CHRYSTIAN DA SILVA MARQUES

GABRIEL REZENDE OLIVEIRA

ADMINISTRAÇÃO CENTRALIZADA E SINCRONIZADA DE SERVIDORES PROXY USANDO SQUID

INCONFIDENTES-MG

2016

CHRYSTIAN DA SILVA MARQUES
GABRIEL REZENDE OLIVEIRA

ADMINISTRAÇÃO CENTRALIZADA E SINCRONIZADA DE SERVIDORES PROXY USANDO SQUID

Trabalho de Conclusão de Curso apresentado como pré-requisito de conclusão do curso de Graduação Tecnológica em Redes de Computadores no Instituto Federal de Educação, Ciência e Tecnologia do Sul de Minas Gerais – Campus Inconfidentes, para obtenção do título de Tecnólogo em Redes de Computadores.

Orientador: Vinícius Ferreira de Souza

INCONFIDENTES-MG
2016

CHRYSTIAN DA SILVA MARQUES
GABRIEL REZENDE OLIVEIRA

ADMINISTRAÇÃO CENTRALIZADA E SINCRONIZADA DE SERVIDORES PROXY USANDO SQUID

Data de aprovação:

27 de outubro de 2016

Orientador: Prof. Vinícius Ferreira de Souza
IFSULDEMINAS – Campus Inconfidentes

Prof. Kleber Marcelo da Silva Rezende
IFSULDEMINAS – Campus Inconfidentes

Prof. Igor Oliveira Lara
IFSULDEMINAS – Campus Inconfidentes

Instituto Federal de Educação, Ciência e Tecnologia Sul de Minas Gerais
Campus Inconfidentes
Praça Tiradentes, 416 - Centro - CEP
37576-000 Telefone: (35) 3464-1200

Administração Centralizada e Sincronizada de Servidores Proxy usando Squid

Chrystian S. Marques[1], Gabriel R. Oliveira[1], Vinícius F. de Souza[1]

[1]IFSULDEMINAS – Campus Inconfidentes – Inconfidentes, MG – Brasil

chrystianmarques2007@gmail.com, g.rezende.oliveira@gmail.com

vinicius.souza@ifsuldeminas.edu.br

Abstract. *This paper presents a solution to centralize and synchronize the management proxy servers using open source software squid. This solution was implemented in a company composed of forty branches, in which all them use a proxy server to monitor web access. Through the squid in the array configuration, it was possible to centralize all proxy changes and replicate them to each branch servers, systematic and automated way. This implementation provided to reduce the time spent on updating servers and prevented any server stayed out of date depending on the configuration performed manually.*

Resumo. *Este trabalho apresenta uma solução para centralizar e sincronizar a administração de servidores proxy usando o software livre squid. Tal solução foi implementada em uma empresa composta por quarenta filiais, na qual todas utilizam um servidor proxy para monitorar o acesso web. Por meio da configuração do squid na matriz, foi possível centralizar todas as alterações de proxy e replicá-las aos servidores de cada filial, de modo sistematizado e automatizado. Essa implementação proporcionou a redução do tempo gasto na atualização dos servidores e evitou que algum servidor ficasse desatualizado em função da configuração realizada manualmente.*

1. Introdução

Atualmente, as organizações utilizam diversas ferramentas para realizar a filtragem de conteúdo web. O objetivo dessa filtragem é permitir ou negar acesso às informações em redes com base nas regras explícitas em uma política de segurança (GURGEL et al., 2015).

Isso acontece porque as empresas esperam de seus colaboradores o melhor rendimento possível. A TI tem uma grande responsabilidade nisso, provendo meios do

desempenho do colaborador ser o esperado pela empresa e evitando que distrações atrapalhem o rendimento de trabalho, como por exemplo, colaboradores acessando sites desnecessários para o desenvolvimento do serviço ou esperando que um site seja liberado. Nesse contexto surge a necessidade da filtragem de conteúdo web por parte da empresa, tendo assim, um controle sobre o acesso HTTP (*Hypertext Transfer Protocol*), com regras de sites liberados, bloqueados, liberação por MAC e IP das máquinas, regras especiais para diretoria, coordenadores e setor de TI, por exemplo.

A ferramenta *squid* é um dos proxies para Linux mais utilizados na Internet. Trata-se de um software robusto, simples e confiável, que melhora o desempenho da conexão e permite criar regras de acesso para servidores web. O *squid* é software livre, nativo da plataforma *Linux*, mas que pode ser executado em outras plataformas. Atua como um servidor *proxy* para alguns serviços específicos da camada de aplicação, tais como HTTP e FTP (*File Transfer Protocol*) (MARCELO, 2006).

Um servidor *proxy* é um agente, localizado entre o cliente e o servidor destino, no qual existe uma lista de regras que valida qualquer requisição ou resposta, permitindo ou negando o acesso. Esse filtro funciona por meio de regras impostas pela política da rede, podendo assim bloquear determinados sites, protocolos ou conteúdo de mensagens, o que representa uma grande vantagem no ambiente corporativo, pois, com esse bloqueio, os funcionários têm menos chance de se distraírem, reduzindo a sua produtividade (GURGEL et al., 2015).

Outra vantagem, é a capacidade de armazenamento temporário de documentos. Uma cópia da página acessada é colocada no cache do *proxy*, que é uma área com um tamanho especificado pelo administrador e que contém as requisições ao sistema. Desse modo, caso um usuário acesse a mesma página a partir de outra estação, o *proxy* fornece para esse usuário a página contida em seu cache, o que otimiza o acesso à Internet. Além disso, o *proxy* pode ser utilizado como ferramenta para auditoria de acessos, já que possui a capacidade de armazenar em seus arquivos de log todas as conexões realizadas (MARCELO, 2006).

A Figura 1 representa, de forma esquemática e simplificada, a topologia de rede empregada nesta implementação. A estação cliente não tem acesso direto ao servidor web na Internet. Como o servidor *proxy* está ciente de toda a comunicação web da rede interna, é possível armazenar as requisições e as respostas de cada cliente na rede interna,

mantendo um histórico arquivado, e usar essas informações para um maior controle da rede.

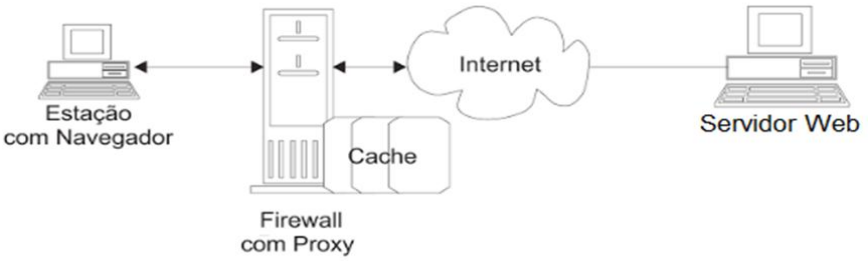

Figura 1 - Representação da topologia de rede com firewall *proxy* (MARCELO, 2006).

Ao aplicar um *proxy* em uma empresa que possui 40 (quarenta) filiais, senão houver um controle assíduo sobre esses servidores, a tendência é que a padronização seja perdida, fazendo com que cada filial tenha suas regras, liberações e bloqueios, sendo que, o correto é haver uma centralização e sincronização das configurações, mantendo um maior controle e administração sobre todos servidores. Essa centralização e sincronização, diminui a probabilidade por falha humana da equipe de TI e também o tempo que seria gasto para uma alteração manual em massa, além de manter todos servidores padronizados.

Este trabalho apresenta uma solução que foi implementada em uma empresa, localizada na cidade de Estiva/MG, que conforme foi citado, possui quarenta filiais e cada uma utiliza um servidor *proxy* para controlar o acesso web. A configuração proposta utilizou o *squid* para centralizar e sincronizar as atualizações e alterações que são realizadas em cada filial, reduzindo assim o tempo consumido para realizar a configuração manual e evitando que algum servidor ficasse desatualizado.

2. Materiais e Métodos

Para a realização da implementação proposta, foram utilizadas as seguintes versões de softwares: *Linux Debian 7.0, Squid 3.1.20, OpenSSH 6.0p1 e Rsync 3.0.9*. O *squid* foi instalado em todas filiais para funcionar como servidor *proxy*. A solução proposta é independente da versão do *squid*, pois manipula arquivos de configuração do mesmo.

Com uma administração descentralizada dos servidores *proxy* de cada filial, uma atividade de liberação ou bloqueio de páginas web exigia, por parte do administrador da rede, um tempo considerável para uma execução dessa tarefa. Estimou-se, por observação, o tempo médio gasto pelo administrador da rede para liberar ou bloquear

determinada página web era de três minutos e meio. Portanto, para executar essa mesma tarefa em quarenta servidores, o tempo seria de, aproximadamente, duas horas e meia.

Verifica-se que, por essa estimativa que esse tempo necessário é inaceitável para os moldes atuais de economia de tempo, padronização e melhoria da qualidade. Sendo que, ainda não foi levado em conta erros humanos, que podem acontecer no processo, por exemplo, o administrador de redes errar uma letra ao liberar ou bloquear a página web. Neste estudo de caso, muitas das vezes ao terminar de liberar ou bloquear uma página web, outra solicitação com mais um site para liberação ou bloqueio era realizada, o que tornava o trabalho repetitivo e cansativo.

Para a conexão entre matriz e filiais foi utilizado o protocolo SSH, que oferece um serviço de acesso remoto, permite transferir arquivos em diferentes formatos e, também, encapsular outros protocolos. O túnel SSH proporciona maior segurança na transferência de dados, visto que o protocolo requer autenticação, utiliza criptografia para proteger as mensagens trocadas e mantém a integridade da conexão estabelecida (MORIMOTO, 2009).

Para isso, é necessária uma troca de chaves entre os servidores da matriz e filiais, essas chaves são responsáveis por criptografar e descriptografar as informações que serão sincronizadas. Essa troca de chaves deve ser feita antes da execução do script de sincronização, conforme o Quadro 1:

Quadro 1. Troca de chaves SSH entre servidores da matriz e filial.

```
ssh-keygen -t rsa
ssh-copy-id -i /root.ssh/id_rsa.pub [usuário da matriz]@[IP servidor
da matriz]
```

Sendo que:

- [usuário da matriz]: Um usuário no servidor da matriz com permissão de leitura e escrita na pasta *squid*;
- [IP servidor da matriz]: Um endereço ao qual as filiais tenham acesso ao servidor da matriz, pode ser um IP público ou mesmo privado se a empresa possuir uma VPN ou conexão ponto a ponto.

Ao aplicar o primeiro comando, os seguintes questionamentos serão apresentados:

- *Enter file in which to save the key (/home/suporte/.ssh/id_rsa)*: local onde a chave SSH será gerada, indicando entre parênteses o caminho padrão que será utilizado neste caso;

- *Enter passphrase (empty for no passphrase)*: senha para a chave SSH, caso necessário, não será utilizada no *script* atual;

- *Enter same passphrase again*: confirmação da senha, caso utilizada.

Após executar esses passos, a chave será gerada e exibida no terminal em formato RSA 2048. Ao aplicar o segundo comando, a senha do [usuário da matriz] será requisitada para que a chave possa ser gravada no servidor da filial e será utilizada para troca de informações futuras.

A partir desse momento, o servidor da filial tem acesso via chave SSH ao servidor da matriz com o usuário informado na troca de chaves, não sendo necessário assim o uso de senha.

O *crontab* foi aplicado para agendar a periodicidade (na configuração utilizada, a cada dois minutos) na qual cada filial conecta-se à matriz para verificar se há alguma alteração nas regras do *squid*. Após a leitura da pasta do *squid*, havendo alguma alteração, ela é copiada para o *squid* da filial. O registro da operação é armazenado em log, identificando o horário da conexão e qual filial fez a atualização, o que permite o monitoramento das atualizações realizadas em cada filial (RICCI; MENDONÇA, 2006).

O arquivo *crontab*, presente no Debian em /etc/crontab, é apresentado no Quadro 2 (a linha em negrito é usada para agendamento da sincronização do *squid*):

Quadro 2. Arquivo *crontab* no Debian.

```
# /etc/crontab: system-wide crontab
# Unlike any other crontab you don't have to run the `crontab'
# command to install the new version when you edit this file
# and files in /etc/cron.d. These files also have username fields,
# that none of the other crontabs do.

SHELL=/bin/sh
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bi
```

```
n

     # m h dom mon dow user  command

     17 *    * * *   root    cd / && run-parts --report /etc/cron.hourly

     25 6     * * *    root     test -x /usr/sbin/anacron || ( cd / &&
run-parts --report /etc/cron.daily )

     47 6     * * 7    root     test -x /usr/sbin/anacron || ( cd / &&
run-parts --report /etc/cron.weekly )

     52 6     1 * *    root     test -x /usr/sbin/anacron || ( cd / &&
run-parts --report /etc/cron.monthly )

     */10 *  * * *   root    sarg

     */2 *   * * *   root    /root/scripts/./sincronizacao.sh

     #
```

A linguagem utilizada para o script na edição dos arquivos foi a Shell Script. O *rsync* foi usado para sincronizar o conteúdo das pastas, de forma a transferir somente as modificações realizadas. O *rsync* verifica o conteúdo de cada arquivo e, caso somente uma parte do arquivo seja alterada, apenas essa modificação é transferida, sem a necessidade de copiar todo o arquivo novamente (MORIMOTO, 2009). O script utilizado usufruiu dos seguintes parâmetros do comando *rsync*:

- -a, --*archive*: modo arquivo, ou seja, o *rsync* não irá copiar pastas, mas arquivo por arquivo;
- -v, --*verbose*: modo verboso, com logs sobre o andamento das cópias;
- -z, --*compress*: comprime durante a transferência, tendo assim uma economia de banda;
- -h, --*human*: saída num formato legível para humanos.

O Quadro 3 apresenta o script utilizado para sincronização:

Quadro 3. Script de Sincronização.

```
#!/bin/bash
############################################################
# Script de Sincronizacao Squid
############################################################
```

```
rsync -avzh [usuário da matriz]@[IP Servidor da
matriz]:/etc/squid3/[pasta da matriz a  ser sincronizada]/
/etc/squid3/[pasta da filial a ser sincronizada]/


ssh [usuário da matriz]@[IP servidor da Matriz] 'filial=[nome da
filial] && echo -e "`date +"%d/%m/%y - %A - %H:%M"`\n$filial
Sincronizada\n------------------------------" >>
/var/log/sincronizacao_squid/$filial.log && echo -e "`date +"%d/%m/%y
- %A - %H:%M"`\n$filial Sincronizada\n------------------------------"
>> /var/log/sincronizacao_squid/sincronizac.log'


invoke-rc.d squid3 reload
```

Sendo que:

- [pasta da matriz a ser sincronizada]: a pasta que será sincronizada por todas filiais, pode ser a pasta principal do *squid* (/etc/squid3), ou pode ser alguma pasta que o administrador de redes definir. Recomenda-se que essa pasta fique dentro da pasta principal do *squid*;

- [pasta da filial a ser sincronizada]: pasta destino na filial onde os arquivos presentes na matriz serão copiados e, novamente, recomenda-se que essa pasta esteja na pasta principal do *squid*;

- [nome da filial]: variável com o nome da filial para registro em log.

Esse script deve estar configurado em todas filiais que serão sincronizadas.

3. Resultados e Discussões

A centralização da administração dos servidores *proxy* das filiais, com o uso do *squid*, trouxe unidade à configuração das regras de liberação/bloqueio de páginas. Assim, o servidor da matriz concentra todas as informações e, quando necessário, realiza as alterações nos servidores de cada filial. O processo ocorre de forma automatizada e sistematizada, o que reduz o tempo total de configuração das alterações em todos os servidores e elimina a possibilidade de algum servidor ficar desatualizado ou apresentar um erro de configuração local.

Foi testado a abordagem inversa, ou seja, após a alteração na matriz essa iria conectar em todas unidades, realizar e aplicar as modificações, porém isso gerou um alto

tráfego de rede e consumo de processamento na matriz e uma possibilidade maior de erro, visto que um único servidor era responsável pelo gerenciamento de todos servidores, da forma atual, cada servidor continua responsável pelo seu serviço, a matriz apenas detém as informações e o poder de alteração delas em todas filiais.

Qualquer alteração que seja realizada localmente em uma filial, será removida após a próxima sincronização entre os servidores, o que confere à matriz o poder de controlar as alterações realizadas, garantindo a padronização da política de acesso em todas as filiais. O Quadro 4 mostra alguns exemplos de arquivos de sites liberados.

Quadro 4. Arquivo de sites liberados na matriz.

```
www.trinus.com.br
https://www.esnfs.com.br/
http://fleetcard.com.br/
http://sistema.fleetcard.com.br/
c3sl.ufpr.br
http://www.validcertificadora.com.br
http://islonline.net
http://www2.copasa.com.br/servicos/2avia2/fatura2avia.asp?
br.com.br/
187.60.159.183
vpn3.direcao.com.br
http://mixsolucoesambientais.com.br
www.algar.com.br/
```

Caso alguma filial fique sem acesso ao servidor da matriz, o acesso à Internet não será comprometido na filial. Quando o acesso à matriz for restabelecido, todas as informações do *squid* serão atualizadas na próxima sincronização. Os logs ficam disponíveis na matriz para consulta e monitoramento da sincronização. O Quadro 5 apresenta um exemplo do arquivo de log de uma filial.

Quadro 5. Arquivo de log de sincronização de uma filial.

08/10/15 - quinta - 13:59 DP1 Sincronizado

08/10/15 - quinta - 13:59 DP1 Sincronizado

08/10/15 - quinta - 14:00 DP1 Sincronizado

08/10/15 - quinta - 14:02 DP1 Sincronizado

08/10/15 - quinta - 14:04 DP1 Sincronizado

14/10/15 - quarta - 09:40 DP1 Sincronizado

O Quadro 6 apresenta um exemplo do arquivo geral de log:

Quadro 6. Arquivo de log de todas filiais.

17/08/16 - quarta - 21:01
DPPindaNorte Sincronizado

17/08/16 - quarta - 21:01
DPContagemNorte Sincronizado

17/08/16 - quarta - 21:01
DPFumaca Sincronizado

17/08/16 - quarta - 21:01
DPCambuiUrbano2 Sincronizado

17/08/16 - quarta - 21:01

DPLeme Sincronizado

17/08/16 - quarta - 21:01
DPAlianca Sincronizado

17/08/16 - quarta - 21:01
DPBelaVista Sincronizado

17/08/16 - quarta - 21:01
DPGuaratingueta Sincronizado

17/08/16 - quarta - 21:01
DPUruacu Sincronizado

Esse log registra o acesso de todas filiais ao servidor matriz, podendo ser monitorado o horário que foi feita a sincronização.

5. Conclusões

A solução proposta para a centralização e sincronização de servidores *proxy* através do *squid* garantiu a padronização da política de acesso em todas as filiais, reduziu o tempo total gasto para alterar ou atualizar localmente todos os servidores e eliminou a possibilidade de erro ou desatualização em algum servidor de forma individual. Todos os arquivos de log ficam disponíveis na matriz, permitindo o controle e o monitoramento das atualizações realizadas. Além disso, a solução apresentada é aberta e independentemente da versão utilizada do *squid*, podendo ser implementada em qualquer plataforma que tenha suporte para *squid*, *rsync*, *crontab* e SSH.

O *rsync* foi essencial para economia de banda, pois com esse comando cada filial faz uma cópia do arquivo de regras do *squid* somente caso haja alteração no arquivo, caso contrário, não será necessária a sincronização.

A ferramenta está em operação na empresa, e sanou vários problemas referentes à falta de informações em algumas filiais e ao desperdício de tempo em processos que poderiam ser automatizados, trazendo segurança, confiabilidade e rendimento para equipe de TI e demais colaboradores da empresa.

Referências

GURGEL, P. H. M. et al. **Redes de Computadores:** da teoria à prática com Netkit. Rio de Janeiro: Elsevier, 2015.

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

MORIMOTO, C. E. **Servidores Linux:** guia prático. Porto Alegre: Sul Editores, 2009.

RICCI, B.; MENDONÇA, N. **Squid:** solução definitiva. Rio de Janeiro: Ciência Moderna, 2006.