UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
DEPARTAMENTO ACADÊMICO DE ELETRÔNICA
CURSO DE ESPECIALIZACÃO EM CONFIGURAÇÃO E GERENCIAMENTO DE SERVIDORES E EQUIPAMENTOS DE REDES

FABIANO MENDONÇA BATISTA

SEGURANÇA DE REDES COM IPTABLES

MONOGRAFIA

CURITIBA
2012

FABIANO MENDONÇA BATISTA

SEGURANÇA DE REDES COM IPTABLES

Monografia apresentada como requisito parcial para obtenção do grau de especialista em Configuração e Gerenciamento de Servidores e Equipamentos de Redes, do Departamento Acadêmico de Eletrônica da Universidade Tecnológica Federal do Paraná.
Orientador: Prof. Augusto Foronda.

CURITIBA
2012

RESUMO

Batista, Fabiano Mendonça. Segurança de redes com iptables. 2012. 52f. Monografia (Especialização em Configuração e Gerenciamento de Servidores e Equipamentos de Redes) – Programa de Pós-Graduação na Universidade Tecnológica Federal do Paraná. Curitiba, 2012.

Atualmente, com o crescimento das redes de computadores, os administradores, empresas, instituições, a sociedade em si tem uma preocupação constante no segmento segurança de redes. Um indivíduo pode usar uma rede para desenvolver um trabalho, fazer pesquisas, buscar conhecimentos ou para obter informações de forma ilícita, prejudicando outras pessoas e, ou instituições.

A idéia deste trabalho é abordar características que envolvem a segurança da informação nas redes de computadores com uma ferramenta chamada firewall, nativo do sistema operacional linux. O emprego dessa ferramenta se tem um melhor gerenciamento da rede, buscando realizar o controle dos dados que o trafegam.

Palavras-chave: Iptables, Linux, Netfilter e Firewall.

ABSTRACT

Batista, Fabiano Mendonça. Network Security with iptables. 2012. 52f. Monograph (Specialization in Configuring and Managing Servers and Network Equipment) - Graduate Program in the Federal Technological University of Paraná. Curitiba, 2012.

Currently, with the growth of computer networks, administrators, businesses, institutions, society itself has a constant concern in the network security segment. An individual can use a network to develop a job, do research, seeking knowledge or information unlawfully harming other people and or institutions.
The idea of this paper is to address characteristics that involve information security in computer networks with a tool called firewall, native linux operating system. The use of this tool has a better network management, seeking to make the control of the data that travels.

Keywords: Iptables, Linux, Netfilter and Firewall.

LISTA DE FIGURAS

SUMÁRIO

1 INTRODUÇÃO

1.1 TEMA

Hoje em dia com o avanço tecnológico é impossível não usar as inúmeras ferramentas oferecidas pelos sistemas computacionais. A partir dessa idéia, empresas, instituições públicas, escolas, universidades, etc, vêm se adequando a esse novo conceito a fim de obter os vários benefícios que eles podem disponibilizar.

Há algumas décadas, a tecnologia da informática entrou num processo de rápida evolução e logo passou a integrar-se no âmbito comercial. Nesse aspecto, os sistemas computacionais passaram a auxiliar bastante nos mais variados processos envolvidos nesse meio. Surgiu então à necessidade de compartilhar recursos, permitir comunicação e concentrar informações comuns a todos os usuários estabelecidos em um mesmo ambiente. Na busca de suprir essa necessidade, surgiram então as redes de computadores.

Com o surgimento das redes, veio à necessidade de adquirir maneiras de proteção para a mesma. A maneira mais simples de proteger uma rede interna é com o isolamento físico da mesma. Desta forma, ninguém de fora será capaz de invadir o sistema sem antes entrar nas dependências físicas da organização. Entretanto, com o crescimento da internet, o isolamento físico tornou-se quase impossível. Para resolver este problema muitas organizações têm usado *firewalls.*

Um *firewall* é um sistema que reforçam a norma de segurança entre uma rede interna segura e uma rede não-confiável como a internet. Os *firewalls* tendem a serem vistos como uma proteção entre a internet e a rede local.

Podendo ser um servidor, um roteador, um computador, um mainframe, uma estação de trabalho *unix* ou a combinação destes, ele é capaz de determinar qual informação ou serviços podem ser acessados de fora e a quem é permitido usar a informação e os serviços de fora. Colocado entre a rede interna e a externa, o *firewall* controla todo o tráfego que passa entre elas, tendo a certeza que este tráfego é aceitável, de acordo com a política de segurança do site, oferecendo uma excelente proteção contra ameaças vindas da rede externa.

1.2 DELIMITAÇÃO DA PESQUISA

A ferramenta abordada nessa pesquisa será o *iptables*, nativo do *linux* e introduzido a partir versão 2.4 do *Kernel* do *GNU/Linux*, com o objetivo de substituir o "*ipchains*" que era utilizado pelas versões 2.2. Este novo *firewall* tem como vantagem ser muito estável (assim como o *ipchains* e *ipfwadm*), confiável e permitir muita flexibilidade na programação de suas regras, mais opções disponíveis para o controle de tráfego e melhor organização devido à organização aprimorada das etapas de roteamento. O *iptables* é um *firewall* em nível de pacotes e funciona baseado em endereços/portas de origem/destino. Ele desempenha suas funções através da comparação de regras, organizadas e armazenadas em tabelas internas, para saber se um pacote tem ou não permissão para adentrar a rede/máquina que está sendo protegida. Em configurações mais restritivas, o pacote é bloqueado e registrado para que o administrador do sistema tenha condições de avaliá-lo posteriormente. O *iptables* também pode ser usado para modificar e monitorar o tráfego da rede, fazer NAT (*Network Address Translation)* , redirecionamento e marcação de pacotes, modificarem a prioridade de pacotes que entram e saem do seu sistema, contagem de *bytes*, dividirem tráfego entre máquinas e criar proteções contra várias técnicas de ataque (anti-spoofing, syn flood, DoS (*Quality of Service*), etc). As possibilidades oferecidas pelos recursos de filtragem *iptables* e a sua eficácia, dependem em grande parte dos conhecimentos do administrador do sistema em relação aos conceitos de funcionamento das redes TCP/IP (*Transmission Control Protocol / Internet Protocol*) e da manipulação precisa das regras que são utilizadas pela ferramenta para fazer a validação de cada pacote que trafega pela rede/máquina protegida por este *firewall.* É necessário que o administrador seja consciente e tenha claro em sua mente o que deseja quais serviços serão protegidos, quais são os endereços que serão aceitos/bloqueados, quais portas terão redirecionamento, etc.

1.3 PROBLEMA

Em uma empresa com 8 (oito) computadores, um switch e um modem ADSL (*Asymmetric Digital Subscriber Line*) precisa urgentemente implementar uma política de segurança para proteção da rede interna para evitar possíveis ataques vindos tanto da rede interna quanto da rede externa (internet). Como atualmente não existe nenhum tipo de segurança de informação, esse seria o maior problema atual da empresa.

Abaixo segue alguns dos problemas levantados que precisam ser vistos:

- A rede precisa ser defendida contra os tipos de ataques mais conhecidos;
- Deve-se liberar o procedimento de troca do endereço IP (*Internet Protocol)* privado do *firewall* para o IP válido, no momento em que ele se conecta com a internet;
- O número de pacotes que chegam a ela por meio do procedimento de checagem de host presente na mesma rede, o *ping* deve ser limitado de modo a não sobrecarregar a conexão nem o sistema;
- Filtrar os dados vindos da internet de forma isolada;
- Permitir que as estações da rede acessem os seguintes serviços disponíveis na internet: *web*, https (*HyperText Transfer Protocol Secure*) e e-mail;
- No servidor há também os serviços SSH *(Secure Shell)* e FTP (*File Transfer Protocol)* que devem ser disponibilizados para as estações locais;
- Gerar um registro das conexões não autorizadas;
- Descartar as conexões não autorizadas com destino às estações da rede, tanto conexões externas e internas;
- Descartar todos os outros tipos de conexões com destino ao *firewall*.

1.4 OBJETIVOS

Abaixo segue os objetivos separados por geral e específico:

1.5 OBJETIVO GERAL

Este trabalho tem como objetivo mostrar maneiras de uso de técnicas de segurança baseadas em *firewall* no sistema operacional *linux*. Serão mostrados também alguns conceitos e políticas de segurança para um bom resultado na segurança de uma rede.

Será apresentado como configurar um *firewall* para proteger toda uma rede, as estações de trabalho, servidores e ainda a disponibilização de internet. Complementarmente, como é o caso do problema citado que servirá de base de estudo neste trabalho. Com o uso de determinadas regras é possível a publicação de quaisquer serviços na internet, como: web, e-

mail, PHP *(Personal Home Page),* FTP, DNS (*Domain Name System)*, etc., a partir de um ou mais servidores que fazem parte de uma rede local.

1.6 OBJETIVOS ESPECÍFICOS

Os objetivos específicos são:

- Definir a política padrão de filtragem com a ação de descartar o que não constar nas regras de liberação;
- Bloquear os sites *facebook* e *orkut;*
- Colocar o Servidor (*web,* FTP, SSH e *e-mail)* numa rede separada utilizando DMZ (D*emilitarized Zone);*
- Proteger a conexão de entrada contra *port scanners*, *trace routers*, ataques e pacotes fragmentados;
- Restringir o número de pacotes ICMP (*Internet Control Message Protocol) ping* em 2 (dois) pacotes por segundo vindos da interface ligada à internet;
- Tratar as conexões de entrada pela interface ligada à internet em uma *chain* à parte;
- Liberar acesso da rede local à internet pelas portas 80 (*web*), 443 (https), 110 (POP3 – *Post Office Protocol*), 25 (SMTP - *Simple Mail Transfer Protocol)* e 53 (DNS) através do mascaramento de seus respectivos endereços IPs;
- Descartar acessos não-autorizados às estações da rede;
- Descartar demais conexões relacionadas ao recurso NAT.
- Liberar a alteração via NAT do IP privado referente à conexão PPPoE *(Point-to-Point Protocol over Ethernet)* antes de ser atribuído o IP válido pelo provedor de acesso à internet;
- Habilitar o roteamento de pacotes entre a rede local e a internet;
- Permitir via NAT, a disponibilização na internet dos serviços do servidor da rede (*web* e e-mail);
- Criar *logs* de tentativas de conexão não autorizadas utilizando a função (-j LOG).

1.7 JUSTIFICATIVA

Com o aumento dos dispositivos conectados em rede, aumentou também a preocupação em proteger suas informações.

Conectar-se à internet sem um firewall é como deixar as chaves do carro no contato, o motor ligado e as portas destravadas enquanto você vai às compras. Embora você possa entrar e sair antes que alguém perceba, também é possível que alguém aproveite a oportunidade. Na internet, os hackers utilizam códigos mal-intencionados, como vírus, worms e cavalos de tróia, para tentar encontrar computadores desprotegidos. O firewall auxilia na proteção da sua máquina contra esses e outros ataques à segurança.

Acredito que nesse trabalho que será desenvolvido irá ajudar muitas pessoas que buscam conhecimento e maneiras de proteger seus dados, o seu estabelecimento de possíveis ataques hackers.

1.8 METODOLOGIA

Para desenvolver esse projeto irei utilizar livros que serão emprestados na biblioteca da instituição de ensino e apostilas disponibilizadas na internet para atingir o objetivo.

2 TEORIAS ESPECÍFICAS DE IPTABLES

Antes de começar a implementar o projeto, temos que conhecer algumas teorias básicas.

O iptables tem por padrão deixar tudo liberado, portanto, ele está utilizando como política padrão o ACCEPT', ou seja, todo e qualquer acesso está liberado.

O firewall iptables tem múltiplas funcionalidades dentro de seus módulos, tabelas e chains. As tabelas indicam exatamente qual o tipo de aplicação vai ser utilizada no firewall. As tabelas padrão são: FILTER, NAT e MANGLE.

As características do *iptables* são:

- Suporte aos protocolos TCP, UDP, ICMP.
- Pode especificar portas de endereço e destino.
- Suporte aos módulos externos, como FTP e IRC *(Internet Relay Chat)*.
- Suporta um número ilimitado de regras por *chains*.
- Pode se criar regras de proteção contra diversos tipos de ataques.
- Suporte para roteamento de pacotes e redirecionamentos de portas.
- Suporta vários tipos de NAT, como o SNAT e DNAT e mascaramento.
- Pode priorizar tráfego para determinados tipos de pacotes.
- Tem suporte a IPV6 *(Internet Protocol version 6)*, através do programa *ip6tables*.

Na figura 1 são apresentadas as principais tabelas (*nat*, *filter* e *mangle*) usadas no *iptables* e suas respectivas *chains*.

TABLE	CHAIN	INTERFACES	
		Entrada (-i)	Saída (-o)
Filter	INPUT	SIM	NÃO
	FORWARD	SIM	SIM
	OUTPUT	NÃO	SIM
Nat	PREROUTING	SIM	NÃO
	POSTROUTING	NÃO	SIM
	OUTPUT	NÃO	SIM
Mangle	INPUT	SIM	NÃO
	FORWARD	SIM	SIM
	OUTPUT	NÃO	SIM
	PREROUTING	SIM	NÃO
	POSTROUTING	NÃO	SIM

Figura 1 – Estrutura de tabelas do iptables [Autoria própria].

2.1 TABELA FILTER

Tabela filter que é o conjunto de regras com finalidades gerais, como bloquear, negar, realizar logs. As regras existentes nesta tabela não têm poder de alterar as configurações dos pacotes. Basicamente todas as regras de filtragem estão nesta tabela, pois ela é de uso geral.

As 3 (três) possíveis chains da tabela filter são:

- *INPUT*: Pacotes cujo destino final é a própria máquina *firewall*.
- *OUTPUT*: Pacotes que saem da máquina *firewall*.
- *FORWARD*: Pacote que atravessa a máquina *firewall*, cujo destino é uma outra máquina. Este pacote não sai da máquina *firewall* e sim de outra máquina da rede ou fonte. Neste caso a máquina *firewall* está repassando o pacote.

2.2 TABELA NAT

As regras da tabela *nat* tem o poder de alterar características de origem ou de destino de um pacote. Como característica de origem entende-se IP de origem ou porta de origem e como características de destino tem-se o IP destino e porta destino. A tabela *nat* possui 3 (três) conjuntos de regras:

- *PREROUTING:* Tratamento do pacote antes de ele ser roteado.

- *POSTROUTING:* Tratamento dado ao pacote após ele ser roteado.
- *OUTPUT:* Pacotes que saem do roteador.

Note que nas regras acimas, temos mais duas ações a SNAT e DNAT:

- SNAT: É utilizada quando queremos alterar o endereço de origem do pacote. Somente a *chain POSTROUTING* pode ser usada na ação SNAT.
- DNAT: É utilizada quando desejamos alterar o endereço de destino do pacote. Está ação é aplicada para fazer redirecionamento de portas, redirecionamento de servidor, *load balance* e *proxy* transparente. As *chains* que podem ser utilizadas para esta ação são *PREROUTING* e *OUTPUT*.
- *REDIRECT*: Pode ser utilizada para fazer redirecionamento de portas. Quando fazemos um redirecionamento de portas usamos o dado --to-port após a ação *REDIRECT*.

2.3 TABELA MANGLE

Essa tabela tem o objetivo de tratar as propriedades do pacote, como prioridade no processamento e marcação. Todas as chains anteriores são aplicáveis na tabela mangle.

Em geral, cada um das *chain* é processado antes do *chain* correspondente na tabela *filter* e *nat* para definir opções especiais para o tráfego (por exemplo, o *chain PREROUTING* da tabela *mangle* é processado antes do *PREROUTING* da tabela *nat*). A *chain* OUTPUT da tabela *mangle* corresponde ao *OUTPUT* da tabela *nat*.

Não iremos abordar e aplicar a tabela *mangle* nesse projeto.

2.4 PORTAS E PROTOCOLOS

Antes de começar a mergulhar nas opções do *iptables*, é importante ressaltar que quanto mais o administrador entender sobre portas e serviços disponíveis na rede, mais rápido ele vai conseguir implementar e manter as regras.

Uma dica bem importante é dar sempre uma olhada (no caso de dúvidas) em /etc/services, pois ele possui a correspondência entre os serviços e portas disponíveis na distribuição.

2.5 REGRAS DE FIREWALL

As regras de *firewall* geralmente são compostas de uma Tabela, Opção, *Chain,* Dados e Ação. Através destes elementos podemos especificar o que fazer com os pacotes.

iptables [- t tabela] [opção] [chain] [dados] -j [ação]

Exemplo:

iptables –A FORWARD –d 192.168.1.1 –j DROP

Tabela: Filter (é a default)
Opção: -A
Chain: FORWARD
Dados: -d 192.168.1.1
Ação: DROP

Na figura 2 temos uma melhor visualização do que foi citado acima:

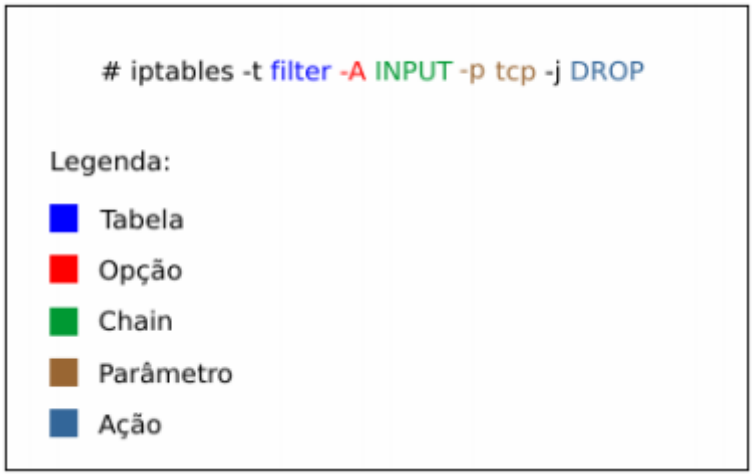

Figura 2 – Exemplo de uma regra iptables [Autoria própria].

Vale observar a respeito da ordem das regras e manejo. Uma delas é que a primeira regra tem prioridade sobre a segunda caso ambas estejam em conflito, veja abaixo o exemplo:

iptables -A FORWARD -p tcp -s 192.168.10.0/24 -j ACCEPT
iptables -A FORWARD -p tcp -s 192.168.10.0/24 -j DROP

A regra que terá validade será a primeira.

Observe a figura 3 abaixo que contém o comando # iptables –t filter –L :

Figura 3 – Comando # iptables –t filter –L sem o "-t filter" [Autor: ODON, Bruno. **Iptables**. Fonte: howtoday.com.br]

É um erro a figura 5 estar com o "-t filter" faltando? Não. O que acontece é que a tabela filter é a padrão do comando iptables e, portanto, não precisa ser declarada quando for utilizada.

2.6 OPÇÕES DE REGRAS DE FIREWALL

Abaixo segue as opções para complementar a tabela:

- -P: Define uma regra padrão;
- -t: Especifica a tabela a ser utilizada;
- -A: Acrescenta uma nova regra as existentes. Este tem prioridade sobre a –P;
- -D: Apaga-se uma regra;
- -L: Lista as regras existentes;
- -F: Apaga todas as regras;
- -I: Insere uma regra nova (prioritária);
- -h: Muito útil, pois mostra a ajuda;

- -n: Não resolve nomes (torna a consulta mais rápida);
- -v: Modo verbose (mais detalhes);
- -R: Substitui uma regra;
- -C: Faz uma checagem das regras existentes;
- -Z: Zera uma regra específica;
- -N: Cria uma nova regra com um nome;
- -X: Exclui uma regra específica pelo seu nome.

2.7 DADOS DE REGRAS DE FIREWALL

Abaixo temos uma breve descrição dos dados para complementar a chain:

- -s: Especifica a origem do pacote. Este pode ser tanto uma rede ou host;
- -d: Especifica o destino do pacote. A sintaxe é a mesma do -s;
- -p: Protocolo usado na regra. Pode ser *tcp, udp, icmp*;
- -i: Interface de entrada, ou seja, placa de rede, modem ou interface de conexão que estará recebendo o pacote a ser tratado;
- -o: Interface de saída. As sintaxes são as mesmas que -i, sendo que neste caso estará enviando o pacote a ser tratado;
- -m: Módulo de será utilizado;
- -j ou –jump: A ação que será tomada;
- !: Exclui determinado argumento;
- --sport: Refere-se a porta de origem. Este deve vir acompanhado das funções -p tcp e -p udp;
- --dport: Refere-se a porta de destino. Assim como a função *–sport*, ela trabalha somente com a -p tcp e -p udp (*User Datagram Protocol)*. A sintaxe é similar a *–Sport*;
- --line-numbers: Número de regras;
- --mac-source: Mac de origem.

2.8 AÇÕES DE REGRAS DE FIREWALL

As principais políticas de ações são:

- ACCEPT: Onde todo e qualquer acesso é liberado.
- REJECT: Onde todo e qualquer acesso é bloqueado, gerando uma mensagem de retorno.
- DROP: Onde todo e qualquer acesso é bloqueado, porém ele não gera nenhuma resposta (exceto para *localhost*).
- LOG: Cria um *log* referente à regra em */var/log/messages*

3 CONFIGURAÇÕES INICIAIS

Na figura 4 apresentada abaixo, temos uma idéia visual do modelo de rede adotada e o objetivo que será alcançado. São mostrada as 7 (sete) estações, 1 (um) servidor de serviços *web*, FTP, SSH e *e-mail*, 1 (um) servidor *firewall*, um *switch*, um roteador ADSL e a internet.

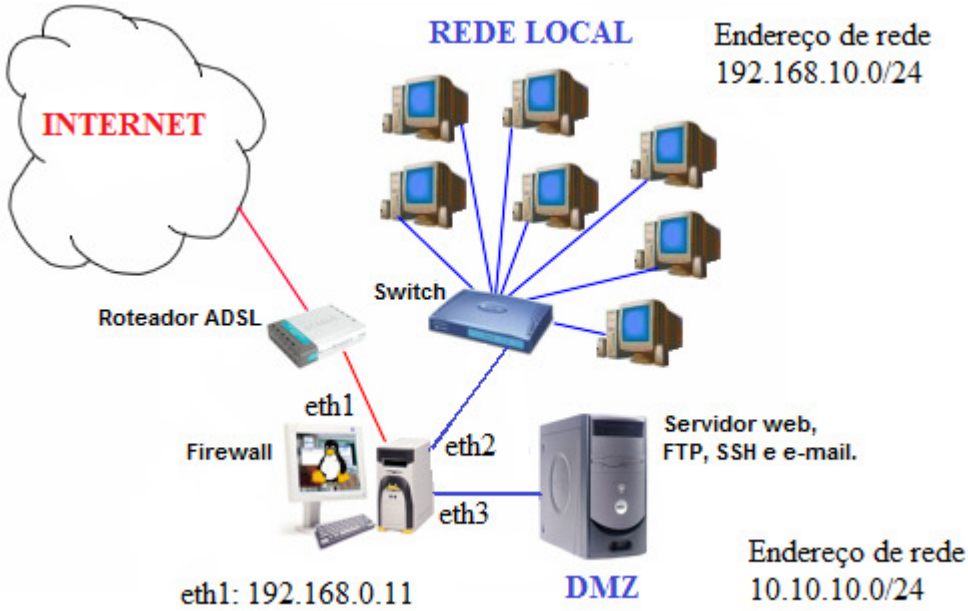

Figura 4 – Rede local adotada para esse projeto [Autoria própria].

Para começarmos a implementar a segurança nessa rede, foi instalado no servidor *firewall* o sistema operacional *Linux Debian* versão 6.0.5 com o *Kernel* 2.6.32-5-amd64.

Logo após foi instalado o *iptables* versão 1.4.8 como o seguinte comando:

root@debian-fmb:/etc # apt-get install iptables

Após a instalação temos que habilitar o redirecionamento da internet entrando no arquivo de configuração abaixo:

root@debian-fmb:/etc# nano sysctl.conf

Foi editado a linha conforme abaixo, salvar e sair:

net.ipv4.ip_forward=0 para net.ipv4.ip_forward=1

Depois aplicamos essa mudança:

root@debian-fmb:/etc# sysctl -p
net.ipv4.ip_forward = 1

Em seguida habilitar o roteamento do kernel:
root@debian-fmb:/ echo 1 > /proc/sys/net/ipv4/ip_forward

3.1 CONFIGURAÇÕES DAS INTERFACES DE REDE

Agora vamos configurar as interfaces de rede *eth1* (externa), *eth2* (rede interna cliente) e a eth3 (rede interna servidor):

root@debian-fmb:/etc/network# nano interfaces

This file describes the network interfaces available on your system
and how to activate them. For more information, see interfaces(5).

The loopback network interface
auto lo
iface lo inet loopback

#Internet (Vinda do modem a cabo com dhcp)
auto eth1
iface eth1 inet dhcp

auto eth2
iface eth2 inet static
address 192.168.10.1
netmask 255.255.255.0

```
auto eth3
iface eth3 inet static
address 10.10.10.1
netmask 255.255.255.0
```

Salve e saia do arquivo, reinicie o serviço para verificar se as configurações estão corretas com o comando:

```
root@debian-fmb:/etc/network# /etc/init.d/networking restart
```

3.2 INSTALANDO O DHCP (Dynamic Host Configuration Protocol)

Para instalação do DHCP aplicamos o comando:

```
root@debian-fmb:/# apt-get install dhcp3-server
```

Após a instalação temos que configurar o dhcpd.conf conforme abaixo:

```
root@debian-fmb:/etc/dhcp# nano dhcpd.conf

#ddns-update-style none;
default-lease-time 600;
max-lease-time 7200;
authoritative;

# CONFIGURACAO DO SERVIÇO DNS
option domain-name-servers 192.168.10.1;


subnet 192.168.10.0 netmask 255.255.255.0
{
 range 192.168.10.2 192.168.10.254;
```

```
 option subnet-mask 255.255.255.0;
 option routers 192.168.10.1;
 option broadcast-address 192.168.10.255;
}

# Aqui o Presidente da empresa tem a preferência pelo endereço IP abaixo:

host presidente
{
 hardware ethernet 08:00:27:4D:36:24;
 fixed-address 192.168.10.20;
 option host-name "presidente";
}
```

Salve a configuração. Para que o DHCP escute somente na rede interna cliente eth2 temos que editar o arquivo DHCP para:

```
root@debian-fmb:/etc/default# nano isc-dhcp-server
INTERFACES="eth2"
```

Após, reiniciar o serviço:

```
root@debian-fmb:/etc/default# /etc/init.d/dhcp3-server restart
```

3.3 CONFIGURANDO O DNS

Após as configurações o nosso compartilhamento de internet já está funcionando. Podemos ligar um switch na porta eth1 do nosso servidor firewall e distribuir internet nas estações, mas os usuários não ficaram satisfeitos, pois ele tem que navegar na internet usando o endereço IP do site como exemplo o 200.154.56.80, que é do site http://www.terra.com.br.

Como não tem como saber o IP de todos os sites que existe no mundo, cada IP tem um nome para ficar mais fácil a navegação e também a memorização.

Através de uma simples configuração iremos resolver esse problema:

root@debian-fmb:/etc# nano resolv.conf

nameserver 8.8.8.8

nameserver 8.8.4.4

Esses endereços IP salvo no *resolv.conf* são servidores DNS gratuitos oferecidos pela Google Inc.

Salva e sai com Ctrl+o e Ctrl+x e torne-o imutável:

chattr +i /etc/resolv.conf

Para reverter:

chattr -i /etc/resolv.conf

4 CONFIGURAÇÕES DO FIREWALL

4.1 ATIVANDO OS MÓDULOS DO IPTABLE

Antes de partirmos para a configuração das regras no iptables, devemos ativar os módulos responsáveis pelo seu funcionamento de acordo com nossa necessidade apresentada. Alguns deles vêm ativados por default em várias distribuições GNU/Linux, mas para garantir que nada saia de errado devemos saber como ativá-los.

Ativando o módulo principal do iptables:

```
# modprobe ip_tables
```

Ativando os módulos das tabelas filter e nat:

```
# modprobe iptable_filter
# modprobe iptable_nat
```

Ativando o módulo reject:

```
# modprobe ipt_REJECT
```

Ativando o módulo multiport:

```
# modprobe ipt_multiport
```

Ativando o módulo responsável pelo mascaramento da conexão:

```
# modprobe ipt_MASQUERADE
```

Ativando o módulo gerador de log do iptables:

```
# modprobe ipt_LOG
```

Nesse momento os módulos iptables necessários já estão ativos. Agora é preciso preparar as tabelas e chains para receber as regras levantadas anteriormente.

4.2 PREPARANDO TABELAS E CHAINS

Abaixo vamos preparar as tabelas filter e nat:

Limpando as regras das tabelas filter e nat:

```
# iptables -t filter -F
# iptables -t nat -F
```

Zerando os contadores das tabelas:

```
# iptables -t filter -Z
# iptables -t nat -Z
```

Abaixo vamos definir a filtragem a cada uma das *chains* das tabelas *filter*. Tudo que não se enquadrar na lista de regras individuais será descartado.

```
# iptables -t filter -P INPUT DROP
# iptables -t filter -P FORWARD DROP
# iptables -t filter -P OUTPUT DROP
```

Através das configurações aplicadas do capítulo 4 até o 4.2, o *iptables* já está preparado para ser utilizado e atender as necessidades da rede de exemplo proposto. Agora devemos implementar a criação das regras de acordo com o que é preciso liberar ou bloquear.

4.3 CONSTRUÇÕES DAS REGRAS NECESSÁRIAS

Com o levantamento de requisitos que foi levantado no exemplo da rede mostrada, as configurações necessárias para se garantir o funcionamento de forma segura serão vistas nesta subseção.

Levando em consideração que 192.168.10.0/24 é a minha rede cliente e 192.168.0.11 é o IP da interface de saída para a internet, vamos dizer para o *kernel* que todo e qualquer pacote que passe pelo *firewall* e tiver como origem a rede local e NÂO (!) tiver como destino sua própria rede, terá seu endereço de IP de origem mudado para 192.168.0.11.

```
# iptables -t nat -A POSTROUTING -s 192.168.10.0/24 ! -d 192.168.10.0/24 -j SNAT
--to 192.168.0.11
```

Reparem que a regra é uma regra baixa (-A). Por que? Porque após o *firewall* tem também uma DMZ com 1 (um) servidor com as aplicações *web*, ftp, ssh e *e-mail* que não tem acesso a internet e eu vou precisar criar SNAT para chegar até eles também. Então, lá vai:

```
# iptables -t nat -I POSTROUTING -s 192.168.10.0/24 -d 10.10.10.0/24 -j SNAT --to
10.10.10.1
```

Reparem que esta regra é alta (prioritária) e, por isso, sera lida por último pelo *kernel*.

Pronto, resolvemos o problema dos clientes em acessar os dois ambientes a rede DMZ com o servidor (*web*, ftp, ssh e *e-mail)* e a saída para a internet, agora precisamos cuidar da questão do DNAT para o servidor (*web*, ftp, ssh e *e-mail)*:

```
# iptables -t nat -I PREROUTING -p tcp -s 0.0.0.0/0.0.0.0 -d 192.168.10.1 --dport 80 -
j DNAT --to-destination 10.10.10.20:80
# iptables -t nat -I PREROUTING -p tcp -s 0.0.0.0/0.0.0.0 -d 192.168.10.1 --dport 443
-j DNAT --to-destination 10.10.10.20:443
# iptables -t nat -I PREROUTING -p tcp -s 0.0.0.0/0.0.0.0 -d 192.168.10.1 --dport 25 -
j DNAT --to-destination 10.10.10.20:25
# iptables -t nat -I PREROUTING -p tcp -s 0.0.0.0/0.0.0.0 -d 192.168.10.1 --dport 110
-j DNAT --to-destination 10.10.10.20:110
```

Garantindo este DNAT para os acessos externos:

iptables -t nat -I PREROUTING -p tcp -s 0.0.0.0/0.0.0.0 -d 192.168.0.11 --dport 80 -j DNAT --to-destination 10.10.10.20:80

iptables -t nat -I PREROUTING -p tcp -s 0.0.0.0/0.0.0.0 -d 192.168.0.11 --dport 443 -j DNAT --to-destination 10.10.10.20:443

iptables -t nat -I PREROUTING -p tcp -s 0.0.0.0/0.0.0.0 -d 192.168.0.11 --dport 25 -j DNAT --to-destination 10.10.10.20:25

iptables -t nat -I PREROUTING -p tcp -s 0.0.0.0/0.0.0.0 -d 192.168.0.11 --dport 110 -j DNAT --to-destination 10.10.10.20:110

Na figura 5 vamos conferir as regras conforme foram citadas acima:

```
root@debianF:~# iptables -t nat -nvL
Chain PREROUTING (policy ACCEPT 1 packets, 239 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 DNAT       tcp  --  *      *       0.0.0.0/0            192.168.0.11        tcp dpt:110 to:10.10.10.20:110
    0     0 DNAT       tcp  --  *      *       0.0.0.0/0            192.168.0.11        tcp dpt:25 to:10.10.10.20:25
    0     0 DNAT       tcp  --  *      *       0.0.0.0/0            192.168.0.11        tcp dpt:443 to:10.10.10.20:443
    0     0 DNAT       tcp  --  *      *       0.0.0.0/0            192.168.0.11        tcp dpt:80 to:10.10.10.20:80
    0     0 DNAT       tcp  --  *      *       0.0.0.0/0            192.168.10.1        tcp dpt:110 to:10.10.10.20:110
    0     0 DNAT       tcp  --  *      *       0.0.0.0/0            192.168.10.1        tcp dpt:25 to:10.10.10.20:25
    0     0 DNAT       tcp  --  *      *       0.0.0.0/0            192.168.10.1        tcp dpt:443 to:10.10.10.20:443
    0     0 DNAT       tcp  --  *      *       0.0.0.0/0            192.168.10.1        tcp dpt:80 to:10.10.10.20:80

Chain POSTROUTING (policy ACCEPT 7 packets, 423 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 SNAT       all  --  *      *       192.168.10.0/24      10.10.10.0/24       to:10.10.10.1
    0     0 SNAT       all  --  *      *       192.168.10.0/24     !192.168.10.0/24     to:192.168.0.11

Chain OUTPUT (policy ACCEPT 7 packets, 423 bytes)
 pkts bytes target     prot opt in     out     source               destination
root@debianF:~#
```

Figura 5 – Conferindo as regras nat aplicadas até o momento [Autoria própria].

Depois de conferido as regras vamos salvar com o nome regras.fw no caminho /root:

iptables-save > /root/regras.fw

Para que as regras entrem em funcionamento automaticamente a cada vez que o servidor *firewal* é reiniciado, vamos incluir uma linha no arquivo “rc.local” no caminho /etc:

iptables-restore < /root/regras.fw
exit 0

Aqui vamos instalar o *sysklogd* que vai no mostrar os *logs* gerados das conexões:

apt-get install sysklogd

Incluir o *log* do *iptables* no arquivo de configuração do servidor *Syslog* conforme a figura 6*:*

nano /etc/syslog.conf
kern.warn -/var/log/iptables.log

Figura 6 – Linha de *log* do *iptables* inserida no /etc/syslog.conf [Autoria própria].

Agora vamos criar a *chain* relacionada eth-input que tratará os pacotes que entrarem pela interface eth1:

iptables -t filter -N eth-input

A seguir veremos as regras que correspondem à tabela *filter* com as *chains* (INPUT, FORWARD, OUTPUT e eth-input). Algumas dessas regras não condizem às necessidades apontadas, foram criadas como um auxílio de sua aplicação prática.

4.4 INPUT

Referência na chain INPUT para que os pacotes vindos da internet sejam analisados separadamente pela *chain* eth-input:

iptables -t filter -A INPUT -i eth1 -j eth-input

Aceitando conexão “loopback”:

iptables -t filter -A INPUT -i lo -j ACCEPT

Pacotes TCP e UDP que tem como destino o *firewall* são aceitos caso apresentarem o estado de conexão (estabelecida e relacionada):

iptables -t filter -A INPUT -p tcp -m state --state ESTABLISHED,RELATED -j ACCEPT

iptables -t filter -A INPUT -p udp -m state --state ESTABLISHED,RELATED -j ACCEPT

Aceitando pacotes ICMP (ping) com tolerância de um pacote por segundo (proteção contra “ping-da-morte”):

iptables -t filter -A INPUT -p icmp -m limit --limit 1/s -j ACCEPT

4.5 FORWARD

Aceitando pacotes ICMP (ping) com tolerância de pacotes por tempo (dois por segundo):

iptables -t filter -A FORWARD -p icmp -s192.168.10.0/24 -m limit --limit 2/s -j ACCEPT

iptables -t filter -A FORWARD -p icmp -s10.10.10.0/24 -m limit --limit 1/s -j ACCEPT

Liberando o repasse de pacotes entre as interfaces (eth1,eth3) e (eth1,eth2) e registrando os dados do tráfego referente a elas. As portas (25, 80, 110 e 443) referem-se respectivamente aos serviços de (SMTP, HTTP, POP3 e HTTPS) que devem estar disponíveis para as redes:

iptables -t filter -A FORWARD -i eth3 -o eth1 -p tcp -m multiport --dports 20,21,22,25,80,110,443 -j ACCEPT

iptables -t filter -A FORWARD -d 10.10.10.0/24 -i eth1 -o eth3 -m state --state ESTABLISHED,RELATED -j ACCEPT

iptables -t filter -A FORWARD -j LOG --log-level warn --log-prefix 'FIREWALL: LAN_DMZ'

Liberando o repasse de pacotes entre as interfaces eth1 e eth2.

iptables -t filter -A FORWARD -i eth2 -o eth1 -p tcp -m multiport --dports 20,21,22,25,80,110,443 -j ACCEPT

iptables -t filter -A FORWARD -d 192.168.10.0/24 -i eth1 -o eth2 -m state --state ESTABLISHED,RELATED -j ACCEPT

Liberando envio de mensagens "net send":

iptables -t filter -A FORWARD -i eth2 -p udp -m multiport --dports 135,137,138 -j ACCEPT

iptables -t filter -A FORWARD -i eth2 -p tcp -m multiport --dports 135,139,445 -j ACCEPT

Liberando acesso ao servidor FTP e SSH as estações da rede:

iptables -t filter -A FORWARD -p tcp -s 192.168.10.0/24 -d 10.10.10.20 --dport 20 -j ACCEPT

iptables -t filter -A FORWARD -p tcp -s 192.168.10.0/24 -d 10.10.10.20 --dport 21 -j ACCEPT

iptables -t filter -A FORWARD -p tcp -s 192.168.10.0/24 -d 10.10.10.20 --dport 22 -j ACCEPT

Aqui vamos bloquear dois sites que não pode ser disponibilizado na rede conforme o comando abaixo:

iptables -I FORWARD -s 192.168.10.0/24 -d facebook.com -p tcp -dport 80 -j REJECT

iptables -I FORWARD -s 192.168.10.0/24 -d facebook.com -p tcp -dport 443 -j REJECT

iptables -I FORWARD -m string --algo bm --string ‘facebook.com’ -j DROP

iptables -I FORWARD -m string --algo bm --string ‘orkut.com’ -j REJECT

A diferença marcante entre os alvos DROP e REJECT é que com o DROP não há mensagem alguma de erro para o cliente que tentou o acesso, enquanto que com o alvo REJECT apresenta uma mensagem ICMP de erro que é enviada para o cliente.

4.6 OUTPUT

Conexões TCP e UDP que forem originadas pelo firewall são aceitas:

#iptables -t filter -A OUTPUT -p tcp -m state --state NEW,ESTABLISHED,RELATED -j ACCEPT

#iptables -t filter -A OUTPUT -p udp -m state --state NEW,ESTABLISHED,RELATED -j ACCEPT

Aceitando conexão “loopback” de saída do firewall a ele mesmo:

iptables -t filter -A OUTPUT -o lo -j ACCEPT

4.7 ETH-INPUT

Estabelecendo um limite para o número de pacotes “ping” (um por segundo) vindos da internet:

iptables -t filter -A eth-input -p icmp -m limit --limit 1/s -j ACCEPT

Registra tentativas de conexão não-autorizadas vindas da internet:

iptables -t filter –I eth-input -p udp -s 0.0.0.0/0.0.0.0 -m multiport --dport 53 -j LOG --log-level warn --log-prefix ‘[FIREWALL: dns]’

iptables -t filter -I eth-input -p tcp -s 0.0.0.0/0.0.0.0 -m multiport --dport 113 -j LOG --log-level warn --log-prefix ‘[FIREWALL: identd]’

iptables -t filter -I eth-input -p udp -s 0.0.0.0/0.0.0.0 -m multiport --dport 111 -j LOG --log-level warn --log-prefix ‘[FIREWALL: rpc]’

iptables -t filter -A eth-input -p tcp -s 0.0.0.0/0.0.0.0 -m multiport --dport 111 -j LOG --log-level warn --log-prefix ‘[FIREWALL: rpc]’

iptables -t filter -A eth-input -p tcp -s 0.0.0.0/0.0.0.0 -m multiport --dport 137:139 -j LOG --log-level warn --log-prefix ‘[FIREWALL: samba]’

iptables -t filter -A eth-input -p udp -s 0.0.0.0/0.0.0.0 -m multiport --dport 137:139 -j LOG --log-level warn --log-prefix ‘[FIREWALL: samba]’

A seguir veremos às regras tabela nat com as *chains* (POSTROUTING, PREROUTING e OUTPUT). Na subseção 4.3 também foram vistos regras de POSTROUTING e PREROUTING.

4.8 POSTROUTING

Mascaramento de IP liberando conexão da rede local para a internet com restrição aos serviços (SMTP, POP3, HTTP e HTTPS):

iptables -t nat -A POSTROUTING -p tcp -m multiport --dports 25,80,110,443 -s 192.168.10.0/24 -o eth1 -j MASQUERADE

iptables -t nat -A POSTROUTING -s 10.10.10.0/255.255.255.0 -o eth1 -j MASQUERADE

iptables -t nat -A POSTROUTING -o eth1 -j MASQUERADE

4.9 PREROUTING

Liberando conexão da rede local à internet limitada aos serviços (SMTP, POP3, HTTP e HTTPS):

```
# iptables -t nat -A PREROUTING -i eth1 -p tcp -s 192.168.0.11 --dport 25 -j ACCEPT
# iptables -t nat -A PREROUTING -i eth1 -p tcp -s 192.168.0.11 --dport 80 -j ACCEPT
# iptables -t nat -A PREROUTING -i eth1 -p tcp -s 192.168.0.11 --dport 110 -j ACCEPT
# iptables -t nat -A PREROUTING -i eth1 -p tcp -s 192.168.0.11 --dport 443 -j ACCEPT
```

Registrando todas as outras tentativas vindas da internet às estações locais:

```
# iptables -t nat -A POSTROUTING -o eth2 -d 192.168.10.0/24 -j LOG --log-level warn --log-prefix '[FIREWALL: SNAT NAO ENCONTRADO.]'
```

Por fim, para que as estações da rede tenham o firewall como seu gateway de rede, é necessário fazer a seguinte configuração em cada uma delas:

```
# route add default gw 192.168.10.1
```

Tudo que foi citado a partir do sub-ítem 4.1 estão num arquivo *script* chamado "regras.fw". Esse *script* foi criado para ser executado automaticamente na inicialização do servidor *firewall.*

Vamos salvar as regras que foram executadas conforme o comando abaixo:

```
# iptables-save > /root/regras.fw
```

5 TESTES REALIZADOS

Depois de aplicada a lógica do que foi proposto no capítulo 4, foram feitos testes baseados em diversos tipos de possibilidades além das que foram determinadas pela solução apresentada ao modelo de rede.

Nesses testes foram usados um servidor *firewall* e um servidor (*web, email,* ftp e ssh) munido com sistema operacional *Linux Debian* versão 6.0.5 com o *Kernel* 2.6.32-5-amd64 juntamente com outro microcomputador usando o sistema operacional *Windows XP*.

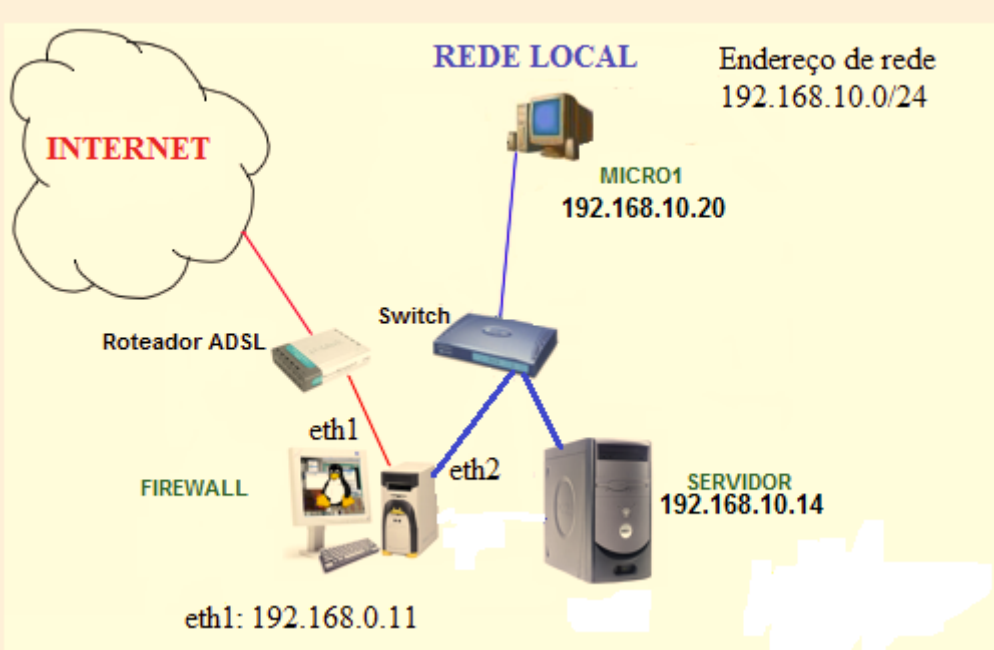

Figura 7 – Topologia da rede testada [Autoria própria].

Na tabela 1 temos a configuração dos computadores integrantes da rede. A representação da topologia pode ser observada na figura 7 mostrada acima.

Host	Endereço IP	Interfaces	Sistema Operacional	Gateway
FIREWALL	eth1: 192.168.0.11 eth2: 192.168.10.1	eth1: Externa eth2: Interna	GNU/Linux Debian 6.0.5	192.168.0.1
MICRO1	192.168.10.20	eth0	Windows XP SP3	192.168.10.1
MICRO2	192.168.10.14	eth1	GNU/Linux Debian 6.0.5	192.168.10.1

Tabela 1 – Configuração dos computadores e da rede testada [Autoria Própria].

TESTE 1: Configurou-se uma regra que descarta quaisquer pacotes ICMP oriundos do MICRO1 da rede. A seguir, a regra foi modificada para rejeitar tais pacotes emitindo uma mensagem de resposta especificando que a rede de destino não foi alcançada. As seguintes regras foram configuradas:

```
# iptables -t filter -A INPUT -i eth2 -s 192.168.10.20 -p icmp -j DROP
# iptables -t filter -D INPUT 1
# iptables -t filter -A INPUT -i eth2 -s 192.168.10.20 -p icmp -j REJECT --reject-with icmp-port-unreachable
```

Resultado obtido: Realizando-se um teste *ping* a partir da MICRO1, as mensagens de erro apresentadas referentes às regras estão relacionadas no Apêndice B. O teste *ping* mostrou que o *firewall* realmente não aceitou os pacotes.

TESTE 2: Nesse teste foi bloqueado a conexão SSH do MICRO1 usando a porta 22 com o comando mostrado abaixo:

```
# iptables -A INPUT -p tcp -s 192.168.10.20 --dport 22 -m state --state NEW -j DROP
```

Resultado obtido: Ao tentar realizar uma conexão por meio do cliente SSH a partir do MICRO1, não se obteve êxito. Tentou-se então, realizar uma conexão a partir do *firewall* para a MICRO1 utilizando a mesma porta do serviço SSH (22), resultando em êxito. O teste de conexão via SSH mostrou que o *firewall* recusou a nova conexão originada pelo cliente SSH e permitiu que a conexão de saída a partir do *firewall* fosse realizada sem problemas. A mensagem de erro apresentada ao MICRO1 foi: *"Network error: Connection time out"*.

TESTE 3: Configurou-se uma regra para registrar *logs* das tentativas de acesso à porta 80 a partir do MICRO2. Em seguida, criou-se outra regra a fim de descartar qualquer tentativa de acesso à porta 80 originada pela mesma. As regras configuradas foram:

```
# iptables -A FORWARD -p tcp -s 192.168.10.14 --dport 80 -j LOG --log-level warn --log-prefix '[FIREWALL: HTTP]'
```

```
# iptables -A FORWARD -p tcp -s 192.168.10.14 --dport 80 -j DROP
```

Resultado obtido: Realizaram-se tentativas de conexão usando o protocolo HTTP, não obtendo sucesso. As mensagens registradas foram gravadas no arquivo (/var/log/iptables.log) e constam no Apêndice D. O teste mostrou que a conexão foi realmente bloqueada e registrada em *logs*, seguindo os critérios estabelecidos no *firewall*.

6 CONSIDERAÇÕES FINAIS

Como visto em todo o capítulo, quanto melhor for o planejamento, a configuração, o levantamento de requisitos, melhor será a confiabilidade da rede amparada pelo *firewall*. O sucesso da implementação de uma rede segura por meio deste recurso, depende unicamente de uma boa previsão de incidentes e uma cuidadosa interpretação dos fatores que devem ser considerados para que se garanta a disponibilização de acesso, recursos e serviços dos quais uma organização pode necessitar.

A facilidade de utilização do aplicativo *iptables* e a intuitiva organização dos elementos de composição das regras se caracterizam como fatores bastante positivos e em conformidade à sua ampla funcionalidade.

De uma maneira geral, foram apresentadas algumas das mais relevantes situações que comumente se encontram num contexto de uma rede de computadores. Em uma empresa que deseja proteger seus dados, expandir suas possibilidades de conexão, diminuir o uso de recursos que geram gastos, o uso de um aplicativo *firewall* é muito importante.

Com o estudo realizado e descrito nas seções que compõem este processo, espera-se alcançar uma evolução do conhecimento relacionado a esse tema, principalmente por terem sido abordadas de forma pertinente, algumas das informações essenciais para sua total compreensão.

6.1 CONCLUSÃO

Em virtude das adversidades constantes às quais são submetidas às redes de computadores, surgiu-se a necessidade de abordar neste trabalho a utilização e as características de uma das ferramentas mais importantes na busca pelo combate dos problemas existentes nesse contexto, originados geralmente por terceiros, como é o caso dos *hackers*, *crackers*, e outros indivíduos com comportamentos análogos.

Este estudo mostrou as características para se implementar uma rede segura e buscou um entendimento de quais parâmetros devemos considerar nessa implementação.

Os testes realizados comprovaram a eficiência do aplicativo *iptables*, que é a interface do usuário com o *framework firewall* nativo do *kernel linux* (*netfilter*), dentro dos parâmetros e do nível de complexidade dos testes descritos e tomados como exemplo.

Concluiu-se por meio deste estudo, que o *firewall iptables* pode se comportar de maneira muito eficiente e estável, mas, mesmo com seus poderosos recursos, não pode ser utilizado como única forma de proteção contra intrusões, já que não é possível somente com o mesmo, realizar um controle mais apurado das informações por ele analisadas.

Um fator que influência nessa limitação é que o *iptables* não possui como padrão a propriedade de verificar o conteúdo dos pacotes e por isso é incapaz de conseguir determinar a existência de códigos maliciosos dentro dos pacotes de dados, apesar de já existir módulos (add-ons) que podem ser carregados no kernel e que se dispõe a resolverem esse problema. Isso ocorre em razão da invasão de privacidade, caracterizada pelo acesso ao conteúdo das informações que são gerenciadas pelo *firewall*. Mesmo assim, o objetivo ao qual o aplicativo *iptables* se propõe a alcançar, é atingido com boa eficácia, principalmente pela rápida e simples interação do mesmo com o *kernel* do sistema *linux*.

As lições aprendidas durante este estudo apresentam-se com toda certeza, como algo que propiciou a reflexão de uma gama de fatores que devem ser considerados na questão da segurança de redes. Portanto, este trabalho é importante não só no que diz respeito ao tema abordado, mas, sobretudo, a todas as outras informações que integram a parte teórica, necessária para a sustentação de seu foco principal.

APÊNDICE A

Todo o desenvolvimento do trabalho *iptables* foram salvo com o comando:

iptables-save > /root/regras.fw

Para uma melhor visualização e entendimento do que foi proposto pelo trabalho, foi criado o arquivo *"firewall"* na pasta */root,* no qual foi dado permissão de execução:

chmod +x firewall

Para fazer funcionar esse *script* manualmente, executamos o seguinte comando abaixo:

./firewall

Segue abaixo na figura 8 o comando depois de ser executado:

Figura 8 – Execução do comando "# ./firewall" [Autoria própria].

Segue abaixo o que foi mostrado após dar um *cat* no arquivo *firewall:*

```
#!/bin/bash

###Limpando regras###
iptables –t filter -F
iptables - t nat  -F
```

```
###Zerando contadores###
iptables –t filter -X
iptables -t nat -X
iptables –tfilter -Z
iptables -t nat -Z

###Ativação de Módulos###
###Módulo principal###
modprobe ip_tables

###Módulos das tabelas filter e nat###
modprobe iptable_filter
modprobe iptable_nat

###Módulo reject###
modprobe ipt_REJECT

###Módulo multiport###
modprobe ipt_multiport

###Módulo responsável pelo mascaramento da conexão###
modprobe ipt_MASQUERADE

###Módulo gerador de log###
modprobe ipt_LOG

#######################Política padrão##########################
###Descartar tudo o que não se enquadrar na lista de regras individuais###
iptables -P INPUT DROP
iptables -P FORWARD DROP
iptables -P OUTPUT DROP
```

###Levando em consideração que 192.168.10.0/24 é a minha rede cliente e ###192.168.0.11 é o IP da interface de saída para a internet, vamos dizer para o kernel que ###todo e qualquer pacote que passe pelo firewall e tiver como origem e rede local e NÂO (!) ###tiver como destino sua própria rede, terá seu endereço de IP de origem mudado para ###192.168.0.11###

iptables -t nat -A POSTROUTING -s 192.168.10.0/24 ! -d 192.168.10.0/24 -j SNAT --to 192.168.0.11

###Reparem que a regra é uma regra baixa (-A). Por que? Porque após o firewall tem ###também uma DMZ com 1 (um) servidor com as aplicações web, ftp, ssh e e-mail que não ###tem acesso a internet e eu vou precisar criar SNAT para chegar até eles também. iptables -t nat -I POSTROUTING -s 192.168.10.0/24 -d 10.10.10.0/24 -j SNAT --to 10.10.10.1

#######
#DNAT#
#######

iptables -t nat -I PREROUTING -p tcp -s 0.0.0.0/0.0.0.0 -d 192.168.10.1 --dport 80 -j DNAT --to-destination 10.10.10.20:80

iptables -t nat -I PREROUTING -p tcp -s 0.0.0.0/0.0.0.0 -d 192.168.10.1 --dport 443 -j DNAT --to-destination 10.10.10.20:443

iptables -t nat -I PREROUTING -p tcp -s 0.0.0.0/0.0.0.0 -d 192.168.10.1 --dport 25 -j DNAT --to-destination 10.10.10.20:25

iptables -t nat -I PREROUTING -p tcp -s 0.0.0.0/0.0.0.0 -d 192.168.10.1 --dport 110 -j DNAT --to-destination 10.10.10.20:110

##
#Garantindo este DNAT para os acessos externos#
##

iptables -t nat -I PREROUTING -p tcp -s 0.0.0.0/0.0.0.0 -d 192.168.0.11 --dport 80 -j DNAT --to-destination 10.10.10.20:80

iptables -t nat -I PREROUTING -p tcp -s 0.0.0.0/0.0.0.0 -d 192.168.0.11 --dport 443 -j DNAT --to-destination 10.10.10.20:443

```
iptables -t nat -I PREROUTING -p tcp -s 0.0.0.0/0.0.0.0 -d 192.168.0.11 --dport 25 -j DNAT --to-destination 10.10.10.20:25
iptables -t nat -I PREROUTING -p tcp -s 0.0.0.0/0.0.0.0 -d 192.168.0.11 --dport 110 -j DNAT --to-destination 10.10.10.20:110

echo 'PREROUTING inicial                                    OK!!!'

###Criando a chain relacionada eth-input que tratará os pacotes que entrarem pela ###interface eth1###
iptables -t filter -N eth-input

######################
#INPUT; INPUT; INPUT#
######################
###Referência na chain INPUT para que os pacotes vindos da internet sejam ###analisados separadamente pela chain eth-input###
iptables -A INPUT -i eth1 -j eth-input

###Aceitando conexão loopback de entrada firewall à ele mesmo###
iptables -A INPUT -i lo -j ACCEPT

###Pacotes TCP e UDP que tem como destino o firewall são aceitos caso ###apresentarem uma conexão estabelecida e relacionada###
iptables -A INPUT -p tcp -m state --state ESTABLISHED,RELATED -j ACCEPT
iptables -A INPUT -p udp -m state --state ESTABLISHED,RELATED -j ACCEPT

###Aceitando pacotes ICMP (ping) com tolerância de um pacote por segundo ###(proteção contra ping-da-morte)###
iptables -A INPUT -p icmp -m limit --limit 1/s -j ACCEPT

echo 'INPUT                                                 OK!!!'
```

```
#####################################
#FORWARD; FORWARD; FORWARD#
#####################################
###Aceitando pacotes ICMP (ping) com tolerância de pacotes por tempo (dois por ###segundo)###
iptables -t filter -A FORWARD -p icmp -s 192.168.10.0/24 -m limit --limit 2/s -j ACCEPT
iptables -t filter -A FORWARD -p icmp -s 10.10.10.0/24 -m limit --limit 2/s -j ACCEPT

###Liberando o repasse de pacotes entre as interfaces eth1 e eth3 e registrando os ###dados do tráfego referente elas. As portas 25, 80, 110 e 443 devem estar disponíveis para a ###rede local###
iptables -t filter -A FORWARD -i eth3 -o eth1 -p tcp -m multiport --dports 20,21,22,25,80,110,443 -j ACCEPT
iptables -t filter -A FORWARD -d 10.10.10.0/24 -i eth1 -o eth3 -m state --state ESTABLISHED,RELATED -j ACCEPT

iptables -t filter -A FORWARD -i eth2 -o eth1 -p tcp -m multiport --dports 20,21,22,25,80,110,443 -j ACCEPT
iptables -t filter -A FORWARD -d 192.168.10.0/24 -i eth1 -o eth2 -m state --state ESTABLISHED,RELATED -j ACCEPT

iptables -t filter -A FORWARD -j LOG --log-level warn --log-prefix '[FIREWALL:FORWARD]'

echo 'Repasse de pacotes eth1 e eth3                    OK!!!'

###Liberando envio de mensagens (net send) entre as estações da rede###
iptables -A FORWARD -i eth2 -p udp -m multiport --dports 135,137,138 -j ACCEPT
iptables -A FORWARD -i eth2 -p tcp -m multiport --dports 135,139,445 -j ACCEPT

echo 'Net Send                                          OK!!!'
```

```
###Liberando acesso ao servidor FTP e SSH à todas as estações da rede###
iptables -t filter -A FORWARD -p tcp -s 192.168.10.0/24 -d 10.10.10.20 --dport 20 -j ACCEPT
iptables -t filter -A FORWARD -p tcp -s 192.168.10.0/24 -d 10.10.10.20 --dport 21 -j ACCEPT
iptables -t filter -A FORWARD -p tcp -s 192.168.10.0/24 -d 10.10.10.20 --dport 22 -j ACCEPT

echo 'SSH e FTP                                                    OK!!!'

###Aqui vamos bloquear dois sites que não pode ser disponibilizado na rede conforme ###o comando abaixo###
iptables -I FORWARD -m string --algo bm --string 'facebook.com' -j DROP
iptables -I FORWARD -m string --algo bm --string 'orkut.com' -j REJECT

echo 'Sites DROP                                                   OK!!!'

############################
#OUTPUT; OUTPUT; OUTPUT#
############################
###Conexões TCP e UDP que forem originadas pelo Firewall são aceitas###
iptables -t filter -A OUTPUT -p tcp -m state --state NEW,ESTABLISHED,RELATED -j ACCEPT
iptables -t filter -A OUTPUT -p udp -m state --state NEW,ESTABLISHED,RELATED -j ACCEPT

echo 'TCP e UDP originados                                         OK!!!'

###Aceitando conexão loopback de saída do Firewall ele mesmo###
iptables -A OUTPUT -o lo -j ACCEPT

echo 'OUTPUT                                                       OK!!!'
```

```
        ############################
        #eth-input; eth-input; eth-input#
        ############################
        ###Estabelecendo um limite para o número de pacotes ping (um por segundo) vindos
###da internet###
        iptables -A eth-input -p icmp -m limit --limit 1/s -j ACCEPT

        echo 'ICMP p/s                                                    OK!!!'

        ###Registra tentativas de conexão não-autorizadas vindas da internet###
        iptables -t filter -I eth-input -p udp -s 0.0.0.0/0.0.0.0 -m multiport --dport 53 -j LOG --
log-level warn --log-prefix '[FIREWALL:dns]'

        iptables -t filter -I eth-input -p tcp -s 0.0.0.0/0.0.0.0 -m multiport --dport 113 -j LOG --
log-level warn --log-prefix '[FIREWALL:identd]'

        iptables -t filter -I eth-input -p udp -s 0.0.0.0/0.0.0.0 -m multiport --dport 111 -j LOG -
-log-level warn --log-prefix '[FIREWALL:rpc]'
        iptables -t filter -I eth-input -p tcp -s 0.0.0.0/0.0.0.0 -m multiport --dport 111 -j LOG --
log-level warn --log-prefix '[FIREWALL:rpc]'

        iptables -I eth-input -p tcp -s 0.0.0.0/0.0.0.0 -m multiport --dport 137,139 -j LOG --
log-level warn --log-prefix '[FIREWALL:samba]'
        iptables -I eth-input -p udp -s 0.0.0.0/0.0.0.0 -m multiport --dport 137,139 -j LOG --
log-level warn --log-prefix '[FIREWALL:samba]'

        echo 'LOG`s                                                       OK!!!'

        ################################################
        #POSTROUTING; POSTROUTING; POSTROUTING#
        ################################################
        ###Mascaramento de IP liberando conexão da rede local para a Internet com restrição
###aos serviços(SMTP, POP3, HTTP e HTTPS)###
```

```
iptables -t nat -A POSTROUTING -p tcp -m multiport --dports 25,80,110,443 -s 192.168.10.0/24 -o eth1 -j MASQUERADE
iptables -t nat -A POSTROUTING -s 10.10.10.0/255.255.255.0 -o eth1 -j MASQUERADE
iptables -t nat -A POSTROUTING -o eth1 -j MASQUERADE

echo 'MASQUERADE                                                    OK!!!'

##########################################
#PREROUTING; PREROUTING; PREROUTING#
##########################################
###Liberando conexão da rede local à internet limitada aos serviços (SMTP, POP3, ###HTTP e HTTPS)###
iptables -t nat -A PREROUTING -i eth1 -p tcp -s 192.168.0.11 --dport 25 -j ACCEPT
iptables -t nat -A PREROUTING -i eth1 -p tcp -s 192.168.0.11 --dport 80 -j ACCEPT
iptables -t nat -A PREROUTING -i eth1 -p tcp -s 192.168.0.11 --dport 110 -j ACCEPT
iptables -t nat -A PREROUTING -i eth1 -p tcp -s 192.168.0.11 --dport 443 -j ACCEPT

echo 'PREROUTING                                                    OK!!!'

###Registrando todas as outras tentativas externas de conexão às estações locais###
iptables -t nat -A POSTROUTING -o eth2 -d 192.168.10.0/24 -j LOG --log-level warn --log-prefix '[FIREWALL:SNAT not found!]'
```

APÊNDICE B

Resultado dos teste como esse comando:

```
# iptables -t filter -A INPUT -i eth2 -s 192.168.10.20 -p icmp -j DROP
```

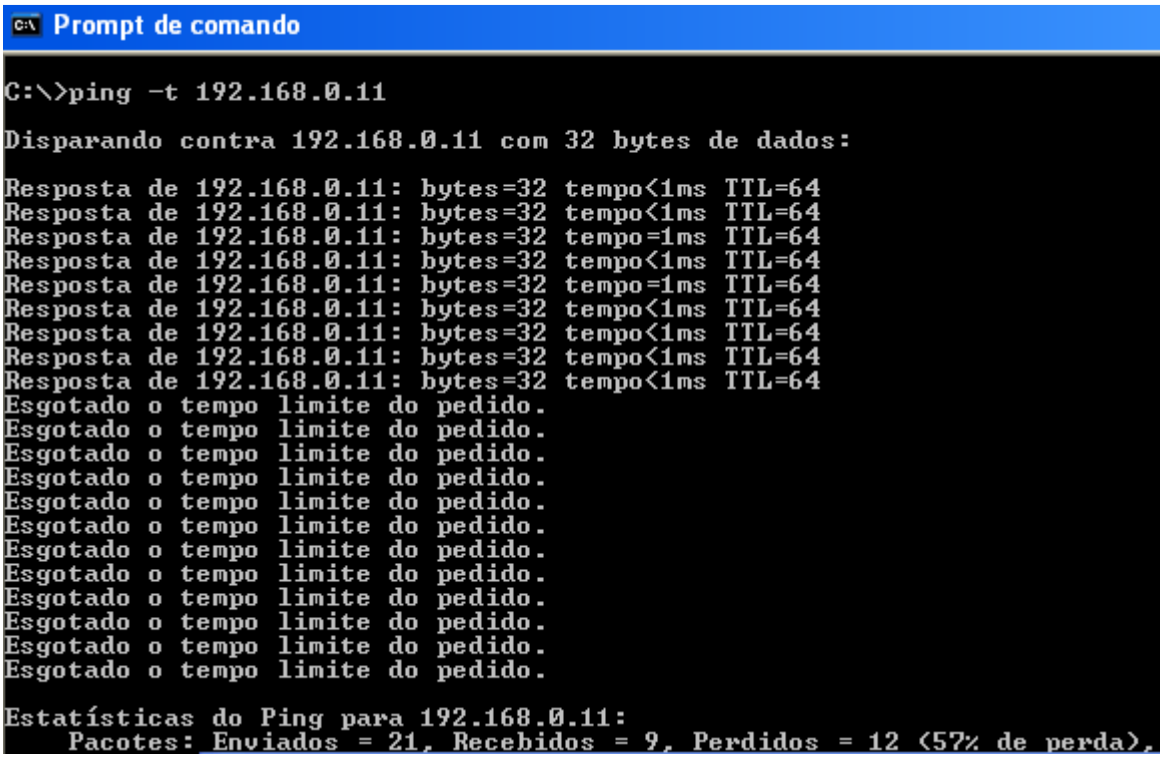

Figura 9 – Teste ping usando DROP [Autoria própria].

Primeiramente comecei usando o teste *ping* e logo após executei o comando no *firewall,* imediatamente o teste *ping* foi interrompido com a seguinte mensagem: “Esgotado o tempo limite do pedido.”

Depois de excluir a regra acima, foi executado no *firewall* o comando que emite uma mensagem de resposta. Note a mensagem “Porta de destino inacessível.” na figura 10 abaixo:

iptables -t filter -A INPUT -i eth2 -s 192.168.10.20 -p icmp -j REJECT --reject-with icmp-port-unreachable

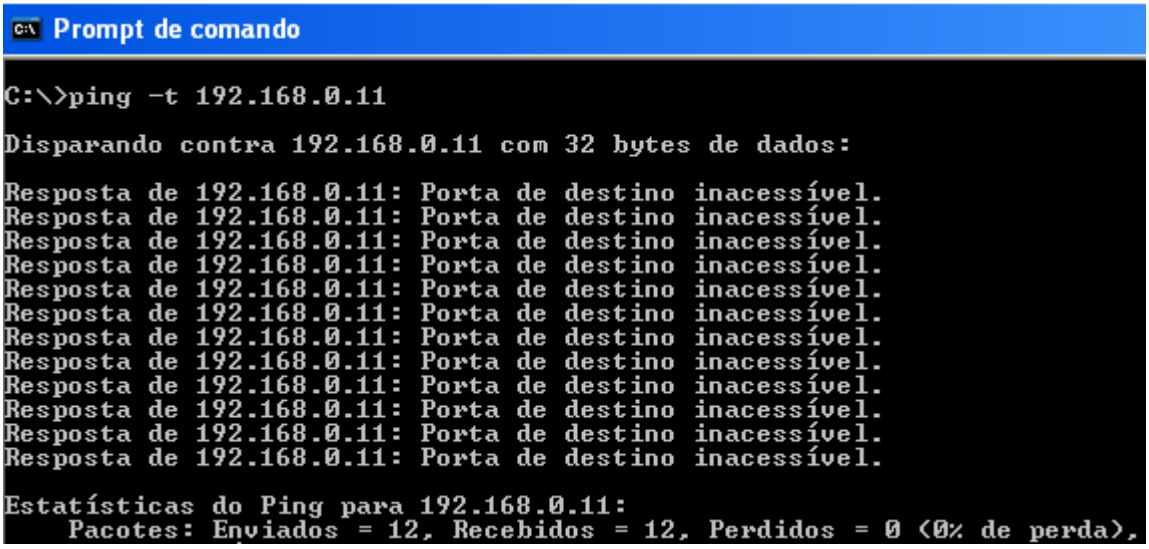

Figura 10 – Teste ping usando REJECT com mensagem ao cliente [Autoria própria].

APÊNDICE C

Segue o teste SSH usando o DROP no comando:

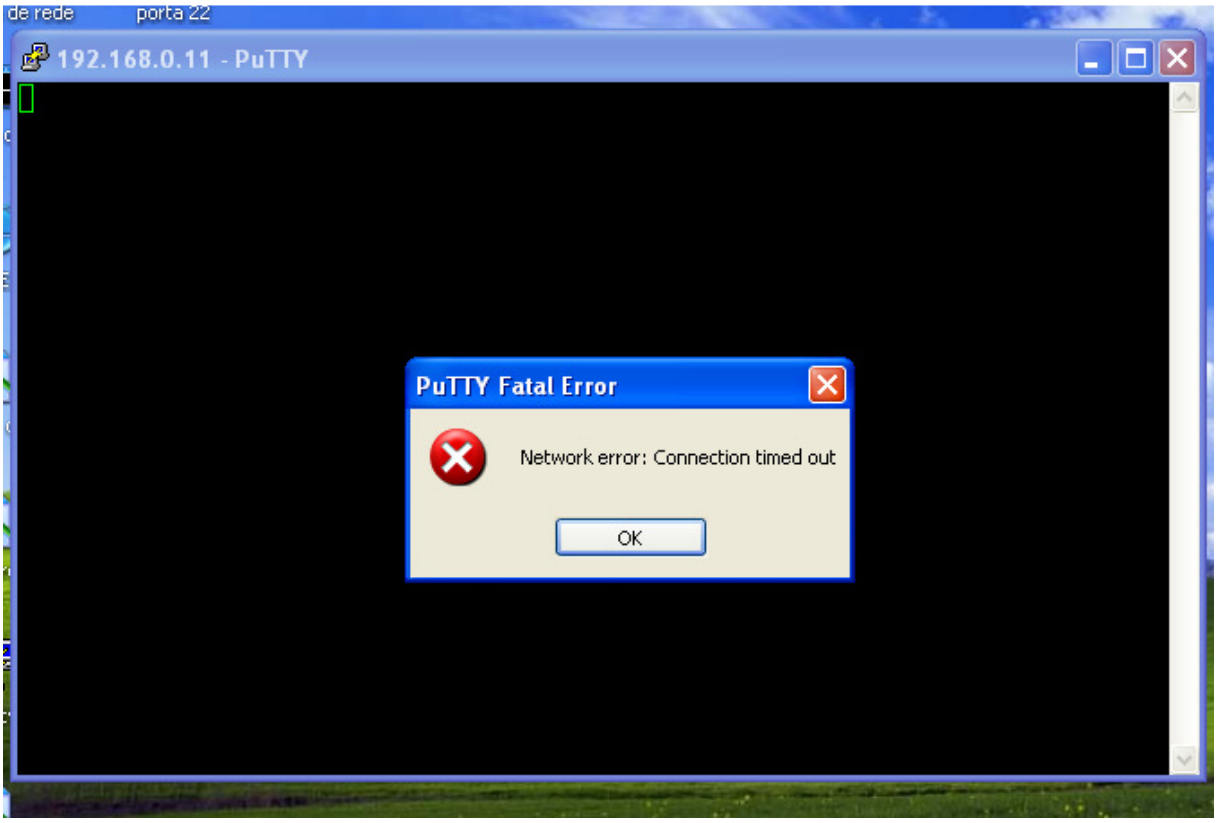

Figura 11 – Erro ao acessar via SSH o *firewall* [Autoria própria].

Se no teste estivéssemos usados o REJECT, teríamos o seguinte erro: "*Network error: Connection reset by peer*".

APENDICE D

Abaixo tem parte do log gerado pelo *firewall iptables*. Para ter acesso ao *log* o mesmo se encontra no caminho (/var/log) com o nome "iptables.log".

O tipo de evento que cobre o *iptables* é o *kern*. Podemos escolher qualquer prioridade, contanto que usemos a mesma na regra do *iptables*, estamos usando aqui a prioridade *warn*.

Comando utilizado para verificar logs:

tail -f /var/log/iptables.log

```
Sep 23 20:52:50 debianF kernel: [15685.088253] [FIREWALL: HTTP]IN=eth2 OUT=eth1 SRC=192.168.10.14 DST=74.125.234.255 LEN=52 TOS=0x00 PREC=0x00 TTL=63 ID=5170
 DF PROTO=TCP SPT=55854 DPT=80 WINDOW=1989 RES=0x00 ACK URGP=0
Sep 23 20:52:50 debianF kernel: [15685.177956] [FIREWALL: HTTP]IN=eth2 OUT=eth1 SRC=192.168.10.14 DST=173.194.42.15 LEN=52 TOS=0x00 PREC=0x00 TTL=63 ID=46232
 DF PROTO=TCP SPT=47705 DPT=80 WINDOW=634 RES=0x00 ACK URGP=0
Sep 23 20:52:50 debianF kernel: [15685.178877] [FIREWALL: HTTP]IN=eth2 OUT=eth1 SRC=192.168.10.14 DST=173.194.42.15 LEN=52 TOS=0x00 PREC=0x00 TTL=63 ID=46233
 DF PROTO=TCP SPT=47705 DPT=80 WINDOW=724 RES=0x00 ACK URGP=0
Sep 23 20:52:50 debianF kernel: [15685.181590] [FIREWALL: HTTP]IN=eth2 OUT=eth1 SRC=192.168.10.14 DST=173.194.42.15 LEN=52 TOS=0x00 PREC=0x00 TTL=63 ID=46234
 DF PROTO=TCP SPT=47705 DPT=80 WINDOW=815 RES=0x00 ACK URGP=0
Sep 23 20:52:50 debianF kernel: [15685.183038] [FIREWALL: HTTP]IN=eth2 OUT=eth1 SRC=192.168.10.14 DST=173.194.42.15 LEN=52 TOS=0x00 PREC=0x00 TTL=63 ID=46235
 DF PROTO=TCP SPT=47705 DPT=80 WINDOW=905 RES=0x00 ACK URGP=0
Sep 23 20:52:50 debianF kernel: [15685.183683] [FIREWALL: HTTP]IN=eth2 OUT=eth1 SRC=192.168.10.14 DST=173.194.42.15 LEN=52 TOS=0x00 PREC=0x00 TTL=63 ID=46236
 DF PROTO=TCP SPT=47705 DPT=80 WINDOW=996 RES=0x00 ACK URGP=0
Sep 23 20:52:50 debianF kernel: [15685.186992] [FIREWALL: HTTP]IN=eth2 OUT=eth1 SRC=192.168.10.14 DST=173.194.42.15 LEN=52 TOS=0x00 PREC=0x00 TTL=63 ID=46237
 DF PROTO=TCP SPT=47705 DPT=80 WINDOW=1086 RES=0x00 ACK URGP=0
Sep 23 20:52:51 debianF kernel: [15685.192281] [FIREWALL: HTTP]IN=eth2 OUT=eth1 SRC=192.168.10.14 DST=173.194.42.15 LEN=52 TOS=0x00 PREC=0x00 TTL=63 ID=46238
 DF PROTO=TCP SPT=47705 DPT=80 WINDOW=1177 RES=0x00 ACK URGP=0
Sep 23 20:52:51 debianF kernel: [15685.192610] [FIREWALL: HTTP]IN=eth2 OUT=eth1 SRC=192.168.10.14 DST=173.194.42.15 LEN=52 TOS=0x00 PREC=0x00 TTL=63 ID=46239
 DF PROTO=TCP SPT=47705 DPT=80 WINDOW=1267 RES=0x00 ACK URGP=0
Sep 23 20:52:51 debianF kernel: [15685.193829] [FIREWALL: HTTP]IN=eth2 OUT=eth1 SRC=192.168.10.14 DST=173.194.42.15 LEN=52 TOS=0x00 PREC=0x00 TTL=63 ID=46240
 DF PROTO=TCP SPT=47705 DPT=80 WINDOW=1358 RES=0x00 ACK URGP=0
Sep 23 20:52:51 debianF kernel: [15685.194701] [FIREWALL: HTTP]IN=eth2 OUT=eth1 SRC=192.168.10.14 DST=173.194.42.15 LEN=52 TOS=0x00 PREC=0x00 TTL=63 ID=46241
 DF PROTO=TCP SPT=47705 DPT=80 WINDOW=1446 RES=0x00 ACK URGP=0
```

Figura 12 – Parte do *log* gerado pelo *iptables* na porta 80 [Autoria própria].

Na figura abaixo segue a tela após usar o comando "DROP" no servidor *firewall:*

Figura 13 – Tela de erro ao acessar a porta 80 (http) [Autoria própria].

7 REFERÊNCIAS

FILIPPETTI, Marco A. **CCNA 4.1 – Guia Completo de Estudo.** 5ª ed. Florianópolis: Visual Books, 2008. 480p.

NETO, Urubatan. ***Linux* Firewall Iptables.** Rio de Janeiro: Editora Ciência Moderna, 2004. 98p.

THOMAS, Ygor. **IPTABLES e suas principais características** Fonte: Viva o Linux. Disponível em: http://www.vivaolinux.com.br/artigo/Dominando-o-iptables-%28parte-1%29. Acesso em 24 de jun. 2012.

JEDI, Pedro Arthur. **Netfilter / Iptables – Parte 1.** Fonte: Under-Linux. Disponível em: http://under-linux.org/blogs/pedroarthurjedi/netfilter-iptables-parte-1-9/. Acesso em 25 de jun. 2012.

DEZA, Alfredo. **Tempo de atividade e segurança de firewall com iptables.** Fonte: IBM. Disponível em: http://www.ibm.com/developerworks/br/library/os-iptables/. Acesso em 25 de jun. 2012.

WENDEL, Almir. **Iptables – Tabela mangles.** Fonte: ONLYTUTORIAIS. Disponível em: http://www.onlytutorials.com.br/2008/11/26/iptables-tabela-mangles/. Acesso em 18 de jul. 2012.

ODON, Bruno. **Iptables - treinamento Linux baseado em laboratórios.** Fonte: HowTo Online. Disponível em: http://brunoodon.com.br/howto-online/firewall-iptables/. Acesso em 14 de set. 2012.

FERREIRA, Johny. **Conhecendo o Iptables – Compartilhamento de Conexão, Mascaramento e Redirecionamento de Pacotes – Parte 2**. Fonte: TI da Hora! . Disponível em: http://johnnyroot.wordpress.com/2012/06/06/conhecendo-o-iptables-compartilhamento-de-conexao-mascaramento-e-redirecionamento-de-pacotes-parte-2/. Acesso em 18 de set. 2012.

SILVA, Gleydson Mazioli. **Foca Linux – Iptables.** Versão 6.40. Fonte: Guia Foca GNU/Linux. Site: http://www.guiafoca.org/. 73p.