Segurança de Redes Utilizando Recursos do Sistema Operacional Linux: Análise das Configurações do Servidor *Gateway* de um Instituto de Ensino Superior

Nilton Lopes de Souza Junior

Orientador: Edson Bruno Novais

JUIZ DE FORA

2014

Agradeço:

Primeiramente a Deus, por me dar forças e não desistir do meu objetivo.

Aos meu pais, principalmente à minha mãe, Vera Lúcia Ribeiro Neves, que em nenhum momento da minha vida me desamparou e que confiou na minha capacidade.

Aos meus irmãos, Nilver Lopes de Souza e Everton Lopes de Souza, que me proporcionaram momentos bons e agradáveis em minha vida, e também maus momentos, mas que serviram como aprendizado e amadurecimento.

Ao professor Edson Bruno Novais que, como orientador, me acolheu sem saber o desafio que seria.

Aos meus amigos do peito, que mesmo longe, lembram com carinho de nossa amizade.

Agradecimento especial à Larissa de Oliveira Mendes, minha futura esposa, com quem quero viver o resto de minha vida e construir a minha felicidade. Ela que confiou e me ajudou para que este trabalho fosse concluído.

Resumo

A segurança das informações tornou-se prioridade em ambientes corporativos, afim de evitar o extravio de dados, diante de uma rede mundial pública, que é considerada um canal de comunicação não confiável.

Devido ao conhecimento limitado e ao mau uso da *Internet* e seus serviços, as empresas têm necessidade de controlar o acesso à *Internet* de seus empregados, pois estes são ingenuamente receptores dos mais diversos tipos de malefícios.

VPN, *firewall* e *proxy* são tecnologias que podem aumentar a segurança da rede. Neste assunto que este trabalho será fundamentado.

Neste trabalho, uma análise foi feita em um servidor de redes de um Instituto de Ensino, com o intuito de verificar o nível de segurança aplicado. O servidor é o *gateway* de todo tráfego da rede e, por este motivo, programas de segurança devem ser instalados e configurados corretamente para preservar o ambiente de rede.

Palavras-chave: Segurança, Linux, *firewall*, Iptables, *proxy*, Squid, VPN

Abstract

Information security has become a priority in corporate environments, in order to avoid the mislaid of data, in front of a worldwide public network, which is considered an unreliable communication channel.

Due to limited knowledge and to the misuse of the Internet and its services, companies need to control the Internet access from their employees, as they are naively receivers of several types of harm.

VPN, firewall and proxy are technologies that can increase the network security. This subject that this work will be based.

In this study, an analysis was done in a network server of an Institute of Education, in order to verify the security level applied. The server is the gateway of the whole network traffic and for this reason, safety programs must be installed and configured properly to preserve the network environment.

Keywords: Security, Linux, firewall, Iptables, proxy, Squid, VPN.

Lista de Figuras

Lista de Abreviaturas e Siglas

Siglas

ACL = *Access Control List*
ADSL = *Asymmetric Digital Subscriber Line*
DDoS = *Distributed Denial of Service*
DHCP = *Dynamic Host Configuration Protocol*
DoS = *Denial of Service*
FTP = *File Transfer Protocol*
HTML = *Hyper Text Markup Language*
HTTP = *Hyper Text Transfer Protocol*
HTTPS = *Hyper Text Transfer Protocol Secure*
ICMP = *Internet Control Message Protocol*
ICP = *Internet Cache Protocol*
IP = *Internet Protocol*
ISP = *Internet Service Provider*
LAN = *Local Area Network*
MAC = *Media Access Control*
NAT = *Network Address Translation*
SARG = *Squid Analysis Report Generator*
TCP = *Transmission Control Protocol*
TOS = *Type of Services*
UDP = *User Datagram Protocol*
URL = *Uniform Resource Locator*
VPN = *Virtual Private Network*
VoIP = *Voice over Internet Protocol*
WAN = *Wide Area Network*

Sumário

1 INTRODUÇÃO

Atualmente, as tecnologias de redes de comunicação vêm sendo desenvolvidas para diversos serviços, como em comunicações móveis, seja para celulares e dispositivos portáteis, ou também para compartilhamento de dados em grandes redes de computadores. A demanda de tanta tecnologia obriga a descoberta e desenvolvimento de novas formas de manter a segurança das informações, seguindo os princípios básicos, que são:

- **Confidencialidade**: proteção das informações contra divulgação a terceiros não autorizados. Uma forma de garantir a confidencialidade das informações seria a imposição de permissões de arquivos e listas de controle de acesso, com o intuito de restringir a acessibilidade de arquivos confidenciais. Uma tecnologia muito difundida no panorama atual e que protege a confidencialidade das informações é a criptografia, garantindo que somente as pessoas certas, possuidoras de uma determinada chave de segurança, tenham o privilégio da leitura das informações.
- **Integridade**: proteção das informações contra modificação por pessoas não autorizadas. Assim como na confidencialidade, a criptografia, a permissão de arquivos e lista de controle de acesso são técnicas para garantir a integridade das informações.
- **Disponibilidade**: garantia de que os usuários autorizados sejam capazes de acessar as informações quando necessário. Ataques de DDoS são comumente utilizados com o objetivo de negar o acesso dos usuários aos serviços. Entre outros fatores que levam a falta de disponibilidade às informações estão quedas de energia, acidentes ou desastres naturais. Para garantir a disponibilidade dos dados, pode-se fazer a replicação de servidores, garantindo que se a informação não estiver em um servidor, providencialmente estará no servidor replicado.

O usuário da *Internet* está propenso a qualquer tipo de ataque, desde envio de vírus até crimes cibernéticos. Segundo o Centro de Estudos, Respostas e Tratamento de Incidentes de Segurança no Brasil (CERT.br) (2013), mais de 90 mil incidentes foram constatados no período entre julho e setembro de 2013. A tabela 1 mostra os

números estatísticos desses incidentes, que são brevemente explicador a seguir, segundo CERT.br (2013).

Tabela 1 - Totais Mensais e Trimestral Classificados por Tipo de Ataque.

Mês	Total	Worm (%)		DoS (%)		Invasão (%)		Web (%)		Scan (%)		Fraude (%)		Outros (%)	
Jul	**30874**	2932	9	123	0	509	1	1710	5	13769	44	7530	24	4301	13
Ago	**28531**	2090	7	24	0	542	1	1942	6	12336	43	8145	28	3452	12
Set	**31482**	2365	7	86	0	419	1	1573	5	15911	50	8538	27	2590	8
Total	**90887**	7387	8	233	0	1470	1	5225	5	42016	46	24213	26	10343	11

- ***Worm***: notificações de atividades maliciosas relacionadas com o processo automatizado de propagação de códigos maliciosos na rede.
- **DoS (*Denial of Service*)**: notificações de ataques de negação de serviço, onde o atacante utiliza um computador ou um conjunto de computadores para tirar de operação um serviço, computador ou rede.
- **Invasão**: um ataque bem sucedido que resulte no acesso não autorizado a um computador ou rede.
- ***Web***: um caso particular de ataque visando especificamente o comprometimento de servidores *web* ou desfigurações de páginas na *Internet*.
- ***Scan***: notificações de varreduras em redes de computadores, com o intuito de identificar quais computadores estão ativos e quais serviços estão sendo disponibilizados por eles. É amplamente utilizado por atacantes para identificar potenciais alvos, pois permite associar possíveis vulnerabilidades aos serviços habilitados em um computador.
- **Fraude**: é qualquer ato ardiloso, enganoso, de má-fé, com intuito de lesar ou ludibriar outrem, ou de não cumprir determinado dever; logro. Esta categoria engloba as notificações de tentativas de fraudes, ou seja, de incidentes em que ocorre uma tentativa de obter vantagem.
- **Outros**: notificações de incidentes que não se enquadram nas categorias anteriores.

1.1 Motivação

Nos últimos anos, a proteção das informações tem sido um desafio a ser tratado por empresas que têm em vista a segurança de sua rede privada. Com a contextualização global e o uso intenso da *Internet*, as redes empresarias estão cada vez mais vulneráveis. Por este motivo, há uma necessidade inevitável do desenvolvimento de ferramentas de segurança para ajudar os administradores de redes a implantarem políticas de segurança eficientes que identifiquem e reduzam o risco de ataques às redes empresariais.

Diante deste desafio, as organizações devem captar minuciosamente os pontos de falha da rede e elaborar planos e procedimentos de segurança da informação com ferramentas de baixo custo e fácil configuração que evitem ameaças e riscos oriundos da *Internet*.

1.2 Objetivos

Os objetivos deste trabalho serão divididos em Objetivo Geral e Objetivos Específicos.

1.2.1 Objetivo Geral

Este trabalho tem o objetivo geral de analisar o nível de segurança do servidor *gateway* de um Instituto de Ensino Superior, através de aplicativos instalados para este fim.

1.2.2 Objetivos Específicos

Os objetivos específicos deste trabalho são:

- Apontar as vantagens e desvantagens que os aplicativos trazem para a segurança da rede do Instituto.
- Indicar possíveis melhorias, abordando pontos de falhas de segurança da rede privada.

1.3 Metodologia

Para alcançar os objetivos deste estudo, será realizado um levantamento bibliográfico sobre os assuntos relacionados com base em teorias e práticas.

Após abordar o conteúdo teórico, será feita uma análise do ambiente de rede, fazendo uma caracterização do elementos que o constitui e um entendimento dos arquivos de configuração das regras dos aplicativos que proveem segurança à rede privada. Será estudada, na prática, qual a funcionalidade de cada regra ou conjunto delas, fazendo comparações com exemplos lidos na parte teórica.

1.4 Organização do Trabalho

Este trabalho de conclusão de curso está organizado em cinco capítulos:

- **Capítulo 1**: introdução do trabalho, mostrando as definições de segurança.
- **Capítulo 2**: embasamento teórico, abordando todo o conhecimento do trabalho para se chegar aos objetivos.
- **Capítulo 3**: estudo experimental, detalhando o entendimento das configurações das regras de um servidor que provê segurança à rede privada do Instituto.
- **Capítulo 4**: resultados do estudo experimental, descrevendo as vantagens e desvantagens do ambiente de rede, a partir da prática e da compreensão do funcionamento dos aplicativos de segurança do servidor, além de propor melhorias, principalmente soluções para os pontos de falha da segurança da rede.
- **Capitulo 5**: Concluir o trabalho, dizendo se os objetivos foram cumpridos.

2 EMBASAMENTO TEÓRICO

Neste capítulo, todo conhecimento necessário para o entendimento deste trabalho será fundamentado. Será feito um estudo das redes de computadores e sua utilização nas empresas e também será introduzido o conhecimento de ferramentas de segurança que foram instaladas no servidor *gateway* do Instituto de Ensino, que é por onde passa todo tráfego da rede privada. Essas tecnologias são a VPN, o *firewall* e o *proxy*.

2.1 Redes de Computadores

Em se tratando de tecnologia computacional, criar uma rede é a prática de ligar dois ou mais dispositivos de computação em conjunto com a finalidade de compartilhamento de dados. As redes são construídas com uma combinação de *hardware* e *software* de computador.

Com relação à sua classificação, as redes podem ser definidas de vários modos diferentes, sendo uma delas a que leva em consideração a área geográfica que se estende. As redes locais (LANs), por exemplo, normalmente abrangem uma única casa, uma escola ou um prédio pequeno, enquanto redes de longa distância (WANs), chegam ao alcance entre cidades, estados ou até mesmo o mundo todo e como exemplo, a *Internet* é a maior WAN pública do mundo.

Redes empresariais e de pequenos escritórios funcionam normalmente com uma ou duas redes locais, cada uma sendo controlada por seu próprio roteador e se combinam tão similar a uma rede doméstica. Um termo utilizado sobre redes é a intranet, que de acordo com Epaminondas (2001) é a composição de um ou mais servidores internos que permitem o compartilhamento de dispositivos como arquivos e impressoras, oferecendo uma revolução na comunicação interna da empresa. Ainda segundo o autor, a intranet pode funcionar isoladamente em uma rede ou via *Internet*, fazendo com que a base de dados da rede seja facilmente acessada de qualquer lugar pelo usuário que tiver autorização para entrar na rede.

A medida que a empresa cresce, sua rede tende a se expandir com números cada vez maiores de redes locais. Quando as corporações estão sediadas em mais de um local, configurando uma conectividade interna entre os seus edifícios de

escritórios e estes estão próximos, é chamada de rede do campus, ou de uma rede de área ampla (WAN), quando abrangem cidades ou até mesmo países.

As empresas estão cada vez mais habilitando suas redes locais ao acesso através de redes sem fio, embora ainda haja por parte de grandes empresas a tendência de ligarem seus edifícios de escritórios com cabeamento *ethernet* de alta velocidade para maior capacidade e desempenho da rede.

A maioria das empresas permitem que seus funcionários tenham acesso à *Internet* através da rede empresarial, sendo que em alguns casos são instaladas tecnologias de filtragem de conteúdo para bloquear o acesso a determinados *sites* ou domínios. Estes sistemas de filtragem usam um banco de dados configurável com nomes de domínio da *Internet*, endereços e palavras-chave que podem violar a política de uso da rede privada.

As instituições também têm aceitado que os funcionários tenham acesso à sua rede a partir de suas casas ou outros locais externos, com um recurso chamado de acesso remoto. Uma empresa pode configurar servidores de VPN para apoiar o acesso remoto, com os computadores dos funcionários configurados para usar *software* de cliente VPN correspondente e configurações de segurança.

Em comparação com redes domésticas, as redes empresariais enviam um volume muito maior de dados através da *Internet*, decorrentes de transferências em *sites* de empresa, *e-mail* e outros dados publicados externamente. Os planos residenciais de banda larga para acesso à *Internet* normalmente dão aos seus clientes uma taxa de dados significativamente maior para *download* em troca de uma taxa mais baixa em *upload*, mas os planos empresariais permitem taxas de *upload* mais elevadas para esta finalidade.

As organizações possuem dados privados valiosos que fazem com que a segurança da rede seja prioridade. Empresas conscientes deste fato, geralmente tomam medidas adicionais para proteger suas redes além do que é geralmente feito em redes domésticas.

Para evitar que dispositivos não autorizados entrem em sua rede, as empresas utilizam sistema de segurança centralizado, que exige a autenticação do usuário através de senha que é verificada em um diretório de rede e que também pode checar a configuração de *hardware* e *software* de um dispositivo para conferir se ele está autorizado a se juntar à rede. Outra medida de segurança seria a definição de senhas

de rede dos usuários que expirem periodicamente, forçando-os a modificarem, aumentando, assim, a segurança da rede.

2.2 *Virtual Private Network*

A VPN, sigla para *Virtual Private Network* (Rede Privada Virtual), tem atraído a atenção de muitas organizações que procuram expandir sua capacidade de rede e reduzir os custos (CUNHA *et al*, 2007), podendo ser encontrada em locais de trabalho e residências, onde ela permite que os usuários entrem em segurança nas redes privadas.

VPN é uma forma de WAN, pois fornece conectividade de rede através de uma distância física longa e sua característica fundamental é a capacidade de trabalhar tanto em redes privadas, bem como em redes públicas como a *Internet*.

Nos últimos anos, muitas organizações têm aumentado a mobilidade de seus funcionários que viajam com frequência ou trabalham em suas residências, encontrando na VPN, uma forma mais conveniente e segura para que eles fiquem conectados à intranet corporativa. Ela pode ainda ser configurada para dar suporte e acesso remoto protegido aos escritórios domésticos corporativos através da *Internet*.

Existem três elementos principais de uma conexão da VPN:

- **Transporte subjacente**: utilizando-se da infraestrutura da *Internet*, que não possui uma segurança adequada, e o protocolo TCP/IP, a VPN adiciona cabeçalhos para que seja feita a transmissão dos pacotes, fazendo com que os dispositivos VPN se comuniquem. A figura 1 exemplifica o novo pacote que será transmitido, após a adição dos novos cabeçalhos.

Figura 1 - Adição de novos cabeçalhos no transporte subjacente

- **Tunelamento**: criação de um túnel virtual para envio de dados de uma extremidade a outra, que primeiramente terão seus pacotes encriptados e serão encapsulados em outro pacote. A figura 2 representa o túnel virtual criado através da *Internet*, onde será feita a transmissão dos pacotes VPN.

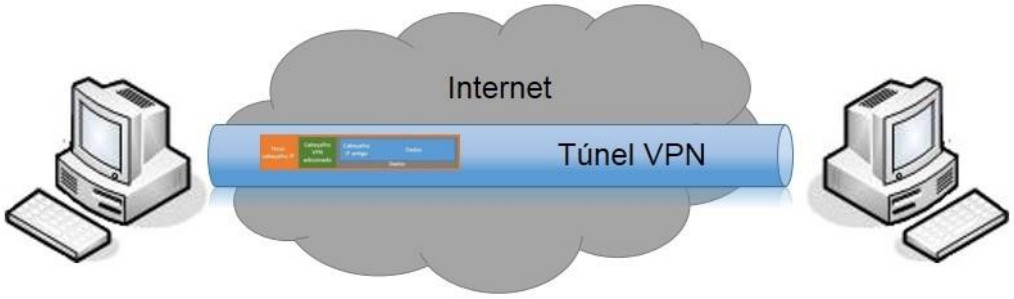


Figura 2 - Pacote encapsulado passando pelo túnel VPN

- **Autenticação das extremidades**: a autenticação garantirá que apenas usuários autorizados transmitirão dados, sendo que geralmente é utilizado um algoritmo *hash* para validar a comunicação. Se parte da mensagem for alterada durante a transmissão, o pacote será descartado e será pedido o reenvio do mesmo.

Existem três topologias em que a VPN é utilizada:

- ***Host - Host***: comunicação direta entre dois *hosts*[1] separados fisicamente.
- ***Host - Gateway***: nesta topologia, em que o *host* se comunica com um servidor *gateway* VPN, tem uma finalidade maior para funcionários de empresas que possuem um escritório doméstico ou que estão sempre em viagem, e precisam se comunicar com a rede empresarial, podendo, até mesmo, ter acesso remoto aos computadores.
- ***Gateway - Gateway***: essa topologia faz a comunicação entre duas redes separadas fisicamente. Uma de suas finalidades é de comunicar as redes de uma empresa matriz com a sua filial, podendo assim, a matriz controlar os recursos e dados.

[1] *Host* é qualquer computador ou dispositivo conectado a uma rede, que conta com endereço IP e nome definidos e é responsável por oferecer recursos, informações e serviços aos usuários ou clientes (VIANNA, 2012).

Uma solução VPN utilizando a *Internet* em uma arquitetura cliente-servidor seria da seguinte maneira:

1. Um dispositivo remoto que queira fazer *login* na rede empresarial, primeiro se conecta a qualquer provedor de serviço de acesso à *Internet* (ISP).
2. Em seguida, o dispositivo inicia uma conexão com o servidor VPN da empresa que é feita através de um cliente VPN instalado no dispositivo remoto.
3. Assim que a conexão é estabelecida, o cliente remoto pode se comunicar com os sistemas internos da empresa através da *Internet*, como se estive em uma rede local.

Além de dar suporte para acesso remoto, uma VPN também pode unir duas redes. Deste modo, a rede remota pode se juntar a uma rede de uma empresa diferente para formar uma intranet estendida, ao invés de apenas um único cliente remoto.

Para uma empresa que visa fornecer uma infraestrutura de rede segura para sua base de clientes, uma VPN oferece duas principais vantagens sobre as tecnologias alternativas: redução de custos e escalabilidade da rede. Já para os clientes que acessam essas redes, a VPN também pode trazer alguns benefícios de facilidade de uso.

Uma VPN pode reduzir custos para uma empresa das seguintes formas:

- Eliminando a necessidade de linhas alugadas de longa distância, que geralmente tem um custo elevado (PALLARES, 2003). Com uma VPN, pode-se utilizar de uma infraestrutura de rede pública, incluindo a *Internet*, para fazer as conexões.
- Reduzindo tarifas telefônicas de longa distância, implementando serviços de teleconferência, tais como Voz Sobre IP (VoIP), em uma via de comunicação segura.
- Dispensando os custos de suporte e manutenção dos servidores, já que é uma tecnologia de fácil utilização.

O custo na construção de uma rede privada de uma empresa com linhas dedicadas pode ser razoável no começo, mas aumenta exponencialmente à medida que ela cresce. Uma empresa com duas filiais, por exemplo, pode utilizar apenas uma linha dedicada para conectá-las, mas com quatro filiais são exigidas seis linhas dedicadas para conectá-las diretamente uma com as outras. Este problema de

escalabilidade é solucionado com qualidade de serviço e segurança utilizando-se a VPN através da *Internet*, principalmente se tratando de empresas em âmbitos internacionais.

Infelizmente, como em qualquer tecnologia, a VPN possui limitações, como por exemplo, a dependência da *Internet* para a realização de suas conexões, devendo estar sempre disponível, o que é praticamente impossível, devido a possibilidade de falhas técnicas dos provedores de *Internet*. As empresas que se utilizarem da VPN, devem considerar essas limitações na implantação e na utilização dessa tecnologia em suas operações. Exige-se, portanto, uma compreensão detalhada dos problemas de segurança da rede e uma cuidadosa instalação e configuração, afim de garantir proteção suficiente em sua rede pública como a *Internet*.

A implantação da VPN é concretizada através de protocolos e configurações dos aplicativos VPN no servidor e no cliente este procedimento não será abordado neste trabalho. Para mais informações, sugere-se consultar em Fagundes (2007).

2.3 Firewall

Firewall é uma tecnologia que restringe o acesso entre uma ou mais redes de computadores internas (LAN) e a rede externa (WAN, *Internet*).

De acordo com Emerson Alecrim (2013)

> *Firewall* é uma solução de segurança baseada em *hardware* ou *software* (mais comum) que, a partir de um conjunto de regras ou instruções, analisa o tráfego de rede para determinar quais operações de transmissão ou recepção de dados podem ser executadas. "Parede de fogo", a tradução literal do nome, já deixa claro que o *firewall* se enquadra em uma espécie de barreira de defesa. A sua missão, por assim dizer, consiste basicamente em bloquear tráfego de dados indesejado e liberar acessos bem-vindos.

Portanto, o *firewall* recolhe a informação, verifica se possui permissão e, se caso negativo, descarta a informação. Para que isto seja possível, deve-se criar uma política de regras, por exemplo, tanto podendo ser totalmente restritivo e liberar apenas algum tráfego de informação, quanto o contrário, ou seja, sendo aberto e fazendo restrição de algum tipo de informação. Isto dependerá da necessidade do

grau de segurança e das condições da rede de computadores, levando em consideração suas vantagens em relação às desvantagens.

A figura 3 representa de forma ilustrativa, uma rede interna sendo protegida pelo *firewall*. Assim, todo tráfego de pacote será passado pelo *firewall* antes de ser redirecionado ao seu destino, seja à Internet ou à rede interna.

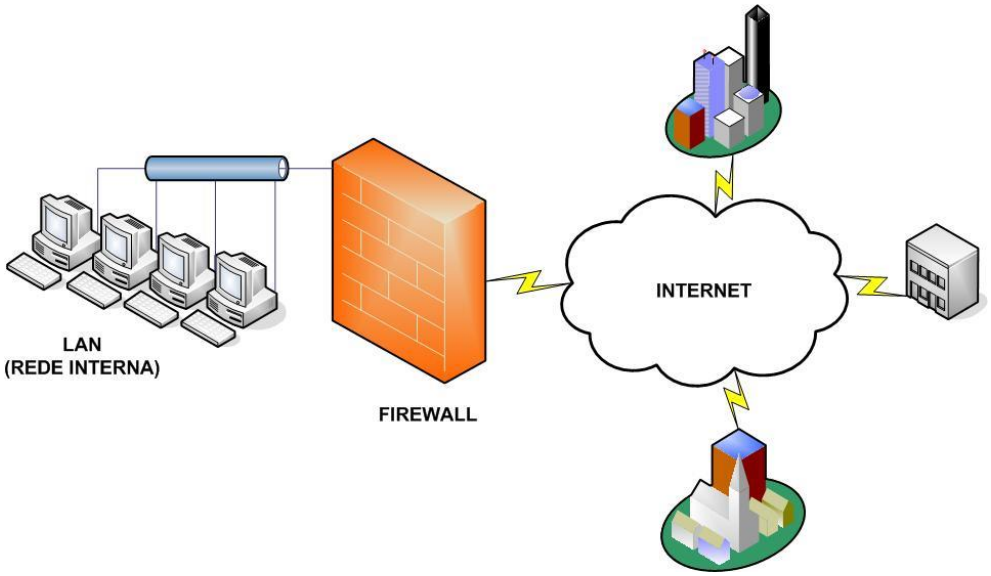


Figura 3 - Representação de um *firewall*

Os tipos mais conhecidos de *firewall* são:

- **Filtro de pacotes (*packet filtering*)**: apesar de oferecer um nível de segurança expressivo, esta é uma técnica mais simples. O *firewall* analisa as informações do cabeçalho de cada pacote, que contém dados como endereços IP de origem e destino, tamanho, tipo do serviço, entre outros, e confrontam com as regras estabelecidas para liberação (ACCEPT) ou bloqueio (DROP) do pacote, podendo até mesmo gerar algum tipo de relatório (*log*) no sistema, com o registro de acesso. A transmissão dos dados é baseada no popular protocolo TCP/IP e sua filtragem ocorre na camada de Rede (endereçamento de dispositivos da rede e dos processos de roteamento) e na camada de Transporte (onde se encontram os protocolos de tráfego de pacotes TCP e UDP). O Iptables pode ser considerado um *firewall* de pacotes e está presente nativamente nos sistemas operacionais Linux e será estudado neste trabalho.
- ***Firewall* de aplicação ou *proxy* de serviços (*proxy services*)**: mais conhecido apenas como *proxy*, ele é um recurso de segurança que serve

como intermediário entre uma rede interna e uma rede externa (*Internet*). Assim como o nome já diz, ele trabalha na camada de Aplicação, analisando os pacotes de serviços como HTTP, FTP, entre outros, podendo bloqueá-los por conter comandos que podem colocar em risco a segurança da rede. O Squid é um *software* livre que se enquadra nesta categoria de *firewall* e também será estudado neste trabalho.

2.3.1 Netfilter

De acordo com Urubatan Neto (2004), Netfilter é uma ferramenta que controla e monitora todo o tipo de fluxo de dados que trafega na estrutura interna do *kernel*[2] do sistema operacional Linux, agilizando tomadas de decisões e processamentos. Sendo assim, o Netfilter processa a informação e com o auxílio de três tabelas, decide se libera ou não o tráfego. As tabelas que auxiliam o Netfilter são: tabela *Filter*, Tabela NAT[3] e tabela *Mangle*. O *kernel* lida com situações, chamadas de cadeias (*chains*), que nas tabelas, armazenam as regras do *firewall* definidas pelo administrador para sua operação.

A tabela *Filter* é a tabela padrão do Netfilter e nela serão guardadas as regras relacionadas a filtragem de pacotes. Ela possui três cadeias:

- **INPUT**: tráfego de pacote que entra no servidor *firewall*.
- **OUTPUT**: tráfego de pacote que sai do servidor *firewall*.
- **FORWARD**: tráfego de pacote que chega no servidor *firewall*, mas deve ser encaminhado a um outro *host*.

Na figura 4, é ilustrado como o pacote é tratado com relação à tabela *Filter*, para que tenha o correto processamento dentro do *kernel*. Então, quando o pacote entra no *kernel*, é analisado para que possa ser feito seu roteamento adequadamente. Caso seu destino seja o servidor que contém o *firewall*, o Netfilter o analisa com as regras da cadeia INPUT para ser enviado ao processamento local. O pacote originado no servidor *firewall* é confrontado com as regras da cadeia OUTPUT para ter sua saída

[2] *Kernel* (núcleo) é o componente central de um sistema operacional que serve como ponte entre os aplicativos e o *hardware*, onde é feito o processamento de dados.

[3] NAT é uma método que troca os endereços IP de origem dos pacotes pelo endereço IP de um *gateway* (roteador ou servidor) da rede, de maneira que o computador da rede interna tenha acesso à Internet.

liberada. Caso o pacote tenha outro *host* como destino, ele é submetido às regras da cadeia FORWARD e encaminhado para o *host* destino.

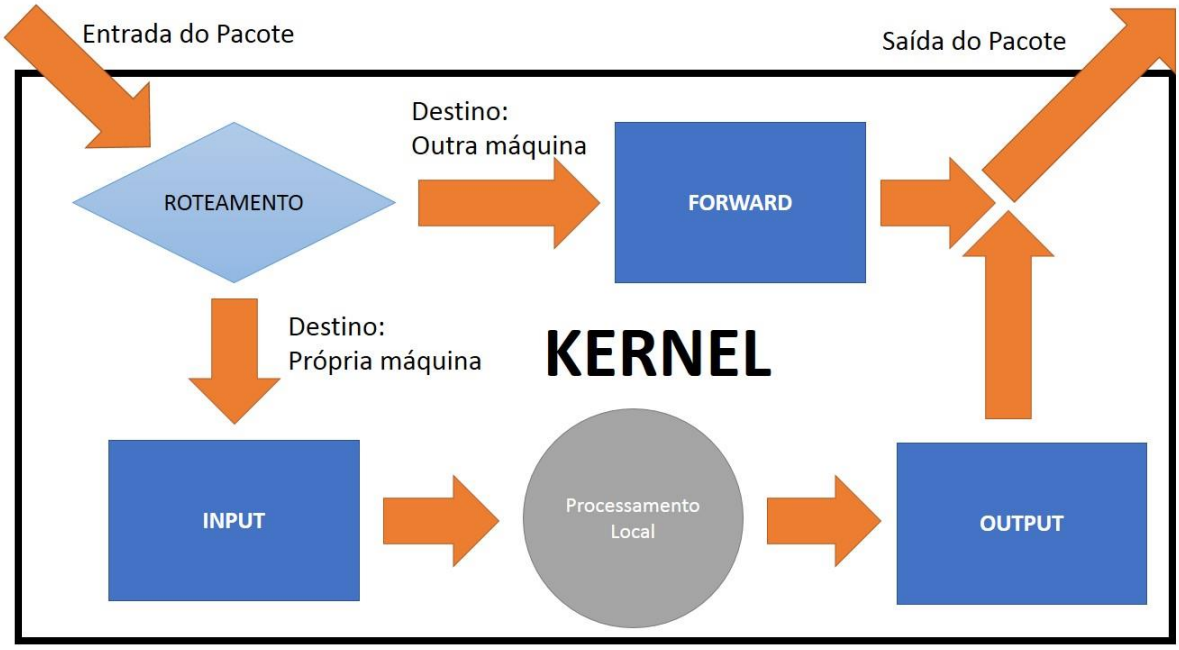


Figura 4 - Esquema do tráfego do pacote da tabela *Filter*

A tabela NAT serve para controlar a tradução dos endereços IP e portas TCP da rede local para a *Internet*, que pode ser útil para fazer com que os dispositivos de uma rede interna tenham acesso à *Internet*, conhecido como Mascaramento (*Masquerading*). Também são três as cadeias desta tabela, que para Neto (2004) são:

- **PREROUTING**: Utilizada quando há necessidade de se fazer alterações em pacotes antes que os mesmos sejam roteados.
- **OUTPUT**: Trata os pacotes emitidos pelo servidor *firewall*.
- **POSTROUTING**: Utilizado quando há necessidade de se fazer alterações em pacotes após o tratamento do roteamento.

A figura 5 ilustra a trajetória do pacote na tabela NAT, com relação à tradução de endereços. Antes de entrar no *kernel*, o pacote é tratado pela cadeia PREROUTING. Já a sua saída deve ser tratada pela cadeia POSTROUTING. A cadeia OUTPUT é utilizada quando a origem do pacote é a servidor que contém o *firewall*.

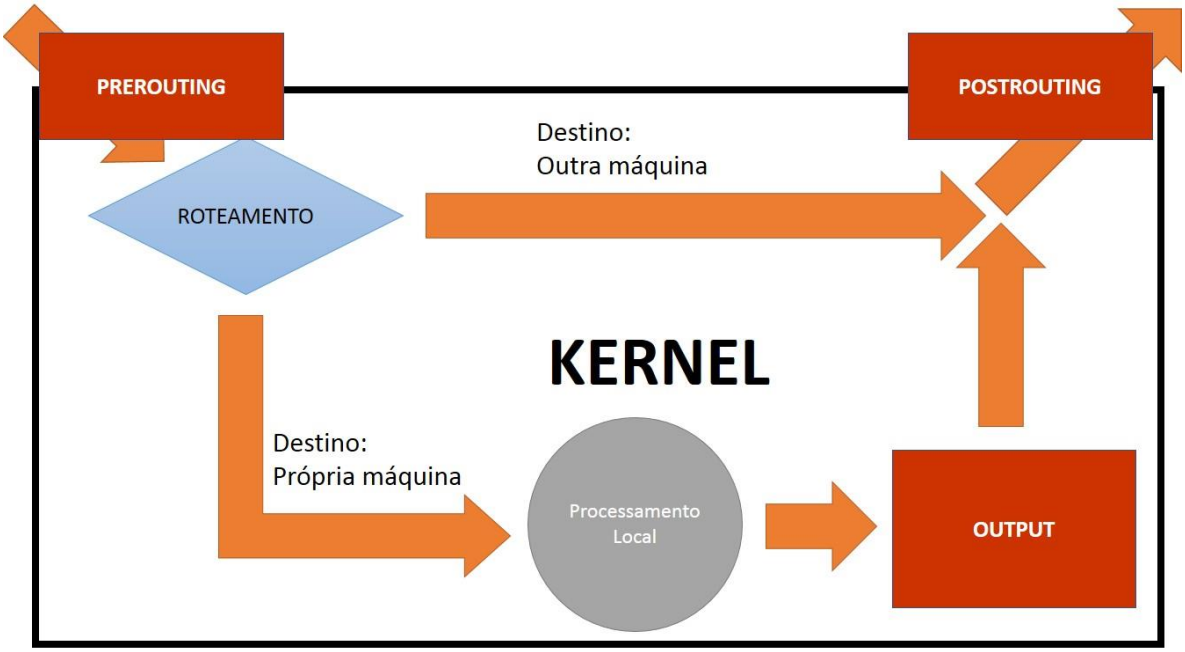


Figura 5 - Esquema do tráfego do pacote da Tabela NAT

Já a tabela *Mangle* pode fazer alteração de prioridade dos pacotes (entrada e saída), de acordo com o tipo de serviço (TOS - Type of Service). Com isto, o tráfego inútil da rede pode ser tratado com prioridade mais baixa do que um tráfego que precisa ser processado rapidamente, tais como os *streaming* de áudio e vídeo em uma videoconferência. Esta tabela possui cinco cadeias (PREROUTING, POSTROUTING, INPUT, OUTPUT e FORWARD), mas são duas as cadeias mais utilizadas, que segundo Neto (2004) suas definições são:

- **PREROUTING**: modifica pacotes dando-lhes um tratamento especial antes que os mesmos sejam roteados.
- **OUTPUT**: altera pacotes de forma “especial” gerados localmente antes que os mesmos sejam roteados.

A figura 6 ilustra todo o caminho que o pacote faz, desde a entrada até a saída do *firewall*, observando que se o pacote for bloqueado em alguma das tabelas, o percurso é interrompido.

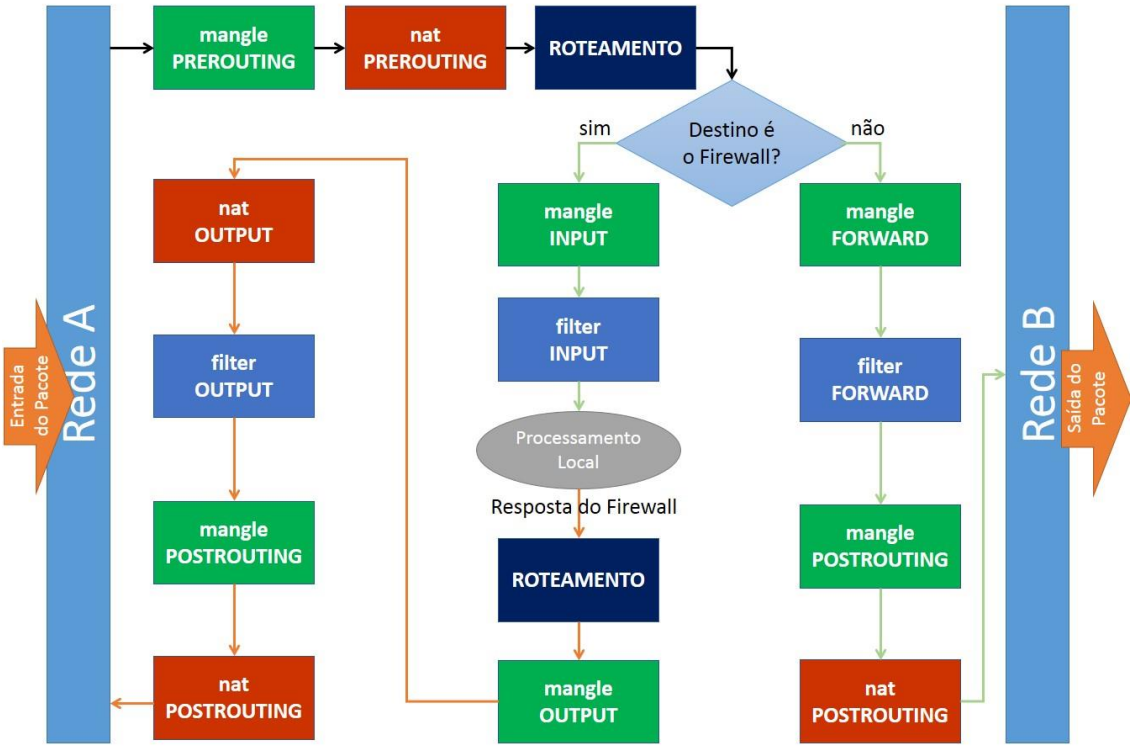


Figura 6 - Esquema da trajetória do pacote em todas as tabelas do Netfilter

Uma ferramenta para a manipulação das tabelas do Netfilter é o Iptables, que foi incorporado à versão 2.4 do *kernel* do Linux em 1999, sucedendo outras versões, sendo este mais completo e estável do que seus antecessores, tendo mais opções de controle de tráfego e apresenta grande segurança tanto para redes mais simples quanto para as mais complexas.

O Iptables tem várias características, que segundo Silva (2007), são:

- Especificação de portas/endereço de origem/destino.
- Suporte a protocolos TCP/UDP/ICMP (incluindo tipos de mensagens icmp).
- Suporte a interfaces de origem/destino de pacotes.
- Manipulação de serviços de *proxy* na rede.
- Tratamento de tráfego dividido em cadeias (para melhor controle do tráfego que entra/sai do dispositivo e tráfego redirecionado).
- Suporte a um número ilimitado de regras por cadeia.
- Muito rápido, estável e seguro.
- Mecanismos internos para rejeitar automaticamente pacotes duvidosos ou mal formados.

- Suporte a módulos externos para expansão das funcionalidades padrões oferecidas pelo código de *firewall.*
- Suporte completo a roteamento de pacotes, tratadas em uma área diferente de tráfegos padrões.
- Suporte à especificação de tipo de serviço para priorizar o tráfego de determinados tipos de pacotes.
- Suporte à especificação de exceções para as regras ou parte das regras.
- Suporte a detecção de fragmentos.
- Suporte a envio de alertas personalizados ao *syslog* sobre o tráfego aceito/bloqueado.
- Redirecionamento de portas.
- Mascaramento.
- Suporte a SNAT (modificação do endereço de origem dos *hosts* para um único IP ou faixa de IPs).
- Suporte a DNAT (modificação do endereço de destino dos *hosts* para um único IP ou fixa de IPs).
- Contagem de pacotes que atravessaram uma interface/regra.
- Limitação de passagem de pacotes/conferência de regra (muito útil para criar proteções contra *syn flood*, *ping flood*, DoS, etc).

O Iptables trabalha com tabelas de regras, onde as tabelas são as mesmas do Netfilter e as regras, que são comandos para realizar alguma ação, são analisadas sequencialmente até o final da lista. Caso a regra tenha algum erro de sintaxe, ela será descartada e uma mensagem será mostrada. As regras definidas pelo usuário são armazenadas em cadeias (INPUT, FORWARD, OUTPUT, PREROUTING e POSTROUTING) que se encontram nas tabelas.

O Iptables faz a leitura das regras na ordem em que elas foram inseridas nas cadeias das tabelas e ao satisfazer a condição de uma regra, é feito um tratamento no pacote de acordo com o alvo mencionado. Por este motivo, deve-se ter cuidado ao inserir regras que possuem maior prioridade sobre as outras. Se nenhuma regra for satisfeita, o pacote terá o tratamento da política padrão (aceitar ou bloquear).

A figura 7, a seguir, ilustra as regras de uma cadeia e o modo de leitura.

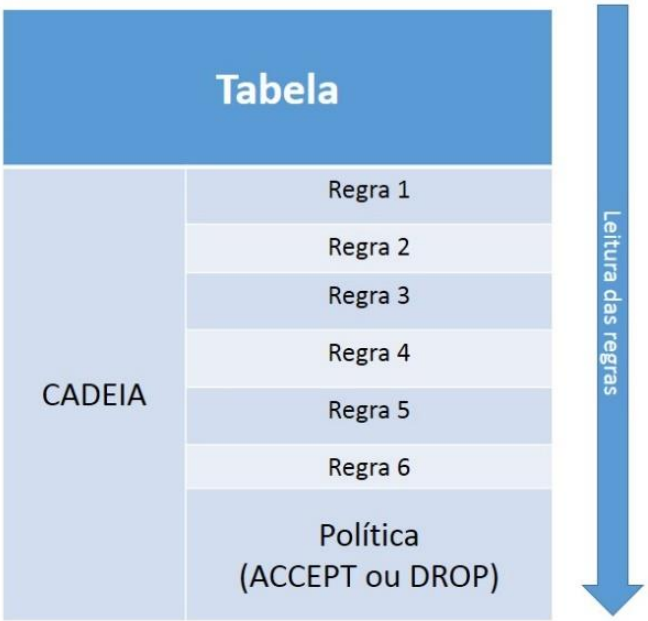


Figura 7 - Ordem de leitura e processamento das regras de uma cadeia

Existe uma sintaxe para a escrita das regras do Iptables, que é:

```
# iptables [tabela] [comando] [ação] [alvo]
```

Todo nome de tabela, comando, ação ou alvo são *case-sensitive*, então deve-se tomar cuidado para não confundir letras maiúsculas e minúsculas, pois em caso de erro, a regra será descartada.

Para associar uma regra a uma tabela, deve-se especificar qual será a tabela utilizada, colocando o comando **-t** [tabela]. Caso não seja especificado a tabela, o Iptables adota a *Filter* por padrão. A seguir, alguns exemplos:

```
# iptables -t filter
# iptables -t nat
# iptables -t mangle
```

Os comandos fazem a manipulação da tabela através das regras e das cadeias correspondentes. Segue uma listagem contendo os comandos e suas descrições:

- **-A**: acrescenta uma nova regra ao final da lista de regras já existentes na tabela.

  Exemplo: acrescentar uma nova regra ao final da cadeia OUTPUT da tabela *Filter*.

  ```
  # iptables -t filter -A OUTPUT -d 192.168.0.1 -j DROP
  ```

- **-D**: deleta uma regra especificada.

  Exemplo: deletar a regra do exemplo anterior.

  ```
  # iptables -t filter -D OUTPUT -d 192.168.0.1 -j DROP
  ```

  Exemplo 2: deletar a regra 3 da cadeia INPUT da tabela *Filter*.

  ```
  # iptables -t filter -D INPUT 3
  ```

- **-L**: lista as regras de tabelas ou cadeias específicas.

  Exemplo: mostrar todas as regras da tabela *Filter*.

  ```
  # iptables -t filter –L
  ```

  Exemplo 2: mostrar todas as regras apenas da cadeia INPUT da tabela *Filter*.

  ```
  # iptables -t filter –L INPUT
  ```

- **-P**: altera o alvo (política) padrão das cadeias. Pode ter apenas os alvos ACCEPT (aceitar) e DROP (bloquear).

  Exemplo: alterar a política padrão da cadeia FORWARD da tabela *Filter* para bloquear os pacote que não se enquadrarem em nenhuma regra.

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

- **-F**: remove todas as regras configuradas

  Exemplo: remover todas as regras da tabela NAT.

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

  Exemplo 2: remover todas as regras apenas da cadeia PREROUTING da tabela NAT.

  ```
  # iptables -t nat -F PREROUTING
  ```

- **-I**: insere uma nova regra na lista. Se não for explicitado a posição, sua inserção será no início (topo) da lista.

  Exemplo: inserir uma regra na cadeia INPUT da tabela *Filter*.

  ```
  # iptables -t filter -I INPUT -d 192.168.0.1 -j DROP
  ```

  Exemplo: inserir uma regra na posição 3 na cadeia OUTPUT da tabela *Filter*.

  ```
  # iptables -t filter -I OUTPUT 3 -d 192.168.0.1 -j ACCEPT
  ```

- **-N**: cria uma nova cadeia em uma tabela especificada.

  Exemplo: criar uma cadeia com nome C_HTTP na tabela *Filter*.

  ```
  # iptables -t filter -N C_HTTP
  ```

- **-X**: apaga uma cadeia que foi criada

  Exemplo: apagar a cadeia criada no exemplo anterior.

  ```
  # iptables -t filter -X C_HTTP
  ```

Algumas configurações necessitam de ações para ter utilidade. Segue, então, uma listagem das ações e suas descrições:

- **-s**: especifica a origem (*source*) do pacote, podendo ser um endereço IP, uma rede ou o nome de um *host*.

  Exemplo: aceitar na cadeia INPUT da tabela *Filter* pacotes com origem IP 192.168.0.10.

  ```
  # iptables -t filter -A INPUT -s 192.168.0.10 -j ACCEPT
  ```

  Exemplo 2: aceitar a entrada de pacotes com origem da rede 192.168.0.0/24.

  ```
  # iptables -t filter -A INPUT -s 192.168.0.0/24 -j ACCEPT
  # iptables -t filter -A INPUT -s 192.168.0.0/255.255.255.0 -j ACCEPT
  ```

- **-d**: especifica o destino (*destination*) do pacote. É válida as mesmas especificações da ação -s.

  Exemplo: bloquear a saída de pacotes com destino IP 192.168.0.10.

  ```
  # iptables -t filter -A OUTPUT -d 192.168.0.10 -j DROP
  ```

- **-p**: especifica o protocolo do pacote.

  Exemplo: bloquear respostas ao *ping* com o endereço IP origem 192.168.0.10 ao endereço IP destino 192.168.0.11.

  ```
  # iptables -t filter -A FORWARD -s 192.168.0.10 -d 192.168.0.11 -p icmp -j DROP
  ```

- **-i**: especifica a interface de entrada a ser utilizada. Não pode ser utilizado na cadeia OUTPUT da tabela *Filter*.

  Exemplo: aceitar os pacotes que entrem pela interface de rede eth0.

  ```
  # iptables -t filter -A INPUT -i eth0 -j ACCEPT
  ```

- **-o**: especifica a interface de saída a ser utilizada. Não pode ser utilizado na cadeia INPUT da tabela *Filter*.

  Exemplo: bloquear os pacotes que saem por todas interface de rede *ethernet*.

  ```
  # iptables -t filter -A OUTPUT -o eth+ -j DROP
  ```

- **--sport**: especifica a porta de origem (*source port*) do pacote. Funciona apenas para os protocolos TCP e UDP.

  Exemplo: bloquear qualquer pacote com destino o endereço IP 192.168.0.10 que utilize o protocolo TCP e originar na fixa de portas de 300 a 350.

  ```
  # iptables -t filter -A OUTPUT -d 192.168.0.10 -p tcp --sport 300:350 -j DROP
  ```

- **--dport**: especifica a porta de destino (*destination port*) do pacote. Similar ao --sport.

  Exemplo: bloquear qualquer pacote com destino o endereço IP 192.168.0.10 que utilize o protocolo TCP com destino a quaisquer portas da faixa de 300 a 350.

  ```
  # iptables -t filter -A OUTPUT -d 192.168.0.10 -p tcp --dport 300:350 -j DROP
  ```

- **-!**: aplica uma exceção a uma regra.

  Exemplo: aceitar pacotes de qualquer *host*, exceto do IP origem 192.168.0.10.

  ```
  # iptables -t filter -A INPUT -s ! 192.168.0.10 -j ACCEPT
  ```

- **-j**: define o alvo do pacote.

  Exemplo: aceitar o encaminhamento de pacotes de qualquer e para qualquer *host*.

  ```
  # iptables -t filter -A FORWARD -s 0.0.0.0/0 -d 0.0.0.0/0 -j ACCEPT
  ```

  Exemplo 2: bloquear o encaminhamento de pacotes de qualquer para qualquer *host*.

  ```
  # iptables -t filter -A FORWARD -s 0.0.0.0/0 -d 0.0.0.0/0 -j DROP
  ```

Alvo é o destino do pacote quando for satisfeita a condição de uma regra. Seu destino também pode ser uma cadeia criada com o intuito de fazer o tratamento de pacotes específicos à regra. A seguir, os principais alvos:

- **ACCPET**: aceita o pacote.
- **DROP**: bloqueia o pacote.
- **REJECT**: rejeita o pacote, mas retorna uma mensagem de erro ao dispositivo emissor.
- **LOG**: cria informações de registro no arquivo */var/log/messages*.

- **REDIRECT**: faz o redirecionamento de portas, tendo que acrescentar a opção --to-port.
- **TOS**: dá prioridade ao pacote de acordo com o tipo de serviço.
- **SNAT**: altera o endereço de origem antes de rotear o pacote.
- **MASQUERADE**: altera o endereço de origem sem precisar do endereço de destino.
- **DNAT**: altera o endereço de destino antes de rotear o pacote.

2.4 Proxy

Proxy é um servidor que funciona como um intermediário entre os dois extremos de uma conexão de rede cliente-servidor, interagindo com as aplicações de rede, mais comumente os servidores e navegadores *web*. Dentro de redes corporativas, o *proxy* é instalado em dispositivos especialmente designados para prover segurança à rede interna (intranet). Alguns provedores de serviços de *Internet* também utilizam servidores *proxy*, como parte da prestação de serviços *on-line* para seus clientes. A figura 8 representa o acesso à *Internet* por um dispositivo de um usuário passando pelo servidor *proxy*.

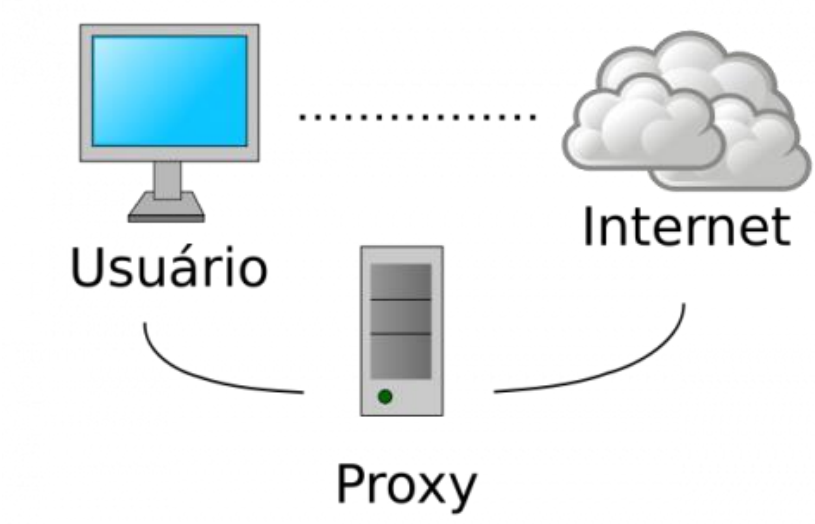


Figura 8 - Esquema de usuário acessando a *Internet* através do *proxy*

Pelo fato de trabalhar na camada de aplicação, o *proxy* tem a capacidade de filtrar conteúdos que o *firewall* de pacotes não consegue. Por exemplo, os servidores *proxy* podem verificar com mais precisão a URL de um pedido de saída para páginas *web*, inspecionando mensagens HTTP. Os administradores de rede podem usar este

recurso para bloquear o acesso à domínios ilegais, mas permitir o acesso a outros *sites*.

Apesar do *firewall* do tipo filtro de pacote poder, além de filtrar por número de porta ou endereço IP, filtrar por nomes de domínio, o servidor *proxy*, além disto, podem também filtrar com base no conteúdo do aplicativo dentro das mensagens de requisição HTTP.

Uma outra finalidade do *proxy* é fazer um armazenamento local (*cache*) de páginas da *Internet* e de arquivos já acessados, com o intuito de agilizar o acesso a estes sem a necessidade de ter que novamente requisitá-los externamente. Para isto, o *proxy* atualiza periodicamente as informações obtidas.

O *cache* de páginas *web* por servidores *proxy* pode melhorar a experiência do usuário das seguintes maneiras:

- Economia de largura de banda, pois haverá menos requisições feitas externamente, aumentando a escalabilidade.
- Melhoria do tempo de resposta das requisições, assim, as páginas *web* podem carregar mais rapidamente no navegador.
- Aumento da disponibilidade de conteúdo, com cópias de páginas e outros tipos de conteúdo estático permanecendo acessíveis no *cache*, mesmo se a fonte original ou um link de rede intermediário ficar *off-line*.

Para garantir que os usuários da rede utilizem o *proxy* sem precisar de configurá-lo manualmente em cada estação, é necessário se utilizar do recurso do *proxy* transparente, que fará com que o *proxy* intercepte os acessos na porta 80 (http), obrigando que toda informação seja passada por regras de controle de acesso, seja feito um registro, autenticação e *cache*. Para isto, o *proxy* trabalhará em conjunto com um *firewall* de pacotes e este fará o redirecionamento das requisições recebidas na porta 80 para a porta de destino do *proxy*.

2.4.1 Squid

Squid é um servidor *proxy* com código-fonte aberto, capaz de aceitar requisições dos protocolos HTTP, HTTPS, FTP e *Gopher*[4]. Ele é muito utilizado por sua flexibilidade e eficiência, além de robustez, segurança e a variedade de recursos disponíveis.

De acordo com Silva (2005), o Squid implementa várias características úteis em ambientes corporativos como:

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

Além destas características citadas, o Squid ainda pode:

- Atuar como *proxy* transparente.
- Liberar ou bloquear o acesso definido por horário.
- Fazer *cache* de DNS.

A configuração do Squid é feita através da edição do arquivo *squid.conf* e sua localização pode variar de acordo com a distribuição Linux e/ou versão do Squid utilizados, mas geralmente está em */etc/squid/squid.conf.* Caso a versão de instalação seja a partir da versão 3, o arquivo de configuração estará localizado em */etc/squid3/squid.conf*. Este arquivo vem originalmente com mais de 3000 linhas, mas a maioria destas são comentários explicando os comandos, que pode ser utilizado como material de apoio para implementação de novas regras.

[4] *Gopher* é um protocolo de redes de computadores que foi desenhado para distribuir, procurar e aceder a documentos na Internet. (WIKIPÉDIA, 2013)

Segue os comandos (*tags*) mais utilizados na configuração e suas descrições:

- **http_port**: define a porta em que o Squid ficará monitorando as requisições. Geralmente é a porta 3128, mas pode ser alterada com este comando.
  `http_port 3128`
- **visible_hostname**: define o nome do servidor *proxy*. Necessário para inicialização do Squid.
  `visible_hostname localhost`
- **cache_mem**: especifica a quantidade de memória dedicada para objetos em trânsito. Em toda especificação de quantidade de memória, deve-se colocar a sigla da unidade de medida desejada (KB, MB).
  `cache_mem 1 MB`
- **cache_dir**: especifica o formato de armazenamento do *cache*, seu diretório, a quantidade de espaço em disco e a quantidade de diretórios e subdiretórios que serão utilizados pelo Squid.
  `cache_dir ufs /var/cache/squid3 100 16 256`
- **maximum_object_size_in_memory**: define o tamanho máximo por arquivo de alocação na memória RAM.
  `maximum_object_size_in_memory 8 KB`
- **access_log**: define o diretório que será armazenado o arquivo de registro de acesso e tentativas de acesso aos conteúdos bloqueados. Deve-se mencionar o formato de gravação, para que os programas geradores de relatórios tenham compatibilidade com o formato.
  `access_log /var/log/squid3/access.log`
- **cache_mgr**: define o contato do administrador de redes.
  `cache_mgr webmaster`
- **error_directory**: define o diretório onde os arquivos de mensagens de erro ao usuário serão acessados. Por padrão, essas mensagens estão no idioma inglês mas podem ser modificadas para outros idiomas, assim como podem ser criadas mensagens próprias.
  `error_directory /usr/share/squid3/errors/Portuguese`

O comando mais importante do Squid é o ACL (*Access Control List*), pois faz a caraterização de objetos, como *hosts* (origem e destino), expressões regulares, hora, nomes de domínios, entre outras. O formato para se criar uma ACL é:

```
acl <nome_acl> <parâmentro> ["arquivo"|exp_reg|<IP>|<domínio>]
```

Segu, enão, uma lista de parâmetros utilizados na criação de uma ACL:

- **src**: utilizado para definir um conjunto de endereços IP ou de redes de origem. Exemplo:
  ```
  acl meu_ip src 192.168.5.18
  ```
- **dst**: utilizado para definir um conjunto de endereços IP ou de redes de destino. Exemplo:
  ```
  acl diretoria dst 192.168.5.53
  ```
- **srcdomain**: utilizado para identificar um conjunto de domínios de um *host* de origem. Exemplo:
  ```
  acl revista srcdomain www.abril.com.br
  ```
- **dstdomain**: utilizado para identificar um conjunto de domínios de um *host* de destino. Exemplo:
  ```
  acl noticia dstdomain www.terra.com.br
  ```
- **time**: utilizado para determinar dias da semana e horário, sendo que para os dias, deve-se usar letras equivalentes (S - domingo, M - segunda-feira, T - terça-feira, W - quarta-feira, H - quinta-feira, F - sexta-Feira, A - sábado). Exemplo:
  ```
  acl hora_intervalo time MTWHFA 12:00-13:00
  ```
- **url_regex**: utilizado para definir um conjunto de expressões regulares contidas nas URLs de páginas *web*. Deve-se ter cuidado ao definir estas expressões regulares, pois podem abranger outras expressões de efeito contrário. Exemplo:
  ```
  acl exp_liberada url_regex computador
  acl exp_negada url_regex puta
  ```
- **proto**: utilizado para definir um conjunto de protocolos. Exemplo:
  ```
  acl telnet proto telnet
  ```
- **port**: utilizado para definir um conjunto de portas de destino. Exemplo:
  ```
  acl safe_ports port 80 22
  ```
- **method**: utilizado para definir um conjunto de métodos de requisições HTTP.
  ```
  acl conexão_http method GET POST
  ```
- **arp**: utilizado para definir um conjunto de endereços MAC, que é o endereço de controle de acesso da placa de rede. Exemplo:
  ```
  acl mac_pc arp 0A:BF:32:23:FB:A0
  ```

Finalizado as ACLs, deve-se especificar seu tratamento, seja para liberar (*allow*), seja para bloquear (*deny*) o acesso. Para isto, existe mais um comando necessário que faz este tratamento. Sua sintaxe é a seguinte:

```
http_access [allow | deny] [!]<nome_acl_alvo>
```

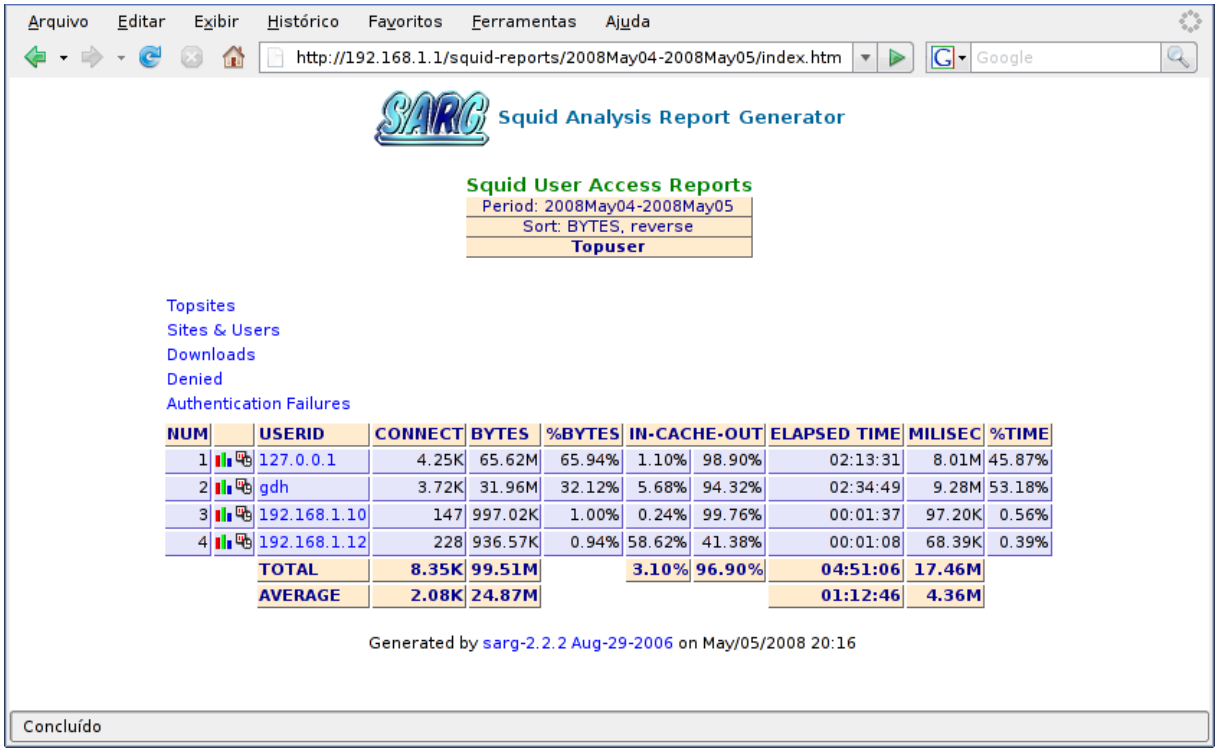
Assim como no *firewall* Iptables, a ordem das regras é importante e uma regra processada antes da outra tem prioridade sobre o tratamento. Por exemplo, se uma regra libera o acesso, mas logo em seguida tem uma regra negando o acesso, o pacote será liberado devido à prioridade de leitura. Portanto, deve-se observar atentamente a ordem das regras de liberação e negação, para que a escolha do administrador da rede tenha prioridade nas regras.

Para facilitar a leitura do arquivo com os registro de acessos e as tentativas a *sites* de conteúdo bloqueado, que contém inúmeras linhas com várias informações, existem os interpretadores de *log* que geram páginas em HTML. Um desses interpretadores é o SARG (*Squid Analisys Report Generator*), que pode ser visualizado em vários idiomas, inclusive o português.

Assim que instalado, ele já pode ser utilizado para gerar os relatórios com estatísticas, mas para melhorar o seu uso, algumas configurações podem ser modificadas. Assim como o Squid, o SARG tem um arquivo de configuração editável que se localiza em */etc/sarg/sarg.conf*. As configurações principais são:

- **language**: define o idioma de exibição do relatório. Por padrão, o idioma é o inglês.
- **access_log**: define o arquivo de registro que será interpretado. Este arquivo deve ser o mesmo configurado no Squid.
- **output_dir**: define o diretório que serão armazenados os relatórios gerados em HTTP.
- **exclude_hosts**: define os endereços IPs, rede ou domínios que serão excluídos do relatório.
- **exclude_users**: define os usuários da rede que serão excluídos do relatório.
- **topsites_num**: define a quantidade de *sites* que farão parte no relatório.

A figura 9 é um exemplo de página HTML gerada pelo SARG, relatando todo os dispositivos que acessaram a *Internet* no dia 05 de maio de 2008.



NUM		USERID	CONNECT	BYTES	%BYTES	IN-CACHE-OUT		ELAPSED TIME	MILISEC	%TIME
1		127.0.0.1	4.25K	65.62M	65.94%	1.10%	98.90%	02:13:31	8.01M	45.87%
2		gdh	3.72K	31.96M	32.12%	5.68%	94.32%	02:34:49	9.28M	53.18%
3		192.168.1.10	147	997.02K	1.00%	0.24%	99.76%	00:01:37	97.20K	0.56%
4		192.168.1.12	228	936.57K	0.94%	58.62%	41.38%	00:01:08	68.39K	0.39%
		TOTAL	**8.35K**	**99.51M**		**3.10%**	**96.90%**	**04:51:06**	**17.46M**	
		AVERAGE	**2.08K**	**24.87M**				**01:12:46**	**4.36M**	

Figura 9 - Exemplo de página HTML gerada pelo SARG

3 ESTUDO EXPERIMENTAL

Neste capítulo será colocado em prática, todo conhecimento teórico adquirido. Será feito uma análise do ambiente de rede do Instituto de Ensino Superior, bem como um entendimento das configurações dos aplicativos instalados no servidor *gateway*, a fim de prover segurança e controle de acesso à *Internet* da rede privada.

3.1 Ambiente de Rede

O ambiente de rede que será estudado será o de uma Instituição de Ensino Superior, localizado na cidade de Juiz de Fora/MG, que funciona com setores administrativos e salas de estudo, possuindo uma biblioteca e um pequeno laboratório de informática. A rede privada se constitui com os seguintes dispositivos:

- 11 estações de trabalho com placa de rede *ethernet*.
- 4 estações de estudo com placa de rede *ethernet*.
- 1 microcomputador, que servirá como *gateway* da rede, com sistema operacional Linux (distribuição Debian Lenny 5.0, *kernel* 2.6.26), com 3 placas de rede, possuindo *firewall* Iptables versão 1.4.6 e *proxy* Squid versão 3.0.STABLE.
- 1 concentrador (*switch*) *ethernet* 10/100 Mbits com 24 portas.
- 1 roteador Wi-Fi com 4 portas *ethernet* 10/100 Mbits.
- Conexão à *Internet* ligada a interface de rede eth0 do *gateway*.
- 3 impressoras com conexão de rede.

O servidor *gateway* ainda possui uma instalação VPN com topologia *Gateway-Gateway*, que o liga como servidor filial, a um servidor da matriz de endereço IP 192.168.250.5, localizado em Caratinga/MG. Para isto, foi instalado o aplicativo OpenVPN versão 2.1_rc8 i486 built on Jul 23 2008 e seu arquivo de configuração se encontra em */etc/openvpn/client.conf* e pode ser visualizado no ANEXO A. Com a VPN, os funcionários administrativos da filial podem acessar com segurança remotamente computadores com programas específicos de finanças, localizados na matriz.

A figura 10 representa um modelo da rede. Os computadores não foram separados por setores da instituição, mas sim pelo papel que eles desempenham e estes foram conectados todos em um *switch*, juntamente com impressoras que possuem conexão de rede. O roteador sem fio está ligado na terceira placa de rede do *gateway* e serve apenas para conexões Wi-Fi, sendo desabilitado seu serviço de DHCP[5]. A figura ainda apresenta uma representação do túnel VPN ligado ao servidor da matriz, através da interface virtual tap0.

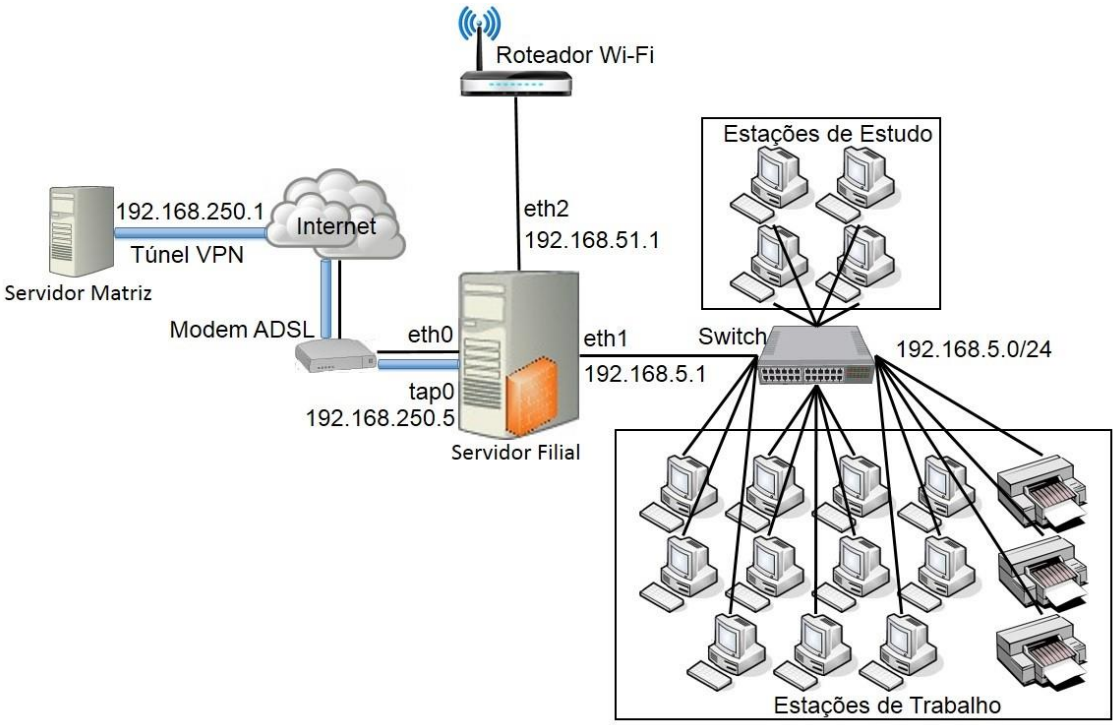


Figura 10 - Representação da rede do Instituto de Ensino Superior

A seguir, será feita uma análise das configurações de segurança da rede do Instituto, explicando passo-a-passo as regras utilizadas para a configuração da política de acesso e segurança da rede.

[5] DHCP: é um protocolo que por meio de um servidor, é capaz de distribuir automaticamente endereços de IP diferentes a todos os computadores à medida que eles fazem a solicitação de conexão com a rede. Essa distribuição dos IPs é feita em um intervalo pré-definido configurado no servidor. Sempre que um dos dispositivos for desconectado, o IP ficará livre para o uso em outra. (PEREIRA, 2009)

3.2 Firewall

Neste tópico, serão analisadas as regras que constituem o *firewall* do servidor *gateway*, por onde passam todas as requisições da rede empresarial. A análise será feita de modo que as regras podem ser analisadas individualmente ou em conjunto com outras, dependendo do contexto e da dependência entre elas.

O Iptables instalado foi a versão 1.4.6, sendo que atualmente ele está na versão 1.4.26. Seu diretório de instalação é */sbin/* e o arquivo de configurações do Iptables é um *script bash*, de nome *firewall.sh*, que é inicializado juntamente com o servidor e se encontra no diretório */root/bin/*. As linhas comentadas serão suprimidas da análise, mas as configurações das regras se encontram na integra no ANEXO B.

```
echo 1 > /proc/sys/net/ipv4/ip_forward #linha 03
```

Ativa o encaminhamento de pacotes no *kernel*. Se esta opção estiver desativada, os computadores não conseguirão se comunicar e nem ter acesso à *Internet*. Essa linha de comando dentro do *script* garante que o encaminhamento será realizado.

Como este arquivo de configuração está em linguagem de *script bash*, pode-se utilizar comandos para ajudar na criação de rotinas. Neste caso, foi utilizado a função de condições *case*, que se caso o usuário digitar no terminal do Linux `#/root/bin/firewall.sh start` (linha 6) ou `#/root/bin/firewall.sh stop` (linha 53), o *firewall* Iptables será iniciado ou parado, respectivamente.

```
/sbin/iptables -t nat -N WLANPOS #linha 07
/sbin/iptables -t nat -N WLANPRE #linha 08
/sbin/iptables -t filter -N WLAN # linha 10
```

Criação de duas cadeias na tabela NAT e uma cadeia na tabela *Filter*, com a finalidade de fazer tratamento de conexão Wi-Fi e autenticação da mesma. As cadeias WLANPRE e WLANPOS farão o tratamento NAT dos pacotes, enquanto que a cadeia WLAN fará o filtro de pacotes da rede sem fio.

```
/sbin/iptables –t filter -P FORWARD DROP #linha 13
```

Faz a mudança da política padrão da cadeia FORWARD para bloquear todo redirecionamento de pacotes que não satisfizer quaisquer regras impostas na cadeia, já que toda cadeia tem uma política padrão para liberar os pacotes.

```
/sbin/iptables –t filter -A FORWARD -s 192.168.5.0/24 -j ACCEPT #linha 16
/sbin/iptables –t filter -A FORWARD -d 192.168.5.0/24 -j ACCEPT #linha 17
/sbin/iptables –t filter -A INPUT -s 192.168.5.0/24 -j ACCEPT #linha 18
/sbin/iptables –t filter -A INPUT -d 192.168.5.0/24 -j ACCEPT #linha 19
```

Liberação da comunicação da rede corporativa. Assim, as estações de trabalho e de estudo podem trocar informações (FORWARD), podendo ser tanto a origem da requisição (`-s`), quanto o destino da obtenção da resposta (`-d`). O mesmo vale para a comunicação com o servidor que contém o *firewall* (INPUT).

```
/sbin/iptables –t filter -A FORWARD -s 192.168.51.0/24 -j WLAN #linha 22
/sbin/iptables –t filter -A FORWARD -d 192.168.51.0/24 -j WLAN #linha 23
/sbin/iptables –t filter -A INPUT -s 192.168.51.0/24 -j WLAN #linha 24
/sbin/iptables –t filter -A INPUT -d 192.168.51.0/24 -j WLAN #linha 25
```

Desvia o tratamento da comunicação da rede sem fio para a cadeia WLAN após a autenticação do usuário, que ao ser concretizada, o dispositivo terá o mesmo comportamento de comunicação contidas nas linhas 16 a 19, mas na rede sem fio.

```
/sbin/iptables –t filter -A INPUT -d 192.168.51.1 -p tcp --dport 3128
  -j DROP #linha 26
```

Bloqueia tentativas de quaisquer dispositivos da rede sem fio acessar diretamente o *proxy* (porta 3128) do *gateway*, evitando que este consiga alguma estratégia para burlar a autenticação da rede e tenha acesso à *Internet*.

```
/sbin/iptables -t nat -A PREROUTING -s 192.168.5.0/24 -p tcp -m tcp -
  -dport 80 -j REDIRECT --to-ports 3128 #linha 30
```

Faz o redirecionamento das requisições de páginas *web* (porta 80) para o *proxy* (porta 3128), fazendo com que este funcione de modo transparente para o usuário.

```
/sbin/iptables -t nat -A POSTROUTING -o eth1 -s 192.168.5.0/24 -p tcp
  --dport 80 -j MASQUERADE #linha 31
```

Realiza o mascaramento (NAT) da rede corporativa, que é a troca do endereço IP da estação, de onde saem as requisições de acesso, pelo endereço IP do *gateway*. Com isto, elas possuem acesso à *Internet*.

```
PORTAS_LIBERADAS_PARA_INTERNET="100 8017 12503 3328 993 1527 4550 3389
  3391 21 22 1823 4452 6891:6900 2222 25 6881 443 110 995 1414 5432
  2086 2082 2631 3306 3050 3007 80 8080 8001 8800 8880 42180 3456
  57028 465 5017 66 5006" #linha 33

for PORT in $PORTAS_LIBERADAS_PARA_INTERNET;do #linha 35
/sbin/iptables –t filter -A FORWARD -i eth1 -s 192.168.5.0/24 -o eth0
  -p tcp --dport $PORT -j ACCEPT #linha 36
/sbin/iptables -t nat -A POSTROUTING -s 192.168.5.0/24 -o eth0 -p tcp
  --dport $PORT -j MASQUERADE #linha 37
done #linha 38
```

Por se utilizar da linguagem de *script bash*, foi feito um vetor para armazenar as portas que devem ter acesso liberado na rede corporativa (cadeia FORWARD da tabela *Filter*) e na *Internet* (cadeia POSTROUTING da tabela NAT). Sendo assim, o comando de laço faz a varredura do vetor, inserindo as regras nas cadeias.

```
/sbin/iptables -t nat -A POSTROUTING -o eth0 -s 192.168.5.0/24 -p tcp
  --dport 5060 -j MASQUERADE
/sbin/iptables -t nat -A POSTROUTING -o eth0 -s 192.168.5.0/24 -p tcp
  --dport 1720 -j MASQUERADE
```

Faz a liberação de duas portas para comunicação com alguns serviços do servidor da matriz. Estas requisições não passam pelo *proxy*, pois utilizam portas diferentes da porta de requisição HTTP.

```
/sbin/iptables -t nat -A POSTROUTING -o eth0 -d
  187.63.203.16/255.255.255.240 -j MASQUERADE #linha 44
/sbin/iptables -t nat -A POSTROUTING -o eth0 -d 187.63.193.74 -j
  MASQUERADE #linha 45
```

Realiza o mascaramento da rede corporativa, cujos destinos são a rede e o servidor da matriz, onde a saída dos pacotes é a interface de rede que tem o modem ADSL (eth0). É neste servidor que é feita a conexão VPN e dá acesso remoto aos computadores da matriz com segurança.

```
/sbin/iptables –t filter -P FORWARD ACCEPT #linha 53
/sbin/iptables –t filter -P INPUT ACCEPT #linha 54

/sbin/iptables -t filter -F #linha 56
/sbin/iptables -t nat -F #linha 57

/sbin/iptables -t filter -X WLAN #linha 59
/sbin/iptables -t nat -X WLANPOS #linha 60
/sbin/iptables -t nat -X WLANPRE #linha 61
```

Faz a interrupção do *firewall*. Nas linhas 53 e 54, as políticas das cadeias são modificadas para aceitar qualquer pacote, já que o intuito não é mais utilizar o *firewall* e deixar aberto para qualquer conexão. Sendo assim, nas linhas 56 e 57 todas as regras são excluídas, limpando qualquer tipo de restrição de acesso à rede. Por fim, as cadeias que foram criadas são excluídas nas linhas 59 a 61, liberando-as da memória e até mesmo para uma possível reinicialização do *firewall*, fazendo com que as cadeias sejam novamente criadas e que nenhum tipo de erro apareça.

```
/sbin/iptables -t nat -A PREROUTING -s 192.168.51.0/24 -j WLANPRE #linha 48
/sbin/iptables -t nat -A PREROUTING -s 192.168.51.0/24 ! -d 192.168.51.1 -p
  tcp -m tcp -j REDIRECT --to-ports 80 #linha 49
/sbin/iptables -t nat -A POSTROUTING -s 192.168.51.0/24 -j WLANPOS #linha 50
```

Desvia o tratamento da comunicação da rede sem fio para que, assim que for autenticado, o dispositivo tenha acesso à *Internet* e aos outros dispositivos

conectados à rede sem fio. A linha 49 faz o redirecionamento de qualquer requisição, cujo protocolo de tráfego seja o TCP, para o servidor *web* (porta 80).

Portanto, quando o usuário se conecta ao roteador Wi-Fi e tenta acessar uma página, ele é redirecionado ao servidor *web* (porta 80) do *gateway*. Assim, abrirá uma página HTML para autenticação, onde será digitado o usuário e senha. Por este motivo, a regra da linha 49 exclui o *gateway* como destino de requisições na porta 80. Esta página HTML contém um código de autenticação em linguagem PHP, que pode ser visualizado no ANEXO C. A parte fundamental do código para este estudo se encontra nas seguintes linhas:

```
exec ("sudo iptables -A WLAN -s {$this->ip} -m mac --mac-source {$this-
  >mac} -j ACCEPT"); // linha 34
exec ("sudo iptables -A WLAN -s {$this->ip} -p tcp --dport 3128 -j
  ACCEPT"); // linha 35
exec ("sudo iptables -A WLAN -s {$this->ip} -p tcp --dport 443 -j
  ACCEPT"); // linha 36
exec ("sudo iptables -A WLAN -d {$this->ip} -j ACCEPT"); // linha 37

exec ("sudo iptables -t nat -A WLANPRE -s {$this->ip} -p tcp -m tcp -
  -dport 80 -j REDIRECT --to-ports 3128"); // linha 39

exec ("sudo iptables -t nat -A WLANPRE -s {$this->ip} -p tcp -m tcp -
  -dport 8080 -j REDIRECT --to-ports 3128"); // linha 40

exec ("sudo iptables -t nat -A WLANPOS -s {$this->ip} -p tcp --dport
  443 -j MASQUERADE"); // linha 42
exec ("sudo iptables -t nat -A WLANPRE -s {$this->ip} -p tcp --dport
  443 -j ACCEPT"); // linha 43
```

Todos os comandos anteriores estão relacionadas com o endereço IP do dispositivo autenticado, guardado na variável `ip` do objeto. Assim sendo, a linha 34 faz a atuação em baixo nível, vinculando o endereço IP do dispositivo ao seu endereço MAC, que é o endereço de controle de acesso da placa de rede. Para isto, nesta regra

foi utilizado um dos módulos do Iptables[6], que é o `mac` e a opção `--mac-source` (MAC de origem).

As demais linhas, assim como na rede corporativa, fazem a liberação de portas do *proxy* e HTTPS (porta 443) (linhas 35 e 36), liberação do trafego de pacotes da comunicação com os dispositivos da rede sem fio (linhas 37), ativação do *proxy* transparente (linha 39 e 40) e realização do NAT para acesso à *Internet* (linhas 42 e 43).

Esta foi a análise do *firewall* Iptables da rede privada do Instituto de Ensino Superior. A seguir, será feito a análise do *proxy* Squid.

3.3 Squid

Neste tópico, serão analisadas as regras que constituem o *proxy* do servidor *gateway*, por onde passam todas as requisições da rede empresarial. A análise será feita de modo que as regras podem ser analisadas individualmente ou em conjunto com outras, dependendo do contexto e da dependência entre elas.

O Squid instalado foi a versão 3.0.STABLE, sendo que atualmente ele está na versão 3.4.STABLE. O arquivo de configurações do Squid, de nome *squid.conf*, é inicializado juntamente com o servidor e se encontra no diretório */etc/squid3/*. Assim como no *firewall* Iptables, as linhas comentadas serão suprimidas da análise, mas as configurações das regras se encontram na integra no ANEXO D. O conteúdo dos arquivos utilizados nas configurações se encontram no ANEXO E, ordenados crescentemente pelo nome. Estes arquivos contêm endereços IP e rede, nomes de domínio, expressões regulares e extensões de arquivos que têm suas permissões liberadas ou negadas. Os arquivos */etc/squid3/rules/proxydeny* e */etc/squid3/rules/urldeny* não tiveram seus conteúdos mostrados, devido seus extensos tamanhos, justificado por ter um grande rigor na política de uso.

```
http_port 3128 transparent #linha 1
```

[6] Módulos do Iptables são formas de ampliar a funcionalidade da ferramenta Iptables (NETO, 2004). Nesta mesma referência, pode-se consultar todos os módulos disponíveis no Iptables.

Especifica a porta do *proxy* como sendo a porta 3128. Além disto, a opção `transparent` faz com o *proxy* seja transparente aos dispositivos.

```
hierarchy_stoplist cgi-bin ? #linha 2
```

Diz ao *proxy* Squid para buscar os dados diretamente na origem, sem passar pelos vizinhos na hierarquia. Esta configuração se refere a conteúdo dinâmico, evitando que o *cache* fique lotado, já que este conteúdo nunca é igual, impossibilitando de fazer seu reuso no *cache*, aumentando, assim, o desempenho.

```
access_log /var/log/squid3/access.log squid #linha 3
```

Especifica o diretório que armazena o arquivo de registros, que poderá ser utilizado para gerar relatórios de estatísticas. O formato deste arquivo é `squid`, que é compatível com o interpretador de *logs*, SARG.

```
refresh_pattern ^ftp: 1440 20% 10080 #linha 4
refresh_pattern ^gopher: 1440 0% 1440 #linha 5
refresh_pattern (cgi-bin|\?) 0 0% 0 #linha 6
refresh_pattern . 0 20% 4320 #linha 7
```

O comando `refresh_pattern` configura como serão tratados os tempos de vida dos objetos no *cache*, usando os valores passados por argumentos para verificar se os objetos armazenados são os mais recentes ou há a necessidade de atualizá-los. As linhas anteriores são configurações padrões do arquivo original, não tendo a necessidade de modificá-las.

```
icp_port 3130 #linha 8
```

Especifica a porta que será feito o tráfego de requisições ICP, que é um protocolo de comunicação entre *caches*.

```
coredump_dir /var/spool/squid3 #linha 9
```

Especifica o diretório que são gravados arquivos *core* em casos de falhas, que representam o estado do aplicativo no momento em que o sistema o finaliza.

```
error_directory  /etc/squid3/errors #linha 11
```

Especifica o diretório que contém arquivos de mensagens de erro. Este diretório possui 32 arquivos, sendo que apenas o arquivo ERR_ACCESS_DENIED foi modificado para apresentar uma mensagem de erro do Instituto, que é mostrado quando um acesso é bloqueado pelo *proxy*.

```
acl localhost src 127.0.0.1/32
acl to_localhost dst 127.0.0.0/8
```

Cria ACLs que especificam o próprio computador que contém o *proxy* como origem e destino de requisições.

```
acl manager proto cache_object #linha 13
```

```
http_access allow manager localhost #linha 81
http_access deny manager #linha 82
```

Cria uma ACL com o protocolo `cache_object` para fazer que o apenas próprio servidor administre as informações sobre o estado do Squid (linhas 81 e 82).

```
reply_body_max_size 100 MB #linha 41
```

Especifica o tamanho máximo permitido para *download* de arquivos em 100 MB. Acima deste limite, o arquivo será bloqueado e exibirá uma mensagem de erro no navegador.

No restante das linhas será feito o controle de acesso através da criação de ACLs e o tratamento destas, permitindo ou bloqueando o acesso a partir do comando `http_access`.

```
acl lanallow src "/etc/squid3/rules/lanallow" #linha 19
```

```
acl landst dst "/etc/squid3/rules/lanallow" #linha 21

no_cache allow landst #linha 86

http_access allow landst #linha 91

http_access allow lanallow #linha 111
```

As ACLs criadas fazem referência às redes de origem (src) e destino (dst) com permissões liberadas para utilizar o *proxy*. Essas redes são: a rede administrativa (192.168.5.0/24), a rede sem fio (192.168.51.0/24) e a VPN (192.168.250.0/24), sendo estas, origem e destino de quaisquer requisições. A regra da linha 86, faz com que um dispositivo da rede que foi acessado, possa ter suas informações armazenadas em *cache* pelo dispositivo que o acessar.

```
acl msnip src "/etc/squid3/rules/msnip" #linha 27
acl urlmsn dstdomain "/etc/squid3/rules/urlmsn" #linha 28
acl wordmsn url_regex -i gateway.dll #linha 29

http_access deny urlmsn wordmsn !msnip #linha 87
```

Faz o bloqueio do aplicativo de bate-papo MSN para todos com exceção do dispositivos de endereços IP contidos no arquivo */etc/squid3/rules/msnip*. Para isto, foram criadas três ACLs: uma com os endereços IP liberados, outra com os servidores de destino do aplicativo MSN e uma outra com uma expressão regular que aparece na URL, sendo que a opção `-i` deixa a expressão em *case insensitive*, ou seja, essa expressão é bloqueada independente se ela contiver letras maiúsculas e/ou minúsculas.

```
acl ipead src "/etc/squid3/rules/ipead" #linha 32
acl urlead dstdomain "/etc/squid3/rules/urlead" #linha 33

http_access allow urlead ipead #linha 84
```

Libera o acesso aos *sites* contidos no arquivo */etc/squid3/rules/urlead* para os endereços IP que estão no arquivo */etc/squid3/rules/ipead*.

```
acl bancoallow dstdomain "/etc/squid3/rules/bancoallow" #linha 36

http_access allow bancoallow #linha 79
```

Libera o acesso a *sites* de bancos, cujos endereços se encontram no arquivo */etc/squid3/rules/bancoallow*.

```
acl urlallow dstdomain "/etc/squid3/rules/urlallow" #linha 39

http_access allow urlallow #linha 80
```

Libera o acesso a todos os endereços IP e *sites* que estão no arquivo */etc/squid3/rules/urlallow*.

```
acl urldeny dstdomain "/etc/squid3/rules/urldeny" #linha 44

http_access deny urldeny #linha 92
```

Bloqueia o acesso aos *sites* que se encontram no arquivo */etc/squid3/rules/urldeny*. Este extenso arquivo contém mais de 1.300 *sites* com conteúdo adverso à política de uso do Instituto.

```
acl bloquear url_regex -i mail.google.com/mail/channel/bind #linha 47

http_access deny bloquear #linha 90
```

Bloqueia o bate-papo do *site* de *e-mails* Gmail. Para isto, se a URL contiver a expressão mencionada, em *case insensitive*, o *proxy* bloqueia o acesso.

```
acl worddeny url_regex "/etc/squid3/rules/worddeny" #linha 50

http_access deny worddeny #linha 103
```

Bloqueia o acesso a *sites* que contenham em sua URL alguma das expressões regulares que estão no arquivo */etc/squid3/rules/worddeny*.

```
acl radiodeny urlpath_regex -i "/etc/squid3/rules/radiodeny" #linha 50

http_access deny radiodeny #linha 96
```

Bloqueia o *download* de arquivos que contenham, na URL do *site*, alguma das expressões regulares que estão no arquivo */etc/squid3/rules/radiodeny*. Essas expressões são extensões de arquivo, que neste caso, são de áudio. O caractere `$` após as extensões significa que a expressão está localizada no final da URL.

```
acl proxydeny dstdomain "/etc/squid3/rules/proxydeny" #linha 55

http_access deny proxydeny #linha 99
```

Bloqueia o acesso à *sites* de *web proxy* disponíveis na *Internet*, com o intuito de evitar que o *proxy* do *gateway* da empresa seja burlado e o usuário tenha acesso a conteúdo bloqueado. O extenso arquivo utilizado no bloqueio está localizado em */etc/squid3/rules/proxydeny*, possuindo mais de 300 *sites*.

```
acl SSL_ports port 443 #linha 67
acl Safe_ports port 80 # http #linha 68
acl Safe_ports port 21 # ftp #linha 69
acl Safe_ports port 443 # https #linha 70
acl Safe_ports port 70 # gopher #linha 71
acl Safe_ports port 210 # wais #linha 72
acl Safe_ports port 1025-65535 # unregistered ports #linha 73
acl Safe_ports port 280 # http-mgmt #linha 74
acl Safe_ports port 488 # gss-http #linha 75
acl Safe_ports port 591 # filemaker #linha 76
acl Safe_ports port 777 # multiling http #linha 77
acl Safe_ports port 5060 #audio #linha 78
acl CONNECT method CONNECT #linha 79
acl Safe_ports port 5560        #VOIP #linha 80
http_access deny !Safe_ports #linha 83
```

```
http_access deny CONNECT !SSL_ports #linha 84
```

A ACL `Safe_ports` armazena uma lista de portas que têm acesso liberado para comunicação, sendo que as demais portas são bloqueadas (linha 83). A `SSL_ports` armazena uma lista de portas seguras, mas neste caso, tem apenas a porta 443 (porta https). Com isto, a linha 84 trata de bloquear o acesso a todas as portas, exceto as portas em `SSL_ports`, para conexão direta (CONNECT). Estas regras de acesso limitam as portas utilizadas pela rede.

```
http_access deny all #linha 103
icp_access deny all #linha 104
```

Para finalizar, todas as requisições que não satisfizerem as regras configuradas serão bloqueadas, definindo assim, uma política restritiva (linha 103). Também serão bloqueados todos os *caches* "vizinhos" de fazerem requisições ao *cache* local através do protocolo ICP (linha 104).

Esta foi a análise do *proxy* Squid da rede privada do Instituto de Ensino Superior. O servidor ainda tem um aplicativo instalado que é o interpretador de *logs*, SARG, na versão 2.2.5, sendo que atualmente ele está na versão 2.3.7. O arquivo de configurações do SARG se encontra no diretório */etc/squid/* e nome *sarg.conf*. Este arquivo não foi modificado, permanecendo as configurações originais.

Controlar os relatórios de estatísticas das filiais não é um trabalho muito fácil para a matriz, o que pode gerar risco à segurança da rede corporativa. A figura 11 comprova que o SARG foi pouco usufruído pelo administrador da rede e que em quase 3 anos de uso, foram poucos os relatórios de estatísticas gerados. A figura 12 mostra um relatório de acesso à *Internet* das estações do Instituto no dia 28 de janeiro de 2014.

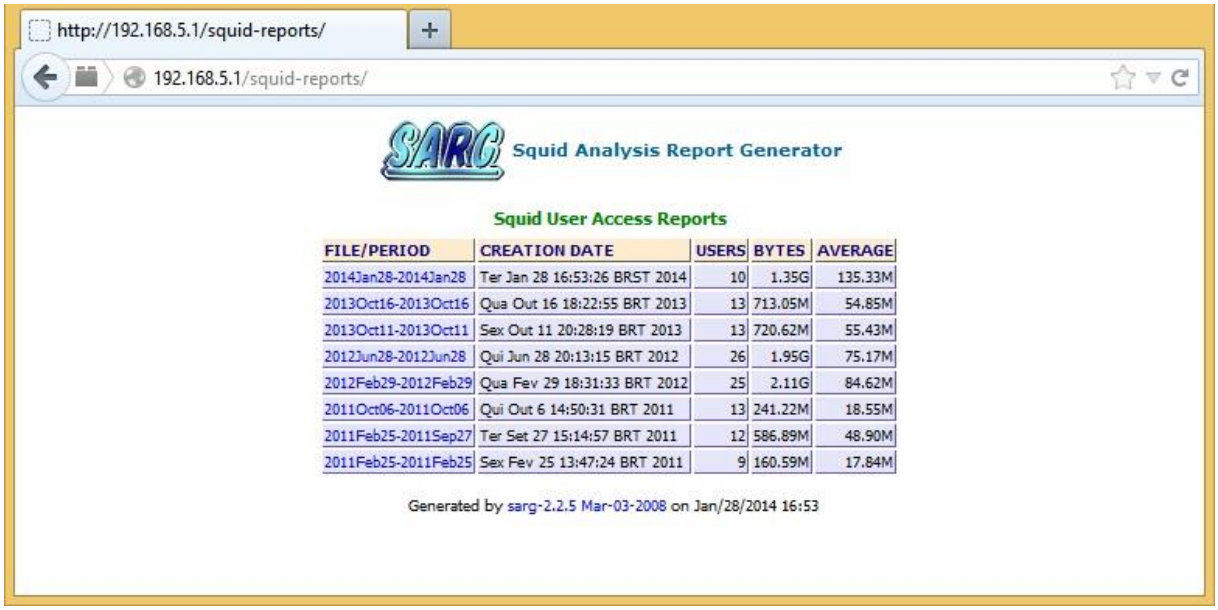


Figura 11 - Página inicial do SARG

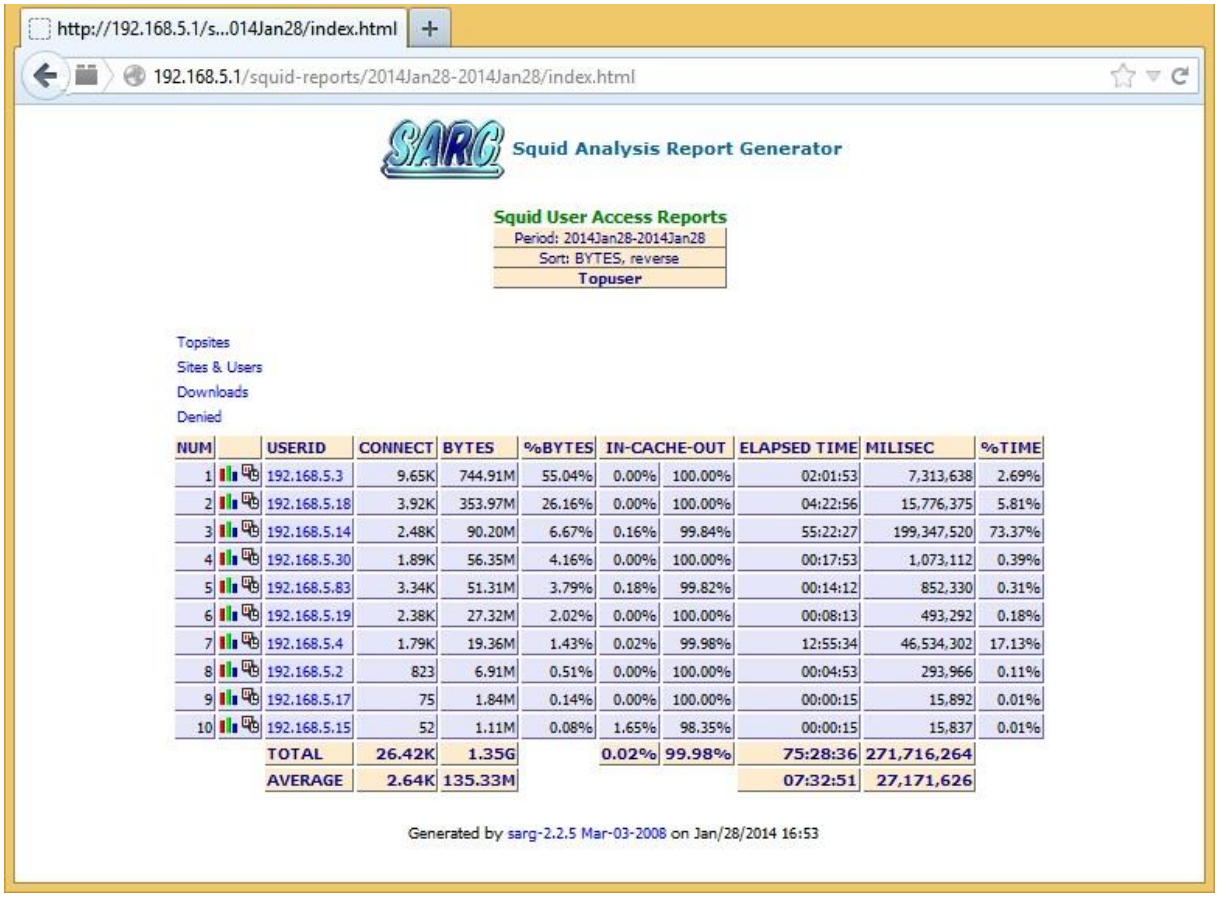


Figura 12 - Relatório gerado pelo SARG no dia 28 de janeiro de 2014

4 RESULTADOS

Após a análise e compreensão do ambiente de rede do Instituto, das configurações do VPN OpenVPN, do *firewall* Iptables e do *proxy* Squid instalados no servidor *gateway*, pode-se ter um levantamento de suas vantagens e desvantagens.

Feito isto, será apresentado propostas de melhoria em relação aos quesitos anteriormente citados.

4.1 Vantagens do Ambiente

Por se tratar de um ambiente de rede simples, configurar as estações não se torna um trabalho muito difícil, que é bom para se fazer a manutenção com mais atenção. Além disto, os relatórios gerados com estatísticas serão menores e mais fáceis de localizar possíveis ameaças à política de uso e à segurança da rede privada.

As vantagens da VPN no ambiente de rede do Instituto são: a matriz poder controlar os servidores *gateways* de suas filais, conseguindo fazer a manutenção destes sem se deslocar de sua cidade; disponibilização de um canal seguro para que as filiais possam acessar remotamente os computadores, localizados na matriz, que possuem aplicativos financeiro e com isso há uma centralização do banco de dados da empresa.

Tanto o *firewall* Iptables quanto o *proxy* Squid possuem configurações simples de regras e eficazes para a política de uso da rede empresarial. Além disto, ambos aplicativos possuem meios para controlar o acesso de cada dispositivo, sendo que o Squid possui uma configuração mais simples para se fazer o tratamento do controle de conteúdo, utilizando as listas de controle de acesso e aplicando a elas, as permissões necessárias.

Outra vantagem do Iptables no ambiente de rede é a criação das configurações de regras em tempo de execução, após autenticação de usuários na rede sem fio, que impede o acesso à *Internet* e à rede por estranhos, sendo registrado o endereço físico da placa de rede (endereço MAC) juntamente com o endereço IP fornecido pelo DHCP do servidor. Apenas usuários autorizados poderão usufruir dos benefícios da rede.

4.2 Desvantagens do Ambiente

Um possível ponto de falha do ambiente da rede privada é a junção das estações de trabalho com as estações de estudo, pois não se sabe qual a intenção do usuário ao utilizar a estação de estudo, até mesmo ingenuamente, pode trazer riscos à rede e tanto o servidor gateway quanto qualquer ponto da rede podem ser alvos de ataques.

A desvantagem da VPN é que se algum ponto (*gateway* matriz e *gateway* filial) não estiver disponível, a comunicação entre as redes não será possível por este canal seguro e os serviços que o servidor da matriz provê, não serão acessados.

Apesar da eficiência do *firewall* Iptables e do *proxy* Squid, o usuário da rede pode burlar o controle de acesso com um pouco de conhecimento sobre os protocolos HTTP e HTTPS. Ao acessar qualquer endereço *web* no navegador, o usuário utiliza implicitamente o protocolo HTTP (porta 80), que no Iptables é redirecionado para o Squid (porta 3128). Mas ao utilizar o protocolo HTTPS (porta 443) na barra de endereços do navegador, o Iptables não tem uma regra que faça o redirecionamento para o Squid, permitindo que o usuário possa acessar conteúdo até então bloqueado pela política de controle de acesso. Esta falha também acontece na rede sem fio, já que, no código de autenticação, a porta 443 não é redirecionada para a porta 3128.

Outra desvantagem das configurações das regras aplicadas no Iptables é que não existe nenhum conjunto de regras que faça a segurança do próprio servidor, deixando-o suscetível a ataques originados tanto de dentro da rede privada quanto da *Internet*.

4.3 Melhorias Propostas

Apontadas as vantagens e desvantagens na sessão anterior, existem algumas mudanças que podem ser feitas para melhoria da rede privada.

Assim como foi dito nas desvantagens, que as estações de estudo podem ser um ponto de ameaça à rede, pode-se desvinculá-las da rede corporativa e criar outra rede para acomodá-las.

Ainda mencionando as estações de estudo, pode-se criar regras no *firewall*, para que esta nova rede possua a mesma autenticação da rede sem fio, criando mais barreiras no uso destas estações e apenas pessoas autorizadas poderiam utilizá-las.

Alguns serviços são imprescindíveis no âmbito corporativo, ainda mais se tratando de matriz e suas filiais. Um destes serviços é a audioconferência, utilizada em reuniões de coordenadores e diretores regionais juntamente com o setor administrativo da matriz. No momento em que uma reunião começa, um pedaço razoavelmente grande da banda larga do provedor de *Internet* tem que estar disponível para o fluxo contínuo dos pacotes de áudio, sem que haja perda. Mas se ao mesmo tempo um usuário da rede descuidadamente fizer um grande tráfego de dados, como por exemplo, *download* de arquivos de tamanho elevado, ambos competirão pelo pedaço da banda. Visando evitar este caso de uso, existe uma opção eficaz para este problema, discutido a seguir.

O *firewall* Iptables possui uma tabela, que foi estudada, mas em nenhum momento foi utilizada, que se chama *Mangle*, que trata de forma especial os pacotes com maior prioridade. Se o tráfego partir do servidor ou da rede privada, será utilizado a cadeia OUTPUT, mas se o tráfego tem entrar no servidor ou na rede privada, então será utilizado a cadeia PREROUTING. Um exemplo será dado a seguir, para demostrar a sua utilização.

```
#iptables –t mangle –A OUTPUT –o eth0 –p tcp --dport 22 –j TOS –set-
  tos 16
#iptables –t mangle –A PREROUTING –i eth0 –p tcp --sport 22 –j TOS –
  set-tos 16
```

No exemplo anterior, foi dada prioridade máxima aos pacotes SSH para o tráfego de saída (`OUTPUT –o eth0`) e para o tráfego de entrada (`PREROUTING –i eth0`) e terão espera mínima para ter suas requisições tratadas.

Um ponto de falha foi apontado nas desvantagens, que é a utilização do protocolo HTTPS para burlar o *proxy*. Para contornar este problema, deve-se apenas acrescentar uma regra redirecionando a porta 443, que é a porta do protocolo HTTPS, para a porta do *proxy*, observando que já existem regras configuradas no Squid para tratamento do protocolo HTTPS.

Uma outra mudança que pode ser feita é utilizar o módulo do Iptables chamada multiport, onde pode-se passar por argumento várias portas ao mesmo tempo. Na configuração do Iptables do servidor, foi utilizada uma estrutura de um vetor para armazenar as portas e um laço para varrer este vetor. Com este módulo, pode-se especificar até 15 portas ao mesmo tempo, o que iria economizar linhas de código,

simplificando-o para facilitar sua compreensão. Segue um exemplo da utilização do módulo `multiport`.

```
/sbin/iptables -t nat -A PREROUTING -s 192.168.5.0/24 -p tcp -m
  multiport --dports 80 443 -j REDIRECT --to-ports 3128 #linha 30
```

A regra no exemplo anterior é uma sugestão, utilizando o módulo `multiport`, para contornar o problema de o usuário poder burlar o *proxy*. Lembrando que já existem regras configuradas de tratamento do protocolo HTTPS no Squid.

Um outra sugestão de melhoria seria a atualização dos aplicativos de segurança do servidor *gateway*, pois com o passar do tempo, novas ferramentas são desenvolvidas e antigas técnicas são aprimoradas.

Além disto, existe um outro aplicativo que pode ser instalado no servidor, chamado SquidGuard, que em conjunto com o *proxy* Squid, gera um banco de dados a partir de uma lista de *sites* para fazer o controle de acesso. As URLs podem ser separadas por classes e tratadas de formas diferentes, especificando ainda mais, quais dispositivos terão acesso permitido/bloqueado a determinados *sites*, pois ele faz um melhor gerenciamento dos usuários, com uma configuração mais simples e intuitiva do que as regras de ACL do Squid.

Para finalizar, podem ser criadas várias regras curtas de segurança para evitar ataques ao servidor *gateway*. Algumas regras serão apresentadas a seguir:

```
# Contra Sys-flood
iptables –A FORWARD –p tcp –m limit 1/s –j ACCEPT

# Contra ping da morte
iptables –A FORWARD –p icmp –icmp-type echo-request –m limit –
  limit 1/s –j ACCEPT

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

#Proteção contra ataques
iptables –A INPUT –m state –state INVALID –j DROP
```

5 CONCLUSÕES

Não se pode negar que a *Internet* traz benefícios ao desenvolvimento corporativo em uma rede privada. É por este motivo que as ferramentas de segurança devem ser instaladas e ter manutenções periódicas, visto que um ponto de falha poderia surgir; além de buscar implementar melhorias através de análises de relatórios para confrontar a utilização da rede com a política de acesso à *Internet*.

O objetivo de descrever as vantagens e desvantagens encontradas nos aplicativos de segurando foi cumprido com êxito no capítulo 4, mostrando os benefícios e alguns pontos de falha do ambiente de rede e das configurações das ferramentas. Além disso, neste mesmo capítulo foi indicado propostas de melhorias que podem resolver parcial ou totalmente as desvantagens discutidas.

Com o resultado obtido neste trabalho, pode-se concluir que o nível de segurança da rede privada do Instituto de Ensino Superior, localizado na cidade de Juiz de Fora/MG, é satisfatório, uma vez que, até o presente momento, não se teve nenhum incidente comprovado de ataque ao servidor, que desempenha um papel principal de controlar o acesso dos usuários (funcionários e alunos) ao conteúdo da *Internet*.

Não obstante a satisfatoriedade da segurança da rede privada, deve-se buscar aperfeiçoar o sistema, para evitar o surgimento de falhas e estudar novos casos de uso sobre possíveis incidentes e aplicar regras que irão impedi-los de acontecer.

Com o conhecimento obtido através da realização deste trabalho, é possível aliar a teoria à prática na manutenção das configurações de segurança estudadas, além de poder aplicar a quaisquer ambientes de rede, criando novas regras de política de controle de acesso com um bom nível de segurança.

Referências

ALECRIM, Emerson. **O que é firewall?** - Conceito, tipos e arquiteturas. InfoWester, 19 Fevereiro 2013. Disponível em: <http://www.infowester.com/firewall.php>. Acesso em: 17 de dezembro de 2013.

CERT.BR. **Incidentes Reportados ao CERT.br** -- Julho a Setembro de 2013. Comitê Gestor da Internet no Brasil. 2013. Disponível em: <http://www.cert.br/stats/incidentes/2013-jul-sep/total.html>. Acesso em: 2 de fevereiro de 2014.

CUNHA, Gerson Nunes da; FAGUNDES, Bruno Alves.; BORGES, Fábio. VPN: Protocolos e Segurança. **Revista de Engenharia da Universidade Católica de Petrópolis**, v. 3, p. 157-168, 2007.

EPAMINONDAS, Jocênio Marquios. **Uma Metodologia para Normatização de Correio Eletrônico em Organizações**. Dissertação (Mestrado em Engenharia de Produção). Universidade Federal de Santa Catarina. Florianópolis, p. 161. 2001.

FAGUNDES, Bruno Alves. **Uma Implementação de VPN**. Trabalho de Conclusão de Curso (Graduação em Ciência da Computação). Fundação de Apoio à Escola Técnica do Estado do Rio de Janeiro. Petrópolis, p. 76. 2007.

GOPHER. In: WIKIPÉDIA, a enciclopédia livre. Flórida: Wikimedia Foundation, 2013. Disponível em: <http://pt.wikipedia.org/wiki/Gopher>. Acesso em: 28 de janeiro de 2014.

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

PALLARES, Alberto Campos. **Redes e Sistemas de Telecomunicações**. Rio de Janeiro: Brasport, 2003.

PEREIRA, Ana Paula. **O que é DHCP?** TecMundo, 2009. Disponível em: <http://www.tecmundo.com.br/2079-o-que-e-dhcp-.htm>. Acesso em: 30 de janeiro de 2014.

SILVA, Gleydson Mazioli da. **Firewall Iptables** - Capítulo 10, Guia Foca GNU/Linux. 2007. Disponível em: <http://www.jsbusiness.com.br/foca/avancado/ch-fw-iptables.htm>. Acesso em: 19 de janeiro de. 2014.

SILVA, Herdwio Carvalho e. **Solução de controle de acesso a Internet em ambientes corporativos**. Trabalho de Conclusão de Curso (Graduação em Ciência da Computação). Universidade Federal de Lavras. Lavras, p. 57. 2005.

VIANNA, Gabriela. **O que é um Host?** TechTudo, 2012. Disponível em: < http://www.techtudo.com.br/artigos/noticia/2012/02/o-que-e-um-host.html>. Acesso em: 13 de fevereiro de 2014.

Anexos

ANEXO A

Conteúdo dos arquivos da configuração VPN no cliente.

"/etc/openvpn/cliente.conf"

```
01  #Nome_do_Servidor_VPN
02  remote   187.63.193.74
03  #porta de conexao
04  port 4550
05  #prontocolo
06  proto tcp-client
07  dev tap
08
09  # Certificados gerados
10  ca ca.crt
11  cert client.crt
12  key client.key
13
14  nobind
15  tls-client
16  pull
17  verb 4
18
19  ns-cert-type server
20
21  up "./vpn.sh"
```

"/etc/openvpn/vpn.sh"

```
01  #!/bin/bash
02
03  function ip () {
04  sleep 4
05  ifconfig tap0 192.168.250.5 netmask 255.255.255.0
06  route add -net 192.168.10.0 netmask 255.255.255.0 gw 192.168.250.254
07  route add -net 192.168.100.0 netmask 255.255.255.0 gw 192.168.250.100
08  }
09
10  ip &
```

ANEXO B

Conteúdo do arquivo de configuração das regras do *Firewall* Iptables.

"/root/sbin/firewall.sh"

```
01 #!/bin/bash
02
03 echo 1 > /proc/sys/net/ipv4/ip_forward
04
05 case $1 in
06 start)
07 /sbin/iptables -t nat -N WLANPOS
08 /sbin/iptables -t nat -N WLANPRE
09
10 /sbin/iptables -t filter -N WLAN
11 # Filter ############################
12 # Bloqueia acesso de todos nao cadastrados
13 /sbin/iptables –t filter -P FORWARD DROP
14
15 #JFADM
16 /sbin/iptables –t filter -A FORWARD -s 192.168.5.0/24 -j ACCEPT
17 /sbin/iptables –t filter -A FORWARD -d 192.168.5.0/24 -j ACCEPT
18 /sbin/iptables –t filter -A INPUT -s 192.168.5.0/24 -j ACCEPT
19 /sbin/iptables –t filter -A INPUT -d 192.168.5.0/24 -j ACCEPT
20
21 #Wireless
22 /sbin/iptables –t filter -A FORWARD -s 192.168.51.0/24 -j WLAN
23 /sbin/iptables –t filter -A FORWARD -d 192.168.51.0/24 -j WLAN
24 /sbin/iptables –t filter -A INPUT -s 192.168.51.0/24 -j WLAN
25 /sbin/iptables –t filter -A INPUT -d 192.168.51.0/24 -j WLAN
26 /sbin/iptables –t filter -A INPUT -d 192.168.51.1 -p tcp --dport 3128 -j DROP
27
28 # Nat ############################
29 #JFADM
30 /sbin/iptables -t nat -A PREROUTING -s 192.168.5.0/24 -p tcp -m tcp --dport 80 -j REDIRECT --to-ports 3128
31 /sbin/iptables -t nat -A POSTROUTING -o eth1 -s 192.168.5.0/24 -p tcp --dport 80 -j MASQUERADE
32 #liberando portas
33
```

```
    PORTAS_LIBERADAS_PARA_INTERNET="100 8017 12503 3328 993 1527 4550 3389 3391 21 22 1823 4452 6891:6900 2222 25 6881
34  443 110 995 1414 5432 2086 2082 2631 3306 3050 3007 80 8080 8001 8800 8880 42180 3456 57028 465 5017 66 5006"
35
36  for PORT in $PORTAS_LIBERADAS_PARA_INTERNET;do
37       /sbin/iptables -t filter -A FORWARD -i eth1 -s 192.168.5.0/24 -o eth0 -p tcp --dport $PORT -j ACCEPT
38       /sbin/iptables -t nat -A POSTROUTING -s 192.168.5.0/24 -o eth0 -p tcp --dport $PORT -j MASQUERADE
39  done
40
41  /sbin/iptables -t nat -A POSTROUTING -o eth0 -s 192.168.5.0/24 -p tcp --dport 5060 -j MASQUERADE
42  /sbin/iptables -t nat -A POSTROUTING -o eth0 -s 192.168.5.0/24 -p tcp --dport 1720 -j MASQUERADE
43
44  #Libera redes Doctum
45  /sbin/iptables -t nat -A POSTROUTING -o eth0 -d 187.63.203.16/255.255.255.240 -j MASQUERADE
46  /sbin/iptables -t nat -A POSTROUTING -o eth0 -d 187.63.193.74 -j MASQUERADE
47
48  #JFWireless
49  /sbin/iptables -t nat -A PREROUTING -s 192.168.51.0/24 -j WLANPRE
50  /sbin/iptables -t nat -A PREROUTING -s 192.168.51.0/24 ! -d  192.168.51.1 -p tcp -m tcp -j REDIRECT --to-ports 80
51  /sbin/iptables -t nat -A POSTROUTING -s 192.168.51.0/24 -j WLANPOS
52  ;;
53  stop)
54  /sbin/iptables -t filter -P FORWARD ACCEPT
55  /sbin/iptables -t filter -P INPUT ACCEPT
56
57  /sbin/iptables -t filter -F
58  /sbin/iptables -t nat -F
59
60  /sbin/iptables -t filter -X WLAN
61  /sbin/iptables -t nat -X WLANPOS
62  /sbin/iptables -t nat -X WLANPRE
63  ;;
    esac
```

ANEXO C

Conteúdo do arquivo utilizado para autenticação na rede sem fio.

"/var/www/wireless/class/auth.class.php"

```
01 <?php
02         require_once("fp.class.php");
03         /**
04          * Classe de Autenticação WireLess
05          *
06          * @author      Cícero Verneck Correa
07          * @since       06/04/2009
08          * @final       14/04/2009
09          */
10         class Auth{
11                 protected $ip;
12                 protected $mac;
13                 /**
14                  * Metodo Construtor
15                  *
16                  * @author      Cicero Verneck Correa
17                  * @since       06/04/2009
18                  * @final       14/04/2009
19                  * @return Auth
20                  */
21                 public function Auth(){
22                         $ipmac          =explode (' ',`/usr/sbin/arp -n | grep """$_SERVER[REMOTE_ADDR] """| sed 's/
   * / /g'|cut -d' ' -f1,3`);
23                         $this->ip       =$ipmac[0];
24                         $this->mac      =$ipmac[1];
25                 }
26                 /**
27                  * Metodo para Logar no firewall
28                  * @author      Cicero Verneck Correa
29                  * @since       06/04/2009
30                  * @final       14/04/2009
31                  */
32                 public function logar(){
```

```
33                          // Aceitando entrada
34                          exec ("sudo iptables -A WLAN -s {$this->ip} -m mac --mac-source {$this->mac} -j ACCEPT");
35                          exec ("sudo iptables -A WLAN -s {$this->ip} -p tcp --dport 3128 -j ACCEPT");
36                          exec ("sudo iptables -A WLAN -s {$this->ip} -p tcp --dport 443 -j ACCEPT");
37                          exec ("sudo iptables -A WLAN -d {$this->ip} -j ACCEPT");
38                          // Redirecionando a porta 80 para o squid
39                          exec ("sudo iptables -t nat -A WLANPRE -s {$this->ip} -p tcp -m tcp --dport 80 -j REDIRECT -
    -to-ports 3128");
40                          exec ("sudo iptables -t nat -A WLANPRE -s {$this->ip} -p tcp -m tcp --dport 8080 -j REDIRECT
    --to-ports 3128");
41                          // Aceitando o tráfego seguro para todos
42                          exec ("sudo iptables -t nat -A WLANPOS -s {$this->ip} -p tcp --dport 443 -j MASQUERADE");
43                          exec ("sudo iptables -t nat -A WLANPRE -s {$this->ip} -p tcp --dport 443 -j ACCEPT");
44                  }
45                  /**
46                   * Classe Para fazer um Log
47                   * @author      Cicero Verneck Correa
48                   * @since       06/04/2009
49                   * @final       14/04/2009
50                   * @param       String $data Dados para serem guardados no Log
51                   */
52                  public function log($data){
53                          $fp = new fp();
54                          $fp->file = "./log/gateway";
55                          $fp->mode = "a+";
56                          $fp->open();
57                          $fp->insert("{$data};{$this->ip};{$this->mac};".date("Y-m-d G:i")."\n");
58                          $fp->close();
59                  }
60          }
61  ~
62
```

ANEXO D

Conteúdo do arquivo de configuração das regras do *proxy* Squid.

"/etc/squid3/squid.conf"

```
001  http_port 3128 transparent
002  hierarchy_stoplist cgi-bin ?
003  access_log /var/log/squid3/access.log squid
004  refresh_pattern ^ftp: 1440 20% 10080
005  refresh_pattern ^gopher: 1440 0% 1440
006  refresh_pattern (cgi-bin|\?) 0 0% 0
007  refresh_pattern . 0 20% 4320
008  icp_port 3130
009  coredump_dir /var/spool/squid3
010
011  error_directory  /etc/squid3/errors
012
013  acl manager proto cache_object
014
015  acl localhost src 127.0.0.1/32
016  acl to_localhost dst 127.0.0.0/8
017
018  #Redes liberadas para acesso ao squid
019  acl lanallow src "/etc/squid3/rules/lanallow"
020
021  acl landst dst "/etc/squid3/rules/lanallow"
022
023  #Libera all ips
024  acl ipsallow src "/etc/squid3/rules/ipsallow"
025
026  #Bloqueia msn
027  acl msnip src "/etc/squid3/rules/msnip"
028  acl urlmsn dstdomain "/etc/squid3/rules/urlmsn"
029  acl wordmsn url_regex -i gateway.dll
030
031  #EAD
032  acl ipead src "/etc/squid3/rules/ipead"
033  acl urlead dstdomain "/etc/squid3/rules/urlead"
034
035  #Sites de Bancos
036  acl bancoallow dstdomain "/etc/squid3/rules/bancoallow"
037
038  #Sites Liberados
039  acl urlallow dstdomain "/etc/squid3/rules/urlallow"
040
041  reply_body_max_size 100 MB
042
043  #Sites Bloqueados
044  acl urldeny dstdomain "/etc/squid3/rules/urldeny"
045
046  #bate papo gmail
047  acl bloquear url_regex -i mail.google.com/mail/channel/bind
048
049  #Palavras bloqueadas
050  acl worddeny url_regex "/etc/squid3/rules/worddeny"
051  #Bloqueio de Radios
052  acl radiodeny urlpath_regex -i "/etc/squid3/rules/radiodeny"
053
054  #Proxys Bloqueados
055  acl proxydeny dstdomain "/etc/squid3/rules/proxydeny"
```

```
056
057 acl SSL_ports port 443
058 acl Safe_ports port 80 # http
059 acl Safe_ports port 21 # ftp
060 acl Safe_ports port 443 # https
061 acl Safe_ports port 70 # gopher
062 acl Safe_ports port 210 # wais
063 acl Safe_ports port 1025-65535 # unregistered ports
064 acl Safe_ports port 280 # http-mgmt
065 acl Safe_ports port 488 # gss-http
066 acl Safe_ports port 591 # filemaker
067 acl Safe_ports port 777 # multiling http
068 acl Safe_ports port 5060 #audio
069 acl CONNECT method CONNECT
070 acl Safe_ports port 5560        #VOIP
071 http_access allow manager localhost
072 http_access deny manager
073 http_access deny !Safe_ports
074 http_access deny CONNECT !SSL_ports
075
076 no_cache allow landst
077
078 http_access allow ipsallow
079 http_access allow bancoallow
080 http_access allow urlallow
081 http_access allow landst
082
083 #EAD
084 http_access allow urlead ipead
085
086 #msn
087 http_access deny urlmsn wordmsn !msnip
088
089 #Bate Papo Gmail
090 http_access deny bloquear
091
092 http_access deny urldeny
093 http_access deny worddeny
094
095 #Radio
096 http_access deny radiodeny
097
098 #Proxy Bloqueador
099 http_access deny proxydeny
100
101 http_access allow lanallow
102
103 http_access deny all
104 icp_access deny all
```

ANEXO E

Conteúdo dos arquivos que são utilizados pelo *proxy* Squid.

E.1: "/etc/squid3/rules/bancoallow"

```
01 .itau.com.br
```

E.2: "/etc/squid3/rules/ipead"

```
01 192.168.5.200 #Voip
02 192.168.5.207
03 192.168.5.180 #Americo
04 192.168.5.12
05 192.168.5.201 #Jesse
06 192.168.5.11
```

E.2: "/etc/squid3/rules/ipsallow"

```
01 192.168.5.254
02 192.168.5.19
03 192.168.5.18
04 192.168.5.11
05 192.168.10.135
06 192.168.10.94
```

E.3: "/etc/squid3/rules/lanallow"

```
01 192.168.5.0/24
02 192.168.250.0/24
03 192.168.51.0/24
```

E.4: "/etc/squid3/rules/msnip

```
01 192.168.5.200 #Voip
02 192.168.5.207
03 192.168.5.12
04 192.168.5.201
```

E.5: "/etc/squid3/rules/proxydeny"

E.6: "/etc/squid3/rules/radiodeny"

```
01 \.aif$
02 \.aifc$
03 \.aiff$
04 \.asf$
05 \.asx$
06 \.au$
07 \.avi$
08 \.m1v$
09 \.m3u$
10 \.med$
11 \.mid$
12 \.midi$
13 \.mov$
14 \.mp2$
15 \.mp2v$
16 \.mp3$
17 \.mpa$
18 \.mpe$
19 \.mpeg$
20 \.mpg$
21 \.ogg$
22 \.pls$
23 \.ra$
24 \.ram$
25 \.rmi$
26 \.snd$
27 \.wma$
28 \.wmv$
29 \.wvx$
```

E.7: "/etc/squid3/rules/urlallow"

```
01 .itau.com.br
02 .blogspot.com
03 .google.com.br
04 .doctum.edu.br
05 192.168.250.150
06 .geocities.com
07 .mercadolivre.com.br
08 .www.terra.com.br
09 200.202.193.19
10 .buntuforum-br.org
11 .telemar-mg.com.br
12 .peper24horas.com.br
13 .unicamp.br
14 .farv.com.br:81/unipac/candida
15 to/inscricao.php
16 .farv.com.br
17 .farv.com.br:81
18 login.yahoo.com
19 .mail.yahoo.com
20 201.59.8.121/255.255.255.248
21 adx.doctum.com.br
22 central.doctum.com.br
```

E.8: "/etc/squid3/rules/urldeny"

E.9: "/etc/squid3/rules/urlead"

```
01 .orkut.com
02 .orkut.com.br
03 .youtube.com
04 .gmodules
05 .superdowloads.com.br
06 .ocio2007.com.br
07 .4shared.com
08 .easy-share.com
```

```
09 .rapidshare.com
10 .sexshare.com
11 .zshare.net
12 .megaupload.com
```

E.10: "/etc/squid3/rules/urlmsn"

```
01 passport.com
02 msn.com.br
03 msn.com
04 sc.msn.com
05 www.msn.be
06 207.46.110.11
07 messenger.msn.com.br
08 messenger.msn.com
09 http.msg.yahoo.com
10 nickname.msn.com.br
11 chat.msn.com
12 chat.msn.com.br
13 msgr.hotmail.com
14 gateway.messenger.hotmail.com
15 http1.msgr.hotmail.com
16 http2.msgr.hotmail.com
17 http3.msgr.hotmail.com
18 http4.msgr.hotmail.com
19 http5.msgr.hotmail.com
20 http6.msgr.hotmail.com
21 http7.msgr.hotmail.com
22 http8.msgr.hotmail.com
23 http9.msgr.hotmail.com
24 http10.msgr.hotmail.com
25 http11.msgr.hotmail.com
26 http12.msgr.hotmail.com
27 http13.msgr.hotmail.com
28 http14.msgr.hotmail.com
29 http15.msgr.hotmail.com
30 http16.msgr.hotmail.com
31 http17.msgr.hotmail.com
32 http18.msgr.hotmail.com
33 http19.msgr.hotmail.com
34 http20.msgr.hotmail.com
```

E.11: "/etc/squid3/rules/worddeny"

```
001 alleensletjes
002 amateuratope
003 atunnel
004 b3u
005 babes
006 BackFox
007 bangbros
008 barely18teens
009 batepapo
010 belladasemana
011 beltrano
012 blig
013 boquete
014 bps
015 btunnel
016 buceta
017 caseira
018 caseiro
019 chatter
020 depravado
021 ebuddy
022 e-messenger
023 festasefotos
024 fliper
025 fliperama
026 flog
027 flogao
028 folia
029 fotolog
030 freeproxy
031 galeradofofante
032 galleries
033 gay
034 games
035 gazzag
036 gigafoto
037 girl
038 goproxing
039 gostosa
040 guardster
041 hentai
042 hi5
043 icq
044 idzap
045 iloveim
046 ILoveIM
047 jogos
048 kproxy
049 kut
050 marcinha
051 meebo
052 megaproxy
053 messbrasil
054 msnanywhere
055 muie
056 mxds.ch
057 myshield
058 ninfetas
059 nncuteteenagers
060 nproxy
061 nubiles
062 ogame
063 oilfight
064 orkut
065 pelada
066 phonefox
067 pixiespillows
068 playboy
069 pokemoncrater
070 pombaloca
071 porn
072 porno
073 porra
074 proxify
075 #proxy
076 proxyweb
077 putaria
```

078 redsex
079 sacanagem
080 safada
081 safadinhas
082 safehazard
083 safelizard
084 sex-tadela
085 sexy
086 shadowbrowser
087 shoosh
088 shooshtime
089 snoopyblocker
090 spynot
091 teen
092 theboy
093 thegirl
094 thumbs
095 tuf
096 unipeak
097 urlencoded
098 userbeam
099 videoslegais
100 virgulagirl
101 virtualfest
102 wproxy
103 xoxota
104 webmessenger
105 ibypass