LAZE - Um analisador de logs do Squid

Natália Gomes Knob[1], Luis Augusto Dias Knob[1], Eduardo Germano da Silva[2], Cristian Cleder Machado[3], Diego Antonio Lusa[1]

[1]Instituto Federal de Educação Ciência e Tecnologia do Rio Grande do Sul - Campus Sertão
Rodovia RS–135, KM 25 - Sertão - RS

[2]Time de Tecnologia da Realize - Crédito, Financiamento e Investimento
Av. Dolores Alcaras Caldas, 90 - Praia de Belas - Porto Alegre - RS

[3]URI - Universidade Regional Integrada - Câmpus de Frederico Westphalen
Av. Assis Brasil, 709 - Bairro Itapagé - Frederico Westphalen - RS

nattgomesadv@gmail.com, {luis.knob,diego.lusa}@sertao.ifrs.edu.br,

eduardo.germano@realizecfi.com.br, cristian@cristian.com.br

Abstract. *This article describes the development of a Squid log analysis tool, namely LAZE, to provide useful information, through reports and graphs, to network managers in business or organizational environment. Also, seek to identify constraints and lack of resources in similar tools, look up to meet needs arising from the need to control users web requests. This paper also describe the difficulties founded in obtaining and managing a large number of logs and their processing in a relational database.*

Resumo. *Este artigo descreve o desenvolvimento de uma ferramenta de análise de logs gerados através do software Squid, chamada LAZE, que busca apresentar informações úteis, a partir de relatórios e gráficos, aos gestores de redes em ambiente empresarial ou organizacional. Ainda, procurou-se identificar limitações e ausência de recursos em ferramentas similares, buscando-se suprir limitações advindas da necessidade de controle de requisições web realizadas pelos usuários. Este artigo também busca descrever os problemas encontrados para a obtenção e gerência de uma grande quantidade de logs e seu tratamento em um banco de dados relacional.*

1. Introdução

O controle das requisições efetuadas pelos usuários de uma grande empresa ou mesmo instituição é, hoje, indispensável à segurança da mesma. A fim de realizar este controle, utiliza-se, na grande maioria dos casos, um servidor de *proxy* SQUID, que, além de ter como principal funcionalidade a realização de *cache* de requisições frequentes de objetos através dos protocolos HTTP, HTTPS e FTP, armazena em *logs* todas as requisições efetuadas. Estes *logs* possibilitam a realização de monitoramento, quando autenticado, sobre o que está sendo acessado, fluxo do tráfego e gerenciamento da rede no que se refere aos usuários.

Com a finalidade de propiciar a melhor análise dos dados brutos dos *logs* gerados, várias ferramentas foram desenvolvidas ao longo dos anos. Contudo, verifica-se na comunidade que muitas dessas ferramentas não apresentam mais atualizações, correção de

erros, ou mesmo funcionalidades necessárias ao analista de segurança da informação. É sob esta justificativa que este trabalho propõe o desenvolvimento de uma ferramenta que supra as necessidades dos profissionais responsáveis pela segurança da informação e que seja *Open Source*.

Para tanto, o presente artigo foi dividido em etapas de desenvolvimento, tendo sido organizado em diferentes seções. Inicialmente, na seção 2 serão apresentadas as considerações gerais sobre o *proxy* Squid e sobre suas particularidades para extração de informações úteis, as quais motivaram o desenvolvimento da ferramenta aqui apresentada, em conjunto a fundamentação teórica sobre o tema e a abordagem dos trabalhos relacionados, comentando-se sobre as principais dificuldades encontradas na comunidade quanto as opções de ferramentas similares. Na seção 3 será trazida a descrição da ferramenta assim como serão apresentados os requisitos e principais telas de interface. Por fim, na seção 4 apresentou-se-ão as considerações finais e o que se pretende realizar no futuro acerca da ferramenta proposta.

2. Fundamentação teórica e trabalhos relacionados

O acesso a conteúdos que estão na rede mundial de computadores acontece através de requisições feitas por um cliente a um fornecedor de serviços conhecido como servidor. Esta arquitetura, chamada de cliente-servidor, prevê a existência de um hospedeiro, sempre em funcionamento, pronto para atender a requisições de diversos outros hospedeiros, como dito acima, conhecidos por clientes. Assim, quando um servidor recebe a requisição de um objeto disponível em seu banco de dados, ele responde a requisição remetendo o objeto solicitado[Kurose 2013].

A fim de acelerar o acesso a objetos já requisitados a um servidor, no início dos anos 1990 foi desenvolvido o *proxy* Squid que, baseado no *Harvest Cache Daemon*, apresentou como função inicial o armazenamento temporário local de conteúdo previamente acessado pelos usuários. Este sistema de cacheamento é amplamente usado, sendo distribuído como software livre sob a licença GNU *General Public License*[Squid-Cache 2007].

Assim, o Squid consiste em um servidor *proxy* servindo de intermediário entre cliente e servidor, suportando requisições HTTP, HTTPS, FTP e outras conforme é possível verificar pela Figura 1. Quando um cliente realiza uma requisição HTTP, por exemplo, o pedido é interceptado pelo *proxy* e as respostas são armazenadas em *cache* permitindo redução de tempo na entrega dos objetos requisitados que já estejam armazenados e possibilitando a redução de tráfego e maior rapidez no carregamento de uma página Web. Quando o Squid consegue satisfazer inteiramente a requisição HTTP ocorre um *cache hit*. Da mesma forma, ocorre o *cache miss* quando o objeto não é encontrado em *cache*[Wessels 2004].

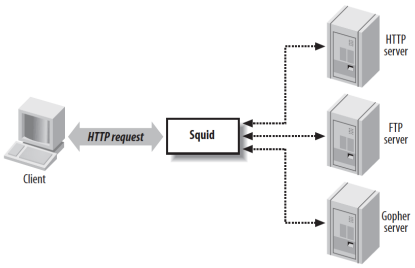
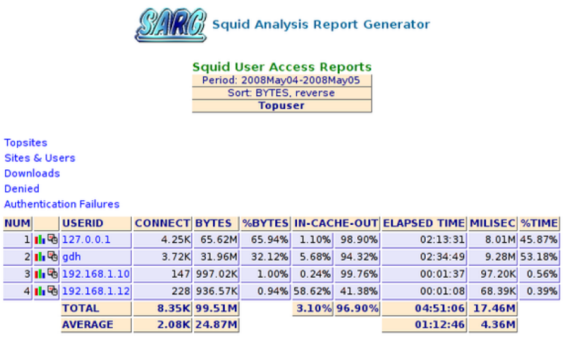


Figura 1. Figura representando a interação entre cliente-proxy-servidor

Além das funções de *cache*, o Squid pode ser utilizado para impedir que usuários visitem sites inadequados em instituições de ensino, empresas ou organizações. Haja vista que o Squid pode ser utilizado de forma transparente ao cliente, dispensando qualquer configuração extra e interceptando todas as requisições HTTP de saída, é possível realizar controle sobre todos os domínios que estão sendo acessados pelos usuários e com isso controlar o que está sendo requisitado por cada cliente HTTP [Visolve 2002].

Nativamente, o SQUID disponibiliza o *log* das requisições contendo informações acerca do horário da requisição *(time)*, quantidade de tempo, em milissegundos, que a transação ocupou o cache *(time elapsed)*, IP do cliente que solicitou a requisição *(remotehost)*, resultado da transação *(code/status)*, quantidade de dados, em bytes, enviados pelo proxy ao cliente *(bytes)*, método *(method)*, endereço web do objeto *(URL)*, identificação de usuário *(rfc931)*, hierarquia de servidores de *proxy* e como a requisição foi resolvida *(peerstatus/peerhost)* e tipo de conteúdo *(type)* [Jeffries 2015].

Existem diversas ferramentas que visam disponibilizar consultas a esses dados ou mesmo relatórios de acordo com o tráfego, clientes ou domínios acessados. Um dos mais famosos e consolidados é o SARG, ou *Squid Analysis Report Generator*, desenvolvido e mantido por Pedro Lineu Orso e programado na linguagem C. Caracteriza-se por ser software livre e é destinado a sistemas operacionais Linux. Como não é um *daemon*, não gera relatórios automaticamente. Para a geração de relatórios é necessário a utilização de *shell script*[Meier 2008]. Apesar de possuir mais robustez é de difícil configuração e perde em desempenho para o SquidLight e para outras ferramentas [Groups 2013]. A Figura 2 apresenta tabela com os usuários que requisitaram mais *bytes* dentre todas as requisições.



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 2. Tabela apresentando os *Top Users* pelo sistema SARG

Por sua vez, o MySar (MySQL Squid Access Report) destina-se prover consultas dinâmicas sendo uma ferramenta de fácil instalação. Foi disponibilizado em 2005 por Giannis Stoilis. Desenvolvido em PHP, utiliza MySQL para persistência de dados e caracteriza-se ainda por ser uma ferramenta web e livre. Embora tenha conquistado usuários entre os anos de 2012 e 2013, não possui mais atualizações a partir de 2013, não havendo correções de eventuais erros ou mesmo disponibilização de novas *features* aos seus usuários[Sourceforge 2019]. A Figura 3 apresenta um relatório diário extraído do sistema acima mencionado.

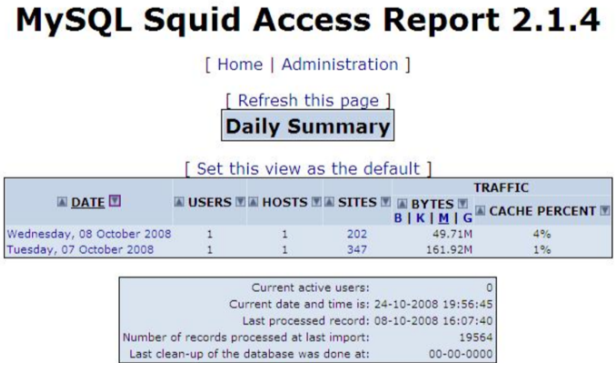


Figura 3. Relatório diário produzido através do sistema MySAR

Além das ferramentas acima citadas, existem muitas outras que se destinam à análise de *logs* gerados pelo Squid. Contudo, pelo que se verifica na comunidade, é comum encontrar sistemas que possuem limitações diferentes ou mesmo falhas que impactam em sua utilização [Blogmaru 2012]. Ainda, pelas telas dos sistemas acima trazidas, é possível notar que há simplicidade excessiva em seu *layout* e em suas tabelas.

Desta forma, propõe-se com este trabalho o desenvolvimento de uma ferramenta web de análise de *logs* de Squid que visa apresentar tabelas e gráficos acerca dos dados coletados, tratados e armazenados em banco de dados PostgreSQL. Serão disponibilizadas consultas de acordo com os domínios mais acessados, usuários com mais acessos entre outras, a fim de propiciar e facilitar a gerência do tráfego e controle dos endereços acessados e objetos requisitados.

3. LAZE, um analisador de logs do Squid

Essa seção descreve os principais parâmetros que permitiram a construção da ferramenta LAZE. Desta forma, nas próximas subseções serão abordados conteúdos relativos a descrição do software, requisitos definidos e diagramas que permitem compreensão mais precisa acerca da ferramenta desenvolvida.

3.1. Descrição da ferramenta

A ferramenta LAZE foi concebida com o propósito geral de proporcionar ao seu utilizador uma grande quantidade de relatórios e estatísticas sobre as requisições capturadas pelo *proxy* Squid e armazenadas em forma de arquivos de *logs*, de forma com que seja possível apurar informações essenciais sobre quantidade de tráfego, domínios mais requisitados, clientes que utilizam maior largura de banda, entre outros. Com isso, entende-se que o controle a conteúdos indesejados possa ser facilitado e melhor definido a nível organizacional/empresarial ou mesmo em instituições de ensino, saúde e outros.

3.2. Definição de requisitos

De acordo com Sommerville, requisitos podem ser definidos como aquilo que o sistema faz, suas restrições de funcionamento e quais serviços oferece. Assim, a definição de requisitos consiste, em termos generalistas, na especificação de funções ou mesmo recursos de um sistema [Sommerville 2011].

Para ferramenta LAZE foram identificados alguns requisitos, conforme descrito a seguir:

- **Controlar o acesso à aplicação por meio de login** - Para ter acesso a ferramenta LAZE, o usuário deverá, inicialmente, realizar o *login*, indicando nome de usuário e senha, para que seja carregado seu perfil. Será guardada informação sobre a sessão e sobre o usuário que realizou o *login* para que haja redirecionamento à página inicial com o reconhecimento de seus dados.
- **Manter usuários do sistema** - O sistema armazenará em banco de dados todas as contas de usuários cadastradas por um administrador, cuja conta será criada pela ferramenta após sua instalação. O administrador do sistema terá perfil próprio o que permite a criação/cadastro de demais usuários. Cada usuário do sistema terá perfil próprio e, a cada cadastro, serão informados nome completo, *alias* ou *username*, senha e e-mail. A senha será armazenada de forma criptografada, a fim de manter sigilo e segurança dos dados. As funcionalidades e permissões serão disponibilizadas de acordo com o perfil do usuário.
- **Receber arquivos de log através de upload** - Os dados brutos das requisições capturadas pelo Squid serão inseridos através do *upload* dos arquivos.
- **Realizar o *parsing* do arquivo carregado** - Carregado o arquivo, será iniciado serviço em *background* para a divisão, tratamento e inserção dos dados em tabelas próprias do banco de dados.
- **Apresentar gráfico dos *top clients*** - Em *dashboard* será apresentado gráfico contendo estatísticas referentes a usuários com maior número requisições realizadas no ano corrente.
- **Apresentar gráfico dos *top domains*** - Em *dashboard* será apresentado gráfico contendo estatísticas referentes a domínios que mais receberam requisições no ano corrente.
- **Apresentar gráfico de total de bytes trafegados** - Em *dashboard* será apresentado gráfico contendo o total de *bytes* trafegados por mês, em ano corrente.
- **Apresentar gráfico de resultado de requisições** - Em *dashboard* será apresentado gráfico contendo o percentual referente à quantidade de requisições, de acordo com seus resultados e de acordo com ano corrente.
- **Apresentar relatório por domínio** - Disponibilização de relatório em forma de tabela contendo informações sobre requisições de acordo com o domínio pesquisado.
- **Apresentar relatório por cliente** - Disponibilização de relatório em forma de tabela contendo informações sobre requisições de acordo com o cliente pesquisado.
- **Apresentar relatório por resultado de requisição** - Disponibilização de relatório em forma de tabela contendo informações sobre requisições de acordo com o resultado destas.
- **Exportar relatório em PDF** - Disponibilização da exportação das estatísticas, após a aplicação de filtros de pesquisa, em arquivo PDF.

3.3. Diagrama Entidade-Relacionamento do Banco de Dados

Optou-se, para o desenvolvimento da ferramenta proposta, a utilização de banco de dados relacional, contudo, com viés dimensional.

Inicialmente, a ferramenta foi concebida para utilizar banco de dados não relacional, tendo-se optado pela utilização do MongoDB. Ocorre que, a natureza dos *logs* de requisições do Squid é eminentemente estruturada. Tendo-se verificado que não havia desempenho satisfatório na recuperação das informações quando armazenadas na forma de documentos para a construção de gráficos e tabelas, escolheu-se migrar para um banco de dados relacional, utilizando-se o *star join*, tipo de modelo baseado em fatos e dimensões.

O modelo fato-dimensões, também nomeado modelo multidimensional, é conhecido como técnica de modelagem comumente utilizada em *Data Warehouse*, propiciando, como principal característica, o rápido acesso a informações para análise e tomada de decisões. Assim, os sistemas dimensionais acabam se diferenciando dos sistemas transacionais tendo em vista que focam principalmente na recuperação de informações de forma mais célere e ágil, não tendo como fator determinante a forma com que ocorrem as inserções. Comumente trabalham com dados não normalizados [Inmon 2005].

O modelo dimensional conta com uma ou mais tabelas que representam fatos, aonde estão contidas ocorrências de dados, e tabelas que representam as dimensões, as quais descrevem um aspecto importante da tabela de fatos. Além disso, o modelo pode ser estruturado em estrela (*star join*) ou floco de neve (*snowflake*) sendo que a diferença entre os dois reside no fato de que a segunda estrutura nada mais é do que a combinação de mais tabelas de fatos em uma estrutura estrela composta [Inmon 2005].

Desta forma, como se pode verificar pela Figura 4, o modelo do banco de dados foi construído sob a estrutura de estrela, onde a tabela fato representa cada linha de um *log* do Squid e as tabelas de dimensões apresentam informações detalhadas sobre campos desta mesma linha de *log*.

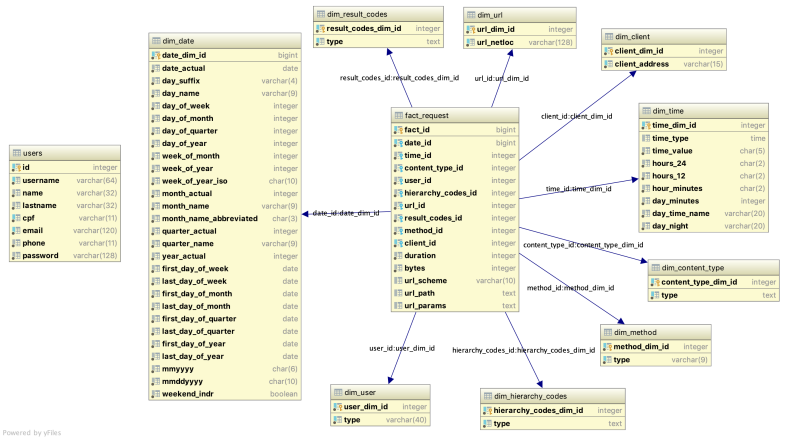


Figura 4. Modelagem do banco de dados da ferramenta LAZE

3.4. Interface com o usuário

A interface de gerenciamento foi construída utilizando a linguagem Python, o *framework web* Flask e um *template* desenvolvido em Bootstrap. Entre as principais funcionalidades

da interface, a figura 5 apresenta o *Dashboard* inicial disponível para o administrador da rede. Nele são apresentados gráficos referentes ao TOP 10 de domínios mais acessados, TOP 10 de usuários por consumo em MB, TOP 10 de conteúdo das requisições, consumo em MB por mês no ano corrente e consumo em MB por dia no mês corrente.

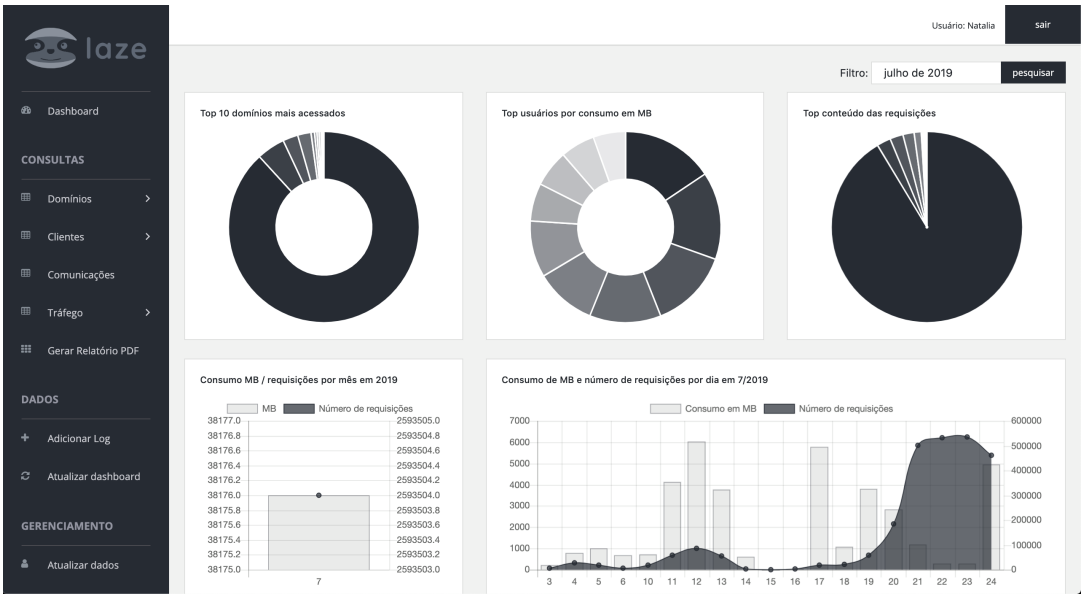


Figura 5. Dashboard da ferramenta LAZE

Além do *dashboard* o administrador possui relatórios por diversas categorias, sendo elas, domínios acessados por mês e dia, endereços do usuário de origem por mês e dia, relação entre domínios e usuários, resultado de requisições, entre outros. Todos os relatórios podem ser ordenados e filtrados por características de acordo com a sua natureza (ex.: duração, tamanho, quantidade de requisições, hora de origem).

Como o LAZE possui um modelo de dados distinto daquele comumente utilizado por ferramentas de importação automática de *logs* do Squid, foi necessário criar um interpretador de *logs* para a adição destes no postgreSQL. Pela quantidade de *logs* gerados, mesmo em um pequeno cenário, o desempenho do Python deixa a desejar em comparação a outras linguagens, tendo em vista que esta não é compilada, optando-se por desenvolver este módulo na linguagem Go, utilizando *go routines* para paralelizar a inclusão no banco de dados.

4. Considerações Finais e Trabalhos Futuros

O artigo propôs a ferramenta LAZE, que possui como objetivo alcançar a aceitação dos usuários e colaborar com a comunidade. Possuindo uma interface *Web*, disponibiliza gráficos e tabelas de acordo com as consultas realizadas, prezando-se pelo desempenho nas consultas realizadas e apresentação de informações em tempo real aos usuários do sistema.

Verificou-se que o sistema atende as necessidades descritas acima e que apresenta diferenciais relevantes em relação a outras ferramentas *open source* estudadas. Isso porque além de ser um sistema *web*, traz gráficos aos usuários e relatórios facilmente exportáveis em formato PDF. Cabe dizer ainda que as informações apresentadas em gráficos

e relatórios são dinâmicas, ou seja, basta que seja realizado um novo *upload* de *log* que estes gráficos e relatórios passarão a considerar os novos dados inclusos em sistema.

Dentre as principais dificuldades enfrentadas para o desenvolvimento da ferramenta inicialmente proposta, cabe citar a limitação dos bancos de dados para armazenar e operar centenas de milhões de registros. Inicialmente, optou-se utilizar o banco de dados MongoDB haja vista que uma requisição de um *log* poderia ser desmembrada e persistida como um documento em uma coleção do banco, sem necessidade de *joins* para a recuperação dos dados. Embora inicialmente o MongoDB parecesse o banco mais adequado, verificou-se uma grande dificuldade em se realizar consultas complexas. Ademais, verificou-se que seu desempenho deixou de ser satisfatório com o aumento gradual de documentos na coleção. Com isso, optou-se migrar para o banco de dados PostgreSQL optando-se por modelagem multidimensional, comumente utilizada em *Data Warehouse*.

Apesar de alcançado o resultado esperado com o desenvolvimento da ferramenta, pretende-se, em trabalhos futuros, apresentar módulo que reporte os principais usuários (clientes em requisições) que possuem muitas requisições a domínios indesejados ou que gastem muito tempo acessando estes mesmos domínios, procurando-se identificar perfis com comportamentos inadequados à política de uso da rede de uma organização.

Referências

Blogmaru (2012). Ferramentas de análise do squid proxy-#01. `http://rauhmaru.blogspot.com/2012/06/ferramentas-de-analise-do-squid-proxy.html`. Abril.

Groups, G. (2013). Sarg x lightsquid no pfsense. `https://groups.google.com/forum/#!topic/oesc-livre/id8s9OEzW9s`. Abril.

Inmon, W. H. (2005). *Building the Data Warehouse*. Wiley, 4 edition.

Jeffries, A. (2015). Feature: Customizable log formats. `https://wiki.squid-cache.org/Features/LogFormat`. Abril.

Kurose, J. F., R. K. W. (2013). *Redes de Computadores e a Internet: Uma abordagem top-down*. Pearson Education do Brasil Ltda., 6 edition.

Meier, A. M. (2008). Sarg, o dedo-duro amigo do administrador. `http://www.linuxnewmedia.com.br/images/uploads/pdf_aberto/LM44_pag60-63.pdf`. Abril.

Sommerville, I. (2011). *Engenharia de Software*. Pearson Prentice Hall, 9 edition.

Sourceforge (2019). Mysql squid access report. `https://sourceforge.net/projects/mysar/support`. Abril.

Squid-Cache (2007). About squid. `http://www.squid-cache.org/Intro/`. Abril.

Visolve (2002). Trans_caching. `https://www.visolve.com/squid/whitepapers/trans_caching.html`. Abril.

Wessels, D. (2004). *Squid: The Definitive Guide*. O'Reilly Media, 1 edition.