Sildenir Alves Ribeiro

Firewall em *Linux*

Monografia apresentada ao Departamento de Ciência da Computação da Universidade Federal de Lavras, como parte das exigências do curso de Pós-Graduação *Lato Sensu* em Administração em Redes *Linux,* para obtenção do título de especialista em Administração em Redes *Linux*.

Orientador
Prof. MSc. Joaquim Quinteiro Uchôa

LAVRAS
Minas Gerais – Brasil
2004

Sildenir Alves Ribeiro

Firewall em Linux

Monografia apresentada ao departamento de Ciência da Computação da Universidade Federal de Lavras, como parte das exigências do curso de Pós-Graduação *Lato Senso* em Administração em Redes Linux, para obtenção do título de especialista em Redes Linux.

Aprovada em 25 de Julho de 2004

_____

Prof. Samuel Pereira Dias

_____

Prof. Dsc. Ricardo Martins de Abreu Silva

_____

UFLA
Prof. Msc. Joaquim Quinteiro Uchôa
(Orientador)

LAVRAS

Minas Gerais – Brasil

2004

Dedicatória

Aos meus pais pela força e pelo carinho a mim demonstrado em todos os momentos de minha vida. Aos meus irmãos e amigos que sempre acreditaram em meu potencial. A minha namorada Viviani Freitas, pela paciência e pela força. E aos professores e colegas do ARL pela troca continua de conhecimentos, essencial para a conclusão deste projeto.

Agradecimentos

Agradeço a Deus pela luz e por ter mostrado-me o caminho nos momentos mais difíceis. A todos os professores do ARL que colaboraram com a realização deste projeto, transmitindo-me os seus conhecimentos, em especial ao professor Joaquim Quinteiro Uchôa pela orientação e pelo apoio irrestrito em todo período do curso. Aos meus colegas de turma que juntos caminhamos durante este longo período visando um único objetivo.

Resumo

Esta monografia trata exclusivamente de assuntos relacionados à segurança em redes de computadores. Abordando temas polêmicos como: estratégia e adoção de políticas de segurança, ataques e invasão a sistemas de computadores e a utilização de *firewalls* como ferramenta de apoio à segurança em rede e sistemas de computadores. O texto traz ainda uma comparação entre os principais sistemas operacionais dispostos no mercado, apontando a usabilidade e portabilidade com relação à implantação e configuração de *software* de *firewalls*. Dentre os diversos *firewalls* disponíveis no mercado, este trabalho destaca o *CheckPoint FireWall-1* como solução proprietária e o *Iptables* como solução gratuita e livre. O *Iptables* é apresentado neste texto com mais riqueza de detalhes e informações por se tratar do objeto fim deste trabalho.

Palavras Chaves: *Firewalls, Checkpoint Firewall-1, Iptables*

Sumário

Lista de Figuras

1 Introdução

A rede mundial de computadores trouxe uma grande preocupação, o risco de invasão de redes e sistemas de computadores. Em contrapartida surgiu o *firewall*, vulgarmente chamado de parede de fogo, que tem como principal finalidade o isolamento entre duas ou mais redes, provendo assim uma proteção maior ao sistema.

Um *firewall* é um sistema, ou um conjunto de sistemas, que força uma política de segurança entre uma rede interna segura e uma rede insegura, como a *internet.* Um *firewall* tende a ser visto como uma proteção entre a *internet* e uma rede privada. De modo geral um *firewall* deve ser considerado como um meio para dividir duas ou mais redes seguras, de uma ou mais redes não seguras, como a *internet.*

Existe vários *software* de *firewalls* no mercado para os diversos tipos de sistemas operacionais, que serão abordados no decorrer deste trabalho.

O objetivo primário deste trabalho é apresentar soluções relativas à segurança em redes de computadores com a utilização dos *firewalls,* abordando e apresentando os vários *software* de *firewalls* para os principais sistemas operacionais. O uso do *Iptables* em empregado *Linux* será abordado com mais destaque devido a sua relevância, apontando as suas principais características e funcionamento.

Este trabalho apresenta a importância e a relevância do uso de *firewalls* em uma rede de computadores. O Capítulo 2 está centrado nos conceitos básicos e definição dos *firewalls*, demonstrando uma visão mais geral sobre os *firewalls*. O Capítulo 3 trás uma questão muito polêmica e extremamente importante para qualquer rede e sistema de computadores: o

uso de políticas de segurança, que é o princípio para obtenção de um sistema, se não seguro, ao menos confiável.

Já no Capítulo 4, o enfoque está nos sistemas operacionais disponíveis no mercado, destacando-se; o *Linux*, o *Solaris* e o *Windows*. Este capítulo também destaca o *Chekpoint Firewall-1*, uma solução proprietária para implantação de um *firewall* em uma rede, objetivando um comparativo com as opções gratuitas e livre, oferecidas para o sistema operacional *Linux*. O Capítulo 5 apresenta os componentes dos *firewalls*, baseando-se nas opções disponíveis para *Linux.*

O Capitulo 6 é dedicado inteiramente ao *firewall Ipatbles*, apresentando seus componentes e sua implantação e configuração. Ao final deste texto encontram-se dois apêndices, o Apêndice A apresenta um conjunto de regras construídas para implementar um *firewall* em um servidor na UNEMAT – Universidade do Estado de Mato Grosso; o Apêndice B traz uma lista contendo os comandos utilizados no *Iptables*.

2 Visão Geral de *Firewalls*

A expansão das redes de computadores e a chegada da *internet,* trouxe para o mundo virtual problemas de segurança de sistemas. Problemas estes, antes não muito freqüentes, já que para conectar-se a uma rede privada um invasor precisava de um modem para efetuar uma discagem direta usando a rede de telefonia pública.

A *internet* possui uma eficiência de comunicação muito grande, o que facilita o acesso às redes privadas. As conexões direta via *internet* facilitam a exploração dos recursos privados em uma rede por pessoas de toda a parte do mundo. Este é um problema sério de segurança, pois as facilidades de acesso fornecidas pela rede mundial de computadores não faz nenhuma restrição quanto ao acesso ao conteúdo disponível e a quem tem acesso a este conteúdo.

Ao conectar uma rede privada a *internet*, está na verdade conectando a rede diretamente a todas as outras redes ligadas à *internet* diretamente. Não existe nenhum ponto central de controle de segurança na rede mundial, isto é uma porta aberta para pessoas mal intencionadas.

Com os problemas inerentes à segurança, vindo junto com a *internet*, surgiram ferramentas, dispositivos e práticas de controle de acesso para tornar as redes mais seguras, como os *firewalls*.

Os *firewalls* podem ser definidos como um computador acompanhado de um *software*, que protege a rede privada da *internet*. Assim, a rede privada fica isolada da *internet*, ou seja, não possui uma comunicação direta com a

internet e sim com o *firewall*. Com isso toda vez que uma solicitação de conexão for realizada com um *host internet*, será realizado primeiro a conexão com o *firewall* e somente depois a conexão com a rede externa é liberada.

A Figura *2.0.1* exemplifica esta definição.

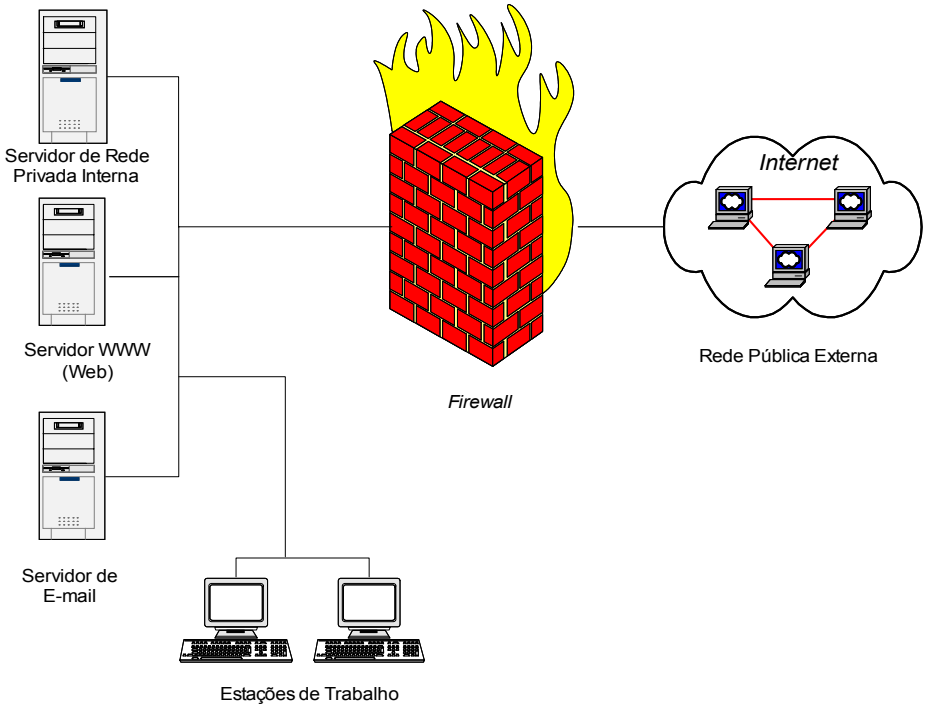

Figura 2.0.1 – Exemplo de um *Firewall*

Esta definição nos faz lembrar uma parede de proteção entre a rede privada e a *internet*, representando assim com fidelidade a tradução de *firewall* do inglês, “parede de fogo”.

Hatch, *et al* (2000), define um *firewall* como uma “parede a prova de fogo” que protege uma máquina ou uma rede local contra o restante do ciberespaço.

2.1 Limitações de um Firewall

É importante observar que um *firewall* é uma ferramenta utilizada para impor uma política de segurança, tão eficiente quanto à própria política de segurança empregada na rede pelo projetista ou administrador, mas que não é capaz de as questões de segurança sozinho.

Conforme abordado por Welch-Abernathy (2003), um *firewall* não poderá proteger um sistema dos seguintes perigos.

- **Uso malicioso dos serviços autorizados.** Um *firewall* não pode, por exemplo, impedir que alguém use uma sessão *TELNET*[1] autenticada, para comprometer as máquinas internas.
- **Usuários que não passam pelo *firewall.*** Um *firewall* só pode restringir as conexões que passam por ele; portanto, não poderá protegê-lo contra pessoas que evitam o *firewall*, por exemplo, por meio de um servidor *dial-up*, que pode ser implantado antes do *firewall*.
- **Engenharia social.** Se os intrusos puderem, de alguma forma, obtiver senhas, as quais não estão autorizados a ter ou comprometer de alguma outra forma os mecanismos de autenticação por meio de engenharia social, o *firewall* não os impedirá. Por exemplo, um invasor poderá ligar para um de seus usuários fingindo ser o administrador do sistema e pedir sua senha para resolver um problema.
- **Falhas no sistema operacional.** Um *firewall* é tão seguro quanto um sistema operacional em que está instalado. Existem muitas falhas presentes nos sistemas operacionais, contra as quais um *firewall* não pode protegê-lo. É por isso que é importante proteger corretamente o sistema operacional aplicando os *patches*[2] de segurança necessário

[1] *TELNET* – Serviços que permite que uma máquina cliente possa acessar e executar tarefas em um servidor.
[2] *Patches – Aplicativos utilizados pra corrigir possíveis erros em um sistema operacional.*

antes de instalar o firewall e atualizando periodicamente os *patches* após a instalação completa do *firewall*.

- **Todas as ameaças que podem ocorrer.** Normalmente os invasores estão à frente dos projetistas e fabricantes de *firewalls,* os quais desenvolvem aplicativos sobre os problemas e falhas descobertos *pelos* invasores*.* Assim, um *firewall* por mais atual que seja não pode proteger a rede e o sistema com totalidade.

2.2 Componentes de um Firewall

Como pode ser visto anteriormente, a idéia central de um *firewall* é manter uma conexão segura com a *internet*. É através dos *firewalls* que podemos controlar inspecionar, aprovar ou rejeitar cada tentativa de conexão feita entre a rede interna e a rede externa, *internet*. Os *firewalls* de grande poder deve proteger uma rede em todas as camadas de *software*, que vai da camada de enlace até a camada de aplicação. Em contrapartida, um *firewall* sozinho, ou seja, um *software* de *firewall* instalado sem estar configurado devidamente, com um banco de regras bem definido, somente irá consumir inutilmente recursos de *hardware* e *software*, aumentando assim, o tempo necessário para conectar, autenticar e criptografar os dados em uma rede. Diante deste problema deve-se estar atento, e configurar adequadamente o *firewall* e seus componentes nas fronteiras da rede, interconexões ou *gateways*, que fornecem o acesso a outras redes.

Um bom *firewall* opera usando seus componentes fundamentais, que são estabelecidos pelos três métodos abaixo.

- **Filtragem de Pacotes:** rejeita pacotes *TCP/IP* de *hosts* não autorizados e rejeita tentativas de conexões de serviços não autorizados;
- ***NAT (Network Address Translation* – conversão de endereços de redes):** converte os endereços *IP* dos *hosts* internos para ocultá-los de qualquer monitoração externa. A *NAT* é também chamada de mascaramento de *IP*;
- **Serviços *Proxy*:** faz com que as conexões de aplicativos de alto nível, em benefícios do *host* interno, quebrem completamente a conexão da camada de rede entre o *host* interno e o *host* externo.

A maioria dos *firewalls* ainda realiza dois outros serviços importantes e fundamentais relacionados à segurança, que são:

- **Autenticação Criptografada:** permite aos usuários da rede pública provar sua identidade ao *firewall* para obter acesso à rede privada com segurança de seus dados garantida.
- **Túneis Criptografados:** estabelece uma conexão segura entre duas redes privadas através de um meio público a *internet*. O tunelamento também é chamado de *VPN*[3].

> *Existem centenas de produtos de **firewalls** no mercado, praticamente todos usam os métodos básicos, filtragem de pacotes, NAT e serviços de **proxies**, para oferecer um serviço de segurança (STREBE; PERKINS, 2000).*

Os conceitos de filtragem de pacotes, *NAT*, *proxies*, autenticação criptografada e túneis criptografados serão melhores apresentados no capítulo 4 deste trabalho.

[3] VPN – *Virtual Private Network* – Rede Privada Virtual

2.3 Segurança Eficaz de Fronteiras

Como mencionado anteriormente, para manter um nível mínimo de segurança, o administrador deve conhecer os conceitos de fronteiras, local onde o *firewall* deve ser implantado, para que as três funções básicas sejam realizadas pelo *firewall*, garantindo assim a segurança mínima desejada. Os *firewalls* ainda precisam ser dedicados, evitando o uso de outros serviços, como *e-mail*, *web* e serviços públicos executados no próprio *firewall*. Esta ultima observação é dispensada se o *software* de serviço acompanha o software do fabricante do *firewall*.

Outro ponto importante que o administrador deve sempre procurar minimizar, são os serviços a serem executados no *firewall*, reduzindo a complexidade do *software* executado na máquina, diminuindo assim, a probabilidade de erros no sistema operacional e no *software* de segurança.

No caso de sistemas operacionais em ambientes gráficos, como: o *Windows NT*, *2000* e *2003 Server*, nenhum dos serviços do painel de controle é necessário para um computador que é executado somente como *firewall*. No caso do *Linux* e *Unix*, instale somente os pacotes necessários para a operação do *firewall* ou selecione a opção de instalação do *firewall*, caso a distribuição tenha. Normalmente não é preciso estes cuidados, porque o programa de instalação irá desativar todos os serviços desnecessários automaticamente.

Para se ter uma total proteção das fronteiras é preciso também reforçar a idéia de um único ponto de controle no *firewall*. Caso sua rede disponha de

mais de um *firewall*, certifique que todos estão configurados corretamente e da mesma maneira.

2.4 Conceitos de Segurança

Conceituar segurança não é uma questão muito simples, além de ser um ponto que sempre gera muita discussão e poucas conclusões em comum, mas é consenso de todos que este assunto deve ser observado com muito cuidado, dada sua tamanha importância e relevância. Dentro deste assunto o que mais tira o sono dos administradores de sistemas é a questão da violação de segurança.

As violações de segurança correspondem à revelação não autorizada de informações, modificações e produções não autorizada da informação e negação de serviço. Evitar estas violações em sistemas complexos é sempre uma tarefa árdua.

A segurança geralmente é vista como uma qualidade de serviço, destinada a garantir o fornecimento e o funcionamento dos serviços oferecidos e não como um serviço específico. A segurança está fundamentada sobre cinco propriedades que devem ser mantidas conforme descreve (Melo, 2004).

Confidencialidade – A informação só deverá ser revelada para usuários autorizados a acessá-la;

Integridade – A informação não poderá ser modificada, intencionalmente ou acidentalmente, por usuários que não possuam direito para tal;

Disponibilidade – O uso do sistema não poderá ser negado, de forma maliciosa, a usuários autorizados;

Autenticidade – O acesso ao sistema só deverá ser feito por usuários autênticos;

Não-repúdio – Um usuário não poderá negar a sua participação na ocorrência de um evento ou transação.

2.5 Vulnerabilidades, Ameaças e Ataques

As violações de segurança ocorrem devido à exploração das vulnerabilidades existentes nos sistemas. As vulnerabilidades em sistemas computacionais sempre estiveram presentes e pode ser destacadas como: um erro de programação, erro na configuração ou até mesmo um erro de operação. Estes problemas podem permitir que usuários não autorizados entrem no sistema ou mesmo que usuários autênticos executem ações não autorizadas, podendo assim comprometer o funcionamento correto do sistema. Tais ações são especificadas por Melo (2003) e apresentadas abaixo:

- **Ameaça**: consiste em uma possível ação que, se concretizada poderá produzir efeitos indesejados ao sistema, comprometendo a confidencialidade, integridade, disponibilidade e a autenticidade.
- **Ataque**: é a concretização de uma ameaça, explorando alguma vulnerabilidade do sistema, executado por algum intruso, de forma maliciosa ou não;
- **Interceptação:** onde uma parte não autorizada obtém acesso à informação (violação de revelação não autorizada de informação);
- **Interrupção:** o fluxo normal da mensagem é interrompido, impossibilitando que a informação chegue ao destino (negação de serviço).

- **Modificação**: uma parte não autorizada modifica a informação recebida da origem e a transmite para o verdadeiro destino (violação de modificação não autorizada de informação).
- **Personificação**: entidade não autorizada transmite uma mensagem maliciosa pela rede, se passando por uma parte autêntica.

Em sistemas computacionais as ameaças são constantes e uma maneira de evitar os ataques é identificar e corrigir as vulnerabilidades existentes nos sistemas, algo que não é tão simples quanto parece. Sistemas mais complexos tendem a possuir mais brechas de segurança, porém, são nesses sistemas que a segurança é mais enfatizada, sabendo que o comprometimento desses sistemas geraria enormes prejuízos financeiros.

3 Políticas e Modelos de Segurança

Este capítulo aborda um tema que não poderia ficar de fora deste trabalho. Trata-se das políticas e modelos de segurança, que são fortemente recomendas em qualquer sistema, desde os mais simples até os mais complexos.

3.1 Políticas de Segurança

Uma política de segurança de sistemas é definida como um conjunto de diretrizes, normas e procedimentos, os quais estabelecem os limites de operação dos usuários. A política de segurança é feita sob medida para um sistema específico e não para uma classe geral de sistemas.

As diretrizes ditadas em uma política de segurança indicam o que cada componente do sistema (usuários, máquinas, etc.) pode ou não pode fazer. As normas indicam o que cada componente está habilitado a fazer e como deverá ser feito. Portanto, são ditados os procedimentos que devem ser tomados para cada estado do sistema, onde o sistema poderia sair desde um estado normal

de funcionamento até um estado de emergência após algum evento inesperado ou malicioso.

Segundo MELO (2002), as políticas de segurança de sistemas diferem em três ramos: segurança física, segurança gerencial e segurança lógica, observados a seguir.

- **Política de segurança física –** preocupa-se em proteger o meio físico em que opera o sistema. São medidas definidas contra desastres como: incêndio, alagamento, terremoto, etc; e também são definidas medidas para proteger o acesso físico ao provedor do sistema, fornecendo meios de proibir o acesso de pessoas não autorizadas.
- **Política de segurança gerencial –** preocupa-se com o ponto vista organizacional, definindo os processos que devem ser tomados para seleção de pessoal, e até definindo os processos para criação e manutenção das próprias políticas de segurança.
- **Política de segurança lógica -** é a mais habitual e bastante utilizada no dia-a-dia. Esta política define quais usuários terão direito de acesso ao sistema e quais são os direitos que cada usuário possuirá. São aplicados aqui dois conceitos: a autenticação, onde um usuário necessita se identificar ao sistema para que possa obter acesso ao recurso; e a autorização, onde o usuário precisa provar que possui direitos sobre o recurso o qual ele deseja acessar.

3.2 Modelos de Segurança

Os modelos de segurança são comumente confundidos com mecanismos de segurança. Um modelo de segurança é uma expressão, muitas vezes, formal de uma classe de políticas de segurança, abstraindo

detalhes e concentrando-se em conjunto de comportamentos que são utilizados como base para defini-las de políticas de segurança.

Em seu trabalho, MELO (2002) afirma que os modelos de segurança são divididos em três tipos básicos: discricionários (*discretionary*), obrigatórios (*mandatory*) e os modelos baseados em papéis *(roles).*

3.2.1 Modelos Discricionários

Os modelos discricionários garantem o acesso de usuários às informações com base na identidade do usuário e nas autorizações que determinam, para cada usuário (ou grupo de usuários) e para cada objeto do sistema, a forma de acesso que o usuário está autorizado a realizar sobre o objeto.

O modelo de matriz de acesso é um modelo conceitual discricionário, o qual especifica os direitos que cada sujeito possui sobre cada objeto do sistema. Os *principais*[4] são tipicamente usuários, processos ou máquinas representando o usuário, podendo assumir diferentes sujeitos em diferentes ocasiões.

3.2.2 Modelos Obrigatórios

Enquanto os modelos discricionários são direcionados a definição, modelagem e controle do acesso à informação, os modelos obrigatórios (*mandatory models*), ou não-discricionários, também se preocupam com o

[4] *Principais*: Termo utilizado comumente para identificar: usuário, processos e máquinas atuando em nome dos usuários de sistemas, considerados aptos pela política estabelecida (política de segurança lógica) em suas ações no sistema. O sentido contrário (usuários não autorizados pelas políticas) é identificado como *intrusos*.

fluxo de informações no sistema. Nos modelos obrigatórios, classes de segurança são atribuídas aos objetos e aos sujeitos, as quais são designadas como rótulos de segurança (*Security label*). Para os objetos o rótulo é chamado de classificação, enquanto que para os sujeitos o rótulo é chamado de habilitação (*clearance*). A classificação representa a sensibilidade dos dados rotulados, enquanto que a habilitação representa a confiança no sujeito em não revelar informações sensíveis a outros sujeitos.

3.2.3 Modelos Baseados em Papéis

Modelos baseados em papéis (*roles-based models*) restringem o acesso dos *principais* às informações de acordo com as atividades que estes desempenham no sistema. Os papéis podem ser definidos como um conjunto de ações e responsabilidades associadas com uma atividade de trabalho em particular. Logo, ao invés de especificar o que cada *principal* está autorizado a fazer, essa autorização é dada aos papéis. Um *principal* que desempenha um papel só estará apto a realizar ações no sistema de acordo com as permissões que o papel possui. Em diferentes situações um *principa*l poderá assumir diferentes papéis e também um papel pode ser assumido por diferentes principais, às vezes simultaneamente.

As vantagens apresentadas pelo modelo baseado em papéis são:

- **Gerência de autorização** - As especificações de autorização são divididas em duas partes, a associação de direitos de acesso a papéis e associação destes *papéis* aos *principais*, deixando assim o gerenciamento de segurança mais simplificado, por exemplo, no caso de um principal ter que desempenhar um novo papel na organização.

- **Hierarquia de *papéis*** - Em muitos tipos de aplicações existe uma hierarquia natural de *papéis* baseada nos princípios da generalização e

especialização, permitindo assim que permissões sejam herdadas e compartilhadas.

- **Privilégio mínimo** - *Papéis* permitem que um *principal* trabalhe com o mínimo privilégio exigido por uma determinada tarefa, garantindo assim, que um *principal* fará uso dos benefícios e privilégio somente para realizar as tarefas especificadas pelos *papéis*.

- **Separação de tarefas (*duties*)** - Tal propriedade parte do princípio que a nenhum *principal* deve ser dado privilégios o bastante além do que o mesmo possa utilizá-los, de forma maliciosa, em benefício próprio. Por exemplo, a pessoa que autoriza os pagamentos de salário não deveria ser a mesma pessoa que os efetua. E ainda, a efetuação dos pagamentos não poderá ocorrer sem que antes ocorra a autorização dos mesmos.
- **Classificação dos objetos** - No modelo baseado em *papéis* a classificação dos *principais* ocorre de acordo com as tarefas que cada um poderá executar no sistema. Analogamente, tal classificação poderia ser dada aos objetos do sistema. Assim, as autorizações de acesso dos papéis fariam sobre as classes de objetos e não em objetos específicos.

3.3 Autenticação e Autorização

O processo de identificação em sistemas computacionais consiste de um conjunto de procedimentos e mecanismos que permitem que agentes externos, como: usuários, dispositivos, etc, sejam identificados como usuários autorizados segundo as políticas de segurança adotadas no sistema. No processo de identificação, alguns mecanismos de segurança exigem um nome de usuário (*login*) e uma senha, assim garantem a identidade do agente

externo, dando a este à possibilidade de usufruir do sistema. Já processo de autenticação, composto por um conjunto de mecanismos e procedimentos, dá ao sistema a possibilidade de assegurar que um principal é realmente quem ele diz ser.

Se considerado um sistema distribuído, em uma comunicação entre duas partes, o serviço de autenticação preocupa-se em assegurar autenticidade da comunicação.

Neste caso, dois aspectos são envolvidos. O primeiro, no início da conexão: o serviço de autenticação garante a autenticidade das duas entidades, garantindo que cada entidade é quem diz realmente ser. O segundo, o serviço deve assegurar que a comunicação não é interferida de tal maneira que uma terceira parte não autêntica consiga personificar uma das duas partes autênticas com o propósito de transmitir ou receber informação ações de forma não autorizada.

Desta forma, um serviço de autenticação pode promover a identificação de principais, a autenticação mútua entre as partes e a ainda garantir a autenticidade dos dados transmitidos entre as partes.

3.4 Definindo uma Política de Segurança

Para se ter uma rede segura não basta configurar um *firewall* na fronteira de sua rede, é preciso antes de qualquer coisa, estabelecer uma política de segurança clara e bem definida, definindo inclusive uma política de uso, onde serão estabelecidas questões de acesso e restrições do usuário a rede e o sistema, respectivamente.

> *MANN, MITCHEL[5], citado por UCHÔA (2003); Uma política de segurança incorpora os resultados de uma análise de risco em um plano que providencia procedimentos para gerenciar um ambiente computacional. Em particular, ela fornece ao administrador do sistema linhas operacionais para o ambiente, tais como regras para o gerenciamento de contas de usuários, procedimentos de instalação de sistemas...*
>
> *SOARES; LEMOS; COLCHER[6], citado por UCHÔA (2003); Uma política de segurança é um conjunto de leis, regras e práticas que regulam como uma organização gerencia, protege e distribui suas informações e recursos. Um dado sistema é considerado seguro em relação a uma política de segurança, caso garanta o cumprimento das leis, regras e práticas definidas nesta política.*

O uso de *firewalls* permite a imposição de uma política de segurança, mesmo não sendo a única ferramenta para impor esta política. Uma boa política de segurança deve vir acompanhada de outros itens fundamentais conforme apontado por (UCHÔA, 2003).

Uma política de segurança deve possuir as seguintes características.

- Ser implementável através de procedimentos de administração de sistemas, regras de uso, ou outros métodos apropriados;
- Ser reforçada com ferramentas de segurança e sansões;
- Definir claramente as áreas de responsabilidade para usuários e administradores de sistemas.

[5] MANN, S; MITCHELL, E.L., Linux System Security: An Administrator's Guide to Open Source Security Tools, New Jersey: Prentice-Hall, 2000.
[6] SOARES, L.F.G.; LEMOS, G.; COLCHER, S. Redes de Computadores: das LANS, MANS e WANS às Redes ATM. 2ª Ed. Rio de Janeiro: Campus, 1995.

Com uma política de segurança claramente definida, o administrador sabe exatamente quem manterá seus *firewalls* e que tipos de mudanças são permitidos. O contrário acontece com uma política mal definida.

> *Uma política de segurança é o primeiro passo crítico rumo à proteção da rede de sua organização (WELCH-ALBERNATHY, 2003).*

Dentro do processo de definição da política de segurança, está também contida uma questão crucial para que uma política de segurança possa ser empregada, a documentação.

Observa-se o seguinte caso: o que foi imposto hoje poderá ser alterado amanhã. As regras podem ser incluídas ou removidas aleatoriamente sem qualquer plano de coerência real. O resultado disso é de uma ferramenta eficaz configurada de modo ineficaz.

Definir uma política de segurança é documentar de forma simples, clara e objetiva o que deverá ser e quais recursos que deve ser protegidos e sob que condições o acesso a esses documentos será concedidos ou negados.

Normalmente são criadas políticas de seguranças extensas, que abordam situações imagináveis gerando documentos grandes, complicados e desajeitados e que ninguém se dispõe a ler e a entender. Assim, o objetivo primeiro do administrador ou projetista é criar um documento simples que possa ser usado por outra pessoa com certo grau de facilidade.

4 *Firewall*: Conceitos e Discussões

Um *firewall* é um sistema, ou grupo de sistemas, que força uma política de segurança entre uma rede interna segura e uma rede insegura como a Internet. Os *firewalls* tende a ser visto como uma proteção entre a *Internet* e uma rede privada. Mas falando de modo geral um *firewall* deve ser considerado como um meio para dividir o mundo em duas ou mais redes: uma ou mais redes seguras e uma ou mais redes não seguras. Um *firewall* pode ser um PC, um roteador, um *midrange*[7], um *mainframe*[8], um *Unix workstation* ou uma combinação destes, determinando quais informações ou serviços podem ser acessados de fora, e quem tem permissão para usar estas informações e serviços.

Geralmente, um *firewall* está instalado no ponto onde a rede interna é segura e a rede externa é insegura, este ponto que é conhecido como ponto de sufoco ou *choke point*.

> *A fim de entender como um firewall trabalha, considere a rede como sendo um edifício onde o acesso deve ser controlado. O edifício tem um salão de entrada como o único ponto de entrada.*

---

[7] *Midrange – Dispositivo que realiza uma ou mais conexão entre um ou mais dispositivos. Em informática um midrange é tido como um servidor para uma rede cliente servidor. Trata-se geralmente de um computador de médio porte.*

[8] *Mainframe – Computador de grande porte.*

> *Neste salão de entrada estão às recepcionistas de visitantes, guarda-costas de visitantes, vídeo-câmeras gravando as ações dos visitantes e leitores de distintivo que autenticam os visitantes que entram no edifício (RUSSEL, 2002).*

Embora, estes procedimentos possam trabalhar bem para controlar o acesso ao edifício, conforme definição de RUSSEL (2002), se uma pessoa, sem autorização tiver sucesso ao entrar, não existe nenhum modo de proteger o edifício contra alguma ação do intruso. Porém, se os movimentos do intruso for monitorado, pode ser possível descobrir qualquer atividade suspeita.

Semelhantemente, um *firewall* é projetado para proteger os recursos de informações da organização controlando o acesso entre a rede interna segura e a rede externa insegura. Todavia, é importante notar que o *firewall* é projetado para permitir que os dados confiáveis passem através dele, negue os serviços vulneráveis e previna a rede interna de ataques de fora. Um ataque recentemente criado pode penetrar no *firewall* a qualquer hora. O administrador da rede deve examinar todos os *logs* e alarmes gerados pelo *firewall* em uma base regular. Caso contrário, não é possível proteger a rede interna de ataques externos.

4.1 Componentes de um Sistema de Firewall

Como mencionado previamente, um *firewall* pode ser um *PC*, um *midrange,* um *mainframe,* uma *Workstation Unix*, um roteador ou a combinação destes. Conforme RUSSEL (2002), dependendo dos requisitos, um *firewall* pode consistir em um ou mais dos componentes funcionais seguintes:

- Roteador de filtro de pacote;
- Aplicação em nível de *gateway - proxy*;
- Circuito de nível de *gateway.*

Cada um destes componentes tem uma função diferente. Geralmente, a fim de construir um *firewall* efetivo, estes componentes são usados juntos. A Figura 4.1.2 ilustra a configuração e o uso destes componentes, onde a rede privada, estabelecida pelos servidores e clientes, está conectada a um concentrador (*hub*), o qual está conectado ao *firewall*, protegendo assim a rede privada da rede pública externa (*internet*).

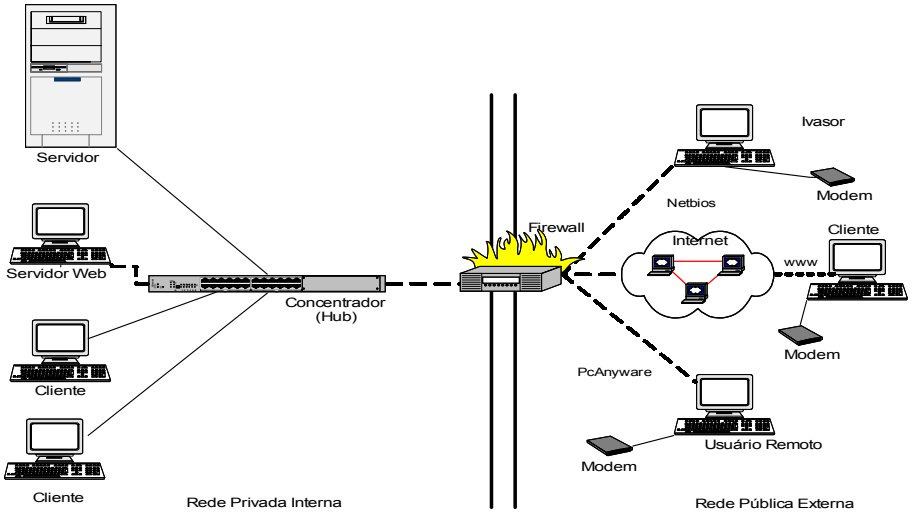

Figura 4.1.2 - Configuração dos componentes de *firewall*

4.2 Roteador de Filtro de Pacote

Na maioria das vezes, a filtragem de pacote é realizada, utilizando-se uma roteador que possa enviar pacotes de acordo com as regras de filtragem. Quando um pacote chegar ao filtro de pacotes, o roteador extrai as informações do cabeçalho do pacote e faz decisões de acordo com as regras de filtro, e decide se o pacote passará por ele ou se será descartado. As informações seguintes podem ser extraídas do cabeçalho de pacote:

- Endereço *IP* da Fonte.
- Endereço *IP* de Destino.
- Porta *TCP/UDP* da Fonte.
- Porta *TCP/UDP* de Destino.
- Tipo de mensagem de *ICMP*.
- Informações de encapsulamento do protocolo (*TCP, UDP, ICMP* ou Túnel de *IP*).

A filtragem de pacotes através de roteadores é ilustrado na Figura *4.2.3,* com a rede privada interna conectada ao roteador, onde os pacotes são filtrados. O roteador por sua vez conecta-se ao *firewall,* que é conectado ao provedor de acesso a internet. Deste modo a rede privada interna fica protegida da rede externa pública, evitando que os usuários externos acessem a rede privada interna diretamente.

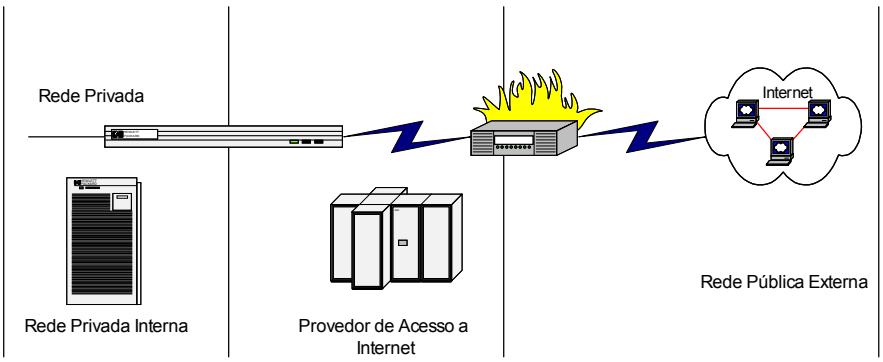

Figura 4.2.3 – Roteador de Filtro de Pacotes

4.3 Filtro de Nível de Serviço

Já que a maioria dos serviços usa números de porta do famoso *TCP/UDP*, é possível permitir ou negar que os serviços usem informações relacionadas a portas no filtro. Por exemplo, um servidor de *FTP* escuta as conexões na porta *TCP* 20 e 21. Então, permitir que conexões de *FTP* passem através de uma rede segura, deve o roteador ser configurado para permitir pacotes que contenha a porta 20 e 21 do TCP em seu cabeçalho. Por outro lado, existem algumas aplicações, como *NFS* e *RPC* e usam portas específicas para cada conexão. Permitindo estes tipos de serviços poderiam causar problemas de segurança.

4.4 Filtro de Nível de Fonte/Destino

As regras de filtragem de pacotes permitem a um roteador negar ou autorizar um pacote de acordo com o destino ou as informações da fonte do cabeçalho do pacote. Na maioria dos casos, se um serviço está disponível, somente aquele servidor particular será permitido para usuários de fora. Outros pacotes que tenham outros destinos ou nenhuma informação de destino em seus cabeçalhos serão descartados.

4.5 Filtro Avançado

Como mencionado previamente, existem tipos diferentes de ataques que ameaçam a segurança de rede isoladamente. Alguns deles podem ser descartados, usando regras de filtro avançadas, como a verificação das opções de IP.

4.6 Limitações do Filtro de Pacote

As regras de filtragem de pacote são às vezes muito complexas. Embora, existam algumas aplicações de teste disponível, é ainda possível deixar alguns buracos na segurança da rede. Os filtros de pacote não fornecem uma proteção absoluta para uma rede. Para alguns casos, pode ser necessário restringir algum conjunto de informações, como por exemplo, um comando que passa através da rede interna segura. Não é possível controlar

os dados com filtros de pacote porque eles não são capazes de entender o conteúdo de um serviço em particular. Para este propósito, deve-se utilizar um controle em nível de aplicação.

4.7 Exemplos de Firewall

Um *firewall* consiste em um ou mais elementos de *software* que é executado em um ou mais *host*. Os *hosts* podem ser sistemas de computador de propósito geral ou especializado como roteadores.

Existem quatro exemplos importantes para *firewalls*, conforme apresentados por ANDREASSON (2002). Estes exemplos são especificados abaixo:

- *Firewal*l de Filtro de Pacote;
- *Firewall* de *Gateway Dual-homed*;
- *Firewall* de *host Screened*;
- *Firewall* de *Sub-rede Screened*.

Estes exemplos de *firewall* são comumente usados em vários tipos de *firewall* e serão descritos abaixo.

4.7.1 *Firewall* de Filtro de Pacote

O *firewall* de filtro de pacote é comumente usado em um tipo de *firewall* barato. Neste tipo de *firewall* existe um roteador entre a rede interna segura e a rede externa.

Uma roteador de filtro de pacote permite o tráfego entre as redes usando as regras de filtro de pacote para permitir ou negar o tráfego. Um roteador de filtro de pacote tem as mesmas desvantagens do filtro de pacote roteado. Geralmente, uma roteador de filtro de pacotes é configurado para negar qualquer serviço se ele não for explicitamente permitido. Embora esta abordagem previna alguns ataques potenciais, ela ainda é aberta a ataques que resultam de configurações impróprias nas regras de filtro.

4.7.2 Firewall de Gateway Dual-homed

Um *host dual-homed* tem pelo menos duas interfaces de rede e pelo menos dois endereços *IP*. Se o *IP* enviado não for ativo, todo tráfego de IP entre as duas interfaces será passa pelo *firewall*. Deste modo, não existe nenhuma opção para um pacote passar pelo *firewall* a menos que o serviço de *proxy* ou *socks* relacionados são definidos no *firewall*. Comparado ao *firewall* de filtro de pacote, o *firewall* de *gateway* de *dual-homed* tem certeza que qualquer ataque que vem de serviços desconhecidos serão bloqueados. Um *gateway* de *dual-homed* cumpre o método em que tudo não permitido é especificamente negado.

4.7.3 Firewall de host Screened

Este tipo de *firewall* consiste em um filtro de rota de pacote e um *gateway* de nível de aplicação. O *router* é configurado para enviar todos os tráfego para o *host* e em alguns casos também para o servidor de informações. Já que a rede interna está na mesma sub-rede que o *host*. A política de segurança pode permitir que os usuários internos possam acessar o lado de fora diretamente, ou forçar que eles utilizem os serviços de *proxy* para acessar a rede externa. Isto pode ser alcançado configurando as regras

de filtro de roteador de forma que o roteador só aceita tráfego que origine do *host* principal.

4.7.4 Firewall de Sub-rede Screened - Zona Desmilitarizada

Este tipo de *firewall* consiste em dois roteadores de filtro de pacote e um *bastian host*[9]. *Screened subnet firewalls* fornece a segurança de nível mais alto no meio dos exemplos de *firewall*. Isto é alcançado criando uma zona desmilitarizada (*DMZ)* entre a rede externa e a rede interna, de forma que, o roteador exterior só permite acesso de fora ao *bastian host* e o roteador interno só permita acesso a rede interna até o *bastian*.

Já que o roteador exterior só anuncia a *DMZ*, para a rede externa, o sistema na rede externa não pode alcançar a rede interna. Semelhantemente, o roteador interno anuncia o *DMZ* para a rede interna, os sistemas da rede interna não podem alcançar a Internet diretamente. Este fornece uma forte segurança, pois um intruso tem que penetrar três sistemas separados para alcançar a cadeia interna.

[9] *Bastion Host: É uma máquina segura instalada em um ponto crítico da rede, chamado de zona desmilitarizada (zona sem proteção) ou DMZ.*

5 *Firewall* e Sistema Operacional

Assim como existe vários sistemas operacionais no mercado, existe também vários *software* de *firewall*. O desafio do administrador de sistema é escolher o *software* de *firewall* certo para o sistema operacional que está sendo utilizado ou que se pretende utilizar. Sempre levando em conta também, o que se pretende proteger e que tipo de organização está administrando.

5.1 A segurança nas Organizações e a Escolha do Firewall *Ideal*

Além da escolha do *firewall* para o sistema operacional certo, o administrador deve se ater também ao tipo de informação que se pretende proteger, sendo o tipo de instituição em que será implantado o *firewall* de relevância primeira, pois, o tipo de rede e sua organização irão determinar as necessidades relativas à segurança. A quantidade de trabalho que se deve proteger e que se tem para proteger em uma rede é marcada pelo tipo de informação que trafega e que esta contida na rede.

Para ter uma visão mais significativa deste assunto tão singular, apresenta-se abaixo alguns tipos de organizações e as sugestões de segurança, levando-se em conta, sempre o tipo de organização e não os

vários níveis de "paranóias" existentes no campo da segurança computacional.

5.1.1 Pequenas Empresas e Escritórios Domésticos

Geralmente aplica-se a este caso redes mais simples, com três ou quatros computadores ligados ponto a ponto, formando assim, uma pequena rede local. Neste caso não é viável, logicamente e financeiramente, ter um computador dedicado para ser um *firewall* da rede. O que se faz geralmente em casos como esses, é utilizar a segurança do provedor para proteger a rede. Esta técnica é pouco eficaz e nunca recomendada.

Como não é economicamente viável para uma organização deste porte adquirir um *software* de *firewall* proprietário[10], a solução seria então, configurar os recursos do sistema operacional para proteger a rede.

Uma solução ótima para esta organização, é a implantação de sistemas operacionais, *software* e *firewalls open source* [11]. È importante frisar que esta recomendação, aplica-se também às empresas de grande porte e outras instituições, inclusive instituições públicas, por ser totalmente gratuitas, legal e extremamente seguro.

5.1.2 Empresas de Médio e Grande Porte

Uma rede com aproximadamente *50* a *100* computadores conectados é um alvo mais tentador para um invasor, especialmente se a rede dispuser de equipamentos com relativo custo e enlaces de *VPNs* com outras redes

---

[10] *Software Proprietário: Software comercial, registrado e patenteado pelo fabricante. Para a aquisição de ferramentas ditas como proprietária deve-se obter legalmente a licença com o proprietário, que de modo geral é cedida através da compra do produto.*

[10] *Open source –Termo atribuído a sistemas operacionais e software gratuitos, e de código aberto licenciados pelo projeto Open/GL.*

grandes de computadores. Esse tipo de rede é mais usual em empresas de médio e grande porte e consequentemente um alvo desejado por *invasores*, pois, devido à sua complexidade é mais fácil encontrar vulnerabilidades. Nas instituições de médio e grande porte, um ou dois computadores não irão pesar no orçamento da empresa, assim recomenda-se o uso de um *firewall* dedicado ou até mesmo *firewalls multihomed*[12].

5.1.3 Órgãos Governamentais, Escolas, Universidades, Organizações Militares e Hospitais.

É comum depararmos com notícias pelo mundo afora, de invasões ou tentativas de invasões de empresas públicas, organizações militares e governamentais. Estas entidades são muito assediadas por praticantes de *"hackerismos".* Na maioria dos casos a invasão ou tentativa de invasão a estas instituições se dá meramente devido ao *status* atribuído ao invasor dentro da comunidade *hacker*. Outro fato que contribui para este assédio deve-se geralmente por tratarem-se de redes de grande porte e conseqüentemente sujeitas a falhas e vulnerabilidades.

Grande parte dos países europeus utiliza *software* livre em suas redes, no caso de paises pobres e emergentes como o Brasil, por ironia, a maioria das redes opera com *software* proprietários. Embora este conceito esteja mudando, mesmo que em passos lentos, ainda existe uma grande restrição por parte dos usuários em utilizar sistemas como o *Linux*. Simplesmente por não conhecerem e não se dar ao trabalho de procurar conhecer esses sistemas.

[12] *Firewall Multihomed – Um firewall que possui múltiplas bases.*

Em casos como este, é fundamental que os administradores e projetistas de redes adotem políticas corretas e sensatas de migração para que os *software* livres não caiam em descrédito perante aos usuários e não seja "queimado", como acontece na maioria dos casos.

5.1.4 Instituições Financeiras e Empresas de *e-comerce*

As Instituições financeiras e empresas de *e-comerce*[13] são também bastantes visadas por invasores, à finalidade geral neste caso, é o desvio de dinheiro e a interceptação de senhas e números de cartões de crédito. Os clientes destas organizações, geralmente, são os que sofrem conseqüências mais drásticas no caso de uma interceptação de uma movimentação financeira ou de uma compra de um produto.

A maioria das empresas que operam neste ramo de atividade possui políticas de segurança bem definidas e recurso de *software* e de máquina para a proteção de seus sistemas e seus clientes, embora nem sempre eficientes.

5.2 Selecionando um Sistema Operacional

Um dos propósitos deste trabalho é apresentar o uso dos *firewalls* nos diversos tipos de sistemas operacionais.

Existe uma grande disponibilidade de sistemas operacionais no mercado, desde gratuitos (*free*), de código aberto (*GNU/GPL*) e proprietários como o caso do *Windows*[14]. Apresenta-se aqui, um paralelo sobre o uso de *firewalls* em alguns destes sistemas operacionais.

---

[13] *E-comerce: Atividade comercial exercida através da internet – comércio eletrônico.*

[14] A marca *Window*s é de propriedade da *Microsoft Corporation.*

A escolha de um sistema operacional e destacada por WELCH-ALBERNATHY (2003) como um processo determinante no aspecto de segurança, devendo o administrador de sistemas uma atenção especial em três pontos principais, os quais são apresentados abaixo.

- Selecionar o Sistema operacional que seja melhor para o seu ambiente;
- Fortalecer corretamente o sistema operacional;
- Instalar o *software* de *firewall* e todos os *packs* de serviços a ele relacionado.

5.3 O Sistema Operacional Adequado

O primeiro passo na montagem de seu *firewall* é a seleção do sistema operacional em que a aplicação irá rodar. Existem várias opções, dentre as quais, destaca-se:

- *Linux*
- *Mac*
- *Solaris*
- *Unix*
- *Windows 2000* e *2003 Server*

Todo sistema operacional possui vantagens e desvantagens, não é objeto de estudo deste trabalho uma avaliação de qual é o melhor ou o pior sistema operacional, mesmo porque, esta é uma questão polêmica e que reserva muitas contradições; mas será apresentado algumas características, vantagens e desvantagens dos sistemas operacionais. Dentre os destacados anteriormente, vamos citar o *Linux, Solaris* e o *Windows.*

Por uma questão de segurança, recomenda-se que a escolha do sistema operacional esteja relacionado ao tipo de aplicação e finalidade, já que nenhum sistema operacional é ótimo para todos os ambientes. O critério isolado e mais importante para a escolha do sistema operacional está centrado no conjunto de habilidades de seus administradores.

Uma das questões apresentadas por WELCH-ALBERNATHYA (2003) é que qualquer que seja o sistema operacional selecionado, garanta que o pessoal de segurança tenha conhecimento suficiente sobre tal sistema, pois mesmo que você tenha o melhor sistema operacional do mundo, certamente terá problemas para montar e mantê-lo se não tiver pessoal habilitado.

5.3.1 O Sistema Operacional Linux

O *Linux* é um sistema operacional baseado no *Unix*, disponível sobre licença pública *GNU*[15]. O que significa que se pode ter acesso gratuito ao código fonte e permitindo inclusive, realizar quaisquer alterações no original adquirido.

As vantagens principais do *Linux* são:

- **Estabilidade**: O sistema operacional *Linux*, embora exista uma variação entre a maioria das distribuições, apresenta uma estabilidade muito satisfatória.
- **Recursos do *UNIX***: O *Linux* compartilha as mesmas vantagens da maioria das versões do *UNIX*.
- **Gerenciamento**: Assim como o *UNIX*, o *Linux* possui um poderoso gerenciamento e interface de linhas de comandos.

[15] *GNU - General Public License – Softwares licenciados pelo projeto GNU são de distribuição gratuita e livre.*

- **Fonte Aberta**: Além de adquirir o sistema operacional gratuitamente, tem-se também o código fonte. O que permite mudar inclusive o *kernel* de acordo com os requisitos de segurança.

O *Linux* possui também algumas desvantagens, de onde pode se destacar:

- **Sistema operacional novo**: *O Linux* é um sistema relativamente novo para o mundo comercial, dependendo do tipo de aplicação que se deseja configurar, pode ainda não estar disponível para o *Linux*.
- **Pessoal Habilitado**: Ainda existe uma carência com relação à pessoal habilitado para o mercado de trabalho.
- **Variação entre distribuições**: Apesar de obedecer a uma padronização, existem muitas variações entre as diversas distribuições disponíveis.

5.3.2 O Sistema Operacional *Solaris (Sparc e x86)*

O *Solaris (Sparc* e *X86)* é um sistema operacional da *Sun Microsystems*. Trata-se de um sistema proprietário e de uso relativamente fácil. O *Solaris* é pouco conhecido no Brasil, tendo seu uso mais difundido nos Estados Unidos e Europa.

O *Solaris* possui algumas vantagens, dentre as quais especificamos abaixo:

- **Bastante Utilizado**: O *Solaris* é bastante utilizado e é uma solução popular para a instalação de *firewalls*. Além disso, existe uma rica documentação disponível sobre o *Solaris* na *Web*, inclusive sobre instalação, configuração e manutenção de *firewalls*.
- **Sistema Operacional da primeira camada:** O *Solaris* é um sistema operacional da primeira camada, o que corresponde que novos

recursos de *firewall* sempre aparecerão primeiro em *SO´s* com esta característica.

- **Acesso pela linha de comandos**: O Solaris possui esta característica, pois é um SO baseado na arquitetura *UNIX*, assim como o Linux. Isso é um facilitador para o diagnóstico do sistema operacional e de aplicações de firewalls, especialmente pelo acesso remoto.
- **Suporte de Hardware de Alto nível**: O S*olaris* costuma oferecer suporte para *hardware* de alto nível, incluindo muita memória, e grandes unidades de discos, significando que é uma plataforma altamente escalável;
- **Software de terceiros**: Embora esta não seja uma vantagem que se compara com outros *software* proprietários, como o *Windows*, existe muitas aplicações que pode-se usar em conjunto, principalmente tratando-se de softwares de *firewalls*, como o *Chekpoint FireWall-1*[16].

Algumas desvantagens do *Solaris* podem ser apresentadas, como:

- **Treinamento**: O *Solaris*, assim como a maioria dos *SO´s* baseados em *UNIX*, exige mais habilidade e treinamento do administrador;
- **Pessoal com experiência**: Pouca gente no Brasil conhece o *Solaris*, e é muito difícil encontrar um administrador experiente. Quando isso acontece o custo deste profissional é muito elevado;

5.3.3 O Sistema Operacional Windows (NT, 2000 e 2003 *Server*)

As plataformas *Windows*, desenvolvidos pela *Microsoft*, são totalmente *software* proprietários e geralmente os *software* desenvolvido para este *SO* deve ter a assinatura ou autenticação da *Microsoft* como um produto

[16] *Checkpoint FireWall-1 é marca registrada da Checkpoint Corporation Inc.*

desenvolvido para as plataformas *Windows*. Principalmente nas versões do *Windows 2003 Server* e do *Windows 2000 Server* 2ª edição. Isto não implica que se configure outro aplicativo de *firewall* que não seja autenticado pela *Microsoft*, principalmente os gratuitos, mas, além da configuração ser muito complicada para estes aplicativos é comum que falhas na configuração seja deixadas, o que deixará o sistema muito vulnerável a ataques.

Algumas vantagens da plataforma *Windows* são apresentas abaixo:

- **Facilidade de Uso**: As plataformas *Windows* possuem uma interface gráfica *GUI*, o que facilita bastante a instalação, configuração e manutenção do sistema operacional e de outros aplicativos, como um *firewall*, por exemplo.
- **Mais Usado**: O *Windows* é o sistema operacional mais usado no mundo, existe muito profissional com habilidades para gerenciamento e a maioria dos aplicativos comerciais são desenvolvidos para plataforma *Windows*.
- **Sistema Operacional de Primeira Camada**: O *Windows* é um sistema operacional de primeira camada, o que corresponde que novos recursos de firewall sempre aparecerão primeiro em *SO´s* com está característica.
- **Softwares de terceiros**: Existe uma grande quantidade de *software* disponibilizado para a plataforma *Windows*. Inclusive *software* de *firewall*. As empresas de softwares comerciais, na sua maioria, desenvolvem aplicativos voltados para a plataforma *Windows*, isso se dá devido a sua popularidade.

O *Windows* possui algumas desvantagens, dentre as quais pode-se destacar:

- **Administração Remota**: Em comparação com os sistemas *UNIX*, o *Windows* é mais difícil de administrar remotamente, já que a maioria das tarefas só pode ser realizada através da interface gráfica. O *Windows 2003 Server*, já apresenta um console para administração remota, mas não tão simples como *o Linux*, por exemplo.
- **Acesso pela linha de comandos:** As plataformas *Windows* não apresentam uma interface poderosa para linhas de comandos, o que torna difícil o diagnóstico, tanto para o *SO* quanto para administração de recursos de *software*, Principalmente para manutenção e administração de softwares de segurança, como os *firewalls*.

5.4 Os Software de Firewalls

Existe uma infinidade de *software* de *firewalls* no mercado, tanto proprietários, quanto gratuitos e de código aberto. Este texto irá apresentar as características e funcionamento do *Check Point FireWall-1*, bem como sua implantação e instalação, como uma solução de *software* de *firewall* proprietário.

A escolha do *Check Point FireWall-1*, deu-se pela sua popularidade, sendo um dos aplicativos de *software* mais usados no mundo e também por ser um aplicativo com versões disponíveis pelos mais diversos sistemas operacionais, inclusive os sistemas baseados em *UNIX,* como o *Linux*.

Como opção de softwares de firewalls gratuitos e de código aberto, será apresentado o *Ipfwadm*, o *Ipchains* e o *Iptables*, sendo que para este último, dedica-se o capítulo 6 inteiramente, apresentando sucintamente a configuração, implantação e a manutenção deste *software* de *firewall*, que é o mais usual e popular na comunidade *Linux*.

5.5 Apresentando O Check Point FireWall-1

O *Check Point FireWall-1* é uma poderosa ferramenta de *firewall* e muito usada e conhecida no confuso meio da segurança computacional. Trata-se de um *software proprietário* e está disponível para os mais diversos sistemas operacionais.

A versão mais recentes para *Linu*x é o *FireWall-1 4.1*, a qual foi adicionada diversos *services pack* para garantir melhor a qualidade da segurança para o sistema operacional. As instruções de instalação e manual vem inseridas no *CD* de instalação do *FireWall-1*, especificando as etapas necessárias para carregar o *software* de *firewal*l. Aqui será apresentado de maneira genérica os passos para instalação e configuração do *Check Point FireWall-*1. Para uma abordagem mas completa recomenda-se a leitura de (WELCH-ALBERNATHY[5]) ou do manual do produto quem vem inserido junto ao *CD* de instalação, conforme comentado anteriormente.

Alguns *software* de *firewalls* proprietário, como o caso do *Chekpoint FireWall-1*, possui uma grande desvantagem com relação a outros *firewalls* proprietários, um custo extra para configuração de alguns pacotes específicos do *firewall*. A razão deste custo extra não pode ser explicada, pois os representantes do *Check Point FireWall-1* não responderam os *vários e-mails* a eles enviados.

5.6 Instalando o Check Point FireWall-1 no Linux

A instalação do *Check Point FireWall-1* em sistemas operacionais baseados em *UNIX*, obedece um certo padrão, que é também aplicado à plataforma *Linux*.

Após a instalação e configuração correta do sistema operacional, obedecendo atentamente as políticas de segurança estabelecidas, deve-se adotar o seguinte procedimento para instalação e implantação do *Check Point FireWall-1.*

Quando o *software* for carregado, a variável *FWDIR* deve ser definida como local onde o *software* será instalado. Essa informação precisa ser incluída no *.cshrc* ou no *.profile* do seu administrador. Deve-se incluir também o cominho *$FWDIR/bin* na sua variável *patch*. Os passos acima são muito simples de serem executados e geralmente nos ambientes *Linux* isto é feito automaticamente pelo próprio sistema operacional.

Em seguida deve dar início ao processo de instalação, lembrando que antes de começar a instalação, verifique se o *hostname* está definido no arquivo /etc/hosts. Visto isso é só executar o comando *cpconfig* para que a instalação seja em fim iniciada. Durante o processo de instalação algumas perguntas comuns serão feitas e uma das opções de resposta deve ser escolhida para que o processo continue.

A seguir apresenta-se as Figuras *5.6.4* e *5.6.5* para melhor entendimento do processo de instalação.

 # cpconfig
Welcome to Check Point Configuration Program
==========
Checking avaliable options. Please Wait…….
Chosing Installation
_____
(1) VPN-1 & FireWall-1 Stand Alone Installation
(2) VPN-1 & FireWall-1 Distributed Installation
Option (1) will install VPN-1 & FireWall-1
Internet GateWay (Management Server and Enforcement Module)
On a single machine
Option (2) will allow install specific
components of the VPN-1 & FireWall-1 Enterprise products
on different machines.
Enter you selection (1-2/a) [2]: **1**
The next Option allows you to choose which module you would like install.

Figura 5.6.4 Instalação do *FireWall-1* – Passo 1

Como pode ser observado os passos de implantação do *FireWall-1* é bastante simples e bastante semelhante com a instalação de vários aplicativos que estamos acostumado a instalar. A próxima ilustração reforça esta afirmação.

Installing VPN-1 & FireWall-1 Stand Alone Installation

Which module you like install?

_____

(1) VPN-1 & FireWall-1 – Limited Hosts (25, 50, 100 or 250)
(2) VPN-1 & FireWall-1 – Unlimited hosts
(2) VPN-1 & FireWall-1 - SecurServer
Option (2) will allow install specific

Enter you selection (1-2/a) [2]: **2**

Figura 5.6.5 Instalação do *Firewall-1* – Passo 2

Após ter instalado o *Check Point FireWall-1*, os passos seguintes é a configuração do firewall no sistema. As opções mais comuns ou primárias indicadas em WELCH-ALBERNATHY (2003) são:

- Configurações da licença, que é obrigatória;
- Configuração do administrador ou administradores, se for o caso;
- Configuração dos clientes;
- Configuração da interface gráfica, não necessária, já que o sistema pode ser gerenciado através de linhas de código;
- Configuração do servidor de *SMTP*;
- Configuração de extensões *SNMP*;
- Configuração dos grupos;
- Configuração do *IP* f*orwarding*;
- Configuração dos Filtros padrões.

O processo de configuração do *FireWall-1* é bastante amigável obedecendo a padronização dos sistemas operacionais baseados em *UNIX*. Estas configurações não serão mostradas aqui, devido a sua extensa documentação. Sugere-se o uma leitura do próprio manual do aplicativo ou a obra do autor citado no decorrer deste capítulo.

6 *Firewalls* e *Linux*

Após ter abordado os conceitos de *firewalls*, política de segurança, os vários tipos de firewall para os diversos tipos de sistemas operacionais, além de muitos outros tópicos relevantes no estudo de *firewalls*, dedica-se este capítulo somente aos *firewalls* para *Linux,* destacando-se o *Iptables*.

Este capítulo apresenta ainda a diferença entre o *Iptables* e o *Ipchains*, bem como sua implementação e a configuração de seus componentes. Algumas das características dos antecessores do *firewall Iptables*, assim como o próprio *Iptables* e as partes que o compõe, como: regras, tabela e sua configuração serão apresentados com mais requinte de detalhes.

6.1 Ipfwadm e *Ipchains*

Dentre os antecessores do *Iptables*, podemos destacar o *Ipfwadm* e *Ipchains*, que ainda pode ser encontrando rodando em muitos sistemas de computadores (RUSSEL, 2002).

6.1.1 Característica do *Ipfwadm*

O utilitário *ipfwadm* foi um dos candidatos para ser uma substituição do antigo utilitário *ipfw*, como encontradas em versões mais velhas do pacote de ferramenta de rede. I*pfwadm* foi feito para ser o mais completo e de mais fácil

uso do que o *ipfw*. Um de seus pré-requisito pra funcionamento é o *Kernel 2.0.x* e *2.1.x..*

As principais características oferecidas pelo *ipfwadm* é apontada por RUSSEL (2003) e destacadas abaixo:

- Mudança das políticas padrões para todas as categorias de *Firewall*;
- Adicionam automaticamente as necessárias regras extras quando o nome do *host* tem mais de um endereço *IP*;
- Suporta especificar o endereço de interface para as regras;
- Suporta especificar o nome da interface para as regras;
- Lista e reajusta automaticamente os contadores de *packet/byte* para configurar um esquema de contabilidade confiável;
- Lista as regras existentes em vários formatos;
- Suporte para funções opcionais (regras de *bidirectional*, *TCP ACK*, e *TCP SYN* combinando);
- Suporte para redirecionamento de pacote, usado para *proxy* transparente;
- Suporte para mascaramento.

6.1.2 Características do *Ipchains*

O *Linux ipchains* é uma reformulação do código do *Linux* de *IPv4 firewalling*, e uma reformulação do *ipfwadm*, ao qual era uma reformulação do *ipfw* do *BSD* (RUSSEL, 2002).

Atualmente o código está no *kerne*l *2.1.102*. Para a série *2.0* do *kerne*l, precisa carregar um *patch* do *kernel* na página da *web*. Algumas destas

mudanças são resultados da evolução do *Kernel,* com alguns resultados do *ipchains* diferentes do *ipfwadm*.

Muitos argumentos foram remapeados: os maiúsculos agora indicam um comando, e letra minúscula agora indica uma opção. Abaixo destaca-se outras características destas mudanças.

- As regras arbitrárias são suportadas.
- Múltiplas portas de fonte e destino não são mais suportados.
- As interfaces só podem ser especificadas por nome e não por endereço.
- Os fragmentos são examinados.
- As opções inversas agora são suportadas.
- Códigos *ICMP* agora são suportados.
- As interfaces de *Wildcard* agora são suportadas.
- As manipulações de TOS são agora checadas.
- Os contadores são agora de *64 bits* em máquinas de *32 bits*.

Outra mudança foi o velho comportamento do *SYN* e *ACK* combinado, que estava previamente ignorado para pacotes não *TCP*, mudou a opção de *SYN* que não era válida para regras de *TCP* não específicas.

6.4 O Iptables

O *Iptables* é o principal *firewall* para o *Linux*, e de longe o mais utilizado pelos administradores de sistemas. Isto se dá graças a sua portabilidade e confiabilidade, além da facilidade de manutenção e manipulação de seu banco de regra. O Iptables é o sucessor dos firewalls mas antigos como o caso do *Ipfwadm* e o *Ipchains*. As sessões a seguir irão tratar as questões

relacionadas ao *Iptables*, abordando as principais características, funcionalidades e configuração deste *software* de *firewall*.

6.5 Diferença entre Iptables *E* Ipchains

Primeiramente, os nomes das *chains* padrão passaram a ser escritos com maiúsculas em vez de minúsculas.

A *flag '-i '* agora significa interface de entrada e funciona apenas nas *chains INPUT* e *FORWARD*. Outra mudança significativa aconteceu com o alvo *DENY*, que agora é *DROP*, e com a *MASQ*, agora *MASQUERADE*, e que utiliza sintaxe diferente. Listar as *chains* mostra automaticamente os contadores e as portas *TCP* e *UDP* agora precisam ser escritas com as opções --*source-port* ou –*sport.*

6.6 Como Criar as Regras

Cada regra é uma linha que o *kernel* olha para descobrir o que fazer com um pacote. Se todos os critérios são encontrados, é realizado o objetivo ou é saltado para uma próxima instrução. Normalmente escreve-se uma a regra da seguinte forma:

iptables [-t tabela] comando [combinação] [trajeto/pulo]

Não existe nada que diz que a instrução do objetivo deve ser na última linha, porém, isso é feito para conseguir uma melhor leitura. Também esse modo de escrita de regras é usado desde que ele é o modo mais habitual de

escrever. Se achar necessário usar outra tabela ao invés da tabela normal e só inserir a especificação da tabela onde for especificado. Porém, não é necessário especificar isto explicitamente todo o tempo já que *iptables* por padrão usa a tabela *filter* para implementar seus comandos. Não é exigido pôr a especificação da tabela nesta localização, pode-se colocar em qualquer lugar da regra, no entanto, é padrão pôr a especificação da tabela no princípio da linha de comando.

Uma coisa para se pensar é que, o comando deve sempre ser o primeiro, ou diretamente após a especificação da tabela. A regra diz ao *iptables* qual é o comando a ser utilizado. Normalmente usa-se a primeira variável para dizer ao programa o que fazer, por exemplo, inserir uma regra ou para somar uma regra ao final das regras, ou para apagar uma regra.

6.7 *Matches*

As *matches* são regras de verificação, que são passadas para o *kernel* dizendo se o pacote é pertencente ou não a regra. Poderá ser especificado para qual endereço *IP*, o pacote, deve vir, ou que interface de rede o pacote deve vir e assim por diante.

Finalmente se tem o objetivo do pacote. Se a *match* for compatível com o pacote então o *kernel* executará a ação no pacote. Pode-se dizer então que, para o kernel enviar o pacote para outra rede ele cria uma *ourself*, que deve ser parte desta tabela. Pode-se dizer também ao *kernel* para "*dropar*" este pacote e não fazer nenhum processo adicional, ou dizer para enviar uma resposta especificada.

As *matches* são classificadas em cinco partes diferentes conforme especificação abaixo:

- ***Matches* genéricas**: como o nome já diz, são genéricas, o que significa que podem ser usadas em todas as regras;
- ***Matches TCP***: podem ser aplicadas somente em pacotes TCP;
- ***Matches UDP***: podem ser aplicadas somente em pacotes *UDP*;
- ***Matches ICMP***: podem ser usadas somente em pacotes *ICMP;*
- ***Matches* especiais**: são as *matches* definidas como: de estado, de propriedade, de limite e assim por diante. É comum encontrar subdivisões ou subcategorias destas *matches*, embora não constituírem matches diferentes.

6.8 Tabelas

A opção -t especifica qual a tabela deverá ser usada. Por padrão, a tabela *filter* é usada. Porém pode-se especificar uma das tabelas a ser usada utilizando a *opção -t.*

Os 3 tipos de tabelas são: a tabela *filters*, a tabela *nat* e a tabela *mangle* **(**ANDREASSON[9])*.*

- **A tabela *NAT***: é usada principalmente para a tradução de endereço de rede. Os pacotes têm um fluxo único que atravessam esta tabela, um por vez. O primeiro pacote de um fluxo é permitido, o resto dos pacotes do mesmo fluxo são automaticamente *NATeados* ou *mascarad*o. Mas isso, no caso deles serem supostos para ter aquelas condições empreendidas a eles. O resto dos fluxos de pacotes em outras palavras não irá atravessar esta tabela de novo, mas ao invés, eles

terão automaticamente as mesmas ações levadas para eles como o primeiro pacote do fluxo. Esta é uma razão para que não tenha qualquer tipo de filtro nesta tabela. As regras de *PREROUTING* são usadas para alterar os pacotes assim que eles entram no *firewall*. As regras de *OUTPUT* são usadas para alterar localmente os pacotes no *firewall*, antes deles chegarem ao roteamento de decisão. Finalmente nós temos a regra de *POSTROUTING* que é usado para alterar pacotes quando eles estão pra sair do *firewall*.

- **A tabela *FILTER***: deve ser usada geralmente para filtrar pacotes. Por exemplo, pode ser aplicado aos pacotes as ações de *"dropar"*, *"logar"*, aceitar ou rejeitar pacotes sem problemas como nas outras tabelas. Existem três regras que se construiu esta tabela. A primeiro é chamado de *FORWARD* é usado em todo pacote não local que não são destinados para nosso *host* local, o *firewall* em outras palavras. *INPUT* é usado em todos os pacotes que são destinados para nosso *host* local e o *OUTPUT* que finalmente é usado para todas as saídas localmente de pacotes.

- **A tabela *MANGLE***: é, principalmente, usada para destrinçar pacotes. Com esta tabela existe a possibilidade de mudar diferentes pacotes como seus cabeçalhos, entre outras coisas. Os exemplos disto seriam mudar o *TTL*, *TOS* ou *mark*. Note que o *mark* não é realmente uma mudança para o pacote, mas uma característica para o pacote que é aparecida pelo *kernelspace* e que outras regras ou programas podem ser usadas mais adiante no *firewall* para filtrar ou avançar o roteamento. A tabela consiste em duas, regras a *PREROUTING* e a regra *OUTPUT*. *PREROUTING* é usado para alterar pacotes da mesma maneira que eles entram no *firewall* e antes deles serem roteados. O *OUTPUT* é usado pra mudar e alterar no local os pacotes

antes deles serem roteados. Nota-se que o mangle não pode ser usado para qualquer tipo de Tradução de endereço de rede ou mascaramento, a tabela *NAT* é favorecida nestes tipos de operações.

6.9 Comandos

Os comandos são usados para dizer o que *iptables* deve fazer com o resto da linha de comando a ele enviado. Normalmente se insere ou apaga algo em alguma tabela.

Apresenta-se a seguir na Figura 6.9.6 alguns dos principais tipos de comandos, com o exemplo de aplicação e a descrição dos comandos usados pelo *iptabes*. Não serão apresentados todos os comandos devido a grande quantidade de comandos existentes e sua extensa descrição. Para maiores informações sobre os comando do *iptables* recomenda-se a leitura ANDREASSON (2002) e MOTA FILHO (2001), onde estes são tratados com clareza e riqueza de detalhes.

Comando -A, --*append*
Exemplo iptables -A INPUT ...
Este comando anexa a regra ao final das regras.
Comando -D, --*delete*
Exemplo iptables -D INPUT --*dport* 80 -j DROP, iptables -D INPUT 1
Este comando apaga uma regra da lista.
Comando -R, --replace
Exemplo iptables -R INPUT 1 -s 192.168.0.1 -j DROP
Este comando substitui a regra velha na linha especificada.
Comando -I, --insert
Exemplo iptables -I INPUT 1 --dport 80 -j ACCEPT
Insere uma regra em algum lugar da lista. A regra é inserida no número real que se é passado.
Comando -L, --list
Exemplo iptables -L INPUT
Este comando listas todas as entradas na lista especificada.

Figura 6.9.6 Comandos do *Iptables*

6.10 Implementando Regras com o *Iptables*

Para mostrar o funcionamento do *iptables* na prática, serão apresentados a seguir alguns testes e seus resultados, feitos com o *Iptables*, que foram executados no servidor *Linux* do Departamento do *3º Grau Indígena* da *UNEMAT*[17].

6.11 Regras de Filtro – *INPUT* e *OUTPUT*

Apresenta-se aqui, alguns conceitos básicos de implementação de filtragem de pacote, onde o objetivo será o de permitir apenas a passagem de uma determinada faixa de *IP* e *"dropando"* todas às outras requisições. A Figura 6.11.8 apresentada um script que representa a filtragem de pacotes.

```
iptables -F
## Esta opção zera todas as regras da tabela filter
iptables -P INPUT DROP
iptables -P OUTPUT DROP
iptables -P FORWARD DROP
## Nesta parte foi setada todas as política da tabela filter como DROP
iptables -A INPUT -s 192.168.200.3 -j LOG --log-prefix "INPUT ACEITO:"
iptables -A OUTPUT -d 192.168.200.3 -j LOG --log-prefix "OUTPUT ACEITO:"
## Nesta parte todos os pacotes vindo de 192.168.200.0 e que forem para o
## mesmo serão logado com o prefixo "INPUT ACEITO:"
iptables -A INPUT -s 192.168.200.3 -j ACCEPT
iptables -A OUTPUT -d 192.168.200.3 -j ACCEPT
##Esta regra diz que tudo que vier de 192.168.200.0 de mascara
##255.255.255.0 e que também forem enviados para o mesmo serão aceitos
iptables -A INPUT -j LOG --log-prefix "INPUT DROPADO:"
iptables -A OUTPUT -j LOG --log-prefix "OUTPUT DROPADO:"
##Esta seção foi especifica que qualquer coisa que não vier nessa regras
##também serão logados.
```

Figura 6.11.7 Regras de filtro *INPUT – OUTPUT*

[17] UNEMAT: Universidade do Estado de Mato Grosso – Campus Barra do Bugres

6.12 Regras de Filtro – *FORWARD*

Este experimento foi tratado com a regra *FORWARD* da tabela filter, onde o principal objetivo e bloquear todos os *INPUTS*, *OUTPUTS* e permitindo o tráfego de somente algumas faixas de *IP. A Figura* a Figura 6.12.9 ilustra os passos deste procedimento.

```
iptables -F
## Esta opção zera todas as regras da tabela filter
iptables -P INPUT DROP
iptables -P OUTPUT DROP
iptables -P FORWARD DROP
## Aqui todas as políticas da tabela filter foram setadas com o DROP
iptables -A FORWARD -s 10.0.0.0/8 -d 192.168.200.0/24 -j LOG --log-
prefix "s:10.0.0.0 d:192.168.200.0 - "
iptables -A FORWARD -d 10.0.0.0/8 -s 192.168.200.0/24 -j LOG --log-
prefix "s:192.168.200.0 d:10.0.0.0 - "
## Esta sessão cria o log para registrar a passagem entra as duas sub-
##redes
iptables -A FORWARD -s 10.0.0.0/8 -d 192.168.200.0/24 -j ACCEPT
iptables -A FORWARD -d 10.0.0.0/8 -s 192.168.200.0/24 -j ACCEPT
## Permissão para comunicação entre as duas sub-redes
iptables -A INPUT -j LOG --log-prefix "INPUT DROPADO - "
iptables -A OUTPUT -j LOG --log-prefix "OUTPUT DROPADO - "
## Neste trecho todo o trafego de INPUT e OUTPUT será logado
```

Figura 6.12.8 Regra de Filtro *FORWARD*

6.13 Regra de *NAT – MASQUERADE*

Neste experimento foi abordado o *target MASQUERADE* na tabela *NAT*, onde o objetivo e que todo tráfego que passa por ele em direção a

internet seja mascarado com seu *IP*. A Figura 6.13.10, ilustra este processo.

```
iptables -F
iptables -F -t nat
## Nesta opção zerei todas as regras da tabela filter e da nat
iptables -P INPUT ACCEPT
iptables -P OUTPUT ACCEPT
iptables -P FORWARD ACCEPT
## As políticas da tabela filter foram setadas com ACCEPT
iptables -A INPUT -j LOG --log-prefix "INPUT - NAT:"
iptables -A OUTPUT -j LOG --log-prefix "OUTPUT - NAT:"
iptables -t nat -A PREROUTING -j LOG --log-prefix "NAT -
PREROUTING:"
iptables -t nat -A POSTROUTING -d 0/0 -j LOG --log-prefix
"MASQUERADE-“
iptables -t nat -A POSTROUTING -d 0/0 -j MASQUERADE
iptables -t nat -A OUTPUT -j LOG --log-prefix "NAT - OUTPUT:"
## Criação de regras de log e de  uma regra de mascaramento.
```

Figura 6.13.9 Regras de *NAT – Mascarede*

6.14 Regra de *NAT – REDIRECT*

Neste experimento foi abordado o *targe*t *REDIRECT* na tabela *NAT*, onde o objetivo é de que todo o tráfego que passa por uma determinada porta redirecionará para outra porta do computador *firewall*, conforme apresentado a seguir na Figura 6.14.11

```
iptables -F
iptables -F -t nat
## Esta opção zera todas as regras da tabela filter e da nat.
iptables -P INPUT ACCEPT
iptables -P OUTPUT ACCEPT
iptables -P FORWARD ACCEPT
## Políticas setadas tabela filter com ACCEPT
iptables -A INPUT -j LOG --log-prefix "INPUT - NAT:"
iptables -A OUTPUT -j LOG --log-prefix "OUTPUT - NAT:"
iptables -t nat -A PREROUTING -s 10.0.0.0/8 -p tcp --dport 333 -j LOG --
log-prefix "NAT -
PREROUTING:"
```

```
iptables -t nat -A PREROUTING -s 10.0.0.0/8 -p tcp --dport 333 -j
REDIRECT --to-port 80
iptables -t nat -A POSTROUTING -j LOG --log-prefix "NAT -
POSTROUTING:"
iptables -t nat -A OUTPUT -j LOG --log-prefix "NAT - OUTPUT:"
```

Figura 6.14.11 Regras de *Nat - Redirect*

Apresentou-se aqui, os principais componentes do *iptables*, suas regras e como efetuar sua configuração. Maiores informações poderão ser adquiridas com as literaturas referenciadas neste trabalho e dispostas no item *8 Referências Bibliográficas*.

7 Considerações Finais

A eficiência e eficácia das redes e dos sistemas de computadores possuem uma relação intrínseca com as questões relacionadas à segurança dos computadores. Essa é uma questão que não dá para negar, e que todo administrador sensato deve ter em mente.

A confiabilidade do sistema e a “segurança” dependem única e exclusivamente dos projetistas e administradores de sistemas, os quais devem adotar políticas de seguranças bem definidas e documentadas para o sistema que está sendo administrado. Aliado a isto, deve-se ter um *software* de *firewall* robusto e bem configurado com manutenção periódica.

Neste trabalho apresentou-se entre as muitas opções de firewall existente, o *Iptables* e o *Check Point FireWall-1* como soluções ótimas para a proteção de uma rede e um sistema de computador.

Com a máxima de que “não há sistema seguro”, concluí-se, que somente o uso de *firewall* não provê toda segurança necessária para uma rede ou um sistema. Devemos para tanto, adotar alguns mecanismos e procedimentos necessários para que todo o sistema se torne o mais seguro possível, como por exemplo montar uma estrutura de rede bem planejada e ter uma boa coleção de *software* que interaja diretamente ou indiretamente com o *firewall*, assim proporcionando uma proteção do mais alto nível de eficiência, além de outros mecanismos de segurança que deve ser observado pelo administrador de sistemas e de rede.

8 Referências Bibliográficas

[1] STREBE, Mathew; PERKINS, Charles – Firewalls; Ed. Makron Books; São Paulo – SP, 2000.

[2] HATCH, Brian; LEE, James; KURTZ, George – Segurança Contra Hackers Linux; 2ª Edição; Ed. Mac Graw Hill; São Paulo – SP, 2003.

[3] WELCH-ABERNATHY, Dameon D; Check Point Firewall-1 – Essencial; Ed. Campus; Rio de Janeiro – RJ, 2003.

[4] MENDES, Wayne Rocha – Linux e os Hackers – Proteja Seu Sistema; Ed. Ciência Moderna; Rio de Janeiro – RJ, 2000.

[5] UCHÔA, Joaquim Quinteiro – Segurança em Redes e Criptografia; Ed. UFLA; Lavras – MG, 2003.

[6] MELO Emerson Ribeiro - Redes de Confiança; 2002; (Disponível por www em: www.das.ufsc.br/seguranca/artigos/dissertacaomestrado.pdf Acessado em: 10 de Junho de 2004)

[7] RUSSELL, Rusty. Linux IPCHAINS-HOWTO; 2002. (Disponível por www em: <http://www.netfilter.org/ipchains/HOWTO.html> Acessado em: 25 de Maio de 2004).

[8] RUSSELL, Rusty. Linux 2.4 Packet Filtering HOWTO; 2002. (Disponível por www em: <http://www.netfilter.org/documenta.html> Acessado em: 20, Maio, 2004).

[9] ANDREASSON, Oskar. Iptables tutorial 1.1.1; 2002; (Disponível por www em: <http://www.netfilter.org/documentation/tutorials/blueflux/iptables-tutorial.html> Acessado em: 20 de Maio de 2004).

[10] MOTA FILHO, João Eriberto: Firewall com Iptables; 2001; (Disponível por www em: <http://www.iptablesbr.cjb.net/> Acessado em: 16 de Maio de 2004).

9 Apêndices

9.1 APÊNDICE A

9.1.1 Script de Firewall Implementado em um Servidor Linux

```
# echo "1" > /proc/sys/net/ipv4/ip_forward
/sbin/modprobe iptable_nat
/sbin/modprobe ip_conntrack
/sbin/modprobe ip_conntrack_ftp
/sbin/modprobe ip_tables
/sbin/modprobe ipt_unclean
/sbin/modprobe ipt_limit
/sbin/modprobe ipt_LOG
/sbin/modprobe ipt_REJECT
/usr/sbin/iptables -F
/usr/sbin/iptables -t nat -F
/usr/sbin/iptables -P FORWARD DROP
/usr/sbin/iptables -A INPUT -i lo -j ACCEPT
/usr/sbin/iptables -A OUTPUT -o lo -j ACCEPT
/usr/sbin/iptables -A FORWARD -s 192.168.100.0/24 -d 0/0 -j ACCEPT
/usr/sbin/iptables -A FORWARD -s 0/0 -d 192.168.100.0/24 -mstate \
        --state ESTABLISHED,RELATED -j ACCEPT
/usr/sbin/iptables -t nat -A POSTROUTING -s 192.168.100.0/24 \
        -d 0/0 -j MASQUERADE
/usr/sbin/iptables -A FORWARD -s 0/0 -d 0/0 -p tcp --dport 137 -j DROP
/usr/sbin/iptables -A FORWARD -s 0/0 -d 0/0 -p tcp --dport 139 -j DROP
/usr/sbin/iptables -A FORWARD -s 0/0 -d 0/0 -p tcp --dport 138 -j DROP
echo Scritp de Firewall ativo  ..... [ OK ]
```

9.2 APÊNDICE B

9.2.1 Comandos Utilizados no *Iptables*

Comando -A, --*append*
Exemplo iptables -A INPUT ...
Este comando anexa a regra ao final das regras.

Comando -D, --*delete*
Exemplo iptables -D INPUT --*dport* 80 -j DROP, iptables -D INPUT 1
Este comando apaga uma regra da lista.

Comando -R, --replace
Exemplo iptables -R INPUT 1 -s 192.168.0.1 -j DROP
Este comando substitui a regra velha na linha especificada.

Comando -I, --insert
Exemplo iptables -I INPUT 1 --dport 80 -j ACCEPT
Insere uma regra em algum lugar da lista. A regra é inserida no número real que se é passado.

Comando -L, --list
Exemplo iptables -L INPUT
Este comando listas todas as entradas na lista especificada.

Comando -F, --flush
Exemplo iptables -F INPUT
Remove todas as regras existentes. No entanto, não altera a política.

Comando -Z, --zero
Exemplo iptables -Z INPUT
Este comando diz ao iptables para zerar todos os contadores em uma lista específica ou em todas as listas

Comando -N, --new-chain
Exemplo iptables -N allowed
Este comando diz ao kernel para criar uma nova regra com o nome especificado na tabela especificada.

Comando -X, --delete-chain
Exemplo iptables -X allowed
Este comando apaga a regra especificada de um tabela Comando -P, --policy
Exemplo iptables -P INPUT DROP
Este comando diz ao kernel para configurar um trajeto padrão, ou uma política em uma tabela.

Comando -E, --rename-chain
Exemplo Iptables -E allowed disallowed

O comando -E diz ao iptables para renomear o nome da tabela do primeiro para o segundo nome.

Um comando deve sempre ser especificado, a menos que queira listar a ajuda que vem embutida no iptables ou conseguir a versão do comando. Para conseguir a versão, use a *opção -v* e para conseguir a mensagem de ajuda, use a *opção -h*. Afigura apresentada a seguir ilustra algumas opções que podem ser usadas junto com alguns diferentes comandos.

Opções -v, --verbose
Comando usado com --list, --append, --insert, --delete, --replace
Este comando mostra uma lista de output e é principalmente usado junto com o comando de lista.

Opções -x, --exact
Comando usado com --list
Esta opção expande os números

Opções -n, --numeric
Comando usado com --list
Esta opção diz ao iptables para listras os valores numérico das saídas.

Opções --line-numbers
Comando usado com --list
O comando --line-numbers é usado para a saída de números de linha junto com o comando –list

Opções -c, --set-counters
Esta opção é usada quando é criada uma regra em algum modo ou modificada alguma regra.

Opções --modprobe
Comando usado com All
A opção --modprobe é usada para dizer ao iptables para verificar os módulos do kernel.