UNIVERSIDADE ESTADUAL DO NORTE DO PARANÁ

CAMPUS LUIZ MENEGHEL

MICAEL MERCADANTE

CONFIGURAÇÃO DE *FIREWALL* PERSONALIZÁVEL EM SERVIDOR UBUNTU ATRAVÉS DE INTERFACE WEB

Bandeirantes

2013

MICAEL MERCADANTE

CONFIGURAÇÃO DE *FIREWALL* PERSONALIZÁVEL EM SERVIDOR UBUNTU ATRAVÉS DE INTERFACE WEB

Trabalho de conclusão de curso apresentado a Universidade Estadual do Norte do Paraná – *campus* Luiz Meneghel, como requisito parcial para a obtenção do grau e Bacharel/Licenciatura em Sistema de Informação.

Orientador: Prof. Me. Luiz Fernando Legore do Nascimento.

Bandeirantes
2013

MICAEL MERCADANTE

CONFIGURAÇÃO DE *FIREWALL* PERSONALIZÁVEL EM SERVIDOR UBUNTU ATRAVÉS DE INTERFACE WEB

Trabalho de conclusão de curso apresentado a Universidade Estadual do Norte do Paraná – Campus Luiz Meneghel – como requisito parcial para a obtenção de grau de Bacharelado/Licenciatura em SISTEMAS DE INFORMAÇÃO com nota final igual a _____, conferida pela Banca Avaliadora formada pelos professores.

Me. Luiz Fernando Legore do Nascimento
Universidade Estadual do Norte do Paraná – UENP

Me. Ricardo Gonçalves Coelho
Universidade Estadual do Norte do Paraná – UENP

Me. Neimar Neitzel
Universidade Estadual do Norte do Paraná – UENP

Bandeirantes, ___ de_____ de 2013.

Dedico esse trabalho a minha família que me apoiou em todos os momentos e difíceis da minha vida.

AGRADECIMENTOS

Agradeço primeiramente a Deus por ter me indicado os caminhos a seguirem, e me dar saúde e força para chegar até aqui.

Especialmente aos meus pais Zaqueu Mercadante e Marlene Pereira Mercadante por acreditar, apoiar e financiar o meu sonho de cursar uma universidade.

Agradeço também a XVII turma, pela união do grupo, e auxílio uns aos outros quando necessário, possibilitando o melhor entendimento e aprendizagem durante o curso.

Aos professores do curso de Sistemas de Informação, que não mediram esforços para me passar os seus conhecimentos. Em especial ao eu orientador Me. Luiz Fernando Legore do Nascimento que além de professor orientador foi, companheiro de trabalho, conselheiro e amigo.

RESUMO

Em pequenas empresas onde geralmente não há um profissional ou técnico com grande conhecimento para se administrar as pequenas redes nela existentes, o uso de ferramentas simples pode ser de grande ajuda. Nesse caso, os recursos financeiros voltados a essa área também são escassos. Assim, em geral ao invés do uso de *software* proprietários, pequenas empresas passam a optar pelo uso de *software* livres, os quais reduzem os custos de licenças, porém aumentam a necessidade de profissionais com maior conhecimento. Observando a dificuldade desses profissionais em configurar regras de *Firewall* através do IPTABLES, e da necessidade em se agregar outras aplicações pertinentes ao processo de gerenciamento de redes, esse trabalho apresenta um *Firewall* utilizando *software* livre, por meio de uma *interface Web.* Esse *Firewall* baseia-se em instruções do próprio sistema operacional Ubuntu(*shell script*), executadas por meio de uma interface gráfica amigável e personalizável para qualquer empresa de pequeno porte. Embora já existam alguns *software* dessa natureza, neste trabalho propõe-se um diferencial dos softwares já existentes. Nesse é apresentado regras de bloqueio utilizando NAT (Tradução de Endereço de Rede) por portas de comunicação. Além disso o *Firewall* é construído para o sistema operacional Ubuntu Server, gráficos sobre o uso da banda de rede interna e externa.

Palavras-chave: *Firewall*, Iptables, Segurança da Informação, Gerência de Redes.

ABSTRACT

In small companies where usually there is not a professional or technician with great knowledge to manage these small networks, the use of simple tools can be of great help. In this case, the financial resources directed to this area are scarce. So in general rather than the use of proprietary software, small companies now choose to use free software, which reduce license costs, but increase the need for professionals with greater knowledge. Noting the difficulty these professionals to configure *Firewall* rules through IPTABLES, and the need to aggregate other applications relevant to the process of network management, this paper presents a *Firewall* using free software, through a Web interface. This *Firewall* is based on instructions of the operating system (shell script), run by a friendly graphical interface and customizable for any small business. Although there are some software of this nature, this paper proposes a differential of existing software. This is shown blocking rules using NAT (Network Address Translation) for communication ports. Also the *Firewall* is built into the operating system Ubuntu Server, graphs on bandwidth usage for internal and external network.

Keywords: *Firewall*, Iptables, Information Security, Network Management.

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE TABELAS

LISTA DE SIGLAS

ACL - *Access Control List* (Lista de Controle de Acessos)

ADSL - *Asymmetric Digital Subscriber Line*(Linha de Assinante Digital Assimétrica)

DMZ - *DeMilitarized Zone* (Zona Desmilitarizada)

DNAT - *Destination Network Address Translation* (Destino de Tradução de Endereços de Rede)

DHCP - *Dynamic Host Configuration Protocol* (Protocolo de configuração dinâmica de host)

FTP - *File Transfer Protocol* (Protocolo de Transferência de Arquivo)

HTML - *Hypertext Markup Language*(Linguagem de Marcação de Hipertexto)

HTTP - *Hypertext Transfer Protocol*(Protocolo de Transferência de Hipertexto)

HTTPS - *HyperText Transfer Protocol Secure*(Protocolo Seguro de Transferência de Hipertexto)

ICMP - Internet Control Message Protocol(Protocolo de Controle de Mensagens da *Internet*)

IDS - *Intrusion Detection System* (Sistema de Detecção de Invasão)

IP - *Internet Protocol* (Protocolo de *Internet*)

IPS - *Intrusion Prevention System* (Sistema de Prevenção de Invasão)

ISO - *International Standards Organization*(Organização dos Padrões Internacionais)

LAN - *Local Area Network* (Rede de Área Local)

MAC - *Machine Address Code* (Código de Endereço de Máquina)

FMUS - *Firewall Manager Ubuntu Server* (Gerenciador de *Firewall* Ubuntu Server)

NAT - *Network Address Translation* (Tradutor de Endereços de Rede)

OSI - *Open Systems Interconnection* (Sistema Aberto de Interconexão)

PHP - *Hypertext Preprocessor* (Processador de Hypertexto)

PPPoE - *Point-to-Point Protocol over Ethernet*(Protocolo Sobre a *Internet*)

SMTP - *Simple Mail Transfer Protocol* (Protocolo de transferência de correio simples)

SNMP - *Simple Network Management Protocol* (Protocolo Simples de Gerência de Rede)

SNAT - *Secure Network Address Translation*(Tradutor de Endereços de Rede Seguro)

SO - Sistema Operacional

TCP/IP - *Transmission Control Protocol/Internet Protocol*(Protocolo de Controle de Transmissão/Protocolo de *Internet*)

TI - Tecnologia da Informação

UDP - *User Datagram Protocol* (Protocolo de Datagrama de Usuário)

SUMÁRIO

1 INTRODUÇÃO

O gerenciamento de uma rede está associado ao controle das atividades e ao monitoramento do uso dos recursos no ambiente da rede. As tarefas básicas desta gerência, resumidamente, são: obter as informações da rede, diagnosticar possíveis problemas e encaminhar para as soluções destes problemas. (PINHEIRO, 2002)

Para cumprir estes objetivos, funções de gerência devem ser embutidas nos diversos componentes da rede, possibilitando detectar, prever e reagir aos problemas que possam ocorrer.

Considerando que as ferramentas para gerência de redes não são usadas por todas organizações e nem abrangem toda a gama de problemas delas, se faz necessário que outros mecanismos de gerência sejam utilizados para suprir suas carências mais evidentes. As informações que circulam em uma rede de computadores devem ser transportadas de modo confiável e rápido. Para que isso aconteça é importante que os dados sejam monitorados de maneira que os problemas existentes, sejam detectados rapidamente e solucionados de forma eficiente. (PINHEIRO, 2002).

Embora que segundo Sêmola (2003) afirme que em uma rede há a necessidade de elementos que garantam a confidencialidade, a integridade, e a disponibilidade de toda informação gerada, nem sempre isso é possível ser observado em todos os locais. Muitas empresas, ao diminuir custos, deixando de usar software proprietário também acabam defasando a segurança das informações. Para isso software de natureza free(gratuito) e ou open source (código aberto) acabam sendo escolhidos e bem aceitos nesses casos.

Contudo, acarretam a necessidade de se utilizar profissionais ainda mais qualificados para cumprirem a dura tarefa de configurá-los, isto é, mesmo utilizando um software livre é preciso que um administrador de redes esteja preparado para implementá-lo, configurando um *Firewall*, que resume-se a um dispositivo da rede de computadores com o objetivo aplicar uma política de segurança à organização. Logo, não é qualquer pessoa com conhecimentos básicos em TI que poderão realizar o serviço. Além disso, pessoas não

autorizadas ou com pouco conhecimento poderão comprometer todas as informações de uma empresa.

Santos e Silva (2013), afirmam que o *Firewall* mais comum e utilizado em um ambiente de software livre é o IPTABLES, essa ferramenta é configurada através de regras utilizando scripts, isto é, instruções em formato de codificação. Cabe ao administrador um excelente conhecimento para ajustá-la de forma coerente as necessidades de sua empresa.

Nesse trabalho é desenvolvido uma ferramenta gráfica em formato Web, para a configuração de *Firewall*, a qual utiliza regras pré-determinadas e ajustáveis ao perfil de cada empresa. Com isso, pretende-se manter um ambiente seguro, de forma mais fácil e rápido. A interface Web para configuração das regras de *Firewall*, que se apresenta, utiliza a linguagem de programação PHP(Editor de Hypertexto), executando as regras editadas em Shell script. Através das instruções básicas do próprio sistema operacional, é possível configurar um *Firewall* de forma mais amigável. Além disso, a ferramenta disponibiliza relatórios sobre o consumo de banda, bem como relatórios de acessos sobre as tentativas de invasão via acesso remoto.

1.1 OBJETIVOS

O ambiente escolhido para essa implementação é o Linux, em uma distribuição Ubuntu Server. Neste sistema operacional são implantados os aplicativos necessários para o uso do *Firewall* IPTABLES onde são efetuados bloqueios à *sites* indesejados, controle de portas de comunicação externas e internas.

1.1.1 *Objetivos Gerais*

Objetiva-se a implementação de uma ferramenta de *Firewall* baseado em GNU/Linux, utilizando regras em IPTABLES, as quais possam ser facilmente personalizadas por administradores de redes com pouco conhecimento em *Shell script.* Como mecanismo facilitador, uma *interface Web* é desenvolvida em PHP*,* a qual permite manipular alguns dos principais aplicativos para o gerenciamento de pequenas e médias redes corporativas. Entre os principais aplicativos, destacam-se as regras de *NAT, Firewall e Proxy.*

1.1.2 *Objetivos Específicos*

- Analisar e comparar alguns dos principais *Firewall* personalizáveis os quais sejam livres e ou de código aberto;
- Apresentar um diferencial entre os *Firewall analisados;*
- Descrever alguns dos principais meios de proteção e filtragem em uma rede;
- Aplicar as ferramentas de segurança no sistema operacional *Ubuntu Server;*
- Produzir relatórios de acessos através do *software* livres ou *shell scripts*;
- Desenvolver uma *interface Web* em PHP, para que o gerenciamento da rede possa ser feita de forma personalizável a qualquer ambiente.

1.2 JUSTIFICATIVA

O *Firewall* Linux é, segundo Pinheiro(2002), uma das ferramentas mais seguras e estáveis quando se refere a sistema de segurança em redes de computadores. Com ela é possível fazer regras e políticas de segurança que se adéquam diretamente as necessidades de uma empresa com rapidez e baixo. Porém possui uma grande desvantagem, a configuração das regras do *Firewall*(IPTABLES) é complexa. Toda sua manipulação é feita por linha de comando, o que gera grandes dificuldades para administradores ou usuários pouco familiarizados com essa ferramenta.

Justifica-se a necessidade em se criar uma *interface* amigável a qual seja capaz de facilitar as configurações desse tipo de ferramenta de segurança. Embora o *Firewall*(IPTABLES) seja uma excelente opção de segurança, entende-se que outras ferramentas agregadas a ela possam contribuir ainda mais para a melhoria do serviço. Nesse caso, destacam-se relatórios de acessos e os filtros de conteúdos.

2 FUNDAMENTAÇÃO TEÓRICA

Esta seção abordará histórico, funcionamento e manutenção de uma rede de computadores, e as ferramentas que foram usadas para a o desenvolvimento do presente trabalho.

2.1 MODELO OSI

A comunicação entre computadores interligados a uma rede, seja ela uma rede local(*LAN - Local Area Network*), ou uma rede mundial(*WAN - Word Area Network*), há necessidade de uma padronização entre os componentes. Visando facilitar o processo de padronização e obter interconectividade entre máquinas de diferentes fabricantes, a Organização Internacional de Padronização (ISO - *International Standards Organization*) aprovou, no início dos anos 80, um modelo de referência para permitir a comunicação entre máquinas heterogêneas, denominado OSI (*Open Systems Interconnection*). (TANENBAUM, 1997).

O Modelo OSI, é composto por 7(sete) camadas, descritas na Tabela 1.

Camada		**Função**
7	Aplicação (Application)	Camada que fornece aos usuários acesso ao ambiente OSI e provê sistemas distribuídos de informação.
6	Apresentação (Presentation)	Camada responsável por prover independência aos processos de aplicação das diferenças na representação dos dados (sintaxe).
5	Sessão (Session)	Camada que provê a estrutura de controle para a comunicação entre as aplicações. Estabelece, gerencia e termina conexões (sessões) entre aplicações.
4	Transporte (Transport)	Camada responsável pela transferência de dados entre dois pontos de forma transparente e confiável com funções como controle de fluxo e correção de erro fim a fim.
3	Rede (Network)	Camada que fornece para as camadas superiores independência das tecnologias de transmissão e comutação usadas para conectar os sistemas. Responsável por estabelecer, manter e terminar conexões.
2	Enlace de dados (Data Link)	Camada responsável pela transmissão confiável de informação através do enlace físico. Envia blocos de dados (frames) com o necessário controle de erro e de fluxo.
1	Física (Physical)	Camada responsável pela transmissão de uma seqüência de bits em um meio físico. Trata das características mecânicas, elétricas, funcionais e procedurais para acessar o meio físico.

Tabela 1 - Camadas Modelo OSI (Fonte: Autor).

2.2 MODELO TCP/IP

Assim como a necessidade de uma padronização entre os diferentes fabricantes existe também necessidade de interligação entre diferentes tecnologias de redes. Baseado no modelo OSI o modelo de referência TCP/IP atualmente é o mais usado em redes locais e tem como base dois protocolos: o TCP *(Transmission Control Protocol)*, um protocolo compatível com a camada 4 do modelo OSI, que fornece um serviço de transporte orientado à conexão, e o IP (*Internet Protocol*), compatível com a camada 3, que fornece um serviço de rede não-orientado a conexão. Isso se deve basicamente à popularização da *Internet*, já que esse protocolo foi criado para ser utilizado na *Internet* (GALLO, 2003). Dessa forma o TCP/IP passou a ser mais utilizado que o modelo OSI.

O TCP/IP é segundo TANEMBAUM (1997), um conjunto de protocolos de interconexão de sistemas, executado em ambiente aberto, utilizando uma arquitetura de quatro camadas, definidas como na Tabela 2:

Camadas
Aplicação
Transporte
Rede
Interface de Rede

Tabela 2 - Camadas Modelo TCP/IP (Fonte: Autor)

- **Camada de Aplicação:** fornece a interface do usuário de rede na forma de aplicativos e serviços de rede. Exemplos de protocolo: *Simple Mail Transfer Protocol* (SMTP RFC821), *File Tranfer Protocol* (FTP RFC959) e Telnet(RFC854);

- **Camada de Transporte:** responsável por organizar as mensagens

recebidas de camadas mais altas nos segmentos, por controlar os erros e por controlar o fluxo de dados. Exemplos de protocolo: TCP(793) e *User Datagram Protocol* (UDP 768);

- **Camada de Rede:** responsável pelo endereçamento dos equipamentos e pela transmissão (ou roteamento) dos dados em redes diferentes. Exemplos de protocolo: IP (RFC 791) e *Internet Control Message Protocol* (ICMP RFC 792); e

- Camada de Interface de rede: responsável por controlar o fluxo de dados e organizar os bits da camada física. Exemplos de protocolo: Ethernet IEEE 802.11 (RFC 3580), IEEE 802.5 *Token Ring* (RFC 1231).

2.3 TRADUÇÃO DE ENDEREÇOS DE REDE BÁSICA (NAT)

Para a comunicação entre computadores que não estão na mesma rede LAN é necessário uma tradução do IP da rede LAN para o IP da rede WAN, essa tradução é feita através do NAT (*Network Address Translation*).

NAT também conhecido como *masquerading* é segundo GALLO (2003) uma técnica que consiste em reescrever os endereços IP de origem de um pacote que passam por um *router* ou *Firewall* de maneira que um computador de uma rede interna tenha acesso ao exterior, essa técnica é transparente para os usuários finais.

A necessidade de tradução de endereços IP surge quando os endereços de uma rede IP internos não podem ser usados fora da rede ou por razões de privacidade ou porque são inválidos para uso fora da rede. Topologia da rede fora de um domínio local pode mudar de várias maneiras.

A princípio o NAT foi criado para reduzir os problemas como a escassez de endereços IPV4(Protocolo de *Internet* Versão 4 RFC 791), que apesar de diversos mecanismos terem surgidos para a resolução desse problema como o IPV6(Protocolo de *Internet* Versão 6 RFC 2460), a técnica de NAT continua sendo a mais utilizada por equipamentos e administradores de rede, além disso, a técnica não serve somente para a finalidade inicial, mas

também para ocultar o endereço IP de origem.

Uma tabela NAT possui, segundo Neto (2004), as propriedades citadas no parágrafo anterior, e para isso contempla 3 situações: *Prerouting*, *Output* e *Postrouting*, apresentadas na Tabela 3.

Situação	Descrição
PREROUTING	Utilizada quando há necessidade de se fazer alterações em pacotes antes que os mesmos sejam roteados;
OUTPUT	Qualquer pacote gerado na máquina filtro e que deva sair para a rede será tratado pela cadeia OUTPUT;
POSTROUTING	Utilizado quando há necessidade de se fazer alterações em pacotes após o tratamento de roteamento;

Tabela 3 - Tabela NAT. (Fonte: O Autor)

2.4 SISTEMA DE PREVENÇÃO DE INVASÃO (IPS)

Com a interligação da rede interna com a externa (NAT), torna-se a segurança das informações mais vulneráveis, através da invasão de seres mal intencionados na rede externa a rede interna de uma organização. Por isso a importância de ferramentas de previsão de intrusão e detecção de intrusão.

Segundo Melo (2006), o IPS(Sistema de Prevenção de Invasão) é uma solução ativa de segurança, isto é, um sistema de prevenção de invasão a qual possui a capacidade de fornecer segurança em todos os níveis de sistemas, desde o núcleo do sistema operacional até os pacotes de dados da rede. São os IPS os responsáveis por proverem políticas e regras para o tráfego de rede.

Além disso, é importante que haja um sistema trabalhando em conjunto com o IPS, nesse caso, torna-se necessário o emprego de um IDS *(Sistema de Detecção de Intrusão)*, ou seja, um sistema que emita alertas para administradores de sistemas e redes em caso de tráfego suspeito.

Segundo NPD-UFES (2013), enquanto o IDS informa sobre um potencial ataque, o IPS promove tentativas de parar o ataque. Um outro grande

avanço sobre o IDS é que o IPS tem a capacidade de prevenir invasões com “assinaturas” conhecidas, mas também pode impedir alguns ataques não conhecidos, devido a sua base de dados de “comportamentos” de ataques genéricos. Visto como uma combinação de IDS e de uma “camada de aplicação *Firewall*” para proteção, o IPS geralmente é considerado a geração seguinte do IDS. De modo geral, IDS é uma solução passiva de segurança ao passo que IPS é uma solução ativa de segurança.

2.5 *FIREWALL*

Além de se preocupar com intrusões na rede, também é papel de um bom administrador filtrar os conteúdos acessados entre as redes ao seu cuidado uma maneira de se administrar esses conteúdos é utilizando *Firewall.*

Segundo Neto (2004), um *Firewall* é uma ferramenta na qual são implementados regras destinados a administração da rede interna e da rede externa. É o administrador da rede o responsável por bloquear ou liberar os acessos em uma determinada rede, visando manter a integridade e a segurança das informações. Além disso, essa ferramenta tem como objetivos:

- Regular o tráfego de dados entre uma rede local e a rede externa não confiável, por meio da introdução de filtros para pacotes ou aplicações; e

- Impedir a transmissão e ou recepção de acessos nocivos ou não autorizados dentro de uma rede local.

A palavra *Firewall* que em inglês significa Parede de Fogo, é simbolizada por um muro que intermédia a rede LAN e WAN, como apresentado na Figura 1.

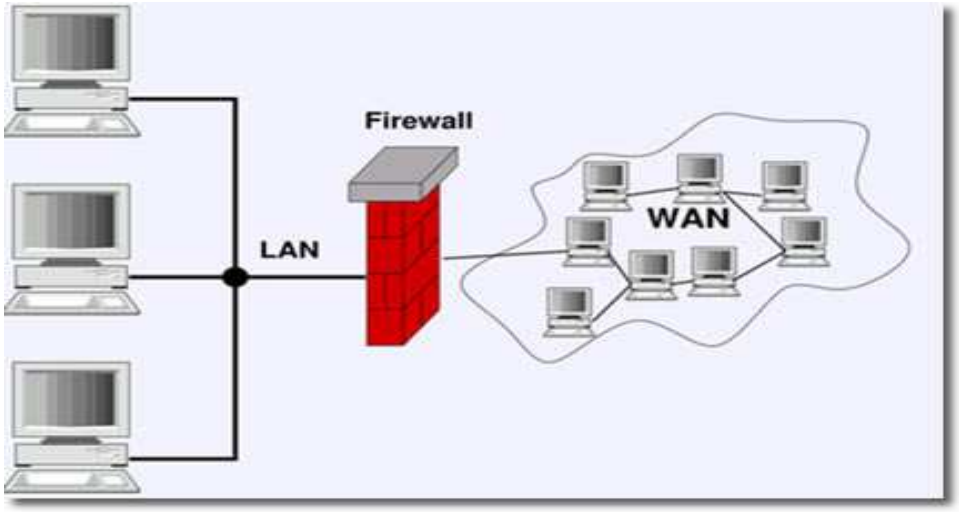

Figura 1 - *Funcionamento de Firewall*
Fonte: http://www.infoescola.com

Toda informação trafegada em uma rede ocorre em formato de pacotes. O início de cada pacote informa para onde ele está indo, de onde veio e o tipo do pacote. A parte inicial deste pacote é chamada cabeçalho. O restante do pacote, contendo a informação propriamente dita, é chamado de corpo do pacote.

Um filtro de pacotes analisa o cabeçalho dos pacotes que passam pela máquina e decide o que fazer com o pacote inteiro. Possíveis ações a serem tomadas em relação ao pacote são:

a. **Aceitar:** o pacote pode seguir até seu destino.
b. **Rejeitar:** o pacote será descartado, como se a máquina jamais o tivesse recebido.
c. **Bloquear:** o pacote será descartado, mas a origem do pacote será informada de que esta ação foi tomada.

O filtro de pacotes do *kernel* (núcleo do sistema operativo do computador) é controlado por *regras de Firewall*, as quais podem ser divididas, segundo Melo (2006) em quatro categorias: a cadeia de entrada (*input chain*), a cadeia de saída (*output chain*), a cadeia de reenvio (*forward chain*) e cadeias definidas pelo usuário (*user defined chain*). Para cada uma destas cadeias é mantida uma tabela de regras separada.

Uma regra de *Firewall* especifica os critérios de análise de um pacote e o seu alvo (*target*). Se o pacote não casa com o padrão especificado

pela regra, a regra seguinte da cadeia é analisada. Se desta vez o pacote casar com o padrão, a regra seguinte é definida pelo alvo, que pode ser o nome de uma cadeia definida pelo usuário, ou um dos seguintes valores especiais:

a. ***ACCEPT*** - Significa que o filtro de pacotes deve deixar o pacote passar.

b. ***DROP*** - Significa que o filtro de pacotes deve impedir que o pacote siga adiante.

c. ***REJECT*** - Assim como *DROP*, significa que o pacote não deve seguir adiante, mas uma mensagem *Internet Control Message Protocol* (ICMP) é enviada ao sistema originador do pacote, avisando-o de que o pacote foi rejeitado.

d. ***MASQUERADE*** - Este alvo somente é válido para a cadeia de reenvio e para cadeias definidas pelo usuário, e somente pode ser utilizado quando o *kernel* é compilado com suporte a IP *Masquerade*. Neste caso, pacotes serão mascarados como se eles tivessem sido originados pela máquina local.

e. ***REDIRECT*** - Este alvo somente é válido para a cadeia de entrada e para cadeias definidas pelo usuário e somente pode ser utilizado se o *kernel* foi compilado com a opção de Transparente Proxy, isto é, uma arquitetura que permite que o navegador cliente não saiba da existência do proxy. Com isto, pacotes serão redirecionados para um *socket* local, mesmo que eles tenham sido enviados para uma máquina remota. Obviamente, isto só faz sentido se a máquina local é utilizada como *gateway* para outras máquinas. Se a porta especificada para redirecionamento é “0”, que é o valor padrão, a porta de destino dos pacotes será utilizada como porta de redirecionamento. Se for especificada uma outra porta qualquer, esta será utilizada, independentemente daquela especificada nos pacotes.

f. ***RETURN*** - Se a regra contendo o alvo *RETURN* foi chamada por uma outra regra, a regra seguinte da cadeia que a chamou é analisada. Caso ela não tenha sido chamada por outra regra, a política padrão da cadeia é utilizada para definir o destino do pacote.

Para Avishai (2013), os *Firewall* podem ser classificados em duas abordagens: *Firewall Stateless* (Filtros de Pacotes) *e Firewall Statefull* (*Firewall* de Estado de Sessão).

2.5.1 *Firewall de Pacotes*

O modo de filtragem por pacotes (S*tateless),* tende a tratar cada pacote roteado pelo *Firewall* como pacotes individuais, que não tenham associação alguma com qualquer outro tráfego. Esse tipo de filtragem é o mais comum e o mais fácil de implementar. (NAKAMURA, 2007).

Uma conexão é sempre bidirecional, assim deve-se criar regras para que o *host* remetente possa enviar pacotes para o *host* destinatário e vice-versa. Além disso, o *host* destinatário necessita enviar pacotes de resposta ao *host* remetente para que haja o estabelecimento de conexões.

2.5.2 *Firewall* de Estado de Sessão

Os *Firewall* de estado de sessão (*Stateful Firewall*) analisam os pacotes e guardam o estado de cada conexão de maneira que seja possível identificar e fazer uma previsão das respostas legítimas, desta forma é possível impedir o tráfego de pacotes ilegítimos.

Filtros baseados em estados: Na verdade, o *Firewall* baseado em estado é uma evolução do filtro de pacotes, pois possui uma tabela de estado que é associada à tabela de regras, o que auxilia na tomada de decisões. Neste *Firewall* as conexões são monitoradas a todo instante e um pacote só pode passar pelo *Firewall* se fizer parte da tabela de estados.

Segundo Ulbrich (2009), com o uso da tecnologia SMLI/Deep *Packet Inspection* o *Firewall* utiliza mecanismos otimizados de verificação de tráfego para analisá-los sob a perspectiva da tabela de estado de conexões legítimas. Simultaneamente, os pacotes também vão sendo comparados a padrões legítimos de tráfego para identificar possíveis ataques ou anomalias. A combinação permite que novos padrões de tráfegos sejam entendidos como serviços e possam ser adicionados às regras válidas em poucos minutos.

2.5.3 *Firewall* Pessoais

Este *Firewall* se diferencia de todos os demais, isto, porque eles não protegem um segmento de rede. Essa proteção se restringe apenas ao equipamento onde está instalado, que geralmente utilizam apenas um canal de comunicação com a *Internet*.

2.6 *Firewall Proxy*

É uma aplicação instalada em um servidor o qual funciona como intermediário entre um navegador Web e a *Internet*. Um *Firewall* proxy ou servidor Proxy, ajuda a melhorar o desempenho na Web, armazenando uma cópia das páginas utilizadas com maior frequência. Quando um navegador solicita a requisição de uma página que está armazenada na coleção do servidor proxy (o cache), ela é disponibilizada pelo servidor proxy, o que é mais rápido do que acessar a Web. Os servidores proxy também ajudam a melhorar a segurança porque podem filtrar alguns tipos de conteúdo da Web, como é o caso de *softwares* mal-intencionados.

Segundo (WESSELS, 2004),o servidor proxy/cache squid trata-se de ferramenta capaz de aceitar requisições *HTTP (* Hypertext Transfer Protocol) e *HTTPS (*HyperText Transfer Protocol Secure) de clientes e capaz de efetuar requisições *HTTP*, *FTP* e *Gopher* para servidores, além de implementar várias características comumente úteis em ambientes corporativos:

- Controle de banda no acesso a *Internet*;
- Redução do tempo de carga de páginas na *Internet*;
- Coleta de estatísticas do tráfego de acesso a *Internet* proveniente da rede privativa;
- Bloqueio de sítios considerados de conteúdo inapropriado;
- Garantia de que somente os usuários autorizados terão acesso a *Internet*;
- Conversão de requisições *HTTPS* de um lado em *HTTP* do outro lado;
- Proteção de máquinas internas de acessos externos uma vez que as requisições a *sites* externos são efetuadas pelo proxy.

Os requisitos de *hardware,* segundo*(WESSELS 2004),* necessários para sua implantação são, em geral, modestos. Mesmo assim, a memória é o recurso mais importante, uma vez que pouca quantidade de memória degrada consideravelmente a performance. Além disso, o espaço em disco é outro fator importante, pois mais espaço em disco significa mais objetos em *cache* e, portanto, menor tempo de resposta. O funcionamento de um *Proxy* permite intermediar as transações entre clientes e servidores. Ele aceita requisições dos clientes, processa e as encaminha ao servidor desejado. Tais requisições podem ser registradas, rejeitadas e modificadas antes do encaminhamento. Além disso, por funcionar como *Cache*, a ferramenta armazena localmente conteúdo de páginas acessadas recentemente com o objetivo de reutilizá-las, aumentando assim a performance pela diminuição do tempo de resposta. A Figura 2 ilustra como são feitos os acessos através do Proxy.

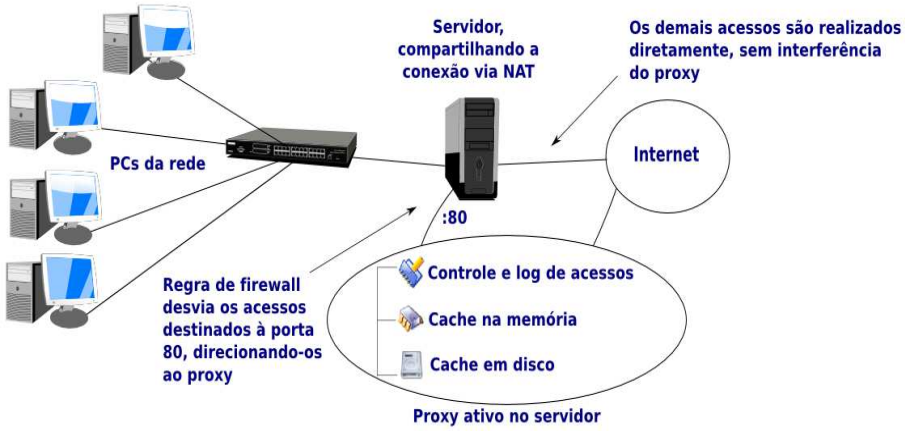

Figura 2 - Funcionamento de um Servidor Proxy. WESSELS (2004)

Essa aplicação permite ainda a implementação de várias funcionalidades através do uso de ACL's (*Access Control List*). Esta implementação permite a criação de listas capazes de filtrar desde simples domínios até tipos de conteúdo especificados.

2.6.1 *Proxy* Transparente

O proxy transparente é segundo Marina Martinez (revista Info Escola, 2010) uma arquitetura que permite que o navegador cliente não saiba da existência do proxy. Ele acha que está solicitando o recurso diretamente ao servidor original, o proxy encarrega-se de capturar e processar a solicitação. A principal vantagem nesta arquitetura é que não é necessária a configuração de *proxy* nos navegadores cliente. Outra vantagem alegada (incorretamente) é que o *proxy* não transparente não impede a conexão direta à *Internet*.

2.7 IPTABLES

O IPTABLES é a principal ferramenta que permite a criação de *Firewall* e NAT's no sistema operacional Linux. Suas principais características são suporte a protocolos ICMP, TCP e UDP, manipular serviços de proxy na rede, tratamento de tráfego dividido em cadeias. O IPTABLES é segundo MELO (2006) muito rápido, estável, seguro, além disso, possui suporte a módulos externos, redirecionamento de portas, suporte a SNAT e DNAT, contagem de pacotes, entre outras funcionalidades.

Em redes de computadores, uma DMZ (zona desmilitarizada) é um computador *host* ou uma pequena rede inserida como uma "zona neutra" entre a rede privada de uma empresa a qual esteja fora da rede pública. Isso impede que usuários de fora, tenham acesso direto a um servidor que tem os dados da empresa. (O termo vem da zona tampão geográfica que foi estabelecido entre a Coréia do Norte e Coréia do Sul na sequência da "ação policial" da ONU no início de 1950.) Uma DMZ é uma abordagem opcional e mais seguro para um *Firewall* e efetivamente funciona como um proxy servidor também.

2.8 SNMP - *SIMPLE NETWORK MANAGEMENT PROTOCOL*

Este protocolo tem como premissa à flexibilidade e a facilidade de implementação, também em relação aos produtos futuros. Sua especificação está contida no RFC 1157.

O SNMP é um protocolo de gerência definido a nível de aplicação, é utilizado para obter informações de servidores SNMP - agentes espalhados em uma rede baseada na pilha de protocolos TCP/IP. Os dados são obtidos através de requisições de um gerente a um ou mais agentes utilizando os serviços do protocolo de transporte UDP para enviar e receber suas mensagens através da rede.

O gerenciamento da rede através do SNMP permite o acompanhamento simples e fácil do estado, em tempo real, da rede, podendo ser utilizado para gerenciar diferentes tipos de sistemas.

A utilização de um número limitado de operações, baseadas em um mecanismo de busca e alteração, torna o protocolo de fácil implementação, simples, estável e flexível. Como consequência reduz o tráfego de mensagens de gerenciamento através da rede e permite a introdução de novas características.

O funcionamento do SNMP é baseado em dois dispositivos o agente e o gerente. Cada máquina gerenciada é vista como um conjunto de variáveis que representam informações referentes ao seu estado atual, estas informações ficam disponíveis ao gerente através de consulta e podem ser alteradas por ele.

2.9 LINGUAGENS PARA DESENVOLVIMENTO DE WEBSITES

Embora haja diversas linguagens voltadas a desenvolvimento de Websites, que permitem uma melhor interação entre Shell Script e uma interface Web são as descritas abaixo:

2.9.1 PHP

A linguagem de programação PHP é utilizada no desenvolvimento de sites Web, devido a sua boa integração com a linguagem HTML. Sua sintaxe é semelhante a algumas linguagens como Java, Perl e C, o que torna sua curva de aprendizagem acentuada. O PHP é o sucessor de um *software* mais antigo desenvolvido em 1995 por Rasmus Lerdoff, conhecido como PHP/FI (Personal Home Page Tools/FormsInterpreter), que era um conjunto de scripts em Perl e não uma linguagem de programação. (THE PHP GROUP, 2008).

No ano de 1997, a terceira versão foi desenvolvida e reescrita por AndiGutmans e ZeevSuraski, numa universidade através de um projeto de e-commerce neste momento seu nome passou a ser chamado simplesmente de PHP. Foi oficialmente laçando em junho de 1998 após 9 meses em versões de teste (THE PHP GROUP, 2008).

Em 1999 foi lançando o PHP 4, ganhando uma nova versão, as suas principais vantagens em relação a versão anterior que podem ser destacas vão desde ganhos consideráveis de performance, novas construções na linguagem e o uso de sessões HTTP. A versão 5 foi lançada em julho de 2004, é a versão atual, sua principal mudança foi a implementação melhorada de orientação a objetos (GROUP, 2008).

2.9.2 HTML

A origem do HTML vem da junção da linguagem SGML e dos padrões HyTime. Esta linguagem criada por Tim Berners-Lee, e possui um conjunto de tags(etiquetas) pré-definidas com suporte para Hipertexto, foi criada para o tráfego de documentos científicos e técnicos, e atualmente conta com recursos multimídia, sendo conhecida mundialmente como a linguagem padrão da Web (VALENTE, 2007).

Com a evolução da linguagem HTML sendo usada para a criação de páginas Web, notou-se a necessidade de torná-la mais flexível, ou seja, manipular visualmente a apresentação, a aparência dos textos. (SILVA, 2003).

Segundo Silva (2003), a HTML foi criada com a ideia de fazer

a transferência de informações de natureza cientifica e de documentos. A princípio a HTML atendia somente a um público alvo, os cientistas, mas com a evolução da linguagem, acabou tornando-se a linguagem mais conhecida no mundo para apresentação de páginas Web.

3 TRABALHOS RELACIONADOS

Esta seção apresenta as ferramentas mais comuns utilizadas pelos administradores de redes, a fim de aplicar regras de segurança e protegê-la contra possíveis invasões.

3.1 PfSense

É um projeto que iniciou-se em 2004 como um projeto secundário do projeto m0n0wall, com o propósito de desenvolver um *Firewall*, mas com foco para as instalações computadores em vez de o foco em *hardware* embarcados.

Segundo Christopher M. Buechler e Jim Pingle é uma distribuição open source personalizada do FreeBSD adaptada para o uso como um *Firewall* e roteador, através de uma interface Web fácil de usar. Além de ser um *Firewall* poderoso, flexível e conter plataforma de roteamento, ele inclui uma longa lista de recursos relacionados e um sistema de pacotes permitindo expansão mais sem adição de vulnerabilidades de segurança e potencial para a distribuição base.

A primeira versão do PfSense classificada como estável foi a 1.5.3. 1.0. Ela foi lançada em 4 de outubro de 2006, acompanhada com uma versão de correção de bugs 1.0.1 em 20 de outubro de 2006. (Christopher M. Buechler e Jim Pingle, 2009)

3.2 ClearOS COMMUNITY EDITION

Desenvolvido pela fundação Clear, é uma aplicação livre e de código aberto voltada para proteger redes de organizações de pequeno e médio porte, baseada na plataforma GNU/Linux. Um de seus diferenciais é ser de fácil configuração através de uma interface Web.

ClearOS segundo GOLÇALVES(2004), é considerado um dos melhores *gateway* e distribuidor *Firewall* da plataforma GNU/Linux, com o recurso de servidor Web. Baseado no CentOS e Red Hat Enterprise Linux. Ele é composto com características como iptables, detecção de intrusão, VPN e

Proxy. Outra vantagem é o rápido processo de instalação desse sistema operacional, aproximadamente 10 a 15 minutos (dependendo do *hardware* onde se aplica).

3.3 ZENTYAL

Anteriormente conhecido como ebox, Zentyal é um *Business Server Small Linux*(Servidor para Pequenas Empresas) e atua como um *gateway*, gerente de Infra-estrutura de Rede, gerente de *Unified Threat Management* (Gestão Unificada de Ameaças), Office Server, Servidor de Comunicações Unificadas ou uma combinação deles. Ele pertence e é patrocinado por uma empresa espanhola chamada *eBox technologies* e pode ser executado utilizando a plataforma Linux/Ubuntu. O processo de instalação é bastante simples e rápido.

3.4 SMOOTHWALL

O Smoothwall Open Source Project foi criada em 2000 para desenvolver e manter Smoothwall Express - um *Firewall* gratuito que inclui seu próprio sistema operacional baseado na plataforma Linux e uma interface Web de fácil uso. O objetivo do projeto era a criação de uma distribuição Linux que pudesse se converter um microcomputador em um dispositivo redundante de *Internet Firewall*.

A versão 0.9.9 foi lançada em setembro de 2001 nesta versão foi incorporada uma plataforma Web multilinguagem, permitindo que usuários com conhecimento menos aprofundado na plataforma GNU/Linux pudesse utilizar e administrar o *Firewall*. Ele também incluiu o Snort ou IDS (Sistema de Detecção de Intrusão) e suporte para modems ADSL (Linha de Assinante Digital Assimétrica) e conexões PPPoE(Protocolo Ponto à Ponto para a *Internet*).

3.5 IPCop

Começou como um adjacente do projeto Smoothwall acima mencionado. No entanto, com o passar do tempo, os dois projetos divergiram se significativamente o suficiente para classificá-los como duas distribuições

completamente diferentes. IPCop, desde a sua criação, ganhou muita popularidade entre a comunidade, sendo até premiado por inovação em 2007 (prêmio Bossie).

O IPCop é um sistema especialmente desenvolvido com o propósito de ser *Firewall*. Diferentemente dos *Firewall* que podem ser instalados sob um sistema operacional com múltiplos propósitos, o IPCop possui em seu sistema somente os componentes necessários para seu funcionamento, anulando a possibilidade de algum conflito com outro programa ou módulo (DEMPSTER e EATON-LEE, 2006).

3.6 CONCLUSÃO DO CAPÍTULO

Na Tabela 4 é apresentado as ferramentas de *Firewall* e suas principais funcionalidades.

Software analisados / **Funcionalidades**	**PfSense**	**Clear OS**	**Smoothwall Express**	**Zentyal**	**IPCop**
Interface *Web* para administração	X	X	X		
PHP 5.3	X				
Gerenciamento de usuário	X		X	X	
Firewall Statefull	X	X		X	X
Proxy Server	X		X	X	X
Filtro de conteúdo	X	X	X	X	
Controle de *Internet*	X			X	
Monitoramento e filtro de Messenger	X		X	X	
VPN (IPSEC / PPTP/ OpenVpn)	X	X		X	X
IDS/IPS	X		X		X
Traffic Shapping (QOS)	X		X	X	
DMZ	X	X	X	X	
SO	X	X	X	X	X
Free	X	X	X	X	X
Open Source	X	X	X	X	X
SNORT		X	X		
Serviço de Arquivo e impressão		X			
MULTI WAN		X			
Relatório de estatísticas (MRTG)		X	X	X	
Controle de e-mail		X	X		

Bloqueio de lista de IP			X		X
NAT	X			X	

Tabela 4 - Tabela comparativa dos softwares analisados nesse trabalho.(Fonte: O Autor)

Nessa tabela comparativa é possível observar que cada ferramenta esta integrada a um SO especifico. Uma análise mais aprofundada de cada ferramenta permitiu que fosse observado a necessidade de uma máquina estar destinada ao uso especifico do *Firewall*.

Nesse trabalho, o *Firewall* proposto permite que um servidor com o sistema operacional Ubuntu Server, possa continuar sendo utilizado para outros serviços, e ainda ser utilizado como *Firewall* de pequenas redes.

A análise feita nessas ferramentas ainda permitiu que fossem comparados alguns recursos como, linguagem de programação e aplicações os quais serviram de base para a implementação do *Firewall* proposto.

4 ESTRUTURA DE FUNCIONAMENTO DO FMUS - *FIREWALL* MANAGER UBUNTU SERVER.

Com o objetivo de criar um *Firewall* personalizável, uma configuração é previamente estruturada afim de atender o publico alvo apresentado nesse trabalho. Nesse caso, foram idealizadas regras de permissão e de bloqueio as quais pudessem ser utilizadas por administradores de pequenas redes. Nessa configuração, foram criadas regras através do IPTABLES em arquivos Shell Script, especialmente desenvolvidas para o sistema operacional Ubuntu Server. A esse conjunto de regras e aplicações interligadas denomina-se nesse trabalho de FMUS - *Firewall* Manager Ubuntu Server, o qual tem por finalidade tornar o gerenciamento de rede de computadores mais amigável intuitivo.

A estrutura de funcionamento do FMUS, apresenta-se da seguinte forma:

i. Inicialmente é solicitado ao administrador de rede os endereços IP's. A partir disso, os endereços são atribuídos às variáveis, que conseguinte são integradas as regras de *IPTABLES;*

ii. Um serviço de DHCP é responsável por atribuir um endereço de rede LAN para todas as máquinas consideradas clientes. Por padrão, cada máquina cliente passa a receber um endereço IP de forma aleatória, contudo para um gerenciamento mais eficaz o FMUS permite que sejam amarrados os endereços IP's aos endereços MAC's. Assim, é assegurado que somente uma máquina especifica irá receber um endereço IP previamente determinado. A partir disso, todas as regras de permissões e bloqueios passam a serem feitas através do endereço IP de cada máquina;

iii. Uma regra de NAT é aplicada por padrão a qual permite que as máquinas clientes possam navegar por todo o conteúdo Web. No entanto, o FMUS permite ao invés de dar

permissão total para qualquer acesso à rede externa(WAN), essa poderá ser feita por portas. Essas portas são declaradas na interface Web do FMUS.

iv. Todo o conteúdo de página Web, isso, a navegação por sites é redirecionado para o servidor proxy (squid), o qual filtra as páginas por palavras chaves, e endereços de sites. Nesse caso, a configuração do serviço proxy é transparente para o usuário.

v. Para o acesso as redes sociais, o FMUS periodicamente busca novos endereços IP´s de servidores de uma determinada redes social os quais encontram-se espalhados pelo mundo e inteja esse endereços em um arquivo texto. Assim, o bloqueio passa a ser feito através de uma lista de endereços IP´s os quais estão em constante atualização.

Um *layout* dessa estrutura de funcionamento do FMUS é exemplificada na Figura 5.

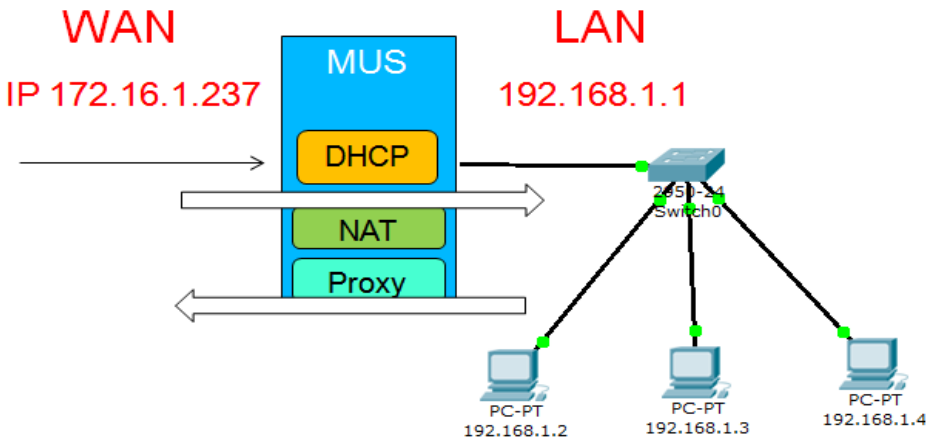

Tabela 5 - Estrutura de funcionamento do FMUS.

Fonte: O Autor.

Os demais itens que o *Firewall* FMUS apresenta, isto é,

relatórios de acessos, consumo da banda, foram incorporados, com o objetivo de permitir o acompanhamento das regras quando aplicadas a uma determinada rede. Dessa forma, ao marcar que não será permitido que haja determinadas máquinas clientes tenham acesso à uma determinada rede social, o relatório permitirá esse acompanhamento, bem como a redução da largura de banda de rede *Internet* trafegada

5 DESENVOLVIMENTO

Nesse capítulo é descrito as regras, as aplicações e as configurações utilizadas no desenvolvimento do *Firewall* FMUS, seguindo a estrutura apresentada no Capítulo 4.

Conforme apresentado no Capítulo 3, foi realizado uma pesquisa de natureza exploratória a qual serviu de base para propor o desenvolvimento da ferramenta de um *Firewall* o qual utiliza uma interface gráfica amigável.

Relatado que somente as regras de IPTABLES não são suficientes para permitir um gerenciamento satisfatório quanto a segurança, outras aplicações foram escolhidas para integrar essa proposta de trabalho.

Ao analisar as ferramentas de *Firewall,* anteriormente descritas, levou em consideração que elas fossem de natureza *open source*, ou seja, de código aberto. Mediante as análises feitas nesses *software,* foi possível apresentar um diferencial de recursos.

Além dos sistema operacional ser Ubuntu Server e não exigir que uma máquina estritamente dedicada a função de *Firewall* as regras de NAT foram criadas de forma personalizável, bem como as permissões de acessos aos conteúdos Web.

5.1 FERRAMENTAS UTILIZADAS PARA O DESENVOLVIMENTO

A implementação da ferramenta *Web* para configuração de servidor, foi realizada utilizando as seguintes linguagens: HTML, PHP e *Shell Script*. A linguagem de marcação HTML foi usada para a criação de algumas páginas *Web*. O PHP foi utilizado para fazer a execução das informações obtidas através das telas de HTML na liguagem *Shell Script*.

O pacote de aplicações utilizados nesse *Firewall* são:

- Squid3 - Servidor Proxy/Cache.
- Apache2 - Servidor de página HTTP.
- dhcp3-server - Servidor DHCP (Dynamic Host Configuration Protocol).

- SNMP - Protocolo Simples de Gerência de Rede.
- MRTG - Análise Gráfica de tráfego.
- SARG - Relatório de acesso do servidor proxy.

5.2 REQUISITOS DO *FIREWALL FMUS (FIREWALL MANAGER UBUNTU SERVER)*

Para a instalação da ferramenta de *Firewall* a qual se propôs nesse trabalho, isto é, o FMUS – Manager Ubuntu Server, o usuário deverá inicialmente ter o Sistema Operacional Ubuntu Server já instalado. Os requisitos de *hardware* necessários, seguem os mesmos solicitado pela Canonical (Comunidade Linux responsável pelo desenvolvimento do Ubuntu). Uma ressalva apenas é quanto o espaço de armazenamento destinado ao cache do servidor proxy e dos logs gerados pelo servidor os quais podem crescer. Nesse caso, recomenda-se que sejam utilizados pelo menos 30 GBytes de espaço a mais que a configuração de *hardware* recomendada pelos desenvolvedores do Ubuntu Server.

Requisito Mínimo:

Processador: Pentium 4, 1GHz

Memória RAM: 512MB

Disco: 5GB

Quantidade placas de rede: 2

Requisito Recomendável:

Processador: Pentium 4, 1GHz ou superior

Memória RAM: 1GB ou mais

Disco: 35GB ou mais

Quanto aos requisitos do sistema, é necessário que a máquina servidora esteja conectada a rede *Internet* e que um pacote "make" esteja instado.

É o pacote "make" o responsável por verificar e instalar todas as aplicações necessárias conforme regras criadas a partir de seu arquivo de

configuração "makefile". Trata-se de uma lista de tarefas a serem executadas para a completa instalação da aplicação a que se destina, neste caso o *Firewall* FMUS.

Caso esse pacote não se encontre instalado o mesmo poderá ser obtido através do comando apresentado no Quadro 1.

Quadro 1 - Instalação make

apt-get -y install make

O *Firewall* FMUS foi desenvolvido para trabalhar tanto na plataforma 32 quanto em 64 *bits*, pois utiliza os mesmos pacotes de aplicações nativos da versão do Ubuntu Server instalada. Esse *Firewall* utiliza como base a codificação em *Shell Script* interligando aplicações já previamente desenvolvidas para o Ubuntu Server, reduzindo drasticamente a incompatibilidade de pacotes e configurações.

A ferramenta de *Firewall* proposta, o FMUS, poderá ser obtida fazendo o *download* no seguinte endereço:

http://www.labinfoclm.uenp.edu.br/mus/mus1-0-0613.tar

Uma vez obtido o pacote de instalação mus1-0-0613.tar, é necessário a descompactação do mesmo, para tanto o usuário deverá executar o comando disposto no Quadro 2.

Quadro 2 - Pacote .tar

tar –zxf mus1-0-0613.tar

Após a descompactação, é necessário acessar a pasta com os arquivos descompactados e executar o comando apresentado no Quadro 3. A partir desse comando é iniciado a instalação de todos os pacotes necessários ao *Firewall* FMUS. É necessário que para a instalação da ferramenta, o usuário seja o administrador do sistema, isto é, o *root*.

Quadro 3 - Executar "make"

make install

5.3 Descrição do pacote de instalação Make File:

Nessa seção é descrito todos os passos o qual a ferramenta make irá executar para a instalação do *Firewall* FMUS.

Passo 01: Atualização do sistema operacional

O regra apresentada no Quadro 4 atualiza o SO, para um melhor aproveito de suas funcionalidades.

Quadro 4 - Regra make "atualiza"

```
atualiza:
        @echo ////////////instalando o Update...\\\\\\\\\\\\\\\\\\\\\\\\
        apt-get -y upgrade update
```

Durante o processo de instalação, o make irá copiar da pasta “arquivos” contidos no pacote de arquivo do FMUS, os arquivos .conf pré configurados, isto é, serão substituídos os arquivos de configuração padrão de cada aplicação, por arquivos de configuração preparados para atender os requisitos do FMUS.

Passo 02: Instalação do apache

O primeiro pacote a ser instalado é o servidor de páginas Apache. A instalação ocorre de forma simples apresentada no Quadro 5, sem mudanças na configuração do mesmo e sem pacotes adicionais a essa aplicação.

Quadro 5 - Regra make "apache"

```
apache:
@echo ////////////////////instalando o Apache...\\\\\\\\\\\\\\\\\\
        apt-get -y install apache2
        /etc/init.d/apache2 restart
```

Passo 03: Instalação do servidor DHCP

A instalação servidor DHCP torna-se necessária para gerenciar IP's às máquinas clientes em relação a identificação e bloquios das mesmas. Nesse caso, o arquivo dhcpd.conf original é subscrito por um arquivo pré-configurado "dhcpd_conf.txt". Como mostrado no Quadro 6.

Quadro 6 - Regra make "dhcp"

```
dhcp:
    @echo ////////////////////instalando o dhcp...\\\\\\\\\\\\\\\\\\\\
    apt-get install dhcp3-server
    cp /arquivos/dhcpd_conf.txt /etc/dhcp/dhcpd.conf
    service isc-dhcp-server restart
```

Passo 04: Instalação do agente SNMP

De forma análoga ao pacote anterior, o arquivo snmpd.conf é substituído por um arquivo pré-configurado "snmpd_conf.txt" contido no pacote de instalação. Esse é subscrito para o arquivo padrão do Ubuntu Server, contido em: "/etc/snmp/". Na sequencia o serviço é iniciado. Os passos dessas regras seguem apresentados no Quadro 7.

Quadro 7 - Regra make "snmp"

```
snmp:
    apt-get -y install snmpd
    cp snmpd_conf.txt /etc/snmp/snmpd.conf
    service snmpd restart
```

Passo 05: Instalação do MRTG

A regra apresentada no Quadro 8 contém dependência da regra anterior "snmp", para a criação dos pacotes necessários. Ela instala o MRTG, que cria uma pasta para o MRTG no diretório do apache, onde vai ficar os arquivos referentes ao MRTG, restarta o serviço de SNMP, modifica o arquivo de configuração do MRTG, cria o arquivo index.html no diretório criado

no apache, direcionando - o para o arquivo de configuração do MRTG e reinicia novamente o MRTG.

Quadro 8 - Regra make "mrtg1"

```
mrtg1: snmp
    apt-get -y install mrtg
    mkdir /var/www/mrtg
    service snmpd restart
    /usr/bin/cfgmaker --global 'WorkDir: /var/www/mrtg' -output /etc/mrtg.cfg
    public@localhost
    /usr/bin/indexmaker --output=/var/www/mrtg/index.html /etc/mrtg.cfg
    export LANG=C
    mrtg
```

Passo 06: Instalação do Servidor Proxy

A regra criada de instalação do Servidor Proxy Squid descrita no Quadro 9, dá permissão para o usuário proxy que é criado durante a instalação a pasta de configuração, copia o arquivo pré configurado para a pasta de configuração e inicia o serviço.

Quadro 9 - Regra make "proxy"

```
proxy:
    apt-get install squid squid-common bind9
    chown proxy:proxy /var/spool/squid3
    cp /arquivos/squid.conf.txt /etc/squid3
    squid3 -z
    squid3
```

Passo 07: Instalação da Ferramenta SARG

Quadro 10 - Regra make "sarg"

```
sarg:
        apt-get install sarg
        cp /arquivos/sarg.conf.txt /etc/sarg/sarg.conf
        sarg
```

Passo 08: Execução do comando make install

Regra para a chamada completa do make apresentada no Quadro 11. Onde suas dependências são as outras regras anteriores. Também é solicitado uma senha para administração do sistema.

Quadro 11 - Regra make "install"

```
install: atualiza php dhcp mrtg1
        @ echo Configure uma senha para acesso do sistema:
        htpasswd -c /usr/local/Firewall/.htpasswd admin
    @echo /////////////// INSTALAÇÃO OK \\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
        @echo O acesso ao sistema de configuração é feito através do endereço http://192.168.1.1/mus
```

5.4 APRESENTAÇÃO DA FERRAMENTA DE *FIREWALL* FMUS

Uma vez instalado todas as aplicações necessárias, o que se segue é a apresentação das telas de configuração da ferramenta. A ferramenta FMUS é inicialmente descrita através do seu menu de configuração.

Para acesso ao *Firewall* FMUS, através de uma interface Web, o usuário poderá se conectar ao servidor através de uma das portas ethernet. Como não é exigido uma interface gráfica para o Ubuntu Server, esse trabalho

descarta o acesso ao servidor via seu acesso local no próprio navegador. O usuário portanto, somente poderá acessar a interface Web da *Firewall* FMUS utilizando um outro equipamento. Nesse caso, recomenda-se que a porta ethernet escolhida para o acesso seja a porta destinada a rede interna (LAN), isto é, a interface de rede (eth1), uma vez que a outra porta (eth0) já esteja sendo ocupada como (WAN). Como o arquivo de configuração padrão já pré-configurou o serviço de DHCP, o computador utilizado para acesso ao *Firewall* FMUS, irá receber um endereço IP classe C para redes LANs. Nesse caso, um IP entre o intervalo 192.168.1.5 a 192.168.1.254. O endereço IP para acesso ao servidor o qual deverá ser acessado via navegador Web é: http://192.168.1.1/mus.

Uma vez acessado o endereço http://192.168.1.1/mus, uma tela de autenticação inicial de *login* e senha será apresentada. Observe que o login e senha foram previamente configurados ao final da instalação da ferramenta, quando ainda estava sendo executado o arquivo make. Após a autenticação um menu com os seguintes itens: Home, WAN/LAN. *Firewall*, Proxy, Redes Sociais e Status é apresentado. Esse menu pode ser observado na Figura 3.

A. Menu Home

Tela inicial (HOME) é possível observar um quadro com as principais informações sobre o servidor. A obtenção desses dados ocorreu através do arquivo /etc/motd, nativo da configuração padrão do Ubuntu Server.

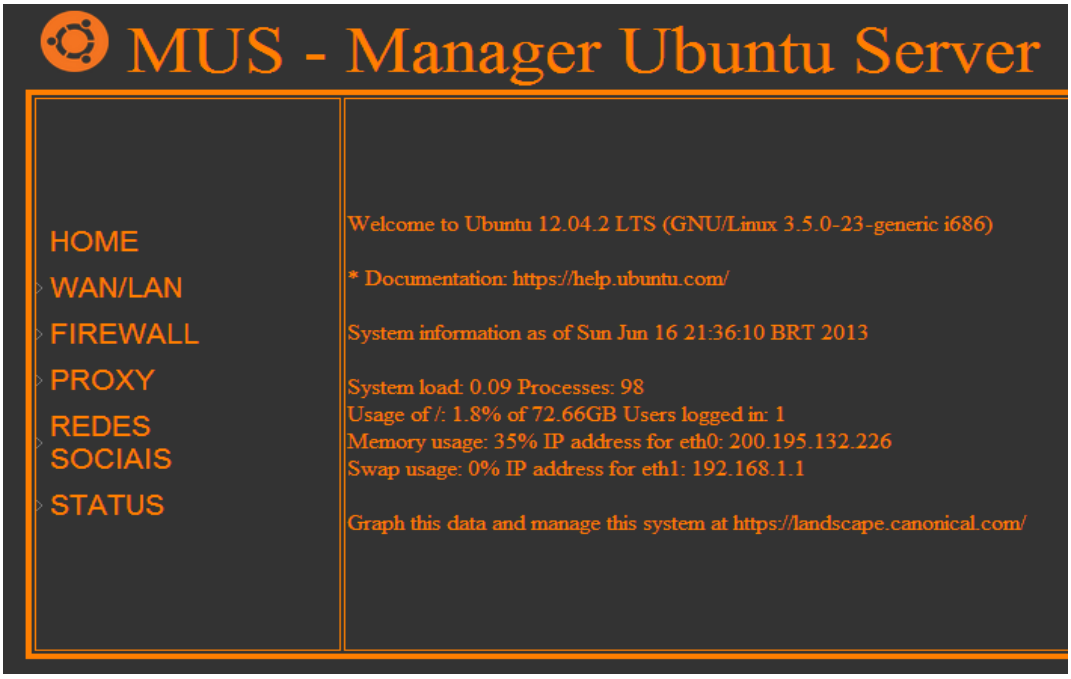

Figura 3 - Tela de apresentação “Home”

B. Menu WAN/LAN

A seguinte tela de configuração permite que o administrador, possa agora configura as informações de suas redes interna (LAN) e externa (WAN). As configurações pré-configuradas são apresentadas nessa tela, e exemplificadas através da Figura 4. Caso o administrador deseje aprender a configurar os endereços manualmente, o botão “Editar código fonte” permite que seja aberto o arquivo "/etc/network/interfaces". As configurações feitas somente terão validade após o administrador clicar em “Enviar Configurações”.

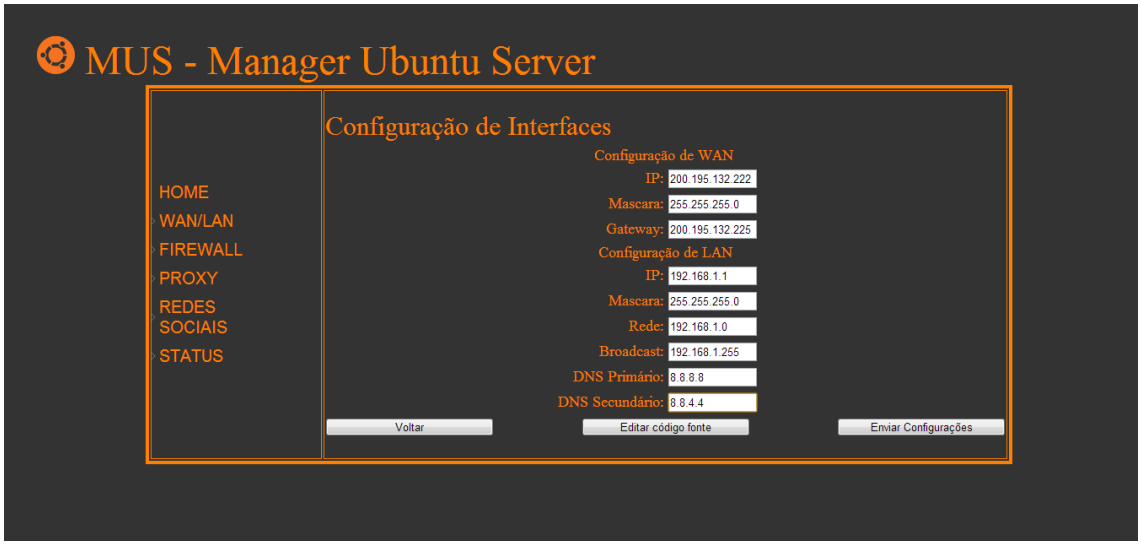

Figura 4 - Tela de configuração dos endereços IP

Para essa tela, a codificação que permite que a interação entre a linguagem PHP e o arquivo texto, estão descritas no Quadro 12:

Quadro 12 - Ação do botão código fonte

```
function ler_texto_linha($caminho){
        $handle = fopen($caminho, "r+");
        if($handle){
            while(($buffer = fgets($handle, 4096)) !== false){
            echo "<BR>".$buffer;
        }
        if(!feof($handle)){
            echo "Erro";
        }
        fclose($handle);
        }
    }
```

Nesse código, a variável "$caminho" indica o diretório onde se encontra o arquivo de configuração "interfaces.conf". O código acima abrirá o arquivo com no centro da página, contendo um botão "Enviar", o qual substitui o arquivo original de configuração pelo arquivo alterado e envia uma mensagem de sucesso na alteração, através do código mostrado no Quadro 13.

Quadro 13 - Ação do botão "enviar"

```
function grava_texto($caminho, $text){
    file_put_contents($caminho, $text);
    echo "<font size=5 color = #CFCFCF align=”center”>Arquivo alterado com SUCESSO!!</font>";
}
```

Quadro 14 - Ação do botão "Enviar Configurações"

```
function wanip($ipwan, $maskwan, $gtwwan, $iplan, $masklan, $redelan, $broadlan, $dns1, $dns2){
    $meuArray = file("/etc/network/interfaces.save"); // coloco todo o arquivo num array
    $arrayModificado = array(); // crio um array vazio.

for($n=0; $n < count($meuArray); $n++) {
echo $meuArray[$n];
echo "<BR>";
if($n == 7) {
    $arrayModificado[$n] = "address ".$ipwan."\n";
}
```

```
                else if($n == 8) {
                    $arrayModificado[$n] = "netmask ".$maskwan."\n";
                }
                else if($n == 9) {
                    $arrayModificado[$n] = "gateway ".$gtwwan."\n";
                }
                else if($n == 14) {
                    $arrayModificado[$n] = "address ".$iplan."\n";
                }
                else if($n == 15) {
                    $arrayModificado[$n] = "netmask ".$masklan."\n";
                }
                else if($n == 16) {
                    $arrayModificado[$n] = "network ".$redelan."\n";
                }
                else if($n == 17)
                    $arrayModificado[$n] = "broadcast ".$broadlan."\n";
                }
                else if($n == 18) {
                    $arrayModificado[$n] = "dns-nameservers ".$dns1."
".$dns2"\n";
                }
                else{
```

```
        $arrayModificado[$n] = $meuArray[$n];
    }
    } //fecho o for
    $bufferArquivo = fopen('/etc/network/interfaces.save',"w+");
    for($n=0; $n < count($arrayModificado); $n++) {
        fwrite($bufferArquivo, $arrayModificado[$n]);
    } // fecho o for
    fclose($bufferArquivo);
    }
```

C. Menu DHCP

A tela para a configuração do serviço de DHCP é apresentada na Figura 5. Caso o administrador da rede deseje modificar os parâmetros da sua rede local. Nesse caso, a configuração dos endereços IP deverá seguir a mesma classe de rede configurada anteriormente na tela anterior.

Figura 5 - Tela de configuração dos endereços IP

D. Menu *Firewall*

A tela para a configuração das regras básicas de *Firewall* podem agora ser observada na Figura 6. Nesta tela, observam-se as opções, “Lista de Portas a serem liberadas” e “Destino de Saída”. Nesse caso, o *Firewall* FMUS parte da ideia que toda comunicação entre as duas interfaces de rede é previamente inexistente. Dessa forma, é necessário que se declare que a comunicação interna ocorrerá com todo o mundo exterior (Destino de saída = 0/0). Caso haja outra aplicação *Web*, como por exemplo, um sistema voltado ao setor de Recursos Humanos da empresa e este esteja fora das dependências da empresa. O *Firewall* FMUS, poderá ser configurado de forma a permitir o acesso a somente esse servidor, sem no entanto, conceder acesso dos seus usuários a toda a rede *Internet*. Nesse caso, o “Destino de saída” é o IP onde se localiza a aplicação do RH da empresa.

As portas também são liberadas sob demanda. Isso evita que as regras de NAT sejam feitas de uma forma genérica. Os usuários internos, somente poderão ter acesso às aplicações Web as quais forem permitidas. Assim, pode-se citar como exemplo o fato da ferramenta utilizada como comunicador instantâneo (*GTalk*) o qual usa as portas de comunicação 5222 e 5223. Caso essas não sejam declaradas, não haverá permissão de acesso.

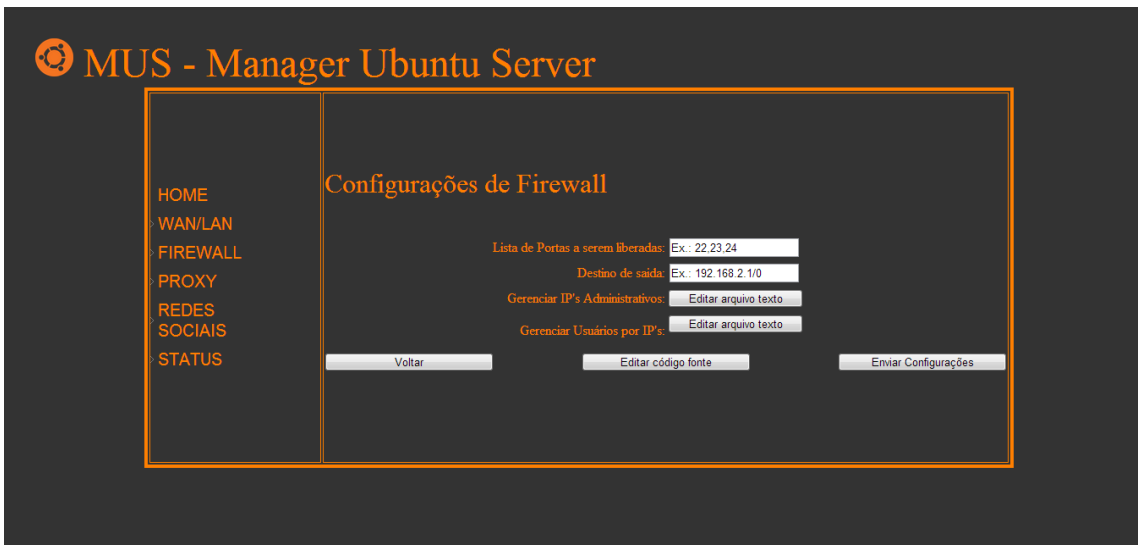

Figura 6 - Configuração da regra básica de Firewall (NAT)

E. Menu Proxy

Considerando que haja alguns endereços IPs destinados a máquinas administrativas ou de pessoas com total liberdade de acesso a rede mundial, esses endereços podem ser adicionados num arquivo, isto é, em uma listagem de IPs os quais fogem a regra geral de bloqueio. O botão “Gerenciar IPs Administrativos” abre um arquivo texto para adição desses endereços. Outro ponto previsto nesse *Firewall*, é que somente os endereços IPs adicionados a lista de IP's dos usuários poderão ter acesso ao endereço de saída (Destino de saída). Por padrão, o acesso é a rede externa foi deixado deliberadamente liberado. O botão “Gerenciar Usuários por IP's” deverá ser acionado para reconfiguração da regra padrão.

Figura 7 - Tela de Configuração do Servidor Proxy

F. Menu SARG

Para os acessos à rede *Internet* é gerado um arquivo contendo os registros de acessos via arquivo de log apresentado na Figura 8. Nesse trabalho foi utilizado a ferramenta SARG (*Squid Analysis Report Generator*) para a tabulação desses logs. Nesse caso, o *Firewall* FMUS, necessita invocar um arquivo no formato HTML gerado pela ferramenta SARG para exibição dos logs gerados. O botão “Gerar relatório” faz a releitura dos logs gerados pelos acessos e cria um novo menu (dentro do relatório SARG) com descrição do período em que o relatório se refere. A codificação do arquivo PHP para a invocação da tela do SARG é apresentada da seguinte forma:

"require('/sarg/index.html');"

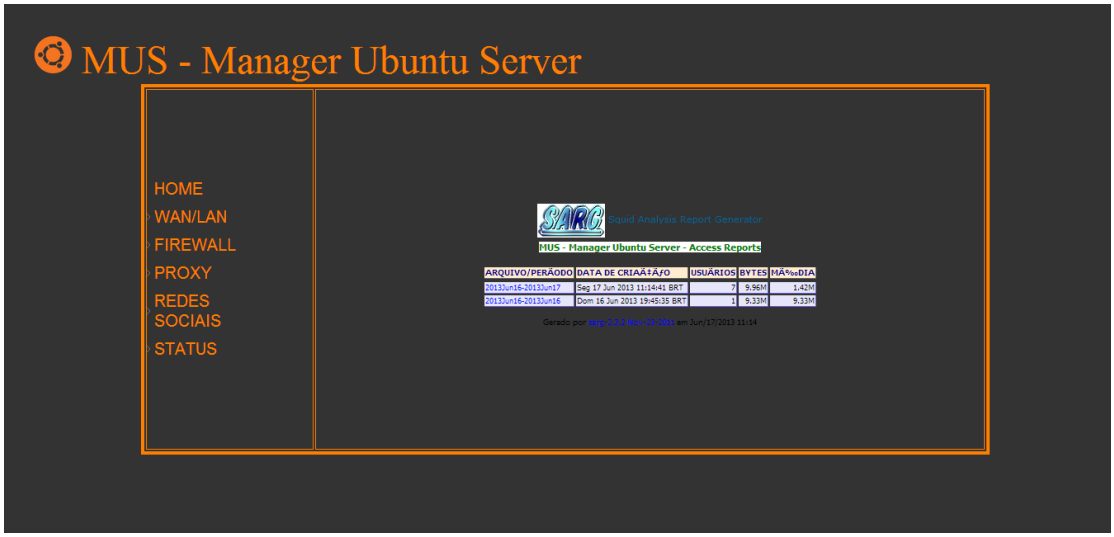

Figura 8 - Relatórios de acessos (Servidor Proxy)

G. Menu Redes Sociais

Para o bloqueio das redes sociais, buscou-se facilitar ao máximo a configuração do *Firewall*. Embora a tela apresentada na Figura 9 seja bastante simples, aquilo que se observa é apenas uma seleção de redes as quais serão ou não bloqueadas. Contudo essas redes são de natureza distribuída, onde não há somente um servidor o qual disponibilize uma determinada rede social. Nesse caso, foi necessário criar um mecanismo o qual pudesse verificar periodicamente quais os servidores de uma determinada rede social os quais estariam ativos.

Figura 9 - Tela de configuração de Bloqueio às Sedes Sociais

Para exemplificar, foi criado um script que executasse os comandos, os quais estão descritos no Quadro 15.

Quadro 15 - Comandos para capturar IP's

```
host connect.facebook.com >
host facebook.com
```

Os endereços IP's obtidos com a resposta desse comando eram então enviados e arquivo texto. Assim, foi possível criar uma lista dinâmica dos endereços dessa rede social. Uma vez obtido essa lista uma regra de *Firewall* utilizando um laço FOR, percorria todo o vetor bloqueando todos os endereços IP's os quais utilizavam a porta 443 (https) para se comunicarem.

O código do Quadro 16 ilustra uma regra de *Firewall* para bloqueio desses endereços IP's. Nesse exemplo, o arquivo ipsfacebook.txt contém a lista de endereços IP's obtidos com o comando host.

Quadro 16 - Exemplo de código para bloqueio de redes sociais

```
for IPREDESSOCIAIS in $(cat /usr/local/Firewall/redessociais_ip.txt); do
    iptables -N FACEBOOK
    iptables -I FORWARD -p tcp -s $LAN -d $IPREDESSOCIAIS -m multiport --dport 80,443 -j DROP
    iptables -I FORWARD -p udp -s $LAN -d $IPREDESSOCIAIS -m multiport --dport 80,443 -j DROP
    iptables -A FACEBOOK -j REJECT
done
```

F. Menu MRTG

Neste Menu apresenta o gráfico do MRTG, demonstrando o consumo de banda da rede.

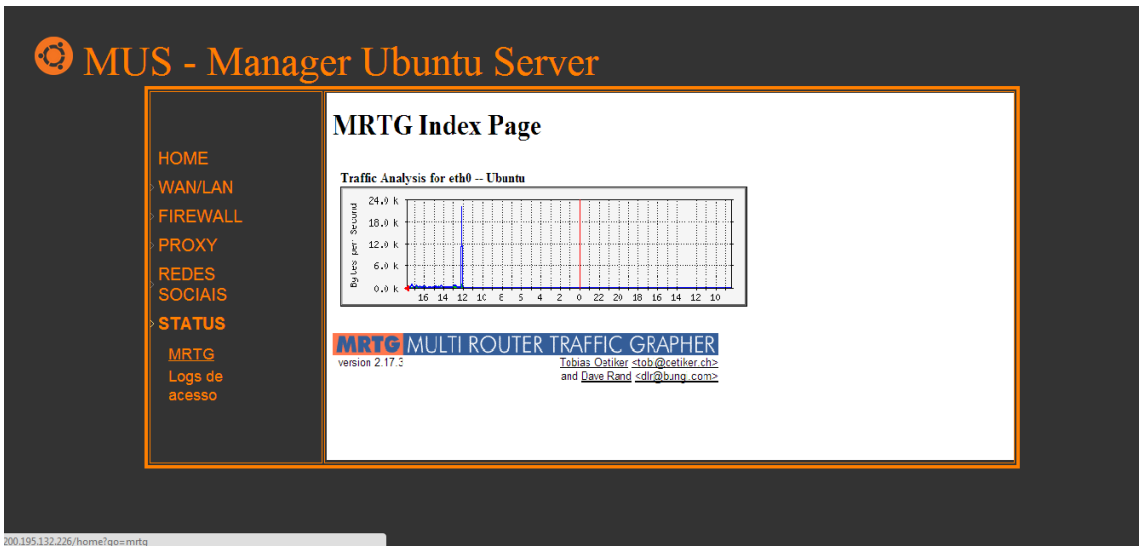

Figura 10 - Gráfico Referente ao Consumo da Banda de Rede

H. Menu "Logs de Acesso"

O menu "Logs de Acesso", permite ao administrador de rede avaliar as conexões via SSH para assim poder identificar possíveis tentativas de intrusão.

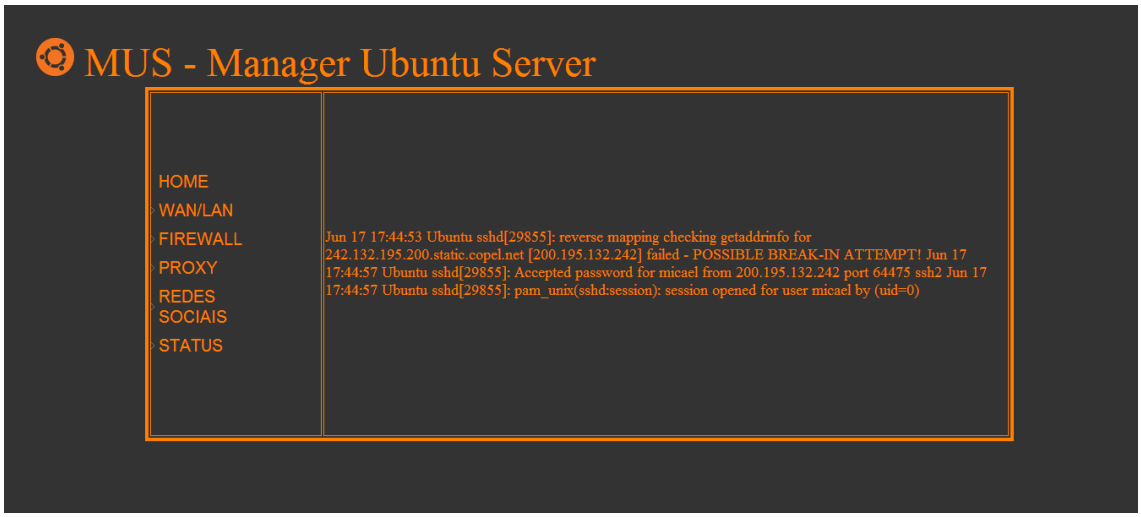

Figura 11 - Registros de Acessos ao Servidor via SSH

concluir

6 CONCLUSÕES E TRABALHOS FUTUROS

Para este trabalho foi inicialmente feito uma pesquisa com diversas outras ferramentas de *Firewall*, baseadas em *software* livre. Dentre as ferramentas estudadas, buscou-se primeiramente encontrar as ferramentas as quais tivessem algo similar ao trabalho apresentado. Uma vez estabelecido os parâmetros de similaridade, buscou-se achar um diferencial entre elas. Nesse caso, o *Firewall* apresentado também denominado por FMUS - Firewall Manager Ubuntu Server, possui o diferencial de ter como base um sistema operacional Ubuntu Server o qual é um dos sistemas que atualmente mais cresce em número de instalações como servidores de rede para pequenas e médias empresas.

Além disso, o FMUS apresenta outro diferencial. As regras de NAT foram criadas individualmente para cada endereço IP e por porta para cada aplicação instalada. Dessa forma, cada *host* somente poderá se conectar com a rede externa uma vez que o seu endereço IP e porta da aplicação a ser acessada estiverem abertas. Além disso, o FMUS permite que hajam comunicações entre a rede interna e externa de forma privilegiada para alguns *hosts*. O FMUS ainda nos apresenta gráficos sobre o consumo da banda e relatórios de acessos de forma simples.

Esse trabalho ainda permite que sua estrutura possa agregar mais recursos, de tal forma a aumentar a sua capacidade em gerencia de redes, deixando até mesmo de abranger somente as redes de pequeno porte. Um exemplo seria um estudo mais a fundo do protocolo SNMP, o qual tem disponibilidade de múltiplas informações dos hosts conectados a rede, permitindo o monitoramento, manutenção e prevenção de erros no ambiente aplicado.

REFERÊNCIA BIBLIOGRÁFICA

BUECHLER ,Christopher M. PINGLE, Jim. pfSense: The Definitive Guide The Definitive Guide to the pfSense Open Source *Firewall* and Router Distribution. Abril 2009.

CISCO SYSTEMS, Inc. "Cisco's PIX *Firewall* Series and Stateful *Firewall* Security. Disponivel em: http://www.cisco.com/warp/public/cc/pd/fw/sqfw500/tech/nat_wp.pdf. Março 2002.

DEMPSTER, Barrie; EATON-LEE, James. *Configuring IPCop Firewalls: Closing Borders with Open Source*. Reino Unido: Packt Publishing, 2006.

GALLO, MICHAEL A., HANCOCK, W. M.: “Comunicação entre Computadores e Tecnologías de Rede”, São Paulo, 2003.

GONÇALVES, Marcus. *Firewall*s: guia completo, trad. Savannah Hartmann. Rio de Janeiro: Ciência Moderna, 2004.

MELO, Sandro. Computação Forense com Software Livre, conceitos, técnicas, ferramentas e Estudos de Casos. Rio de Janeiro, Editora Altabooks, 1ª edição, 2008.

MELO, Sandro. Exploração de Vulnerabilidades em Redes TCP/IP. Rio de Janeiro, Editora Altabooks, 2ª edição, 2006.

NAKAMURA, Emilio Tissato; GEUS, Paulo Lício. Segurança de Redes: em ambientes cooperativos. São Paulo: Novatec, 2007.

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

NPD - Núcleo de Processamento de Dados da UFES - Universidade Estadual

do Espírito Santo. http://www.npd.ufes.br/node/87, acessado em 10/02/2013.

PINHEIRO, José Mauricio Santos. Gerenciamento de Redes de Computadores: Uma Breve Introdução. http://www.projetoderedes.com.br/artigos, agosto 2002.

SANTOS, Wesley Mauricio Barboza dos. SILVA, Michel da. Segurança em *Firewall* Linux com Proxy Server Baseada na Norma de Segurança Brasileira ABNT NBR 17799. revista.ulbrajp.edu.br, acesso em 20 de Janeiro de 2013.

SÊMOLA, Marcos. Gestão da Segurança da Informação - *Uma visão executiva*. São Paulo: Campus, 2003. Páginas 164.

SIMÕES, Agripio**.** OpenSSH – Secure Remote Shell **-** http://www.aprigiosimoes.com.br/2009/10/05/ssh-em-linux-e-unix - Publicado em 2009 - Acessado em 15/06/2013.

TANENBAUM, Andrew S. Redes de Computadores. 3ª Edição. Rio de Janeiro: Campus, 1997. 923 p.

TANNENBAUM, Andrew S. REDES de Computadores. 4ª ed. Rio de Janeiro: Editora Campus, 2003.

TELECO, Inteligência em Telecomunicações,

ULBRICH, Henrique Cesar , Valle, James Della. Universidade Hacker. Editora: Digerati Books, Edição: 6, 2008.

WESSELS, D. Squid The Definitive Guide", Sebastopol – CA – USA, O' Reilly, 2004.

WILLIAMSON, Matt. *pfSense 2 Cookbook* . Packt Pub Limited, 2011.

WOOL, Avishai. Ph.D., Packet Filtering and Stateful *Firewall*s . Department of Electrical Engineering, Tel Aviv University,Computer Technologies and Information Sciences, http://www.eng.tau.ac.il/~yash/hinsec-171.pdf, acessado em 28/03/2013.