Universidade Federal de Lavras

INTERFACE ADMINISTRATIVA PARA FIREWALL DE INTERNET EM AMBIENTE LINUX

Ari Palu Junior

2005

ARI PALU JUNIOR

INTERFACE ADMINISTRATIVA PARA FIREWALL DE INTERNET EM AMBIENTE LINUX

Monografia de Pós-Graduação apresentada à banca examinadora do Departamento de Ciência da Computação da Universidade Federal de Lavras, como parte das exigências para obtenção do titulo de Especialista em Administração de Redes Linux.

Orientador:
Prof. Joaquim Quintero Uchôa

Lavras
Minas Gerais - Brasil
2005

ARI PALU JUNIOR

INTERFACE ADMINISTRATIVA PARA FIREWALL DE INTERNET EM AMBIENTE LINUX

Monografia de Pós-Graduação apresentada à banca examinadora do Departamento de Ciência da Computação da Universidade Federal de Lavras, como parte das exigências para obtenção do titulo de Especialista em Administração de Redes Linux.

Aprovada em _____ de _____ de 2005.

_____
Prof. Douglas Machado Tavares

_____
Prof. Heitor Augustus Xavier Costa

_____
Prof. Joaquim Quintero Uchôa
(Orientador)

Lavras
Minas Gerais - Brasil
2005

DEDICATÓRIA

Dedico este trabalho aos meus pais, pois sempre estão ao me lado quando preciso, serei eternamente grato. A Deus, que proporcionou que eu chegasse a onde estou, e permanecerá comigo em todos os momentos da minha vida. E aos meus amigos administradores de redes, por criticarem construtivamente as idéias discutidas neste documento.

AGRADECIMENTOS

Aos meus Pais, Ari e Avanete, a minha namorada Kelley Janine, pois sempre que preciso estão ao meu lado nos momentos mais difíceis.

Aos meus amigos e colegas administradores de redes, que me incentivaram e ajudaram na coleta de informações para compor este documento, em especial aos amigos Fábio, vulgo *Gardenal Mam* e também ao Marcelo Gomes, popularmente conhecido como *Seu Barriga depois da Gripe.*

Aos professores, transmissores de conhecimento e notório saber.

Aos desenvolvedores des sistemas Open Source, pois permitem o uso de Softwares GNU em nossos computadores.

RESUMO

Esta monografia tem como base estudar e aplicar os conceitos de segurança em redes de computadores, a fim de tornar-las menos sujeitas a ataques. O mecanismo principal de segurança estudado nesta pesquisa é o *Firewall*. A fase de implementação visa projetar e implementar um protótipo de uma interface gráfica administrativa para *Firewall* de Internet baseado em ambiente operacional Linux.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS

ACK	Acknowledgment
ARPANET	Advance Research Projects Agency Network
DMZ	Delimitarized Zone
DNS	Domain Name Server
EGP	Exterior Gateway Protocol
FIN	Icmp
IGP	Interior Gateway Protocol
IHL	Internet Header Length
IP	Internet Protocol
ISO	International Standardization Organization
NAT	Network Address Tranlastion
OSI	Organization Standard International
OSPF	Open Shortest Path First
POSIX	Portable Operating System Interface For Computer Environments
PSH	Push
RIP	Routing Information Protocol
RST	Restart
SYN	Synchronous
TCP	Transfer Control Protocol
TCP/IP	Transfer Control Protocol / Internet Protocol
UDP	User Datagram Protocol
URG	Urgent

1 INTRODUÇÃO

Com o crescimento das redes locais de computadores e a sua interligação à Internet, rede mundial de computadores, a questão da segurança passou de um mero conceito a uma tecnologia amplamente aplicada e valorizada. As mais diferentes corporações preocupam-se e dispõem de quantias significativas de seus orçamentos para implantar um processo de segurança da informação dentro de seus ambientes, a fim de protegê-los das mais diversas eventualidades.

Um dos recursos mais utilizados atualmente por essas corporações para restringir com segurança o acesso de suas redes a Internet, é o *Firewall*. Este sistema de segurança, que traduzido do inglês representa uma parede de proteção ao fogo, localiza-se entre as duas redes como uma suposta parede que filtra todo o fluxo de informações entre estas redes [ZWICKY 2001].

Encontram-se hoje em dia diferentes arquiteturas de *Firewall* de fabricantes diversos para finalidades específicas. Estas arquiteturas variam, podendo ser um *hardware* proprietário ou até mesmo um *Firewall* implementando com ferramentas gratuitas em sistemas operacionais do padrão POSIX[1], como o popular Linux e sua ferramenta atual de filtragem de pacotes o *Netfilter*/*IPTables* [RUSSELL 2001].

Para construir um sistema de defesa, *Firewall*, é necessário antes determinar o que se precisa proteger, contra quem e quais as formas de

[1] POSIX: http://www.open-std.org/jtc1/sc22/WG15/

proteção disponíveis atualmente. Sendo assim, com este conhecimento, pode-se iniciar o processo de implantação de um sistema de segurança.

Dentro dos conceitos e preceitos de segurança, este Trabalho de Conclusão de Curso de pós-graduação tem como objetivo elaborar uma fundamentação teórica sobre o mecanismo de segurança *Firewall*, tendo como foco principal à ferramenta *Netfilter*/*IPTables*.

Está pesquisa também se dedica a criação de um protótipo de uma interface gráfica para administrar um Firewall em Linux, NetFilter/IPTables, disponibilizando por meio desta, recursos do modo texto de maneira prática e eficaz ao administrador de redes.

Como justificativa, esse estudo permite atestar que as diversas soluções a serem implantadas através dos mecanismos de segurança geram resultados positivos. Além disso, podem ser agregados mecanismos que agem interativamente com estes sistemas e os administradores de redes. Podendo ser uma solução prática e viável, através do uso desta interface gráfica, onde o administrador terá uma melhor praticidade e eficácia na gerência do *Firewall*.

No capítulo 2, são apresentados os conceitos sobre segurança em redes de computadores, assim como os potencias riscos que elas sofrem, definindo os pontos vulneráveis, os inimigos e as formas de proteção.

Em seguida, no capítulo 3, é apresentada uma conceituação sobre protocolos de comunicação de dados com ênfase no TCP, IP, UDP, ICMP e modelo OSI.

No capítulo 4, é abordado o mecanismo de segurança em redes *Firewall*, assim como suas principais características e arquiteturas.

No capítulo 5, são apresentadas uma conceituação de interfaces, com sua interação aos aspectos humanos e métricos de projetos. Neste mesmo capitulo, relaciona-se também o projeto de interfaces para *Web*.

Logo em seguida, no capítulo 6, é abordada a fase de implementação da interface proposta como projeto final, citando as suas características técnicas, assim como de uso.

No capítulo final, trata-se da conclusão desta monografia situando o objetivo alcançado e sugestões de melhoras futuras para o protótipo desenvolvido.

2 SEGURANÇA EM REDES DE COMPUTADORES

Quando se pensa em segurança, é necessário ter a seguinte premissa em mente: “Quanto maior o nível de segurança, menor o nível de privilégios“, conforme Soares [SOARES 2003]. Aplicando esta premissa para a segurança em redes de computadores, obtém se a conclusão que para uma rede ser o mais segura possível seu usuário irá possuir o menor número de recursos possíveis. Deste modo, ao estruturar políticas de segurança para um determinado ambiente computacional, é de suma importância estabelecer as reais necessidades de recursos que os usuários necessitam, para que conseqüentemente se ofereça a máxima capacidade em segurança.

2.1 Segurança em Redes Corporativas

Está claro, para todas equipes de gerência das empresas, que a Internet é uma ferramenta poderosa de relacionamento entre as mesmas e seus clientes. O que precisa ser abordado de forma tão clara, em conjunto com esta verdade, é a real necessidade da segurança das redes corporativas que são interligadas a Internet. O mundo corporativo possui valores e informações que necessitam de total privacidade e sigilo.

2.2 O que proteger

Ao abordar segurança de redes em ambientes corporativos, logo se pensa em três itens nos quais as organizações devem se preocupar em proteger [ZWICKY 2001]. São eles:

a) os dados, ou seja, as informações mantidas nos computadores;

b) os recursos, isto é, os próprios computadores;

c) a reputação da corporação.

2.2.1 Os dados

Quando se fala em dados de uma corporação existem três características que precisam ser abordadas. São elas:

a) sigilo: as informações não devem ser visualizadas por pessoas não autorizadas;

b) integridade: as informações não devem ser alteradas por pessoas não autorizadas;

c) disponibilidade: as informações devem estar disponíveis para as pessoas autorizadas.

2.2.2 Os recursos

Os cuidados que devem ser tomados com os recursos de uma corporação não são apenas os de proteger contra ataques de invasores que desejam apenas apagar arquivos de sistemas.

2.2.3 A reputação

Na maioria dos casos, as invasões acontecem por questões de cunho pessoal a fim de obter um benefício monetário ou satisfação pessoal. Um invasor se sentindo atraído pelo desafio de obter acesso a um sistema específico pode causar danos irreparáveis. Por tanto, a reputação é o fator principal ao implantar segurança em ambientes computacionais, sejam eles corporativos ou não [ZWICKY 2001].

2.3 Contra quem proteger

Diz-se que, ameaça é uma possível violação da segurança de um sistema. As principais ameaças às redes de computadores são:

a) destruição, remoção, perda de informação ou de outros recursos;

b) roubo e modificação da informação;

c) interrupção de serviços.

As ameaças podem ser divididas em acidentais e intencionais. Ameaças acidentais são as que não estão ligadas a uma intenção premeditada. As ameaças intencionais vão desde a observação de dados com ferramentas simples de monitoramento de redes a ataques sofisticados baseado no funcionamento do sistema. A formalização de uma ameaça intencional gera um ataque.

2.4 Tipos de ataques

Os ataques ou as ameaças são efetuados em uma série de estágios, usando várias ferramentas e técnicas. Conforme Perkins [PERKINS 2002], um ataque visando uma invasão consiste em:

- Engenharia social;
- Exploração de Erros e Vulnerabilidades;
- Ataques internos;
- Ataques via varredura de Endereços e portas;
- Recusa ou negação de serviços;
- Analise de trafego de origem;
- Roteamento de origem;
- Seqüestro de origem;

Essas formas serão detalhadas nas subseções a seguir.

2.4.1 Engenharia social

A engenharia social é uma das técnicas mais perigosas de obtenção de dados e difícil de ser combatida. Consiste em levantar informações vitais, sem que as pessoas percebam que possuem informações importantes.

Segundo Silva [SILVA 2004], é o termo utilizado para qualificar os tipos de intrusão não técnica. Freqüentemente, envolve a habilidade de enganar pessoas de modo que possam disponibilizar informações com o objetivo de violar os procedimentos de segurança.

Silva complementa, existem algumas medidas que podem atenuar a participação do componente humano, como:

- A conscientização das pessoas sobre o valor da informação;
- A segurança física nas dependências de uma organização;
- O uso de Política de Segurança para estabelecer procedimentos que eliminem quaisquer trocas de senhas;
- O Controle de Acesso aos usuários a fim de que estes possam ter acesso aos recursos computacionais e assim realizar suas atividades.

2.4.2 Exploração de erros e vulnerabilidades

A exploração de erros e vulnerabilidades de sistemas são um dos meios mais comuns utilizados para início de ataques. O monitoramento e constante pesquisa sobre vulnerabilidades dos sistemas instalados são as melhores maneiras de garantir a segurança em altos níveis. Fazem parte destes procedimentos a atualização de versões dos programas, a instalação de correções de segurança e o uso de sistemas de notícias sobre segurança.

2.4.3 Ataques internos

Ocorre quando usuários legítimos, funcionários propriamente ditos, comportam-se de modo não autorizado. Conhecido também como ataque direto, estes ocorrem objetivando o roubo de informações para vende-las ou por descontentamento perante a organização.

2.4.4 Ataque via varredura de endereços e portas

Através de uma varredura de endereços de rede pode se descobrir sistemas contaminados com programas os quais permitem conexões em determinadas portas, possibilitando até mesmo um controle remoto total da máquina, caso esta seja invadida.

2.4.5 Recusa ou negação de serviço

O ataque de negação de serviço, popularmente conhecido como *Denial Of Service (DOS)*, ocorre quando um *host,* computador cliente ou servidor, não executa sua função corretamente ou atua de forma que impeça que outros *hosts* executem suas funções. Os métodos mais usados para esse tipo de ataque são: ping da morte, redirecionamento de DNS ou redirecionamento de rota.

2.4.6 Análise de tráfego de rede

Programas que são capazes de capturar a atividade e informações codificadas da rede (tais como senhas e as transmissões que passam entre os servidores) são conhecidos como *Sniffers*. Segundo [MARCELO 2000], estes programas podem ser capazes de escutar outros equipamentos da rede registrando não apenas informações de conteúdo, mas também de origem e destino de pacotes podendo localizar *hosts* importantes na estrutura da rede pesquisada. [GERLACH 1999] complementa que quando estes programas estão instalados em Redes Ethernets, literalmente quaisquer dados que trafegam na rede podem ser capturados.

2.4.7 Roteamento de origem

O protocolo de comunicação de dados, TCP/IP, inclui uma opção pouco usada que permite enviar dados a partir de um computador fazendo-o parecer outro computador. Esta é uma técnica sofisticada de autenticar uma máquina para outra, forjando pacotes de um endereço de origem confiável, conhecido também como *Spoofing*.

2.4.8 Seqüestro de conexões

Nesta técnica, o invasor opera entre dois computadores na rede, um cliente e um servidor. Ao abrir uma conexão entre o cliente e o servidor, o computador do invasor à detecta e simula ser ambos os lados.

Para o cliente, ele recebe os dados a serem enviados para o servidor e de posse desses, se conecta e recebe do servidor os dados que deveriam ser enviados aos clientes e os repassa de volta ao cliente. Deste modo, ele administra todo o tráfego da conexão e obtêm na maioria dos casos dados que possibilitam novas invasões a esse servidor. Esta técnica também é conhecida como homem no meio ou *Hijacking Attack*.

2.5 Tipos de hackers

O adjetivo *hacker* emprega-se a qualquer indivíduo que tenha conhecimento e apreço em saber detalhadamente todas as questões envolvidas com o universo dos computadores. Atualmente, este adjetivo foi deturpado em conseqüência da massificação de ataques na Internet por alguns desses especialistas em informática. Em se tratando de indivíduos que efetuam ataques a alvos na Internet, a melhor classificação para estes seria a de *cracker*, que nada mais é que um hacker dotado dos mesmos conhecimentos só que com propósitos ilegais [ANONIMO 2000].

Aprender e praticar a arte do *hacking* é uma atividade que leva tempo, até anos, dependendo dos objetivos. A palavra *hacker* traz a mente o estereótipo de adolescentes magros e unicamente interessados no brilho de seu monitor de computador. Notoriamente, esses indivíduos são a maior parcela dos milhões de *hackers* que existem ao redor do mundo, mas não são a maior ameaça. Portanto, consideram-se apenas dois tipos sérios de *hacker*, o subempregado e os que são pagos para efetuar tal atividade [PERKINS 2002].

Segundo Perkins [PERKINS2002], os *hackers* se enquadram nas seguintes categorias:

- Especialista em segurança;
- Hackers estudantes;
- Hackers adultos subempregados;

- Hackers criminosos;
- Funcionários recentidos;

Essas categorias serão detalhadas nas subseções a seguir.

2.5.1 Especialista em segurança

A maioria destes possuem conhecimentos suficientes para efetuar com sucesso a invasão de um sistema computacional em rede. Mas, descobriram que existe um mercado abrangente e bem remunerado quando aplicado seus conhecimentos para a segurança e proteção corporativa.

2.5.2 Hackers estudantes

Tais indivíduos são classificados como estudantes perante sua situação econômica, visto que estudam ou participam de algum projeto escolar e, conseqüentemente, são subsidiados por suas famílias. Geralmente sua atividade estudantil está relacionada a algum curso que envolve a área de computação. A ameaça que podem provocar não consiste em ataques tão sofisticados, pois na maioria das vezes utilizam recursos e ferramentas criadas por *hackers* mais sérios. Um sistema sério e eficaz de proteção e segurança é capaz de manter esses *hackers* à distância.

2.5.3 Hackers adultos subempregados

A maioria das informações e ferramentas usadas pelos diversos *hackers* são produzidas por este grupo. Estes *hackers* atuam na busca de se tornarem imortais na comunidade *hacker* a fim de impressionar seus colegas com desafios realizados e informações conseguidas. Criados a partir de uma questão social, pois a grande parte destes provêm de países que possuem alta qualidade educacional e condições econômicas que não possibilitam estarem empregados, estes na maioria das vezes estão apenas a procura de uma atividade que lhe traga remuneração.

2.5.4 Hackers criminosos

Do ponto de vista social, estes se enquadram como qualquer criminoso real, pois vão as buscas do que desejam não importando os meios e as eventuais conseqüências geradas a partir de seus atos. Exercem essa atividade por vingança e lucro monetário.

2.5.5 Funcionários ressentidos

O mais perigoso de todos os indivíduos com intenção de *hackear* uma corporação são os seus funcionários, possuem tanto os meios quanto os motivos para prejudicar seriamente uma rede de

computadores, enquadram-se no tipo de incidente de maior dificuldade de ser detectado antes de acontecer. Os ataques promovidos por esse grupo são dos mais variados, indo de um administrador de redes mal intencionado, que inspeciona sem autorização a correspondência eletrônica, a um faxineiro, que desliga ou interrompe a comunicação de dados através de um cabo de transmissão ou equipamento de rede.

2.6 Formas de Proteção

2.6.1 Política de segurança

Política de segurança é um conjunto de leis, regras e práticas que regulam como uma organização administra, protege e distribui suas informações e recursos. Ainda que os ataques de *hackers* sejam muito comentados e estejam em evidência, eles não são tão numerosos quanto os problemas de segurança que ocorrem dentro das empresas [NBSO 2003].

A implementação de uma política de segurança baseia-se na aplicação de regras que limitam o acesso de um usuário a informações e recursos, com base na comparação do seu nível de autorização relativo a essa informação ou recurso. Assim, a política de segurança define o que é e o que não é permitido em termos de segurança durante operações de um sistema. É necessário adotar uma forte política de segurança em uma rede para evitar ocorrências de ataques, de modo a

proteger as informações, considerando que elas constituem, muitas das vezes, o maior patrimônio da empresa. O bom funcionamento da política de segurança depende muito do conhecimento e participação dos usuários [PENTA 1999].

2.6.2 Plano de contingência

As perdas referentes a qualquer problema, demonstram que as empresas deveriam ter atuado de alguma maneira e não o fizeram. Isto é um ponto negativo aos olhos dos clientes e fornecedores e péssimo para acionistas e funcionários. Plano de contingência é uma alternativa segura para resolver problemas que estão prejudicando as atividades da comrporação, ações simples que vão desde informações sobre equipamentos e programas, até o número do telefone particular dos funcionários e as ações a serem efetuadas em momentos críticos [CARUSO 2001].

2.6.3 Criptografia

Para que seus segredos permaneçam seguros, é necessário adicionar proteções ao computador na qual não são fornecidas pelo Sistema Operacional. Sendo assim, [Burnett 2002] diz que a criptografia é um método usado para transformar arquivos legíveis em algo ilegível. Pode ser usada para codificar dados e mensagens antes que esses sejam enviados por vias de comunicação, mesmo que sejam

interceptados, dificilmente possam ser decodificados (decifrados). Para garantir a privacidade, são usados princípios como um algoritmo que funciona como uma fórmula ou uma função matemática que converte os dados originais em um texto cifrado. Esses algoritmos dependem de uma variável chamada chave, que é fornecida pelo usuário e funciona como uma senha, pois somente de posse dela será possível decifrar o texto [TANENBAUM 2001].

2.6.4 Controle de acesso

Os mecanismos de controle de acesso são usados para garantir que o acesso a um recurso seja limitado aos usuários devidamente autorizados. As técnicas incluem a utilização de listas ou matrizes de controles de acesso, que associam recursos a usuários autorizados ou senhas, cuja posse determina os direitos de acesso do usuário que a possui. No Linux, o *root* é o usuário que decide a maioria das permissões do sistema, que podem ser identificadas simplesmente listando uma estrutura de diretórios. Geralmente, arquivos de configuração do sistema para operação da rede, por padrão, somente o *root* é quem tem permissão [ANÔNIMO 2000].

2.6.5 Firewall

A implementação de um *Firewall* é usada para criar um ponto de controle de segurança na fronteira de uma rede privada. Ao exercer a

atividade de roteamento entre a rede privada e as demais redes, os *Firewalls* inspecionam toda a comunicação, permitindo ou não o tráfego de pacotes, de acordo com as regras pré estabelecidas [MARCELO 2002].

Os *Firewalls* dividem-se em duas categorias básicas:

a) *firewalls* de nível de rede ou filtro de pacotes;

b) *firewalls* de *gateway* de aplicativo ou serviços de *proxy*.

O assunto *Firewall* será tratado mais detalhado no Capitulo 4.

3 PROTOCOLOS

Os protocolos de comunicação de dados são aplicados para coordenar a troca de informações entre dispositivos de rede. Estes estabelecem o mecanismo pelo qual cada dispositivo reconhece as informações úteis de outros dispositivos alojados na rede. Para que essas informações possam trafegar através de uma rede, elas precisam ser agrupadas em um pacote de dados lógico, conhecido apenas como pacote de dados. Este pacote é um fluxo de *bytes* que associado a ele tem-se um cabeçalho e um corpo. Os cabeçalhos contem informações como origem, destino, tamanho, e tipo de pacote. O corpo do pacote contém os dados que se deseja transmitir através da rede [RAY 1996].

3.1 Modelo de Comunicação em Camadas

Um sistema de comunicação é responsável por comunicar os vários nós de um sistema distribuído. A partir desse sistema, é possível que qualquer nó de uma determinada rede possa transmitir e receber informações para um outro nó ou ponto desta rede. Estes sistemas normalmente são muito complexos, dessa maneira eles são divididos em camadas, afim de que cada camada se torne responsável por uma etapa deste processo [RAY 1996].

3.2 Modelo de Comunicação OSI

O modelo de referência *Open Systems Interconection* – OSI, foi desenvolvido pela ISO, *International Standardization Organization*, como um modelo para a arquitetura de um protocolo de comunicação de dados entre dois computadores.

Conforme [SOARES 2003], o modelo OSI é composto de sete camadas apresentadas a seguir:

1. **física:** Camada responsável pela transmissão de uma seqüência de bits de forma não estruturada em um meio físico. Trata das características mecânicas, elétricas, funcionais e procedurais para acessar o meio físico;

2. **enlace de dados:** Camada responsável pela transmissão confiável de informação através do enlace físico. Envia blocos de dados, pacotes, com a necessária sincronização, controle de erro e de fluxo;

3. **rede:** Camada que fornece para as camadas superiores independência das tecnologias de transmissão e comutação usadas para conectar os sistemas. Responsável por estabelecer, manter e terminar conexões;

4. **transporte:** Camada responsável pela transferência de dados entre dois pontos de forma transparente e confiável

com funções como controle de fluxo e correção de erro fim a fim;

5. **sessão:** Camada que provê a estrutura de controle para a comunicação entre as aplicações. Estabelece, gerencia e termina conexões (sessões) entre aplicações;

6. **apresentação:** Camada responsável por prover independência aos processos de aplicação das diferenças na representação dos dados;

7. **aplicação:** Camada que fornece aos usuários acesso ao ambiente OSI e provê sistemas distribuídos de informação.

7 - Camada de Aplicação
6 - Camada de Apresentação
5 - Camada de Sessão
4 - Camada de Transporte
3 - Camada de Rede
2 - Camada de Enlace
1 - Camada Física

Figura 1: Modelo OSI [SOARES 2003].

3.3 Modelo TCP/IP

O TCP/IP, TCP – *Transfer Control Protocol* ou protocolo de controle de transferência e IP – *Internet Protocol* ou protocolo de Internet, é uma família de protocolos usados para comunicação em redes de computadores. Esta arquitetura surgiu em 1975 na rede Arpanet sendo a percussora da Internet. Uma das principais características do TCP/IP é que suas especificações são públicas e genéricas, permitindo, dessa forma, sua implementação por diversos fabricantes [CYCLADES 2000].

3.3.1 IP

O IP é o protocolo que atua na camada de rede, provendo as funções necessárias de roteamento dos dados entre as várias redes interconectadas. Ele é responsável por fazer a troca de pacotes de um modo mais simples não oferecendo nenhum serviço adicional além do roteamento, exceto por alguns recursos muito simples para checagem da entrega dos pacotes ou da integridade do conteúdo recebido [MARQUEZ 2001].

O protocolo IP é um serviço de comunicação sem conexão direta entre dois pontos finais, havendo várias rotas para a ligação. Por causa deste fator, é possível que haja gargalos de comunicação, gerando engarrafamento e perda de pacotes. Outro fator que pode ocorrer por não ser orientado a conexão é a chegada de pacotes fora de ordem,

quando o datagrama IP é fragmentado. A falta de controle de fluxo também constitui uma de suas deficiências. Por estar em um camada acima da camada de enlace, modelo OSI, o IP oculta as redes pelas quais ele passa, facilitando sua instalação e tornando-o mais robusto [MARQUEZ 2001].

3.3.2 O datagrama IP

Segundo [TANENBAUM 2001], as unidades de transferência de dados gerenciadas pelo IP são conhecidas como datagamas. Estas unidades contêm os dados recebidos das camadas superiores e os dados do cabeçalho IP. O cabeçalho IP possui as informações de controle acrescentadas na camada Internet para que o protocolo IP possa interpretar ao executar suas funções. A seguir, a Figura 2 ilustra o formato de um datagrama IP:

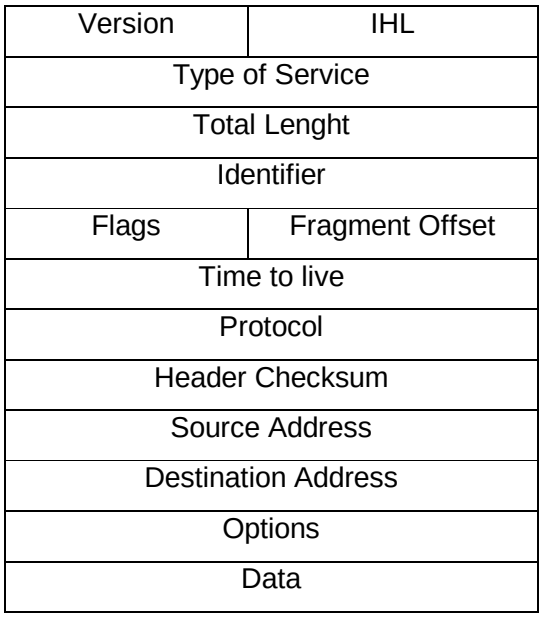

Figura 2: Datagrama IP [TANENBAUM 2001].

Os campos que compõem um cabeçalho IP são descritos a seguir:

a) *version* – indica a versão do protocolo em uso, atualmente a versão 4;

b) IHL – *Internet Header Lenght*: informa o comprimento do cabecalho IP em unidades de 32bits;

c) *type of service* – tipo de serviço: indica a qualidade de serviço requerida pelo datagrama;

d) *total length* – comprimento total: informa o comprimento total do datagrama em bytes, incluindo o cabeçalho do IP;

e) *identifier* – identificador: é número único com propósito de orientar a recomposição dos datagramas fragmentados;

f) *flags* – indica os atributos relativos à fragmentação dos datagramas;

g) *fragment offse*t – indica o deslocamento de blocos no datagrama;

h) *time to live* – tempo de vida: indica o tempo em segundos no qual um datagrama permanece válido antes de ser descartado;

i) *protocol* – protocolo: indica o protocolo da camada superior para o qual os dados contidos no datagrama devem ser passados;

j) *header checksum* – checagem de cabeçalho: indica a integridade do cabecalho IP em nível de bit;

k) *source address* – endereço de origem: indica o endereço de origem do host;

l) *destination address* – endereço de destino: indica o endereço de destino do host;

m) *options* – opções: indica configurações relacionadas a opções de controle;

n) *data* – dados: indica o dados a serem trafegados.

3.3.3 Endereçamento IP

Na versão mais utilizada atualmente, IPv4, o endereço IP é um número composto de 32 bits, divididos em 4 conjuntos de 8 bits cada, que é atribuído a cada *host* um endereço único, conforme visto na Tabela 1. De acordo com Soares [SOARES 2003], cada conjunto desse pode ser representado de forma decimal ao invés da forma de bits, assumindo valores de 0 a 255.

1 7	8 15	16 23	24 32
1100 1000	1111 0001	0011 1000	0001 1001
↓	↓	↓	↓
200	241	120	25
⇓			
200.241.120.25			

Tabela 1: Exemplo de endereçamento IP [SOARES 2003].

Uma primeira parte desses bits é usada para identificar a rede à qual o *host* está conectado e a parte restante é usada para identificar o *host* na rede. Os endereços IPs foram divididos em cinco classes, definidas pelos bits iniciais do endereço, conforme Tabela 2:

Endereço de Rede	Classe
0	A
10	B
110	C
1110	Multicast
1111	Reservado

Tabela 2: Divisão do IP em classes [SOARES 2003].

Além das cinco classes, têm-se ainda três faixas de endereços reservados para as redes privadas, ou redes inválidas, sendo estes endereços ignorados pelos roteadores. Conhecidos também como IPs falsos, os mesmos são os seguintes:

a) de 10.0.0.0 a 10.255.255.255 ➔ Classe A;

b) de 172.16.0.0 a 172.31.0.0 ➔ Classe B;

c) de 192.168.0.0 a 192.168.255.255 ➔ Classe C.

Analisando a forma com que os endereços IPs são distribuídos para as redes e seus respectivos *hosts*, podemos analisar o quadro a seguir:

	Redes	Hosts
Classe A	256	16.777.214
Classe B	65536	65534
Classe C	16.777.216	254

Tabela 3: Análise do número de hosts e redes [SOARES 2003].

3.3.4 Sub-redes

É possível ainda delimitar uma rede diminuindo-a em sub-redes. As sub-redes são redes sem classe e podem ser criadas baseadas na informação do endereço de difusão diferentes. Estes endereços são delimitados pelas máscaras de difusão de rede ou máscaras de rede. A mascara de rede define quantos bits são utilizados para o endereço de rede e quantos bits são utilizados para especificar o endereço de *hosts* dentro dessa sub-rede [CYCLADES 2000].

Além do endereço IP, os *hosts* da rede passam a ter uma máscara de rede. A máscara de rede tem o mesmo formato do endereço IP, possuindo 32 bits, representado por quatro números decimais separados por ponto. Caso não seja usada a notação de sub-redes, aplica-se a máscara padrão da rede seja esta da classe A, B ou C. A seguir as máscaras referentes a cada uma dessas classes:

Classe	Máscara	Nº de bytes para rede	Nº de bytes para hosts
A	255.0.0.0	1	3
B	255.255.0.0	2	2
C	255.255.255.0	3	1

Tabela 4: Máscaras de acordo com a classe da rede [SOARES 2003].

3.3.5 Roteamento

As redes precisam interconectar-se entre si, e para isso é aplicado o conceito de roteamento. As redes podem interligar-se por meio de nós comuns, que são chamados de roteadores. Os roteadores possuem múltiplas interfaces de redes, conectadas cada uma delas a diferentes redes. Estes por sua vez possuem uma tabela de endereçamento de redes, na qual os pacotes devem consultar para alcançar o endereço de destino, conhecido como tabela de roteamento [PERKINS 2002].

Quando as tabelas de roteamento são elaboradas e atualizadas manualmente, diz-se que este tipo de roteamento é estático. Já quando os roteadores trocam informações entre si a fim de montar suas próprias tabelas classifica-se este tipo de procedimento como roteamento dinâmico.

O roteamento estático possui rota fixa, sendo que sua tabela uma vez criada não é mais alterada. As rotas são fixas e caminhos alternativos são tomados só em caso de falhas. Por ser bastante simples este método pode efetuar uma má utilização do meio de comunicação, a não ser que o tráfego da rede seja bem regular e bastante conhecido.

Em contrapartida o roteamento dinâmico, por sua vez, é mais adequado a ambientes que possuem várias redes interconectadas sendo que estes ambientes são mais complexos em nível de gerência

de rotas. Para que haja o roteamento dinâmico é necessária a constante atualização da tabela de rotas do roteador. Essa atualização é feita através de protocolos de roteamento, os quais se classificam de acordo com o algoritmo de mapeamento das rotas em *distance vector* e *link state*. Além dessa classificação os protocolos de roteamento podem se qualificar de acordo com a sua amplitude na rede, podendo ser interna, IGP – *Internal Gateway Protocol* ou externa, EGP – *Exterior Gateway Protocol.*

3.3.6 TCP

O TCP é um protocolo confiável, baseado em conexão e encapsulado no IP. Este garante a entrega dos pacotes, assegura o sequenciamento dos pacotes, e providenciam um *checksum* que valida tanto o cabeçalho quanto os dados do pacote. No caso da rede perder ou corromper um pacote TCP/IP durante a transmissão, é tarefa do TCP retransmitir o pacote que falta [RAY 1996].

Porém, essa confiabilidade tem um preço. Os cabeçalhos dos pacotes TCP requerem o uso de bits adicionais para assegurar o correto sequenciamento da informação, bem como um *checksum* obrigatório para garantir a integridade do cabeçalho e dos dados. Para garantir a entrega dos pacotes, o protocolo também requer que o destinatário informe o recebimento do pacote. Esta informação de recebimento,

ACKs de *acknowledgments*, gera tráfego adicional na rede, diminuindo a taxa de transferência de dados em favor da confiabilidade .

O uso dos dois protocolos juntos – IP e TCP – formam a combinação mais conhecida na rede Internet: TCP/IP. Por serem os dois principais protocolos de comunicação na Internet, a maioria dos serviços foram escritos e funcionam utilizando as implementações e funcionalidades do TCP/IP [RAY 1996].

O protoloco TCP é utilizado em diversos serviços pelas camadas superiores de comunicações, onde os mais utilizados são:

a) gerenciamento orientado à conexão;
b) transferência de dados com segurança;
c) re-seqüência de envio;
d) controle de fluxo;
e) multiplexação;
f) precedência e segurança;
g) encerramento de conexões.

3.3.7 O cabeçalho do segmento TCP

Os *hosts* que trafegam dados TCP trocam os mesmos na forma de segmentos. Um segmento, conforme Tanenbaum [TANENBAUM 2001], consiste em um cabeçalho fixo de 20 bytes acrescido da parte de dados. O cabeçalho TCP apresenta o seguinte formato:

Source Port								Destination Port
Sequence Number								
Acknowledgment Number								
Data Offset	Reserved	URG	ACK	PSH	RST	SYN	FIN	Window
Checksum								Urgent Pointer
Options								Padding
Data								

Figura 3: Cabeçalho TCP [TANENBAUM 2001].

Os campos acima citados são detalhados a seguir:

a) *source port* – porta de origem – indica a aplicação que está disparando a conexão;

b) *destination port* – porta de destino – indica a aplicação onde serão requisitadas informações no destino;

c) *sequence number* – número de seqüência – contém o número de seqüência do primeiro octeto no campo de dados do usuário. O seu valor especifica a posição para o envio das próximas informações ao receptor;

d) *acknowledgment number* – número de conhecimento – é usado para confirmar o recebimento de dados. Seu valor é o próximo número de seqüência esperado pelo receptor que deve ser enviado pelo emissor;

e) *data offset* – deslocamento de dados – indica onde começa a parte de dados no segmento TCP. Separa cabeçalho e dados;

f) *reserved* – reservado – é reservado para uso futuro de correções de eventuais erros do protocolo;

g) URG - indica que o segmento contém uma mensagem de urgência;

h) ACK - indica que o segmento contém uma mensagem de confirmação de recebimento;

i) PSH - indica que os dados no buffer de recebimento têm que ser enviados para a aplicação;

j) RST - indica que o segmento contém uma mensagem de reinício de conexão;

k) SYN - indica que o número de seqüência deve ser sincronizado. É usado na seqüência de conexão;

l) *FIN* indica que o segmento contém uma mensagem de finalização de conexão. A máquina que envia o pacote FIN não deseja receber mais dados;

m) *window* – janela – indica quantos octetos é suportado pelo receptor. É complemento do campo de conhecimento para cálculo da janela de envio;

n) *checksum* – soma de verificação – é usado para soma de verificação do segmento, garantindo que as informações chegaram livres de problemas no receptor. Verificam tanto cabeçalho quanto dados;

o) *urgent pointer* – ponteiro de urgência – só é utilizado e verificado quando o bit de urgência está indicado. Serve para apontar onde começa a informação de urgência dentro da parte de dados do segmento TCP;

p) *options* – opções – é utilizado para alguma eventual opção implementada por algum fabricante ou mudança no protocolo;

q) *padding* – preenchimento – é utilizado para o preenchimento do cabeçalho TCP;

r) *data* – dado – é utilizado para tráfego de dados entre as aplicações.

3.3.8 Sockets

Conforme visto na Figura 3, o diagrama que representa o segmento TCP, implementa o conceito de portas para orientar uma conexão entre dois pontos. Uma porta é apenas uma abstração, dentro do endereço IP, ao serviço ou aplicação de destino dos dados. Um exemplo para isto é o serviço HTTP, que utiliza a porta 80, em uma determinada máquina, para navegação em páginas Web.

Segundo [PERKINS 2002], o *socket* é a concatenação entre o endereço IP e uma determinada porta de comunicação. Um par de *sockets* identifica unicamente cada conexão em uma rede. O *socket* de envio é o endereço IP fonte mais número de porta de origem, enquanto o *socket* de recebimento corresponde ao endereço IP de destino mais número de porta de destino.

3.4 UDP

O conjunto de protocolos da Internet também abrange um protocolo de transporte sem conexão. Este protocolo é chamado de UDP - *User Datagram Protocol* ou Protocolo de Datagramas de Usuários. Este protocolo serve para pequenas transferências, onde não é necessária uma conexão ou que o tempo e custo operacional de conexão e desconexão seria muito alto. Este protocolo, por não utilizar

controle e fluxo de conexão, é utilizado somente para envio de requisições e respostas a pequenos protocolos das camadas superiores [TANENBAUM 2001].O UDP, como o TCP, também utiliza o conceito de portas de comunicação, tendo sua estrutura mostrada na Figura 4:

Source Port	Destination Port
Length	Checksum
Data	

Figura 4: Formato do Datagrama UDP [TANENBAUM 2001].

Os campos citados são detalhados a seguir:

a) *source port* - porta de origem - identifica a porta de onde saíram os dados com destino ao receptor. Este item é opcional e quando não preenchido é inserido o valor padrão zero;

b) *destination port* - porta de destino: identifica a porta destino para onde estão seguindo os dados;

c) *length* – tamanho - indica o tamanho do datagrama UDP incluindo o cabeçalho UDP e a parte de dados;

d) *checksum* - soma de verificação - é um valor opcional de 16 bits para verificação do cabeçalho e dados do UDP;

e) *data* - dados - indica os dados a serem trafegados.

3.5 ICMP

Conforme [PERKINS 2002], o pacote ICMP, *Internet Control Message Protocol*, é a maneira com que o protocolo IP envia informações de status da rede. Através do ICMP, os vários dispositivos de uma rede podem identificar problemas de roteamento, congestionamento, qualidade de serviço entre outros. Um exemplo de serviço de rede com ICMP é o comando ping, o qual retorna o status do endereço de destino do pacote. É importante ressaltar a total atenção sobre este protocolo por diversas brechas de segurança que são exploradas através dos serviços que ele oferece.

4 FIREWALL

Os *Firewalls* são usados para criar pontos de controle de segurança nas fronteiras das redes privadas, ao fornecer a função de roteamento entre a rede privada e a Internet. A principal função de um *Firewall* é inspecionar toda a comunicação que traféga entre as redes aceitando ou rejeitando de acordo com as suas regras programadas [PERKINS 2002].

Este sistema de segurança pode ser dividido em duas categorias:

a) filtros de pacotes;
b) serviços *proxy*.

4.1 Filtro de Pacotes

A filtragem dos pacotes é um dos principais mecanismos que, mediante regras definidas pelo administrador em um Firewall, permite ou não a passagem de datagramas IP em uma rede [MARCELO 2002].

Os filtros podem ser implementados em roteadores ou nas pilhas de TCP/IP dos servidores. As filtragens implementadas nos roteadores evitam que o tráfego suspeito alcance à rede de destino e a implementação dos módulos de filtros TCP/IP nos servidores simplesmente evitam que máquinas específicas respondam ao tráfego suspeito [PERKINS 2002].

Existem dois tipos básicos de filtragem de pacotes:

a) filtro de pacotes sem estados;
b) filtro de pacotes com inspeção de estados.

4.1.1 Filtro de pacotes sem estados

São configurados em roteadores de fronteiras que aumentam a segurança determinando se um pacote deve ou não ser encaminhado com base nas informações nele contidas. Este filtro não retém informações sobre o estado das conexões que estão sendo usadas. Desta forma, os filtros de pacotes sem estados não conseguem determinar se aceitam ou não os fragmentos dos pacotes. Os *Firewalls* modernos usam informações de estado para acompanhar o status da conexão e, dessa maneira, conseguirem controlar de forma mais prática o roteamento dos pacotes na rede [PERKINS 2002].

A funcionalidade do filtro de pacotes sem estado, ocorre com base em dados dos campos do cabeçalho do protocolo, filtrando os campos mais úteis:

a) filtro de protocolo: este filtro efetua sua função com base no conteúdo do campo tipo de protocolo IP. Este campo pode ser usado para discriminar todo um conjunto de serviços, como UDP, TCP, ICMP;

b) filtro de endereço IP: esta filtragem por endereço IP permite limitar as conexões para *hosts* e rede específicas com base em seus endereços IP, conforme esta representado na Figura 05.

c) filtro de porta TCP/UDP: a filtragem por esse campo do pacote comumente é a mais usada, pois permite especificar para que serve o pacote. Conhecida também como filtragem de protocolo, pois o número da porta do protocolo corresponde ao protocolo de nível mais alto.

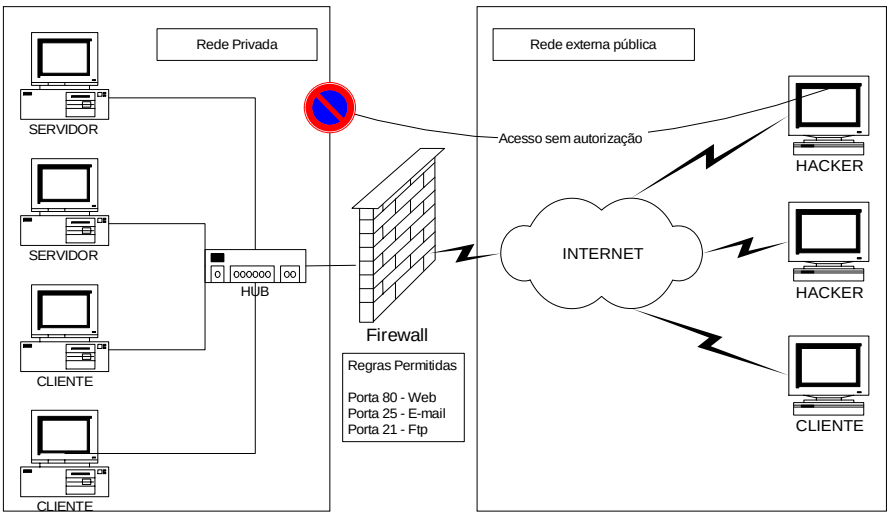

Figura 5: Funcionamento de um filtro de pacotes sem estados [PERKINS 2002].

4.1.2 Filtro de pacotes com inspeção de estados

Os filtros de pacotes possuem algumas falhas, sendo elas oriundas de que um único pacote em uma comunicação não contém informações suficientes para saber se ele deve ou não ser aceito. Os filtros de pacotes com inspeção resolvem esse problema retendo em sua memória os estados de todas as comunicações que passam pelo *Firewall* e, a partir desses estados que foram guardados em tabelas, eles conseguem determinar se um pacote deve ou não ser bloqueado [PERKINS 2002].

Semelhante a filtragem de pacote normal, mas com o monitoramento dos estados das conexões correntes que trafegam pelo *Firewall* o filtro de pacotes com estado não permite que serviços passem através do *Firewall*, a não ser os que estão programados para isso em suas regras e as conexões abertas em suas tabelas de estados. Veja Figura 6.

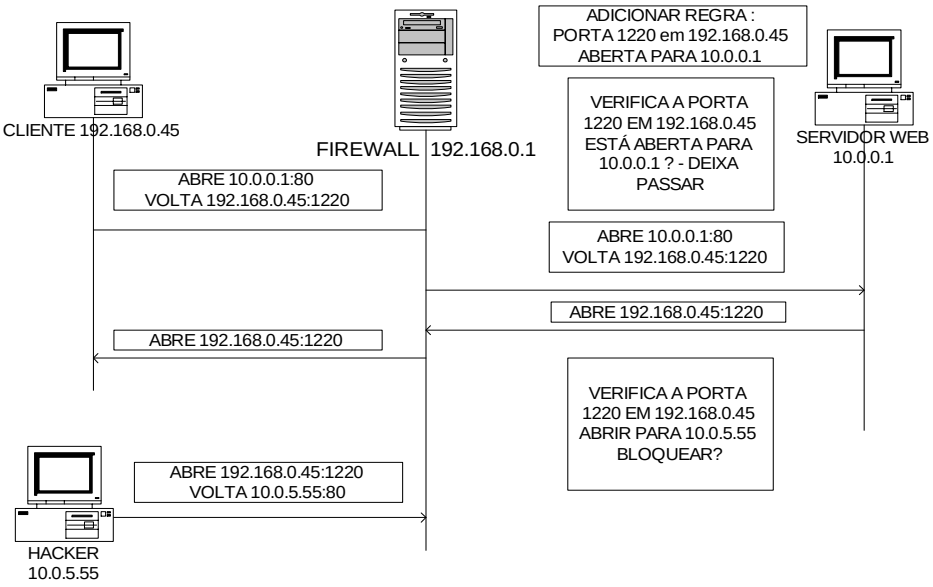

Figura 6: Funcionamento de um filtro de pacotes com estados [PERKINS 2002].

4.1.3 Filtragem de pacotes com estado no Linux

Os dois principais componentes para filtragem de pacotes e construção de um *Firewall* em ambiente Linux são o *Netfilter* e o *IPTables*. O *Netfilter* é a ponte entre o *kernel*, núcleo do Linux, e a estrutura de regras, sendo o atual sucessor do *IPChains* e do *Ipfwadm*. O *IPTables* tem a finalidade de construir as regras a serem usadas pelo *Firewall*. Estas regras podem ser aplicadas diretamente através do console do sistema operacional ou serem gravadas em um determinado arquivo a fim de que não sejam perdidas caso seja feita uma nova inicialização do sistema [RUSSELL 2001].

As regras são comandos definidos pelo IPTables, que permitem que ele realize um determinado evento. Essas são armazenadas nas *chains* e processadas na ordem de sua inserção. Segundo [MARCELO 2002], uma *chain* é uma área onde as regras definidas pelo administrador são armazenadas para sua operação. As principais *chains* que fazem parte do IPTables são :

a) i*nput* : define os pacotes de entrada na rede;

b) *output*: define os pacotes de saída da rede;

c) *forward*: define os pacotes a serem encaminhados, usados normalmente para mascaramento, sendo explicado com mais detalhes no item 4.2.

O *IPTables* possui tabelas aonde as *chains* são armazenadas, existem 3 tipos de tabelas, são elas :

a) tabela *filter*: filtragem padrão contendo as três *chains* (*Input,Output e Forward*);

b) tabela *nat*: usada para tradução de endereços possuindo outras *chains* como *Prerouting*, *Output* e *Postrouting*;

c) tabela *mangle*: usada para alterações especiais como, por exemplo, modificar algum tipo de serviço.

4.1.4 Principais comandos do NetFilter/IPTables

Para que as regras sejam formadas pelo *IPTables e* possam ser interpretadas pelo *Netfilter*, existem alguns comandos que são empregados para a sua construção. Basicamente, o *IPTables* oferece três políticas [RUSSELL 2001]:

a) *accept*: aceita o pacote recebido;
b) *drop*: nega pacote e não o retorna de volta;
c) *reject*: nega o pacote e retorna um aviso de erro ao emissor.

Em seqüência a essas três políticas, podemos definir os comandos a serem usados, que são os seguintes:

a) -A : adiciona ou atualiza uma regra;

b) -I : apenas adiciona uma nova regra;

c) -D : exclui uma regra específica;

d) -P : define a regra padrão;

e) -L : lista todas as regras armazenadas;

f) -F : exclui todas as regras armazenadas;

g) -R : substitui uma regra armazenada;

h) -C : efetua uma checagem das regras básicas;

i) -N : cria uma regra com nome específico;

j) -X : exclui uma regra com nome específico.

Em seqüência temos alguns parâmetros padrões, que são:

a) -p : define qual o protocolo deve ser tratado (TCP, UDP e ICMP);

b) -s ou -d : define o endereço de origem ou de destino em que a regra irá atuar.

c) –i : define a interface de rede por onde os pacotes são recebidos e enviados;

d) -j : define a direção de uma ação baseada em regras similares;

4.1.5 Principais características do NetFilter/IPTables

As principais características citadas pelo [RUSSELL 2001] são:

a) especificação de portas e endereço de origem e destino;

b) suporte a protocolos TCP/UDP/ICMP;

c) suporte a interfaces de origem e destino de pacotes;

d) tratamento de tráfego dividido em *chain*;

e) permissão de um número ilimitado de regras por *chain*;

f) rapidez, estabilidade e segurança;

g) existência de mecanismos internos para rejeitar automaticamente pacotes duvidosos ou mal formados;

h) suporte completo a roteamento de pacotes, estes tratados em uma área diferente a de tráfegos padrões;

i) suporte a especificação de tipo de serviço para priorizar o tráfego de determinados tipos de pacotes;

j) permissão para especificar exceções para as regras ou parte das regras;

k) suporte a detecção de fragmentos;

l) redirecionamento de portas;

m) suporte Nat – Mascaramento.

4.1.6 Exemplos de regras para Netfilter/IPTables

A seguir seguem exemplos de regras que são aplicadas ao Netfilter/IPTables [MARCELO 2002]:

a) nega todos os pacotes vindos de qualquer rede;

```
#iptables -A INPUT -j DROP
```

b) nega todos os pacotes ICMP;

```
#iptables -A INPUT -p icmp  -j DROP
```

c) rejeita todos os pacotes da interface eth0;

```
#iptables -A INPUT i eth0 -j REJECT
```

d) nega todos os pacotes vindos de qualquer endereço da rede 192.168.15.0;

```
#iptables -A INPUT -s 192.168.15.0/24 -j
DROP
```

e) aceita todos os pacotes da interface de *loopback*;

```
#iptables -A INPUT -s 127.0.0.1/32 -i lo -j
ACCEPT
```

f) aceita todos os pacotes da máquina 192.168.10.1;

```
#iptables -A INPUT -s 192.168.10.1/32 -j
ACCEPT
```

g) permite a saída de pacotes icmp da máquina 192.168.10.2;

```
#iptables -A OUTPUT -p icmp -s
192.168.10.2/32 -j ACCEPT
```

h) aceita conexões na porta TCP 3128 para rede 192.168.10.0;

```
#iptables -A INPUT -p tcp -dport 3128 -s
192.168.10.0/24 -j ACCEPT
```

i) permite a passagem de pacotes da rede 192.168.10.0 com destino a máquina 200.250.18.18 nas portas TCP 25,110;

```
#iptables -A FORWARD -p tcp –m multiport --
dport 25,110 -s 192.168.10.0/24 -d
200.250.18.18 -j ACCEPT
```

j) protege contra o ping da morte;

```
#iptables -A FORWARD -p icmp --icmp-type
echo-request -m limit --limit 1/s -j ACCEPT
```

k) protege contra ataques *Syn-flood*;

```
#iptables -A FORWARD -p tcp -m limit --limit
1/s -j ACCEPT
```

A opção limit é usada para limitar a taxa de coincidência em regras, ou seja, a taxa de acertos. O parâmetro --limit 1/s indica 1 (um) por segundo. É utilizado para evitar alguns tipos de negação de serviços e escaneamento de portas.

l) protege contra escaneamento de portas avançados;

```
#iptables -A FORWARD -p tcp --tcp-flags
SYN,ACK,FIN,RST -m limit --limit 1/s -j
ACCEPT
```

m) protege contra pacotes danificados ou suspeitos;

```
 #iptables -A FORWARD -m unclean -j DROP
```

n) o SNAT muda o endereço de origem do pacote. Isto é feito pela chain POSROUTING, antes do pacote ser enviado para uma rede externa. Utilizado para o compartilhamento de Internet para toda um rede;

```
#iptables -t nat -A POSTROUTING -i eth0 -s
192.168.10.0/24 -j SNAT --to 200.250.18.18
```

o) faz com que uma requisição para porta 80 de 200.250.18.18 seja enviada para 192.168.10.1;

```
#iptables -t nat -A PREROUTING -i eth0 –d
200.250.18.18 -p  TCP -dport 80 -j  DNAT --
to 192.168.10.1
```

O DNAT muda o endereço de destino do pacote. Isto é feito pela chain PREROUTING, depois do pacote ter chegado.

p) permite criar um *proxy* transparente, como veremos no capitulo 4.3.

```
#iptables -t nat -A PREROUTING -o eth0  -p
TCP -dport 80 -j  REDIRECT –to-port 3128
```

O REDIRECT é um caso especial de DNAT, onde é feito um redirecionamento. Utilizado para fazer o *proxy* transparente.

4.2 NAT

A NAT, *Network Address Translation*, tem por finalidade principal a conversão de endereços de uma rede privada em endereços válidos para uma rede como a Internet, conhecido também como mascaramento. Além de tornar disponível mais endereços privados às redes, ela propícia também a ocultação de informações em nível de TCP/IP sobre os *hosts* internos parecendo que todas as requisições sejam oriundas de uma máquina apenas [LEÃO 2001].

As principais funções de um *Firewall* NAT são:

a) conversão estática: geralmente usada para servidores que se encontram atrás do *Firewall*;

b) conversão dinâmica: utilizada para grandes números de máquina interna de uma rede quando elas precisam ter acesso a redes como a Internet;

c) conversão com balanceamento de carga: normalmente usada para que um único endereço IP responda por vários endereços de uma rede interna. Ex: Site da Web, www.google.com.br, que tem milhares de visitas instantâneas.

A fim de explicar de uma forma mais clara todo o serviço NAT efetuado pelo *Firewall*, segue o exemplo.

Tem-se uma máquina na rede interna endereçada em 10.1.1.7 e quer-se estabelecer uma conexão com um serviço WEB no endereço 200.190.180.30. Envia-se um pacote, 10.1.1.7:1234, para 200.190.180.30:80. A interface interna do *Firewall*, 10.1.1.1 recebe esta solicitação e efetua a conversão, conforme ilustrado na Tabela 5:

Tabela de Conversão	
➔	Origem: 10.1.1.7:1234
➔	Destino: 200.190.180.30:80
✱	Conversão: 200.50.40.22:23456

Tabela 5: Tabela de Conversão da NAT [PERKINS 2002].

O *Firewall,* este endereçado com sua interface externa em 200.50.40.22, a qual irá receber de volta o retorno da solicitação e encaminhá-la para a tabela de conversão. Nas tabelas serão verificadas todas as entradas na qual corresponde ao pacote recebido e converter de forma reversa o mesmo para que a máquina de origem da rede interna possa receber a informação solicitada.

4.3 Serviços Proxy

Um serviço *proxy*, também conhecido como *gateway* de aplicação, é utilizado para substituir as tentativas de conexão a servidores dirigidos para fora da rede interna e em seguida efetuar a solicitação ao servidor de destino real em nome do cliente. Quando este servidor retorna os dados, o *proxy* os transmite de volta para o cliente. Analisando por uma outra óptica, os *proxys* realizam um ataque benigno como se houvesse uma pessoa no meio do caminho, exemplificando como qualquer roteador entre duas redes poderia realizar qualquer tipo de processamento sem a devida permissão [PERKINS 2002]. Outra função interessante do serviço de *proxy* é a possibilidade de *cache*, que

é um armazenamento do conteúdo acessado com maior freqüência na rede externa, reduzindo assim os acessos feitos a ela.

De acordo com [PERKINS 2002], a melhor maneira de ter um serviço de *proxy* implementado é tê-lo junto ao *Firewall*. Desta forma, as políticas de segurança implementadas pelo próprio *Firewall* servem como forma de segurança ao serviço de *proxy*, visto que ele não possui autodefesa.

O serviço de *proxy* transparente é mais sofisticado que os outros pois possui as funções de filtro e mascaramento de IP's [PERKINS 2002]. No caso de implementação, o *proxy* conecta-se ao servidor remoto e solicitar as informações pertinentes em nome do cliente que está sendo bloqueado na rede interna. As informações obtidas dessa maneira são retornadas ao cliente que solicitou usando a função de NAT do serviço de *Firewall*, parecendo ter sido executada diretamente pelo servidor remoto.

O *proxy* ouve as solicitações oriundas de clientes da rede interna e então envia tais solicitações para a rede externa como se o próprio servidor *proxy* fosse o cliente de origem. Ao receber o retorno da solicitação, o serviço de *proxy* transmite ao cliente solicitador este retorno. Sendo ilustrado na Figura 7:

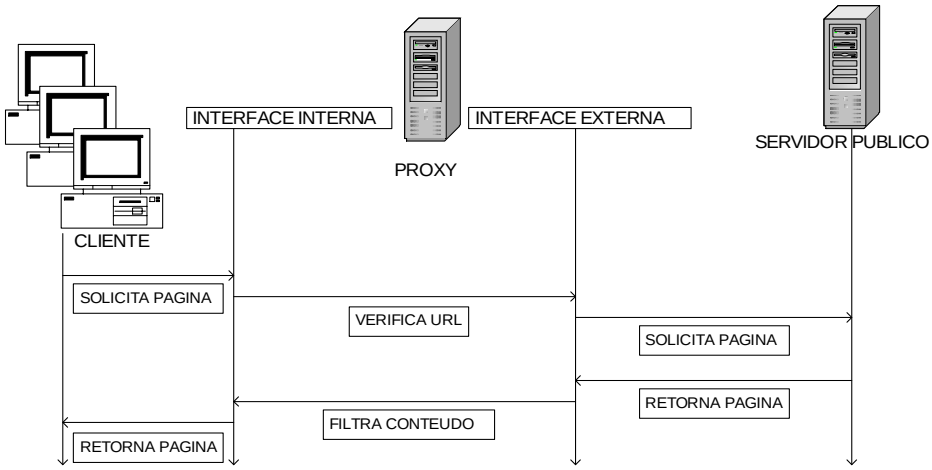

Figura 7: Funcionamento do serviço proxy [PERKINS 2002].

4.4 Arquiteturas de Firewall

Um *Firewall* pode ser implementado através de várias arquiteturas, isso irá depender da necessidade de segurança e a topologia da rede. Para cada tipo de rede, é necessária a configuração de um tipo específico de *Firewall.* Existem conceitos padrão para a construção da arquitetura de proteção de rede, porém as regras sempre são mudadas e instituídas conforme a política de segurança do ambiente. Atualmente, existem algumas variações nas arquiteturas principais de *Firewalls*, onde estão se constituindo novos paradigmas e padrões [NORTHCUTT 2002].

As principais arquiteturas dos *Firewalls,* segundo [PERKINS 2002], são:

a) *dual homed host;*

b) *screened host;*

c) *screened subnet*.

4.4.1 Dual homed Host

Este arquitetura é composta por duas interfaces de rede e normalmente é configurada de maneira que os pacotes não são diretamente passados de uma rede para outra. As máquinas da Internet podem se comunicar com o *Firewall* da mesma maneira que as máquinas da rede privada, porém o tráfego entre as duas redes é controlado pelo *Firewall*, ilustrado na Figura 8.

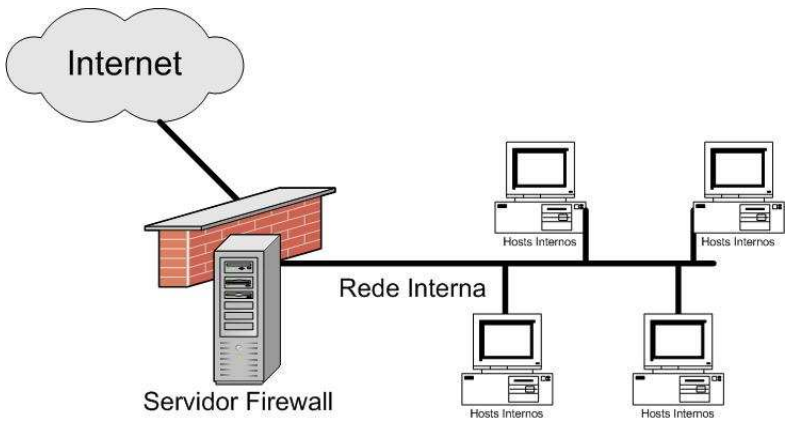

Figura 8: Arquitetura de Firewall Dual Homed Host [PERKINS 2002].

Neste tipo de arquitetura, o *Firewall* pode agir como o roteador entre estas redes, porém a funcionalidade de roteador é desabilitada,

assim como a comunicação entre redes. Para a comunicação das máquinas internas para máquinas externas, são utilizados os serviços de *proxy.*

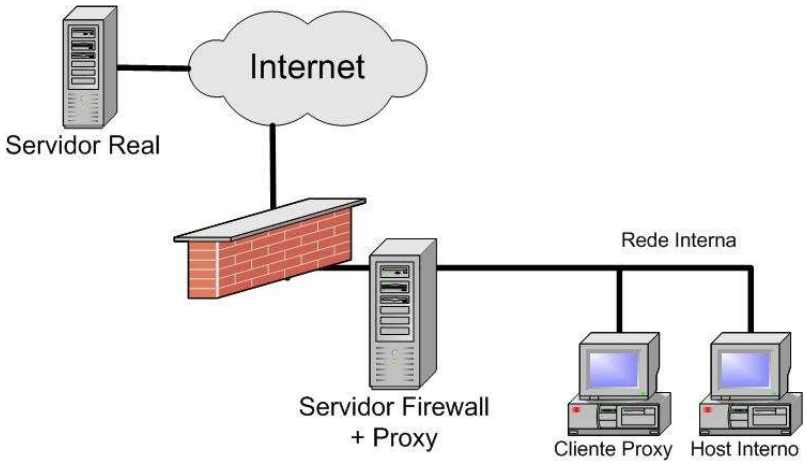

Figura 9: Arq. Firewall Dual Homed Host com servidor Proxy [PERKINS 2002]

4.4.2 Screened Host

A arquitetura *Screened Host*, assim como na arquitetura *Dual Homed Host,* disponibiliza os serviços de proteção com uma máquina ligada a duas redes. Porém, os serviços de rede são disponibilizados por uma máquina ligada a rede interna. Neste tipo de arquitetura, a segurança é feita através de filtro de pacotes no roteador que liga as duas redes. A máquina interna responsável pelos serviços de rede é chamada de *bastion host,* ilustrado na Figura 10.

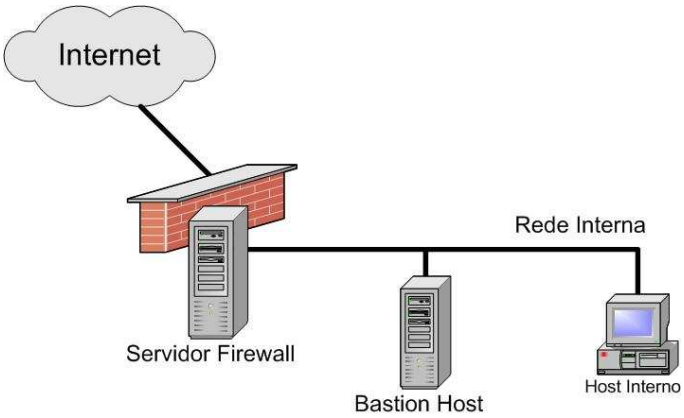

Figura 10: Arquitetura de Firewall Screened Host [PERKINS 2002]

O roteador é configurado para que a única máquina que receba conexões e requisições externas seja sempre o *bastion host*, porém, mesmo sendo a única máquina que recebe conexões o roteador é configurado de modo a permitir somente os serviços definidos, filtrando os demais.

Devido ao perigo eminente que o *bastion host* é submetido, a definição de segurança desta máquina é total. O roteador é também responsável pela permissão da conexão da rede interna à rede externa direta ou indiretamente.

É possível haver três tipos de permissões:

a) não é permitido nenhum tipo de conexão direta a rede externa, apenas por meio de servidores proxy;

b) é permitido o acesso de máquinas internas à rede externa, com filtro de pacotes e serviços;

c) é apenas permitido o acesso à rede externa para somente algumas máquinas definidas pela política de segurança interna, deixando a conexão das demais máquinas por meio de servidores proxy ou não permitindo o acesso à rede externa.

4.4.3 Screened subnet

A arquitetura de *Firewall Screened Subnet* adiciona mais segurança à arquitetura *screened host* retirando o *bastion host* da rede interna, colocando-o numa rede periférica que também é conhecida como DMZ, *Delimitarized Zone* ou Zona Desmilitarizada, e isolando esta rede periférica da rede interna.

A *Screened Subnet* é considerada como o último nível de arquitetura em *Firewalls*, isolando totalmente os *bastion hosts (Veja Figura 11)*, estes podendo ser atacados diretamente. O impacto na segurança é significativo, apesar do aumento na complexidade de rotas e configuração de filtros e regras, pois são necessários dois roteadores com filtragem de pacotes (um roteador externo e outro interno) e uma nova rede separada da rede corporativa interna.

A segurança nesta arquitetura é maior, pois, ao contrário das arquiteturas anteriores e suas variações, quando um *bastion host* sofre

uma invasão, a rede interna não está vulnerável, necessitando que ainda seja quebrada a segurança no segundo roteador. O nível de complexidade também aumenta para o invasor que não tem conhecimento dos endereços falsos da rede interna, nem sequer a rota para tais máquinas.

A configuração da filtragem de pacotes nos dois roteadores está separada, podendo ter, no roteador interno, regras mais rigorosas de filtragem e negação de serviços, enquanto são inviáveis certos tipos de filtragem de pacotes no roteador externo.

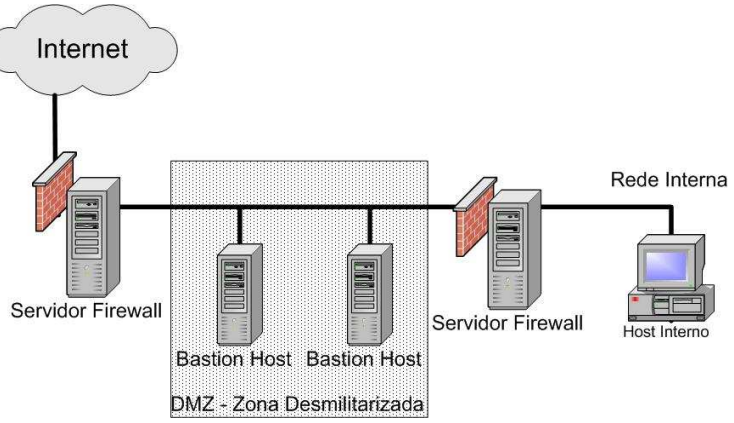

Figura 11: Arquitetura de Firewall Screened Subnet [PERKINS 2002].

Alguns exemplos de filtragem que podem ser implantados nos roteadores internos são:

a) criação de regras para permissão de conexões para somente serviços em servidores *proxy*, fazendo com que as

máquinas internas só possam utilizar serviços Internet através de *proxy*;

b) criação de regras para permissão de abertura de conexões da rede interna para externa, somente neste sentido, ou seja, somente a rede interna poderá abrir conexões, evitando conexões a *back-doors,* máquinas infectadas por vírus ou com programas de precedência duvidosa.

5 INTERFACE

O termo interface é aplicado normalmente àquilo que interliga dois sistemas. No processo de interação usuário-sistema, a interface é o combinado de software e hardware necessário para viabilizar e facilitar os processos de comunicação entre o usuário e uma determinada aplicação [CYBYS 1995].

5.1 Fatores Humanos

A seguir são abordados os principais fatores necessários no processo de interação entre pessoas.

5.1.1 Percepção humana

Ao se tratar um sistema que interage, baseado em software, os fatores humanos são amplamente relacionados através de diferentes significados. O ser humano percebe o mundo por meio de um sistema sensorial, ainda mais quando uma IHC (Interface Homem Computador) é considerada, predominam os sentidos visuais, táteis e auditivos. Dessa forma, o usuário recebe as informações, armazena-as e as processa usando raciocínio indutivo e dedutivo [PRESSMAN 2003].

5.1.2 Interação homem – computador

É uma extensão da disciplina de ergonomia, pelo fato de que a ergonomia estuda os sentidos e as capacidades motoras das pessoas ao utilizarem máquinas em geral, e interação homem – computador abrange não só estes estudos como também a análise da capacidade mental que possibilita às pessoas produzirem informações processáveis por computadores.

Essas informações visam fornecer aos pesquisadores e desenvolvedores de sistemas, explicações e previsões para fenômenos de interação usuário-sistema e resultados práticos para o design da interface de usuário [PRESSMAN 1995].

5.1.3 Habilidade e comportamento humano

Um fator importante a observar é existência de diferenças do nível de habilidades entre vários usuários, personalidade e comportamentos. Se o uso de uma interface é fácil para um analista de sistemas, para um usuário leigo de computadores ela pode ser difícil. O projetista de interface precisa realizar um levantamento estatístico do perfil dos futuros usuários de uma interface a fim de torná-la o mais próximo do seu nível de habilidade [PRESSMAN 1995].

5.2 Ergonomia

A ergonomia é o estudo da adaptação do trabalho ao homem. Parte-se do conhecimento do homem para fazer o projeto do trabalho, ajustando-o às capacidades e limitações humanas. Segundo [CYBYS 1995], Dominique Scapim realizou estudos visando facilitar a recuperação de conhecimento ergonômico. Este estudo resultou em uma lista de 8 critérios, sendo divididos em sub-critérios. Os principais critérios são: Condução, Carga de Trabalho, Controle Explicito, Adaptabilidade, Gestão de Erros, Consistência, Significado dos Códigos e Compatibilidade [CYBYS 1995].

5.3 Engenharia de Usabilidade

A usabilidade é a forma como um sistema foi desenvolvido considerando o aspecto de facilidade de uso pelo usuário. Um sistema apresenta um alto grau de usabilidade quando, as tarefas de diferentes níveis de complexidade executadas, forem visualizadas e concluídas de forma fácil e eficaz, fazendo com que o usuário enfoque mais o trabalho e não a forma como fazê-lo [PERAZOLO 2003].

A condição principal para obter um alto padrão de usabilidade é criar uma interface com um alto nível de interação com o usuário. Antigamente, as interfaces exigiam de seus usuários o conhecimento de inúmeros comandos para utilizá-la. Este problema foi minimizado com a criação das interfaces gráficas, que fazem seus comandos serem

executados de forma intuitiva, acabando com o pré-requisito de usuário experiente para seus sistemas [PERAZOLO 2003].

Segundo [WINCKER 2001], a qualidade da interação dos usuário com uma determinada interface está associada aos seguintes princípios:

a) facilidade de aprendizado;
b) facilidade de lembrar como realizar uma tarefa após algum tempo;
c) rapidez no desenvolvimento de tarefas;
d) baixa taxa de erros;
e) satisfação subjetiva do usuário.

5.4 Projeto de Interface

O projeto da interface é sem duvida uma das etapas mais importantes no desenvolvimento de um sistema computacional. Da mesma forma que existem inúmeras metodologias a serem escolhidas para um projeto de sistema. Fundamentalmente, o projeto de uma interface tem início com a prototipação de algumas telas e a sua apresentação para o usuário final do sistema afim de uma avaliação [PRESSMAN 1995].

Conseqüentemente, o usuário informa as alterações necessárias retornando ao projetista de interface ou ao programador e este as faz criando um novo protótipo. Com isso, tem-se um ciclo de criação,

avaliação, manutenção e novamente uma avaliação conhecida também como ciclo de vida de prototipação ou espiral [PRESSMAN 1995].

5.4.1 Critérios para interfaces

Para que alcançar o maior proveito na aplicação de interfaces, sejam elas gráficas ou não, precisa-se seguir alguns critérios relacionados à qualidade ergonômica de uma Interface Homem Computador [CYBYS 1995].

a) condução: o sistema como um todo deve conduzir visualmente a pessoa que o usa, de forma a aconselhar, orientar e informar proporcionando um fácil aprendizado levando em conta a presteza e o retorno imediato de respostas que este oferece;

b) carga de trabalho: quanto menor a carga de trabalho menor a possibilidade do usuário cometer erros;

c) controle explícito: o usuário tem controle total do processamento da aplicação;

d) adaptabilidade: diz respeito quanto a capacidade de adaptação da interface em relação a necessidade e preferências do usuário;

e) gestão de erros: são os mecanismos que evitam a ocorrência de erros, favorecendo a correção dos erros caso esses ocorram;

f) consistência: está diretamente ligado a homogeneidade para diferentes contextos criando assim um padrão;

g) significado dos códigos: refere-se a clareza em exibir códigos ou denominações para simbolizar uma informação;

h) compatibilidade: relaciona-se a um prévio entendimento que possa haver entre o conhecimento do usuário e as tarefas a serem executadas no software através da interface.

Basicamente, o projeto das interfaces divide-se em dois tipos: interfaces por linha de comando e interfaces gráficas.

5.4.2 Interface por linha de comando

Nos tempos mais remotos, quando os computadores só ofereciam o sistema operacional como interface única, qualquer tarefa era executada a partir da linha de comando seguida de inúmeros parâmetros para a sua realização. Para concluir com sucesso tal tarefa, era necessário que o usuário dominasse a aplicação executada. O uso da interface por linha de comando provia a sensação de total controle do sistema [FRAINER 1991].

Os usuários mais experientes com estas interfaces podiam explorar ao máximo a sua performance criando comandos complexos, com ou sem mensagens de retorno e em certas vezes até mesmo concatená-los a fim de automatizar determinada tarefa do sistema.

Em contrapartida, estas interfaces apresentavam erros de sintaxe ou execução, ainda mais quando era necessário o uso de vários comandos para a realização de uma determinada tarefa exigindo, dessa forma, treinamento rigoroso dos usuários para interagir com elas [FRAINER 1991].

5.4.3 Interface gráfica

A dificuldade encontrada no manuseio das interfaces por linha de comando deve-se à escassez de recursos que tais computadores proviam. Com o desenvolvimento da tecnologia junto ao aumento de processamento e armazenamento, tem-se hoje interface que exploram os sentidos humanos tornando a interação humano computador mais apropriada. Estas interfaces exploram o uso de mouse, ícones e janelas [PRESSMAN 1995].

5.5 Projeto de Web Sites

Segundo [PERAZOLO 2003], o projeto de *Web sites* envolve tanto uma parte artística como de engenharia:

a) parte artística: Nesta parte o desenvolvedor deve ser criativo, inovador e mostrar seu diferencial, porém deve-se tomar cuidado para não poluir visualmente a interface;

b) parte de engenharia: É especificado todo processo de planejamento, arquitetura, projetos de componentes e manutenção do *site*.

O projeto de *web sites* pode ser dividido em projeto de página, projeto de conteúdo e projeto do *site.*

5.5.1 Projeto da página

Neste projeto, deve-se preocupar com a organização, onde visa direcionar a atenção, priorizar informação, simplificar a navegação e reduzir erros, precisa possuir projetos gráficos para conduzir o olhar do usuário e possuir um *layout* padrão [PERAZOLO 2003]. Assim deve se tratar 3 itens:

a) tela: O conteúdo da página deve corresponder a pelo menos metade do *design* da página, em torno de 50%;

b) tempo de resposta: os conselhos básicos com relação ao tempo de resposta são: Um décimo de segundo é o limite para fazer com que o usuário sinta que o sistema está reagindo instantaneamente; Um segundo é o limite para que o fluxo de pensamento do usuário permaneça interrupto e dez segundos são o limite para incomodar o usuário;

c) *links:* As principais regras para construção de *links* são: Evitar o uso de “Clique Aqui” pois dificilmente atraem a atenção do usuário; sublinha-se a palavra chave do *link*; Os títulos dos *links* devem ser claros e explicativos; Manter um padrão para as cores dos *links*.

5.5.2 O Projeto do conteúdo

Neste projeto, deve-se ter preocupação com o planejamento do conteúdo onde é decidido que tipo de informação será colocada e a forma de representação [PERAZOLO 2003]. Assim, deve se tratar 4 itens:

a) Escrita para *web sites*: Não somente a ortografia é importante é preciso também planejamento. Há 3 regras principais para a escrita de textos: Ser breve na escrita, não escrever mais do que 50% do texto que normalmente é impresso em publicações; Escrever tendo em vista a

facilidade de leitura, não exigir que o usuário leia blocos de textos longos e contínuos; Usar hipertexto, para segmentar informações longas em várias páginas;

b) legibilidade: existem algumas regras que garantem a legibilidade no *site*: usar cores de alto contraste entre o texto e o fundo, a legibilidade ótima requer texto preto e fundo branco; usar fundos de cores lisas ou padrões de fundo extremamente sutis; usar fontes de tamanhos suficientes, para que as pessoas possam ler o texto; fazer com que o texto fique imóvel; evitar o uso de letras maiúsculas para o texto;

c) imagens e fotografias: São importantes para representação do conteúdo da página, além de chamar a atenção do usuário. O lado negativo é que imagens necessitam de um tempo elevado de *download*;

d) animação: Se as imagens tornam um *web site* mais atraente, as animações proporcionam um impacto ainda maior. Porém, este recurso só é favorável em determinadas ocasiões: mostrar continuidade nas transições; indicar dimensionalidade nas transições, indicar movimento vai-e-vem em alguma dimensão navegacional; ilustrar a mudança do tempo; multiplexar o monitor, mostrar múltiplos objetos de informação ao mesmo tempo; enriquecer as representações gráficas; visualizar estruturas tridimensionais.

5.5.3 Projeto do site

Neste projeto deve-se ter a preocupação com o planejamento da navegação [PERAZOLO 2003], assim deve se tratar alguns aspectos:

a) *homepage*: A *homepage* é a página inicial do web *site*, portanto deve ter alguns elementos adicionais com relações as demais: um diretório com as principais áreas de conteúdo; um resumo das notícias ou promoções mais importantes e um recurso de busca, quando aplicáveis; um título que demonstre o objetivo do *web site*; um logotipo que marque visualmente o *web site*;

b) navegação: Deve ser bem projetada para ajudar ao usuário a se localizar dentro do *web site*. Para ajudar o usuário, existem recursos nos *browsers* atuais como: botão voltar, lista de histórico e a apresentação visual diferente dos *links* visitados anteriormente pelo usuário.

6 PROTÓTIPO

6.1 Objetivo

O objetivo do protótipo desenvolvido neste trabalho é fornecer uma interface gráfica que permita uma melhor interação entre um administrador de redes e a ferramenta de segurança *Netfilter/IPTables*, em modo texto, para *Linux*. Conseqüentemente, através do uso desta interface gráfica, o administrador terá uma melhor praticidade e eficácia na gerência do *Firewall*.

6.2 Metodologia

A metodologia utilizada para o desenvolvimento está relacionada ao ciclo de vida espiral, que demonstra várias interações entre o desenvolvedor e o usuário caracterizando assim o modelo de prototipação evolutiva [PRESSMAN 1995]. Este ciclo de vida está representado na Figura 12.

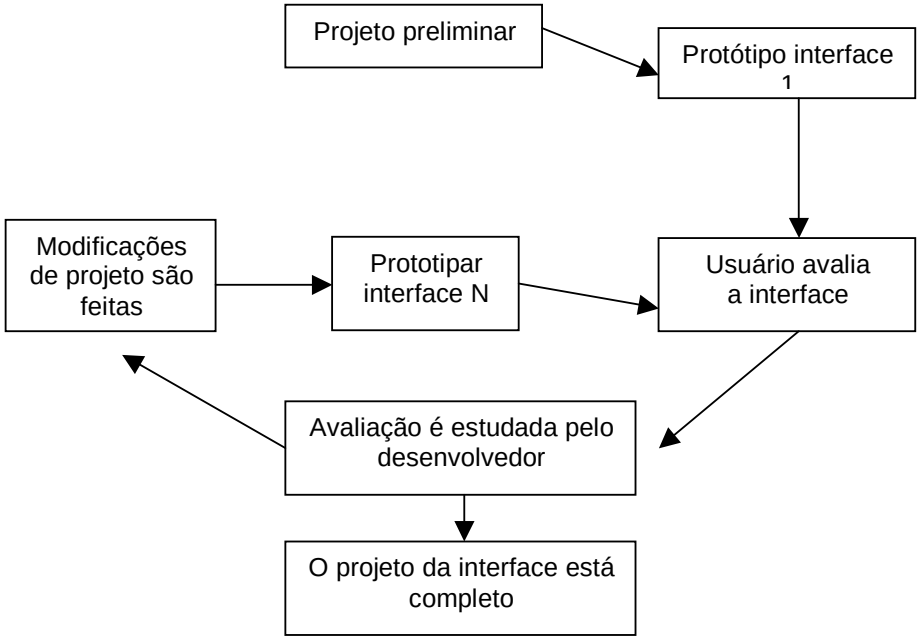

Figura 12: Ciclo de vida espiral para prototipação de interfaces [PRESSMAN 1995].

6.3 Tecnologias utilizadas

Esta seção apresenta as tecnologias que foram utilizadas durante o desenvolvimento deste projeto.

6.3.1 Hyper Text Markup Language

A linguagem *HTML* (*Hyper Text Markup Language*) é uma das tecnologias que formam a base de qualquer projeto para Web. Ela é um

padrão para apresentação de hipertexto, com recursos de estruturação de texto, inclusão de imagens e multimídia, além da criação de âncoras para interligação entre os documentos relacionados [RAPOSO 2002].

A *HTML* possui um conjunto de elementos para interface. Este conjunto limita-se basicamente a botões, caixas de seleção, caixa de entrada para texto e imagens que podem ser mapeadas.

Atualmente existem inúmeras ferramentas que auxiliam na utilização de *HTML*, fornecendo meios visuais para a "programação" nesta linguagem. Contudo, a *HTML* é o requisito básico para um projeto *Web*, podendo ser aplicada em conjunto com outras tecnologias de desenvolvimento para *Web*. A ferramenta usada para desenvolvimento da interface foi o *NVU Editor 0.90*[2].

6.3.2 Ambiente Servidor

Para que fosse colocada em prática a utilização da interface, foi instalado um servidor no ambiente operacional *Linux Red Hat 9.* Este tendo como serviço instalado, o Apache na sua versão 1.3.28 , *daemon* do *HTTPD* servidor de páginas para *Web*.

Junto a este serviço, funciona em paralelo o serviço *PHP* na versão 4.3.2, que possibilita a interação da interface *HTML*, e o servidor *Linux*. Cabe ressaltar que toda esta interação só é possível a partir de

[2] NVU: http://www.nvu.com/download.html

uma prévia autenticação através da interface, através do *PHP*. Esta por sua vez automaticamente cria uma sessão para cada usuário a fim de manter a integridade de autenticação.

A autenticação ocorre em modo seguro visto que está incorporado ao Apache, onde existe o módulo *OpenSSL 0.9.6b* permitindo que o tráfego de dados sejam criptografados.

A interação que ocorre entre o *PHP* e o servidor *Linux* é necessária para que as configurações que foram realizadas na interface e armazenadas no banco de dados, MySql na versão 3.23.57, possam ser repassadas ao servidor. Para melhorar a agilidade do gerenciamento das tabelas e as informações no banco de dados, foi instalado a ferramenta PhpMyAdmin na versão 2.6.1.

Em determinada ação, aplicar regras ao *Firewall*, realizada na interface gráfica, as informações que foram gravadas no banco de dados geram um arquivo texto previamente formatado que contém todas as regras ativas para entrar em vigor no *Firewall.* A ferramenta *NetFilter/IPTables* possui como característica básica, ler a partir de um arquivo texto as regras que devem ser implantadas para seu funcionamento.

Desta forma, a interface ao gerar este arquivo por meio do *PHP*, efetua uma interação direta com o servidor Linux, mais propriamente com o serviço *NetFilter/IPTables* executando uma ação de parada, atualização de seu arquivo de regras e reinicio do serviço concluindo assim todo o processo interação, a partir da interface gráfica.

6.3.3 Ambiente Cliente

Para que exista interação entre o administrador de redes e a interface gráfica, é necessária apenas a utilização de um navegador de páginas *Web* que permita a visualização de imagens e frames. Além disso, pede-se que a sua criptografia seja igual ou superior a 128 bits para que se possa acessar com sucesso a área de autenticação da interface.

6.3.4 Diagrama de entidade e relacionamento

O projeto de banco de dados é obtido em duas fases:

a) modelagem conceitual: sendo nesta fase construído um modelo conceitual, no formato de um diagrama de entidade e relacionamento (DER). Tal modelo captura as necessidades da aplicação em termos de armazenamento de dados de forma independente da implementação;

b) projeto lógico: esta etapa visa transformar o modelo conceitual obtido em um modelo lógico. Este modelo define como o banco de dados será implementando em um sistema gerenciador de banco de dados específico (SGBD) [HEUSER 2000].

Utilizando essa metodologia, foi gerado o banco de dados para o protótipo, formando em um primeiro momento seu modelo conceitual e em sequida, seu esquema físico para o sistema gerenciador de bancos de dados *MySql*.

Foi gerado o diagrama físico e lógico e a partir da ferramenta *PhpMyAdmin*, foi extraído os scripts de criação em modo texto para o modelo lógico, e estes podem ser encontrados no Anexo A.

As Figuras 13 e 14 apresentam os diagrama de entidades e relacionamento do protótipo, nos modelos lógico e físico respectivamente.

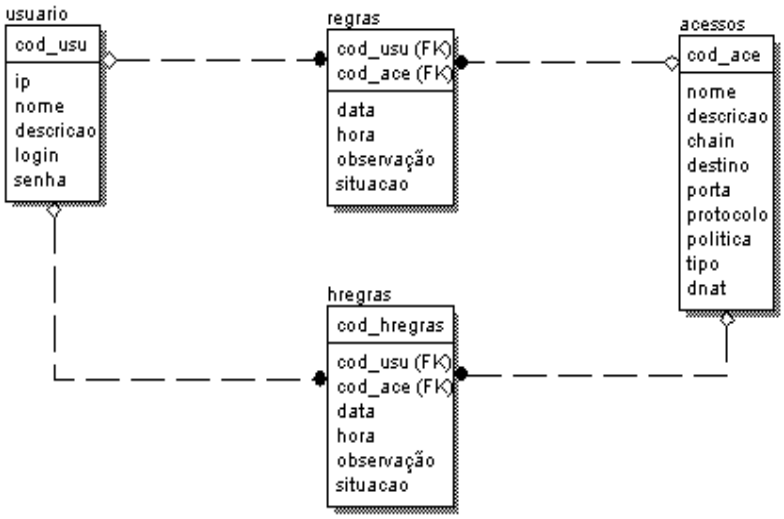

Figura 13: Diagrama de entidade e relacionamento protótipo (lógico).

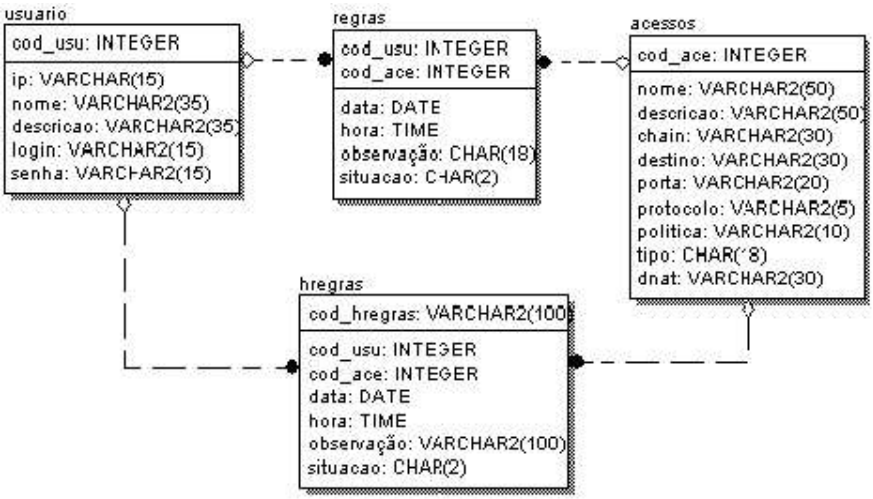

Figura 14: Diagrama de entidade e relacionamento protótipo (físico).

6.3.5 Projeto Navegacional

O projeto navegacional do protótipo é apresentado nas Figuras 15, 16, 17, 18, 19 e 20. Sendo estruturado de forma hierárquica para uma melhor visualização.

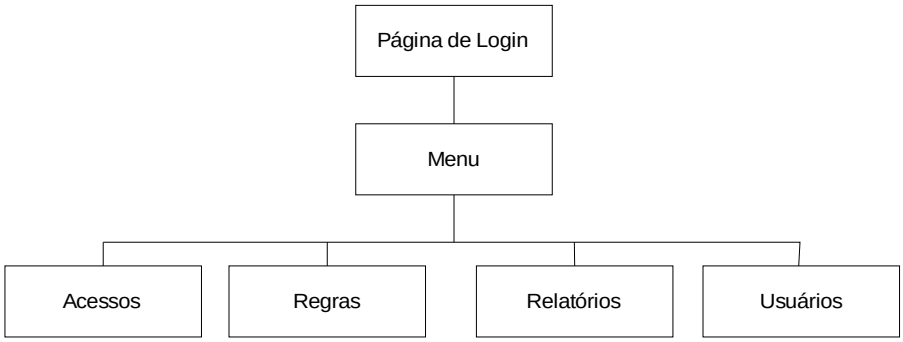

Figura 15: Diagrama Navegacional do Protótipo.

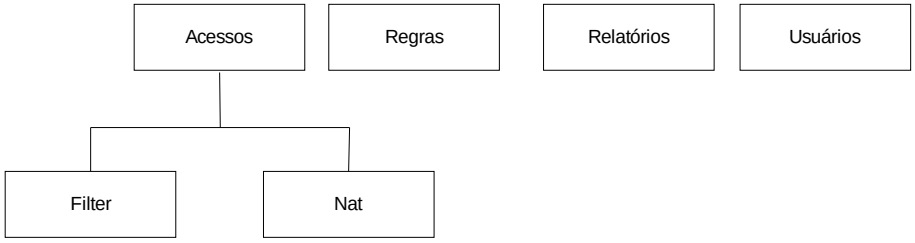

Figura 16: Diagrama Navegacional de acessos.

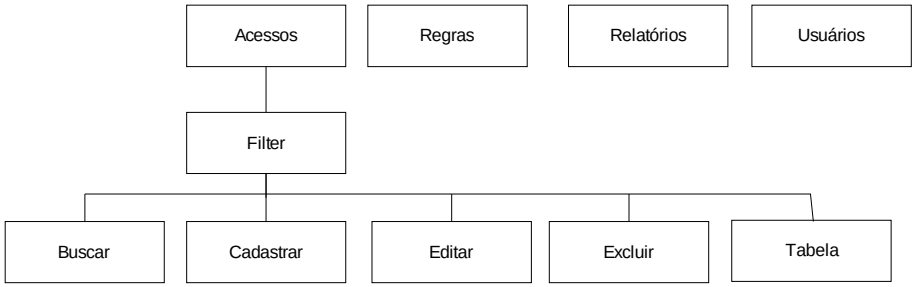

Figura 17: Diagrama Navegacional de Acessos - Filtros.

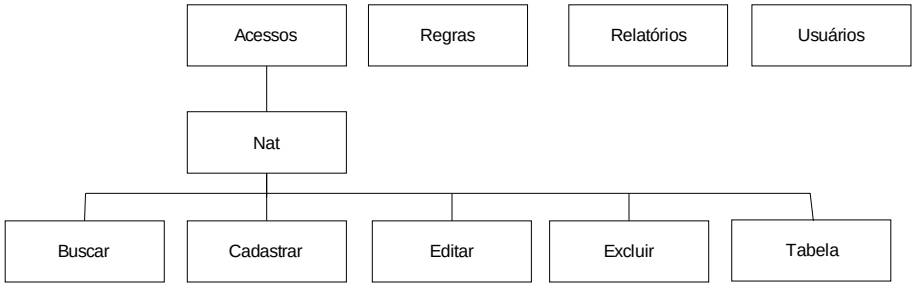

Figura 18: Diagrama Navegacional de Acessos - Nat.

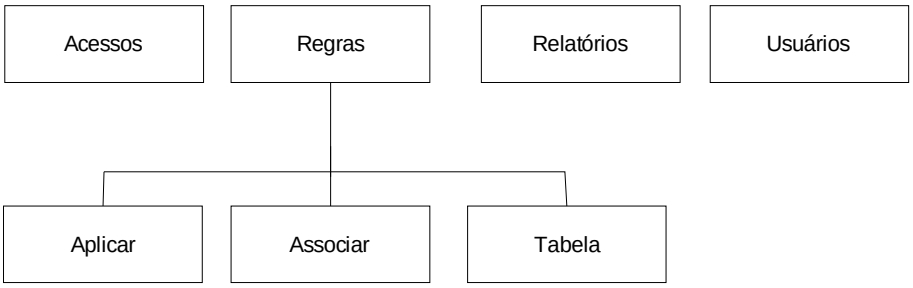

Figura 19: Diagrama Navegacional de regras.

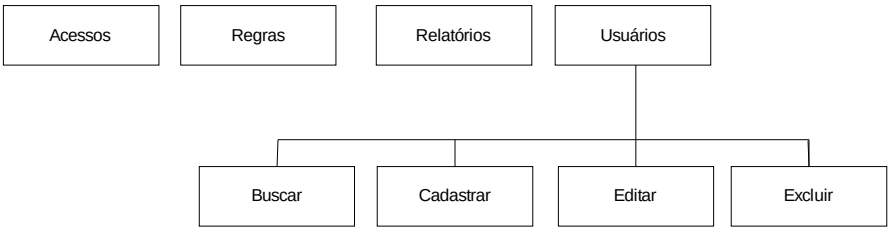

Figura 20: Diagrama Navegacional de Usuários.

6.3.6 Funcionamento do Protótipo

Ao iniciar o uso da interface, é preciso autenticar-se com um usuário para administrar a aplicação, sendo que o primeiro usuário será inserido diretamente do Banco de Dados. Como mostra a Figura 21, onde foi criado um usuário para ter acesso a aplicação. Caso exista erro ao logar, automaticamente o usuário da interface recebe a mensagem de discordância de login ou senha, conforme a Figura 22.

Figura 21: Tela de Login

Figura 22: Falha na Autenticação de Usuário

Após logar-se na interface, é necessário criar os usuários, estes por atribuição tem um número IP, nome e descrição, conforme Figura 23.

Figura 23: Cadastro de Usuário

Além do cadastro de usuários, têm-se a busca, a edição e a exclusão como mostram as Figuras 24, 25, 26 e 27, respectivamente.

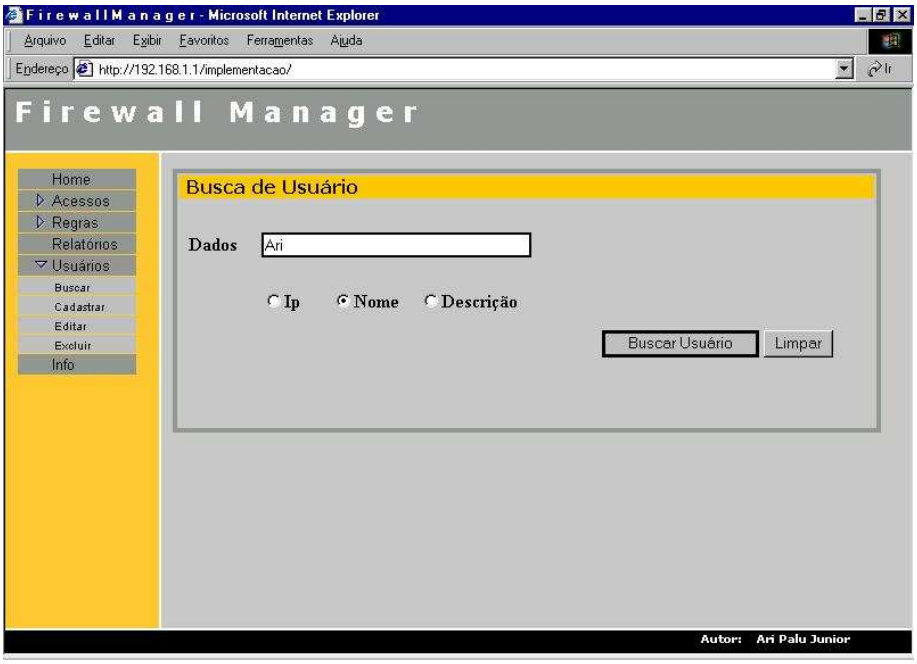

Figura 24: Busca de Usuário

Figura 25:Resultado da Busca de Usuário

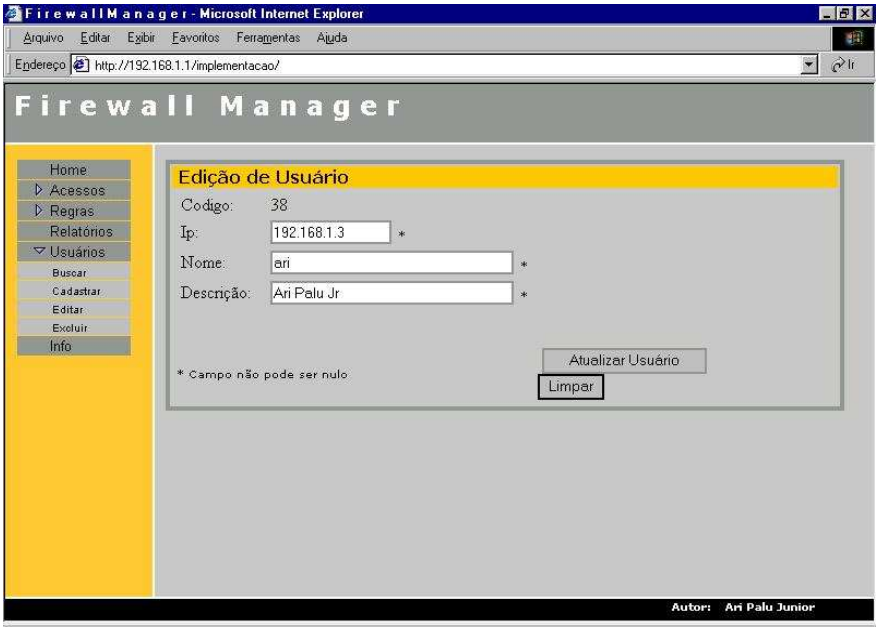

Figura 26: Edição de Usuário

Figura 27: Exclusão de Usuário

Tanto o cadastro de Filtro quanto o de Nat, exibidos nas Figuras 28 e 29, possuem busca, edição e exclusão de registros previamente armazenados no banco de dados. O cadastro de acesso do Filtro ou Nat são necessários para que se possa associar a eles um usuário, criando assim uma regra do *Firewall.*

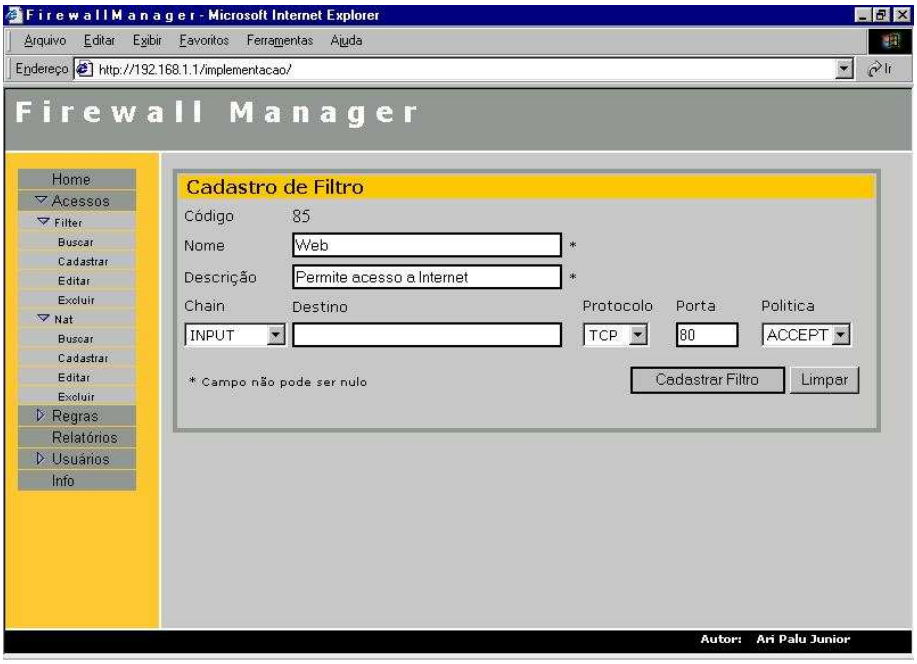

Figura 28: Cadastro de Filtro

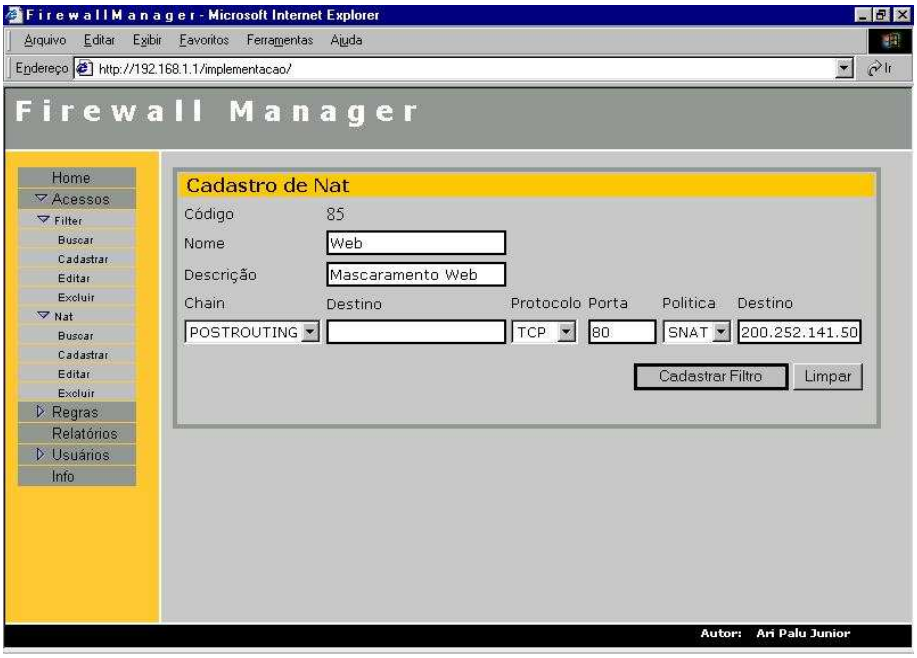

Figura 29: Cadastro de Nat

A associação de um usuário com um determinado acesso cadastrado gera uma regra no *Firewall.* Após essa associação, a regra gerada fica em estado de Espera, ou seja, armazenada na base de dados, mas não *ativa* no *Firewall.* Para que se possa aplicar as regras e, conseqüentemente, torná-las *Ativas*, é necessário clicar em aplicar, sub-item de Regras no menu da interface, conforme ilustrado na Figura 34. Este processo coleta todas as associações existentes entre usuários e acessos, transformando-as as mesmas para o estado de Ativas, ou seja, tornando as regras realmente ativas no *Firewall*.

Feito isso, é criado um arquivo texto, padrão do IPTables, para que o atual seja atualizado. Desta maneira, o aplicativo faz uma

interação com o serviço Netfilter/IPTables efetuando uma parada no serviço, atualização do arquivo de regras e reinicio do serviço com as novas regras aplicadas e ativas. Caso exista uma edição ou exclusão de algum usuário ou acesso, automaticamente o arquivo de regras é atualizado com as devidas alterações uma vez que este processo de alteração interfere diretamente com as regras em execução no *Firewall.*

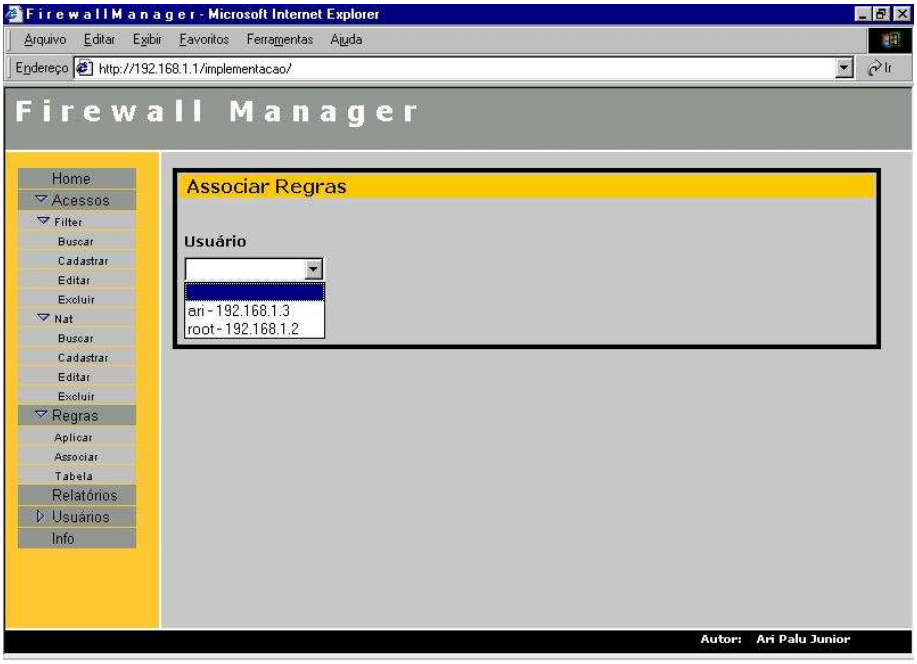

Figura 30: Escolha de Usuário para Associação

Figura 31: Seleção do usuário com Acesso – Sem Associação

Figura 32: Associação de Usuário com Acesso – Associação

Figura 33: Associação de Usuário com Acesso – Regra Gerada

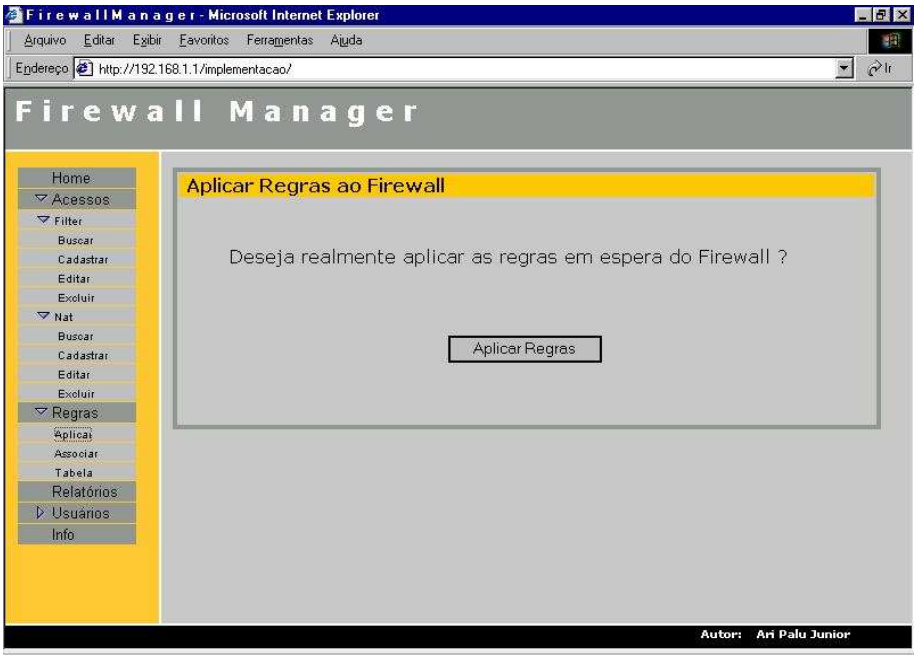

Figura 34: Aplicando as Regras no Firewall

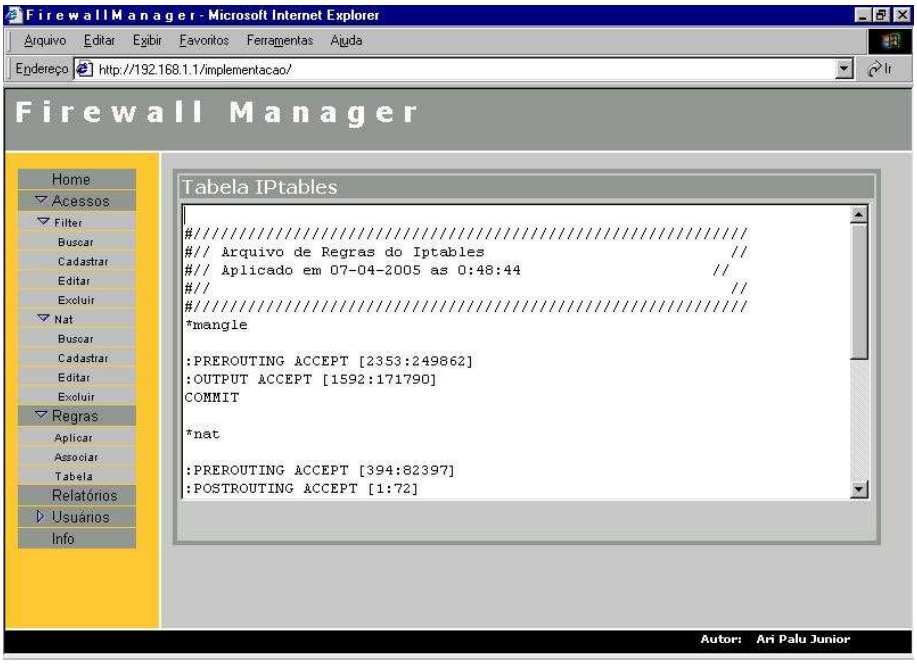

Figura 35: Tabela de Regras do Firewall

Figura 36: Associação de Usuário com Acesso – Regras Ativas

7 CONCLUSÃO

A atual realidade das redes de computadores estabelece um elevado grau de segurança. Os *Firewalls,* quando implementados de forma segura e concisa em relação a estas redes, podem vir a constituir uma excelente proposta de segurança. As várias arquiteturas e as diferentes soluções de *Firewall* disponíveis atualmente permitem ao administrador de rede, total eficácia para que consiga elevar a segurança de seu ambiente computacional.

O estudo realizado permite atestar que as diversas soluções a serem implantadas através do mecanismo de segurança *Firewall* retornam resultados positivos independente da sua complexidade.

Agregar mecanismos que agem interativamente com estes sistemas de segurança e os administradores pode ser uma solução prática e viável em se tratando de soluções em modo texto como o *NetFilter/IPTables*.

Como sugestão para trabalhos futuros, é de extrema importância para o administrador de redes o uso de ferramentas de maneira integrada, que através do ambiente gráfico, possa-se unir as soluções de maior importância em favor da corporação. Sendo assim, é de extrema importância viabilizar a centralização de toda a gerência dos mecanismos de segurança existentes em uma rede.

8 REFERÊNCIAS BIBLIOGRÁFICAS

[ANONIMO 2000] ANONIMO - Segurança Máxima - 3ª ed. Rio de Janeiro, Editora Campus.

[BERNSTEIN 1997] BERNSTEIN, Terry. Segurança na Internet. 1ªed. Rio de Janeiro, Editora Campus.

[BURNETT 2002] BURNETT, Steve. Criptografia e Segurança: O Guia Oficial RSA. 1ª ed., Rio de Janeiro, Campus, Brasil.

[CARUSO 2001] CARUSO, Marcos. Plano de contingência: sua empresa pode ficar de fora do negócio? URL: http://www.modulo.com.br. [ONLINE 10/2004]

[CYBYS 1995] CYBYS, Walter de Abrel. Ergonomia de Interface Humano Computador. URL: http://www.labiutil.inf.ufsc.br. [ONLINE 02/2005]

[CYCLADES 2000] CYCLADES, Guia de Internet de Conectividade. 6ª ed. São Paulo, Editora Senac.

[FRAINER 1991] FRAINER, Antonio Severo. Interfaces Inteligentes.

[GERLACH 1999] GERLACH, Cristiano. Técnicas Adotadas pelos Crackers para entrar em Redes Corporativas e Redes Privadas. RNP News Generation, v. 3, n°. 2.
URL: http://www.rnp.br/newsgen/9903/crackcorp.shtml. [ONLINE Fev/2005].

[HEUSER 2000] HEUSER, Carlos A. Projeto de Banco de Dados. 3ª ed. Porto Alegre, Editora Sagra Luzzato.

[LEÃO 2001] LEÃO, Osmar Ribeiro. Alternativa de segurança em redes de computadores para Linux em arquiteturas TCP/IP. URL:http://suporte.planetarium.com.br. [ONLINE 04/2004].

[MARCELO 2000] MARCELO, Antonio. Linux: Ferramentas Anti-Hackers. 1ª ed. Rio de Janeiro, Editora Brasport.

[MARCELO 2002] MARCELO, Antonio. Firewalls em Linux. 3ª ed. Rio

de Janeiro, Editora Brasport.

[MARQUEZ 2001] MARQUEZ, Alexandre Fernandez. Segurança em Redes IP URL: http://www.modulo.com.br. [ONLINE 02/2005].

[NBSO2003] Políticas de Segurança. URL:http://www.nbso.nic.br. [ONLINE 05/2004]

[NORTHCUTT 2002] NORTHCUT, Stephen. Desvendando Segurança em Redes. 1ª ed. Rio de Janeiro, Editora Campus.

[PENTA 1999] RFC 2196 - Políticas de Segurança. URL: http://penta.ufrgs.br/gereseg/rfc2196/ [ONLINE 03/2005]

[PERAZOLO 2003] PERAZOLO, Maurício. DIEC – Desenvolvimento de Interfaces Ergonomicamente Corretas. Relatório Final - Unopar, Londrina.

[PERKINS 2002] PERKINS, Charles. Firewalls. 1ª ed. São Paulo, Makron Books Brasil.

[PRESSMAN 2003] PRESSMAN, Roger S.. Engenharia de software. 3ª ed. São Paulo, Makron Books, Brasil.

[RAPOSO 2002] Alberto B. Interação na Web. URL: http://www.dca.fee.unicamp.br [ONLINE 10/2004].

[RAY 1996] RAY, Lee. Desvendando o TCP/IP. 1ª ed. Rio de Janeiro, Editora Campus.

[RUSSELL 2001] Russell, Rusty. Linux 2.4 Packet Filtering HOWTO. Revision: 1.19. Netfilter, 2001. Disponível em: http://www.netfilter.org/documentation/HOWTO/pt/packet-filtering-HOWTO.html. [Online 04/2005];

[SILVA 2004] SILVA, Antonio Mendes. Entendendo e Evitando a Engenharia Social – Revista Espaço Acadêmico, n° 43, ano IV.

[SOARES 2003] SOARES, Luiz Fernando Gomes. Redes de computadores. 2ª ed. Rio de Janeiro, Editora Campus.

[STANGER 2002] STANGER, James; LANE, Patrick T. Rede Segura Linux. 1ª ed., Rio de Janeiro, Editora Alta Books.

[TANENBAUM 2001] TANENBAUM, Andrew S.. Redes de computadores. Rio de Janeiro, Editora Campus.

[WINCKER 2001] WINCKER, Marco. IV Workshop sobre Fatores Humanos e Sistemas Computacionais.

[ZWICKY 2001] ZWICKY , Elizabeth D.. Construindo Firewalls para a Internet. 2ª ed. Rio de Janeiro, Editora Campus.

9 BIBLIOGRAFIA CONSULTADA

[CASTAGNETTO 2001] CASTAGNETTO, Jesus. Professional PHP programando São Paulo: 2ª ed.Makron Books,

[SILVA 2002] SILVA, Osmar J.. Programando com PHP 4. São Paulo: Ed.Érica, 2002.

[WELLING 2003] WELLING, Luke; THOMSON, Laura. PHP e MySQL desenvolvimento Web Rio de Janeiro. Ed. Campus

10 ANEXOS

10.1 Anexos A

Segue abaixo o Script de criação do Banco de dados da aplicação gerado pelo PhpMyAdmin no Linux.

```
# phpMyAdmin SQL Dump
# version 2.6.1-rc6
# http://www.phpmyadmin.net
#
# Host: localhost
# Generation Time: Abr 09, 2005 at 15:27 PM
# Server version: 3.23.57
# PHP Version: 4.3.2
#
# Database : `fire`
#
#
# Table structure for table `acessos`
#
CREATE TABLE `acessos` (
  `cod_ace` int(4) NOT NULL auto_increment,
  `nome` varchar(50) NOT NULL default '',
  `descricao` varchar(50) NOT NULL default '',
  `chain` varchar(20) NOT NULL default '',
  `destino` varchar(30) default NULL,
  `porta` varchar(20) default NULL,
  `protocolo` varchar(5) default NULL,
  `politica` varchar(10) default NULL,
  `tipo` char(2) default NULL,
  `dnat` varchar(31) default NULL,
  PRIMARY KEY  (`cod_ace`),
  UNIQUE KEY `cod_ace` (`cod_ace`)
) TYPE=MyISAM AUTO_INCREMENT=73 ;
#
# Table structure for table `hregras`
#
CREATE TABLE `hregras` (
  `cod_usu` int(4) NOT NULL default '0',
  `cod_ace` int(4) NOT NULL default '0',
  `tipo` char(2) NOT NULL default '',
  `data` date NOT NULL default '0000-00-00',
```

```
  `hora` time NOT NULL default '00:00:00',
  `observacao` varchar(100) NOT NULL default '',
  `situacao` char(2) NOT NULL default ''
) TYPE=MyISAM;
#
# Table structure for table `regras`
#
CREATE TABLE `regras` (
  `cod_usu` int(4) NOT NULL default '0',
  `cod_ace` int(4) NOT NULL default '0',
  `tipo` char(2) NOT NULL default '',
  `data` date NOT NULL default '0000-00-00',
  `hora` time NOT NULL default '00:00:00',
  `observacao` varchar(100) NOT NULL default '',
  `situacao` char(2) NOT NULL default '',
  PRIMARY KEY  (`cod_usu`,`cod_ace`)
) TYPE=MyISAM;
#
# Table structure for table `usuarios`
#
CREATE TABLE `usuarios` (
  `cod_usu` int(4) NOT NULL auto_increment,
  `ip` varchar(15) NOT NULL default '',
  `nome` varchar(35) NOT NULL default '',
  `descricao` varchar(35) NOT NULL default '',
  `login` varchar(15) NOT NULL default '',
  `senha` varchar(15) NOT NULL default '',
  PRIMARY KEY  (`cod_usu`),
  UNIQUE KEY `ip` (`ip`)
) TYPE=MyISAM AUTO_INCREMENT=34 ;
```