FACULDADE DE TECNOLOGIA DE AMERICANA

Curso Superior de Tecnologia em Segurança da Informação

Jânilson Neves Souza

Proxy squid

Uma solução eficiente de proxy

Americana, SP

2016

FACULDADE DE TECNOLOGIA DE AMERICANA

Curso Superior de Tecnologia em Segurança da Informação

Jânilson Neves Souza

Proxy squid

Uma solução eficiente de proxy

Trabalho de Conclusão de Curso desenvolvido em cumprimento à exigência curricular do Curso Superior de Tecnologia em Segurança da Informação, sob a orientação do Prof. Esp. Rogério Nunes de Freitas

Área de concentração: Segurança da Informação.

Americana, SP.

2016

FICHA CATALOGRÁFICA – Biblioteca Fatec Americana - CEETEPS
Dados Internacionais de Catalogação-na-fonte

S715p

SOUZA, Jânilson Neves
Proxy squid: uma solução eficiente de proxy. / Jânilson Neves Souza. – Americana: 2016.
52f.

Monografia (Curso de Tecnologia em Segurança da Informação). - - Faculdade de Tecnologia de Americana – Centro Estadual de Educação Tecnológica Paula Souza.
Orientador: Prof. Esp. Rogério Nunes de Freitas

1. Segurança em sistemas de informação I. FREITAS, Rogério Nunes de II. Centro Estadual de Educação Tecnológica Paula Souza – Faculdade de Tecnologia de Americana.

CDU: 681.518.5

Jânilson Neves de Souza

PROXY SQUID

Trabalho de graduação apresentado como exigência parcial para obtenção do título de Tecnólogo em Segurança da Informação pelo CEETEPS/Faculdade de Tecnologia – FATEC/ Americana.
Área de concentração: Segurança da informação.

Americana, 08 de dezembro de 2016.

Banca Examinadora:

Rogerio Nunes de Freitas (Presidente)
Especialista
Fatec Americana

Rodrigo Nogueira Tofani (Membro)
Especialista
Fatec Americana

Alberto Martins Junior (Membro)
Mestre
Fatec Americana

AGRADECIMENTOS

Gostaria de agradecer aos meus pais, irmãos, colegas e amigos pelo apoio sempre prestado e, ao meus professores, em especial ao meu orientador que tanto ajudou para que eu conseguisse elaborar esse trabalho.

DEDICATÓRIA

Dedico este trabalho aos meus pais e meus irmãos que me apoiaram nesta jornada.

RESUMO

A Tecnologia da Informação (TI) tem uma presença forte, e importante papel no cotidiano das pessoas, e nas tarefas diárias de empresas e demais organizações, sejam públicas ou privadas. Devido a este fato, a TI se tornou muito importante para todas as pessoas e organizações. O objetivo desse trabalho é abordar itens pertinentes à Tecnologia da Informação, e de maneira mais específica abordar uma de suas tecnologias que tem por finalidade auxiliar na segurança da informação e na disponibilidade dos serviços providos pela TI, o *proxy squid*. O *proxy squid* foi o escolhido por sua eficiência no que se propõe a fazer e, principalmente por ser um *software* gratuito. Essa escolha deu-se também em função de consultas e pesquisas qualitativas em livros e artigos correlacionados. Com isto conseguiu-se adquirir uma perfeita análise do melhor *software* de *proxy* existente no mercado atual. Por fim foi feito um experimento mostrando a implementação do *software squid* em um ambiente virtual composto por um servidor e dois computadores clientes. Neste trabalho foi então apresentado o funcionamento do *proxy squid*, bem como suas configurações e bloqueios a *sites* da Internet, mostrando ainda que é possível e viável sua implementação para um melhor controle de tráfego em uma rede de computadores.

Palavras chaves: *proxy*; *squid*; informação.

ABSTRACT

Information Technology (IT) has a strong presence and important role in the daily lives of people and in the daily tasks of companies and other organizations, whether public or private. Due to this fact, IT has become very important for all people and organizations. The purpose of this work is to address issues related to Information Technology, and more specifically to address one of its technologies that aims to assist in information security and availability of services provided by IT, the *proxy squid*. The *proxy squid* was chosen because of its efficiency in what it proposes to do and mainly because it is free software. This choice was also due to queries and qualitative research in books and correlated articles. With this it was possible to acquire a perfect analysis of the best existing *proxy* software in the market today. Finally, an experiment was done showing the implementation of the *squid* software in a virtual environment composed of a server and two client computers. In this work, it was presented the operation of the *proxy squid*, as well as its configurations and blocks to Internet sites, showing that it is possible and feasible to implement it to better control traffic in a computer network.

Keywords: *proxy*; *squid*; information.

Sumário

LISTA DE FIGURAS

SIGLAS

DHCP – Dynamic Host Configuration Protocol.
DNS – Domain Name System.
ACL – Access Control List.
IP – Internet Protocol.
TI – Tecnologia da Informação.
RFC – Request for Comments.

1. INTRODUÇÃO

Atualmente a tecnologia da informação está inserida em todos os segmentos, seja social, empresarial ou governamental. Está presente no cotidiano de todos, e a cada dia que passa esses segmentos se tornam mais dependentes dela dada sua importância. Diante dessa magnitude em que se tornou a esfera da tecnologia da informação surgiu a necessidade de se buscar meios de tornar seus serviços cada vez mais disponíveis e na mesma proporção, seguro à todos que de seus serviços faz uso.

O trabalho tem como objetivo abordar de maneira simplificada os assuntos pertinentes à tecnologia da informação, mais especificamente sobre a segurança da informação, mas com a finalidade maior de falar de um software que auxilia em todo esse processo de se garantir a segurança da informação, será falado do *proxy squid*. Primeiramente será falado da segurança da informação, da importância em proteger as informações contra acesso não autorizado, garantido segundo os pilares da segurança da informação, a disponibilidade, integridade e confidencialidade dos dados, para que elas estejam sempre disponíveis e integras quando solicitadas, porém, disponíveis apenas para pessoas autorizadas.

Depois serão abordados os mais conhecidos serviços de rede. Um breve comentário sobre DHCP (Dynamic Host Configuration Protocol), DNS (Domain Name System), *firewall* e *proxy*. Destes, o *proxy* será o item mais abrangido, será falado do que é um servidor *proxy* e sobre os principas tipos existentes, bem como as vantagens em se utilizar cada um deles.

Na sequência o item abordado será o *squid*. Nesta seção será falado o que é um *proxy squid*, sua história, como funciona e qual a sua função entre a rede pública e a rede privada.

Por fim será feita um experimento de implementação do *proxy squid*. O ambiente utilizado para tal experimento será um ambiente virtual, onde através do *software* VirtualBox será instalado um servidor Ubuntu onde o *squid* e seus serviços serão instalados, e ainda dois computadores clientes que serão utilizadas para demonstrar

principalmente os bloqueios a *sites* da Internet predeterminados e configurado dentro do sistema *squid*. O experimento tem por finalidade demonstrar que é possível e viável instalar um *proxy squid* dentro de qualquer organização, para fazer o controle de tráfego de rede, melhorar seu desempenho com o uso de *cache* e bloqueio a *sites* que não façam parte do escopo e não sejam interessantes aos colaboradores da mesma acessarem.

Também será configurado no servidor Ubuntu o serviço de DHCP, para distribuir de maneira dinâmica IPs aos computadores da rede. Será mostrado uma maneira rápida e fácil de se fazer essa configuração, para que não haja a necessidade de configurar os IPs em cada computador da rede de maneira manual.

2. SEGURANÇA DA INFORMAÇÃO

Nos dias atuais o mundo é composto e interligado por informações. Não é segredo para ninguém a sua grande importância em serem protegidas estas informações, sejam elas para pessoas, empresas, governos, enfim, para todos. Seria um prejuízo irreparável se uma empresa tivesse que começar a buscar clientes do zero, por exemplo, caso o seu banco de dados de clientes fosse perdido por um desastre natural, ou por algum outro motivo. Poderia ser também irremediável a perda e/ou vazamento de dados de um governo, o qual deveria ser tratado com o maior grau de sigilo por ser segredo de estado. Então, para dar suporte à toda esta questão tem-se a segurança da informação, definida em NBR ISO 17799 (2005) como sendo a proteção da informação de vários tipos de ameaças para garantir a continuidade do negócio, minimizar o risco ao negócio, maximizar o retorno sobre os investimentos e as oportunidades de negócio.

A informação é descrita em NBR ISO 17799 (2005), como um ativo que, como qualquer outro ativo importante, é essencial para os negócios de uma organização e consequentemente necessita ser adequadamente protegida. Para complementar o conceito de informação, Sêmola (2014) afirma que informação é o conjunto de dados utilizados para a transferência de uma mensagem entre indivíduos e/ou máquinas em processos comunicativos (isto é, baseados em troca de mensagens) ou transacionais (isto é, processos em que sejam realizadas operações que envolvam, por exemplo, a transferência de valores monetários). Afirma ainda que a informação pode estar presente em inúmeros elementos desse processo, chamados ativos, os quais são alvo de proteção da segurança da informação, ou ser manipulada por eles. O mesmo autor define um ativo como:

> “É todo elemento que compõe os processos que manipulam e processam a informação, a contar da própria informação, o meio em que ela é armazenada, os equipamentos em que é manuseada, transportada e descartada.”

Diante dessa afirmação, pode se dizer que ativo no contexto geral é todo e qualquer elemento que tem valor para a organização e/ou indivíduo, fazendo necessário a busca de meios para garantir a sua proteção de forma a garantir também a continuidade dos negócios.

Para dar mais enfase a importância de se proteger os ativos da informação, Sêmola (2014) define segurança da informação como uma área do conhecimento dedicada à proteção de ativos da informação contra acessos não autorizados, alterações indevidas ou sua indisponibilidade. E de forma mais ampla o mesmo autor a considera como a prática de gestão de riscos em incidentes que impliquem o comprometimento dos três principais conceitos da segurança: confidencialidade, integridade e disponibilidade da informação. Estes que serão vistos mais adiante.

2.1. A IMPORTÂNCIA DA SEGURANÇA DA INFORMAÇÃO

Antes de falar dos três conceitos que são considerados os pilares da segurança da informação é necessário ressaltar que a segurança da informação é tida como de fundamental importância para manutenção e continuidade dos negócios, seja para os setores públicos e/ou setores privados, como afirma NBR ISO 17799 (2005):

> "A segurança da informação é importante para os negócios, tanto do setor público como do setor privado, e para proteger as infraestruturas críticas. Em ambos os setores, a função da segurança da informação é viabilizar os negócios como o governo eletrônico (*e-gov*) ou o comércio eletrônico (*e-business*), e evitar ou reduzir os riscos relevantes. A interconexão de redes públicas e privadas e o compartilhamento de recursos de informação aumentam a dificuldade de se controlar o acesso. A tendência da computação distribuída reduz a eficácia da implementação de um controle de acesso centralizado."

Com a ideia acima, a segurança da informação se torna indispensável quando se fala em continuidade do negócio, uma vez que sua função é relatada como sendo exatamente a de viabilizar a continuidade dos negócios, evitando e reduzindo desta forma os riscos mais relevantes que poderiam vir a comprometer a continuidade dos negócios para toda e qualquer organização.

2.2. OS TRÊS PILARES DA SEGURANÇA DA INFORMAÇÃO

Como foi falado mais acima, a segurança da informação gira em torno de três principais conceitos de segurança que são considerados os pilares da segurança da informação, a confidencialidade, a integridade e a disponibilidade. Estes três conceitos juntos garantem segundo Sêmola (2014) a proteção de ativos da informação contra acessos não autorizados, alterações indevidas ou sua indisponibilidade. Logo abaixo tem-se mais informações sobre cada um desses conceitos na visão do mesmo autor.

2.2.1. CONFIDENCIALIDADE

Toda informação deve ser protegida de acordo com o grau de sigilo de seu conteúdo, visando a limitação de seu acesso e uso apenas às pessoas a quem é destinada.

2.2.2. INTEGRIDADE

Toda a informação deve ser mantida na mesma condição em que foi disponibilizada pelo seu proprietário, visando protegê-la contra alterações indevidas, intencionais ou acidentais.

Laureano (2005) destaca ainda que o item integridade não pode ser confundido com a confiabilidade do conteúdo (seu significado) da informação. Uma informação pode ser imprecisa, mas deve permanecer integra (não sofrer alterações por pessoas não autorizadas).

2.2.3. DISPONIBILIDADE

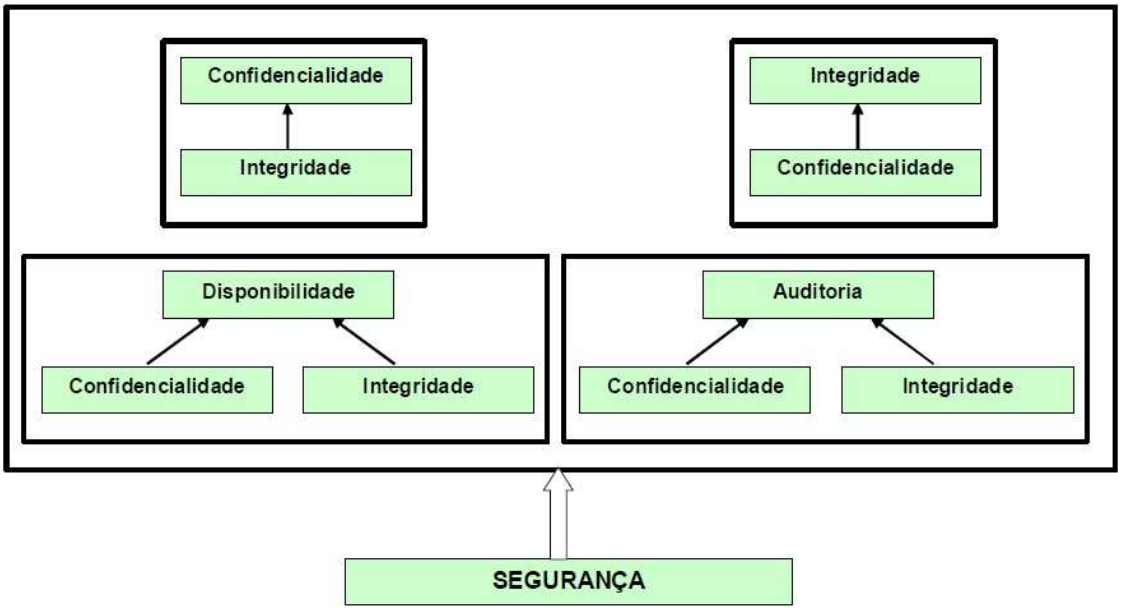
Toda a informação gerada ou adquirida por um indivíduo ou instituição deve estar disponível aos seus usuários no momento em que eles necessitem delas para qualquer finalidade.

Laureano (2005) ressalta ainda que para uma segurança da informação efetiva e verdadeiramente segura, alguns outros conceitos devem ser respeitados e seguidos, como:

- **Autenticidade** – Garante que a informação ou o usuário da mesma é autêntico; Atesta com exatidão, a origem do dado ou informação;
- **Não repúdio** – Não é possível negar (no sentido de dizer que não foi feito) uma operação ou serviço que modificou ou criou uma informação; Não é possível negar o envio ou recepção de uma informação ou dado;
- **Legalidade** – Garante a legalidade (jurídica) da informação; Aderência de um sistema à legislação; Característica das informações que possuem valor legal dentro de um processo de comunicação, onde todos os ativos estão de acordo com as cláusulas contratuais pactuadas ou a legislação política institucional, nacional ou internacional vigentes.
- **Privacidade** – Foge do aspecto de confidencialidade, pois uma informação pode ser considerada confidencial, mas não privada. Uma informação privada deve ser vista / lida / alterada somente pelo seu dono. Garante ainda, que a informação não será disponibilizada para outras pessoas (neste caso é atribuído o caráter de confidencialidade a informação); É a capacidade de um usuário realizar ações em um sistema sem que seja identificado.
- **Auditoria** – Rastreabilidade dos diversos passos que um negócio ou processo realizou ou que uma informação foi submetida, identificando os participantes, os locais e horários de casa etapa. Auditoria em software significa uma parte da aplicação, ou conjunto de funções do sistema, que viabiliza uma auditoria; Consiste no exame do histórico dos eventos dentro de um sistema para determinar quando e onde ocorreu uma violação de segurança.

Para ilustrar melhor este conceito, tem-se a Figura 1 logo a seguir, mostrando a junção dos itens confidencialidade, integridade e disponibilidade citados acima, com a finalidade de proporcionar ao final do processo a permanente segurança da informação.

Figura 1: A relação dos princípios para a obtenção da segurança da informação.



Fonte: (LAUREANO, 2005, p. 13)

Laureano (2005) explica a imagem acima da seguinte forma:

> "A confidencialidade é dependente da integridade, pois se a integridade se um sistema for perdida, os mecanismos que controlam a confidencialidade não são mais confiáveis. A integridade é dependente da confidencialidade, pois se alguma informação confidencial for perdida (senha de administrador do sistema, por exemplo) os mecanismos de integridade podem ser desativados. Auditoria e disponibilidade são dependentes da integridade e confidencialidade, pois estes mecanismos garantem a auditoria do sistema (registros históricos) e a disponibilidade do sistema (nenhum serviço ou informação vital é alterado)."

Portanto, como mostra a Figura 1 a segurança da informação só é completa com a junção de todos esses conceitos envolvidos no processo, assegurando desta forma a verdadeira segurança sobre aquele que é considerado o principal ativo das organizações, a informação.

3. SERVIÇOS DE REDE

Tanenbaum (2003) fala que um serviço é especificado formalmente por um conjunto de primitivas (operações) disponíveis para que um processo do usuário acesse o serviço. E que essas primitivas informam ao serviço que ele deve executar alguma ação ou relatar uma ação executada por uma entidade par.

Dentre os serviços de rede, os mais conhecidos são: DHCP, DNS, Firewall e *Proxy*. O DHCP e o DNS são assim especificados por Ball e Duff (2004):

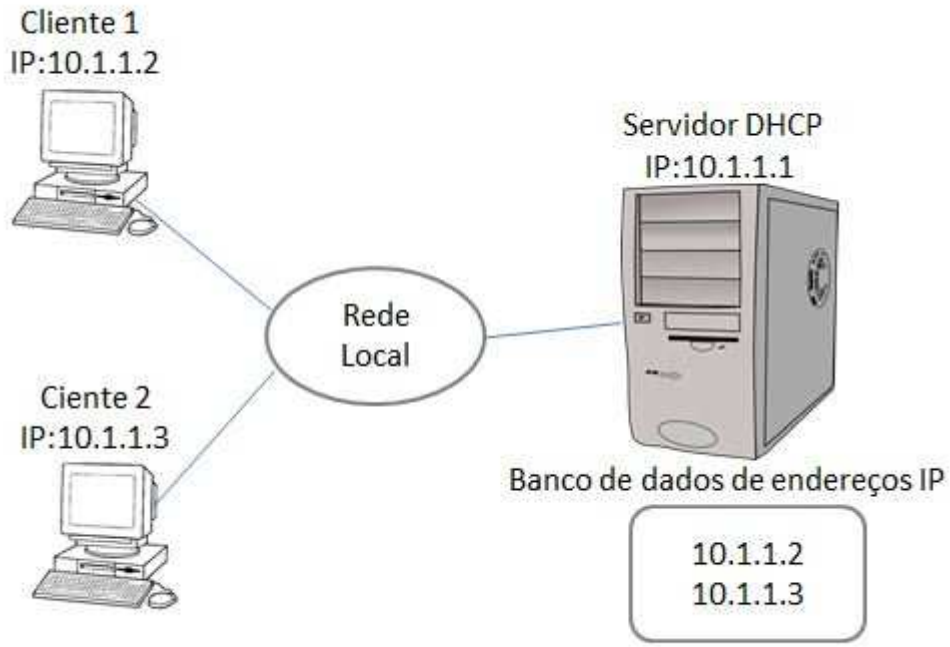
3.1. SERVIÇO DHCP

O DHCP fornece armazenamento persistente de paramentros da rede, mantendo informações de identificação para cada cliente que pode se conectar com a rede. Segundo os mesmos autores, os três pares mais comuns de informações de identificação são:

- Endereço de sub-rede/host da rede – Usado pelos hosts para se conectar com a rede sem restrições.
- Sub-rede/nome de host – Permite que o host especificado se conecte com a sub-rede.
- Endereço de sub-rede/Hardware – Permite que um cliente especifico se conecte com a rede, após obter o nome de host do DHCP.

Como pode-se observar na Figura 2 abaixo, os clientes são ligados à mesma rede do servidor DHCP, formando uma rede local. E a finalidade básica deste servidor DHCP é retirar de seu banco de dados de endereços IP e fornecer estes mesmos endereços IP aos comptudadores clientes da rede de maneira dinâmica, sem que haja a necessidade de configurar manualmente em cada computador da rede.

Figura 2: Servidor DHCP fornecendo IP aos computadores clientes.



Fonte: Próprio autor.

3.2. SERVIÇO DNS

O DNS é fundamental para muitos tipos de operações de rede, especialmente ao se fornecer conexão ao mundo exterior pela internet. O DNS foi projetado para tornar a atribuição e a transformação de nomes de host rápida e confiável, e para fornecer um espaço de nome consistente e portável para os recursos da rede.

Para Tanenbaum (2003) a essência do DNS é a criação de um esquema hierárquico de atribuição de nomes baseado no domínio e de um sistema de bancos de dados distribuídos para implementar esse esquema de nomenclatura. Ele é usado principalmente para mapear nomes de hosts e destinos de mensagens de correio eletrônico em endereços IP, mas também pode ser usado para outros objetivos. O DNS é definido nas RFCs 1034 e 1035.

3.3. SERVIÇO DE *PROXY*

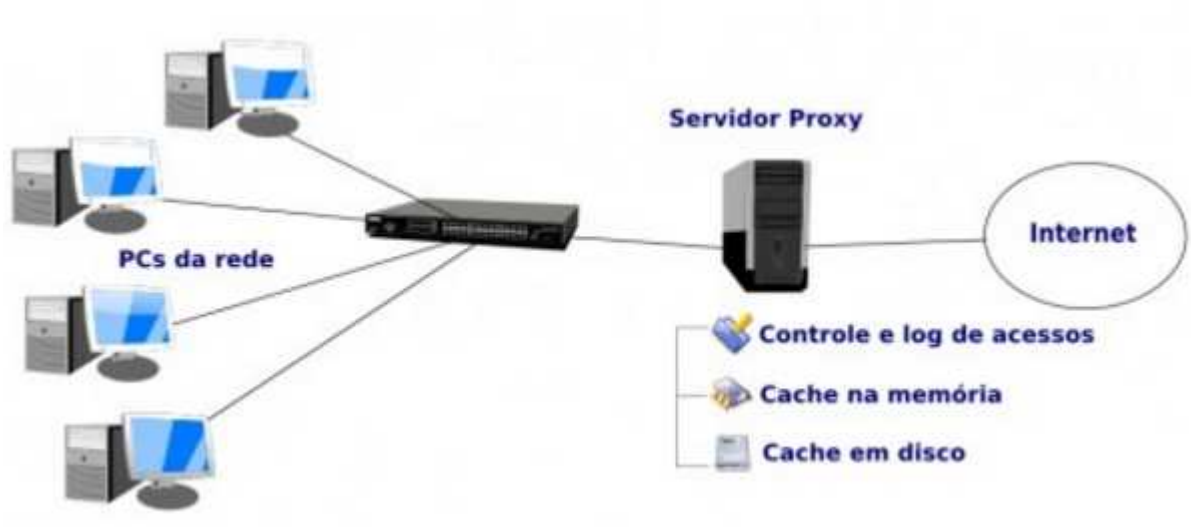
Nos ambientes organizacionais de hoje há uma grande dependência do uso da Internet e vários sistemas *online* que fazem o uso da mesma. Diante desta necessidade e preocupados sobretudo no que diz respeito a segurança da informação e uma melhor performance da rede dentro desse ambiente, as empresas começaram a fazer o uso dos serviços de *proxy*. Segundo Zanoni (2007) um servidor *proxy* é uma peça importante em uma rede interna que tenha contato com outra pública, pois implementa uma série de facilidades e controles.

Os serviços de um servidor *proxy* tem como finalidade servir de intermediário entre os vários computadores de uma rede privada e uma rede pública, como a Internet, afirma Zanoni (2007):

> O objetivo principal de um servidor *proxy* é possibilitar que máquinas de uma rede privada possam acessar uma rede pública, como a Internet, sem que para isto tenham uma ligação direta com esta. O servidor *proxy* costuma ser instalado em uma máquina que tenha acesso direto à internet, sendo que as demais efetuam as solicitações através desta. Justamente por isto este tipo é chamado de *Proxy*, pois é um procurador, ou seja, sistema que faz solicitações em nome de outros.

Na Figura 3 Morimoto (2008) apresenta um esquema básico de funcionamento de um servidor *proxy*. Onde o *proxy* funciona como um *cache* de páginas e arquivos, armazenando informações já acessadas. Quando alguém acessa uma página que já foi carregada, o *proxy* envia os dados que guardou no *cache*, sem precisar acessar a mesma página repetidamente. Isso acaba economizando bastante banda, tornando o acesso mais rápido.

Figura 3: Esquema de funcionamento do servidor proxy.



Fonte: Morimoto (2008)

Logo, o *proxy* é um serviço instalado no computador (servidor) onde a Internet chega vindo diretamente de um provedor. Desta forma todas as solicitações que os micros pertencentes a essa rede fizerem serão recebidos por ele, e ele se encarregará de buscar a página solicitada no servidor de origem desta página, porém ele também trabalha com a configuração de *cache*, onde se é guardado por determinado tempo uma cópia desta página, e caso a página solicitada já esteja neste *cache* não será necessário a busca desta página em seu servidor de origem, fazendo com que o tempo de resposta seja mais rápido.

3.4. *FIREWALL*

Proxy muitas das vezes é confundido com *firewall*, e é importante lembrar que existe diferença entre ambos. Já foi visto a definição de *proxy*, sendo agora necessário a abordagem da definição de *firewall*.

Neto (2004) afirma que:

> ***Firewall*** é um programa que detém autonomia concedida pelo próprio sistema para pré-determinar e disciplinar todo o tipo de tráfego existente entre o mesmo e outros hosts/redes; salvo situações onde o *Firewall* é um componente de soluções denominado "*Firewall*-in-a-box", onde neste caso, trata-se não tão somente de um software e sim de um agrupamento de componentes incluindo

> software e hardware, ambos projetados sob medida para compor soluções de controle perante o tráfego de um host/rede.

Portanto, o *proxy* se enquadra dentro da expressão *Firewall-in-a-box* descrito por Neto (2004), sendo um software de gerenciamento de tráfego. No entanto, *firewall* também está incluso nesta mesma expressão segundo afirmação acima. Para esclarecer, o *firewall* tem como objetivo primário filtrar, predeterminar todo e qualquer tipo de tráfego existente em uma rede, já o *proxy* visa e tem como principal objetivo servir de intermediário entre uma rede privada e uma pública (a Internet por exemplo), e ainda melhorar a performance da rede provendo mais rapidez e agilidade ao acessar páginas da Internet que uma vez acessada ficam armazenadas no *cache* do servidor, para que quando solicitadas novamente sejam entregues imediatamente ao solicitante, não sendo necessário que seja feito uma nova solicitação externa.

3.5. PRINCIPAIS TIPOS DE *PROXY*

Tem-se basicamente três tipos de *proxy*. A seguir estão apresentadas a definição dos diferentes tipos de *proxy* e as vantagens de cada um deles:

3.5.1. *PROXY* TRANSPARENTE

Segundo Morimoto (2008) o *proxy* transparente permite configurar o *squid* e o *firewall* de forma que o servidor *proxy* fique escutando todas as conexões na porta 80. Mesmo que alguém tente desabilitar o *proxy* manualmente nas configurações do navegador, ele continuará sendo usado.

Vantagens do *proxy* transparente segundo Carvalho (2003) :

- ocultar a utilização do *proxy* aos usuários que acessam internet;
- forçar os usuários a utilizarem o *proxy*, mesmo que eles não queiram;
- implementar a política real de acesso a sites proibidos, evitando os usuários mais "espertos" de desabilitar a utilização do *proxy* na navegação.

3.5.2. *PROXY WEB*

Silva (2011) informa que o *proxy web* corresponde à "função" conhecida pela maioria das pessoas. Nesta configuração, o *proxy* tem a função de compartilhar a Internet com a rede local, receber as requisições feitas pelos clientes e buscar o que foi solicitado nos servidores *web*.

Vantagens do *proxy web* na visão do mesmo autor:

- Controle de acesso: o acesso à internet pode ser controlado com base no horário, endereço IP do cliente, login e sites com conteúdo indesejado;
- Cache de páginas: o *proxy* guarda informações das páginas acessadas. Quando alguém acessa um endereço, o servidor procura primeiro nos seus arquivos armazenados, caso já possua a página, não precisa buscá-la novamente. O que acaba tornando a navegação mais rápida e evitando acessos desnecessários à *web*;
- Relatórios de acesso: todos os logs de acesso são armazenados, o que permite que possam ser criados relatórios dos acessos realizados pelos clientes através do servidor.

3.5.3. *PROXY* REVERSO

Para Silva (2011) o *proxy* reverso é um servidor instalado entre a Internet e os servidores *web* internos de uma empresa. As requisições externas são direcionadas a um servidor interno por meio de um roteamento feito pelo *proxy* reverso. Dessa forma, ele é a única interface para as requisições externas.

Vantagens do *proxy* reverso na visão do mesmo autor:

- Segurança: como o *proxy* é a única interface externa da rede, ele "esconde" os demais servidores;
- Criptografia: a criptografia SSL pode ser delegada ao *proxy* ao invés dos servidores internos;

- Balanceamento de carga: o servidor pode distribuir a carga para vários servidores da rede;
- *Cache*: assim como o *web proxy*, o *proxy* reverso pode manter em *cache* o conteúdo estático das requisições realizadas, ajudando assim a diminuir a carga dos servidores *web*;
- Compressão: o *proxy* reverso pode tornar o acesso mais rápido através da compressão do conteúdo acessado.

Existem alguns *proxies* atualmente rodando em servidores *proxy*, como o ISA Server da Microsoft por exemplo, porém o escolhido para este trabalho é o *squid*, que segundo Marcelo (2006) é o *proxy* mais utilizado em provedores e empresas e que consegue, de maneira soberba, atender as mais diversas necessidades. Este será o tema do próximo capítulo.

4. *PROXY SQUID*

O *squid* é um *software open source*, ou seja, de código aberto, que foi fundado por autorga da NFS (NCR-9796082), que fechou uma pesquisa sobre tecnologias de armazenamento em *cache*. O financiamento IRcache acabou e alguns anos mais tarde o projeto *squid* continuou através de doações voluntárias e investimentos comerciais ocasionais. *squid* ([s.d.])1

Segundo Teotonio (2010) o *squid* é um servidor *proxy* que suporta HTTP, HTTPS, FTP e outros. Ele reduz a utilização da conexão e melhora os tempos de respostas fazendo *cache* de requisições frequentes de páginas *web* numa rede de computadores.

Para complementar o conceito do que é *squid*, Rassilan (2009) afirma que *squid* é um servidor *proxy* utilizado para gerenciar o acesso a Internet (rede externa), pois ele implementa um controle sobre o conteúdo que deve ou não ser acessado pelas máquinas clientes gerenciadas por este servidor.

O mesmo autor informa que *squid* trabalha com ACLs (Listas de Controle de Acesso) e através dessas listas de controle ele se torna uma poderosa ferramenta na administração de tráfego de conteúdo entre a rede interna e a externa. E são através dessas listas que conseguimos criar e definir o controle de acesso a Internet de forma simples e flexível, tornando o *squid* uma ferramenta precisa em seu objetivo, que é basicamente bloquear o acesso a determinados *sites*, que podem ser fornecidos dentro de um arquivo de texto, e os que não estiverem dentro deste arquivo o *squid* reconhece que são *sites* de livre acesso.

Abaixo a relação das ACLs mais comuns ainda segundo o mesmo autor:

- srcdomain - tipo indicado para verificar o domínio da máquina cliente. Os domínios serão obtidos por resolução reversa de IP, o que pode causar atrasos para a resposta da requisição. A definição do domínio deve ser feita da seguinte

[1] Disponivel em: <http://www.squid-cache.org/Intro/ >. Acesso em: 27 set. 2016.

forma: ".meudominio.com.br", não podendo ser esquecido o "." (ponto) no início;

- time - usado para especificar dias da semana e horários. Os dias da semana são definidos através de letras que os representam, e os horários através de intervalos na forma hora:minuto_inicio-hora:minuto_final. Os dias da semana são especificados assim: S - Sunday (Domingo), M - Monday (Segunda-feira), T - Tuesday (Terça-feira), W - Wednesday (Quarta-feira), H - Thursday (Quinta-feira), F - Friday (Sexta-feira) e A - Saturday (Sábado);
- src - tipo utilizado para indicar endereços IP de origem. Pode-se especificar um endereço de rede, como 192.168.16.0/24, um endereço de um determinado host, como 192.168.16.10/24 ou uma faixa de endereços, como 192.168.16.10-192.168.16.20/24;
- dst - semelhante ao tipo anterior, mas está relacionada ao endereço de destino;
- dstdomain - usado da mesma forma que srcdomain, entretanto com relação ao destino;
- srcdom_regex - avalia o domínio usando expressões regulares. Seu uso é semelhante às duas anteriores, acrescentando a flexibilidade do uso da expressão regular;
- dstdom_regex - usado da mesma forma que srcdom_regex, entretanto com relação ao destino;
- url_regex - este tipo percorre a URL à procura da expressão regular especificada. Deve ser observado que a expressão é case-sensitive, para que seja case-sensitive deve ser usada a opção -i. É o tipo mais comum de ACL, dada a flexibilidade proporcionada pelo uso de expressões regulares;
- urlpath_regex - tipo semelhante à url_regex, mas procura a expressão regular na URL sem levar em conta o nome do servidor e o protocolo, isto quer dizer que a procura será feita apenas na parte da URL após o nome do servidor, como por exemplo, na URL http://www.servidor.com.br/pasta/sexo.html, a procura será realizada apenas na parte /pasta/sexo.html. Ela é também case-sensitive, para que seja case-insensitive deve ser usada a opção -i;

- port - realiza o controle pela porta de destino do servidor, neste tipo deve ser especificado o número da porta;
- proto - serve para especificar o protocolo, como por exemplo FTP ou HTTP;
- method - especifica o tipo de método usado na requisição, como por exemplo GET, CONNECT ou POST;
- browser - usa uma expressão regular para tentar "casar" com os dados do cabeçalho HTTP e combinando então com o navegador utilizado pelo cliente;
- ident - Realiza o controle de acesso baseado no nome do usuário. Este tipo requer um servidor Ident rodando na máquina do cliente;
- ident_regex - semelhante a ident, mas utilizando expressão regular;
- *proxy*_auth - tipo usado para implementar autenticação de usuários no *proxy*. A autenticação é feita com uso de softwares externos. Podem ser passados os nomes dos usuários ou usada a opção REQUIRED para que seja autenticado qualquer usuário válido;
- snmp_community - tipo usado para especificar o nome da comunidade SNMP para que se possa monitorar o *squid* através deste protocolo;
- maxconn - especifica um limite de conexões vindas de um determinado cliente, interessante para uso com outras ACLs de forma a limitar quantidades de conexões para determinados endereços específicos;
- req_mime_type - especifica uma expressão regular para ser verificada no cabeçalho da requisição em busca de um tipo MIME que coincida com o especificado;
- arp - tipo usado para construir lista de acesso baseada no MAC Address da interface de rede do cliente, ou seja, em vez de endereço IP da placa, usa-se o seu endereço MAC.

Nas figuras a seguir, são mostradas as versões do *squid*, desde as mais antigas (Figura 4) até a atual (Figura 5), e até mesmo a versão beta (Figura 6) já disponilizada para quem queira instalar e efetuar testes.

Figura 4: Versões antigas do squid.

versão	First Date versão estável	Último lançamento	Última data de lançamento
3.4	09 de dezembro de 2013	3.4.14	01 de agosto de 2015
3.3	09 de fevereiro de 2013	3.3.14	01 de maio de 2015
3.2	14 de agosto de 2012	3.2.14	01 de maio de 2015
3.1	29 de março de 2010	3.1.23	09 de janeiro de 2013
3.0	13 de dezembro de 2007	STABLE26	28 de agosto de 2011
2.7	31 de maio de 2008	STABLE9	16 de março de 2010
2.6	01 de julho de 2006	STABLE23	17 de setembro de 2009
2.5	25 de setembro de 2002	STABLE14	20 de maio de 2006
2.4	20 de março de 2001	STABLE7	02 de julho de 2002

Fonte: *Squid* ([s.d.])[2].

As versões do *squid* parte da versão 2.4, elaborada no ano de 2001..

Figura 5: Versão atual adequado para uso.

versão	Último lançamento	
3.5	08 de setembro de 2016	3.5.21
langpack	17 de agosto de 2016	ROLLING

Fonte: *Squid* ([s.d.])[3].

A versão 3.5 é a versão mais atual e adequada para implementação do *squid* segundo seus desenvolvedores.

Figura 6: Versão beta para testes

versão	Último lançamento	
4.0	08 de setembro de 2016	4.0.14

Fonte: *Squid* ([s.d.])[4].

[2] Disponivel em: <http://www.squid-cache.org/Versions/>. Acesso em: 23 set. 2016.
[3] Disponivel em: <http://www.squid-cache.org/Versions/>. Acesso em: 23 set. 2016.
[4] Disponivel em: <http://www.squid-cache.org/Versions/>. Acesso em: 23 set. 2016.

Os desenvolvedores deixam claro que na versão beta pode ser encontrados vários bugs e problemas, não sendo recomendados que o implementem em ambientes que desejam serem estáveis e seguros.

4.1. REQUISITOS

Segundo a Documentação do SUSE LINUX ([s.d.]) dentre os fatores de maior importância é a determinação máxima de carga de rede que o sistema deve suportar. Salienta ainda que é importante prestar mais atenção aos picos de carga, porque esses picos podem ser superiores a quatro vezes a média dos picos ocorridos em um dia. Frisa também que em dúvida, seria melhor superestimar os requisitos do sistema, isso porque se o *squid* trabalhar no limite de sua capacidade pode haver uma perda grave de qualidade do serviço entregue. Os itens a seguir foram restirados da documentação SUSE LINUX e mostram os fatores do sistema e hardware interessante para a implementação do *squid* de acordo com a documentação especificada:

- **Discos Rígidos**

A documentação informa que a velocidade exerce uma importante função no processo de *cache*, de tal forma que este fator requer uma atenção especial.

> Para discos rígidos, este parâmetro é descrito como tempo de busca aleatório, medido em milissegundos. Como os blocos de dados que o *squid* lê no disco rígido ou grava dele tendem a ser pequenos, o tempo de busca do disco rígido é mais importante que seus throughput de dados. Para fins de um *proxy*, os discos rígidos com velocidades de alta rotação são provavelmente a melhor opção, porque permitem que a cabeça de leitura-gravação seja posicionada no ponto desejado mais rapidamente. Uma possibilidade para acelerar o sistema é usar vários discos simultaneamente ou empregar matrizes de RAID de distribuição.

- **Tamanho do cache do disco**

Para o tamanho do *cache* é importante que deixe um espaço de armazenamento não muito pequeno, pois se for muito pequeno o *cache* será facilmente arquivado, uma vez que os objetos mais antigos serão sobrescritos e substituídos pelos objetos mais novos. A documentação cita como exemplo, um *cache* de 1 GB de armazenamento. Quando esta quantidade de armazenamento estiver disponível para

o *cache* e os usuários só navegarem 10 MB por dia, levaria mais de 100 dias para preencher o cache, desta forma a probabilidade do objeto solicitado ser encontrado no cache seria muito alta.

> A maneira mais fácil de determinar o tamanho de cache necessário é considerar a taxa máxima de transferência da conexão. Com uma conexão de 1 Mbit/s, a taxa máxima de transferência é de 125 KB/s. Se todo este tráfego terminar no cache, em uma hora teria adicionado 450 MB e, considerando-se que todo esse tráfego seja gerado em apenas oito horas de trabalho, alcançaria 3,6 GB em um dia. Como a conexão normalmente não é usada até o seu limite superior de volume, pode-se presumir que o volume de dados total tratado pelo cache seja aproximadamente de 2 GB. Por esta razão é necessário um espaço em disco de 2 GB no exemplo, para que o *squid* guarde os dados interessantes pesquisados no cache.

- **Memória RAM**

Para a documentação do SUSE LINUX, a quantidade de memória RAM de *cache* deve ser superior a 4 GB, pois:

> O *squid* também armazena referências de objeto e objetos solicitados freqüentemente na memória principal do cache para acelerar a recuperação destes dados. A memória de acesso aleatório é muito mais rápida do que um disco rígido.

Informa ainda que existe outros dados que o *squid* necessita manter na memória, são eles: tabela com todos os endereços IP tratados, *cache* de nome de domínio exato, os objetos solicitados com mais freqüência, listas de controle de acesso, buffers, etc. Por isso a importância de se ter memória RAM suficiente para um melhor desempenho do sistema. E o que a documentação recomenda é memória RAM superior a 4 GB em um ambiente com grande tráfego de rede.

- **Processador (CPU)**

A documentação informa que o *squid* não é um programa que requer ou necessite de um itenso uso de CPU, pois o processador só terá mais exigência no momento de armazenar e verificar os conteúdos de *cache*. Ressalta que para uma melhor eficiência é necessário que se dê prioridade para a aquisição de discos mais rápidos e mais memória RAM.

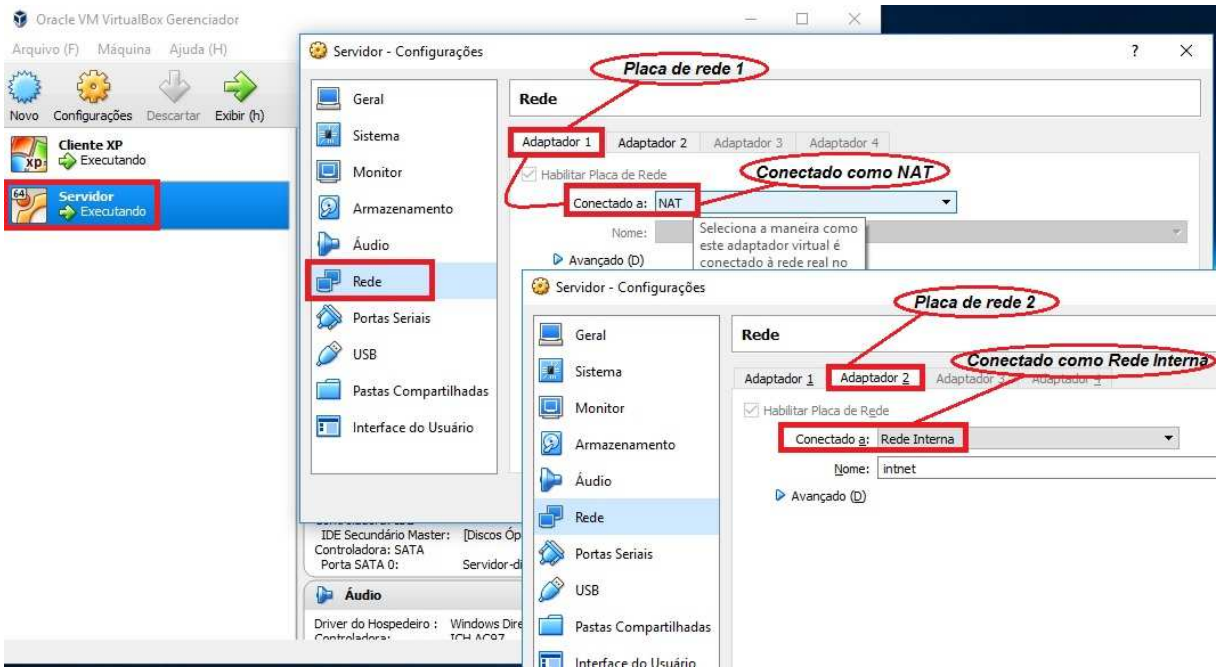
5. EXPERIMENTO DE IMPLEMENTAÇÃO DO *PROXY SQUID*

Este experimento tem como objetivo demonstrar o quão possível é a implementação do servidor *proxy squid* em um ambiente organizacional, tendo como principal aliado o fato de ele ser totalmente gratuito e muito eficiente em suas tarefas. Para demonstrar sua configuração e funcionamento será utilizado um ambiente virtual composto por três computadores virtuais, sendo um o servidor com um sistema operacional Ubuntu Server versão 16.04.1 onde o *squid* será configurado, e os outros dois com sistema operacional Windows XP versão 5.1.2600, que servirão como computadores clientes utilizados principalmente para demonstrar bloqueios a *sites* da Internet. O ambiente virtual utilizado para implementar esta demonstração será o Oracle VM VirtualBox versão 5.1.6.

Para a elaboração e implementação deste experimento foi feito pesquisas em alguns artigos com a finalidade de adquirir o conhecimento necessário para que o resultado final fosse satisfatório do ponto de vista que todas as configurações necessárias fossem aplicadas nesse ambiente virtual. Será mostrado em seguida como se deu a implementação e configuração do ambiente virtual segundo (ZANONI, 2007; TEOTONIO, 2010; RASSILAN, 2009).

Primeiramente será feita a instalação do sistema operacional Ubuntu Server versão 16.04.1 no VirtualBox, porém, esta experimento parte do princípio de que os computadores virtuais (servidor e clientes) já estejam instalados, restando desta forma fazer as demais configurações necessárias para que o ambiente esteja pronto para demonstrar o funcionamento do *proxy squid*. Para o servidor serão configuradas duas placas de rede, a primeira será configurada como NAT e a segunda como Rede Interna, como ilustrado na imagem abaixo:

Figura 7: Configuração das interfaces de rede no servidor *squid*.

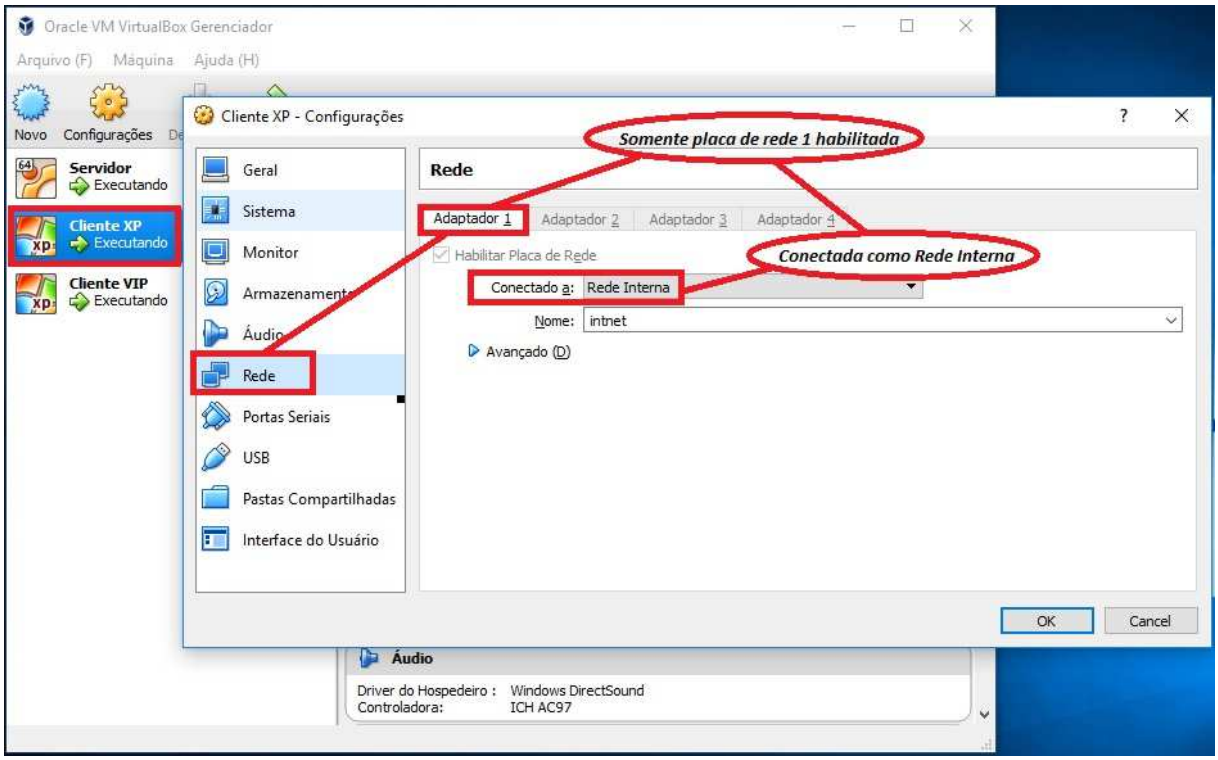


Fonte: Próprio autor.

Configuração do servidor:
Memória RAM: 512 MB.
Memória ROM: 8 GB.
Sistema Operacional: Ubuntu Server 16.04.1.

Além da configuração e instalação do Ubuntu Server, será feita a instalação e configuração do cliente, onde serão efetuado os testes de bloqueio a *sites* da Internet. Para o cliente será configurado no virtualBox dois computadores com sistema operacional Windows XP versão 5.1.2600, com a placa de rede configurada como rede interna como mostra a Figura 8 abaixo:

Figura 8: Configuração da interface de rede nos clientes.



Fonte: Próprio autor.

Configuração dos computadores clientes:
Memória RAM: 256 MB.
Memória ROM: 8 GB.
Sistema Operacional: Windows XP versão 5.1.2600.

Uma vez instalado e configurado as duas interfaces de rede do Ubuntu Server como descrito, é necessário fazer as configurações do sistema. A configuração do sistema consiste em instalar os pacotes e serviços necessários para que o *squid* funcione. Será utilizado o símbolo “#” com o comando a ser utilizado na frente para simbolizar que é um comando. Primeiramente deve-se logar com usuário root no servidor, usuário este que tem os privilégios para fazer as alterações necessárias no sistema. Para isso digite:

 # sudo su

Logo após, forneça a senha para se autenticar como usuário root.

Uma vez logado com usuário root, pode-se começar a instalação dos pacotes, agora digite:

 # apt-get update

Logo em seguida:

 # apt-get upgrade

Esses dois comandos serve para fazer a instalação de pacotes de atualização do sistema operacional.

Após a atualização do sistema operacional já se consegue instalar o *squid*, digite:

 # apt-get install squid

Após o término da instalação do *squid*, será criado no diretório /etc/squid o arquivo de configuração squid.conf, onde toda sua configuração ficará. O arquivo squid.conf é considerado o coração do *squid*, ele vem todo comentado, esses comentários serve para explicar o seu funcionamento. Porém o arquivo já vem funcional com suas configurações padrão, e parte do princípio de bloquear a todos os *sites* da Internet aos micros da rede. Porém, já tem indicado no aquivo o espaço para a configuração das ACLs, onde o administrador é livre para fazer as configurações que desejar.

Antes de fazer as configurações das ACLs, pode se instalar e configurar o servidor DHCP. Neste caso vai ser utilizado o mesmo servidor onde o *squid* está instalado. O servidor DHCP irá fornecer de maneira dinâmica IP aos computadores clientes da rede virtual criada, para que não haja a necessidade de configurar o IP manualmente em cada computador da rede. Será configurado no DHCP uma reserva de IP para que este IP fique atrelado ao computador desejado, e nemhum outro computador da rede possa recebe-lo.

Para instalar o serviço de DHCP, com usuário root digite no terminal o comado abaixo:

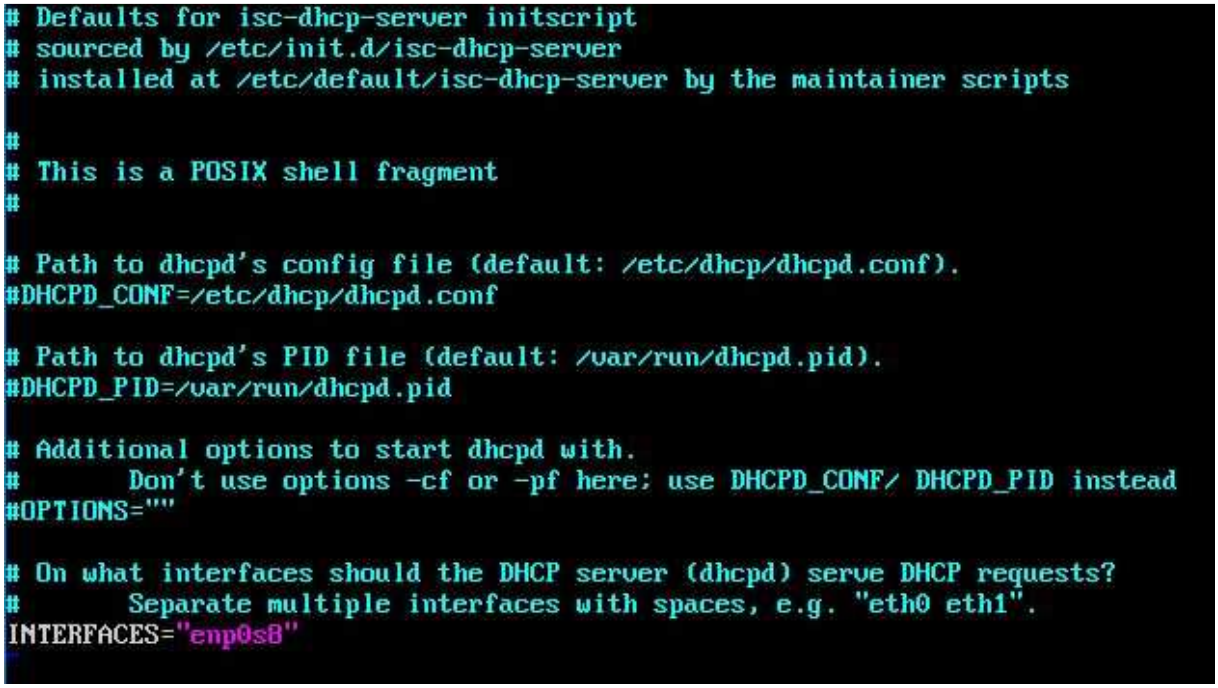
 # apt-get instal isc-dhcp-server

Vá até o diretório /etc/dhcp/ e edite o arquivo dhcpd.conf para que fique desta forma:

```
ddns-update-style none;
default-lease-time 600;
max-lease-time 7200;
log-facility local17;
subnet 10.1.1.0 netmask 255.255.255.0 {
range 10.1.1.2 10.1.1.102; # Faixa de IP a ser distribuída pelo DHCP.
option subnet-mask 255.255.255.0; # Mascara de subrede.
option routers 10.1.1.1; # IP da segunda interface de rede.
option broadcast-address 10.1.1.255; # Endereço de broadcast.
# Fixar o IP 10.1.1.10 ao computador de nome cliente (Computador da diretoria que
# terá todos os sites da Internet liberado).
host cliente {
hardware ethernet 08:00:27:45:50:33: #macaddress do computador de nome cliente
fixed-address 10.1.1.10; #IP a ser fixado para o macaddress especificado
Option host-name “cliente”;
}
}
```

Logo depois vá ao diretório /etc/default/ e edite o arquivo isc-dhcp-server e edite a última linha para que fique igual a Figura 9:

Figura 9: Arquivo de configuração /etc/default/isc-dhcp-server.



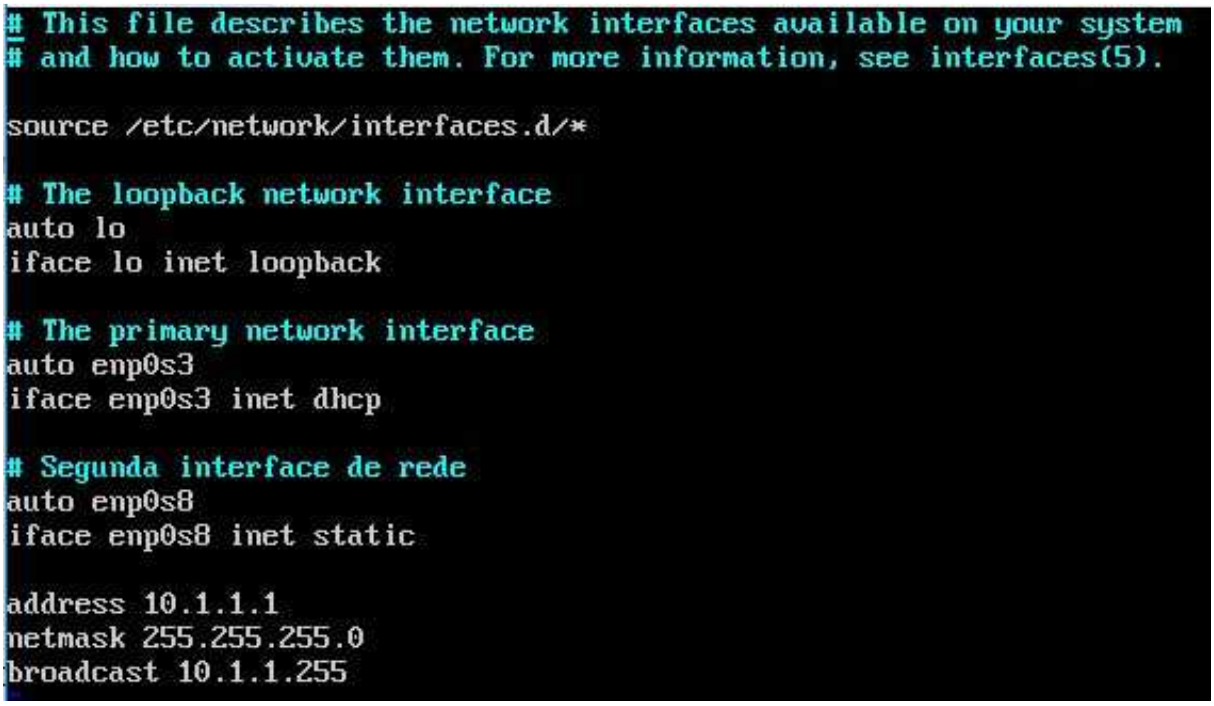
Fonte: Próprio autor.

INTERFACES=”enp0s8”

Nas versões anteriores o nome dado a segunda interface de rede era eth1 e a primeira era eth0, porém nas novas versões mudou-se o nome, e nesse caso em específico o nome da segunda interface é enp0s8, e a primeira enp0s3, porém o resultado final será o mesmo.

Logo depois vá ao diretório /etc/network e edite o arquivo interfaces para que fique da forma que está na imagem abaixo:

Figura 10: Arquivo de configuração /etc/network/interface.



Fonte: Próprio autor.

Isso indica que a segunda interface de rede está sendo configurada com IP estático e, determina também qual será a máscara de sub-rede a ser utilizada e ainda qual é o endereço de broadcast.

É necessário também habilitar o encaminhamento de pacotes IPv4, para isso se faz necessário ir ao diretório /etc/ e editar o arquivo sysctl.conf, apenas descomentando a linha abaixo.

Net.ipv4.ip_forward=1

Após salvar e sair do arquivo digite o comando abaixo para validar as mudanças feitas no arquivo:

 # Sysctl –w net.ipv4.ip_forward-1

O *squid* opera por padrão na porta 3128, por isso se faz necessário fazer o redirecionamento do tráfego da porta 80 para a porta onde o *squid* opera, a 3128. Para fazer esse direcionamento utiliza-se a feramenta *iptables* para criar a regra de redirecionamento. A regra é feita da seguinte forma na linha de comando:

 # iptables –t nat –A PREROUTING –i epn0s8 –p tcp –dport 80 –j REDIRECT –to-port 3128

No arquivo de configuração squid.conf serão inseridas apenas as regras de liberação de acesso e bloqueios a *sites* da Internet, já que as outras configurações variam de acordo a necessidade da rede onde o *squid* está sendo implementado. Como falado anteriormente o arquivo squid.conf vem com milhares de linha, quase todas são comentários sobre o *squid.* Para ficar visualmente mais claro pode se descomentar todo o arquivo deixando o arquivo limpo com suas configurações padrão. Para descomentar todo o arquivo digite o commando abaixo no terminal:

 # egrep “^#|^$” /etc/squid/squid.conf > /tmp/squid.conf

Isso irá criar um novo arquivo squid.conf no diretório /tmp/ todo descomentado.

Logo depois é necessário que faça um arquivo backup do squid.conf, fazendo uma cópia do arquivo original. Para essa operação digite o commando abaixo no terminal:

 # cp /etc/squid/squid.conf /etc/squid/squid.conf.backup

Este comando irá copiar um arquivo igual ao original (squid.conf) com o nome squid.conf.backup no mesmo diretório (/etc/squid/).

Logo depois exclua o arquivo squid.conf:

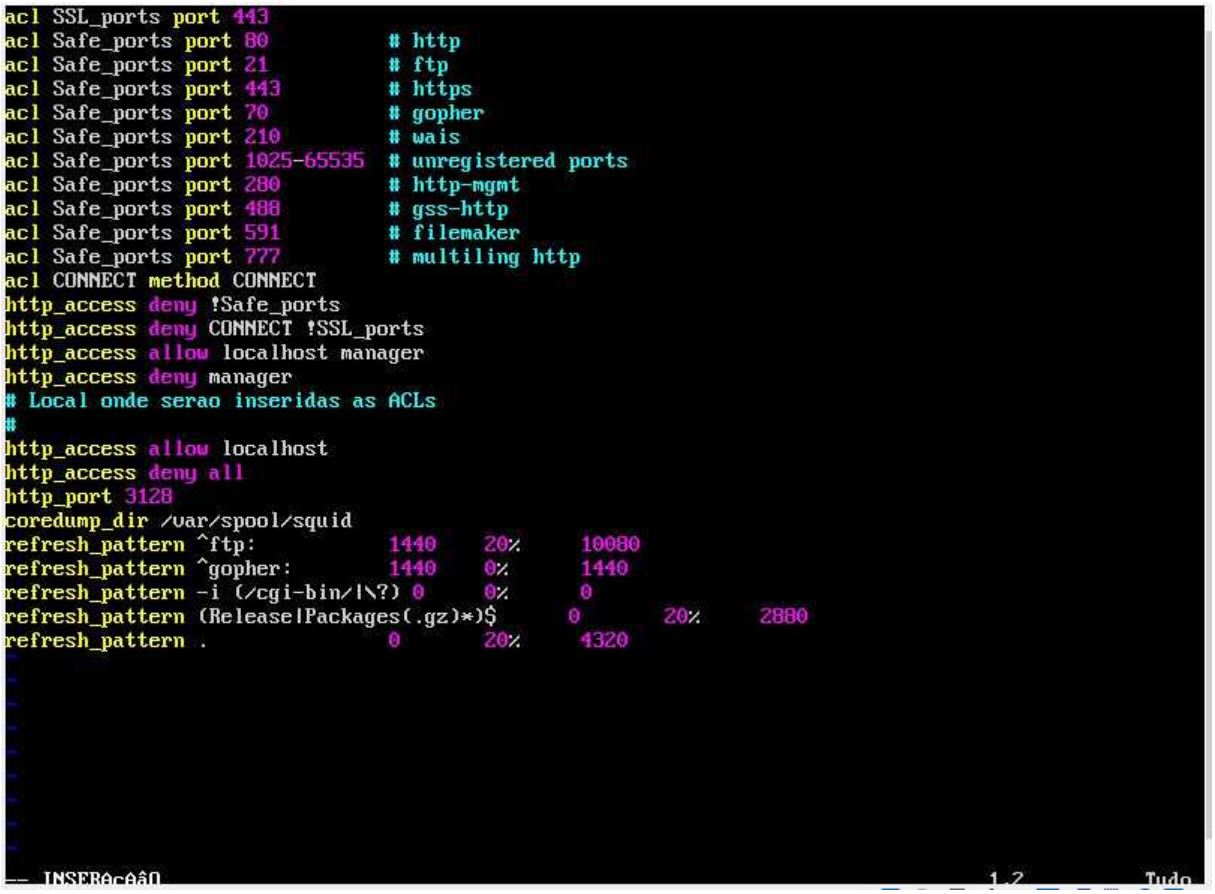
 # rm squid.conf

Em seguida já pode copiar o arquivo limpo, sem comentários que foi criado no diretório /tmp/. Faça isso com o comando abaixo:

 # cp /tmp/squid.conf /etc/squid/squid.conf

Agora o arquivo está limpo e pronto para receber as configurações adicionais desejadas pelo administrador do sistema. Logo abaixo temos a Figura do arquivo limpo e com um comentário de onde serão inseridas as ACLs que serão criadas para demonstrar os bloqueios.

Figura 11: Arquivo de configuração /etc/squid/squid.conf limpo.



Fonte: Próprio autor.

Agora serão incluídas as ACLs e as regras no espaço informado na imagem acima. Foram organizadas da forma como a Figura 12 demonstra:

Figura 12: Criação das ACLs e regras.

```
# ACLs criadas pelo Administrador do sitema
acl rede_local src 10.1.1.0/24
acl diretoria src 10.1.1.10
acl palavras_bloqueadas url_regex -i "/etc/squid/palavras_bloqueadas.txt"
acl sites_bloqueados url_regex -i "/etc/squid/sites_bloqueados.txt"
acl redes_sociais url_regex -i "/etc/squid/redes_sociais.txt"
acl horario_almoco time 12:00-13:00
acl dominio_bloqueado dstdomain .youtube.com.br

http_access allow diretoria
http_access allow redes_sociais horario_almoco
http_access deny redes sociais
http_access deny dominio_bloqueado
http_access deny sites_bloqueados
http_access deny palavras_bloqueadas
http_access allow rede_local
#
```

Fonte: Próprio autor.

A seguir tem-se o arquivo de configuração com os comentários sobre cada ACL e regra criada. Foram aplicadas algumas dos vários tipos de ACLs que foram abordadas no capitulo 4:

Cria a ACL de nome rede_local do tipo src.

acl rede_local src 10.1.1.0/24

Cria a ACL de nome diretoria do tipo src.

acl diretoria src 10.1.1.10

Cria a ACL de nome palavras bloqueadas do tipo url_regex.

acl palavras_bloqueadas url_regex –i “/etc/squid/palavras_bloqueadas.txt”

Cria a ACL de nome sites_bloqueados do tipo url_regex.

acl sites_bloqueados url_regex –i “/etc/squid/sites_bloqueados.txt”

Cria a ACL de nome redes_sociais do tipo url_regex.

acl redes_sociais url_regex –i “/etc/squid/redes_sociais.txt”

Cria a ACL de nome horario_almoco do tipo time.

acl horário_almoco time 12:00-13:00

Cria a ACL de nome domínio_bloqueado do tipo dstdomain

acl domínio_bloqueado dstdomain .youtube.com.br

Regra que desbloqueia tudo para o IP 10.1.1.10 da diretoria.
http_access allow diretoria
Regra que desbloqueia as redes sociais no horário de almoço.
http_access allow redes_sociais horário_almoco
Regra que bloqueia as redes sociais.
http_access deny redes_sociais
Regra que bloqueia o domínio .youtube.com.br
http_access deny domínio_bloqueado
Regra que bloqueia a lista de sites contidas em sites_bloqueados.txt.
http_access deny sites_bloqueados
Regra que bloqueia a lista de palavras contidas em palavras_bloqueados.txt.
http_access deny palavras_bloqueadas
Regra que desbloqueia para os computadores da rede tudo o que ainda não foi bloqueado.
http_access allow rede_local

É necessário criar os arquivos redes_sociais.txt, sites_bloqueados.txt e palavras_bloqueadas no diretório /etc/squid/. Dentro de cada arquivo coloca-se as palavras e url de *sites* que desejarem. No arquivo redes_sociais.txt por exemplo, pode colocar as palavras facebook, twitter, instagram, whatsapp, enfim, em cada arquivo será colocado as palavras que for necessário para efetuar o bloqueio.

A primeira regra tem a finalidade de liberar todos os *sites* da Internet para o computador do diretor, de IP 10.1.1.10. Este IP foi reservado no DHCP para que apenas o computador do diretor o receba, não havendo risco de que um outro computador da rede receba este IP e tenha acesso a todos os *sites* da Internet ou crie qualquer tipo de conflito na rede.

A segunda regra libera os itens contidos no arquivo redes_sociais.txt no horário de almoço, para que todos usuários possam navegar nestes tipos de *sites*.

A terceira regra bloqueia definitivamente os itens contidos no arquivo redes_sociais.txt, para que os usuário só utilizem no horário de almoço.

A quarta regra bloqueia o domínio .youtube.com.br para todos os usuários da rede, exceto para o diretor que tem todos os sites liberados.

A quinta regra bloqueia todos os sites contidos no arquivo sites_bloqueados.txt.

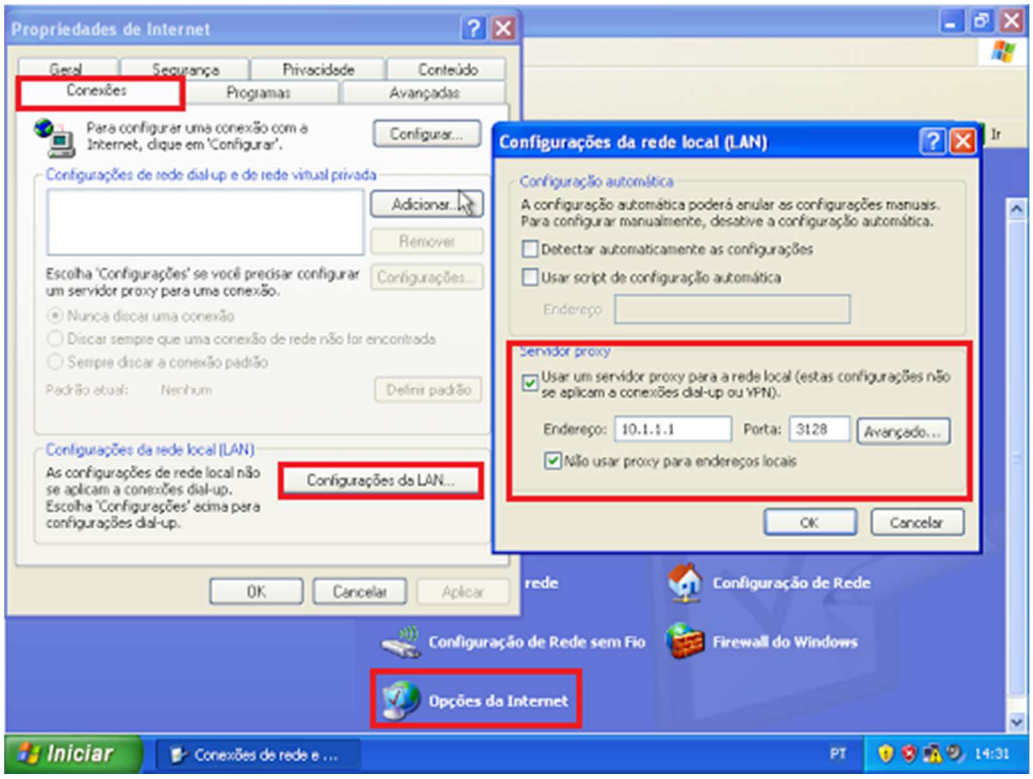
A sexta regra bloqueia todos os sites que contenha na url alguma das palavras contidas no arquivo palavras_bloqueadas.txt.

Por fim a última regra criada libera todos os demais sites que não foram bloqueados nas regras anteriores para todos os usuários da rede.

É importante salientar ainda que nas regras padrão do *squid* tem a regra “http_access deny all”. Esta regra bloqueia todos os computadores que estejam fora da rede de acessarem qualquer tipo de *site*.

Após todas essas configurações finalizadas é só configurar o *proxy* nos computadores clientes para que os testes possam ser efetuados. É importante lembrar que, como falado no capítulo 3, seção 3.4, pode se configurar o *proxy squid* para que ele possa funcionar de maneira transparente, sem a necessidade de configura-lo nos computadores clientes, ou então configura-lo do tipo autenticado, onde o usuário deve inserir um usuário e senha para se autenticar ao abrir o *browser*. No entanto, nesse caso será configurado manualmente o *proxy* nos computadores clientes para que sejam feito os testes. A configuração do *proxy* nos computadores clientes é feita como na figura abaixo, clicando com o *mouse* no botão Iniciar>Painel de controle>conexões de rede e de internet>opções da internet.

Figura 13: Configuração do *proxy* nos computadores clientes.



Fonte: Próprio autor.

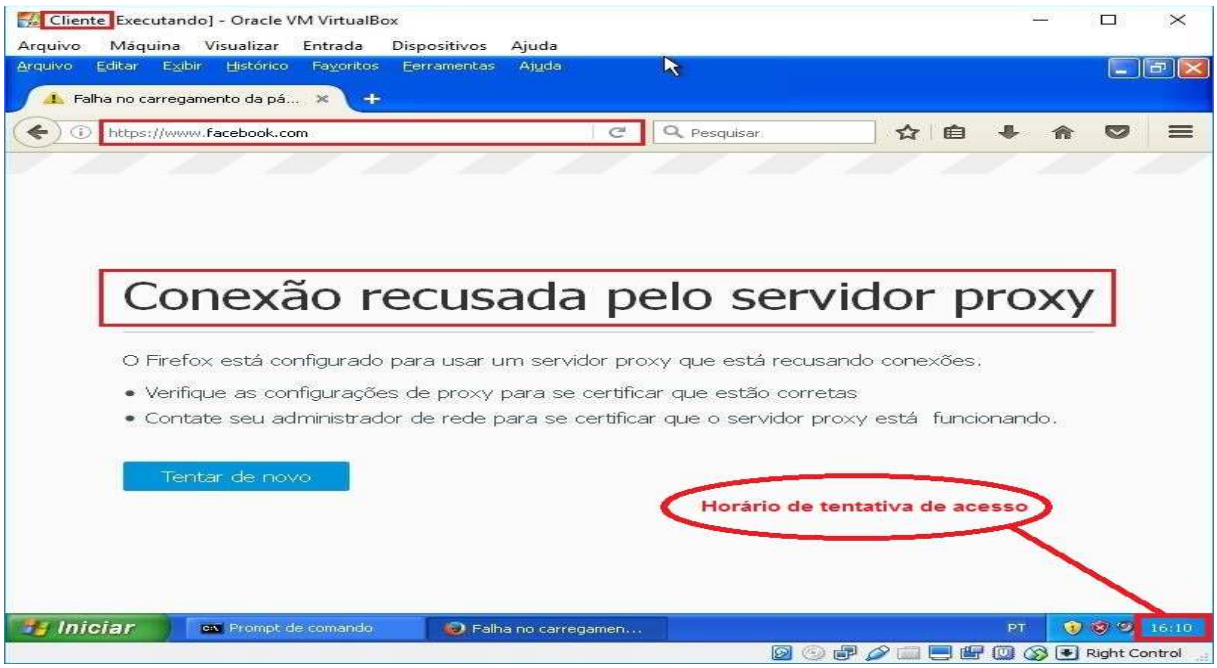
No campo reservado para inserir o endereço, é necessário informar o IP do servidor *proxy squid* (10.1.1.1), e no campo reservado para inserir a porta, é necessário inserir o número da porta onde opera o *proxy squid* (3128).

A seguir será mostrado uma série de figuras mostrando os bloqueios a *sites* da Internet de acordo com as ACLs e regras criadas e explicadas anteriormente. Primeiramente será mostrado na Figura 14 o computador de nome Cliente VIP (computador da diretoria) que segundo a primeira regra criada tem acesso liberado à todos os *sites* da Internet, acessando o *site* de uma rede social (www.facebook.com). E logo em seguida a Figura 15 mostra um cliente normal que tem o acesso às redes sociais liberado apenas no horário de almoço (12:00 as 13:00) tentando o acesso ao mesmo *site* (www.facebook.com) fora do horário de almoço (as 16:10) e sendo bloqueado pelo *proxy.*

Figura 14: Computador da diretoria acessando rede social.

Fonte: Próprio autor.

Figura 15: Computador cliente com conexão a rede social bloqueado pelo *proxy*.

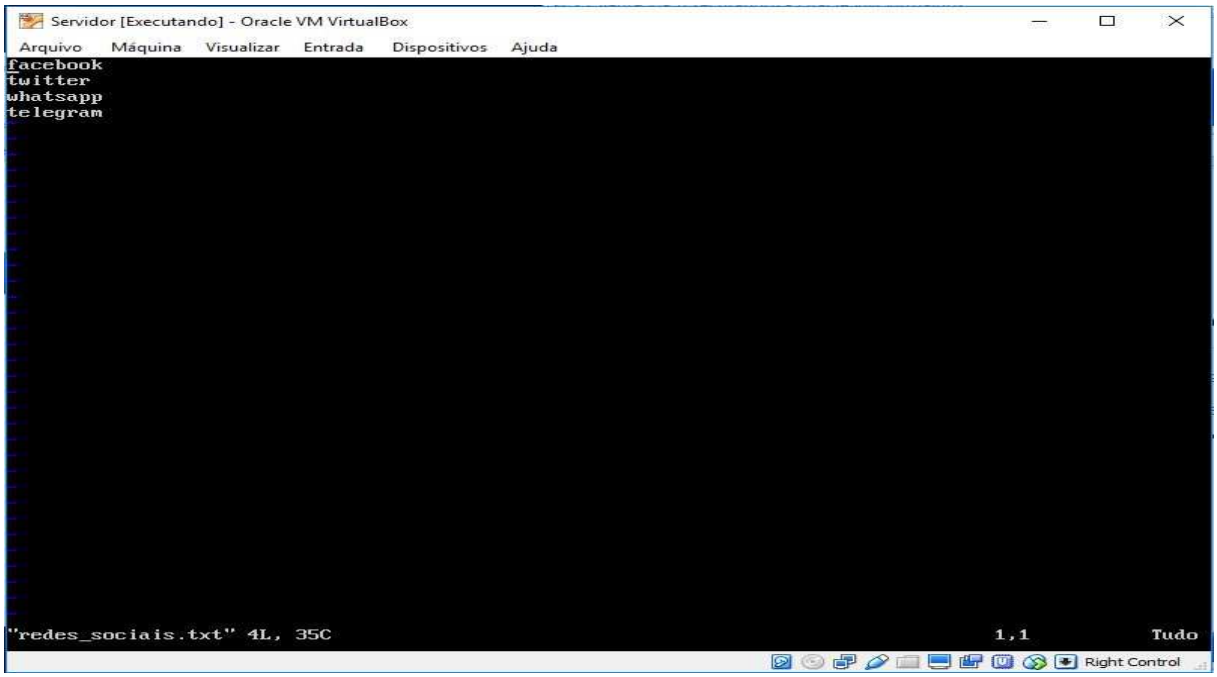


Fonte: Próprio autor.

A Figura 16 mostra o arquivo de configuração de nome redes_sociais.txt que determina quais as redes sociais que serão liberadas para os usuários acessarem no

horário de almoço e as quais serão bloqueadas fora desse horário, que é o caso da Figura 15 acima.

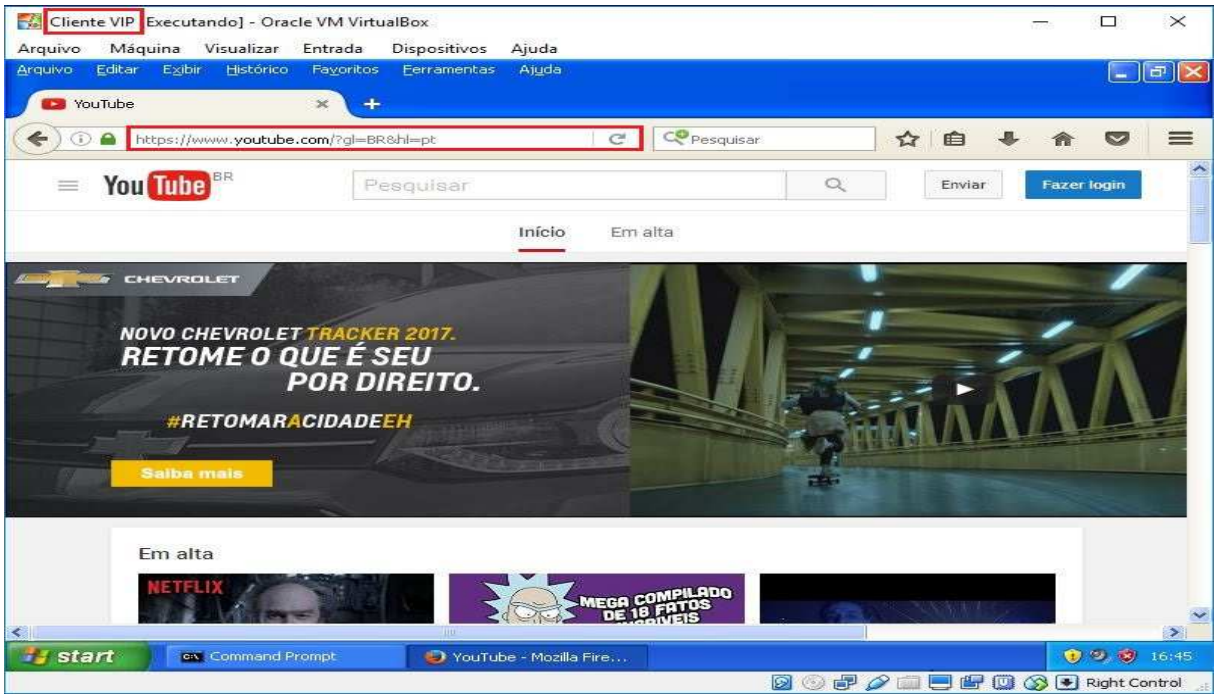
Figura 16: Arquivo de configuração redes_sociais.txt



Fonte: Próprio autor.

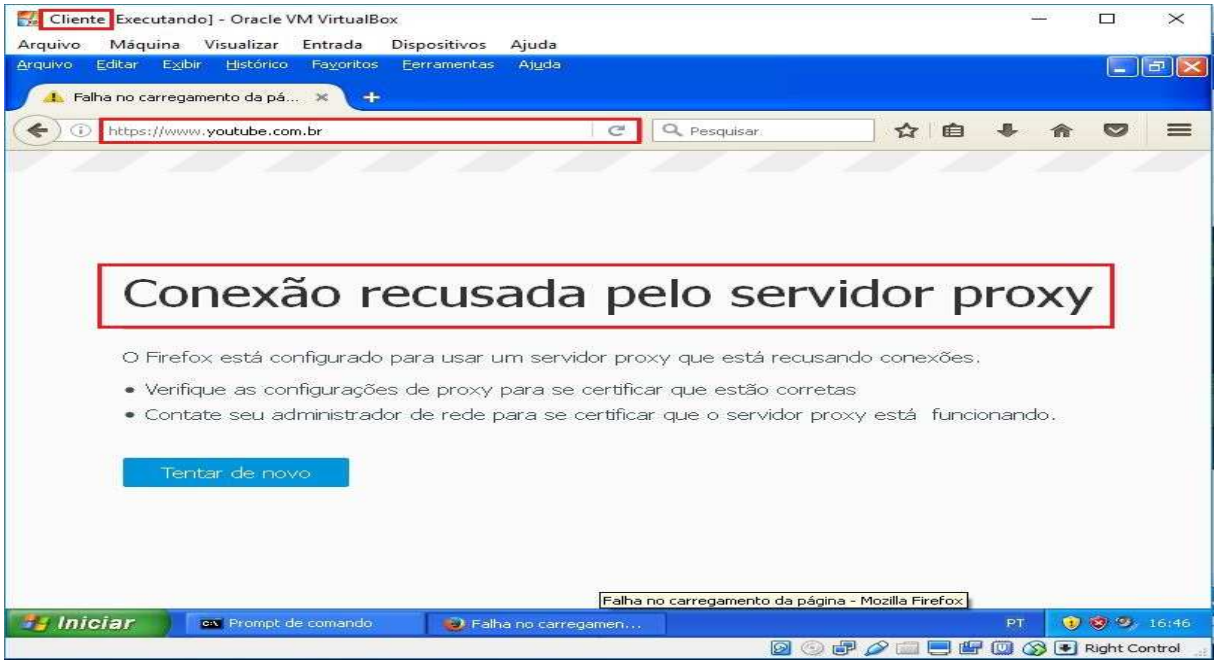
Já nas figuras a seguir tem-se o resultado da regra criada para bloquear o domínio .youtube.com.br para todos os computadores da rede, exceto para o computador da diretoria que tem acesso irrestrito e consegue o acesso a esse domínio como mostra a figura 17, diferente dos demais computadores da rede que tem acesso negado a esse domínio como mostra a Figura 18.

Figura 17: Domínio .youtube.com.br liberado para diretoria.



Fonte: Próprio autor.

Figura 18: Domínio .youtube.com.br bloqueado para computador da rede.

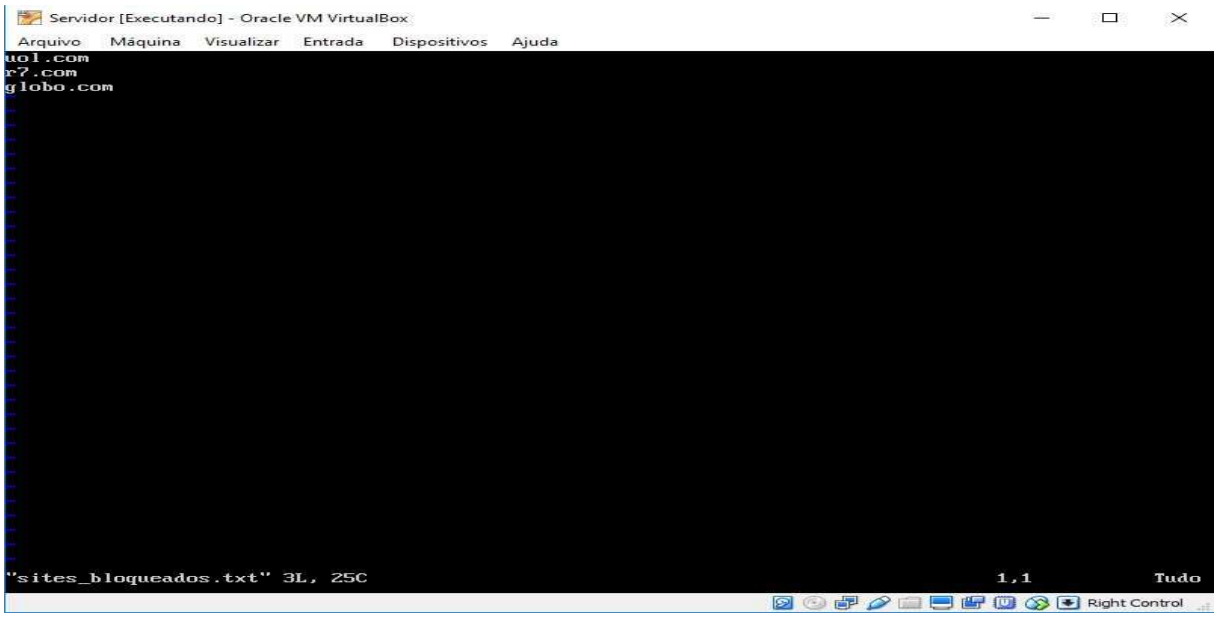


Fonte: Próprio autor.

As próximas figuras irão mostrar o resultado da regra de bloqueio à lista de *sites* contidas no arquivo sites_bloqueados.txt. A Figura 19 mostra o arquivo

sites_bloqueados.txt com a lista de *sites* a serem bloqueados. A Figura 20 mostra que como sempre todos os *sites* são acessados no computador da diretoria (Cliente VIP), ao mesmo tempo que os *sites* listados no arquivo sites_bloqueados.txt são bloqueados para os outros computadores da rede, como mostra a Figura 21 do *site* r7.com sendo bloqueado pelo *proxy.*

Figura 19: Arquivo de configuração sites_bloqueados.txt.

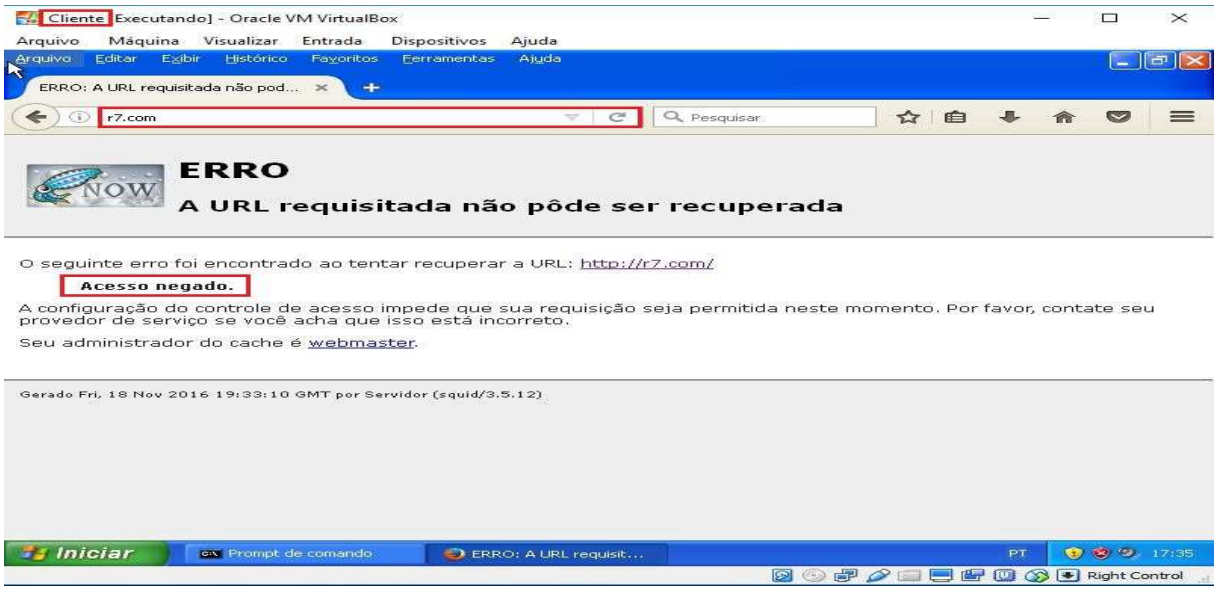


Fonte: Próprio autor.

Figura 20: *Site* r7.com liberado para o computador da diretoria.

Fonte: Próprio autor.

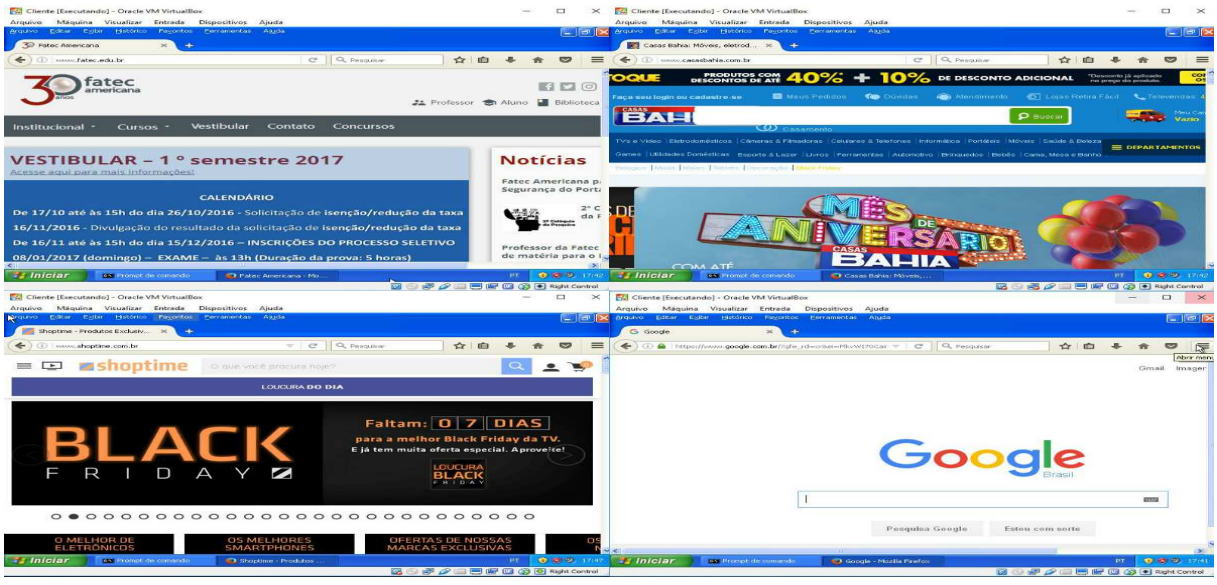
Figura 21: *Site* r7.com sendo bloqueado para computador da rede.



Fonte: Próprio autor.

Para finalizar as ilustrações de bloqueios e liberação de *sites* da Internet, a Figura 22 mostra alguns *sites* (shoptime, fatec, casas bahia e o google) sendo acessados pelo computador que representa os demais computadores da rede, e que sofreu vários bloqueios. No entanto, a última regra que foi criada, de nome rede_local, libera acesso á todos os outros *sites* da Internet que não consta nas listas de bloqueios.

Figura 22: *Sites* fora da lista não são bloqueados pelo *proxy*.



Fonte: Próprio autor.

CONSIDERAÇÕES FINAIS

A partir da apresentação de todos os dados considerados importantes à segurança da informação no âmbito da Tecnologia da Informação, pode se concluir que o *proxy squid* vem a contribuir de forma importante para um melhoramento dos seus serviços. Essa afirmação tem como embasamento o fato de que o *squid* além de proporcionar uma melhoria na performance da rede através do seu serviço de *cache,* onde é possível um maior controle do tráfego de dados da rede, ele também trabalha com regras de acesso, liberando e bloqueando acesso a *sites* da Internet. Isso se torna importante uma vez que será liberado acesso aos usuários da rede somente a *sites* confiáveis, eliminando de forma considerável as chances de o computador utilizado pelo mesmo contrair algum tipo de *vírus* e, que esse se espalhe por toda a rede.

Outra questão importante que faz o *proxy squid* se tornar ainda mais atrativo é o fato de ser uma ferramenta totalmente gratuita. Em época de crise e orçamentos cada vez mais enxuto, esta ferramenta vem para unir o útil ao agradável, sem custos se torna possível a implementação de um software que irá fazer um melhor controle de tráfego da rede, bloqueando e liberando portas, *sites* da Internet e tamanhos de objetos que poderão ser armazenados ou até mesmo baixados.

É importante salientar que, para que se tenha uma rede segura é necessário mais do que um software de *proxy* eficiente, se faz necessário também a obtenção de mais softwares de segurança, como um bom antivírus por exemplo, e também a implementação de políticas de acesso atrelada à treinamentos de todos usuários da rede por parte da equipe de segurança da informação da empresa.

Por fim pode-se dizer que a metodologia utilizada neste trabalho foi a melhor escolha, uma vez que permitiu encontrar uma quantidade de material suficiente relacionado ao tema proposto. Isso contribuiu para que o objetivo final de implementar um ambiente virtual com o *proxy squid* fosse alcançado. De todo o material consultado e abordado neste trabalho o que se pode concluir é que o *proxy squid* é um software muito eficiente no que se propõe a fazer, oferecendo principalmente serviços de bloqueio a *sites* da Internet e controle sob o tráfego de objetos que irão trafegar na rede. Ressaltando ainda o fato de o mesmo ser uma ferramenta totalmente gratuita,

dispensando dessa forma altos investimentos com ferramentas pagas e, que no final tem basicamente os mesmos objetivos do *squid*.

REFERÊNCIAS BIBLIOGRÁFICAS

ABNT . **ABNT NBR ISO/IEC 17799**: Tecnologia da informação - Técnicas de segurança - Código de prática para a gestão da segurança da informação. 2 ed. Rio de Janeiro: ABNT, 2005. 116 p.

CARVALHO, Joabes Carlos de. Proxy Squid transparente. **Viva o Linux**, [S.L], set. 2003. Disponível em: <https://www.vivaolinux.com.br/artigo/Proxy-Squid-Transparente>. Acesso em: 27 set. 2016.

HTTP://GUIDALINUX.ALTERVISTA.ORG/.**Requisitos do sistema.** Disponível em: <http://guidalinux.altervista.org/suselinux-manual_pt_br-10.1-10/sec.squid.sysneeds.html >. Acesso em: 07 nov. 2016.

LAUREANO. **Gestão de segurança da informação**. Disponível em: <http://www.mlaureano.org/aulas_material/gst/apostila_versao_20.pdf>. Acesso em: 23 set. 2016.

MARCELO, Antonio. **Squid**: configurando o proxy para linux. 5 ed. Rio de Janeiro: Brasport, 2006. 73 p.

MORIMOTO, Carlos E.. **Servidores linux**: guia prático. 2 ed. Porto Alegre - RS: Sulina, 2008. 735 p.

RASSILAN, Racy. Configurar servidor proxy squid (Ubuntu). **Viva o Linux**, [S.L], 03 set. 2009. Disponível em: <https://www.vivaolinux.com.br/artigo/Configurar-servidor-proxy-Squid-(Ubuntu)>. Acesso em: 23 set. 2016.

SILVA, Natália Vaz. Proxy Reverso com Apache. **Viva o linux**, [S.L], out. 2011. Disponível em: <https://www.vivaolinux.com.br/artigo/Proxy-Reverso-com-Apache>. Acesso em: 27 set. 2016.

SQUID-CACHE.ORG. **Squid versions**. Disponível em: <http://www.squid-cache.org/versions/>. Acesso em: 23 set. 2016.

SQUID-CACHE.ORG. **What is squid?**. Disponível em: <http://www.squid-cache.org/intro/>. Acesso em: 23 set. 2016.

SÊMOLA, Marcos. **Gestão da segurança da informação**: uma visão executiva. 2 ed. Rio de Janeiro: Elsevier Editora Ltda, 2014. 192 p.

TANENBAUM, Andrew S.. **Redes de computadores**. 4 ed. Rio de Janeiro: Campus, 2003. 632 p.

TEOTONIO, Italo Diego. Squid - Configuração básica, funcional e limpa. **Viva o linux**, [S.L], fev. 2010. Disponível em: <https://www.vivaolinux.com.br/artigo/Squid-Configuracao-basica-funcional-e-limpa>. Acesso em: 07 out. 2016.

ZANONI, Guilherme Souza. Servidor proxy (squid). **Viva o linux**, [S.L], mai. 2007. Disponível em: <https://www.vivaolinux.com.br/artigo/Servidor-proxy-(Squid)>. Acesso em: 23 set. 2016.