus pode infectar outros arquivos - por exemplo, os documentos de um processador de textos - que, por sua vez, infectam novos usuários que recebem documentos gerados por este editor.

b)Serviços de compartilhamento de arquivos: Um verme pode se beneficiar de servidores de arquivos disponíveis para infectar os arquivos contidos neles. Quando os usuários abrem esses arquivos, também são infectados.

c)Disquetes/CD-ROMs e outras mídias removíveis: Discos infectados contaminarão qualquer máquina nas quais forem inseridos.

d)Mensagens de correio eletrônico (e-mail): Um vírus pode explorar falhas em um programa de correio eletrônico da vítima e enviar a si mesmo para todos os contatos da agenda do usuário. Desse modo, a identificação do remetente (e-mail e endereço IP da máquina) como uma pessoa confiável, aumenta a probabilidade que a nova vítima aceite, abra e/ou execute anexos, infectando também a sua máquina.

*Spywares*

Os *spywares* podem ser classificados como uma nova modalidade de vírus, que infectam máquinas a partir de acessos por parte destas a determinados *sites*, e coletam informações sobre essa máquina e sobre os acessos realizados pelos usuários, de modo a traçar perfis de uso e enviá-los ao atacante.

Ferramentas maliciosas semelhantes recebem o nome de *Adwares*, sem diferenciação consistente, sob o ponto de vista de métodos de infecção e fucionamento, do *spyware*, que justifique a sua catalogação como um método de ataque diferente.

*Phishing Scam*

Conhecido como fraude "pega-bobos", o *phishing scam* é uma mistura de engenharia social e keyloggers específica para coleta de dados privativos de usuários domésticos.

O volume de dinheiro já removido de contas de usuários fraudados em um curto espaço de tempo é tão considerável, que soluções corporativas têm sido propostas para o combate à disseminação dos agentes deste tipo de fraude virtual.

O ataque consiste no envio de e-mails ao maior número possível de usuários com algo atrativo, que o leve a "clicar" em um *link* que tanto pode levar o usuário a uma página falsa montada para coleta de dados, ou induzí-lo a baixar e executar um programa (*keylogger* oculto).

A partir de então, o *software* malicioso instalado passa a monitorar o que é digitado pelo usuário em busca de informações como senhas, números de conta bancária e informações pessoais, enviando-os ao atacante remoto.

Nem mesmo a robustez de soluções como os "teclados virtuais" implementados nos *homebankings* são barreiras para alguns destes ataques, que podem capturar imagens da tela no momento dos "cliques" nos teclados virtuais.

*Pharming Scam*

Este tipo de ataque visa alterar a rota dos pacotes das máquinas-alvo destinados a determinados *sites* (principalmente de *homebanking* e comércio eletrônico), enviando-os a *sites* falsos montados com base nos legítimos.

Geralmente, esse desvio de rota é feito inserindo-se entradas no arquivo "hosts" do usuário, arquivo este que é lido antes de consultas a servidores de nomes (DNS) em busca do endereço IP, correspondente a uma determinada URL. Com a informação neste arquivo de que o *site* de um banco, por exemplo, tem como endereço IP a máquina que hospeda o *site* falso, o usuário é roteado de forma transparente para a referida máquina.

Também já foram reportados ataques *pharming scam* em que os desvios de rotas eram feitos em roteadores entre o usuário e o *host* destino. Porém, este tipo de ataque pressupõe o comprometimento dos roteadores para a alteração da tabela de rotas, ataque geralmente de difícil execução.

### 2.1.4 Procedimentos de Segurança para Redes Corporativas

A diminuição na ocorrência de incidentes de segurança em redes corporativas pode ser fruto direto de algumas medidas periódicas extremamente importantes, porém dificilmente efetuadas pelos administradores destas redes.

**Atualização de** *Software*

É a principal recomendação encontrada em qualquer literatura, palestra e curso na área de segurança da informação.

Praticamente todos os ataques que conseguem êxito contra máquinas em uma rede corporativa são fruto da exploração de vulnerabilidades em seus serviços ou mesmo em seu sistema operacional, cuja correção já estaria disponível em versões mais novas desses programas, mas que, porém, não chegam a ser instaladas.

*Patches*, *hotfixes* e *service packs* devem ser aplicados periodicamente, tão logo sejam disponibilizados, principalmente em máquinas servidoras de aplicação com acesso externo, de modo a minimizar as possibilidades de ataques exitosos contra suas informações.

**Testes de Penetração (Sondagens Internas)**

Os testes de penetração contra a própria rede se justificam como medida preventiva, ou seja, a possível identificação (e correção) de vulnerabilidades, antes que possam ser exploradas por atacantes.

É recomendável a realização periódica deste tipo de atividade por uma equipe interna preparada para tal. Se isso não for possível, deve-se apelar para empresas especializadas no assunto.

Não é recomendável que se contratem *hackers* para a realização dos testes de penetração, já que se desconhecem os princípios éticos e morais destas pessoas, que podem não se adequar aos princípios da instituição [Moreira 2001].

Uma vez escolhida a instituição que irá efetuar os testes, é primordial que seja celebrado um contrato onde, entre outras, deve conter algumas considerações [Moreira 2001]:

* As informações obtidas, bem como as vulnerabilidades, não podem de forma alguma ser divulgadas;

* Todas as senhas, arquivos importantes e confidenciais obtidos devem ser entregues à instituição contratante;

* Todos os testes que utilizarem ferramentas devem, de forma antecipada, ser previamente acordados;

* É de fundamental importância que o responsável pelo teste não saiba da estrutura de proteção que a instituição utiliza atualmente, já que em uma situação normal de ataque este tipo de informação não é público;

* Todos os testes devem caminhar para que sejam identificadas vulnerabilidades do ambiente computacional investigado, isso quer dizer que os testes não devem de forma alguma violar a privacidade e os direitos individuais;

* Quanto ao horário mais adequado para a realização dos testes, deve-se observar os horários de menor atividade;

* Ao final dos testes, a instituição contratada deve entregar um relatório com as recomendações detalhadas relativas a todas as vulnerabilidades encontradas.

*Peopleware*

A educação e conscientização dos usuários de máquinas em uma rede corporativa constituem um grande passo em busca de uma rede minimamente segura.

Uso de senhas robustas, não divulgação de informações privilegiadas em fóruns não apropriados, submissão à política de utilização de e-mails, acesso Web e demais serviços da instituição, dentre outras medidas comportamentais, quando adotadas irrestritamente pelos usuários minimizam as possibilidades de ataques como engenharia social, *password crackers* (ataque do dicionário), phishing scam, além de vírus, *worms* e *spywares*.

**Definição e Implantação de uma Política de Segurança**

Em [Moreira 2001] encontra-se um conjunto de conceitos e definições relacionados à política de segurança de uma organização, que são apresentados a seguir de forma resumida.

A Política de Segurança pode ser entendida como sendo um conjunto de normas e diretrizes destinadas a proteção dos ativos da Organização.

No documento que estabelece essa política de segurança deve estar descrito a forma que a instituição deseja que seus ativos sejam:

* Protegidos
* Manuseados
* Tratados

O objetivo de qualquer política de segurança é o de definir as expectativas da instituição quanto ao uso de seus recursos (computadores e rede), estabelecendo procedimentos com o intuito de prevenir e responder a incidentes relativos à segurança.

Dentre outros aspectos, uma política de segurança deve:

* Ser flexível com relação às mudanças necessárias;
* Ser simples na comunicação;
* Ser objetiva e curta;
* Conter regras simples;
* Ser consistente, de acordo com as outras políticas da corporação;
* Ser aplicável utilizando os equipamentos e tecnologias de rede existentes;
* Estar de acordo com as leis locais, estaduais e federais;
* Ser facilmente acessível a todos os membros da instituição;
* Definir um conjunto claro de metas de segurança;
* Definir com precisão cada um dos assuntos discutidos na política;
* Mostrar claramente a posição da instituição sobre cada ponto;
* Descrever a justificativa da política, independente dos assuntos secundários;
* Definir sobre que circunstâncias determinado item é aplicável;
* Definir as regras e as responsabilidades dos membros da organização com respeito a cada uma das diretivas definidas;
* Descrever ou associar as conseqüências do não-cumprimento da política descrita, preferencialmente com punições já existentes na instituição (pode-se basear na CLT);
* Indicar informações para contato, mais detalhes e esclarecimentos de qualquer uma das diretivas;

* Definir as expectativas de privacidade dos usuários;

* Incluir a responsabilidade sobre a definição de temas não especificamente definidos para a resolução de impasses.

**CSIRTs - Grupos de Resposta a Incidentes de Segurança**

Detectar incidentes de segurança contra redes de computadores, embora não seja uma tarefa complicada, não é também algo tão simples. Muitos atacantes não deixam rastros, e os que deixam, muitas vezes são sutis [Moreira 2001].

Analisar de forma consciente e responsável os registros de acesso, identificando, catalogando e reportando incidentes de segurança exige dedicação e formação.

Os CSIRTs (*Computer Security Incident Response Team*) são grupos treinados e qualificados para lidar de forma apropriada com estes tipos de incidentes, coletando evidências e reportando com brevidade e consistência a ocorrência desses incidentes aos responsáveis pelas redes em que estão as máquinas atacantes e aos grupos de segurança dos *backbones* aos quais estão conectadas as redes envolvidas no incidente.

### 2.1.5 Componentes de *Hardware* e *Software* Dedicados à Segurança de Redes

Além das ferramentas de sondagens internas e soluções específicas de identificação para o combate a incidentes de segurança, alguns componentes têm se consolidado como vitais para a manutenção de uma rede minimamente segura. São componentes baseados em *hardware* e *software* dedicados, sempre que possível, exclusivamente à identificação e combate a estes incidentes.

*Firewalls*

**Visão Geral**

*Firewall* é um dispositivo de rede, podendo ser um computador dedicado, um roteador ou um *software* executando em uma máquina de uso geral, que filtra o acesso a uma rede de computadores, protegendo a rede corporativa contra acessos não-autorizados e permitindo que pessoas autorizadas tenham acesso aos serviços da Internet a partir da rede interna da instituição [Moreira 2001]. O *firewall* é uma barreira inteligente entre a rede local da corporação e a Internet, através da qual só passa tráfego autorizado [(pseudônimo do autor) Marcio 2000].

O motivo principal da instalação de *firewalls* é o controle de acesso em nível de *kernel* [Hatch et al. 2002], realizando a filtragem antes, durante e/ou após o processo de roteamento dos pacotes.

Uma alternativa para selecionar o tráfego que deve ter acesso autorizado pelo *firewall* é o uso da regra: “o que não for expressamente permitido é proibido” [Moreira 2001]. Esta norma tem sido utilizada quase que pela totalidade das redes corporativas, devido à quantidade de portas e serviços a monitorar e com o aumento dos problemas de segurança em redes de computadores. A implementação da norma contrária, “o que não for

expressamente proibido é permitido", demanda muitos esforços no bloqueio de tráfego indevido e, conseqüentemente, pode implicar em sobrecarga do *firewall* pelo número de regras restritivas.

Como resultado da aplicação dessas normas, pode-se dizer que alguns *firewalls* dão maior ênfase ao bloqueio de tráfego, enquanto outros enfatizam a permissão do tráfego. O importante, porém, é configurar o *firewall* de acordo com a política de segurança da instituição que o utiliza, estabelecendo o tipo de acesso que deve ser permitido ou negado [Moreira 2001].

**Evolução de** *Firewalls*

a) Primeira Geração - Filtros de Pacotes

Esta geração inicial de *firewalls* somente controlava a origem e o destino dos pacotes de mensagens da Internet [(pseudônimo do autor) Marcio 2000]. O papel do *firewall* era o de assumir as regras de filtragem dos roteadores (ACLs), de modo a aliviar o volume de processamento nesses roteadores, isentando-os da responsabilidade pela análise e bloqueio de determinados pacotes.

A utilização de *firewalls* desta geração também se justificava em função das limitações encontradas no uso de ACLs em roteadores: interface de configuração pouco amigável, impossibilidade de registro local de logs de acesso/bloqueio, além de questões administrativas envolvendo interesses distintos entre corporações. No cenário mostrado na Figura 2.6, um roteador serve a duas redes com administradores diferentes e, conseqüentemente, o acesso às regras do roteador implicaria em compartilhamento da sua senha de root. Caso esse acesso não fosse possível, o administrador em questão não poderia inserir regras de filtragem específicas para sua rede.

Uma alternativa a esse cenário seria uso de dois *firewalls* (entre as redes internas e o roteador) sob responsabilidade da administração local de cada uma destas redes. Essa solução, além de reduzir a carga de processamento do roteador, tornaria mais seguro e controlado o acesso ao equipamento de segurança e permitiria a inserção de regras específicas para cada rede no respectivo *firewall* local (Figura 2.7).

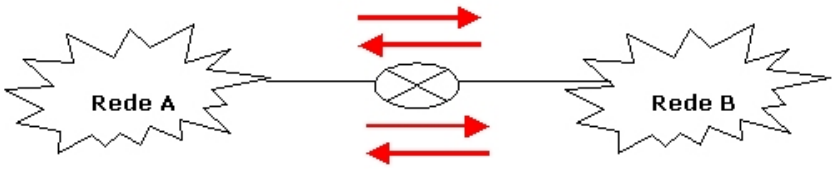


Figura 2.6: Conexão de redes com roteador e sem *firewalls*

b) Segunda Geração - Incorporação de NAT (Network Address Translation)

A segunda geração dos *firewalls* se deu com a incorporação de uma técnica de conversão de endereços (NAT) à máquina do *firewall*. Na implementação de alguns sistemas

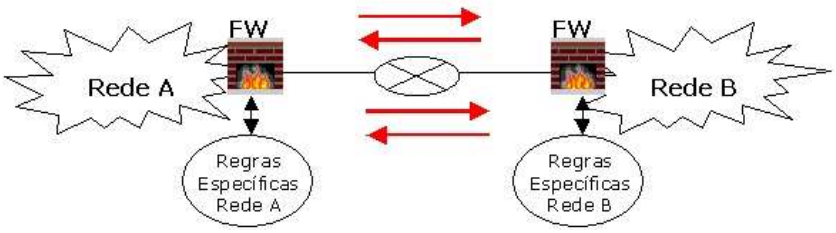


Figura 2.7: Conexão de redes com roteador e com *firewalls*

operacionais, as tarefas de NAT e filtragem de pacotes, embora residentes na mesma máquina, eram realizadas por ferramentas distintas, enquanto em outras implementações, uma mesma ferramenta realizava ambas as tarefas.

A partir de então, o uso de mascaramento (masquerade) de endereços IP privados para acesso à rede externa, utilizando temporariamente um único endereço externo (NAT N:1) passou a ser uma nova funcionalidade dos *firewalls*.

A conversão direta e fixa de endereços públicos em privados (NAT 1:1), em que determinadas máquinas da rede interna (geralmente servidores de aplicação) poderiam ser acessadas a partir da rede externa através de seu endereço público (mapeado para seu endereço privado) também fazia parte desta solução, e se encontrava disponibilizada a partir de então.

c) Terceira Geração - Checagem de Estados

Até esse momento, a dificuldade dos *firewalls* era diferenciar os pacotes que entravam na rede como resposta a solicitações internas, dos pacotes que, partindo da rede externa, buscavam iniciar conexões em máquinas da Intranet.

A inspeção do estado dos pacotes (*stateful inspection*) marcou uma nova era para os *firewalls*. Sua terceira geração, com a possibilidade de restringir o acesso de pacotes vindos da rede externa, liberando aqueles relacionados a conexões estabelecidas a partir de máquinas internas e bloqueando os demais. Dessa forma, tornou-se possível evitar vários tipos de ataques conhecidos até então, aumentando consideravelmente a segurança da rede corporativa.

d) Quarta Geração - Filtragens Específicas em Nível de Aplicação

Na época, uma das maiores limitações para os *firewalls* na detecção e bloqueio de ataques contra redes corporativas eram os ataques contra as implementações de serviços liberados pelo *firewall*, ou seja, a exploração de vulnerabilidades nas aplicações em execução acessadas a partir de portas válidas (serviços tradicionais), utilizadas para prover acesso a partir de máquinas externas a informações da instituição.

Nestes casos específicos, informações como endereços IP, portas, protocolos e estados de conexão não eram suficientes para identificar e eventualmente bloquear a exploração das vulnerabilidades dos programas.

O “mito” de que os dados da camada de aplicação só deveriam ser manipulados pelos equipamentos das extremidades da conexão (cliente e servidor) caiu por terra, diante da necessidade de filtragem das informações transportadas nesta camada, de modo a identificar ataques em andamento contra a corporação.

A quarta geração de *firewalls* é marcada, portanto, por implementações que disponibilizam parâmetros para configuração de filtragem neste nível específico.

**Re-roteamento**

O re-roteamento, no contexto deste trabalho, é uma técnica que pode ser utilizada para desviar tráfegos específicos baseando-se em determinados parâmetros das regras de filtragem.

Parecido, mas não igual, à conversão de endereços de rede (NAT), o re-roteamento não modifica o cabeçalho da camada de rede (endereços IP de origem e destino), apenas altera o fluxo de um determinado pacote, desviando-o para tratamento por um sub-conjunto de regras, ou mesmo para um outro roteador *firewall* específico.

O desvio de alguns pacotes, dentro do próprio *firewall*, para um sub-conjunto de regras específicas ficou conhecido como uso de cadeias ou sub-chains [Russel 2001]. Esse procedimento se constitui em um mecanismo bastante adequado para diminuir o retardo natural que seria causado pela submissão de todos os pacotes que atravessam o *firewall* à checagem de todas as regras de filtragem.

Conforme ilustrado na Figura 2.8, a partir destas funcionalidades, pacotes destinados a determinadas portas podem ser submetidos a cadeias de filtragem (conjuntos de regras) específicas, enquanto todo o tráfego diferente deste perfil seria submetido a um conjunto menor de regras e, conseqüentemente, teria um retardo menor ao atravessar o *firewall*.

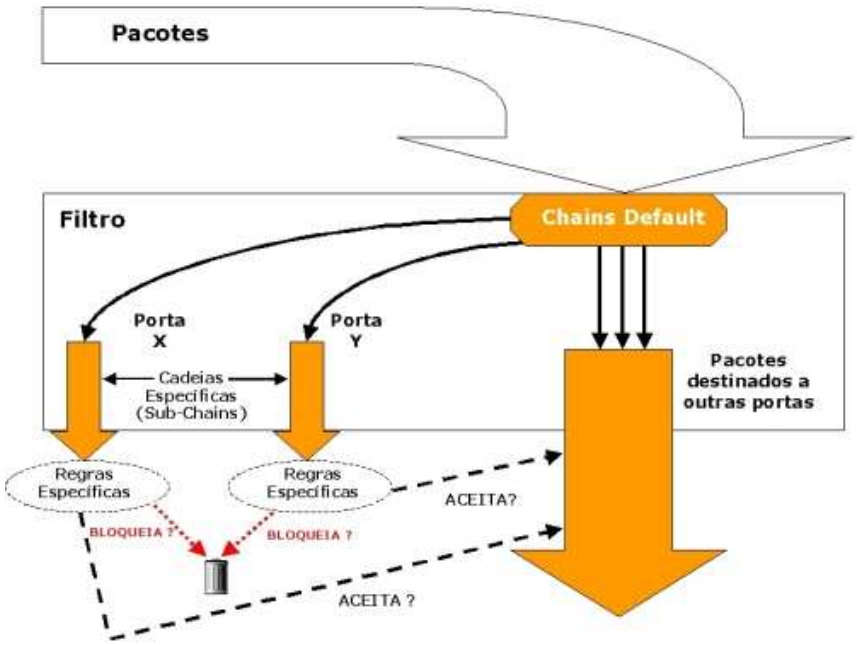


Figura 2.8: Desvio de tráfego para filtragens específicas no próprio *firewall*

Embora esse procedimento reduza consideravelmente o volume de dados processa-

dos, o redirecionamento baseado em filtragens específicas, não para sub-chains, mas para outras máquinas, implica em uma menor utilização de recursos do *firewall* principal, que passa a ser utilizado prioritariamente para as filtragens mais abrangentes.

Algumas implementações parciais deste tipo de solução, como o Iproute2 (ver tópico 3.2.2) baseiam-se somente nos endereços de origem e destino do pacote, enquanto outras mais completas, como o módulo ROUTE.patch (ver tópico 3.2.3) adicionado ao Netfilter/iptables, *firewall* padrão do GNU/Linux, podem basear-se em qualquer informação das camadas da pilha TCP/IP.

Nesse último enfoque, a tabela de rotas original do *firewall* não é alterada. Cada pacote que chega ao *firewall* pode ser analisado e ter seu próximo destino, a partir de então, re-roteado, pela ferramenta ou módulo, para outro roteador e/ou *firewall*.

### Sistemas de Detecção de Intrusões (IDS)

Dentro de um processo mais amplo que inclui a prevenção, a monitoração e a reação a ataques, a detecção de intrusões pode ser enquadrada como parte da etapa de monitoração [Moreira 2001].

### Justificativa de Utilização de IDS

Uma ferramenta de IDS é apenas mais uma estratégia de segurança em uma rede corporativa. O componente principal e vital, dentre os mecanismos de segurança, seguramente é o *firewall*. Um IDS deve, pois, complementar, e nunca substituir, o *firewall* da rede [Moreira 2001].

Algumas justificativas para a utilização de IDS em redes corporativas são relacionadas a seguir [Moreira 2001]:

* Permite uma resposta automática a um ataque, sem a intervenção humana;

* Agiliza o tempo e a facilidade de identificação e reação aos ataques, pois não depende de análises dos logs do sistema operacional;

* Quanto antes se detectar uma tentativa de ataque ao *site* da instituição, maiores serão as chances de impedir sérios e potenciais gastos com o comprometimento do sistema como um todo;

* Conhecendo os ataques direcionados à rede monitorada, torna-se mais fácil a elaboração de um esquema de defesa mais específico;

* Detectando invasores e prevenindo que a rede seja utilizada como “ponte” para ataques a outras instituições, livra-se a instituição de embaraços e custos legais;

* Após a ocorrência de um ataque efetivo, torna-se possível a análise da técnica utilizada, auxiliando na tomada de medidas adequadas para evitar que a mesma situação se repita.

### Características

O IDS é capaz de detectar e alertar os administradores quanto a possíveis ataques ou comportamentos anormais na rede da organização. Informações importantes sobre

tentativas de ataque, que não se podem obter através do uso das ferramentas tradicionais de filtragem, a exemplo dos *firewalls*, podem ser conseguidas por meio desses sistemas.

As informações geradas por estes sistemas podem oferecer subsídios suficientes para que a organização melhore sua proteção contra quaisquer tipos de ataque, sejam internos, ou contra máquinas com visibilidade externa [Nakamura & Geus 2002].

A seguir são apresentadas as principais características de um IDS [Nakamura & Geus 2002]:

* Monitoração e análise de atividades dos usuários e sistemas;

* Avaliação da integridade de arquivos importantes do sistema e de arquivos de dados;

* Análise estatística do padrão de atividade, com base em assinaturas de ataque conhecidas;

* Análise de atividades anormais na rede corporativa;

* Detecção de erros de configuração no sistema;

* Detecção de ocorrências em tempo real;

* Fornecimento de informações valiosas sobre atividades “maliciosas” na rede;

* Análise com base em cada caso, com resposta apropriada para cada um deles;

* Identificação do destino do ataque;

* Gerenciamento central, garantindo que todos os casos sejam analisados e respondidos de maneira consistente;

* Transparência, pois o sistema não indica quais pontos ou segmentos da rede estão sendo monitorados;

* Capacidade de registro do ataque, de modo a aprender com os ataques realizados e preparar uma defesa melhor;

* Flexibilidade de resposta, com a capacidade de reação, para a prevenção de possíveis danos;

* Configuração, tomando cuidado com as respostas “falso positivo”, caso em que um alarme falso é enviado quando a tentativa de ataque não existe, o que pode ser tão perigoso quando um ataque real.

Após a detecção de uma tentativa de ataque, diversos tipos de ações podem ser tomadas como resposta, dentre elas [Nakamura & Geus 2002]:

* Reconfiguração do *firewall*;

* Alarme (som);

* Aviso de SNMP para sistemas de gerenciamento de redes;

* Geração de eventos para o sistema operacional (como o MS-Windows);

* Geração de logs por meio do Syslog (serviço de registro de logs do UNIX);

* Envio de e-mail;

* Envio de mensagem para o pager;

* Gravação das informações sobre o ataque;

* Gravação das evidências do ataque para análise posterior (computação forense);

* Execução de um programa capaz de manipular o evento;

* Finalização da conexão.

**Tipos de IDS**

Os dois tipos inicialmente encontrados de IDS foram o *Host-Based Intrusion Detection System* (HIDS) e o *Network-Based Intrusion Detection System* (NIDS). O processo evolutivo que acontece com toda tecnologia levou ao desenvolvimento de um IDS híbrido (*Hybrid IDS*), que aproveita as melhores características do HIDS e do NIDS [Nakamura & Geus 2002].

a) *Host Based Intrusion Detection System*

O *Host Based Intrusion Detection System* (HIDS) faz o monitoramento do sistema, com base em informações de arquivos de logs ou de agentes de auditoria. O HIDS pode ser capaz de monitorar acessos e alterações em importantes arquivos do sistema, modificações nos privilégios dos usuários, processos do sistema, programas que estão sendo executados, uso da CPU, entre outros aspectos [Nakamura & Geus 2002].

O HIDS pode também realizar a checagem da integridade dos arquivos do sistema, por meio de *checksums*. Essa é uma característica importante, porque arquivos corrompidos, por *backdoors* por exemplo, são detectados antes que causem problemas mais sérios. Na maioria das vezes, eles são considerados ferramentas, ao invés de sistemas, pois não são capazes de emitir alertas em tempo real.

A análise de *logs*, realizada pelo HIDS, faz com que ataques de força bruta, por exemplo, possam ser detectados. Porém, ataques mais sofisticados podem passar despercebidos.

Os pontos fortes do HIDS são [Nakamura & Geus 2002]:

* O HIDS pode verificar o sucesso ou a falha de um ataque, com base nos registros (*logs*) do sistema;

* Atividades específicas do sistema podem ser monitoradas detalhadamente, como acessos a arquivos, modificação em permissões de arquivos, *logon* e *logoff* de usuários e funções do administrador;

* Ataques que ocorrem fisicamente no servidor (*keyboard attack*) podem ser detectados pelo HIDS;

* Ataques que utilizam criptografia podem passar sem serem notados pela rede, mas podem ser descobertos pelo HIDS, pois o sistema operacional primeiramente decifra os pacotes que chegam ao equipamento;

* É independente da topologia da rede, podendo ser utilizado em redes segmentadas por switches;

* Gera poucos “falso positivos”, ou seja, os administradores recebem poucos alarmes falsos de ataques;

* Não necessitam de *hardware* adicional.

Os pontos fracos que devem ser considerados nos HIDS são [Nakamura & Geus 2002]:

* É difícil de se gerenciar e configurar em todos os *hosts* que devem ser monitorados;

* Dependendo da plataforma em que a solução foi desenvolvida, pode ser dependente do sistema operacional;

* Não é capaz de detectar ataques de rede;

* Caso o HIDS seja invadido, as informações podem ser perdidas antes da análise;

* Necessita de espaço de armazenamento adicional para os registros do sistema;

* Por terem como base, também, os registros do sistema, podem não ser tão eficientes em sistemas que geram poucas informações de auditoria;

* Apresenta diminuição de desempenho no *host* monitorado.

b) *Network-Based Intrusion Detection System*

O *Network Based Intrusion Detection System* (NIDS) monitora o tráfego do segmento da rede onde está presente, geralmente com a interface de rede atuando em modo "promíscuo". A detecção é realizada por meio de captura e análise dos cabeçalhos e conteúdos dos pacotes, que são comparados com padrões ou assinaturas conhecidos [Nakamura & Geus 2002].

O NIDS é eficiente, principalmente contra ataques como *port scanning*, *IP spoofing* ou *SYN flooding*, e é capaz de detectar também ataques de *buffer overflow* e ataques contra serviços conhecidos, por meio da utilização de uma base de conhecimento com padrões e assinaturas de ataques.

Os pontos positivos dos NIDS são [Nakamura & Geus 2002]:

* O monitoramento pode ser fornecido para múltiplas plataformas;

* Analisando cabeçalhos e o *payload* de pacotes, ataques de rede podem ser detectados;

* O NIDS pode monitorar atividades suspeitas em portas conhecidas (serviços conhecidos);

* Os ataques podem ser detectados e identificados em tempo real, e o usuário pode determinar rapidamente o tipo de resposta apropriado;

* O NIDS é capaz de detectar não só os ataques, mas também as tentativas de ataques que não tiveram resultado;

* Com o NIDS funcionando, é difícil que um atacante possa apagar seus rastros, caso consiga invadir um equipamento (seria necessário invadir o próprio NIDS);

* O atacante terá dificuldades em saber se existe ou não um NIDS monitorando suas atividades (pelo modo passivo de atuação);

* Não causa impacto no desempenho da rede;

Os pontos negativos que podem ser encontrados em NIDS são [Nakamura & Geus 2002]:

* Perda de pacotes em redes saturadas;

* Dificuldade de compreensão de protocolos de aplicação específicos, como o SMB (*Server Message Block*);

* Não é capaz de monitorar tráfego cifrado;

* Dificuldade de utilização em redes segmentadas, principalmente com switches.

c) Hybrid Intrusion Detection System

A utilização de um modelo híbrido de IDS tem se tornado padrão, uma vez que os modelos tradicionais (HIDS e NIDS) geram logs específicos que são diferentes entre si, sendo que ambos contribuem para uma visão mais completa da intrusão, fornecendo dados valiosos para a detecção da mesma.

O IDS híbrido opera como o NIDS, coletando o tráfego da rede, processando os pacotes, detectando e respondendo a ataques. A diferença é que ele faz isso como um HIDS, ou seja, processa os pacotes endereçados à própria máquina. Com isso, desaparece o problema de desempenho, comum no NIDS, uma vez que cada máquina analisa somente o

tráfego destinado a ela e não o tráfego para todos os servidores da rede. Por outro lado, ainda persiste o problema de escalabilidade, pois um IDS híbrido deve ser instalado em cada equipamento [Nakamura & Geus 2002].

### Redes Privadas Virtuais - VPNs

As Redes Privadas Virtuais (*VPN - Virtual Private Networks*) oferecem o recurso de criar uma conexão de rede privada através de uma rede pública. Sem esse tipo de tecnologia, as organizações teriam que alugar ou implantar linhas dedicadas de comunicação entre elas para garantir uma comunicação remota segura. Para a maioria das empresas, o custo do aluguel e instalação dessas conexões seria tão caro que tornar-se-ia inviável financeiramente [Tittel 2003].

Em alguns casos, ainda, pode ser fisicamente impossível instalar links dedicados de transmissão devido a barreiras geográficas ou mesmo falta de espaço disponível nos dutos físicos existentes, necessitando do uso de conexões sem fio, com sinais enviados de forma aberta com possibilidade de interceptação. Assim, as Redes Privadas Virtuais podem ser usadas por organizações que precisam se comunicar através de redes (cabeadas ou não), sobre as quais não se têm controle ou que não sejam de sua propriedade [Tittel 2003].

A forma mais usual de se implementar uma VPN é através da utilização de um túnel encriptado dentro de um canal inseguro, ou seja, procedimentos de encriptação por meio de algoritmos seguros são realizados antes da transmissão dos dados e, no destino, as informações são desencriptadas pelo destinatário, que pode analisar o conteúdo transmitido com garantia de privacidade na transferência dos dados.

A ocorrência de interceptações de tráfego em comunicações por onde trafegam VPNs não é preocupante, já que, para ter acesso às informações o atancante teria que desencriptar os pacotes, tarefa matematicamente impossível em um curtoo espaço de tempo, considerando-se os recursos tecnológicos disponíveis na atualidade.

### *LogHosts*

Embora as ferramentas de registro de *logs* armazenem informações importantes para a identificação da origem de ataques e comandos executados na máquina-alvo, o comprometimento parcial ou total destes logs dificulta e até mesmo inviabiliza uma análise pós-invasão.

Tanto o uso pelo atacante de ferramentas automatizadas, como os *logcleaners*, visando esconder seu rastro ou até procedimentos mais radicais como o apagamento de todos os arquivos de log podem ser um problema para a análise de intrusões.

O conceito de *loghosts* surgiu neste contexto, e baseia-se na configuração de um servidor de *logs* centralizado (*loghost*), que recebe cópia em tempo real de tudo o que é registrado em máquinas importantes da rede.

Desta forma, até mesmo a exclusão total dos dados de uma máquina-alvo e formatação de seu disco rígido não seriam suficientes para encobrir os passos de um atacante, seguramente armazenados no *loghost* para avaliação posterior.

*Honeypots/Honeynets*

Se os sistemas de detecção de intrusões podem ser utilizados como fonte de aprendizado sobre novos ataques, além de sua função principal, que é a de detecção desses ataques, os *honeypots* podem ensinar muito mais. Eles são também conhecidos pelos termos *sacrifical lamb, decoy, booby trap, lures* ou $fly-traps$, e funcionam como armadilhas para os atacantes [Nakamura & Geus 2002].

Um *honeypot* não contém dados ou aplicações muito importantes para a organização, e seu único propósito é de se passar por um legítimo equipamento da instituição, que é configurado para interagir com um atacante em potencial. Assim, os detalhes da técnica utilizada e do ataque em si podem ser capturados e estudados. Podem ser utilizados programas especiais para tal função. O *honeypot* pode ser empregado também para que o IDS da organização seja testado e aperfeiçoado.

Um ponto importante a ser considerado é que é preciso tomar cuidado para que o *honeypot* não seja utilizado como ponte para outros ataques.

As *Honeynets* são redes formadas por vários honeypots, empregando, geralmente, equipamentos utilizando sistemas operacionais diferentes, de modo a estudar ataques contra diferentes serviços em diferentes arquiteturas de *hardware* e *software*.

### 2.1.6 Análise de Modelos de Filtragem (Cenários)

É possível identificar um conjunto de cenários distintos de filtragem de pacotes nas redes atuais. A diversidade destes cenários foi possibilitada pelas novas funcionalidades incorporadas aos *firewalls*, pelas novas necessidades de filtragem e soluções disponíveis para sua implementação. Nas sub-seções seguintes são analisados esses vários cenários.

#### Filtragem de Pacotes em Nível de Rede

É o cenário clássico, quando ainda não era possível distinguir claramente um *firewall* de um roteador, já que o nível de filtragem se resumia em bloquear ou liberar pacotes analisando unicamente os endereços IP de origem e destino (Figura 2.9).

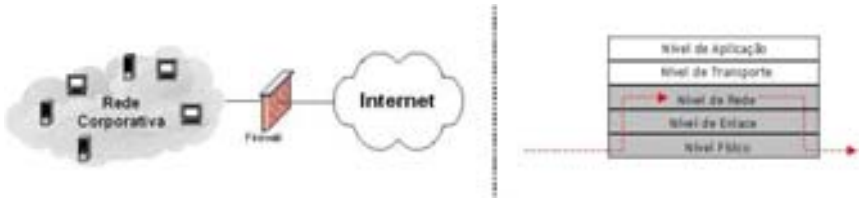


Figura 2.9: Filtragem de Pacotes em Nível de Rede

### Filtragem de Pacotes em Nível de Transporte

Geralmente disponível em *firewalls*, é também de possível implementação em alguns roteadores com capacidade de configurar ACLs com filtragem por portas. Esse método consiste em realizar a filtragem baseando-se em informações sobre os sockets de origem e destino (endereço IP origem, endereço IP destino, porta origem, porta destino) e outras informações sobre o protocolo de transporte (Figura 2.10).

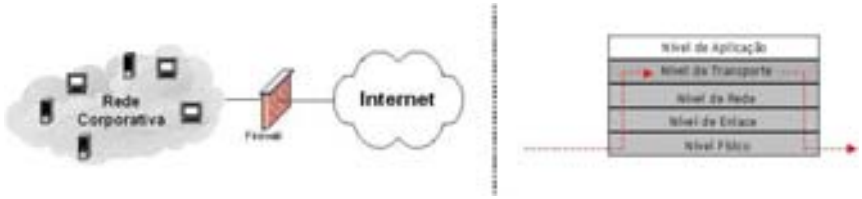


Figura 2.10: Filtragem de Pacotes em Nível de Transporte

### Filtragem de Pacotes em Nível de Aplicação

Consiste na análise das informações de controle contidas nos cabeçalhos de todos os protocolos da pilha TCP/IP, inclusive os dados da aplicação (payload), para a realização de filtragens baseadas em análise de seu conteúdo (Figura 2.11).

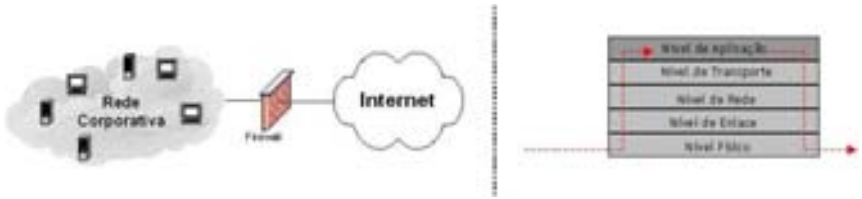


Figura 2.11: Filtragem de Pacotes em Nível de Aplicação

A filtragem de pacotes em nível de aplicação, por ser a mais complexa das três mencionadas anteriormente, pode apresentar diferentes características, que serão analisadas a seguir.

### Filtragem por Comparação de *Strings*

Esta é uma modalidade específica de filtragem em nível de aplicação, implementada por algumas ferramentas de produção e outras em fase de testes, em que a análise e tomada de decisão se dá comparando os dados transportados em nível de aplicação com *strings* pré-definidas.

Em geral, estas *strings* são conjuntos de assinaturas de ataque ou identificação de tráfego de conteúdo proibido na instituição.

Além de módulos experimentais do Linux Netfilter/Iptables, esta funcionalidade é também implementada por ferramentas como a Hogwash [Larsen 2001] para identificação e bloqueio de padrões de tráfego baseado em um conjunto de assinaturas.

### Centralizando a Detecção de Intrusões no *Firewall*

O IDS tradicional é conhecido como uma ferramenta passiva, isto é, a detecção de intrusões ocorre indicando que um incidente pode estar em andamento, sem tomar nenhuma decisão efetiva para seu bloqueio. A indicação do incidente é feita geralmente através de notificação ao administrador ou pelo envio de comandos para que outra ferramenta (na maioria dos casos o *firewall*) realize o bloqueio de pacotes subseqüentes vindos da origem identificada. Normalmente, os pacotes responsáveis pela identificação da intrusão pelo IDS não chegam a ser bloqueados.

Neste modelo, a comunicação remota entre IDS e *firewall* deve ser implementada de forma cautelosa, já que implica em um procedimento de inserção de novas regras de forma automática, sem intervenção direta do administrador, a partir de uma outra máquina da rede onde o IDS está em execução. A ocorrência de um falso positivo (alerta gerado por um IDS indicando um falso incidente de segurança) não permitiria uma análise criteriosa do administrador antes do bloqueio efetivo e isso tem sido suficiente para que muitos tenham receio deste tipo de implementação.

Para uma integração total entre IDS e *firewall*, de modo a bloquear pacotes identificados como "maliciosos" antes do seu repasse ao destino, alguns IDS (como o Snort IDS [Roesch 1998]) passaram a dispor de modos de funcionamento ativo, podendo responder aos incidentes com poder de filtragem, sem a necessidade de comunicação com o *software* específico de filtragem no *firewall* [Metcalf & Julien 2006] [Jonkman 2003] [Knobbe 2001].

Deste modo, o modelo ideal para esta integração é a instalação, na mesma máquina em que o *firewall* está em execução, do IDS com modo de funcionamento ativo. Desta forma seria possível identificar o tráfego suspeito baseado em suas assinaturas de ataque e bloquear o repasse desses pacotes. Além disso, existiria ainda a possibilidade de envio de resets (pacotes com a flag TCP RST ativa) para a máquina de origem do incidente, de modo a "resetar" a conexão maliciosa.

Assinaturas de ataque em nível de aplicação dariam a esta solução o status de filtro em nível de aplicação, e a este modo de funcionamento do IDS, a característica de execução de forma centralizada na mesma máquina do *firewall*.

A eliminação do IDS de rede seria óbvia, já que a ferramenta incorporou-se ao *firewall*, visando a deteção de intrusões e filtragem de pacotes durante seu roteamento.

### Centralizando o Anti-Vírus no *Firewall*

A facilidade de atualização e manutenção de um ambiente centralizado de filtragem de vírus, em detrimento da dificuldade em atualizar versões e listas de assinaturas de novos

vírus nos desktops das intranets, tem demonstrado, na prática, um maior grau de eficiência das soluções de anti-vírus.

Este modo de funcionamento consiste em redirecionar o tráfego de/para portas relacionadas aos serviços de envio e recebimento de e-mails, como 25/TCP (SMTP), 110/TCP (POP3) e 143/TCP (IMAP) para um agente intermediário (*proxy* de aplicação), dotado de regras de filtragem, em busca de padrões que identifiquem a presença de vírus/worms anexados às mensagens e, naturalmente, realizando o bloqueio do repasse destas mensagens.

Tendo em vista que a filtragem deve ser realizada durante o repasse dos e-mails entre as redes interna e externa, a integração deste tipo de solução ao *firewall* é uma opção viável, em teoria, por diminuir o número de elementos de filtragem e, conseqüentemente, o número de pontos de falha durante o trajeto dos pacotes da origem ao destino.

Algumas empresas que comercializam soluções de *firewall* e de anti-virus têm buscado realizar a "venda casada", oferecendo o conjunto de soluções integradas como um produto completo.

#### Filtragem de *Spam* no *Firewall*

Seguindo a mesma linha dos modelos de incorporação de filtragem de vírus/*worms* aos *firewalls*, os filtros de spam também necessitam analisar o tráfego de e-mails antes de seu repasse ao destino e, assim, a incorporação de soluções de filtragem de spam ao *firewall* também representam uma sumarização de elementos de funções semelhantes.

A filtragem de spam realiza a identificação de origens suspeitas, a partir de listas negras de servidores sem restrições para envio de e-mails, conhecidas como ORBL's (*Open Relay Black Lists*) e realiza o bloqueio do repasse de mensagens com este padrão.

A investigação do conteúdo dos e-mails em busca de padrões notadamente utilizado por spammers complementa as técnicas utilizadas por este tipo de ferramenta.

A incorporação destas técnicas de filtragem ao *firewall* e o bloqueio de origens notadamente suspeitas diretamente no *firewall* ratificam a necessidade de integração dos filtros de *spam* ao *firewall*.

#### Filtragem de Tráfego P2P (Peer-to-Peer) no *Firewall*

a) A Origem com o Napster

Em maio de 1999, Shawn Fanning, um estudante universitário de 18 anos, criou um *software* que combinava o sistema de mensagens instantâneas do IRC (*Internet Relay Chat*), os sistemas de compartilhamento de arquivos do MS-Windows e do Unix e ferramentas de busca por padrões. O seu objetivo era criar um sistema que tornasse mais fácil compartilhar arquivos mp3. O seu programa foi batizado de Napster (apelido do criador da ferramenta na universidade). Depois disso, Shawn fundou uma instituição com o mesmo nome e passou a distribuir o aplicativo cliente gratuitamente.

O Napster tornou-se a aplicação de Internet de crescimento mais rápido de todos os tempos, atingindo um pico de 13.6 milhões de usuários em fevereiro de 2001 [ComScore 2006]. Pelo feito, Shawn Fanning foi capa da revista Time.

O Napster, embora possa ser considerado o primeiro de uma série de programas P2P para o compartilhamento de multimídia, como Gnutella, Kazaa, E-Donkey, E-Mule, dentre outros, não é propriamente um sistema P2P, já que depende de um servidor de diretórios central, que contém um banco de dados com todos os clientes da aplicação com informações como: endereço IP da máquina, nome do usuário, lista de arquivos que está compartilhando e se o usuário encontra-se on-line no momento.

Cada usuário, para acessar o sistema, precisa instalar um cliente Napster em sua máquina e registrar-se no servidor, isto é, ser incluído na base de dados do servidor de diretórios com suas informações específicas.

Sete meses após a sua fundação, a empresa foi processada pela Recording Industry Association of America (RIAA) por violação da lei de copyright. A RIAA alegava que a Napster tinha ilegalmente criado um negócio, baseado no uso de material com copyright, que ela não tinha o direito de distribuir. A Napster, por sua vez, dizia que ele estava simplesmente fornecendo um serviço e que eram os usuários que estavam distribuindo o material entre si.

Depois de muitos embates jurídicos, a Napster chegou à falência em setembro de 2002, sendo vendida posteriormente para a Roxio Inc. que transformou-a em um negócio legal.

b) O Surgimento dos verdadeiros P2P

Com a repercussão do Napster e diante dos problemas causados pelo seu modo de funcionamento, surgiram posteriormente aplicativos como a Gnutella e Freenet, que não têm um servidor central, sendo completamente peer-to-peer.

O fato de existirem computadores atuando como peers na Internet não é novidade, isto é, o fato da arquitetura ser P2P não basta para explicar as mudanças recentes no uso da Internet.

O impacto do uso destes aplicativos se deu pela condição de que os nós destas novas redes P2P são estações comuns com conexão à Internet, funcionando até então somente como clientes de serviços tradicionais. O provimento distribuído de informações na forma de arquivos de tamanhos consideráveis, compartilhados diretamente *peer-to-peer*, foi o real causador dos problemas desta nova modalidade de aplicativo, principalmente pelo tráfego adicional gerado na Internet e nas redes corporativas.

c) O Bloqueio de conexões P2P em nível de rede e transporte

Um ponto comum neste tipo de aplicativo era a utilização de uma porta específica para a realização da troca de arquivos. Assim, nos *firewalls* que necessitavam bloquear o tráfego de determinada aplicação P2P bastava que fossem inseridas regras de bloqueio às referidas portas específicas.

d) O Bloqueio de conexões P2P em nível de aplicação

Em reação ao bloqueio de conexões P2P em nível de transporte, não demorou para que os implementadores de aplicativos P2P alterassem o seu modo de funcionamento, possibilitando a configuração por parte dos próprios usuários, atribuindo portas aleatórias para estabelecimento de conexão e troca de arquivos.

Sem a informação de quais portas estariam sendo utilizadas para as trocas de arquivos entre as máquinas executando aplicativos P2P, a única alternativa de filtragem passou a ser a análise dos dados na camada de aplicação.

A identificação de padrões de strings de início de conexão, para cada um dos novos protocolos P2P nas novas aplicações, passou então a fazer parte de assinaturas de alguns IDS e módulos específicos de alguns *firewalls*. Assim, nos pacotes iniciais de conexões P2P, o IDS ou o *firewall* (configurados adequadamente) identificam e bloqueiam o estabelecimento destas conexões, inibindo o início da troca de arquivos.

Também nesta solução, a incorporação deste nível de filtragem ao *firewall* passou a ser uma opção bastante aceita entre os administradores de redes.

**Outras Filtragens Específicas no** *Firewall*

Pode-se resumir em três os principais tipos de filtragens não convencionais, em nível de aplicação:

* A incorporação de módulos específicos nos *firewalls* para a realização de filtragem de pacotes em nível de aplicação baseados em assinaturas;

* A utilização de IDS ativos instalados no mesmo *hardware* do *firewall* (funcionando, na prática, como um *firewall*); e

* Os proxies de aplicação, implementados inicialmente para a intermediação e filtragem de tráfego Web (HTTP) e de e-mail (SMTP).

Estas implementações demonstraram na prática que a incorporação de novas regras de filtragem para análise de tráfego de novos aplicativos, que porventura surjam demandando filtragens específicas em nível de aplicação, é uma questão apenas de configuração das ferramentas, que, pela variedade de ocorrência desse tipo de filtragem, têm se tornado naturalmente flexíveis quanto à adição de novas regras.

### 2.1.7 Impactos da Concentração de Elementos de Filtragem

Os benefícios da concentração de soluções de filtragem no mesmo *hardware* em que o *firewall* principal da rede corporativa está em execução é questionável, quando o volume de pacotes analisados, principalmente em nível de aplicação, demanda um percentual considerável dos recursos da máquina (capacidade de processamento e memória), resultando em comprometimento do roteamento de pacotes não-suspeitos, devido ao aumento do tempo de retenção dos pacotes na máquina, antes do repasse aos respectivos destinos.

O aumento da utilização de recursos do *hardware* pode, ainda, resultar em descarte de pacotes antes mesmo do processamento, análise e repasse dos mesmos, caracterizando momentaneamente uma negação de serviço (denial of service), por saturação dos recursos da máquina no processamento das aplicações nela concentradas.

A necessidade de aumento do nível e diversificação dos tipos de filtragem realizadas no perímetro da rede corporativa é um fato inconteste. As limitações físicas (e suas conseqüências) pela utilização do modelo baseado na concentração destes elementos em uma única máquina representam um problema que cresce proporcionalmente ao tamanho das redes e ao volume e diversidade de tráfego, podendo resultar em um comprometimento significativo do *hardware*, a ponto de inviabilizar a implantação do modelo e suas novas funcionalidades.

Por tratar-se de um problema relativamente novo, a determinação do grau de comprometimento deste tipo de solução não encontra referências na literatura atual, constituindo

parte deste trabalho, como forma de justificar a solução proposta.

A descentralização física, mantendo a concentração lógica, destes elementos de filtragem surge como uma solução para estes problemas.

### 2.1.8 Trabalhos Relacionados

Durante a realização deste trabalho surgiram algumas novas soluções para filtragem em nível de aplicação, em sua maioria incorporando ao *firewall* filtros que analisam a carga (*payload*) do pacote em protocolos de aplicação específicos, com destaque para os protocolos P2P que utilizam portas aleatórias e implicam em aumento considerável de tráfego nas redes pelo número de conexões que utilizam e volume de dados compartilhados entre usuários destes serviços.

Para este tipo de filtragem, pode-se citar o *layer7* e o *ipp2p* ambos módulos adicionais a serem incorporados ao Netfilter/Iptables, além do *match string* utilizado nos testes deste trabalho.

Outros trabalhos relacionados que merecem destaque são os baseados em utilização de *bridges* posicionadas entre o *firewall* e a rede interna e/ou entre o *firewall* e o roteador externo. Este tipo de solução transfere para a *bridge* as filtragens específicas (em nível de aplicação, por exemplo) liberando o *firewall* deste tipo de filtragem.

Estas soluções apresentam, pelo menos, duas desvantagens. A primeira é que, mesmo diminuindo o volume de filtragens no *firewall*, o grande volume de tráfego que não estaria subordinado a regras específicas de filtragem é submetido às regras da *bridge*. Assim, embora não seja um elemento de roteamento (a captura de pacotes é feita em nível de enlace), a *bridge* interfere na vazão do tráfego de todos os pacotes que entram ou saem da rede, já que estes têm que cruzar a *bridge* (e seus filtros) na entrada e na saída. A segunda desvantagem é o fato de a *bridge* ser mais um ponto crítico de falha na rede, isto é, o seu comprometimento pode implicar na paralisação de todo o tráfego de entrada e saída da rede.

O surgimento dos IPS (*Intrusion Prevention Systems*) como apoio aos *firewalls* também pode ser visto como trabalho relacionado.

Os IPS funcionam de forma semelhante aos NIDS, porém, posicionam-se entre o *firewall* e o roteador externo e funcionam de forma ativa, isto é, prestam o "primeiro combate" aos ataques externos, identificando possíveis ataques à rede, baseando-se em políticas pré-determinadas, e bloqueando o referido tráfego antes de atingir o *firewall*.

Funcionando de forma semelhante às *bridges*, os IPS têm sido apresentados como solução adicional de segurança pelos principais fabricantes de soluções de segurança atualmente no mercado. Também merece destaque o IPS HLBR (*Hogwash Light BR* [Filho & Araújo 2005]), solução *Open Source* em desenvolvimento, encabeçada por brasileiros.

Mesmo apresentando (no caso dos IPS corporativos) como diferencial o *hardware* otimizado para este tipo de filtragem, os IPS apresentam as mesmas desvantagens da utilização de *bridges* como filtros auxiliares, tendo em vista que têm o mesmo posicionamento físico na rede. Um dos fabricantes, a 3Com, porém, apresenta em sua solução, o Tipping Point IPS [3Com 2006], um *hardware* adicional, localizado entre a interface de rede do IPS e o meio físico, que monitora o funcionamento do IPS e, em caso de problemas

em seu funcionamento realiza o chaveamento, isolando o IPS e garantindo a passagem do tráfego entre o roteador e o *firewall*. Ainda assim, o IPS em funcionamento interfere na vazão do tráfego dos pacotes que não deveriam ser submetidos às regras de filtragens específicas, além de (no caso dos IPS corporativos) serem uma solução de custo bastante elevado.

# Capítulo 3

# SuRFE: Sub-Rede de Filtragens Específicas

A implementação de uma sub-rede de filtragens específicas (SuRFE), liberando o *firewall* principal da concentração de mecanismos adicionais de filtragem, embora mantenha o retardo no tráfego dos pacotes re-roteados, uma vez que novos hops são adicionados no trajeto destes pacotes, alivia a carga de utilização dos recursos do *firewall* principal, reduzindo (se não eliminando) os descartes de pacotes causados pela sobrecarga em função da filtragem.

Outro aspecto positivo desse modelo se refere à diminuição natural da retenção de pacotes que não coincidem com os padrões definidos nas regras de triagem para re-roteamento e filtragens específicas: esses pacotes não suspeitos sofrem agora somente um atraso adicional mínimo, devido à análise preliminar realizada no *firewall* para a realização desta triagem e à filtragem em nível de rede e transporte no próprio *firewall*.

## 3.1 Visão Geral

A implementação de uma SuRFE consiste em retirar do *firewall* principal os filtros específicos, principalmente os que analisam o nível de aplicação, mantendo naquele equipamento somente as regras que identifiquem a necessidade de submissão de determinados pacotes a esses filtros.

As regras no *firewall* principal passam a ser baseadas somente em informações dos cabeçalhos de rede e transporte, aplicadas a todos os pacotes que atravessam o *firewall*.

Em situações específicas, (pacotes cujas informações coincidem com os padrões especificados nas regras de triagem), pacotes são então re-roteados para a SuRFE, onde conjuntos específicos de regras de filtragem são aplicados, conforme esquema ilustrado na Figura 3.1.

Como resultado da aplicação dessas regras específicas na SuRFE, os pacotes podem ser bloqueados, protegendo as máquinas e usuários da rede interna, ou re-encaminhados ao *firewall* principal.

No retorno ao *firewall* principal, os pacotes são então submetidos a um outro conjunto de filtros, (também baseados na análise de cabeçalhos de rede e transporte) e, caso não

haja bloqueio e descarte por estas regras, são então encaminhados ao fluxo de dados que se destina à intranet corporativa.

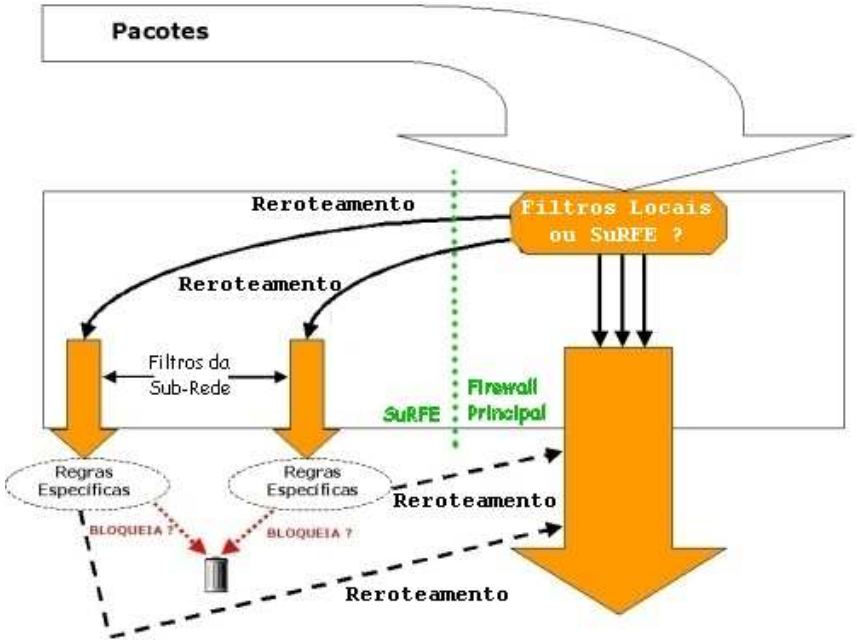


Figura 3.1: Esquema de Funcionamento da SuRFE

## 3.2 Caracterização do Ambiente de Implantação

No uso de uma SuRFE em um ambiente de produção, o *firewall* principal mantém seu posicionamento físico e lógico, realizando a filtragem perimetral da rede, tendo, porém, duas interfaces adicionais para ligação com a SuRFE, conforme mostrado na Figura 3.2. Assim, o *firewall* deve incluir as seguintes interfaces:

* Uma interface ligada diretamente ao roteador externo (conexão à Rede Externa);

* Uma interface ligada a um *switch* interno (conexão à Rede Interna);

* Duas interfaces ligadas à SuRFE, sendo uma delas utilizada para enviar o tráfego re-roteado e a outra para o retorno de pacotes que não tenham sido bloqueados pelas regras de filtragem da SuRFE.

No *firewall* principal são então inseridas regras específicas para determinar que pacotes devem ser re-roteados para a SuRFE, de modo que os demais pacotes não sofram nenhum retardo adicional, sendo submetidos somente às regras tradicionais de filtragem no próprio *firewall* principal. Essas regras são determinadas pelo interesse particular de

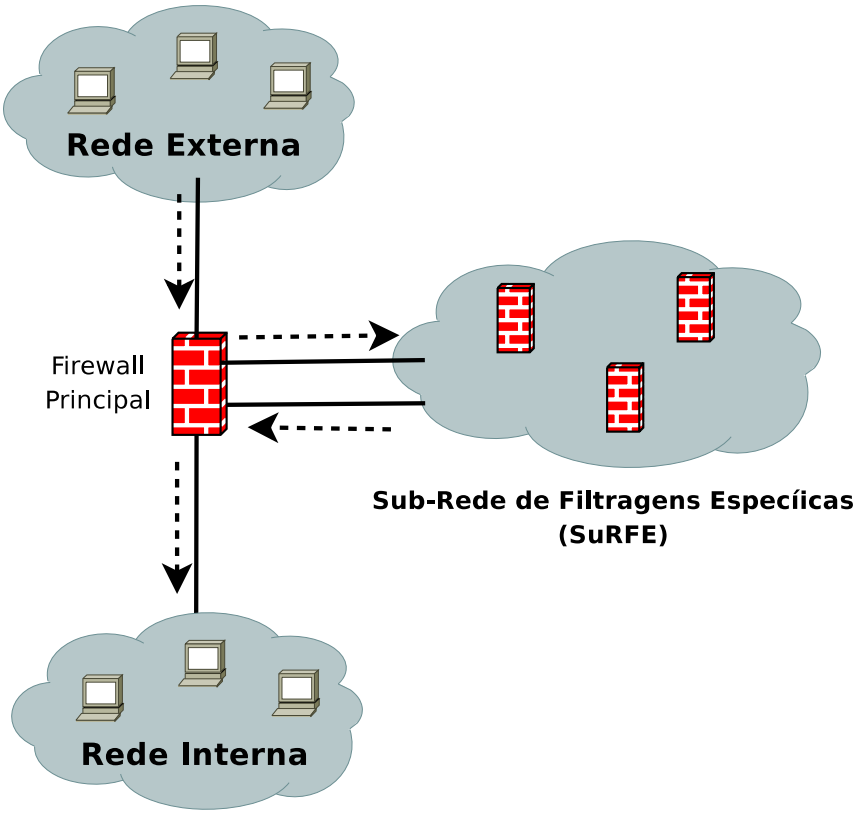


Figura 3.2: Visão geral de uma sub-rede de filtragens específicas

cada instituição em examinar certos tipos de tráfego e sua capacidade de realizar essa verificação através de máquinas dedicadas a essa filtragem.

Os pacotes re-roteados seguem para a SuRFE, onde são submetidos a filtragens específicas de acordo com a sua caracterização de tráfego, isto é, baseando-se em parâmetros como endereço IP de origem, endereço IP de destino, porta de origem, porta de destino e tipo de protocolo, o tráfego é direcionado para um (ou mais) filtro(s) da SuRFE, com regras de filtragens até o nível de aplicação.

Caso, durante a filtragem, o tráfego seja caracterizado como malicioso, ou simplesmente não permitido na rede da instituição, a própria SuRFE realizará o bloqueio e descarte dos pacotes. Caso contrário, os pacotes retornam ao *firewall* principal para prosseguirem ao seu destino.

Para permitir a modificação do *kernel* do *firewall* principal, de modo a realizar o re-roteamento de pacotes específicos, optou-se por utilizar o netfilter/iptables, *firewall* de código-fonte aberto e sem restrições de modificação, disponível a partir do *kernel* 2.4 do sistema operacional Linux. Para os testes, de fato, optou-se pela utilização do *kernel* 2.6.15 configurado com suporte aos módulos necessários.

### 3.2.1 O módulo de re-roteamento

A tabela de rotas padrão do *kernel* dos principais sistemas operacionais somente permite a definição de rotas baseadas no destino dos pacotes. Para a realização de roteamento baseado na origem ou mesmo baseado em campos da camada de transporte, foi necessária a utilização de um módulo adicional ao *kernel* para prover esta funcionalidade.

Existem, atualmente, para utilização no *kernel* do Linux, duas soluções que possibilitam este tipo de roteamento: Iproute2 e ROUTE.patch.

### 3.2.2 Iproute2

O iproute2 nasceu da necessidade do Linux se adequar às novas tecnologias utilizadas atualmente na Internet. Novos conceitos foram criados e outros antigos foram modificados, tornando obsoletas e ineficientes as clássicas ferramentas das diferentes formas de UNIX, como "ifconfig", "arp", "route", entre outras.

Ao contrário das ferramentas tradicionais, o iproute2 centraliza toda a configuração de rede e de controle de banda em apenas uma ferramenta para cada um destes fins. Estes comandos são o "ip"e o "tc". O primeiro visualiza ou modifica as configurações de rede (roteamento, endereços de rede, endereços físicos, dentre outros) e o segundo tem como finalidade a configuração do controle de tráfego.

Tradicionalmente, cada roteador contém apenas uma tabela de roteamento com os destinos e caminhos respectivos. Com o iproute2 pode-se ter até 256 tabelas de roteamento diferentes, que serão consultadas ou não a partir da comparação de cada pacote com as regras da política de roteamento. Deste modo, antes do pacote ser roteado, ele é submetido a um conjunto de regras semelhante a um *firewall* e que determina qual tabela será usada para aquele pacote específico.

Para realizar roteamento com o iproute2, baseando-se em parâmetros da camada de transporte, é possível utilizá-lo em conjunto com o iptables, através da extensão MARK. Dessa maneira, o iptables, baseando-se em detalhes da camada de transporte e/ou rede, "marca"os pacotes que devem ser analisados pelo iproute2. O re-roteamento pelo iproute2 é feito, então, de acordo com as marcas nos pacotes que chegam até a sua tabela de roteamento.

### 3.2.3 ROUTE.patch

Trata-se de um módulo experimental, porém estável, a ser incorporado ao iptables (é necessária a aplicação de um *patch* e a recompilação do *kernel* para oferecer suporte a esta funcionalidade). Esse módulo habilita a configuração não convencional de rotas, permitindo o roteamento de um pacote recebido através de uma interface ou para um *host* específico, independente do destino inicial do pacote.

No re-roteamento nenhuma característica do pacote é modificada, como endereços de origem e destino ou portas, apenas a rota de destino é alterada, de acordo com a verificação realizada pelo iptables.

Ao contrário do iproute2, o ROUTE.patch é aplicado diretamente no *kernel* do Linux, alterando o netfilter e habilitando o seu uso como parâmetro do próprio iptables.

Pela simplicidade de uso e eficiência na utilização, optou-se por utilizar este módulo no re-roteamento necessário à implementação da SuRFE.

## 3.3 Aspectos de Implementação

### 3.3.1 Aspectos de *hardware* e *software*

Para implementar uma SuRFE são incorporadas ao *firewall* principal da rede duas placas de rede adicionais para o desvio (re-roteamento) dos pacotes específicos.

O sistema operacional de todas as máquinas do ambiente é o GNU/Linux (distribuição Debian) e o *software* específico para filtragem (*firewall*) é o Netfilter/Iptables, ambos com código fonte aberto e utilização sem restrições, mesmo na necessidade de alterações para adequação aos ambientes testados.

Para a realização do re-roteamento, o *kernel* do *firewall* principal foi recompilado, de modo a oferecer suporte ao módulo de re-roteamento e, conseqüentemente, às regras de desvio incorporadas no início da tabela de filtragens.

O nível de filtragem e robustez da SuRFE pode variar de acordo com as necessidades e tamanho de cada rede. Uma lista não exaustiva de cenários de implementação está descrita na próxima seção.

### 3.3.2 Cenários de Implementação

A seguir estão descritos os cenários propostos para a implementação de uma SuRFE.

#### SuRFE com uma única máquina

O modelo apresentado na Figura 3.3 é o de mais fácil implementação e de menor custo, já que envolve somente uma máquina adicional à estrutura pré-existente, e duas placas de rede no *firewall*, para o desvio dos pacotes que serão analisados e retorno dos pacotes que não foram bloqueados pelas regras de filtragem.

Entretanto, a utilização do re-roteamento deve ser um recurso suportado e implementado no *firewall* principal, já que somente os pacotes que se deseja analisar serão re-roteados para o filtro de aplicações, caracterizando um desvio baseado no serviço e/ou rede de origem/destino, sem modificação do cabeçalho, enquanto os demais pacotes serão filtrados e/ou repassados para seus destinos sem o re-roteamento.

#### SuRFE com Balanceamento de Carga

Neste segundo modelo, mostrado na Figura 3.4, insere-se um balanceador de carga entre o *firewall* principal e os *firewalls* da SuRFE, visando distribuir o processo de filtragem de aplicações. Para isto pode ser utilizado um módulo de balanceamento no próprio *firewall* principal, eliminando-se o elemento intermediário e tornando o número de máquinas da SuRFE escalável.

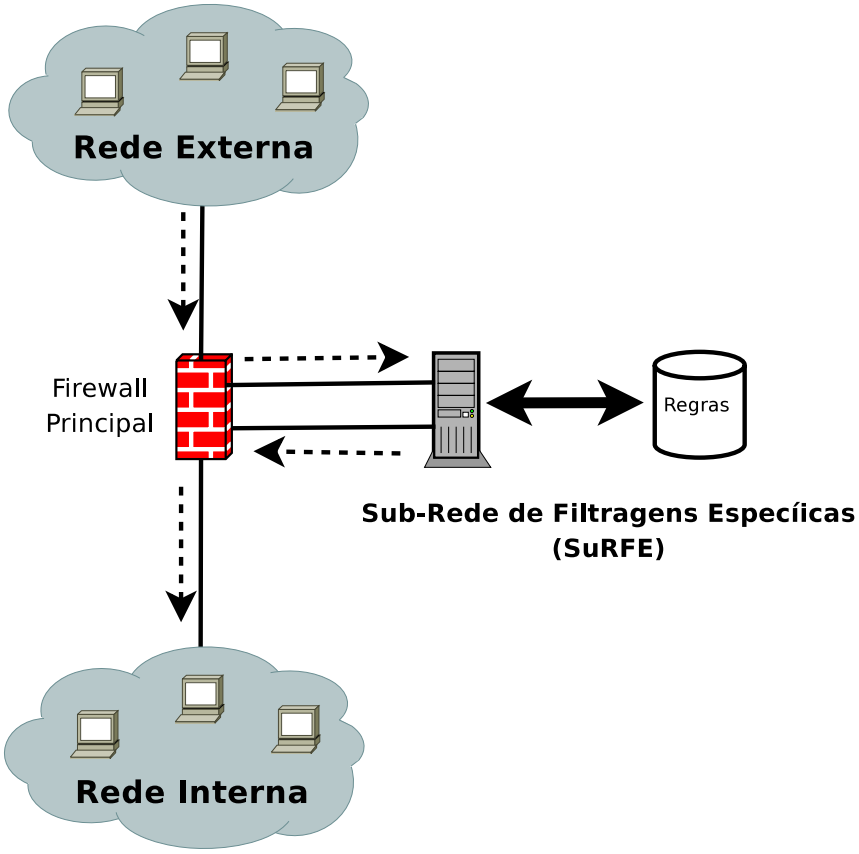


Figura 3.3: SuRFE com uma única máquina

As máquinas da SuRFE deverão ter todas o mesmo perfil de *software*, mas não necessariamente o mesmo perfil de *hardware*.

O mesmo perfil de *software* se justifica pela condição de que qualquer uma das máquinas da SuRFE poderá receber o pacote re-roteado, devendo ter em seu filtro as regras necessárias à análise de qualquer pacote.

O perfil de *hardware* não necessariamente igual se justifica pela possibilidade de atribuição de pesos ao balanceador de modo a destinar uma maior quantidade de pacotes a máquinas da SuRFE com *hardware* mais robustos e, conseqüentemente, menos pacotes a máquinas da SuRFE com menos recursos de *hardware*.

Além da escalabilidade, permitindo a remoção ou inserção de novas máquinas à SuRFE, bastando para isto a adequação das regras do balanceador, esta solução visa garantir uma maior redundância e disponibilidade do serviço, já que em caso de falha em uma das máquinas da SuRFE, bastaria eliminá-la do balanceamento, ficando a filtragem sendo realizada normalmente pelas demais máquinas.

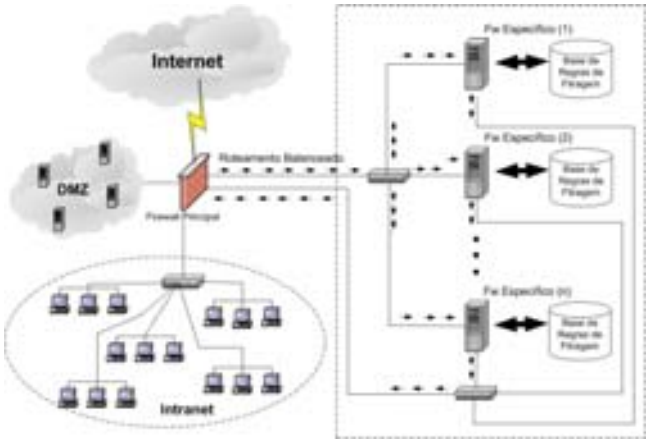


Figura 3.4: SuRFE com Balanceamento de Carga

**Surfe com Separação por Tipo de Filtragem**

Neste último modelo apresentado na Figura 3.5, a SuRFE é formada por *firewalls* com regras específicas para cada filtragem específica (porta ou conjunto de portas). Assim, o *firewall* principal redireciona os pacotes ao *firewall* específico, de acordo com a aplicação destino contida em cada um deles.

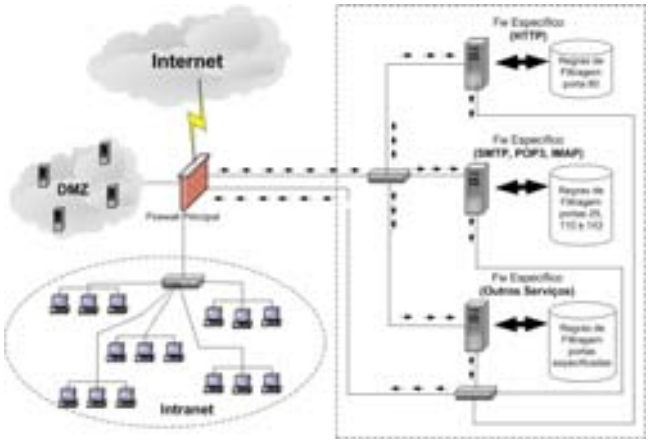


Figura 3.5: SuRFE com Separação por Tipo de Filtragem

# Capítulo 4

# Metodologia e Ambiente de Testes

Este capítulo apresenta a fundamentação para a execução dos testes de validação da proposta apresentada nessa dissertação, abrangendo os objetivos para a realização destes testes, cenários, métricas de desempenho, técnicas de medição conhecidas e identificação das técnicas de interesse, bem como as ferramentas disponíveis e utilizadas para a realização desses testes.

## 4.1 Objetivos dos Testes

A escolha dos parâmetros de medição para os testes deve levar em conta o que é necessário medir, para comprovar a eficiência do uso de uma SuRFE.

O objetivo principal é medir o impacto no processamento dos pacotes que passam pelo *firewall* com e sem o uso de uma SuRFE, e provar que:

1) Filtragens específicas não convencionais (especialmente as que manipulam a camada de aplicação dos pacotes) implicam em uma carga adicional de processamento podendo comprometer as tarefas básicas do *firewall* principal pelo esgotamento de recursos de *hardware*;

2) Essas filtragens também provocam atraso desnecessário em pacotes que não coincidem com padrões das regras de triagem (e que, portanto, estes pacotes não deveriam estar sujeitos a filtragens específicas);

3) Com a utilização de uma SuRFE observa-se uma diminuição na carga de processamento no *firewall* principal;

4) Os pacotes que não deveriam ser tratados em filtros específicos, quando submetidos a uma solução utilizando uma SuRFE têm uma maior vazão (menor atraso) em seu roteamento ao destino final;

5) Os pacotes que devem ser tratados em filtros específicos, quando submetidos a uma solução utilizando uma SuRFE, têm um atraso maior, porém o incremento do atraso não é significativo, ou seja, o resultado final do uso da solução proposta compensa a ocorrência deste retardo.

## 4.2 Cenários de Testes

Para a realização dos testes foi utilizado um ambiente de produção modificado, de modo a representar o primeiro cenário dentre os descritos no tópico 3.3.2.

Este ambiente, implementado em uma rede real, é a rede da UFRN, incorporando a SuRFE ao *firewall* principal da instituição para a realização dos testes (Figura 4.1).

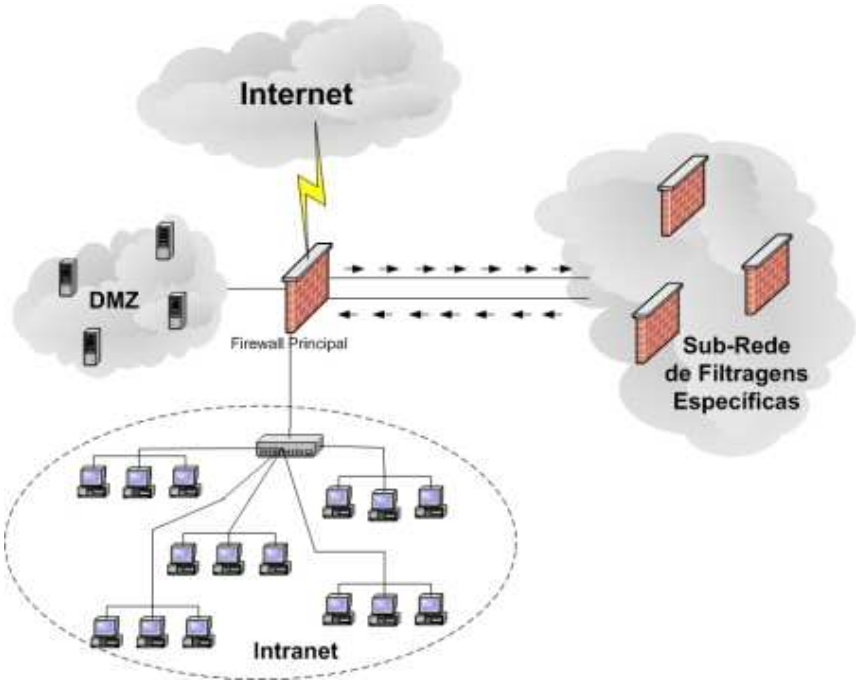


Figura 4.1: SuRFE aplicada à Rede UFRN

Pelo *firewall* principal da rede UFRN trafegam todos os pacotes de/para as redes internas da instituição e de/para uma zona desmilitarizada (DMZ) onde estão os servidores públicos de aplicações com visibilidade externa (acessíveis a partir da Internet). O link externo (ligado ao Ponto de Presença da RNP no estado - PoP/RN), atualmente, tem uma largura de banda de 34Mbps e as conexões internas (à intranet e à DMZ) são feitas através de interfaces Gigabit Ethernet (1000Mbps).

Para a realização dos testes, foram utilizados programas específicos para a medição dos parâmetros relevantes descritos na seção 4.3.

Três cenários de testes podem se basear nos modelos de implementação apresentados na seção 3.3.2. Estes cenários diferenciam-se entre si pelo número de máquinas da SuRFE, utilização ou não de métodos de redundância (alta disponibilidade e balanceamento de carga) e abrangência de filtros em cada uma delas.

Para a validação dos objetivos propostos, entretanto, utilizou-se a implementação de um cenário simples, utilizando-se uma única máquina na SuRFE (primeiro cenário da

Seção 3.3.2). Apesar da simplicidade do modelo quando comparado aos demais cenários apresentados, estima-se que ele seja suficiente para verificar os objetivos citados na seção anterior. Os outros cenários incorporam funcionalidades adicionais ao esquema mais simples, adequando o modelo proposto a eventuais problemas físicos na implementação em um ambiente real. Entretanto, o uso desses cenários mais complexos em testes comparativos, usando ou não a SuRFE, deve conduzir a conclusões similares às obtidas com o cenário mais simples.

## 4.3 Métricas de Desempenho e Técnicas de Medição

A medição de parâmetros em redes é importante para avaliar o desempenho de mecanismos de filtragem em *firewalls* e soluções auxiliares. O comportamento assimétrico do tráfego nestas redes, porém, dificulta a obtenção de algumas métricas.

O grupo de trabalho da Internet Engineering Task Force (IETF), que trata de métricas de desempenho para o protocolo IP (*IP Performance Metrics - IPPM Working Group*) [Zekauskas & Uijterwaal 2006], tem produzido documentos que especificam essas métricas e propõem metodologias para o seu cálculo. Existem algumas ferramentas ativas de monitoramento que foram criadas com base nesses documentos, injetando pacotes na rede para observar como eles se comportam em um caminho específico.

### 4.3.1 Métricas Conhecidas

Os padrões estabelecidos pelo IPPM-WG estão detalhados em quatro RFCs:
* RFC 2678 - Define os Parâmetros de Conectividade;
* RFC 2679 (One-way delay);
* RFC 2680 (One-way packet loss); e
* RFC 2681 (Two-way delay).

A partir da análise destes padrões, pode-se distinguir três grupos de técnicas de medição conhecidas:
* Medições de Largura de Banda;
* Medições *Hop-by-hop*; e
* Medições *End-to-end*.

#### Medições de Largura de Banda

Nestas medições é avaliada a possibilidade de execução de aplicações e serviços. A seguir é apresentada a evolução das técnicas e ferramentas para este tipo de medição:
* Conceito de Dispersão de Pacotes: Jacobson, 1988;
* Packet-pair: Keshav, 1991;
* Bprobe e Cprobe: Carter e Crovella, 1996;
* Tcpanaly: Paxson, 1996;
* One-packet: Jacobson, 1997;
* Pchar, Mah e Clink: Downey, 1999;
* Packet Tailgating e Nettimer: Lai e Baker, 2000;

* Pathrate: Dovrolis, 2001; e

* Pathload: Jain e Dovrolis, 2002.

Estas técnicas/ferramentas buscam medir:

a) Largura de Banda de Contenção

- Bottleneck Bandwidth ou Capacity

- Taxa Máxima (camada IP) que um fluxo pode alcançar em um caminho quando não existe tráfego (sem carga).

b) Lagura de Banda Disponível

- Available Bandwidth

- Taxa Máxima (camada IP) que um fluxo pode alcançar em um caminho na presença de tráfego (com carga).

c) Largura de Banda Utilizada

- Quantidade de tráfego em um enlace num determinado momento.

**Medições** *Hop-by-hop*

Medição de parâmetros utilizando agentes em todos os componentes da rede (elementos intermediários de roteamento e *hosts* em todas as redes interligadas)

Esta técnica é geralmente utilizada quando existem vários equipamentos ativos de roteamento entre as redes, de modo a identificar os elementos intermediários que comprometem o desempenho da rede como um todo para, por exemplo, determinar novas rotas ou indicar a necessidade de incremento de recursos em um ou mais roteadores da sub-rede de roteamento.

**Medições** *End-to-end*

Este tipo de medição requer a cooperação apenas dos pontos terminais e pode ser a única forma de monitorar um caminho que inclui várias redes, situação comum em ambientes de rede de produção.

São métricas definidas neste tipo: a conectividade, o atraso (de ida e de ida/volta), variação de atraso (*jitter*), perdas, reordenação de pacotes e taxas de utilização.

As medições podem ser feitas de forma ativa, através de uma análise em tempo real, ou passiva, analisando os resultados coletados anteriormente.

**Medições de Desempenho de** *Hardware*

Além da necessidade de medir o desempenho dos equipamentos ativos de rede no tratamento dado aos pacotes e sua relevância no tráfego de rede, deve-se considerar também a necessidade de técnicas de medição, que contemplem o desempenho do *hardware* e o comportamento destes equipamentos ativos de rede no que diz respeito ao esgotamento (ou não) de recursos, a ponto de comprometer o seu funcionamento.

### 4.3.2 Definição das Métricas de Interesse

Para a validação da proposta contida nessa dissertação, os parâmetros de interesse são encontrados dentro do escopo das técnicas de medições *end-to-end* e de desempenho de *hardware*, ou seja, os parâmetros que devem ser medidos durante a realização dos testes são os seguintes:

* atraso no processamento de pacotes;
* vazão; e
* carga de processamento (no *firewall* principal).

Todos estes parâmetros são medidos com e sem o uso de uma SuRFE.

### 4.3.3 Considerações sobre Grandezas Estatísticas

Para garantir a consistência dos resultados dos testes, se faz necessário realizar uma análise estatística prévia sobre o comportamento do tráfego de entrada/saída, que será alvo das medidas realizadas. O objetivo de tal análise é determinar valores significativos para algumas grandezas estatísticas relacionadas aos intervalos em que as medidas de tráfego devem ser realizadas.

#### Intervalo de Amostra

Para a definição do intervalo de amostra, observou-se o tráfego em ambiente de produção, em busca de um padrão de tráfego por unidade de tempo.

No ambiente de produção utilizado para os testes (a rede de computadores da UFRN), a ferramenta MRTG [Oetker 2006] é utilizada para construir, através da coleta de tráfego realizada a cada 5 minutos via protocolo SNMP, o gráfico diário (Figura 4.2), semanal (Figura 4.3), mensal e anual do tráfego no *link* principal da rede.

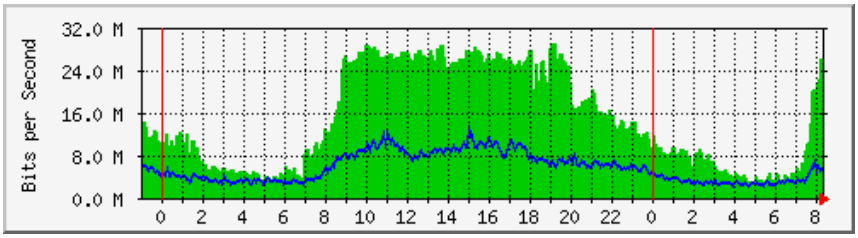


Figura 4.2: Tráfego Diário Típico - Rede UFRN

Analisando-se os gráficos que apresentam o tráfego na rede, pode-se observar comportamentos periódicos semelhantes, determinando um padrão aceitável para uma representação estatística de coleta durante a semana:

* Existem dois padrões de tráfego distintos ao longo do dia, ou seja, maior e menor tráfego de acordo com horários de pico de utilização;

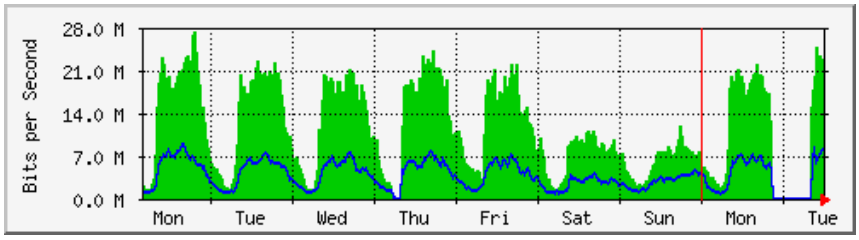


Figura 4.3: Tráfego Semanal Típico - Rede UFRN

* O tráfego em cada dia da semana tem volume semelhante nas semanas subsequentes, conforme foi constatado nos gráficos mensais;

* O tráfego no sábado e domingo é menor que nos demais dias da semana.

Uma análise desses gráficos permite observar que:

* O tráfego no período compreendido entre 08:00h e 20:00h aproxima-se do limite do *link* da instituição;

* O tráfego fora deste horário cai para cerca de 10% (dez por cento) do limite do *link* da instituição.

**Intervalo de Amostra Utilizado**

Com base nas análises apresentadas na seção anterior, a coleta de dados ficou definida como devendo ser realizada em dois momentos distintos (período de maior tráfego [08:00h as 20:00h] e período de menor tráfego [20:01h as 07:59h]).

## 4.4 Ferramentas para Medição dos Parâmetros de Interesse

Diante das opções disponíveis, são apresentadas a seguir as ferramentas utilizadas nos testes e que formaram os subsídios para a análise dos resultados obtidos.

### 4.4.1 *Load Average*

O parâmetro chamado de *Load Average* [Gunther 2003] é a média da soma do número de processos aguardando na fila para entrar em execução, somado ao número de processos em execução nos últimos 1, 5 e 15 minutos. A carga (*load*) medida não refere-se à utilização da CPU, mas ao tamanho total da fila apresentada a partir de amostras em três séries de momentos diferentes.

Esta ferramenta de medição, nativa em todos os sistemas UNIX, Linux e BSD, é utilizada para análise de carga dos sistemas e planejamento de sua capacidade de recursos computacionais (*capacity planning*) [Emmons 2006].

A operação da ferramenta não necessita da instalação de programas adicionais. O parâmetro *load average* é apresentado como resultado dos comandos *top, uptime, procinfo* ou pode-se, ainda, verificar diretamente o conteúdo da variável de ambiente */proc/loadavg*.

Os resultados não levam em conta o número de processos, somente os que estão aguardando por algum recurso para entrar em execução (processador, memória, dispositivos de armazenamento ou rede).

Segundo [Gunther 2003] é aconselhável manter o sistema com *load average* próximo a 1. Valores maiores que 5 indicam sobrecarga de utilização da máquina, podendo comprometer o seu funcionamento normal.

A tabela 4.1 apresenta um exemplo de utilização do *load average*, coletado diretamente da variável de ambiente específica.

```
#cat /proc/loadavg
1.35 1.08 1.00 1/399 26591
```

Tabela 4.1: Exemplo de Utilização da Ferramenta *Load Average*

Neste exemplo, os três primeiros valores indicam que a média de carga no último minuto foi de 1.35, a média nos últimos 5 minutos foi de 1.08 e a média nos últimos 15 minutos foi de 1.00.

### 4.4.2 Iperf

O Iperf [Tirumaia et al. 2005] é um *software* de análise de desempenho de banda e cálculo de perda de datagramas em redes, mantido pela Universidade de Illinois, sob licença GPL.

Baseado no paradigma cliente/servidor, para a medição de desempenho de elementos de filtragem de tráfego UDP e/ou TCP deve-se instalar o cliente e o servidor em segmentos de rede diferentes daquele em que o equipamento ativo testado esteja ligado. No caso desse trabalho deve-se instalar o cliente e o servidor Iperf nos segmentos de rede anterior e posterior ao *firewall* principal, de forma a medir a vazão de pacotes.

Existem versões desta ferramenta para praticamente todos os principais sistemas operacionais em uso na atualidade, como Unix, Linux, FreeBSD, OpenBSD, MacOS, Solaris e Windows.

A tabela 4.2 apresenta um exemplo de utilização do Iperf, considerando a ativação da ferramenta (servidor) configurada para aguardar conexões na porta 81/TCP, enquanto a tabela 4.3 apresenta os resultados da ferramenta do lado cliente para uma carga específica.

Neste exemplo, conforme resultados apresentados na tabels 4.3, para uma carga de 6.19 Mbytes, a vazão média foi de 5.18 Mbits por segundo.

### 4.4.3 Netperf

O Netperf [Jones 2005] pode ser usado para medir o desempenho de diferentes tipos de operações de rede. Seu principal foco é o volume de transferência de arquivos e o

```
# iperf -s -p 81
– – – – – – – – – – – – – – – – – – – – – – – – – – – – – – – – – – – – –
Server listening on TCP port 81
TCP window size: 85.3 KByte (default)
```

Tabela 4.2: Exemplo de Uso da Ferramenta IPerf - Ativação do Servidor

```
# iperf -c <ip_do_servidor> -p 81
– – – – – – – – – – – – – – – – – – – – – – – – – – – – – – – – – – – – –
Client connecting to <ip_do_servidor>, TCP port 81
TCP window size: 16.0 KByte (default)
– – – – – – – – – – – – – – – – – – – – – – – – – – – – – – – – – – – – –
local <ip_cliente> port 3685 connected with <ip_servidor> port 81
0.0-10.0 sec ** 6.19 MBytes ** 5.18 Mbits/sec
```

Tabela 4.3: Exemplo de Uso da Ferramenta Iperf - Configuração do Cliente e Resultado

desempenho de requisição/resposta, usando tanto TCP quanto UDP e a Interface Berkeley Sockets.

O Netperf foi desenvolvido a partir do modelo cliente-servidor básico. Assim, existem dois módulos executáveis - netperf e netserver. O Netserver implementa o lado do servidor, enquanto o netperf implementa o lado cliente.

Na execução do Netperf, inicialmente, dá-se o estabelecimento da conexão com o sistema remoto. Esta conexão é então utilizada para passar as informações do teste de configuração e resultados de/para o sistema remoto.

Os testes do Netperf podem ser classificados em dois grupos: teste de *streams* e teste de requisição/resposta. O teste de *streams* mede o desempenho de transferência de arquivos, enquanto o teste de requisição/resposta mede as transações por segundo para um dado volume de requisições e respostas. Uma transação é definida como a troca de uma requisição simples e uma resposta simples.

A tabela 4.4 apresenta um exemplo de utilização do Netperf, considerando a ativação da ferramenta (servidor) configurada para aguardar conexões na porta 80/TCP, enquanto a tabela 4.5 apresenta a configuração do lado cliente e a tabela 4.6 apresenta os resultados da ferramenta para uma carga específica.

Para este exemplo, o Netperf utilizou uma carga de 16.384 bytes, realizando testes pelo período de 10,08 segundos, apresentando como resultado uma vazão média de 5.06 x $10^6$ bits por segundo, conforme mostrado na tabela 4.6.

```
# netserver -p 80
Starting netserver at hostname 0.0.0.0 port 80 and family AF_UNSPEC
```

Tabela 4.4: Exemplo de Uso da Ferramenta Netperf - Ativação do Servidor

```
# netperf -H <ip_do_servidor> -p 80
TCP STREAM TEST from 0.0.0.0 (0.0.0.0) port 0 AF_INET to <ip_do_servidor>
(ip_do_servidor) port 0 AF_INET
```

Tabela 4.5: Exemplo de Uso da Ferramenta Netperf - Ativação do Cliente

### 4.4.4 ab - Apache Benchmark

O ab (apache benchmark) é uma ferramenta disponível na instalação do servidor *web* Apache [Apache 2006], para medir o desempenho no acesso a um determinado conteúdo em um servidor *web*, descartando o carregamento de css (*cascading style sheets*), javascript e imagens, e sem levar em consideração qualquer acesso à memória *cache* de *browser*.

Esta ferramenta permite especificar o número de acessos, número de consultas simultâneas e conteúdo (arquivo) a acessar no *site* remoto, e é geralmente utilizada para medir o tempo de acesso a um servidor *web* (geração de tráfego HTTP).

O servidor, neste tipo de teste, é o próprio servidor *web*; e o cliente, a ferramenta ab, a partir da qual deve ser informado o número de conexões ao servidor *web*, além do número de instâncias simultâneas para a realização destas conexões.

No cliente (ab) também é informado o arquivo a ser acessado no servidor. Geralmente o arquivo *index.html*, página *web* padrão disponível no servidor.

A tabelas 4.7, 4.8 e 4.9 apresentam um exemplo de utilização do ab. Na tabela 4.7, a ativação de um servidor web (Apache) configurado para aguardar conexões na porta 80/TCP. A tabela 4.8 apresenta a configuração do lado cliente e a tabela 4.9 apresenta os resultados da ferramenta.

| Recv Socket Size *bytes* | Send Socket Size *bytes* | Send Message Size *bytes* | Elapsed Time *secs* | Throughput $10^6 bits/sec$ |
|---|---|---|---|---|
| 87380 | 16384 | 16384 | 10.08 | 5.06 |

Tabela 4.6: Exemplo de Uso da Ferramenta Netperf - Resultados

```
# service apache2 start
Starting apache 2.0 web server....
Proto||Endereço Local||Endereço Remoto||Estado||PID/Program name
tcp ||0 0.0.0.0:80 ||0.0.0.0:* ||LISTEN||4201/apache2
```

Tabela 4.7: Exemplo de uso da Ferramenta ab - Ativação de um Servidor Web

```
# ab -n 100 -c 1 http://servidor:80
```

Tabela 4.8: Exemplo de uso da Ferramenta ab - Configurações do Cliente

Considerando os resultados deste exemplo, apresentados na figura 4.9, para 100 conexões ao servidor web, não houve perda de dados, todas as conexões foram bem sucedidas, o tempo gasto para todas as conexões foi de aproximadamente 1.18 segundos, a página HTML carregada a cada conexão tinha tamanho igual a 11.800 bytes, o total de bytes transferidos foi de 39.700, a taxa de requisições por segundo foi de 84,39, o tempo médio por requisição foi de 11,849 ms e a taxa média de transferência foi de 32,07 Kbytes por segundo.

```
Server Software: Apache/2.0.55
Server Hostname: servidor
Server Port: 80

Document Path: /
Document Length: 118 bytes

Concurrency Level: 1
Time taken for tests: 1.184942 seconds
Complete requests: 100
Failed requests: 0
Write errors: 0
Total transferred: 39700 bytes
HTML transferred: 11800 bytes
Requests per second: 84.39 [#/sec] (mean)
Time per request: 11.849 [ms] (mean)
Time per request: 11.849 [ms] (mean, across all concurrent requests)
Transfer rate: 32.07 [Kbytes/sec] received
```

Tabela 4.9: Exemplo de uso da Ferramenta ab - Resultados

# Capítulo 5

# Testes Realizados e Análise de Resultados

Neste capítulo são apresentados os resultados dos testes realizados, com vistas a mostrar o comportamento dos mecanismos de filtragem com e sem a implementação de uma SuRFE.

Os resultados, expressos em gráficos e tabelas, são a sumarização dos resultados da aplicação das ferramentas descritas no capítulo anterior. A configuração básica destas ferramentas, bem como exemplos de resultados produzidos por elas estão descritos no capítulo anterior.

De forma a verificar se os objetivos descritos no capítulo anterior foram atingidos, os seguintes testes foram realizados, sem e com a presença de uma SuRFE:

* Testes de Carga no *Firewall* Principal
* Testes de Vazão no *Firewall* Principal
* Testes de Atraso na Transmissão de Pacotes

## 5.1 Testes de Carga no *Firewall* Principal

### 5.1.1 Sem Uso de uma SuRFE

Estes testes visam verificar o primeiro dos objetivos apresentados na seção 4.1. Assim, os resultados destes testes apresentam o comportamento do *firewall* principal diante da inclusão de regras de filtragens específicas não convencionais (regras de filtragem em nível de aplicação).

Nestes testes, a carga média (*load average*) [Emmons 2006] da máquina é monitorada em intervalos de 15 minutos, tempo necessário para o cálculo da média neste período, quando a máquina é submetida a regras de filtragem específicas. O experimento consiste em incrementar o número de regras de filtragem até o esgotamento ou sobrecarga no uso de recursos do *hardware* do *firewall* principal.

*Load Average*

A tabela 5.1 apresenta os resultados de carga média normal no *firewall* principal (sem aplicação de regras de filtragem específicas), e o comportamento quando foram inseridas

50, 100, 500, 1000 e 3000 regras de filtragem em nível de aplicação. Na figura 5.1 estes resultados são apresentados graficamente para facilitar a comparação dos resultados.

| | 1 min (média) | 5 min (média) | 15min (média) |
|---|---|---|---|
| sem regras | 0.97 | 1.07 | 1.25 |
| 50 regras | 1.23 | 1.13 | 1.09 |
| 100 regras | 1.47 | 1.30 | 1.22 |
| 500 regras | 2.81 | 2.05 | 1.91 |
| 1000 regras | 5.18 | 4.60 | 3.79 |
| 3000 regras | 10.19 | 8.91 | 5.37 |

Tabela 5.1: Resultado dos Testes de Carga no *Firewall* Principal (sem SuRFE)

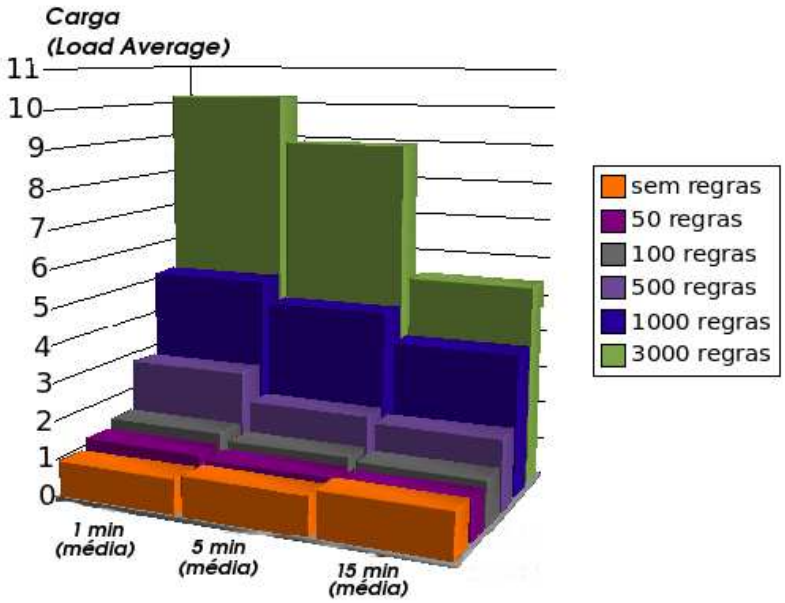


Figura 5.1: Carga Média no *Firewall* Principal (sem SuRFE)

O comportamento do hardware do *firewall* principal sem filtragens específicas (em nível de aplicação) apresenta um valor médio satisfatório (*load average* médio em torno de 1), indicando que a fila de processos não está impactando o funcionamento normal da máquina e que as regras atuais de filtragem não comprometem os recursos de *hardware*.

Na aplicação de 50 regras de filtragem em nível de aplicação, houve um aumento no *load average* de 26,8%, ultrapassando a média de 1, porém, mesmo com a submissão destas regras, a máquina respondia normalmente a execução de comandos, situação que

não se alterou significativamente na aplicação de 100 regras (mesmo com o aumento de 51,55% no *load average* , quando comparado com a carga média medida no sistema sem regras específicas).

A resposta da máquina aos demais processos passou a causar impactos negativos (lentidão notória na execução de comandos) a partir dos testes com 500 regras (quando o *load average* aproximou-se de 3) e 1000 regras (carga média ultrapassou 5).

Os testes foram encerrados quando, ao aplicar 3000 regras de filtragem em nível de aplicação, o acesso a partir das máquinas internas para a rede externa (cruzando o *firewall*) ficou gradativamente mais lento, os novos acessos locais ou remotos ao *firewall* (novas sessões) passaram a ser recusados e, pouco após a carga média ultrapassar 10, o hardware esgotou seus recursos e deixou de responder completamente, sendo necessária a reinicialização (*reset*) da máquina.

### 5.1.2 Com Uso de uma SuRFE

Com a implementação de uma SuRFE no ambiente de testes, esta seção apresenta o conjunto de testes realizados para medir a carga (baseada na mensuração do *load average*) no *firewall* principal quando aplica-se as regras específicas não nesta máquina, mas, em máquinas localizadas na SuRFE.

A tabela 5.2 apresenta os resultados de testes semelhantes aos realizados na seção 5.1.1. A diferença da situação anterior, é que nos testes atuais as regras de fitragem foram aplicadas no *firewall* principal com uso de uma SuRFE.

| | 1 min (média) | 5 min (média) | 15min (média) |
|---|---|---|---|
| sem regras | 0.97 | 1.03 | 1.19 |
| 50 regras | 0.97 | 1.01 | 1.07 |
| 100 regras | 0.99 | 1.04 | 1.21 |
| 500 regras | 1.01 | 1.07 | 1.18 |
| 1000 regras | 1.03 | 1.10 | 1.20 |
| 3000 regras | 1.06 | 1.12 | 1.19 |

Tabela 5.2: Resultado dos Testes de Carga no *Firewall* Principal (com SuRFE)

Analisando os resultados mostrados na tabela 5.2, observa-se apenas um leve aumento de carga com o aumento do número de regras inseridas, não significativo por manter-se próximo a 1, justificado pelas regras de re-roteamento inseridas para desvio do tráfego específico a ser submetido às regras de filtragem da SuRFE.

### 5.1.3 Análise Comparativa dos Resultados

As figuras 5.2, 5.3 e 5.4 apresentam gráficos comparativos das cargas médias no *firewall* principal quando este apenas re-roteia os pacotes que necessitam de filtragens específicas para a SuRFE, mantendo neste *firewall* principal somente as regras convencionais

de filtragens não específicas e as regras de re-roteamento para a SuRFE com os resultados coletados nos testes da seção 5.1, ou seja, as situações de testes do *firewall* principal sem SuRFE x *firewall* principal com SuRFE.

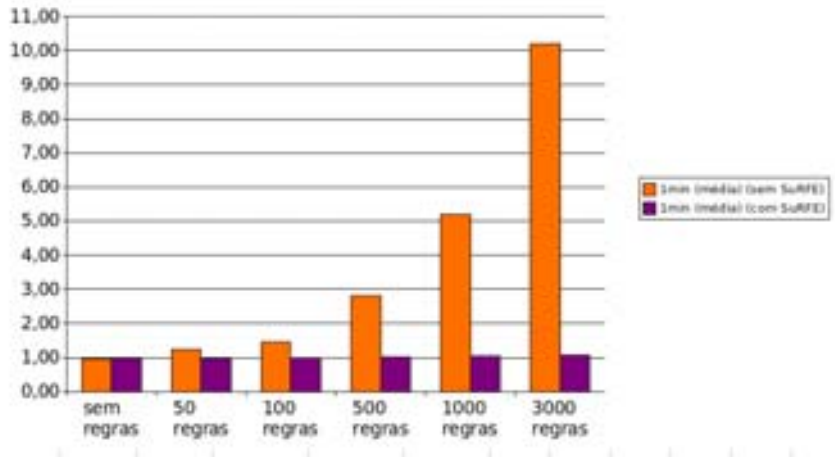


Figura 5.2: Carga no *Firewall* Principal com e sem uso de SuRFE (média em 1min)

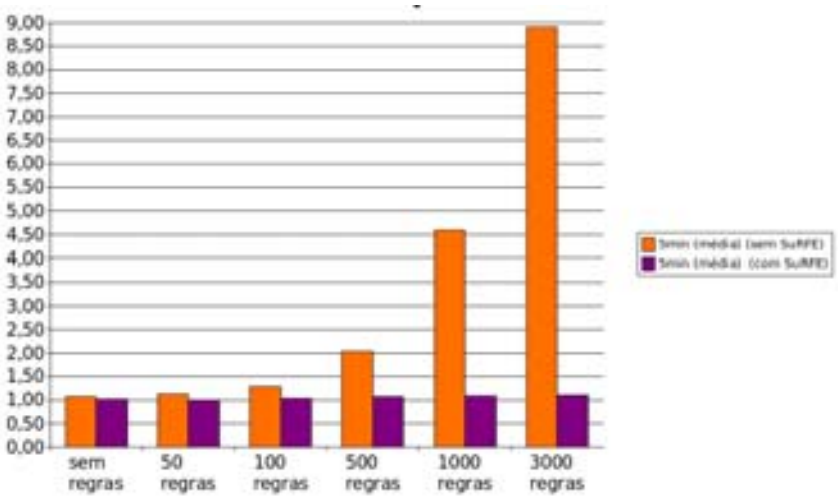


Figura 5.3: Carga no *Firewall* Principal com e sem uso de SuRFE (média em 5min)

O uso de uma SuRFE para executar as regras específicas de filtragem no nível de aplicação, conforme observado nos resultados, não impacta na carga do *firewall* principal.

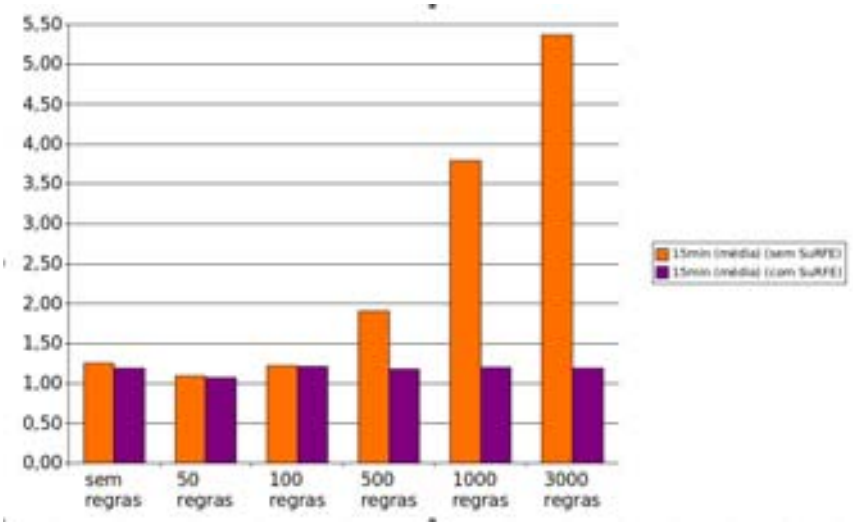


Figura 5.4: Carga no *Firewall* Principal com e sem uso de SuRFE (média em 15min)

O comprometimento dos recursos observado nos testes da seção 5.1 ao serem aplicadas regras específicas de filtragem ao *firewall* principal, que se mostrou proporcional ao número de regras inseridas, não se repete nos testes quando o mesmo número de regras é submetido à SuRFE, deixando ao *firewall* principal apenas a função de re-roteamento deste tráfego específico.

A inserção de uma SuRFE implica, portanto, na diminuição significativa da carga submetida ao *firewall* principal quando é necessário inserir regras de filtragens específicas (como regras em nível de aplicação). O resultados dos testes desta seção avaliam, porém, apenas os impactos do uso da SuRFE no *firewall* principal, já que as filtragens específicas são re-roteadas à SuRFE.

## 5.2 Testes de Vazão no *Firewall* Principal

Conforme mostrado na seção 5.2.1, a metodologia empregada nestes testes utiliza as ferramentas de medição especificadas na seção 4.4, onde mediu-se inicialmente a vazão no *firewall* principal tendo sido inseridas regras tradicionais de filtragem (sem regras em nível de aplicação), em períodos de maior utilização (horários de pico) e de menor utilização. Em seguinda verificou-se o comportamento desta mesma métrica ao serem inseridas regras de filtragem em nível de aplicação, nas mesmas quantidades utilizadas no teste anterior.

Na seção 5.2.2 são apresentados os testes de vazão no *firewall* prinicpal com a presença de uma SuRFE, seguindo-se uma metodologia semelhante à adotada na seção anterior.

Na última seção é realizado então um estudo comparativo entre os dois conjuntos de testes, de forma a evidenciar o impacto sofrido pela vazão de pacotes no *firewall* principal,

quando se realiza filtragens específicas nele sem a utilização de uma SuRFE. Verifica-se assim parte do objetivo 4, descrito na seção 4.1.

### 5.2.1 Sem Uso de uma SuRFE

Para realizar estes testes utilizou-se as ferramentas Iperf e Netperf.

*Testes com as Ferramentas Iperf/Netperf*

Os resultados apresentados pelas ferramentas Iperf e Netperf foram semelhantes (diferenças percentualmente desprezíveis). Assim, para a apresentação dos resultados destes testes de vazão, optou-se por utilizar a média dos resultados das duas ferramentas.

A tabela 5.3 apresenta os resultados obtidos e a figura 5.5 apresenta o gráfico comparativo das duas medições de modo a mensurar o impacto na vazão de dados no *firewall* principal quando incorpora-se a este regras de filtragem em nível de aplicação, sem a utilização de uma SuRFE.

| | Vazão Média em horários de menor utilização | Vazão Média em horários de maior utilização |
|---|---|---|
| sem regras | 245 Mbps | 239 Mbps |
| 50 regras | 241 Mbps | 237 Mbps |
| 100 regras | 222 Mbps | 217 Mbps |
| 500 regras | 167 Mbps | 143 Mbps |
| 1000 regras | 131 Mbps | 107 Mbps |
| 3000 regras | 32 Mbps | 0 Mbps |

Tabela 5.3: Resultado dos Testes de Vazão no *Firewall* Principal (sem SuRFE)

Conforme pode-se observar no gráfico da figura 5.5, as curvas de diminuição de vazão nos horários de maior e menor utilização são semelhantes para as mesmas quantidades de regras adicionadas. Os valores sempre menores para o gráfico dos testes em horários de maior utilização justifica-se pela concorrência entre os dados do experimento e o tráfego normal da rede.

Analisando a performance da vazão diante do incremento no número de regras de filtragem em nível de aplicação, constata-se a diminuição significativa desta vazão diante do atraso imposto aos pacotes pela submissão a estas novas regras. O decremento nos valores da vazão chegam a zero nos testes em horários de maior utilização da rede, quando os pacotes são submetidos a 3000 regras de filtragens específicas, inviabilizando a utilização deste nível de filtragem em cenários como esse, impactando significativamente no tráfego de todos os pacotes que atravessam o *firewall* principal.

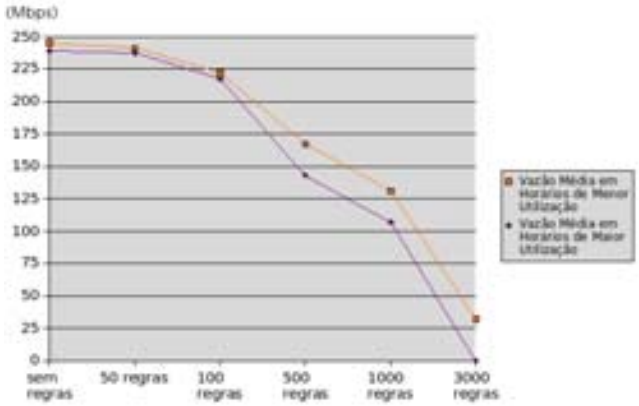


Figura 5.5: Vazão no *Firewall* Principal (sem SuRFE)

*Testes com a ferramenta ab*

Nos testes da seção anterior, comprovou-se a queda de performance da vazão no *firewall* principal para todos os pacotes que cruzam a máquina. Como testes complementares, são apresentados a seguir os resultados dos testes com a ferramenta *ab*, direcionados para o tráfego de pacotes HTTP. Com base nesses testes, observa-se que, mesmo não havendo regras de filtragem para este tipo de tráfego dentre as regras inseridas nos testes, existe também o impacto na vazão deste tráfego, tornando-se, de mesmo modo, mais significativo à medida em que incrementa-se o número de regras de filtragem em nível de aplicação.

Os testes com o *ab* foram realizados com as mesmas medidas (quantidades de regras) utilizadas nos testes com o Iperf e Netperf, porém, em dois cenários distintos: para 1 e para 1000 requisições HTTP. Foram medidos, nestes testes, o tempo médio de resposta (em segundos) e a taxa de transferência em cada situação (em Kbytes por segundo).

A tabela 5.4 apresenta estes resultados, expressos graficamente nas figuras 5.6 e 5.7.

| | Tempo de Resposta (1 req) | Tempo de Resposta (1000req) | Taxa de Transferência (1 req) | Taxa de Transferência (1000 req) |
|---|---|---|---|---|
| sem regras | 0,1293 | 1,1849 | 842,12 | 562,60 |
| 50 regras | 0,1557 | 1,9712 | 807,34 | 501,19 |
| 100 regras | 0,5741 | 3,2570 | 773,40 | 374,65 |
| 500 regras | 1,1984 | 5,2741 | 401,17 | 193,23 |
| 1000 regras | 2,2198 | 7,4156 | 154,96 | 65,06 |
| 3000 regras | – | – | – | – |

Tabela 5.4: Resultado dos Testes de Vazão de Tráfego HTTP

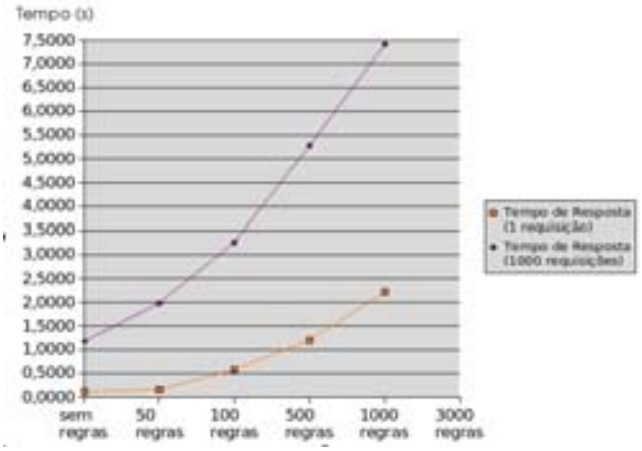


Figura 5.6: Testes com o *ab* :: Tempo de Resposta

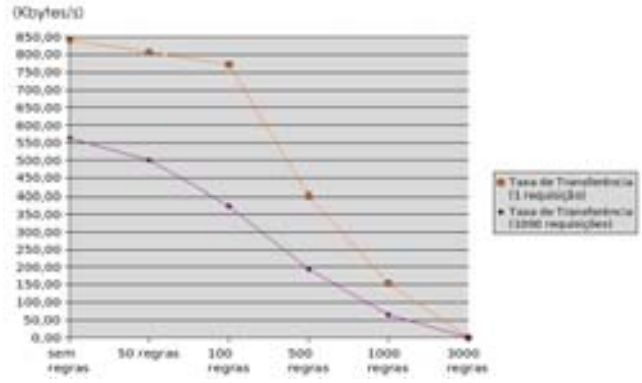


Figura 5.7: Testes com o *ab* :: Taxas de Transferência

Os resultados da ferramenta *ab* ratificam os resultados anteriores apresentados pelas ferramentas Iperf e Netperf. A vazão no *firewall* principal diminui gradativamente com o incremento no número de regras, até que, com 3.000 regras, o esgotamento de recursos da máquina faz com que a ferramenta não apresente resultados já que a solicitação do *ab* não chega ao servidor pois o *firewall* principal não consegue realizar o repasse dos pacotes.

A figura 5.6 apresenta os resultados de tempo de resposta (que tende a infinito quando o número de regras chega a 3.000), enquanto a figura 5.7 apresenta as taxas de transferência, que chegam a zero neste mesmo momento.

O tráfego de pacotes sem relação com as regras de filtragem em nível de aplicação, portanto, é diretamente afetado pela inserção de regras deste tipo no *firewall* principal. O esgotamento dos recursos da máquina pela inserção destas regras específicas, em determinado momento (neste exemplo ao aproximar-se de 3.000 regras inseridas) é capaz de interromper o repasse destes pacotes para o destino.

### 5.2.2 Com Uso de uma SuRFE

Os testes apresentados nesta seção medem o tempo médio gasto pelos pacotes que não necessitam ser submetidos a regras específicas para cruzar o *firewall* principal em um cenário com a implementação de uma SuRFE, isto é, estando o *firewall* principal carregado apenas com regras de filtragem e re-roteamento.

A finalidade da realização desses testes é provar os objetivos 2, 4 e 5, descritos na seção 4.1, que se referem aos aspectos do atraso sofrido pelos pacotes submetidos a filtragens específicas.

*Testes com as Ferramentas Iperf/Netperf*

Os resultados apresentados pelas ferramentas Iperf e Netperf, a exemplo dos testes da seção anterior, também foram semelhantes (diferenças percentualmente desprezíveis). Assim, para a apresentação dos resultados destes testes de vazão, optou-se por utilizar a média dos resultados das duas ferramentas.

A tabela 5.5 apresenta os resultados obtidos da vazão de dados no *firewall* principal quando incorporam-se as regras de filtragem em nível de aplicação na SuRFE e não diretamente sobre o mesmo, medidos em horários de maior e menor utilização.

| | Vazão Média em horários de menor utilização | Vazão Média em horários de maior utilização |
|---|---|---|
| sem regras | 245 Mbps | 239 Mbps |
| 50 regras | 245 Mbps | 239 Mbps |
| 100 regras | 245 Mbps | 239 Mbps |
| 500 regras | 245 Mbps | 239 Mbps |
| 1000 regras | 245 Mbps | 239 Mbps |
| 3000 regras | 245 Mbps | 239 Mbps |

Tabela 5.5: Resultado dos Testes de Vazão no *Firewall* Principal (com SuRFE)

Conforme era de se esperar, movendo-se as regras de filtragem específicas do *firewall* principal para uma SuRFE, a vazão de dados no *firewall* principal não mais depende do número de regras inseridas. Seu valor só é função das regras de filtragem tradicionais e das regras destinadas ao re-roteamento, cujo número é fixo para uma determinada política de segurança.

### 5.2.3 Análise Comparativa dos Resultados

Conforme pode-se observar nos gráficos das figuras 5.8 e 5.9, enquanto a vazão média tende a zero com o aumento do número de regras específicas aplicadas ao *firewall* principal sem a utilização de SuRFE, com o uso de uma SuRFE não há impacto na vazão média dos pacotes que cruzam este *firewall* e não são re-roteados para a SuRFE.

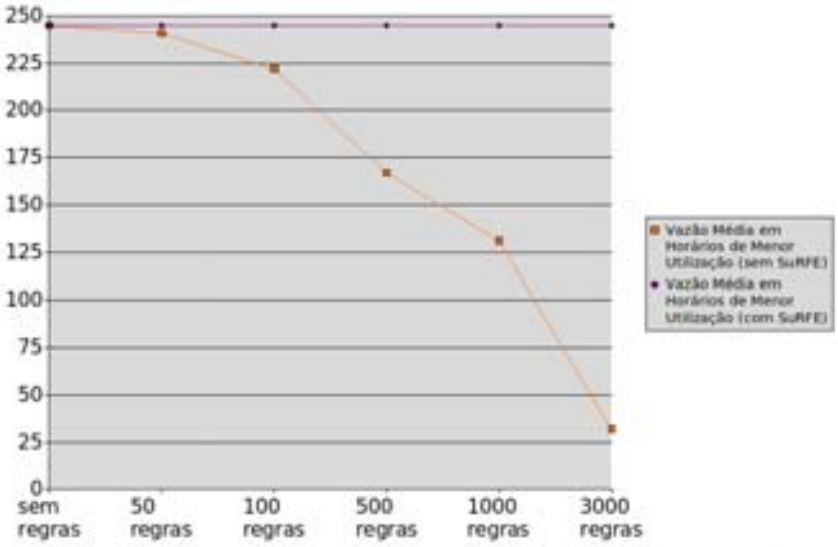


Figura 5.8: Vazão no *Firewall* Principal (com e sem SuRFE) em Horário de Menor Utilização

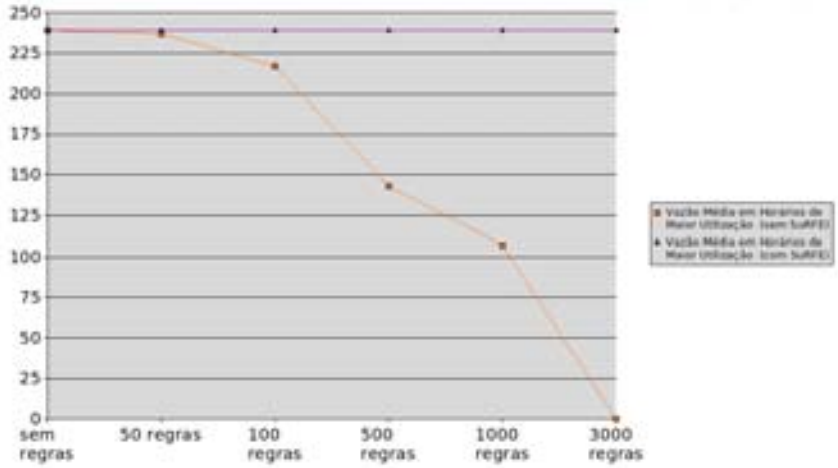


Figura 5.9: Vazão no *Firewall* Principal (com e sem SuRFE) em Horário de Maior Utilização

Esse é um dos grandes ganhos conseguidos com a utilização de uma SuRFE, pois garante-se um bom nível de vazão para os pacotes que não precisam sofrer filtragens específicas, deslocando-se o tratamento dos outros pacotes para fora do *firewall* principal.

Assim, o aumento do tráfego de pacotes que precisa sofrer filtragens específicas pode se resolvido melhorando-se a capacidade da SuRFE, com a introdução de novas máquinas, conforme sugerido na seção anterior.

## 5.3 Testes de Atraso na Transmissão de Pacotes

O objetivo dos testes descritos nesta seção é mensurar o atraso adicional aplicado aos pacotes que são re-roteados pelo *firewall* principal à SuRFE para a realização de filtragens específicas. Para isso fez-se uso da ferramenta *ab* para medir o tempo de resposta e taxa de transferência no ambiente testado.

Para que elementos de hardware não influenciassem na comparação de resultados, a máquina utilizada para os testes, na qual foram implementadas as regras de filtragem, possuía as mesmas características do *firewall* principal. O objetivo principal dos testes desta seção não era o de saturar a máquina, mas aplicar a mesma quantidade de regras de filtragens específicas e medir a vazão dos pacotes na SuRFE comparando os resultados à vazão apresentada na seção 5.2.2. Por isso, só foram inseridas até 1000 regras, condição que não implicou em comprometimento significativo na carga das máquinas.

A tabela 5.6 apresenta os resultados da ferramenta *ab* nos mesmos cenários da seção 5.2.2, isto é, para 1 e para 1000 requisições HTTP.

| | Tempo de Resposta (1 req) | Tempo de Resposta (1000req) | Taxa de Transferência (1 req) | Taxa de Transferência (1000 req) |
|---|---|---|---|---|
| sem regras | 0,1486 | 1,3620 | 715,11 | 471,82 |
| 50 regras | 0,1792 | 2,2690 | 681,10 | 429,19 |
| 100 regras | 0,6610 | 3,7443 | 652,49 | 321,17 |
| 500 regras | 1,3811 | 6,0675 | 343,93 | 164,83 |
| 1000 regras | 2,5493 | 8,5201 | 134,52 | 52,41 |

Tabela 5.6: Resultado dos Testes de Vazão de Tráfego HTTP com uso de SuRFE

### 5.3.1 Análise Comparativa dos Resultados

Apresenta-se a seguir a comparação dos resultados apresentados através dos gráficos mostrados nas figuras 5.10 a 5.13.

A análise dos resultados das medições do tempo de resposta, tanto para 1 como para 1000 requisições, apresenta um aumento médio percentual de 15% no tempo de resposta.

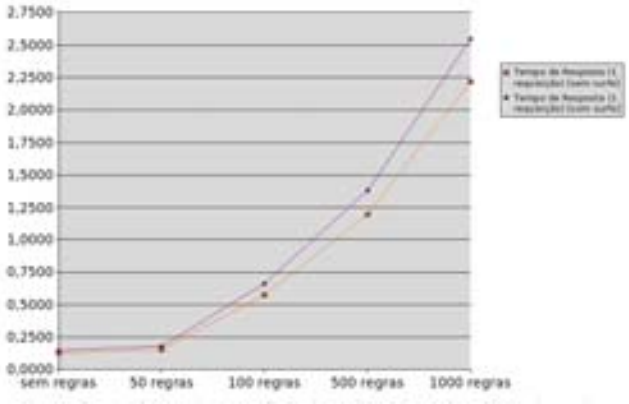


Figura 5.10: Tempo de Resposta (1 requisição) :: Com e sem SuRFE

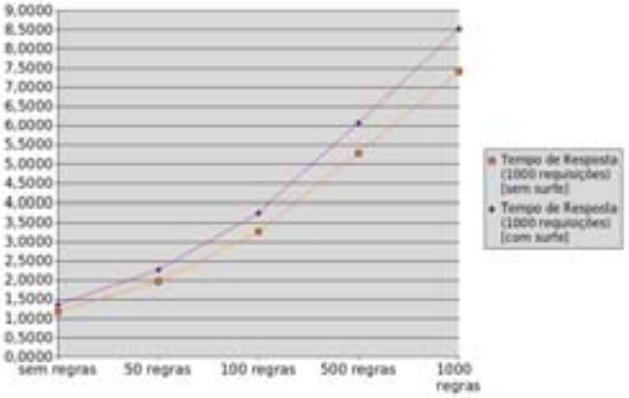


Figura 5.11: Tempo de Resposta (1000 requisições) :: Com e sem SuRFE

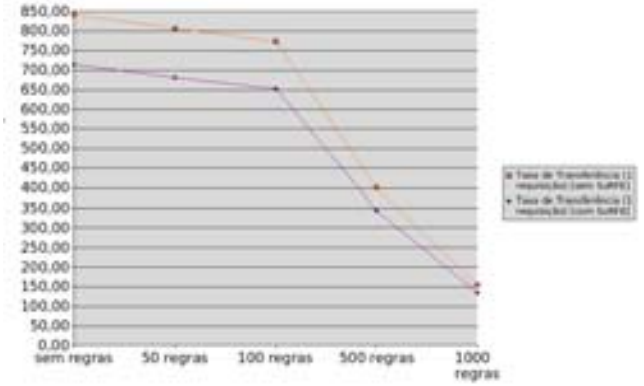


Figura 5.12: Taxas de Transferência (1 requisição) :: Com e sem SuRFE

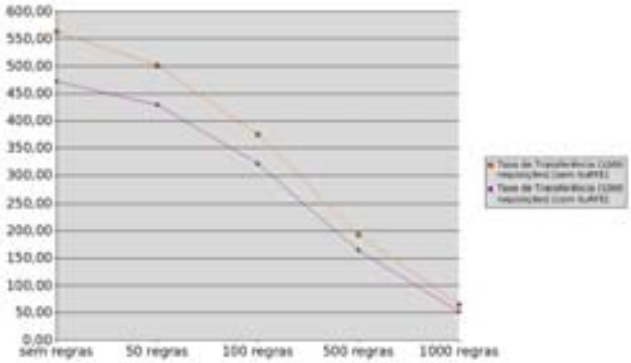


Figura 5.13: Taxas de Transferência (1000 requisições) :: Com e sem SuRFE

Este mesmo percentual médio é observado na diminuição das taxas de transferência medidas para 1 e para 1000 requisições, quando comparados os resultados sem e com o uso de SuRFE.

Este percentual é justificado pelo incremento nos elementos de filtragem (a SuRFE se constuti em mais um hop no tráfego dos pacotes), e pelas regras de re-roteamento de/para a SuRFE sobre os pacotes que necessitam de filtragens específicas.

# Capítulo 6

# Conclusões e Trabalhos Futuros

Neste capítulo são apresentadas as contribuições decorrentes deste trabalho, as conclusões a que se chegou após o desenvolvimento do mesmo e as perspectivas de trabalhos futuros.

## 6.1 Contribuições e Conclusões

Este trabalho apresentou a SuRFE (Sub Rede de Filtragens Específicas), uma solução para minimizar o impacto causado pela filtragem de pacotes em nível de aplicação sobre o volume de dados que atravessa o *firewall*, através de uma técnica de desvio de apenas parte deste tráfego para análises específicas.

Com a implementação de uma SuRFE é possível tratar um determinado fluxo de dados, com um maior nível de detalhes, na filtragem em busca de padrões em todos os níveis da pilha TCP/IP.

A análise de resultados, demonstra que o impacto causado pela sobrecarga de recursos do *firewall* principal aos pacotes não objeto de filtragem de regras específicas (notadamente filtros em nível de aplicação) pode ser significativo, e que o uso de uma SuRFE pode amenizar, e até praticamente anular os efeitos destas regras, ao realizar o re-roteamento de tráfegos específicos para esta sub-rede de filtragens específicas.

O re-roteamento de pacotes para uma SuRFE, porém, implica em transferir do *firewall* principal para a SuRFE os filtros específicos e as conseqüências deste nível de filtragem (observadas nos testes sem o uso da SuRFE). É necessário considerar a implementação dos cenários sugeridos (e não testados) neste trabalho para tratar situações em que o número de filtros implique em comprometimento da SuRFE (no modelo como foi testado). O incremento no número de máquinas da SuRFE, o uso de balanceamento de carga e/ou divisão de filtros por tipo (dentro da SuRFE) podem resolver os problemas de comprometimento de recursos de hardware transferidos do *firewall* principal para a SuRFE.

O aumento médio de 15% observado no tempo de resposta nos testes com a SuRFE, conforme os testes realizados, incidem somente sobre os pacotes que cruzam a SuRFE, não impactando diretamente no tráfego dos demais pacotes. Mesmo este retardo sobre o tráfego filtrado justificam, ainda, o uso da SuRFE pela diminuição na carga de processamento do *firewall* principal.

Finalmente, os benefícios à vazão de tráfego não filtrado, (ou com níveis de filtragem

menos específicos), porém, fazem da SuRFE uma alternativa fortemente recomendável, principalmente em redes com alto volume de tráfego, necessidades de filtragens específicas e sobrecarga do *firewall* principal.

Os testes, principalmente os realizados em ambiente de produção, corresponderam ao esperado, comprovando na prática que a utilização da solução apresenta resultados significativos na resolução de problemas de comprometimento de *firewalls* pela necessidade de filtragens específicas.

A SuRFE foi apresentada em artigo na *International Conference on Cyber Crime Investigation* (ICCyber'2006) e publicada em seus anais [Galvão & Fialho 2006].

## 6.2 Trabalhos Futuros

A implementação dos cenários adicionais apresentados no Capítulo 3.4.1 (Figura 3.4 e 3.5) com o balanceamento de carga e separação por tipo de filtragem incorporados à SuRFE convencional é uma proposta de trabalho futuro, dando continuidade a este trabalho.

A aplicabilidade da SuRFE também pode ser melhorada com o aumento do número de serviços/protocolos filtrados na SuRFE, como tráfego de e-mail (filtragem de vírus e *spam*), ataques a serviços específicos, como FTP, DNS e SSH.

# Referências Bibliográficas


3Com (2006), Tipping point ips, Página na internet, 3Com Corporation.
**URL:** *http://www.tippingpoint.com/*

Anderson, J. P. (1972), Computer security technology planning study, Vol. 2, EUA: Electronic Systems Division, Bedford, Massachusetts, EUA.

Apache, Software Foundation (2006), ab - apache http server benchmarking tool, Página na internet.
**URL:** *http://httpd.apache.org/docs/2.0/programs/ab.html/*

Bell, D. E. (1974), Secure computer systems: A refinement of the mathematical model, Vol. 3, EUA: Electronic Systems Division, Bedford, Massachusetts, EUA.

Bell, D. E. & L. J. Lapadula (1973), Secure computer systems: Mathematical foundations, Vol. 1, EUA: Electronic Systems Division, Bedford, Massachusetts, EUA.

Bell, D. E. & L. J. Lapadula (1974), Secure computer systems: A mathematical model, Vol. 1, EUA: Electronic Systems Division, Bedford, Massachusetts, EUA.

Comer, D. E. (1995), Internetworking with tcp/ip. : Principles, protocols and architecture, Vol. 1, New Jersey, EUA.

ComScore, MediaMetrix (2006), Media metrix - measuring the digital world, Página na internet.
**URL:** *http://www.comscore.com/metrix/*

Cronkhite, C. & J. Mccullough (2001), Hackers, acesso negado, Vol. 1, Rio de Janeiro.

DoD (1985), Trusted computer system evaluation criteria ("the orange book"), Vol. 1, Department of Defense, Washington, EUA.

Emmons, J. (2006), Unix load averages explained, Página na internet.
**URL:** *http://www.lifeaftercoffee.com/2006/03/13/unix-load-averages-explained/*

Filho, J. E. M. & A. B. Araújo (2005), Hlbr - hogwash light br, Página na internet.
**URL:** *http://hlbr.sourceforge.net/*

Galvão, Ricardo Kléber Martins (2002), Uma ferramenta gráfica para filtragem de strings com netfilter/iptables, *em* 'Anais do SSI - Simpósio Segurança em Informática', Vol. 1, ITA/CTA, São José dos Campos, Brasil.

Galvão, Ricardo Kléber Martins & Sergio Vianna Fialho (2006), Surfe - sub-rede de filtragens específicas, Vol. 1, International Conference on Cyber Crime Investigation (ICCyber), Brasília, DF.

Gunther, N. (2003), Unix load average part 1: How it works, Página na internet.
**URL:** *http://www.teamquest.com/resources/gunther/display/5/index.htm/*

Hatch, B., J. Lee & G. Kurtz (2002), Hackers expostos - linux, Vol. 1, São Paulo, SP.

Jones, R. (2005), Netperf benchmark, Página na internet.
**URL:** *http://www.netperf.org/netperf/NetperfPage.html/*

Jonkman, Matt (2003), Bleeding edge threats, Página na internet.
**URL:** *http://www.bleedingsnort.com/*

Knobbe, Frank (2001), Snortsam, Página na internet.
**URL:** *http://www.snortsam.net/*

Kurose, J. F. & K. W. Ross (2003), Rede de computadores e a internet: uma nova abordagem, Vol. 1, São Paulo, SP.

Landwehr, C. E. (1981), A survey of formal models for computer security, Vol. 1, Washington, EUA.

Larsen, Jason (2001), Hogwash - host ids, Página na internet.
**URL:** *http://sourceforge.net/projects/hogwash/*

Marcelo, A., F. Cerqueira & F. Saraiva (2000), Linux : Ferramentas anti-hackers, Vol. 1, Rio de Janeiro, RJ.

Metcalf, W. & V. Julien (2006), Snort in-line, Página na internet.
**URL:** *http://snort-inline.sourceforge.net/*

Moreira, N. S. (2001), Segurança mínima - uma visão corporativa da segurança de informações, Vol. 1, Rio de Janeiro, RJ.

Nakamura, E. T. & P. L. Geus (2002), Segurança de redes em ambientes cooperativos, Vol. 1, São Paulo, SP.

Oetker, T. (2006), Mrtg - the multi router traffic grapher, Página na internet.
**URL:** *http://oss.oetiker.ch/mrtg/*

(pseudônimo do autor) Marcio (2000), A internet e os hackers - ataques e defesas, Vol. 1, São Paulo, SP.

Roesch, Martin (1998), Snort ids, Página na internet.
**URL:** *http://www.snort.org/*

Rufino, Nelson Murilo O. (2002), Segurança nacional - técnicas e ferramentas de ataque e defesa de redes de computadores, Vol. 1, São Paulo, SP.

Russel, Rusty (2001), Linux ipchains-howto, Página na internet.
**URL:** *http://tldp.org/HOWTO/IPCHAINS-HOWTO.html/*

Tirumaia, A., F. Qin, J. Dugan, J. Ferguson & K. Gibbs (2005), Iperf - a tool to measure network performance, Página na internet.
**URL:** *http://dast.nlanr.net/Projects/Iperf/*

Tittel, E. (2003), Teoria e problemas de rede de computadores, Vol. 1, Porto Alegre, RS.

Ware, W. H. (1979), Security controls for computer systems. report of defense science board task force on computer security, Vol. 1, California, EUA.

Watkins, S. & A. Calder (2003), It governance - a manager's guide to data security & bs 7799 / iso 17799, Vol. 2, EUA.

Zekauskas, M. & H. Uijterwaal (2006), Ietf - internet protocol performance metrics (ippm), Página na internet.
**URL:** *http://www.advanced.org/IPPM/*