SEGURANÇA DA INFORMAÇÃO: UM ESTUDO DE CASO NA UNIVERSIDADE FEDERAL DE SÃO JOÃO DEL-REI

LUÍS FERNANDO DE ABREU PORTO

LAVRAS
MINAS GERAIS - BRASIL
2011

LUIS FERNANDO DE ABREU PORTO

SEGURANÇA DA INFORMAÇÃO: UM ESTUDO DE CASO NA UNIVERSIDADE FEDERAL DE SÃO JOÃO DEL-REI

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 a obtenção do titulo de especialização.

Orientador
Prof. Dsc. Joaquim Quinteiro Uchôa

Co-orientador
Esp. Roosevelt Mairink dos Santos Júnior

LAVRAS
MINAS GERAIS - BRASIL
2011

LUIS FERNANDO DE ABREU PORTO

SEGURANÇA DA INFORMAÇÃO: UM ESTUDO DE CASO NA UNIVERSIDADE FEDERAL DE SÃO JOÃO DEL-REI

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 a obtenção do titulo de especialização.

APROVADA em _____ de _____ de _____

Prof. _____

Prof. _____

Prof. Dsc. Joaquim Quinteiro Uchoa
(Orientador)

Esp. Roosevelt Mairink dos Santos Júnior
(Co-orientador)

LAVRAS
MINAS GERAIS - BRASIL
2011

DEDICATÓRIA

“Dedico aos meus pais Dair e Isaura, que despertaram desde cedo em mim o interesse e gosto pela busca de conhecimentos. Sem isso, talvez eu nunca tivesse concluído este curso.”

AGRADECIMENTOS

“Agradeço ao meu orientador, Prof. Joaquim Quinteiro Uchoa, e ao meu co-orientador, Roosevelt Mairink dos Santos Júnior, pela paciência, incentivo e orientação no trabalho desenvolvido.

Aos amigos que sempre estiveram por perto, em especial: Kito, Davi, Helder, Branda, Babi, Felipe, Joely, Luísa, Rafaela, Cecília, Luciana e Rodrigo.

À minha namorada Raquel, pela paciência e carinho mesmo em dias estressantes durante o curso.

Agradeço a Deus acima de tudo, que me conduziu com sabedoria até aqui.”

SUMÁRIO

LISTA DE FIGURAS

RESUMO

O presente trabalho objetiva mostrar a importância de um software de *firewall* para uma organização. Será apresentado um projeto de segurança da informação, desenvolvido na Universidade Federal de São João del-Rei (UFSJ), contemplando a instalação e configuração de um servidor de *firewall*, para controle do tráfego de pacotes; a instalação e configuração de um serviço de *proxy*, para acelerar a navegação na Internet e, ao mesmo tempo, permitir o controle dos conteúdos acessíveis pela rede; e o desenvolvimento de um sistema de gerenciamento do *firewall*, para melhoria da gerência. Enfim, serão enfatizados os aspectos que a nortearam a implementação desse projeto e, principalmente, como ele contribuiu para aumentar a segurança da informação nessa instituição.

Palavras-chave: Segurança da Informação, *Firewall*, *Proxy*, Controle de Conteúdo.

1. Introdução

A segurança da informação é fundamental para garantir o bom funcionamento dos sistemas e evitar que invasores capturem informações essenciais para uma empresa.

Com o crescimento das redes de computadores, os recursos de *hardware* e *software* das empresas ficam expostos a ameaças externas e/ou internas. Nesse ambiente, falhas de segurança podem causar impactos dos mais diferentes níveis, que podem ir desde simples constrangimentos até perda de mercado. Em função da necessidade de proteger os dados, os recursos e os próprios computadores, surgiram ferramentas de bloqueio de acessos indesejados, denominadas *firewalls*.

O *software* de *firewall* é um dispositivo que fica instalado em um *host* ou servidor que interconecta uma rede interna a uma rede externa. Seu objetivo principal é proteger a rede interna, filtrando e analisando os pacotes que transitam por meio dele. Baseado em algumas análises, o *firewall* pode avaliar se os pacotes podem transitar pela rede ou devem ser descartados.

No sistema operacional Linux, os sistemas de *firewall* vêm sendo aperfeiçoados constantemente, oferecendo cada vez mais recursos e, conseqüentemente, propiciando maior segurança da informação.

Nesse contexto, o presente trabalho objetiva mostrar um estudo de caso na Universidade Federal de São João del-Rei. Será apresentada uma aplicação prática de um *software* de *firewall*, instalado em um servidor Linux, enfatizando-se os aspectos que a nortearam a implementação dessa ferramenta e, principalmente, como ela contribuiu para aumentar a segurança da informação nessa instituição.

No tocante à metodologia para o desenvolvimento do trabalho, foram utilizadas a pesquisa bibliográfica, constituída principalmente de livros e artigos

científicos, o que permitirá uma visão ampla sobre o assunto, e a pesquisa exploratória, constituída do estudo supracitado.

Com relação à organização do texto, o trabalho está organizado em cinco capítulos.

O segundo capítulo faz uma contextualização geral de Segurança da Informação, abrangendo as políticas de segurança, caracterizando as vulnerabilidades e descrevendo os diversos tipos de ameaças e ferramentas de proteção.

Por se tratar do tema central do trabalho, o terceiro capítulo aborda o tema *firewall*, apresentando: tipos de firewall, localização de um *firewall*, filtragem de pacotes e principais características do *Iptables*.

O quarto capítulo traz um estudo de caso desenvolvido no Núcleo de Tecnologia da Informação (NTINF) da Universidade Federal de São João del-Rei – UFSJ.

Por fim, no último capítulo serão apresentadas as considerações finais, bem como as perspectivas futuras deste trabalho.

2. Segurança da Informação

Com o advento da informática, o mundo tornou-se cada vez mais interligado e as informações passaram a ser instantâneas. A globalização acarretou a necessidade do desenvolvimento de novas tecnologias, possibilitando realizar diversas funcionalidades para facilitar o trabalho desenvolvido nas empresas e o contato destas com o público externo.

Contudo, tais facilidades muitas vezes expõem os computadores a vulnerabilidades e ameaças. Para evitar essas exposições, surge a segurança da informação, no instante em que a informação torna-se decisória e fundamental nos planos organizacionais.

Segundo Dias (2000), na sociedade da informação, ao mesmo tempo que as informações são consideradas o principal patrimônio de uma organização, elas estão também sob constante risco como nunca estiveram antes. Com isso, a segurança de informações tornou-se um ponto crucial para a sobrevivência das instituições.

Segundo Alves (2006), a Segurança da Informação "visa proteger a informação de forma a garantir a continuidade dos negócios, minimizando os danos e maximizando o retorno dos investimentos e as oportunidades de negócios."

Tão logo uma empresa disponibilize suas informações em ambientes informatizados e compartilhados, surgirá a necessidade de resguardar tais informações da melhor maneira possível, garantindo a segurança, a disponibilidade e o uso de sistemas, tecnologias e recursos humanos qualificados.

Dentro desse contexto, Fontes (2006) afirma que a segurança da informação possui os seguintes princípios:

- *Confidencialidade* – disponibilizar informação somente a pessoas autorizadas e que necessitam da mesma;
- *Legalidade* – expor informação dentro das normas organizacionais;
- *Auditabilidade* – permitir o acesso somente registrado, possibilitando identificar o usuário e sua ação;
- *Integridade* – garantir informação verdadeira e livre de alterações não autorizadas;
- *Disponibilidade* – manter a informação acessível, sempre que requisitada;

No decorrer deste capítulo, a Segurança da Informação será abordada em diversos aspectos, como sua conceituação, políticas de segurança, vulnerabilidades, ameaças e principais ferramentas de proteção.

2.1. Conceitos Básicos

De acordo com Fontes (2006), Segurança da Informação é o conjunto de orientações, normas, procedimentos, políticas e demais ações que tem por objetivo proteger o recurso informação, possibilitando que o negócio da organização seja realizado e sua missão seja alcançada.

Sêmola (2003) conceitua Segurança da Informação como "uma área do conhecimento dedicada à proteção de ativos da informação contra acessos não autorizados, alterações indevidas ou sua indisponibilidade".

Portanto, é possível dizer que a Segurança da Informação é fundamental para proteger a empresa contra qualquer ameaça às suas informações e/ou informações que estejam sob sua responsabilidade.

Silva Filho (2008) descreve a Segurança da Informação como "um conjunto de medidas que visam a proteger e preservar informações e sistemas,

assegurando-lhes integridade, disponibilidade, não repúdio, autenticidade e confidencialidade". Esses elementos constituem os cinco pilares da segurança da informação e, portanto, são essenciais para assegurar a integridade e confiabilidade em sistemas de informações. Nesse sentido, esses pilares, juntamente com mecanismos de proteção, têm por objetivo prover suporte à restauração de sistemas de informações, adicionando-lhes capacidades de detecção, reação e proteção.

Silva Filho (2008) ainda afirma que o uso desses pilares é feito em conformidade com as necessidades específicas de cada organização. Assim, o uso pode ser determinado pela suscetibilidade das informações ou sistemas de informações, pelo nível de ameaças ou por quaisquer outras decisões de gestão de riscos.

Dessa forma, fica evidente que esses pilares são essenciais no mundo atual, onde há ambientes de natureza pública e privada conectados a nível global. Por isso, torna-se necessário dispor de uma estratégia, a fim de compor uma arquitetura de segurança que venha unificar os propósitos dos cinco pilares.

Por fim, Promon (2005) amplia o conceito de segurança da informação, considerando que ela implica a identificação das várias vulnerabilidades e ameaças junto as diversos sistemas da informação de uma empresa, independentemente da maneira em que são compartilhados ou armazenados (digitais ou impressos).

2.2. Política de Segurança da Informação

Segundo Silva, Carvalho e Torres (2003), política de segurança da informação

> "... é um conjunto reduzido de regras que definem, em linhas gerais, o que é considerado pela empresa como aceitável ou inaceitável, contendo ainda referencias às medidas a impor aos infratores. Esta política deverá referenciar todas as outras políticas existentes na empresa que contenham regras de segurança, bem como fazer alusão às normas de segurança."

Nesse sentido, a política de segurança da informação é fundamental para normatizar as estratégias vinculadas à segurança dos sistemas de uma empresa. Estas normas possibilitam, entre outras coisas, fiscalizar acessos não autorizados e incorretos.

Recomenda-se a criação de uma comissão responsável pela política de segurança da informação, para divulgação, atualização e distribuição das normas de segurança dentro da empresa. Tal divulgação pode ser através de folhetos entregues aos funcionários no processo de admissão e/ou educação continuada, distribuídos por *e-mails*, através de cartazes, ou até mesmo palestras.

O documento que descreve as normas deve conter, no mínimo, os seguintes itens:

- Definição da segurança da informação;
- Resumo das metas e importâncias;
- Comprometimento da direção da empresa;
- Definição das responsabilidades na gestão da segurança;
- Registro das políticas que devem ser seguidas;
- Conformidade com a legislação e cláusulas de contratos existentes.

Para Sêmola (2003), uma política de segurança estabelece padrões, responsabilidades e critérios para o manuseio, armazenamento, transporte e descarte das informações, dentro do nível de segurança estabelecido sob medida pela e para a empresa.

Finalmente, convém destacar que a política de segurança deve ser do conhecimento de todos os colaboradores da empresa, sendo utilizada de forma consciente pelos mesmos.

2.3. Vulnerabilidades

Vulnerabilidades são fraquezas ou deficiências presentes nas informações de um sistema. São pontos onde quaisquer sistemas são suscetíveis a um ataque. Ou seja, é uma condição encontrada em determinados recursos, processos e configurações.

Freqüentemente são realizadas tentativas de burlar a segurança de um sistema para colher e/ou modificar dados, prejudicar o sistema, entre outras finalidades. Esses objetivos escusos podem ser alcançados por meio das vulnerabilidades. Desse modo, é imprescindível que essas falhas sejam evitadas e, se porventura elas acontecerem, que sejam corrigidas o mais rápido possível, evitando que a empresa sofra uma quebra de segurança.

Sêmola (2003) classifica as vulnerabilidades em três categorias: tecnológicas, físicas e humanas.

Tecnológicas

- Equipamentos de baixa qualidade;
- Criptografia fraca;
- Sistema operacional desatualizado;
- Configuração imprópria dos *firewalls*;
- *Links* não redundantes;
- Configuração imprópria do roteadores;
- Falhas nos sistemas;

- Autorização de acesso lógico inadequado.

Físicas

- Ausência de gerador de energia;
- Ausência de normas para senhas;
- Ausência de fragmentador de papel;
- Mídias de *backups* mal acondicionadas;
- Falta de controles físicos de acesso;
- Instalações elétricas impróprias;
- Cabeamento não estruturado.

Humanas

- Falta de treinamento;
- Falta de qualificação;
- Ausência de políticas de Gestão de Pessoas;
- Ambiente organizacional ruim.

A Figura 1 apresenta um panorama com estas principais categorias de vulnerabilidades, de acordo com Sêmola (2003).

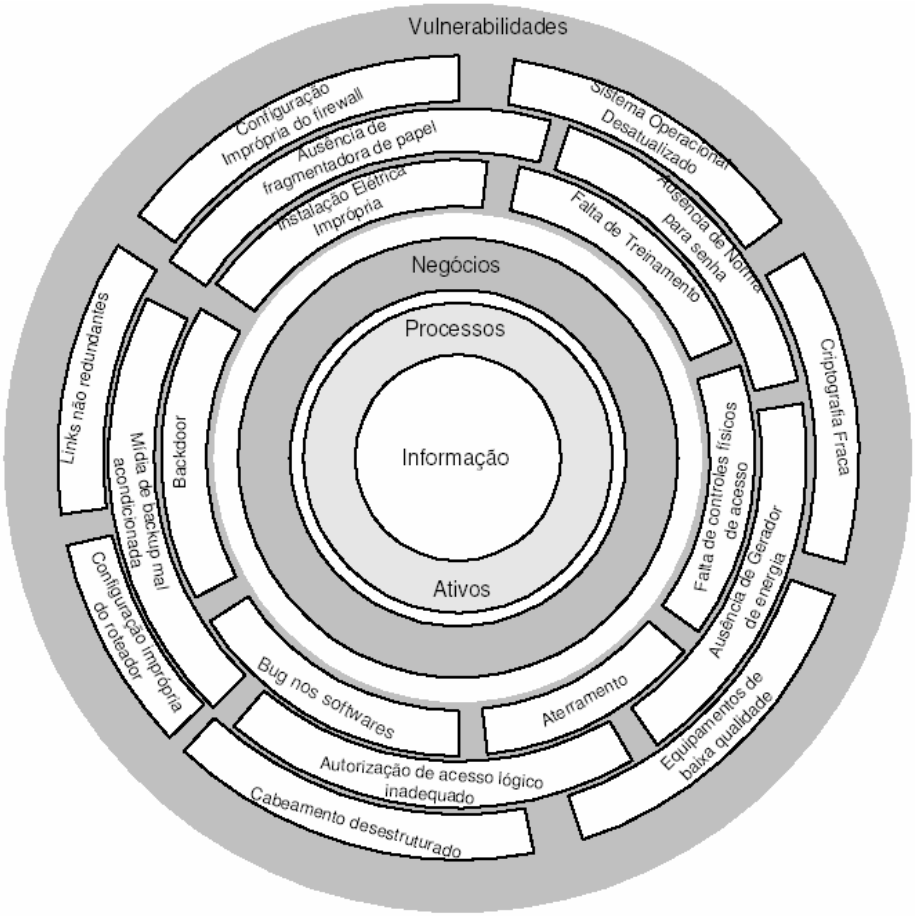

Figura 1: Diversidade panorâmica das vulnerabilidades
Fonte: Sêmola (2003)

2.4. Ameaças

À medida que os avanços tecnológicos vão acontecendo, os sistemas de informação ficam suscetíveis a diversas ameaças, como *malwares*, *hackers*, espionagem industrial, engenharia social, entre outras. Para evitar essas ameaças é imprescindível que a segurança da informação acompanhe os avanços

tecnológicos e que os profissionais da área atualizem seus conhecimentos sobre este assunto.

As ameaças podem ser definidas como qualquer ação, acontecimento ou entidade que possa agir sobre um sistema, por meio de uma vulnerabilidade, gerando um determinado impacto. Isto implica dizer que, para que uma ameaça seja realmente consistente, é necessário que o sistema apresente algum tipo de vulnerabilidade.

Sêmola (2003) conceitua ameaças como:

> "... agentes ou condições que causam incidentes que comprometem as informações e seus ativos por meio da exploração de vulnerabilidades, provocando perdas de confidencialidade, integridade e disponibilidade e, conseqüentemente, causando impactos aos negócios de uma organização."

O autor ainda divide as ameaças em três categorias, segundo sua intencionalidade.

- *Naturais* – decorrentes de fenômenos da natureza, como incêndios naturais, enchentes, terremotos, tempestades, poluição.
- *Involuntárias* – são inconscientes, podendo ser causadas por acidentes, erros, falta de energia, etc.
- *Voluntárias* – são propositais, causadas por agentes humanos como *hackers*, invasores, espiões, ladrões, criadores e disseminadores de *malwares*, incendiários, etc.

A seguir serão abordados sucintamente quatro tipos de ameaças: *malwares*, DoS, *Sniffers* e Engenharia Social.

2.4.1 *Malwares*

A palavra *malware* é formada pela união das palavras *malicious* e *software*. Portanto, *malware* é qualquer *software* que objetiva se infiltrar em um sistema de computador de forma ilícita, a fim de danificar ou cometer roubo de informações (restritas ou não).

Esta ameaça abrange diversos tipos de programas maléficos que atacam os sistemas de informação de forma diferenciada, ultrapassando suas barreiras de segurança. Segundo o Comitê Gestor da Internet no Brasil (CGI.BR, 2006) os principais *malwares* existentes são:

- *Vírus* – programa ou parte de um programa de computador, normalmente malicioso, que se propaga infectando, isto é, inserindo cópias de si mesmo e se tornando parte de outros programas e arquivos de um computador. O vírus depende da execução do programa ou arquivo hospedeiro para que possa se tornar ativo e dar continuidade ao processo de infecção.
- *Worm* – programa capaz de se propagar automaticamente através de redes, enviando cópias de si mesmo de computador para computador. Esse tipo de *malware* se dissemina de forma rápida e em grande quantidade, contaminando uma rede rapidamente e dificultando a sua exterminação.
- Cavalo de Tróia – programa, normalmente recebido como um "presente" (por exemplo, cartão virtual, álbum de fotos, protetor de tela, jogo, etc.), que além de executar funções para as quais foi aparentemente projetado, também executa outras funções normalmente maliciosas e sem o conhecimento do usuário.

- *Adware* – é um tipo de software especificamente projetado para apresentar propagandas, seja através de um *browser*, seja através de algum outro programa instalado em um computador. O *adware* causa enormes transtornos aos usuários de computadores, uma vez que abre janelas do *browser* de forma involuntária, interrompendo o seu trabalho e podendo até mesmo gerar conflitos entre os demais programas.
- *Spyware* – termo utilizado para se referir a uma grande categoria de *software* que tem o objetivo de monitorar atividades de um sistema e enviar as informações coletadas para terceiros. Dessa forma, o *spyware* objetiva invadir o sistema, interceptando informações do usuário. Além disso, pode capturar *logins* e senhas de acesso e verificar os *sites* acessados/favoritos, enviando todas as informações coletadas sem a percepção do usuário.
- *Keylogger* – programa capaz de capturar e armazenar as teclas digitadas pelo usuário no teclado de um computador.
- *Bot* – programa capaz se propagar automaticamente, explorando vulnerabilidades existentes ou falhas na configuração do *software* instalado em um computador. Adicionalmente ao *worm*, dispõe de mecanismos de comunicação com o invasor. Ou seja, o *bot* pode ser controlado remotamente: utilizando um *bot*, é possível uma pessoa monitorar diversos computadores, em locais diferentes, e exibir comandos simultâneos.
- *Rootkit* – um invasor, ao realizar uma invasão, pode utilizar mecanismos para esconder e assegurar a sua presença no computador comprometido. O conjunto de programas que fornece estes mecanismos é conhecido como *rootkit*. É importante ressaltar que o nome *rootkit* não indica que as ferramentas que o compõem são usadas para obter acesso privilegiado (*root* ou Administrador) em um computador, mas sim para mantê-lo.

Isto significa que o invasor, após instalar o *rootkit*, terá acesso privilegiado ao computador previamente comprometido, sem precisar recorrer novamente aos métodos utilizados na realização da invasão, e suas atividades serão escondidas do responsável e/ou dos usuários do computador.

2.4.2 DoS (*Denial of Service*)

O DoS (*Denial of Service*, ou Negação de Serviço) é uma ameaça que consiste na emissão de grande quantidade de arquivos, levando o computador a reiniciar ou executar trabalhos inúteis, gerando a lentidão ou interrupção de um sistema. O usuário, na tentativa de melhorar o acesso ao sistema, acaba desabilitando arquivos legítimos.

O DoS pode ser classificado em dois tipos:

- *DDoS* (*Distributed Denial of Service*) – ocorre quando o invasor possui vários computadores ao seu domínio e realiza um ataque simultâneo, dificultando a identificação do mesmo.
- *DRDoS* (*Distributed Reflection Denial of Service*) – ocorre quando diversos pacotes adulterados são enviados com o IP da vítima para um grande número de computadores, e estes respondem para o invasor. Esse ataque também é conhecido como *Ping Flood*.

2.4.3 *Sniffers*

Um s*niffer* é um *software* analisador de rede. Apesar de ser concebido para o monitoramento de redes de computadores, ele pode ser usado para fins

maliciosos, como obter arquivos confidenciais, senhas e outras informações. Nesse caso, configura-se como uma ameaça que realiza a captura de diversos arquivos em redes de computadores, conforme as especificações do invasor.

2.4.4 Engenharia Social

A Engenharia Social é um artifício utilizado por uma pessoa mal intencionada, para obter informações sigilosas e importantes por meio da confiança das pessoas. Esse tipo de ameaça não necessita de computadores ou de uma rede – depende muito da habilidade de induzir e convencer o usuário a fornecer informações essenciais ao sistema.

Segundo Silva, Carvalho e Torres (2003), a Engenharia Social tem como objetivo detectar o grau de vulnerabilidade da organização a "ataques sociais". Poderá ser realizada pelo responsável pela segurança, por outros elementos da empresa ou por prestadores deste tipo de serviços: existem empresas no mercado que os oferecem como pacotes isolados ou como parte de auditorias mais amplas. Pode ser composta por um conjunto simples de tentativas de obtenção de informação através, por exemplo, do telefone, ou poderá incluir ações mais complexas como tentativas de entrada nas instalações.

2.5. Ferramentas de Proteção

Embora haja um grande número de ameaças e vulnerabilidades atualmente, existem algumas ferramentas que podem ser utilizadas para tornarem os sistemas de informação menos suscetíveis a ataques.

A seguir serão abordadas as seguintes ferramentas de proteção de sistemas de informação: antivírus, *backup* e sistemas de detecção de intrusões (IDS).

Apesar de o *firewall* ser uma das mais importantes ferramentas de proteção de sistemas, por se tratar do tema central do trabalho, optou-se por apresentar este tema com mais detalhes, num capítulo a parte.

2.5.1 Antivírus

Os antivírus, como o próprio nome diz, são ferramentas usadas para proteger os computadores contra diversos tipos de vírus.

Devido ao grande número de vírus trafegando na rede mundial de computadores, as empresas devem utilizar antivírus em seus sistemas, visto que a não utilização colocaria em risco constante todas as suas informações.

Os antivírus podem ser divididos em duas categorias: gratuitos e por assinatura. Para melhorar a efetividade do antivírus, são necessárias constantes atualizações, conforme esclarecem Silva, Carvalho e Torres (2003):

> "Estas soluções obrigam a permanente atualização das bases de dados de assinaturas (e, com menos freqüência, dos motores de detecção e de remoção), bem como a disseminação dessas atualizações por todos os sistemas a proteger. A conseqüência, caso não exista um cuidadoso planejamento prévio, pode ser um enorme esforço de atualização dos produtos antivírus existentes que, ao ritmo de aparecimento de novos vírus, pode tornar-se uma batalha perdida."

2.5.2 *Backup*

Diante dos vários tipos de ameaças, existem aquelas que podem atacar os sistemas de informação fazendo a eliminação de alguns arquivos que fazem com que os sistemas não funcionem corretamente, é por esse motivo que existem os *backups*, que são meios de fazer uma cópia em local separado.

Atualmente os *backups* são essenciais para as empresas e cada dia mais complexos, conforme descrito por Silva, Carvalho e Torres (2003):

> "À medida que aumenta a capacidade de armazenamento disponível e cresce a complexidade dos sistemas de processamento de informação, o volume de dados armazenados segue esta tendência, atingindo proporções significativas. As empresas, cada vez mais, deparam-se com a necessidade de proteção de um conjunto complexo de informação, disperso por vários suportes e gerado por diferentes aplicações. Felizmente, as soluções de salvaguarda, ou *backup*, atuais acompanharam esta evolução e oferecem hoje níveis de desempenho e de proteção amplamente satisfatórios."

Devido à importância da informação para uma empresa, é fundamental que se realiza o *backup* diariamente para evitar não só *malwares* como também falhas no sistema e problemas de *hardware*.

2.5.3 Sistemas de Detecção de Intrusão

Os Sistemas de Detecção de Intrusão (*Intrusion Detection Systems* – IDS) são ferramentas de segurança que colhem e analisam dados, buscando detectar e impedir os ataques à rede.

Suzuki (2007) afirma que um IDS é composto de “sensores que geram os eventos de segurança, um console que controla os sensores e monitora os eventos/alertas e um *engine*, que grava os eventos num banco de dados e utiliza as regras configuradas para gerar alertas”. O autor ainda aponta que a má configuração de um IDS pode acarretar os seguintes problemas:

- *Falso-Positivo* – ocorre quando o trafego legítimo é considerado um ataque. É considerado um sério problema, pois o IDS pode bloquear tráfego legítimo, parando serviços fundamentais para uma empresa.
- *Falso-Negativo* – ocorre quando um ataque não é percebido pelo IDS, passando como se fosse tráfego legítimo.

3. *Firewalls*

Um *firewall* é um sistema que impõe uma política de controle de acesso entre duas redes, tendo as seguintes propriedades (Cheswick, Bellovin e Rubin, 2003):

- Todo tráfego de dentro para fora de uma rede, e vice-versa, deve passar pelo *firewall*.
- Apenas tráfego autorizado, como definido pela política de segurança local, terá permissão de passar.
- O próprio *firewall* deve ser imune a penetrações.

De acordo com a NBSO (2003), um *firewall* é um instrumento importante para implantar a política de segurança da sua rede. Ele pode reduzir a informação disponível externamente sobre a sua rede, ou, em alguns casos, até mesmo barrar ataques a vulnerabilidades ainda não divulgadas publicamente e para as quais correções não estão disponíveis. Por outro lado, *firewalls* não são infalíveis. A simples instalação de um *firewall* não garante que uma rede esteja segura contra invasores. Assim, um *firewall* não pode ser a única linha de defesa – ele é mais um dentre os diversos mecanismos e procedimentos que aumentam a segurança de uma rede.

Outra limitação dos *firewalls* é que eles protegem apenas contra ataques externos ao *firewall*, nada podendo fazer contra ataques que partem de dentro da rede por ele protegida.

3.1 Tipos de *Firewall*

Roger (2005) afirma que atualmente existem três tipos de *firewalls*: filtro de pacotes, filtro de pacotes com base no estado da conexão e filtros de pacotes na camada de aplicação.

Os filtros de pacotes funcionam permitindo ou eliminando pacotes com base em seus endereços de origem ou destino, ou nos números de porta. As decisões são tomadas com base no conteúdo do pacote que o *firewall* está recebendo ou enviando.

Um *firewall* com filtro de pacotes com base no estado da conexão (*stateful packet filter*) baseia suas ações utilizando dois elementos: dados contidos no cabeçalho do pacote e na tabela de estados, que armazena informações do estado de todas as conexões que estão trafegando através do *firewall* e usa estas informações em conjunto com as regras definidas pelo administrador, para permitir ou não a passagem de um determinado pacote.

Os *firewalls* com filtros na camada de aplicação são mais complexos, pois utilizam um código especial para filtrar a aplicação desejada. Por exemplo, os *firewalls* com filtros na camada de aplicação podem identificar vírus anexos às mensagens (e-mails) que estão chegando ou saindo de um ambiente computacional. Outro recurso disponível neste tipo de *firewall* são os registros de todo o conteúdo do tráfego enviado ou recebido.

3.2 Escolha de um *Firewall*

De acordo com a NBSO (2003), a escolha de uma solução de *firewall* está atrelada a fatores como custo, recursos desejados e flexibilidade. Entretanto, um ponto essencial é a familiaridade com a plataforma operacional do *firewall*.

A maioria dos *firewalls* está disponível para um conjunto reduzido de plataformas operacionais, e a sua escolha deve se restringir a um dos produtos que roda sobre uma plataforma com a qual os administradores da rede tenham experiência.

Existem, basicamente, duas razões para esta recomendação. A primeira delas é que o administrador da rede deve estar familiarizado o suficiente com o sistema onde o *firewall* será executado para configurá-lo de forma segura. A existência de um *firewall* instalado em um sistema inseguro pode ser até mais perigosa do que a ausência do *firewall* na rede.

A segunda razão é que os produtos tendem a seguir a filosofia da plataforma onde rodam; por exemplo, a maioria dos *firewalls* para Windows é configurada através de menus e janelas, ao passo que muitos *firewalls* para Unix são configurados por meio de arquivos texto.

3.3 Localização de um *Firewall*

A localização dos *firewalls* na rede depende normalmente da sua política de segurança. Contudo, existem algumas regras que se aplicam à grande maioria dos casos:

- Todo o tráfego deve passar pelo *firewall*. Um *firewall* só pode atuar sobre o tráfego que passa por ele. A eficácia pode ser severamente comprometida se existirem rotas alternativas para dentro da rede (*modems*, por exemplo). Caso não seja possível eliminar todos esses caminhos, eles devem ser documentados e fortemente vigiados através de outros mecanismos de segurança.
- Deve-se ter um filtro de pacotes no perímetro da rede. Esse filtro pode estar localizado entre o roteador de borda e o interior da rede ou no

próprio roteador, se ele tiver esta capacidade. O filtro de pacotes de borda é importante para tarefas como bloqueio global de alguns tipos de tráfego e bloqueio rápido de serviços durante a implantação de correções após a descoberta de uma nova vulnerabilidade.

- Deve-se colocar os servidores externos em uma *DeMilitarized Zone* (DMZ). É recomendável colocar os servidores acessíveis externamente (Web, FTP, correio eletrônico, etc.) em um segmento de rede separado e com acesso altamente restrito, conhecido como DMZ. A principal importância disso é proteger a rede interna contra ataques provenientes dos servidores externos. Por exemplo, suponha que um atacante invada o servidor Web e instale um *sniffer* na rede. Se este servidor Web estiver na rede interna, a probabilidade de ele conseguir capturar dados importantes (tais como senhas ou informações confidenciais) é muito maior do que se ele estiver em uma rede isolada.
- Deve-se considerar o uso de *firewalls* internos. Em alguns casos, é possível identificar na rede interna grupos de sistemas que desempenham determinadas tarefas comuns, tais como desenvolvimento de *software*, *webdesign* e administração financeira. Nesses casos, recomenda-se o uso de *firewalls* internos para isolar estas sub-redes umas das outras, com o propósito de aumentar a proteção dos sistemas internos e conter a propagação de ataques bem-sucedidos.

3.4 Filtragem de Pacotes de um *Firewall*

De acordo com Ribeiro (2004), existem basicamente dois critérios de filtragem que podem ser empregados em *firewalls*. O primeiro é o de *default deny*, ou seja, todo o tráfego que não for explicitamente permitido é bloqueado.

O segundo *default allow*, é o contrário, ou seja, todo o tráfego que não for explicitamente proibido é liberado.

A configuração dos *firewalls* deve seguir a política de segurança da rede. Se a política permitir, é recomendável adotar uma postura de *default deny*. Esta abordagem é, geralmente, mais segura, pois requer uma intervenção explícita do administrador para liberar o tráfego desejado, o que minimiza o impacto de eventuais erros de configuração na segurança da rede. Além disso, ela tende a simplificar a configuração dos *firewalls*.

O tráfego para a DMZ deve ser altamente controlado. As únicas conexões permitidas para os sistemas dentro da DMZ devem ser as relativas aos serviços públicos (acessíveis externamente). Conexões partindo da DMZ para a rede interna devem ser na sua maioria, tratadas como conexões oriundas da rede externa, aplicando-se a política de filtragem correspondente.

3.5 *Iptables*

Segundo Neto (2004), o *Iptables* compõe a quarta geração de sistemas *firewalls* no Linux, que foi incorporada à versão 2.4 do *kernel*. Ele é uma versão mais completa e tão estável quanto seus antecessores *Ipfwadm* e *Ipchains*, implementados nos *kernels* 2.0 e 2.2, respectivamente.

O *Iptables* é amplamente utilizado devido às funções de *firewall* estarem agregadas à própria arquitetura do *kernel*. O Linux utiliza um recurso independente em termos de *kernel* para controlar e monitorar todo o tipo de fluxo de dados dentro de sua estrutura operacional.

A função do *kernel* é de trabalhar ao lado de processos e tarefas. Por esse motivo, foi agregado ao *kernel* um módulo chamado *Netfilter* para controlar seu próprio fluxo interno. Criado por Marc Boucher, James Morris, Harald Welte e Rusty Russel, o *Netfilter* é um conjunto de situações agregadas inicialmente ao

kernel do Linux e divididas em tabelas. Sob uma ótica mais prática, é possível considerar o *Netfilter* como um grande banco de dados, que contém em sua estrutura três tabelas padrão: *Filter, Nat* e *Mangle*.

3.5.1 Tabela *Filter*

A tabela *Filter* é a tabela padrão do *Netfilter*, que trata das situações implementadas por um *firewall* de filtro de pacotes. O *kernel* começa com três listas de regras (denominadas *firewall chains* ou apenas *chains*) na tabela *Filter*, a saber (Neto, 2004):

- INPUT – contém as regras para pacotes cujo destino é o próprio *firewall*.
- FORWARD – contém as regras para pacotes que foram originados em algum *host* e vão precisar passar pelo *firewall* para chegar ao seu destino (ou seja, pacotes que atravessam o *firewall*).
- OUTPUT – contém as regras para pacotes originados pelo próprio *firewall*.

Uma *chain* é uma lista de regras. Cada regra diz o que deve ser feito com determinado pacote, conforme o respectivo cabeçalho do pacote. Se uma regra não se referir a um determinado pacote, então a próxima regra na *chain* será consultada. Se não houver mais regras a consultar, o *kernel* analisará a política da *chain* para decidir o que fazer (aceitar ou rejeitar o pacote, por exemplo).

A Figura 2 apresenta um diagrama de funcionamento da tabela *Filter*.

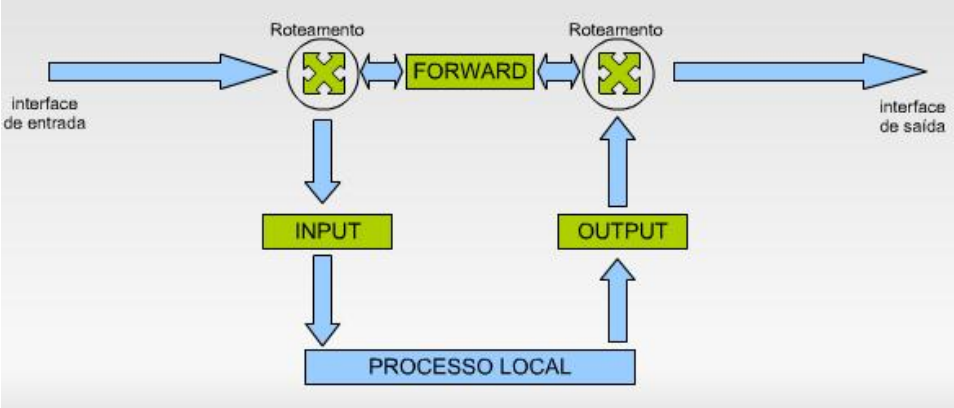

Figura 2: Funcionamento da tabela *Filter* do *Iptables*

Quando um pacote chega, o *kernel* analisa o destino do pacote. Esse processo é chamado de roteamento (*routing*).

- Se o pacote for destinado à própria máquina, ele será encaminhado para a *chain* INPUT. Se o pacote passar pela *chain* INPUT, ele será enviado para a máquina de destino.
- Se o *kernel* possui suporte para encaminhamento (*forwarding*) e o pacote é destinado a outra interface de rede, o pacote vai para a *chain* FORWARD. Se o pacote for aceito, ele será enviado. Por outro lado, se o *kernel* não possuir suporte a *forwarding*, o pacote será descartado.

Finalmente, um programa rodando na máquina *firewall* pode enviar pacotes. Esses pacotes passam pela *chain* OUTPUT imediatamente. Se o pacote for aceito, continuará o seu caminho. Caso contrário, o pacote será descartado.

3.5.2 Tabela *Nat*

A tabela *Nat* implementa as funções de NAT (*Network Address Translation*) no *host* do *firewall*. Existem vários recursos que utilizam NAT. Os mais conhecidos são:

- Mascaramento (*masquerading*);
- Redirecionamento de portas (*port forwarding* ou PAT);
- Redirecionamento de servidores (*forwarding*);
- Proxy transparente (*transparent proxy*);
- Balanceamento de carga (*load balance*).

O NAT pode ser classificado em:

- SNAT (*Source NAT*) – utilizado quando se deseja alterar o endereço de origem de um pacote. O mascaramento é um exemplo de SNAT.
- DNAT (*Destination NAT*) – utilizado quando se deseja alterar o endereço de destino de um pacote. O redirecionamento de portas, o redirecionamento de servidores, o *load balance* e o *proxy* transparente são exemplos de DNAT.

As *chains* da tabela Nat são:

- PREROUTING – utilizada para analisar pacotes que estão entrando no *kernel* para sofrerem NAT. Esta *chain* pode fazer ações de NAT com o endereço de destino do pacote. Ou seja, ela permite realizar DNAT.

- OUTPUT – utilizada para analisar pacotes gerados na própria máquina e que sofrerão NAT. Esta *chain* pode fazer ações de NAT com o endereço de destino do pacote. Ou seja, ela também permite realizar DNAT.
- POSTROUTING – utilizada para analisar pacotes que estão saindo do *kernel*, após sofrerem NAT. Esta *chain* pode fazer ações de NAT com o endereço de origem do pacote. Ou seja, ela permite realizar SNAT.

A Figura 3 apresenta um diagrama de funcionamento da tabela *Nat*.

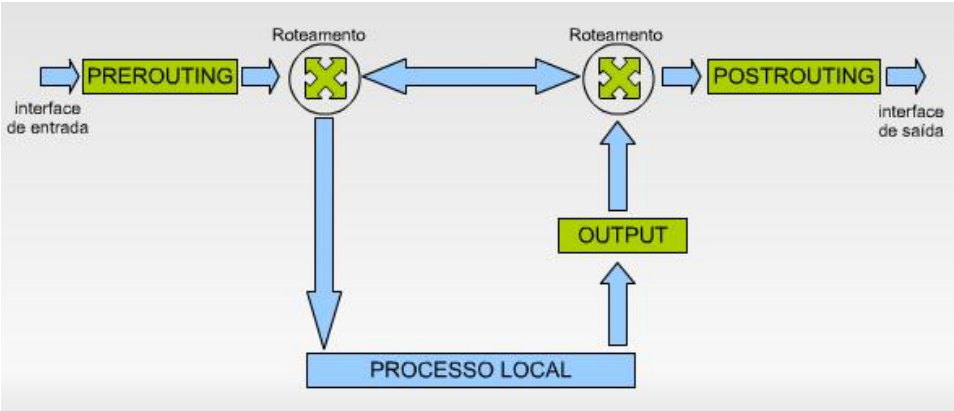

Figura 3: Funcionamento da tabela *Nat* do *Iptables*

3.5.3 Tabela *Mangle*

A tabela *Mangle* permite especificar ações especiais para o tratamento do tráfego que atravessa as *chains*. Nesta tabela existem cinco *chains*: PREROUTING, POSTROUTING, INPUT, OUTPUT e FORWARD.

A Figura 4 apresenta um diagrama de funcionamento da tabela *Mangle*.

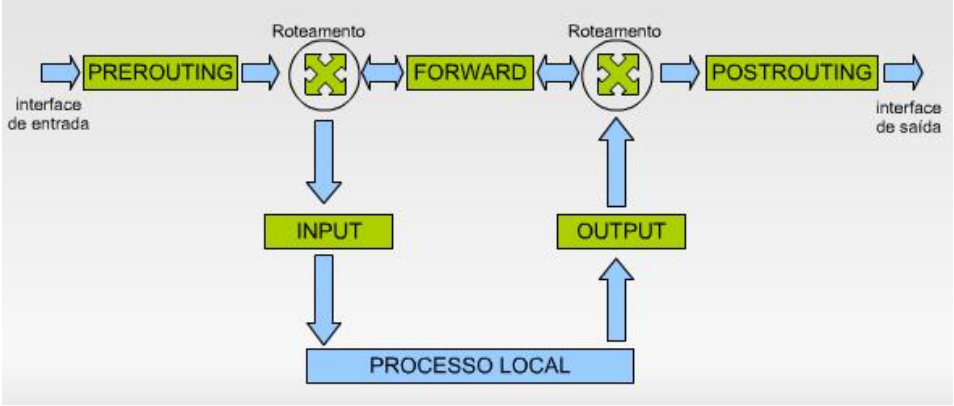

Figura 4: Funcionamento da tabela *Mangle* do *Iptables*

Opções como o tipo de serviço (*Type of Service*, ou TOS) podem ser especificadas na tabela *Mangle*, para classificar e aumentar consideravelmente a velocidade do tráfego em tempo real. O TOS permite filtrar pacotes que trafegam na rede, capturando informações sobre o tipo de serviço ao qual o pacote se destina. Estas informações estão contidas no cabeçalho de cada pacote.

Uma das vantagens da utilização do tipo de serviço é dar prioridade ao tráfego de pacotes interativos (como os do ICQ, IRC, servidores de *chat*, entre outros). Por exemplo, com o TOS especificado, mesmo que haja um *download* consumindo boa parte da banda da rede, os pacotes com prioridade interativa serão enviados antes, aumentando a eficiência do uso de serviços na referida máquina.

Em geral, cada uma das *chains* da tabela *Mangle* é processada antes da *chain* correspondente nas tabelas *Filter* e *Nat*, justamente para permitir a aplicação de opções especiais para o tráfego. Por exemplo, a *chain* PREROUTING da tabela *Mangle* é processada antes da *chain* PREROUTING da tabela *Nat*.

A Figura 5 ilustra o relacionamento entre as *chains* da tabelas *Mangle*, *Filter* e *Nat*.

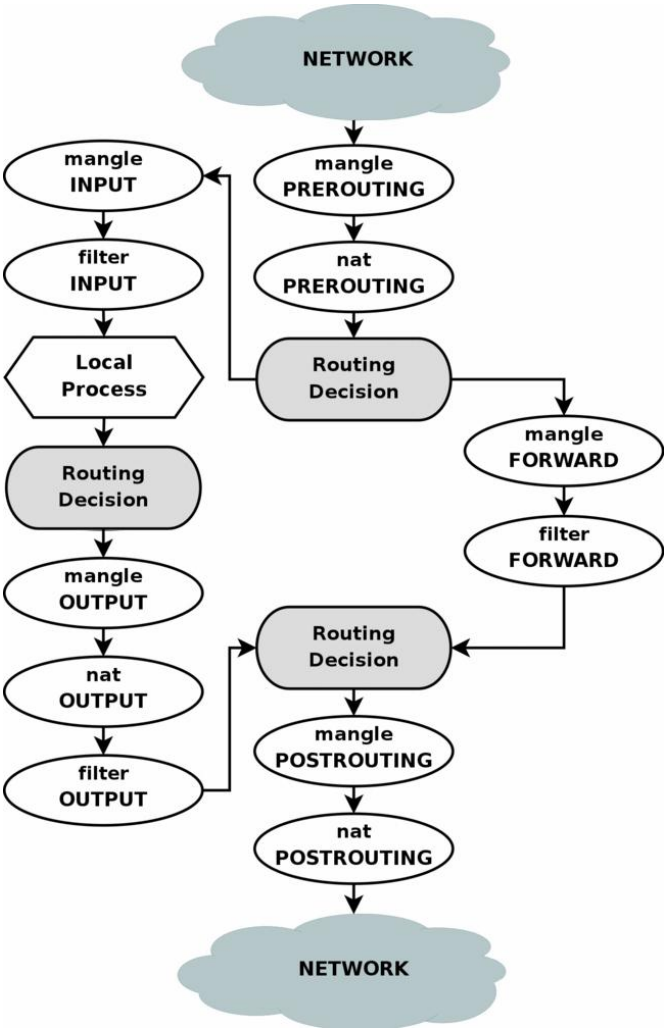

Figura 5: Relacionamento entre as *chains* das tabelas *Mangle*, *Filter* e *Nat*
Fonte: Adaptada de Andreasson (2006)

3.5.4 *Iptables* em Sistemas Linux

Conforme dito na seção 3.5, o Linux possui funções de *firewall* agregadas ao *kernel* pelo módulo *Netfilter*. Para que fosse possível moldar o *Netfilter* conforme as necessidades de cada *host*, rede ou sub-rede, foram desenvolvidas ferramentas de manipulação nativa (*Front-End*). Essas ferramentas permitem controlar as *chains* contidas nas tabelas, agregando regras de tráfego.

O *Iptables* é uma ferramenta de *Front-End* desenvolvida por Rust Russel (que participou do projeto de desenvolvimento do *Netfilter*), em colaboração com Michel Neuling, e foi incorporada à versão 2.4 do *kernel* em julho de 1999. É composto pelos seguintes aplicativos:

- *iptables* – aplicativo principal do pacote *Netfilter* para protocolos IPv4.
- *ip6tables* – aplicativo principal do pacote *Netfilter* para protocolos IPv6.
- *iptables-save* – aplicativo que salva todas as regras inseridas na sessão ativa e/ou em memória, em determinado arquivo informado pelo administrador do *firewall*.
- *iptables-restore* – aplicativo que restaura todas as regras salvas pelo *software iptables-save*.

Por estar incorporada diretamente ao *kernel*, a configuração do *Iptables* não se dá por via de arquivos de configuração – sua manipulação é realizada por síntese digitada em *shell*, ou seja, uma regra na *shell* somente estará valendo para aquela sessão em memória. Uma vez que o computador *firewall* for reiniciado ou desligado, tais regras serão perdidas e não mais poderão ser resgatadas. Para resgatar as regras atuais do sistema armazenadas em memória RAM é utilizada a ferramenta *iptables-save*, que as armazena em um arquivo.

Para restaurar ou reaplicar as regras salvas é utilizada a ferramenta *iptables-restore*.

O *Iptables*, além de realizar as tarefas de forma veloz, segura, eficaz e econômica, apresenta várias funcionalidades, tais como:

- implementação de filtros de pacotes;
- desenvolvimento de QoS sobre o tráfego;
- suporte a *Source Network Address Translation* (SNAT) e DNAT;
- direcionamento de endereços e portas;
- mascaramento de tráfego;
- detecção de fragmentos;
- monitoração de tráfego;
- bloqueio de ataque de *Spoofing*, *Syn-Flood*, DoS, *scanners* ocultos, "*pings* da morte", entre outros.

A sintaxe básica do *Iptables* é definida por:

iptables [-t <tabela>] [comando] [ação] [alvo]

As tabelas são as mesmas que compõem o *Netfilter*: *Filter*, *Nat* e *Mangle*. Por exemplo:

iptables -t filter
iptables -t nat
iptables -t mangle

A tabela *Filter* é a tabela padrão do *Iptables*. Se for adicionada uma regra sem a flag **–t**, o *Iptables* aplicará situações contidas na tabela *Filter* a essa regra. Já no caso das tabelas *Nat* e *Mangle* é necessário especificar sempre.

Segundo Neto (2004), os comandos das *chains* são definidos por:

- **–A:** adiciona uma nova entrada ao fim da lista de regras;
- **–D:** apaga uma regra específica da lista;
- **–L:** exibe as regras existentes na lista;
- **–P:** altera a política padrão das *chains*. Inicialmente, todas as *chains* estão configuradas como ACCEPT, ou seja, aceitam todo e qualquer tipo de tráfego;
- **–F:** remove todas as entradas adicionadas à lista de regras, sem alterar a política padrão (**–P**);
- **–I:** insere uma nova regra ao início da lista de regras;
- **–R:** substitui uma regra já adicionada por outra;
- **–N:** permite inserir ou criar uma nova *chain* em uma tabela específica;
- **–E:** renomeia uma *chain*;
- **–X:** apaga uma *chain* criada pelo administrador do *firewall*.

As seguintes ações podem ser configuradas:

- **–p:** especifica o protocolo aplicado à regra. Pode ser qualquer valor numérico especificado no arquivo **/etc/protocol** ou o próprio nome do protocolo (TCP, UDP, ICMP, etc.);
- **–i:** especifica a interface de entrada a ser utilizada. Como um *firewall* possui mais de uma interface, esta ação acaba sendo muito importante para distinguir a qual interface de rede determinado filtro deve ser aplicado;

- **–o:** especifica a interface de saída a ser utilizada e se aplica da mesma forma que a ação **–i**. Porém, somente as regras de OUTPUT e FORWARD se aplicam neste caso;
- **–s:** especifica a origem do pacote ao qual a regra deve ser aplicada. A origem pode ser um *host* ou uma rede;
- **–d:** especifica o destino do pacote ao qual a regra deve ser aplicada. Sua utilização se dá da mesma maneira que a ação **–s**;
- **!** significa exclusão, e é utilizada quando se deseja aplicar uma exceção a uma regra. É utilizada juntamente com as ações **–s**, **–d**, **–p, –i** e **–o**;
- **–j:** define o alvo (*target*) do pacote, caso o mesmo se encaixe em uma regra;
- **--sport:** indica a porta de origem do pacote. Com essa opção é possível aplicar filtros com base na porta de origem do pacote (somente protocolo TCP ou UDP);
- **--dport:** indica a porta de destino do pacote e funciona da forma similar à ação **--sport**.

Os seguintes alvos podem ser configurados:

- **ACCEPT:** corresponde a aceitar, ou seja, permitir a entrada e a passagem do pacote em questão;
- **DROP:** corresponde a descartar. Um pacote conduzido para este alvo (*target*) é descartado imediatamente. O *target* DROP não informa ao dispositivo emissor do pacote o que houve;
- **REJECT:** corresponde a rejeitar. Um pacote conduzido para este alvo (*target*) é automaticamente descartado. A diferença do REJECT para o DROP é que o REJECT retorna uma mensagem de erro ao *host* emissor do pacote informando o que houve;

- **LOG:** cria uma entrada de *log* no arquivo de *logs* do sistema sobre a utilização dos demais alvos (*targets*). Justamente por isso, o LOG deve ser utilizado antes dos demais alvos;
- **RETURN:** retorna o processamento da *chain* anterior sem processar o resto da *chain* atual;
- **QUEUE:** encarrega um programa em nível de usuário de administrar o processamento de fluxo atribuído ao mesmo;
- **SNAT:** altera o endereço de origem das máquinas clientes antes de os pacotes serem roteados;
- **DNAT:** altera o endereço de destino das máquinas clientes;
- **REDIRECT:** realiza o redirecionamento de portas em conjunto com a opção **--toport**;
- **TOS:** prioriza a entrada e saída de pacotes baseados em seu tipo de serviço.

3.6 Comentários Finais

Por se tratar do tema central do presente trabalho, este capítulo abordou o tema *firewall*. Foram apresentados os principais tipos de *firewall* existentes: filtro de pacotes, filtro de pacotes com base no estado da conexão e filtros de pacotes na camada de aplicação.

Com relação aos critérios de filtragem que podem ser empregados em *firewalls*, dois tipos foram abordados: *default deny*, no qual todo o tráfego que não for explicitamente permitido deve ser bloqueado; e *default allow*, no qual todo o tráfego que não for explicitamente proibido deve ser liberado. Independente do critério escolhido é imprescindível que a configuração do *firewall* siga a política de segurança da rede.

Por fim, considerando sua utilização no estudo de caso descrito no capítulo a seguir, o *Iptables* foi abordado de forma um pouco mais detalhada. Foram apresentadas suas principais características, as tabelas que o compõem (*Filter*, *Nat* e *Mangle*) e a relação existente entre elas, a forma de tratamento dos pacotes e a utilização do *Iptables* em sistemas Linux, com os principais comandos de *chains*, ações e alvos.

4. Estudo de Caso – Universidade Federal de São João del-Rei

Neste capítulo será apresentado um estudo de caso desenvolvido no Núcleo de Tecnologia da Informação (NTINF) da Universidade Federal de São João del-Rei – UFSJ. Inicialmente, será feita uma contextualização do NTINF, a partir da descrição de um breve histórico do Núcleo, da sua estrutura organizacional e das suas competências.

Em seguida será apresentado um projeto de Segurança da Informação desenvolvido pelo NTINF, abrangendo os recursos computacionais utilizados e a implementação do *firewall*, de um serviço de *proxy* e de um sistema de gerenciamento do *firewall* via *web*.

4.1 Contextualização do Ambiente Organizacional

4.1.1 Breve Histórico

O Núcleo de Tecnologia da Informação – NTINF foi criado junto com a Universidade Federal de São João del-Rei – UFSJ. Entretanto, na época de sua criação, o órgão não possuía essa nomenclatura, sendo denominado de Centro de Processamento de Dados – CPD. Em 1990, teve seu nome modificado para Núcleo de Informática – NINFO e, no ano de 2001, com a transformação da Instituição em Universidade, teve o seu nome novamente alterado para Núcleo de Tecnologias de Informação. Propositalmente, as mudanças realizadas no nome do órgão, durante os seus 19 anos de existência, procuraram refletir o significado momentâneo da informática na Instituição.

O NTINF foi criado no início da microinformática, o que permitiu que toda a estrutura de informática na Instituição fosse criada com base na microinformática. Nessa época, alguns fornecedores tentaram impedir a

implantação da microinformática na Instituição, destacando a importância dos equipamentos de grande porte. Por isso, deve-se destacar o papel da equipe do CPD naquele momento, que resistiu às pressões internas e externas pela utilização de tecnologias de grande porte e apostou todas as suas expectativas na microinformática, que acabou prevalecendo até os dias atuais.

Até o ano de 1989, havia apenas quatro servidores lotados no órgão – um analista, um programador e dois digitadores. Eles desempenhavam tarefas tradicionais dos CPDs daquela época, como, por exemplo, a elaboração da folha de pagamento. Entretanto, no final de 1989, foram contratados três programadores e três digitadores. Conseqüentemente, houve uma mudança cultural na equipe, que passou a desenvolver sistemas voltados para a operacionalização da Instituição. Além disso, com o surgimento das redes de computadores, tal mudança cultural permitiu a descentralização na utilização dos recursos de informática e o surgimento de novos sistemas integrados. Nesse momento, adotou-se o *software Novell* como gerenciador da rede de computadores e a linguagem *Clipper* para o desenvolvimento de sistemas.

No ano de 1996 a Internet chegou à Instituição, com uma velocidade de 64 kbps. Entretanto, era restrita aos docentes e a poucos técnicos da área administrativa. De 1996 a 1998, devido a questões políticas, a Internet esteve sob responsabilidade da então Vice-Diretoria de Assuntos Acadêmicos – VIDAC e não do Núcleo de Informática. Obviamente, esta separação trouxe vários problemas técnicos e operacionais que acabaram resultando, no final de 1998, na criação do Setor de Administração e Gerenciamento da Internet dentro do Núcleo de Informática.

No final de 1999, considerando o avanço das redes de computadores e a necessidade de ampliação da UFSJ, um grande projeto de rede foi colocado em prática. Os serviços executados englobaram a substituição de antigas conexões de cabos coaxiais por cabos UTP, a interligação de novos prédios à rede

existente e a conexão entre os *campi* de São João del-Rei por meio de fibras óticas.

Todas estas ações possibilitaram o aumento da taxa de transferência de arquivos via rede de 10Mbps para 100Mbps e/ou 1Gbps. Consequentemente, este avanço também permitiu à equipe do Núcleo de Informática optar por desenvolver sistemas utilizando outras linguagens de programação, como *Delphi* e PHP, utilizando os recursos da rede.

4.1.2 Estrutura Organizacional

O NTINF é um órgão de assessoramento subordinado diretamente à Reitoria da UFSJ, sendo responsável pela informática na Instituição e atuando não somente em nível administrativo, mas também no contexto acadêmico e científico. São algumas de suas atribuições:

- Assessorar, propor e implementar políticas de Sistemas de Informação e políticas de Internet e Rede para a UFSJ;
- Elaborar o Plano de Tecnologia da Informação para a UFSJ, seguindo as diretrizes maiores fixadas no planejamento estratégico, em interação com as diversas áreas usuárias e em consonância com as comissões e comitês ligados ao assunto, existentes e que venham a ser formados;
- Desenvolver, implantar, efetuar manutenção e dar suporte e treinamento em Sistemas de Informação desenvolvidos pelo NTINF;
- Dar manutenção em equipamentos de informática pertencentes ao patrimônio da UFSJ ou oriundos de projetos institucionais;
- Dar suporte a aplicativos legalizados ou livres, utilizados no âmbito da UFSJ, de acordo com as competências da equipe do NTINF;
- Manter a rede física e lógica da UFSJ em funcionamento;

- Manter a Internet em funcionamento;
- Manter a segurança, a integridade e a confiabilidade das bases de dados, assim como a segurança dos Sistemas de Informação.

Em termos da estrutura organizacional, atualmente o NTINF compreende o Setor de Internet e Redes (SETIR), o Setor de Desenvolvimento de Sistemas de Informação (SEDSI), a Área de Manutenção de Equipamentos e a Área de Suporte aos Usuários.

A Figura 6 apresenta a estrutura organizacional do NTINF.

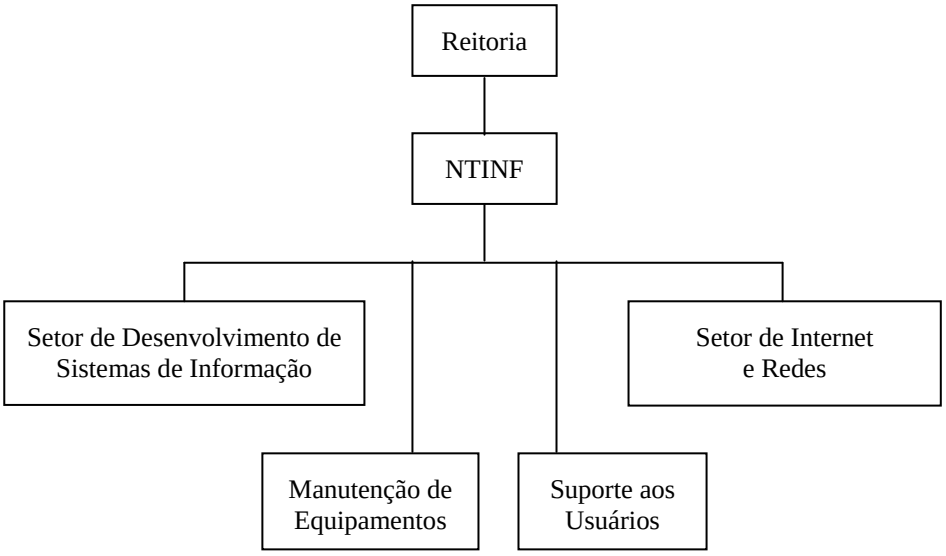

Figura 6: Estrutura Organizacional do NTINF

Com relação às funções desempenhadas, o NTINF é administrado por um Diretor, indicado e nomeado pelo Reitor da UFSJ. Cada Setor tem uma Chefia indicada pelo Diretor do NTINF e nomeada pelo Reitor da UFSJ.

Em caso de afastamentos autorizados ou férias do Diretor do NTINF, será indicado o Chefe do SEDSI ou o Chefe do SETIR como seu substituto imediato.

4.1.3 Competências

Ao Diretor do NTINF compete:

- Planejar, supervisionar e coordenar as atividades do NTINF;
- Cumprir e fazer cumprir as normas e orientações dos Órgãos Superiores da UFSJ;
- Baixar normas e atos de serviços relativos à sua unidade;
- Assessorar o Reitor em assuntos do âmbito de sua competência;
- Promover estudos, reuniões e apresentar sugestões para o aperfeiçoamento e desenvolvimento das atividades;
- Acompanhar a execução de contratos ligados diretamente às atribuições do NTINF;
- Zelar pelo cumprimento da legislação vigente quanto à utilização dos recursos de informática;
- Integrar a comissão de informática;
- Fazer proposições sobre a política de Tecnologia de Informações da UFSJ;
- Participar da formulação, acompanhamento e avaliação da implantação de novos sistemas em Tecnologia de Informação, no âmbito de suas competências;
- Apresentar relatório anual do NTINF à Reitoria;
- Propor e elaborar normas para a melhoria de serviços.

Ao Chefe do SEDSI compete:

- Desenvolver, implantar, efetuar manutenção e dar suporte e treinamento em Sistemas de Informação desenvolvidos pelo NTINF;
- Estabelecer políticas de segurança de acesso e proteção dos Sistemas de Informação, bem como pelos equipamentos sob responsabilidade do SEDSI;
- Manter a segurança, a integridade e a confiabilidade das bases de dados sob responsabilidade do SEDSI;
- Manter, documentar e organizar usuários, grupos e seus respectivos acessos aos Sistemas de Informação;
- Pesquisar, testar e implementar novas tecnologias de desenvolvimento de Sistemas de Informação;
- Assessorar o Diretor do NTINF em assuntos do âmbito de sua competência;
- Promover estudos, reuniões e apresentar sugestões para o aperfeiçoamento e desenvolvimento das atividades.

Ao Chefe do SETIR compete:

- Manter e organizar usuários, grupos e seus respectivos acessos à rede Internet;
- Zelar pelo cumprimento de normas de utilização da Internet;
- Estabelecer políticas de segurança de acesso e proteção às informações dos equipamentos de Internet e Rede;
- Manter a segurança, a integridade e a confiabilidade das bases de dados sob responsabilidade do SETIR;

- Gerenciar o portal da UFSJ;
- Manter, documentar e projetar a rede física da UFSJ;
- Estabelecer políticas de uso da rede física da UFSJ;
- Pesquisar, testar e implementar tecnologias de rede;
- Assessorar o Diretor do NTINF em assuntos do âmbito de sua competência;
- Promover estudos, reuniões e apresentar sugestões para o aperfeiçoamento e desenvolvimento das atividades.

4.2 Implementação de um Projeto de Segurança da Informação

Com o advento do Programa de Apoio a Planos de Reestruturação e Expansão das Universidades Federais (REUNI)[1], do Governo Federal, a UFSJ iniciou um processo de expansão. A ampliação de cursos de graduação e de pós-graduação, a contratação de novos docentes e técnicos administrativos, a construção de novos prédios e a abertura de novos *campi* são apenas alguns exemplos de ações tomadas pela universidade.

Essa expansão impactou não apenas o Ensino, a Pesquisa e a Extensão universitária, mas também o segmento administrativo, em especial na área de informática, que necessitou acompanhar tal crescimento.

Nesse cenário de expansão, em 2008, o NTINF começou a implementar um novo projeto de Segurança da Informação, particularmente voltado à reestruturação da segurança lógica da rede da UFSJ. Os objetivos específicos do projeto são:

[1] *Site* oficial: http://reuni.mec.gov.br

- Instalação e configuração de um servidor de *firewall*, a fim de controlar todo o tráfego da rede UFSJ;
- Instalação e configuração de um serviço de *proxy*, para acelerar a navegação na Internet e, ao mesmo tempo, permitir o controle dos conteúdos acessíveis pela rede UFSJ;
- Desenvolvimento de um sistema de gerenciamento do *firewall*, a fim de propiciar uma gestão adequada e eficiente pelo Setor de Internet e Redes.

4.2.1 *Firewall*

Considerando que, na UFSJ, todos os serviços de internet são implementados utilizando o sistema operacional Linux, e considerando ainda que o Linux possui funções de *firewall* agregadas ao *kernel* pelo módulo *Netfilter* (conforme dito na seção 3.5.4), o NTINF optou pelo *Iptables* como *front-end* para o serviço de *firewall*.

Por estar incorporada diretamente ao *kernel*, a configuração do *Iptables* é realizada por síntese digitada em *shell*. Dessa forma, por questões de praticidade, o NTINF criou um *shell script*, que é executado toda vez que a respectiva máquina é iniciada.

De forma geral, o *script* de *firewall* da UFSJ está organizado da seguinte forma:

1. Ativação de diretivas do *kernel*. Nesse caso, foram utilizados os seguintes comandos:

```
echo 8192 > /proc/sys/net/ipv4/neigh/default/gc_thresh1
echo 8192 > /proc/sys/net/ipv4/neigh/default/gc_thresh2
echo 8192 > /proc/sys/net/ipv4/neigh/default/gc_thresh3

# Protecao contra IP Spoofing
echo 1 > /proc/sys/net/ipv4/conf/all/rp_filter

# Registrando no Log os pacotes com origem errada
echo 1 > /proc/sys/net/ipv4/conf/all/log_martians

# Ativando forwarding de pacotes
echo "1" > /proc/sys/net/ipv4/ip_forward
```

Os três primeiros comandos referem-se a uma configuração do *kernel* para evitar estouro do limite da tabela ARP. O arquivo `/proc/sys/net/ipv4/neigh/default/gc_thresh1` define a quantidade mínima de endereços ARP que devem estar na tabela para que o coletor de lixo (*garbage collector*) seja disparado. Caso a tabela tenha menos endereços que o definido em `gc_thresh1`, o coletor não será executado.

O arquivo `/proc/sys/net/ipv4/neigh/default/gc_thresh2` define a quantidade máxima de endereços ARP que a tabela deverá suportar antes de chamar o coletor de lixo. Caso a tabela tenha mais endereços que o definido em `gc_thresh2`, o coletor recolherá todos os endereços que estiverem lá por mais de 5 segundos.

O arquivo `/proc/sys/net/ipv4/neigh/default/gc_thresh3` define a quantidade máxima de endereços ARP que a tabela deverá suportar como um todo. Caso a tabela tenha mais endereços que o definido em `gc_thresh3`, o coletor será executado imediatamente, removendo as entradas mais antigas e mantendo-a dentro do limite estipulado.

Os dois comandos subsequentes foram utilizados por motivo de segurança. O *kernel* permite filtrar pacotes recebidos por uma interface que, originalmente, não deveria ser usada para enviar pacotes para determinado endereço IP. Esta opção é controlada por meio da variável `rp_filter`. Uma

outra variável relacionada, `log_martians`, indica que deve-se registrar nos *logs* do sistema a chegada de pacotes nessas condições.

2. Carregamento dos módulos necessários. Nesse caso, foram utilizados os seguintes comandos:

```
modprobe iptable_nat
modprobe iptable_filter
modprobe ip_tables
modprobe ip_nat_ftp
modprobe ip_conntrack
modprobe ip_conntrack_ftp
modprobe ipt_REJECT
modprobe ipt_REDIRECT
modprobe ipt_MASQUERADE
modprobe ipt_LOG
```

O comando `modprobe` é utilizado para carregar módulos no Linux. Os módulos acima referem-se à ativação do *Iptables*, de suas tabelas (`iptable_nat` e `iptable_filter`) e de outras opções, como, por exemplo, o registro de *logs* (`ipt_LOG`), o descarte (`ipt_REJECT`), o redirecionamento (`ipt_REDIRECT`) e o mascaramento (`ipt_MASQUERADE`) de pacotes.

3. Definição das políticas padrão e inicialização das regras. Nesse caso, foram utilizados os seguintes comandos:

```
# Chains personalizadas
iptables -N Equipamentos_Bloqueados
iptables -N Controle
iptables -N Regras_de_Saida

# Politicas padrao
iptables -P INPUT DROP
iptables -P OUTPUT DROP
iptables -P FORWARD DROP
iptables -t nat -P PREROUTING ACCEPT
iptables -t nat -P POSTROUTING ACCEPT
iptables -t nat -P OUTPUT ACCEPT
```

```
# Inicializacao das regras
iptables -F INPUT
iptables -F OUTPUT
iptables -F FORWARD
iptables -t nat -F PREROUTING
iptables -t nat -F POSTROUTING
iptables -t nat -F OUTPUT
iptables -t nat -o eth0 -A POSTROUTING -j MASQUERADE
```

Por medida de segurança, a regra estabelecida para o tráfego de pacotes foi a *default deny*. Assim, apenas as portas TCP realmente confiáveis e necessárias são liberadas. Todas as outras portas são bloqueadas por padrão e só são liberadas mediante autorização. Nesse caso, há a necessidade de uma intervenção explícita do administrador para liberar o tráfego desejado, o que minimiza o impacto de eventuais erros de configuração na segurança da rede.

4. Configurações iniciais e portas liberadas do servidor de *firewall*. Nesse caso, foram utilizados os seguintes comandos:

```
# I/O Liberado para loopback
iptables -A INPUT -i lo -j ACCEPT
iptables -A OUTPUT -o lo -j ACCEPT

# Intranet
iptables -A INPUT -i eth1 -m state --state RELATED,ESTABLISHED -j
ACCEPT
iptables -A OUTPUT -o eth1 -j ACCEPT

# Internet
iptables -A INPUT -i eth0 -m state --state RELATED,ESTABLISHED -j
ACCEPT
iptables -A OUTPUT -o eth0 -j ACCEPT
```

```
# Portas liberadas do servidor de firewall
iptables -A INPUT -i eth1 -p tcp --dport 22 -j ACCEPT    # SSH
iptables -A INPUT -i eth1 -p tcp --dport 80 -j ACCEPT    # HTTP
iptables -A INPUT -i eth1 -p tcp --dport 3306 -j ACCEPT  # MySQL
iptables -A INPUT -i eth1 -p tcp --dport 53 -j ACCEPT    # DNS
iptables -A INPUT -i eth1 -p udp --dport 53 -j ACCEPT    # DNS
iptables -A INPUT -i eth1 -p tcp --dport 443 -j ACCEPT   # HTTPS
iptables -A INPUT -i eth1 -p udp --dport 547 -j ACCEPT   # DHCPv6
iptables -A INPUT -i eth1 -p tcp --dport 547 -j ACCEPT   # DHCPv6
iptables -A INPUT -i eth1 -p udp --dport 3130 -j ACCEPT  # Squid
iptables -A INPUT -i eth1 -p tcp --dport 3130 -j ACCEPT  # Squid

# Ping interno
iptables -A INPUT -i eth1 -p icmp -j ACCEPT

# Forwarding de entrada
iptables -A FORWARD -i eth0 -o eth1 -m state --state
RELATED,ESTABLISHED -j ACCEPT
```

Os dois primeiros comandos permitem a entrada e a saída de pacotes da interface de rede local (*loopback*).

Os dois comandos subsequentes (seção denominada `Intranet`) controlam a entrada e a saída de pacotes da interface de rede `eth1`, referente aos pacotes oriundos da rede interna da UFSJ. De forma análoga, os dois próximos comandos (seção denominada `Internet`) controlam a entrada e a saída de pacotes da interface de rede `eth0`, referente aos pacotes oriundos da rede externa à UFSJ.

Em seguida, são executados comandos para liberação de determinadas portas no servidor de *firewall*, de forma que a entrada de pacotes dessas portas seja permitida. As portas liberadas são 22 (SSH), 53 (DNS), 80 (HTTP), 443 (HTTPS), 547 (DHCPv6), 3306 (MySQL) e 3130 (Squid).

Por fim, os dois últimos comandos permitem, respectivamente, o tráfego de pacotes ICMP na interface de rede `eth1` e o encaminhamento de pacotes da rede externa (interface `eth0`) para a rede interna (interface `eth1`) da UFSJ.

5. Configurações de “placas de rede lógicas” e respectivas portas liberadas.

   Na UFSJ existem vários computadores da rede interna que precisam ser acessados externamente. Para tanto, tais computadores precisariam ter números IP válidos. Entretanto, pois questões de segurança e até mesmo de localização física, não é possível alocar números IP válidos diretamente para tais computadores, uma que eles não podem ser conectados diretamente à DMZ.

   Nesse caso, o NTINF utilizou um recurso do próprio Linux, adicionando “placas de rede lógicas” à configuração do sistema e, em seguida, liberando o tráfego apenas de determinadas portas TCP, conforme necessário.

   Para tanto, foi utilizado o seguinte modelo de comandos:

```
# Host 01
ifconfig eth0:1 <ip_valido_host>
route add -host <ip_valido_host> dev eth0:1 gw
<ip_valido_servidor_firewall>

# Portas liberadas para o Host 01
iptables -A PREROUTING -i eth0 -t nat -p tcp -d
<ip_valido_host> --dport <porta> -j DNAT --to
<ip_interno_host>:<porta>

iptables -A FORWARD -i eth0 -o eth1 -p tcp -s 0.0.0.0/0 -d
<ip_interno_host> --dport <porta> -j ACCEPT

iptables -A FORWARD -m state --state ESTABLISHED -i eth1 -o
eth0 -s <ip_interno_host> -j ACCEPT
```

6. Ativação do *proxy*.

   Dada a sua importância para o controle de conteúdos acessíveis na rede da UFSJ, a configuração do *proxy* será apresentada com mais detalhes na seção 4.2.2. O comando utilizado no *firewall* para redirecionar a navegação para o *proxy* foi:

```
# Proxy para a rede interna
iptables -t nat -A PREROUTING -p tcp -s 0.0.0.0/0 -d !
<classe_ip_rede_interna> --dport 80 -j DNAT --to
<ip_interno_servidor_firewall>:3128
```

7. Configuração de demais regras de saída.

   Eventualmente, o NTINF pode fazer liberações de acesso a endereços previamente bloqueados. Para tanto, deve ser configurada uma regra de saída. O modelo de comando utilizado foi:

```
# Regra de saida
iptables -A Regras_de_Saida -i eth1 -o eth0 -p <protocolo> -s
<ip_interno_host> -d <ip_externo_desejado> --dport <porta> -j
Controle
```

8. Controle de acesso por IP e *MacAddress*.

   É regra geral que todo novo computador da UFSJ, para ter acesso à rede, deve passar pelo NTINF. Chegando primeiramente na Área de Manutenção de Equipamentos, o novo computador é cadastrado num sistema de informação (mais detalhes na seção 4.2.4), onde ficam registradas as suas configurações – tipo de processador, quantidade de memória e de HD, o endereço MAC da placa de rede (*MacAddress*), entre outras. É nesse momento que o novo computador recebe um número IP, cuja classe varia de acordo com a sua localização na UFSJ.

   Por questões de segurança, o número IP e o *MacAddress* são utilizados para controlar o acesso à rede da UFSJ. Dessa forma, apenas os computadores previamente cadastrados pelo NTINF terão acesso liberado.

   Como as informações dos computadores são cadastradas num banco de dados MySQL, foi necessário utilizar um *script* PHP para realizar a conexão

ao banco de dados, obter as informações necessárias e escrever, num arquivo *shell script* de saída, as respectivas regras de acesso.

O modelo do arquivo gerado pelo *script* PHP é:

```
#!/bin/bash
iptables -A Controle -m mac --mac-source <mac_address_host> -s
<ip_interno_host> -j ACCEPT
```

Em seguida, o arquivo *shell script* gerado é executado pelo *shell script* do *firewall*. Nesse caso, foram utilizados os seguintes comandos:

```
iptables -F Controle

chmod +x <nome_arquivo_gerado>.sh
/<nome_arquivo_gerado>.sh

iptables -A Controle -j REJECT
```

9. Geração do arquivo de configuração do DHCP.

   Por medida de segurança, números IP válidos são atribuídos apenas aos *hosts* que realmente precisam ser acessíveis "fora" da rede da UFSJ. Nos demais casos, números IP "falsos" (ou privados) são atribuídos aos computadores, através de NAT (*Network Address Translation*).

   Por essa razão, o servidor de *firewall* também provê o serviço de DHCP, uma vez que o sistema deve utilizar o banco de dados de computadores para gerar o script de configuração do DHCP. Dessa forma, a configuração de rede dos computadores clientes não precisa ser modificada manualmente toda vez que ocorrer uma alteração.

De forma análoga ao controle de acesso por IP e *MacAddress*, também foi necessário utilizar um *script* PHP para realizar a conexão ao banco de dados, obter as informações necessárias e escrever o arquivo de configuração do servidor DHCP. Com esse procedimento, o NTINF consegue garantir que apenas computadores cadastrados recebam números IP do servidor DHCP e, consequentemente, consigam acessar a rede da UFSJ.

O modelo do arquivo gerado pelo *script* PHP é:

```
# Arquivo dhcpd.conf
ddns-update-style ad-hoc;
deny unknown-clients;
option domain-name <nome_do_dominio>;
option subnet-mask <mascara_sub_rede>;
authoritative;

subnet <ip_sub_rede> netmask <mascara_sub_rede> {

  host <identificacao_do_host> {
      hardware ethernet <macaddress_host>;
      fixed-address <ip_interno_host>;
      option routers <ip_interno_firewall>;
      option domain-name-servers <ip_servidor_dns_interno>;
  }

}
```

Por fim, o servidor DHCP deve ser reiniciado, para que as configurações atuais entrem em vigor.

4.2.2 *Proxy*

Segundo Morimoto (2008), usar um *proxy* é diferente de simplesmente compartilhar uma conexão diretamente, via NAT. Ao compartilhar via NAT, os

computadores da rede acessam a Internet diretamente, sem quaisquer restrições. Nesse caso, o servidor apenas repassa as requisições recebidas. Já o *proxy* não se limita a repassar as requisições: ele analisa todo o tráfego de dados, separando o que pode ou não pode passar e guardando informações para uso posterior.

Morimoto (2008) afirma que as vantagens de usar um *proxy* são basicamente três:

- É possível impor restrições de acesso com base no horário, *login*, endereço IP da máquina e outras informações, além de bloquear páginas com conteúdo indesejado. É por isso que quase todo *software* de filtro de conteúdo envolvem o uso de algum tipo de *proxy*.
- O *proxy* funciona como um *cache* de páginas e arquivos, armazenando informações já acessadas. Quando alguém acessa uma página que já foi carregada, o *proxy* envia os dados que guardou no *cache*, sem precisar acessar a mesma página repetidamente. Isso acaba economizando bastante banda, tornando o acesso mais rápido. Dependendo da configuração, o *proxy* pode apenas acelerar o acesso às páginas ou servir como um verdadeiro *cache* de arquivos, armazenando atualizações do sistema operacional e/ou *downloads* diversos, por exemplo.
- Uma terceira vantagem de usar um *proxy* é que todos os acessos realizados através dele são registrados em arquivos de *log*.

Por esses motivos, o NTINF optou pela instalação de um serviço de *proxy*, utilizando o *software Squid* e o configurando como *proxy* transparente. Nessa configuração, o *Squid* atua conjuntamente ao *firewall*, de forma que o servidor *proxy* fica "escutando" todas as conexões na porta 80. Nesse caso, o *proxy* intercepta os acessos na porta 80, obrigando todos os pacotes a passar pelas suas regras de controle de acesso, *log*, autenticação e *cache*.

Além disso, mesmo que algum usuário tente desabilitar o *proxy* manualmente nas configurações do navegador, ele continuará sendo usado.

Outra grande vantagem de utilizar o *Squid* como *proxy* transparente é que este recurso permitiu usar o *proxy* na rede da UFSJ sem precisar configurar manualmente o endereço em cada computador. Para tanto, foi necessário apenas usar o endereço IP do servidor rodando o *proxy* como *gateway* da rede (conforme pode ser observado nas configurações do servidor DHCP, na seção 4.2.1).

A Figura 7 ilustra a arquitetura do serviço de *proxy* transparente implementado pelo NTINF.

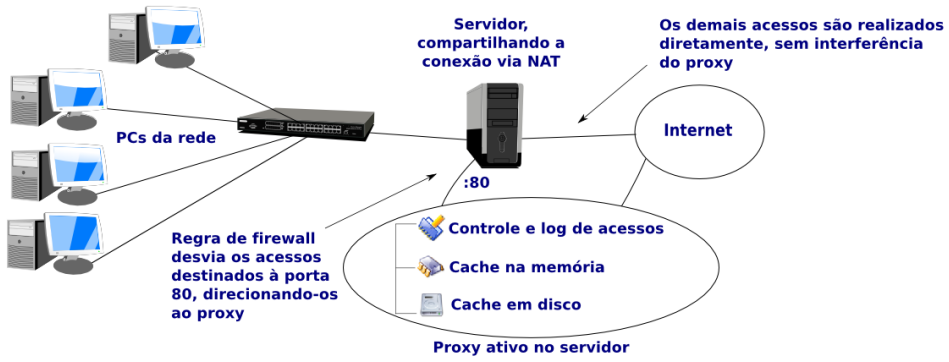

Figura 7: Arquitetura do serviço de *proxy* transparente

4.2.3 Controle de Conteúdo

A UFSJ, através de uma Resolução aprovada por seu Conselho Diretor[2], definiu critérios para o acesso de conteúdos da Internet pela comunidade universitária. De forma geral, os critérios são classificados em três tipos:

[2] Resolução CONDI nº 009, de 25 de novembro de 2009.

- Bloqueio total;
- Bloqueio por horário, liberando o acesso nos horários em que a instituição não mantém atividades regulares;
- Liberação total, sem restrições.

O acesso aos seguintes conteúdos deve ser totalmente liberado:

- *Sites* acadêmicos;
- *Sites* de notícias;
- Serviços de correio eletrônico;
- *Sites* bancários;
- *Sites* de comércio eletrônico;
- *Sites* de postagens (fóruns, *blogs* e *fotologs*, por exemplo);
- *Sites* de serviços públicos.

Nos Laboratórios de Prática de Ensino, os seguintes conteúdos devem possuir bloqueio por horário:

- Serviços de mensagens instantâneas (MSN, *Google Talk* e *Skype*, por exemplo);
- Redes sociais (*Orkut*, *Twitter*, *Facebook*, *Badoo* e *Simi*, por exemplo);
- *Sites* de vídeos (*Youtube*, *Myspace*, *Yahoo Vídeos* e *Google Vídeos*, por exemplo);
- *Sites* de *downloads* (*Baixaki*, *Superdownloads* e *Tucows*, por exemplo).

Nos Laboratórios de Ensino, os seguintes conteúdos devem possuir bloqueio total:

- Serviços de mensagens instantâneas (MSN, *Google Talk* e *Skype*, por exemplo);
- Redes sociais (*Orkut*, *Twitter*, *Facebook*, *Badoo* e *Simi*, por exemplo);
- *Sites* de vídeos (*Youtube*, *Myspace*, *Yahoo Vídeos* e *Google Vídeos*, por exemplo);
- *Sites* de *downloads* (*Baixaki*, *Superdownloads* e *Tucows*, por exemplo).

Além disso, os seguintes conteúdos devem possuir bloqueio total, em toda a rede da UFSJ:

- Serviços de compartilhamento de arquivos (*BitTorrent*, *Emule* e *Kazaa*, por exemplo);
- *Sites* de conteúdo adulto e pornografia;
- Jogos;
- *Software* malicioso (*trojans* e *worms*, por exemplo);
- Anexos executáveis de *e-mails*.

Devido à definição desses critérios de acesso e procurando estar em sintonia com a Política de Segurança determinada pela UFSJ, o NTINF, a partir de novembro de 2009, precisou implantar uma forma de controle de conteúdo à solução de segurança.

Bloquear domínios e endereços IP individuais utilizando o *Squid* até poderia ser um caminho a ser tomado pelo NTINF, a fim de bloquear páginas específicas. Entretanto, no caso de páginas conteúdo adulto e pornografia, por exemplo, seria inviável tentar bloqueá-las manualmente, dado o número expressivo de páginas dessa natureza existentes na Internet.

Por outro lado, existem grupos destinados a manter listas com URLs de páginas pornográficas, páginas de cassinos e jogos e páginas ilícitas em geral, que são atualizadas freqüentemente. Por serem construídas através da combinação dos esforços de muitas pessoas, auxiliadas por ferramentas semi-automáticas de indexação e classificação de conteúdo, estas listas permitem bloquear a maior parte das páginas ilícitas sem muito esforço.

Morimoto (2008) afirma que estas listas nada mais são do que longas listas de *links*, com um por linha. Elas até poderiam ser usadas diretamente no *Squid*, através da opção **url_regex**, mas, por serem arquivos muito grandes, o desempenho seria ruim, já que o *Squid* processa cada linha dos arquivos a cada acesso, o que consome muito processamento.

Nesse contexto, o NTINF optou pela utilização do *software SquidGuard*[3], que, trabalhando conjuntamente ao *Squid*, permite usar longas listas de URLs, com milhões de *links*, sem comprometer o desempenho do servidor *proxy*. Por meio dessas listas, o *SquidGuard* se encarrega de bloquear a maior parte das páginas impróprias e é possível fazer ajustes manuais, conforme necessário.

De forma geral, o *script* do *SquidGuard* da UFSJ está organizado da seguinte forma:

1. Configuração dos diretórios das listas e de *logs*. Nesse caso, foram utilizados os seguintes comandos:

```
dbhome <diretorio_das_listas_do_squidguard>
logdir <diretorio_de_logs_do_squidguard>
```

[3] *Site* oficial: http://www.squidguard.org

2. Definição dos *Source Addresses*, ou seja, dos grupos de computadores cuja navegação deve ser controlada pelo *SquidGuard*. Para criar um *Source Address*, foi utilizado o seguinte modelo de comandos:

```
src <nome_do_grupo> {

   ip <numero_IP_01> <numero_IP_02> <numero_IP_03>
<faixa_de_IP_inicio>-<faixa_de_IP_termino>

}
```

3. Definição das *Destination Classes*, ou seja, das listas cujo conteúdo será controlado pelo *SquidGuard*. Para criar uma *Destination Class*, foi utilizado o seguinte modelo de comandos:

```
dest <nome_da_classe> {
    domainlist    <diretorio_da_classe>/domains
    urllist       <diretorio_da_classe>/urls
}
```

No caso da UFSJ, as *Destination Classes* utilizadas foram *games*, *gambling*, *onlinegames*, *porn*, *spyware*, *virusinfected*, *proxy*, *hacking*, *dialers*, *malware*, *phising*, *redirector*, *warez*, *instantmessaging* e *socialnetworking*.

4. Definição das Políticas de Controle.

Na seção final do arquivo de configuração, deve-se relacionar, para cada *Source Address* definido anteriormente, quais *Destination Classes* serão utilizadas, e se o conteúdo da referida *Class* deve ser bloqueado ou permitido pelo *SquidGuard*.

Para a UFSJ foram definidos três tipos de políticas: uma para os administradores do sistema, cuja navegação é totalmente liberada (sem quaisquer restrições); outra para determinados computadores da rede, cuja navegação deve ser controlada segundo as normas pré-estabelecidas; e outra para os demais computadores da rede. Para tanto, foram utilizados os seguintes modelos de comandos:

```
acl {
    admin {
        pass all
    }

    <nome_do_grupo_01> {
        pass !games !gambling !onlinegames !porn !spyware
!virusinfected !proxy !hacking !dialers !malware !phishing
!redirector !warez all
    }

    <nome_do_grupo_02> {
        pass !games !gambling !onlinegames !porn !spyware
!virusinfected !proxy !hacking !dialers !malware !phishing
!redirector !warez !instantmessaging !socialnetworking all
    }

    default {
        pass none
        redirect http://www.ufsj.edu.br/
acesso_negado.php?url=%u
    }
}
```

A opção **redirect** faz com que todos os acessos bloqueados sejam redirecionados de forma transparente à URL especificada. Dessa forma, o usuário tentando acessar páginas impróprias é sutilmente direcionado a uma página com conteúdo mais amigável.

A Figura 8 apresenta um exemplo de página informativa utilizada na UFSJ.

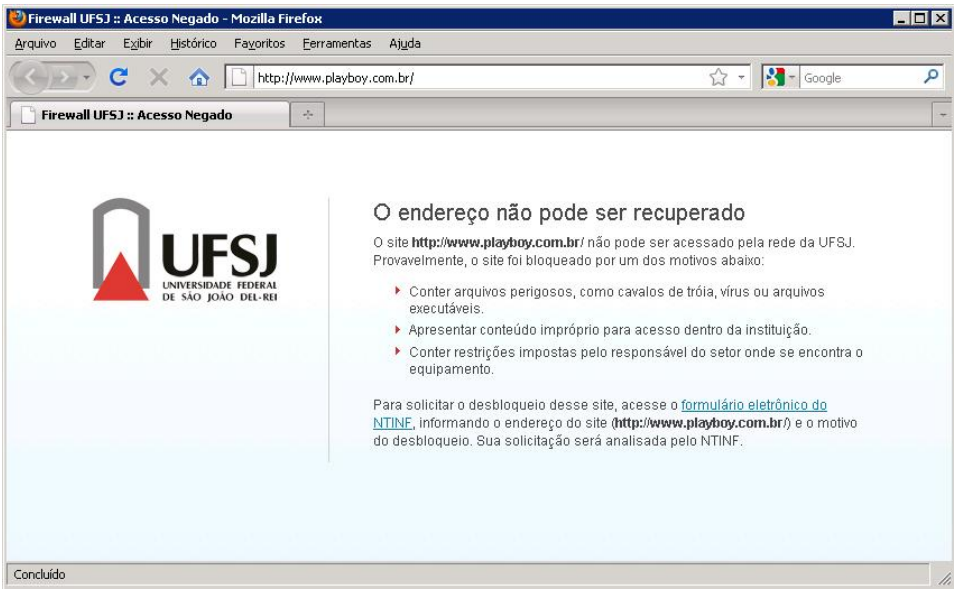

Figura 8: Página informativa de conteúdo bloqueado na UFSJ

É importante ressaltar também que, antes que possam ser efetivamente utilizadas, as listas precisam ser convertidas para o formato Berkeley DB, que permite um acesso muito mais rápido do que seria possível ao manipular diretamente os arquivos em texto (Morimoto, 2008).

Para tanto, os seguintes comandos foram utilizados:

```
# squidGuard -C all

# chown -R <usuario_do_squid>:<grupo_do_squid>
<diretorio_das_listas_do_squidguard>/*

# find <diretorio_das_listas_do_squidguard> -type f | xargs
chmod 644

# find <diretorio_das_listas_do_squidguard> -type d | xargs
chmod 755
```

O primeiro comando deve ser executado sempre que forem incluídas novas listas na configuração.

O segundo comando permite ajustar as permissões de acesso aos arquivos, garantindo que o *Squid* tenha acesso a eles.

Os dois últimos comandos complementam a configuração, fazendo com que todos os arquivos das listas do *SquidGuard* sejam configurados com permissões 644, e as respectivas pastas com permissões 755. Isso previne o aparecimento de diversos erros relacionados a permissões incorretas para os arquivos.

Depois de gerar a configuração do *SquidGuard*, o próximo passo foi alterar a configuração do *Squid*, para informar que o SquidGuard será utilizado conjuntamente. Para tanto, as seguintes linhas de comando foram acrescentadas ao arquivo de configuração do *Squid*:

```
redirect_program <caminho_do_squidguard>
redirect_children 64
redirector_bypass off
```

A opção **redirect_children** ajusta o número de processos do *SquidGuard* que o *Squid* manterá abertos. Segundo Morimoto (2008), aumentar o número ajuda a melhorar o desempenho do *proxy* em grandes redes, onde o servidor recebe um volume muito grande de requisições.

A opção **redirector_bypass off** não permite que o *Squid* continue funcionando quando o *SquidGuard* travar ou deixar de funcionar por qualquer motivo.

Por fim, o *Squid* deve ser reiniciado, para que as configurações atuais entrem em vigor.

4.2.4 Sistema de Gerenciamento do *Firewall* via *Web*

É regra geral que todo novo computador da UFSJ, para ter acesso à rede, deve passar pelo NTINF. Chegando primeiramente na Área de Manutenção de Equipamentos, o novo computador deve ser cadastrado e receber um número IP.

A fim de propiciar uma gestão adequada e eficiente pelo SETIR, foi desenvolvido um sistema de gerenciamento do *firewall* via *web*. Denominado *Firewall Manager*, o sistema foi construído na linguagem PHP[4], utilizando o MySQL Server[5] como SGBD.

A Figura 9 apresenta a tela inicial do *Firewall Manager*.

Figura 9: Tela inicial do *Firewall Manager*

[4] *Site* oficial: http://www.php.net
[5] *Site* oficial: http://www.mysql.com

As opções do *Firewall Manager* incluem o cadastramento e o bloqueio/desbloqueio de equipamentos, além da geração de relatórios de equipamentos por unidade organizacional e de equipamentos bloqueados.

Para o cadastramento de um novo equipamento, ilustrado na Figura 10, o primeiro passo é informar o número de patrimônio. Em seguida, devem ser digitados o endereço IP e o *MacAddress* do equipamento. Para cada uma dessas informações o sistema faz uma verificação prévia, não permitindo o cadastro de números de patrimônio, IPs ou *MacAddress* de forma repetida.

Figura 10: Cadastramento de equipamento no *Firewall Manager* (parte 1)

Em seguida, conforme ilustrado na Figura 11, devem ser informados os dados complementares, tais como a descrição do equipamento, a unidade

organizacional responsável, a unidade organizacional onde o equipamento está localizado, a localização (*campus*) e a descrição da localização.

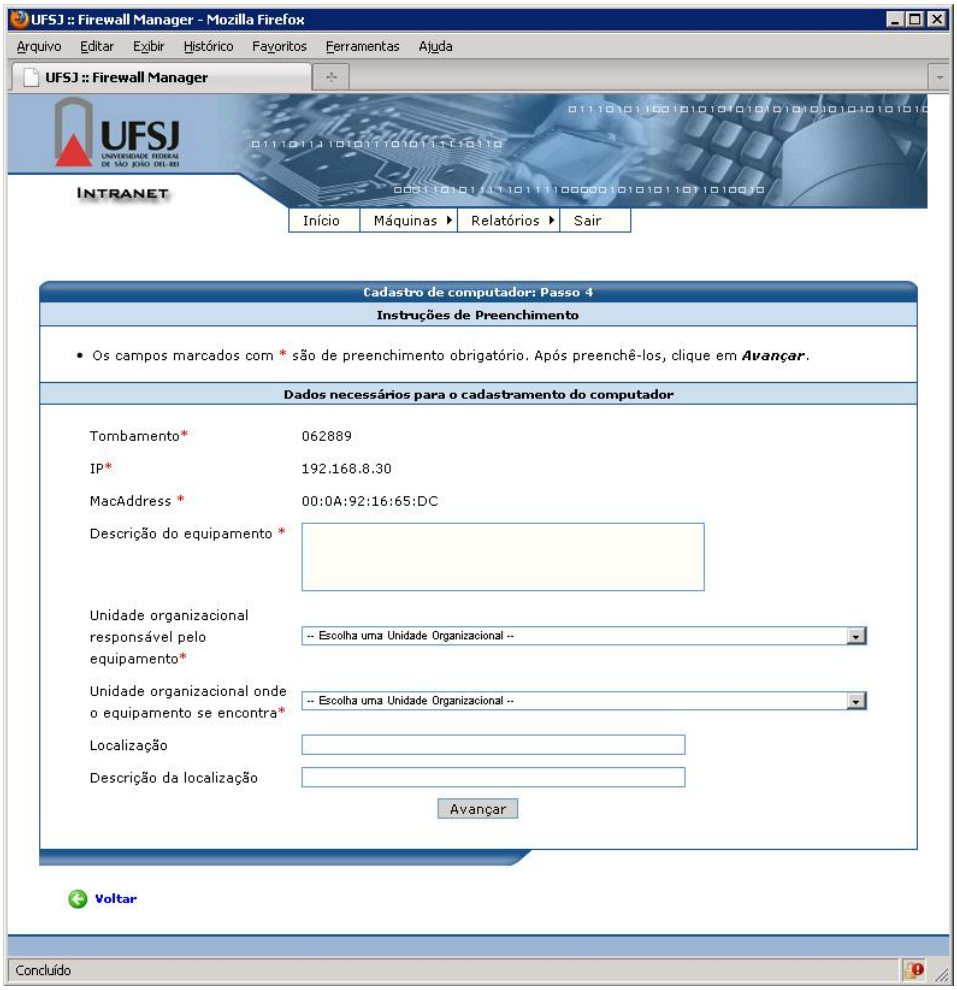

Figura 11: Cadastramento de equipamento no *Firewall Manager* (parte 2)

É importante ressaltar que o cadastramento de um equipamento é feito de forma dinâmica. Isto é, o *Firewall Manager* interage diretamente com o *Iptables* e com o DHCP, escrevendo novas regras para que o equipamento possa ter

acesso à rede imediatamente. Para tanto, é utilizado o comando **–I** do *Iptables* (mais detalhes na seção 3.5.4) e o arquivo de configuração do DHCP é gerado novamente via *script* (mais detalhes na seção 4.2.1).

O bloqueio/desbloqueio de equipamentos é feito de forma bastante simples. Após localizar o equipamento pelo número de patrimônio, ou pelo endereço IP, ou pelo *MacAddress*, basta um clique para bloquear/desbloquear um equipamento. A Figura 12 apresenta um exemplo desse procedimento.

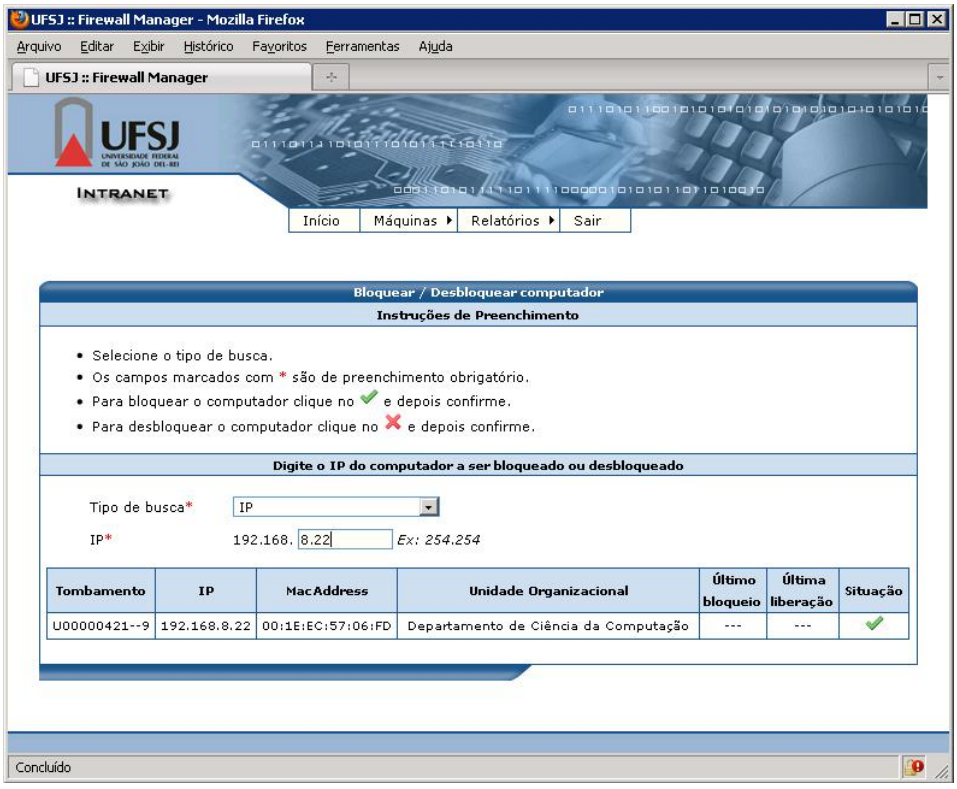

Figura 12: Bloqueio de equipamento no *Firewall Manager*

De forma análoga ao cadastramento, o bloqueio/desbloqueio de um equipamento é feito de forma dinâmica. Para bloquear um equipamento, é

utilizado o comando **–D** do *Iptables*, enquanto que para desbloquear é utilizado o comando **–I** (mais detalhes na seção 3.5.4). Em ambos os casos, o arquivo de configuração do DHCP também é gerado novamente via *script* (mais detalhes na seção 4.2.1).

Por fim, as Figuras 13 e 14 apresentam exemplos de relatórios gerados pelo *Firewall Manager*.

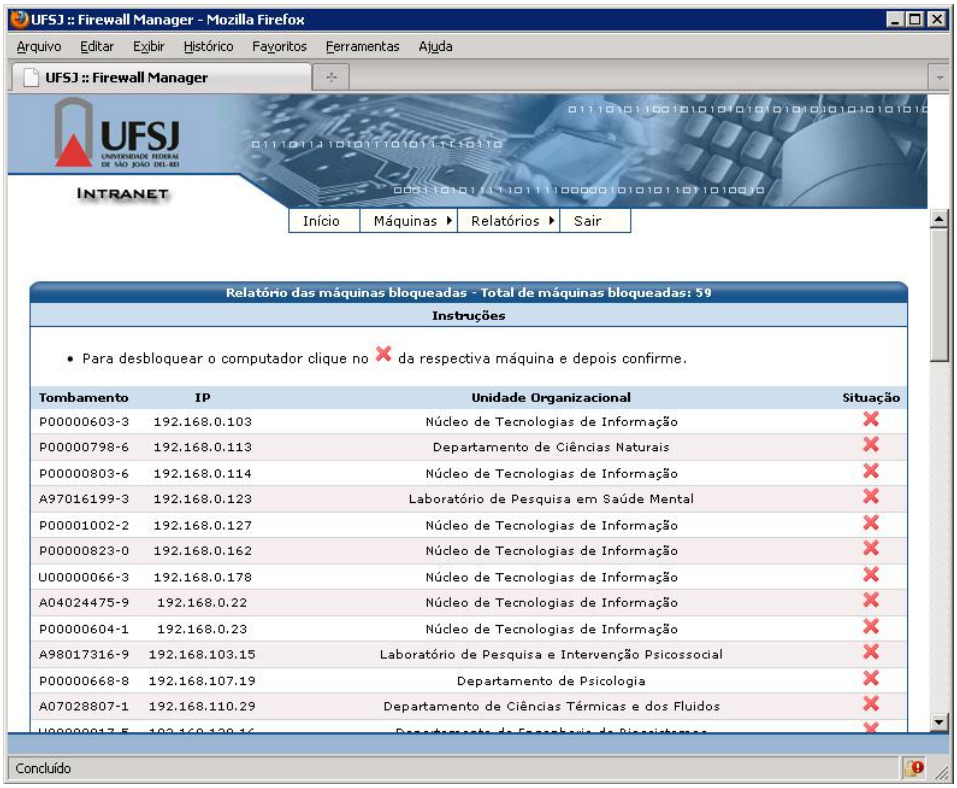

Figura 13: Relatório de Equipamentos Bloqueados

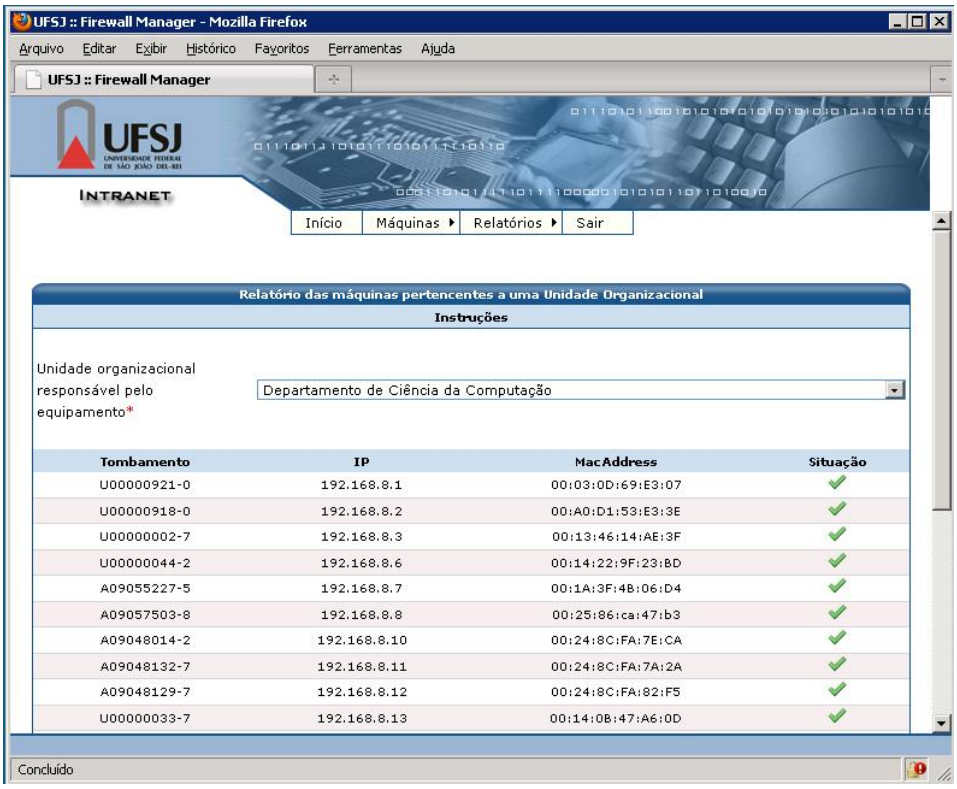

Figura 14: Relatório de Equipamentos por Unidade Organizacional

4.3 Resultados e Discussão

O projeto de Segurança da Informação ora apresentado tinha como objetivo principal reestruturar a segurança lógica da rede da UFSJ. Nesse sentido, é possível perceber claramente que esse objetivo foi alcançado.

Por meio da instalação de um servidor de *firewall*, todo o tráfego da rede da UFSJ passou a ser controlado. Atualmente, o SETIR possui *logs* de toda a navegação dos usuários da rede, o que permite verificar comportamentos indesejados, como tentativas de propagações de vírus, ou ataques à rede, por exemplo.

Considerando a quantidade crescente de usuários da rede da UFSJ, outra preocupação do NTINF era garantir uma velocidade de navegação adequada. Nesse sentido a instalação de um serviço de *proxy* trouxe vários benefícios.

O primeiro deles foi a construção de um *cache* de páginas e arquivos, armazenando informações já acessadas. Quando algum usuário da UFSJ acessa uma página que já foi carregada, o *proxy* envia os dados que guardou no *cache*, sem precisar acessar a mesma página repetidamente.

Outro benefício importante foi a possibilidade de atender às definições da Política de Controle de Conteúdo da UFSJ. Seguindo essa política, foram definidas restrições de acesso com base nos endereços IP das máquinas e, principalmente, implantou-se o bloqueio de páginas com conteúdo impróprio e/ou prejudicial à rede.

Outra vantagem do serviço de *proxy* está no fato que todos os acessos realizados através dele na rede UFSJ são registrados em arquivos de *log*. Com isso, assim como no *firewall*, é possível verificar, entre outras informações, o acesso a conteúdos que deveriam estar bloqueados e, por algum motivo, não foram restringidos adequadamente.

No entanto, convém ressaltar que tais verificações ainda são feitas de forma muito incipiente pelo SETIR. Muitas vezes são utilizados comandos para varredura dos *logs*, na busca de padrões que caracterizam determinados comportamentos. Ou seja, essa gerência não é realizada de forma automatizada, tampouco é mantido um histórico de verificações.

Nesse sentido, existem vários aplicativos que poderiam auxiliar sobremaneira esse trabalho de gerência. Por exemplo, poderia ser utilizado o *software* SARG *(Squid Analysis Report Generator)*[6], que é um interpretador de *logs* para o *Squid*.

[6] *Site* oficial: http://sarg.sourceforge.net

Sempre que executado, o SARG cria um conjunto de páginas, divididas por dia, com uma lista de todas as páginas que foram acessadas e a partir de que máquina da rede veio cada acesso. Os acessos são organizados por IP, mostrando as páginas acessadas por cada um, a quantidade de dados transmitidos, o tempo gasto em cada acesso, as tentativas de acesso bloqueadas pelos filtros de conteúdo e outras informações. Além disso, os relatórios do SARG são inicialmente organizados por período, sendo que os mais antigos são mantidos.

Ainda com relação à gerência feita pelo SETIR, é importante destacar que o desenvolvimento de um sistema de gerenciamento do *firewall* (*Firewall Manager*) contribuiu bastante para a eficiência dos trabalhos cotidianos. O cadastramento de novos computadores na rede é feito de forma automática, uma vez que o sistema interage diretamente com o *Iptables* e o DHCP, escrevendo regras e arquivos de configuração de forma dinâmica. De forma análoga, o bloqueio de computadores também pode ser feito em tempo de execução, interrompendo o acesso de máquinas comprometidas e, consequentemente, garantindo a integridade da rede.

Apesar disso, o *Firewall Manager* também pode ser aprimorado. Considerando que a UFSJ dispõe de um sistema próprio de registro de patrimônio, o *Firewall Manager* poderia interagir com esse sistema. Por meio do número de patrimônio, as informações dos equipamentos poderiam ser obtidas diretamente, sem a necessidade de novos cadastramentos, evitando redundâncias.

Outro aprimoramento importante seria a distribuição automática de endereços IP. Como os IPs na rede da UFSJ são divididos em classes específicas, de acordo com os setores da instituição, o *Firewall Manager* poderia, no momento do cadastramento, atribuir um IP automaticamente a um determinado equipamento, a partir do referido setor responsável. Dessa forma, a

distribuição de IPs seria feita de forma racional e a usabilidade do *Firewall Manager* melhoraria consideravelmente.

Por fim, convém destacar uma questão importante: a forma de autenticação dos computadores na rede da UFSJ. A utilização do número IP e do *MacAddress* para controlar o acesso à rede, apesar de possuir suas vantagens, não restringe a possibilidade de um *MacAddress* ser clonado. Nesse sentido, a fim de evitar esse problema, uma alternativa interessante seria modificar o tipo de autenticação, passando a verificar os usuários da rede, e não os computadores.

5. Considerações Finais

O presente trabalho mostrou a importância de um *software* de *firewall* para uma organização. Através de um estudo de caso na Universidade Federal de São João del-Rei (UFSJ), foi apresentada uma aplicação prática de um *software* de *firewall*, instalado em um servidor Linux, enfatizando-se os aspectos que a nortearam a implementação dessa ferramenta e, principalmente, como ela contribuiu para aumentar a segurança da informação nessa instituição.

A expansão pela qual a UFSJ vem passando impactou sobremaneira o segmento administrativo, em especial na área de informática, que necessitou acompanhar tal crescimento. Nesse cenário de expansão, em 2008, o Núcleo de Tecnologia da Informação (NTINF) começou a implementar um Projeto de Segurança da Informação, particularmente voltado à reestruturação da segurança lógica da rede da UFSJ.

Além de apresentar os benefícios que a UFSJ obteve com esse projeto, o presente trabalho também discutiu alguns pontos a serem aprimorados, apontando sugestões de sistemas e de procedimentos para a melhoria da gerência realizada pelo NTINF.

Por fim, como trabalho futuro, convém destacar que o NTINF pretende expandir esse projeto para os outros *campi* da UFSJ, localizados fora de São João del-Rei[7]. Com isso, além de promover a adoção da política de segurança da informação em toda a universidade, o NTINF terá condições de realizar, de fato, uma gerência ampla e eficiente na UFSJ.

[7] Atualmente, a UFSJ possui três *campi* localizados fora de São João del-Rei: *Campus* Alto Paraopeba, em Ouro Branco/MG; *Campus* Centro-Oeste Dona Lindu, em Divinópolis/MG e *Campus* Sete Lagoas, em Sete Lagoas/MG.

6. Referências Bibliográficas

ALVES, Gustavo A. **Segurança da Informação: uma visão inovadora da gestão**. Rio de Janeiro: Ciência Moderna Ltda., 2006. 115 p.

ANDREASSON, Oskar. **Iptables Tutorial**. 2006. Disponível em: http://www.frozentux.net/iptables-tutorial/iptables-tutorial.html. Acesso em: 10 fev. 2011.

CGI.BR – Comitê Gestor da Internet no Brasil. **Cartilha de Segurança para Internet – Parte VIII: Códigos Maliciosos (*Malware*)**. Versão 3.1, 2006. Disponível em: http://cartilha.cert.br/download/cartilha-08-malware.pdf. Acesso em: 03 ago. 2010.

CHESWICK, William R.; BELLOVIN, Steven M.; RUBIN, Aviel D. ***Firewall* e Segurança na Internet**. 2ª ed. Tradução: Edson Frumankiewicz. São Paulo: Bookman, 2003. 400 p.

DIAS, Cláudia. **Segurança e Auditoria da Tecnologia da Informação**. Rio de Janeiro: Axcel Books do Brasil, 2000. 218 p.

FONTES, Edison. **Segurança da Informação: o usuário faz a diferença**. 1ª ed. São Paulo: Saraiva, 2006. 172 p.

MORIMOTO, Carlos E. **Servidores Linux: Guia Prático**. 2ª Ed. São Paulo: GDH Press, 2008. 736 p.

NBSO. **Práticas de Segurança para Administradores de Redes Internet**. São Paulo, 2003. Disponível em: http://www.cert.br/docs/seg-adm-redes/seg-adm-redes.html. Acesso em: 21 set. 2010.

NETO, Urubatan. **Dominando Linux: *Firewall Iptables***. Rio de Janeiro: Ciência Moderna, 2004. 112 p.

PROMON – Business & Tecnology Review. **Segurança da Informação: um diferencial determinante na competitividade das corporações**. Rio de Janeiro, 2005. Disponível em: http://www.promon.com.br/portugues/noticias/download/Seguranca_4Web.pdf. Acesso em: 17 jun. 2010.

RIBEIRO, Uirá. **Certificação Linux**. Rio de Janeiro: Axcel Books do Brasil, 2004. 472 p.

ROGER, Denny. ***Firewalls*: falsa sensação de segurança**. Fortaleza, 2005. Disponível em: http://web.archive.org/web/20070309204446/ www.secforum.com.br/article.php?sid=2758&mode=thread&order=0. Acesso em: 22 set. 2010.

SÊMOLA, Marcos. **Gestão da Segurança da Informação: uma visão executiva**. Rio de Janeiro: Campus Elsevier, 2003. 160 p.

SILVA, Pedro T.; CARVALHO, Hugo; TORRES, Catarina B. **Segurança dos Sistemas de Informação: gestão estratégica da segurança empresarial**. 1ª ed. Portugal: Centro Atlântico, 2003. 256 p.

SILVA FILHO, Antônio Mendes da. **Segurança da Informação: sobre a necessidade de proteção de sistemas de informações**. 2008. Disponível em: http://www.espacoacademico.com.br/042/42amsf.htm. Acesso em: 15 mai. 2010.

SUZUKI, Luiz Henrique. **Implantação de um *Honeypot* e proposta para metodologia de Gerenciamento de Projetos de *Honeynets***. Brasília: UnB, 2007. 80 p.