Squid? NAT? Transparente? Hein?

Introdução

Este guia tem como objetivo ser o mais simples possível, porém isto não quer dizer que ele vai ser superficial (Ao contrário de outras documentações de minha autoria).

Resolvi começar a documentar alguns tópicos de GNU/Linux a algum tempo atrás, começando com este guia e como conseqüência deste, surgiram o sobre NIS e o sobre Postfix (Que também precisam de um update).

Para os que já conheciam o guia nos outros formatos (PDF e HTML), não estão totalmente interagidos com o modelo Wiki (E tampouco estou eu. Sou tão novato quanto você nesse negócio). Como todo o WIKI, agora você pode editar este texto! Além disso, o guia vai estar mais atualizado, partindo da contribuição de novas pessoas.

Notem que eu vou levar algum tempo até editar todo o conteúdo e enviar para o Wiki, por isso ele vai crescendo aos poucos, ok?

Este guia é disponibilizado no formato "as is", não possui garantia alguma de que os procedimentos aqui citados vão funcionar e julgo-me isento de qualquer responsabilidade que este possa vir a causar.

Nota de Copyright

Tabela de conteúdo

Introdução

Atualmente, os servidores proxy deixaram de ser um simples privilégio e passaram a ser uma necessidade dentro das mesmas. Neste guia eu darei uma introdução ao assunto e explicarei (Ou pelo menos tentarei) como montar um proxy. Nós veremos também que existem vários tipos de proxies, como o proxy transparente, o proxy autenticado ou simplesmente um NAT, roteando os pacotes oriundos da internet.

Proxy e Cache

Provavelmente ao ler este guia você já saiba o quê é um servidor proxy e só quer saber como você faz para montar um. Para os inexperientes, você pode continuar a ler isto aqui.

Um proxy serve para basicamente uma coisa: tratar as requisições dos clientes nos protocolos suportados por ele, seja bloqueando URL's de sites indevidos ou controlando o acesso à WWW pelos usuários.

O Squid além de um servidor proxy, também integra um sistema de caching para requisições feitas à internet. Um servidor cache é aquele que armazena informações de requisições temporariamente em disco para uma visualização mais rápida posterior.

Eis o funcionamento de um servidor proxy-cache passo-a-passo:

1 - O usuário João faz uma requisição para o site X, cujo endereço é "www.x.com.br";
2 - Ele aperta enter logo depois que ele digitou o endereço do site na barra de endereços de seu navegador predileto;
3 - O navegador faz uma requisição pela página para o servidor proxy;
4 - O servidor proxy recebe a requisição cliente e checa se a página está em cache. Em caso positivo, ele manda para o cliente requisitante a página armazenada em cache. e a história morre por aqui. Em caso negativo, ele salta para o próximo passo;
5 - Ele faz uma requisição para a internet da página que o cliente solicitou. Feita a transferência da página, ele armazena a mesma no cache e envia a página para o cliente solicitante;

Nota-se que com a implementação de servidores proxy-cache sua banda é melhor utilizada e melhor controlada, pois o navegador não faz inúmeras requisições duplicadas para a internet.

O que é o Squid?

O Squid é o servidor proxy HTTP mais comum em plataformas UNIX-Like. Ele surgiu de um projeto entre o governo americano e a Universidade do Colorado.

Atualmente o Squid trabalha com os protocolos HTTP, HTTPS, FTP, Gopher e WAIS e é o proxy que possui o maior número de co-projetos.

ATENÇÃO! O Squid não suporta NENHUM outro protocolo que não seja um dos citados acima! Isto inclui: SMTP, POP e IMAP!

Proxy? NAT? Transparente? Hein?

Você deve estar se pergunta nesta altura do campeonato um pouco da terminologia que eu vou usar neste guia (Caso vocÊ seja um faixa-branca). As diferenças entre proxy transparente, proxy autenticado, proxy isso, aquilo e por aí vai.

Proxy Transparente - É um redirecionamento de portas feito com o IPTables. Todos os pacotes com destino à porta 80 para a internet são redirecionados para a porta do Squid. Ele é chamado de "transparente" porque o usuário muitas vezes nem sabe qeu ele está passando por um proxy para conseguir a sua página. Muito utilizado em migrações de outros tipos de proxy e para garantir que todo mundo esteja passando pelo proxy, como medida de segurança.
Proxy Autenticado - É um proxy que exige autenticação para liberar a navegação. Quando o usuário abre o navegador, é aberta uma janela a mais, solicitando um login e uma senha para a liberação da internet.
NAT - O NAT (Network Address Translator) é um redirecionamento de pacotes oriundos da rede local para a rede pública e vice-versa, só roteando os pacotes, sem tratamento algum.

Vantagens e desvantagens

Bem, talvez você ainda precise de mais algumas informações sobre qual tipo de proxy utilizar e quais sobre qual tipo de proxy utilizar e quais serão os empecilhos que você pode encontrar na utilização de cada um destes itens.

Proxy Transparente - A grande vantagem do proxy transparente é que todo o tráfego referente ao protocolo HTTP é redirecionado para ele. Sendo assim, não há como o usuário alterar alguma configuração na máquina localmente para tentar "burlar" o controle feito pelo Squid. Por outro lado, o proxy transparente é incompatível com o proxy autenticado e com algumas tecnologias utilizadas hoje em dia na internet, como alguns softwares da Caixa e algumas tecnologias de criptografia SSL com HTTP (HTTPS);
Proxy Autenticado - Uma grande vantagem do proxy autenticado é que ele permite o controle individual de acesso a nível de usuário, e não por máquinas (hosts). Algo que eu acho que é uma desvantagem é que se você fechar a janela de navegação utilizada e abri-la novamente, a senha será requisitada novamente. Por outro lado, imagine se você está navegando, fecha o navegador e sai pra almoçar. Então o cara que senta ao seu lado e está concorrendo à uma vaga com você vai na sua mesa e acessa um site de sacanagem, como sendo você. Outra coisa que eu acho um defeito no proxy autenticado é que alguns softwares que utilizam tunneling por HTTP, mas que não são navegadores - como alguns anti-vírus - não oferecem suporte à configuração deste tipo de proxy, tendo você que ou editar a sua regra de redirecionamento ou desativar o proxy por alguns minutos enquanto você atualiza o anti-vírus;
NAT - O NAT não é só muito útil, como é uma necessidade, tornando-se impossível o ato de construir um gateway para a internet sem o uso do mesmo. Lembrem-se que o Squid não trabalha com todos os protocolos. Sendo assim, sem NAT, sem SMTP, POP e IMAP. O NAT também é ótimo para uma "situação temporária" caso o seu servidor proxy esteja em manutenção, intermitente, em fase de implementação ou morto. As desvantagens do NAT é que ele não oferece nenhum controle do que está sendo visualizado pelos seus clientes na internet e tampouco um serviço de cache.

Outros servidores proxy para GNU/Linux e Microsoft Windows

É claro que o Squid não é o único servidor Proxy para GNU/Linux. Aqui eu irei listar outros conhecidos tanto para GNU/Linux quanto para Microsoft Windows.

Delegate - É um proxy de autoria de Yutaka Sato, suportando uma gama de protocolos. É um projeto mais recente que o Squid, mas que vem evoluindo rapidamente. Uma desvantagem é que não há tanta documentação para a instalação. O site oficial é http://www.delegate.org
OOPS! - É um proxy que quebra alguns dogmas da informática, como a utilização de arquivos grandes como arquivos de cache, ao invés de "um site, um arquivo". Olhando a sua configuração, eu cheguei à conclusão que ela é um pouco parecida com a configuração do BIND. É um projeto interessante pelo qual eu vou buscar mais informações assim que eu tiver um tempo livre. O site oficial é http://zipper.paco.net/igor/oops.eng

Internet Connection Sharing (ICS) - É o proxy oficial da Microsoft e que é utilizado nos sistemas operacionais Microsoft Windows. Possui uma facilidade em sua configuração e utilização, porém não possui opções de configuração e vários bugs que prejudicam sua segurança e funcionamento. Às vezes acontecem problemas cuja causa você simplesmente não sabe qual é.

Instalação do Squid

Neste capítulo nós vamos descrever a instalação do Squid de diversas maneiras, para todos os gostos e distribuições (Ou pelo menos tentar).

Instalando via APT

Como eu sou um usuário Debian, eu vou explicar primeiro o jeito mais fácil.

Partindo do princípio que você já tenha o seu APT e seus repositórios de preferência no sources.list, você pode simplesmente rodar este comando.

```
# apt-get install squid
```

Todos os pacotes que o Squid necessita para rodar serão puxados também, para eliminar quaisquer dependências. Caso você esteja com dúvidas em relação ao APT, faça uma pesquisa no Google e procure pelo termo "sources.list" ou ainda execute o seguinte software e configure alguns repositórios já disponíveis.

```
# apt-setup
```

Pronto. O Squid está instalado.

Instalação via pacotes .deb

Esta é uma situação completamente difícil de acontecer (Praticamente impossível), mas caso você queira instalar o Squid por seus pacotes no formato .deb, execute o seguinte comando:

```
# dpkg -i squid-common-x.y.z.deb
# dpkg -i squid-x.y.z.deb
```

Onde x.y.z é a versão do Squid.

Notem que eu primeiro instalei o pacote "squid-common" para satisfazer dependências

Instalação via pacotes .rpm

Este tipo de instalação é próprio para as pessoas que utilizam distribuições baseadas em Red Hat (Conectiva, Mandrake, SuSE e por aí vai).

```
# rpm -ivh squid-x.y.z.rpm
```

Instalação via código-fonte

Compilar um software pelo source de vez em quando é interessante, principalmente quando as suas necessidades são muito específicas. Talvez você queira compilar o Squid se você precisa de controle por MAC address, o que é interessante se você possui DHCP na sua rede e não possui um proxy autenticado, então o controle deve ser feito por MAC address ou setando sempre os mesmos IP's para as máquinas que possuem uma regra no Squid, o que não é lá muito conveniente...

Bom, puxado o source do Squid, é assim que se instala:

```
# cp squid-x.y.z-src.tar.gz /usr/src/
# cd /usr/src
# tar xvfz squid-x.y.z-src.tar.gz
# cd squid-x.y.z
# ./configure
# make
# make install
```

Para configurar o Squid de uma forma quase que completa, pode-se utilizar o configure com os seguintes parâmetros:

```
# ./configure  \
--prefix=/usr \
--exec_prefix=/usr/sbin \
--bindir=/usr/sbin \
--sbindir=/usr/sbin \
--libexecdir=/usr/lib/squid \
--sysconfdir=/etc/squid \
--localstatedir=/var/spool/squid \
--datadir=/usr/share/squid \
--enable-async-io \
--enable-storeio=coss,ufs,aufs,diskd,null \
--enable-linux-netfilter \
--enable-arp-acl \
--enable-removal-policies=lru,heap \
--enable-snmp \
--enable-delay-pools \
--enable-htcp \
--enable-poll \
--enable-cache-digests \
--enable-underscores \
--enable-referer-log \
--enable-useragent-log \
--enable-auth=basic,digest,ntlm \
--enable-carp \
--enable-diskd \
--enable-icmp \
--enable-ssl \
--enable-default-err-language=Portuguese \
--enable-tproxy \
--with-pthreads \
--with-large-files
```

Caso você queira saber quais as opções que o Squid suporta, leia o arquivo "configure" ou o README que acompanha o source.

Instalação em FreeBSD

Se você instalou o diretório de ports do FreeBSD, a instalação será simples, bastando utilizar os comandos abaixo:

```
# cd /usr/ports/www/squid25/
# make
# make all install
```

Ou por meio de um pacote pré-compilado:

```
# mount /cdrom
# mkdir /usr/ports/distfiles
# cp /cdrom/packages/All/squid-x.y.z.tgz
# /usr/ports/distfiles
# pkg_add -v /usr/ports/distfiles/squid-x.y.z.tgz
```

Instalação em OpenBSD

Existem duas maneiras de se instalar o squid no OpenBSD, a primeira é instalar o pacote já compilado

1) Baixe o pacote já compilado no site do OpenBSD (http://www.openbsd.org) e depois:

```
# pkg_add squid-x.y.z.tgz
```

2)Pelo ports

```
cd /usr/ports/www/squid
make show=FLAVORS
```

Ele irá mostrar as seguintes opções para você:

```
transparent snmp
```

Escolha qual você quer utilizar e: (se quiser apenas um squid normal, não especifique nada)

```
env FLAVOR="transparent" make
env FLAVOR="transparent" make install
make clean
```

Instalação em Microsoft Windows 2000

Baixe o setup do site do CYGWIN (http://www.cygwin.com) e selecione o pacote do Squid durante a instalação. Proceda como faria qualquer instalação em plataforma Microsoft (Next, Next, Finish).

O arquivo squid.conf

O arquivo squid.conf é o principal arquivo de configuração do Squid. Ele zela pela simplicidade das tags, mas não muito pelo tamanho. Isto porque ele é em si um manual das configurações disponíveis. Ele possui cerca de 2 mil linhas.

Limpando o arquivo squid.conf

Pode parecer uma futilidade para a maioria dos administradores, mas é recomendável uma limpeza no squid.conf, pois ele vem com muitas opções e comentários (que são explicações das tags) e que geralmente não têm tanta utilidade se você possuir um guia ao seu lado ou simplesmente sabe o que está fazendo. Por outro lado, não é recomendável que os novatos façam isso, mesmo para alcançarem um conhecimento melhor das demais tags.

Antes de fazer a limpeza, é recomendável que você faça um backup do seu arquivo original, em caso quaisquer problemas:

```
# cp /etc/squid/squid.conf /etc/squid/squid.conf.defaults
```

Depois limpe executando o seguinte comando:

```
# egrep -v "^#|^$" squid.conf.defaults > squid.conf
```

Algumas das tags mais importantes

Aqui eu irei descrever as tags mais básicas para o funcionamento do Squid. Aqui ainda não serão citadas as ACL's. Quero lembrar que um proxy que apenas sejam configuradas estas tags ainda não é um proxy funcional.

http_port

```
Padrão: http_port 3128
```

Este parâmetro define a porta em que o serviço Squid irá escutar por requisições. Por padrão esta opção estará comentada, não necessitando ser descomentada caso não desejando alterá-la.

cache_mem

```
Padrão: cache_mem 8 MB
```

Embora pareça tentador imaginar que essa seja a quantidadede memória que o processo do Squid poderá utilizar, tal pensamento está errado. Este parâmetro configura a quantidade de memória utilizada para cache e objetos em trânsito, e não a quantidade de memória reservada para o Squid.

cache_dir

```
Padrão: cache_dir ufs /var/spool/squid 100 16 256
```

Nesta opção são configurados os números de diretórios, subdiretórios e tamanho do cache. Desfragmentando a linha para estudo, ficaria assim:

cache_dir - Nome da tag;
ufs - É a forma de armazenamento de cache. Existe também a opção aufs, que usa posix threads, o UFS é recomendado até 5 req/s, acima disto voce pode usar um dos sistemas a seguinte:

AUFS, DISKD, COSS

/var/spool/squid - Diretório onde o cache do Squid ficará;
100 - Espaço em disco que o cache do Squid poderá ocupar, contado em MB;
16 - Quantidade de diretórios que o cache do Squid possuirá; (dica, cuidado para nao usar um valor muito grande e aumentar o seek time do hd)
256 - Quantidade de subdiretórios que o cache do Squid possuirá; (dica, não altere este valor, 256 é um valor já otimizado)

cache_access_log

```
Padrão: cache_access_log /var/log/squid/access.log
```

Define o arquivo de log de acessos do Squid. Caso queira saber quem acessou determinada página da internet, é através deste arquivo que descobrirá.

cache_mgr

```
Padrão: cache_mgr email
```

Este parâmetro tem a finalidade de especificar o e-mail do administrador do proxy. Caso o serviço Squid venha a ser terminado de forma anormal, o usuário do correio eletrônico especificado será alertado através de um e-mail. Este e-mail também é informado quando alguma página de erro é mostrada ao usuário cliente.

cache_effective_user

```
Padrão: cache_effective_user squid
```

Informa ao Squid com qual nome de usuário ele deve rodar.

É recomendável não utilizar qualquer serviço rodando como o usuário root, uma vez que se o servidor seja invadido utilizando brechas do serviço em questão, o invador possuirá privilégios de root no sistema.

Nas últimas versões do Squid foi criado um bloqueio que impede que o mesmo seja executado como root.

cache_effective_group

```
Padrão: cache_effective_group squid
```

Tem a mesma função da tag acima, mas ao invés de trabalhar com o usuário do Squid, ele vai trabalhar com o grupo.

visible_hostname

```
Padrão: visible_hostname none
```

Tenho lido muito na internet problemas relacionados à esta tag. Ela é que define o hostname que fica visível nas mensagens de erro do Squid apresentadas para os clientes e, caso não seja setada, o Squid não starta. Por isso coloque alguma coisa nela parecida com isto:

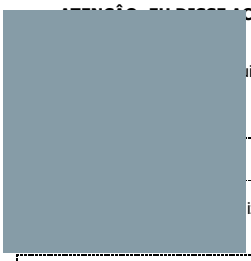
```
visible_hostname squid.seudominio.com.br
```

Access Control Lists (Também conhecidas como ACL's)

Esta é a melhor parte (E consequentemente a que vai te dar mais dor de cabeça para entender) do Squid, o uso das ACL's.

Uma ACL nada mais é do que a ferramenta que o Squid utiliza para especificar quem pode, quem não pode, o quê pode e o que não pode.

ATENÇÃO, EU DISSE ACIMA A PALAVRA "ESPECIFICA", NÃO "PROÍBE". A ACL ESPECIFICA UMA INFORMAÇÃO PARA SER BLOQUEADA POR MEIO DO HTTP_ACCESS, COMO VEREMOS MAIS À FRENTE.



uita coisa legal com as ACL's.

```

```

izada para especificar um determinado host ou uma determinada rede de origem.

```
dst
```

Endereço IP de destino. Utilizada para especificar um determinado host ou uma determinada rede de destino.

```
dstdomain
```

Domínio de destino. Utilizado para restringir acesso à um determinado ou para identificar um domínio de destino.

```
time
```

Hora e dia da semana. Especifica um determinado horário.

```
port
```

Número da porta de destino, usado para especificar acesso à determinada porta de um servidor.

```
url_regex
```

Utilizado para comparar uma string à uma URL inteira. Muito utilizado para fazer o bloqueio de sites indevidos.

```
urlpath_regex
```

Tem uma função semelhante à anterior, porém procura apenas em pedaços do caminho da URL. Muito utilizado para bloquear extensões, como veremos mais à frente.

```
proto
```

Especifica um protocolo de transferência.

```
proxy_auth
```

Somente utilizada caso você esteja utilizando autenticação. Serve para especificar nomes de usuários.

Definindo ACL's

Dentro do arquivo de configuração do Squid, o squid.conf, você vai encontrar uma área que é a mais ideal para declarar as suas ACL's. Este espaço é onde as ACL's começam a ser definidas, facilmente identificada pela presença das mesmas.

Para declarar ACL's, a sintaxe básica é a seguinte:

```
acl <nome da acl> <tipo da acl> <string>|"<endereço de arquivo>"
```

Um exemplo prático de ACL:

```
acl palavra_proibida url_regex -i sexo
```

A ACL acima bloqueia todos os sites que contenham em seu endereço a palavra "sexo".

Outros exemplos serão dados mais à frente, dependendo do tipo da ACL.

A tag http_access

É simplesmente inútil o uso de ACL's sem o uso da tag http_access. É ela que trava ou libera o que a ACL está estipulando.

Exemplo:

```
http_access deny proibido
```

Se nós considerarmos o conjunto ACL + http_access, ficaria:

```
acl proibido url_regex -i sexo
http_access deny proibido
```

O que o conjunto acima faz é proibir que qualquer site que possua em seu endereço a palavra "sexo" seja exibido para o requisitante.

Nem sempre você vai utilizar um http_access para cada ACL que você fizer, às vezes você pode combinar duas ACL's em um único http_access, como nós veremos mais à frente.

Ordem das ACL's

Esta é uma outra dúvida que é comum entre os iniciantes:

"Qual a ordem que devo utilizar as minhas ACL's? Porque eu estou colocando tudo certinho e não está funcionando!".

Bem, a resposta para essa pergunta não é tão simples assim. Eis o que o Squid faz quando vai tratar das ACL's.

1 - O Squid vai ler todas as ACL's e testar se todas as ACL's declaradas possuem uma sintaxe correta e se elas estão sendo referenciadas por algum http_access;
2 - Depois disso, se ele iniciar normalmente (Pode ser que outros fatores impeçam isto), ele irá começar a testar todas as requisições que são feitas para ele e tentar casar as mesmas com as regras que as ACL's estipulam em conjunto com os http_access;
3 - Caso uma URL case com uma ACL, ele ignorará todas as outras ACL's para aquela requisição.

Uma outra maneira mais prática de tentar implementar isso é fazer da seguinte maneira:

1 - Primeiro coloque as ACL's que estipulam uma excessão à alguma regra de bloqueio que virá à seguir;
2 - Depois coloque as suas ACL's que vão bloquear sites e tudo o mais;
3 - Só então você coloca as suas ACL's liberando o acesso.

ATENÇÂO: ENTENDA QUE A ORDEM DAS ACL'S E DOS HTTP_ACCESS NÃO PRECISA SER A MESMA, MAS É INTERESSANTE MANTER ALGUMA ORGANIZAÇÃO!

Casos

Bom, aqui eu vou citar alguns casos e alguns modos principais para configurar o seu Squid. É nesse capítulo que nós vamos entrar mais a fundo na prática do Squid.

Habilitando o Squid

Essa é a coisa mais fácil de se fazer com o Squid: botar ele para funcionar sem filtro nenhum. Claro que se você optou por fazer isso, é mais fácil que você utilize um NAT.

Para botar o Squid para rodar, basta encontrar a linha:

```
http_access deny all
```

Para:

```
http_access allow all
```

E depois reinicie o serviço:

```
# /etc/init.d/squid restart
```

Talvez ele dê algum problema relacionado à tag **visible_hostname**, mas é só corrigi-la conforme o capítulo 3.2.8.

Restringindo o acesso ao Squid

Se nós quisermos que somente uma rede da nossa empresa acesse o proxy, nós podemos definir faixas de IP's ou redes inteiras para a utilização do proxy. As opções de relay serão tratadas na seção "5.11. Segurança em um servidor Squid".

Quando você encontrar a linha **http_access allow all**, ela vai estar liberando acesso à todos os hosts, já que a ACL **"all"** está especificando todos os hosts.

Para arrumar isto, você deve encontrar e comentar as linhas:

```
acl all src 0.0.0.0/0.0.0.0
http_access allow all
```

Agora crie uma nova ACL do tipo **"src"**, especificando a rede interna:

```
acl redeinterna src 192.168.0.0/24
```

Agora autorize a ACL que você acabou de criar por meio de um http_access:

```
http_access allow redeinterna
```

Como visto acima, nós estamos somente permitindo o uso ao proxy pela rede interna. Agora, caso você queira especificar uma range de IP's, faça assim:

```
acl faixa_adm src 192.168.0.10-192.168.0.50
http_access allow faixa_adm
```

Pronto, está tudo certo. Só ressalto a importância que você deve dar para a ordem das ACL's.

Bloqueando sites indevidos no proxy

Bem, este exemplo já foi citado lá em cima, mas de qualquer jeito, vamos abrangê-lo um pouco mais.

O tipo de ACL **url_regex** serve para nós compararmos termos dentro de uma URL para que possamos compará-la e saber se esta palavra está ou não liberada e se os usuários vão ou não, visualizar a página. Nós utilizamos muito isso quando desejamos que sites pornográficos não sejam vistos por usuários dentro de uma empresa.

Vamos ao exemplo:

Adicione a seguinte ACL:

```
acl palavra url_regex -i sex
```

Agora bloqueie o acesso com o http_access:

```
http_access deny palavra
```

Com este exemplo, todos os sites que possuam a palavra **"sex"** em seu endereço não serão visualizados pelos usuários. O parâmetro **"-i"** serve para que o Squid não distingua entre maiúsculas ou minúsculas.

Porém neste momento você deve estar pensando que não é prático definir uma ACL para cada palavra que você deseje bloquear. E realmente não é. Por isso nós vamos declarar listas inteiras de palavras negadas.

Eis o exemplo:

Primeiro nós vamos criar o arquivo texto que nos vai servir como lista de palavras bloqueadas e damos à ela permissões de leitura:

```
# touch /etc/squid/lists/blocked
# chmod 755 /etc/squid/lists/blocked
```

Nele insira todas as palavras proibidas. Lembre-se que você deve adicionar uma palavra por linha.

Após isto, nós criamos a ACL da seguinte maneira:

```
acl blocked url_regex -i "/etc/squid/lists/blocked"
```

E bloqueamos o acesso com o http_access:

```
http_access deny blocked
```

Feito isto, todos os sites que contém em seu endereço palavras como "sex", "sexo", "sexologia" vão ser bloqueados, porém como visto no último exemplo, nem todos os sites que contém a palavra "sex" é um site de sacanagem. Mas como distinguir os sites que podem ser liberados dos que não podem? Simples, vamos criar uma outra lista para as excessões como "sexologia".

Vamos seguir o mesmo procedimento. Criamos uma lista também para as palavras não bloqueadas.

```
# touch /etc/squid/lists/unblocked
# chmod 755 /etc/squid/lists/unblocked
```

Nesta lista nós também vamos colocar todos os sites que são excessões à regra, como "www.sexologia.org". Também um site por linha. Se você preferir colocar as palavras somente, tudo bem também.

Depois você vai ter que juntar as duas ACL's em um único http_access, desta maneira:

```
http_access deny blocked !unblocked
```

Note a utilização do sinal de exclamação, significando uma inversão no sentido da regra. Vá se acostumando com este sinal, porque ele é extremamente útil no seu dia-a-dia.

Pronto, o procedimento está feito. Agora tudo o que você deve fazer é reiniciar o serviço do Squid:

```
# /etc/init.d/squid restart
```

Configurando o Squid para solicitação de autenticação

Este é um caso muito interessante e muito requisitado também pelos clientes. A solicitação de autenticação para os usuários.

Quanto nós configuramos o Squid desta maneira, o mesmo solicita autenticação sempre que algum usuário abrir o navegador e só libera o uso da internet mediante uma autenticação bem-sucedida. É uma implementação muito interessante em ambientes acadêmicos ou corporativos.

O Squid por si só não efetua a autenticação sozinho, necessitando da ação de módulos adicionais de autenticação, sendo o mais comum deles o **ncsa_auth** e o **smb_auth**, porém existem outros módulos para autenticação em outros tipos de bases de autenticação, como LDAP e MySQL.

Infelizmente, autenticação e proxy transparente não combinam e não podem viver em conjunto, pois os pacotes são corrompidos no redirecionamento de portas. Eu ensinarei meios para que você force os seus usuários à passarem pela autenticação no capítulo "5.12. Segurança em um servidor Squid". Por isso, eu sugiro a sua leitura logo após este capítulo.

Sempre que o usuário abrir e fechar o navegador, ele vai ter que se autenticar novamente. Algumas pessoas vêem isto como algo negativo, por ser inconveniente para o usuário ter que se autenticar toda hora. Por outro lado é uma questão de segurança também. Imagine que o usuário X sai para o almoço e deixa seu computador ligado, nisso o usuário Y, que está disputando a vaga de gerente com ele, vai até sua mesa e abre meia dúzia de sites de sacanagem. Ponto-final para a promoção do usuário X.

O módulo de autenticação ncsa_auth

Para utilizar o módulo de autenticação **ncsa_auth**, execute os seguintes passos:

Primeiro localize o módulo ncsa_auth:

```
# find /usr -name ncsa_auth
```

Na época de publicação desta versão do guia, o módulos de autenticação do Squid localizavam-se em **/usr/lib/squid/** em distribuições Debian e baseadas nela.

Após encontrado, crie um link simbólico para o **/usr/bin**:

```
# ln -s /usr/lib/squid/ncsa_auth /usr/bin/ncsa_auth
```

NOTA: Eu só crio este link porque eu tenho preguiça de ficar digitando o caminho inteiro do binário do ncsa_auth.

Agora nós teremos que editar o squid.conf e alterar algumas linhas nele. São elas:

```
auth_param basic program
auth_param basic realm
auth_param basic children
acl autenticacao proxy_auth REQUIRED
http_access allow autenticacao
```

Elas servem respectivamente para:

Determinar o módulo que vai ser responsável pela autenticação e o arquivo de senhas a ser utilizado;
Determinar a mensagem que irá ser colocada na janela desolicicação de autenticação;
Determinar o número de processos de autenticação que serão iniciados;
Forçar os usuários à se autenticarem.

Eis o que você deve colocar nas tags:

```
auth_param basic program
```

Nesta tag você pode colocar o seguinte valor:

```
auth_param basic program /usr/bin/ncsa_auth /etc/squid/squidpasswd
```

```
auth_param basic realm
```

Aqui você pode colocar algo como:

```
auth_param basic realm Digite seu login e senha:
```

```
auth_param basic children
```

Bem, aqui você deve colocar um valor sábio referente ao número de usuários que você tem na sua rede. Eu gosto de pegar o número de usuários que eu tenho na rede e dividir por dois.

```
acl autenticacao proxy_auth REQUIRED
```

Para os que pensaram que iriam escapar das ACL's aqui, SE ENGANARAM! Esta ACL especifica o uso da autenticação em si.

```
http_access allow autenticacao
```

Bem, como toda outra ACL, você precisa defini-la e determinar uma ação para ela utilizando-se de um http_access. Esta é a razão de ter um aqui também. Este http_access serve para forçar que todos os usuários se autentiquem para conseguirem acesso à internet.

Só saliento novamente a importância em prestar prestar muita atenção na ordem na hora de declarar as suas ACL's e seus http_access.

Feito isto, toda a parte mais difícil já está feita. Agora as únicas coisas que você a fazer é cadastrar os seus usuários com o comando **htpasswd** e reiniciar o serviço do Squid.

```
# htpasswd -c /etc/squid/squidpasswd fulano
```

O parâmetro **"-c"** é somente necessário na primeira execução, para a criação do arquivo de senhas, sendo dispensável nas próximas execuções do programa.

ATENÇÂO: Alguns usuários vieram me reclamar que o pacote Squid da distribuição deles não contém o htpasswd, nem em pacote separado e muito menos no pacote do Squid. Eu constatei este problema nas distribuições: Mandrake, Conectiva e Gentoo. No caso do Mandrake e do Gentoo, o problema pôde ser contornado instalando o pacote httpd-utils ou algum outro pacote referente ao Apache (Para os que não sabem, as autenticações de diretório do Apache e o ncsa_auth utilizam a mesma criptografia).

Feito isto, você pode reiniciar o serviço do Squid com o comando:

```
# /etc/init.d/squid restart
```

E da próxima vez que o usuário abrir o navegador, a senha irá ser requisitada.

O módulo de autenticação smb_auth

Esta é uma opção muito interessante também, caso você possua um PDC - Primary Domain Controler (No caso, o Samba) - na sua rede.

Aqui eu não irei tratar da configuração do Samba e estou considerando que o mesmo foi configurado corretamente.

Uma das principais vantagens deste método é que você irá utilizar a mesma base de dados de usuários do Samba para autenticar os seus usuários do Squid. Com isto você ganha uma centralização à mais dos seus usuários e menos dor de cabeça com senhas esquecidas, já que seus usuários só terão que decorar uma senha, sem falar que o usuário pode utilizar o próprio Windows para alterar a sua senha do Squid, e ele irá alterar todas as senhas ao mesmo tempo, não precisando de utilitários como o chpasswd (Que nós iremos ver mais à frente).

O modo que ele funciona é o seguinte:

1 - O usuário abre o navegador e a janela solicitando autenticação é apresentada à ele;
2 - Ele fornece o login e senha;
3 - O módulo de autenticação smb_auth loga-se no servidor Samba com o login e senha fornecidos e tenta ler o arquivo **proxyauth** (Que nós iremos ver mais à frente);
4 - Caso ele consiga ler o arquivo e o valor retornado seja "allow", ele libera o acesso à internet. Caso contrário, ele não permite o acesso do usuário.

ATENÇÃO: A utilização do smb_auth ao invés do ncsa_auth não faz com que o usuário ao mesmo tempo que se loga na rede do Microsoft Windows, logue-se também no proxy e nem faz com que o usuário não precise se autenticar novamente se ele fechar e abrir o proxy.

Bom, vamos lá. Primeiramente você precisa extrair o smb_auth:

```
# tar xvfz smb_auth-x.y.z.tar.gz
# cd smb_auth-x.y.z
```

No Makefile, edite os parâmetros **"SAMBAPREFIX"** e **"INSTALLBIN"** para os diretórios onde você respectivamente instalou o Samba e o diretório onde você quer instalar o smb_auth.

NOTA PARA USUÁRIOS DEBIAN: O pacote do Squid do APT já vem com o smb_auth compilado e configurado corretamente, dispensando estes primeiros passos.

Em seguida, compile o binário com os comandos:

```
# make
# make install
```

Então crie o arquivo **"proxyauth"** no seu diretório **"netlogon"** (Como eu não sei onde você criou o seu diretório netlogon, o problema é SEU!):

```
# echo allow > /home/netlogon/proxyauth
```

ATENÇÃO: Este arquivo deve estar no formato de texto do MS-DOS. Sendo assim, você precisa de um aplicativo como o "unix2dos" ou o **"flip"** para convertê-lo ou ainda usar o próprio VIM e o uso do comando **":set ff=dos".**

Dê permissão de leitura e execução para ele:

```
# chmod 755 /home/netlogon/proxyauth
```

Caso você tenha setado no parâmetro **"INSTALLBIN"** um diretório que fique longe, crie um link, assim como nós fizemos no caso do ncsa_auth:

```
# ln -s /usr/local/bin/smb_auth /bin/smb_auth
```

Bem, a parte da instalação do smb_auth está terminada por aqui. Agora vem a parte da configuração dentro do Squid.

Primeiramente, encontre a linha **"auth_param basic program"** e deixe-a da seguinte maneira:

```
auth_param basic program /bin/smb_auth -W dominiodosamba -U 192.168.1.1
```

NOTA: As linhas acima são uma só.

Traduzindo, os parâmetros **"-W"** e **"-U"** servem, respectivamente, para definir o domínio do Samba e o IP do servidor Samba. É interessante especificar o IP por dois motivos: Primeiro porque caso o primeiro método falhe, o segundo é tentado e porque gera menos broadcasting na sua rede.

Configure também as linhas **"auth_param basic realm"** e **"auth_param basic children"** do seu squid.conf.

Agora insira a ACL para forçar a autenticação, juntamente com o seu http_access:

```
acl autenticacao proxy_auth REQUIRED
http_access allow autenticacao
```

NOTA: O smb_auth necessita dos programas clientes do Samba (smbclient, smbmount, etc) para funcionar.

Caso você tenha problemas utilizando o smb_auth, faça alguns testes com o modo debug dele, utilizando o parâmetro **"-d".**

Pronto, está funcionando!

Utilização do chpasswd

O chpasswd é um artifício muito interessante para quem utiliza proxy com autenticação. Ele permite, por meio de scripts no formato cgi-bin, que os seus usuários possam trocar a própria senha no proxy (Caso você não esteja utilizando o módulo de autenticação smb_auth).

O chpasswd pode ser puxado no mesmo site do SARG, já que ambos são desenvolvidos pela mesma pessoa.

Um requerimento básico para a utilização do chpasswd é possuir o Apache instalado. Novamente eu não irei detalhar a configuração do Apache e estou presumindo que o mesmo já está configurado e funcionando perfeitamente.

Eis os passos para a instalação:

```
# tar xvfz chpasswd-x.y.z.tar.gz
# cd chpasswd-x-y-z
# ./configure --prefix=/usr/local/etc/chpasswd \
--enable-language=English --enable-cgi-dir=/var/www/html/cgi-bin
# make
# make install
```

NOTA: Na linha "./configure..." as duas seguintes são uma só.

Descrição dos parâmetros:

--prefix - Diretório onde os arquivos de configuração se localizam;
--enable-language - Língua que irá ser utilizada;
--enable-cgi-dir - Diretório onde estarão contidos os cgi-bin's do Apache.

Feito isto, entre no diretório onde você especificou onde as configurações ficarão. Lá edite o **chpasswd.conf** para as configurações que melhor lhe convém. O arquivo de configuração é bem simples e auto-explicativo, por isto a configuração não será abordada aqui.

Em seguida, copie os compilados para a sua pasta de cgi-bin's do Apache (isto se já não tiverem sido copiados na hora do "make install") e pronto!

Restringindo o horário de acesso

Aqui é onde nós citamos outro tipo interessante de controle que nós podemos implementar em um escritório ou em um meio acadêmico: o controle por horário.

Para nós fazermos isto, nós fazemos o uso da ACL do tipo **time**.

Exemplo:

```
acl horariopermitido time MTWHF 08:00-18:00
http_access deny !horariopermitido
```

Novamente o sinal de exclamação aparece, invertendo o sentido da regra. Interpretando a regra fica assim: Ele vai negar o uso do proxy em todos os horários, **COM EXCESSÃO** do horário especificado na ACL horariopermitido.

Você deve estar estranhando o **"MTWHF"** na frente da ACL. Ela especifica os dias da semana conforme esta tabela:

S - Sunday (Domingo)
M - Monday (Segunda)
T - Tuesday (Terça)
W - Wednesday (Quarta)
H - Thursday (Quinta)
F - Friday (Sexta)
A - Saturday (Sábado)

Viu? Não é difícil, mas uma noção de inglês é imprescindível para qualquer um...

O chefe pentelho

Acontece com todos. Este de fato é um dos casos mais chatos que acontecem em meio de trabalho. O seu chefe adora proibir as coisas para os usuários mortais, mas ele quer ter o controle completo.

Para contornar a situação, faça o seguinte:

Caso você utilize proxy convencional, sem autenticação:

```
acl fidaputa src 192.168.1.23
```

E permitir a lista de palavras negadas para ele, assim:

```
http_access allow blocked fidaputa
```

Caso você utilize proxy autenticado:

```
acl fidaputa proxy_auth carlos
```

E permitir a lista de palavra negadas para ele, assim:

```
http_access allow blocked fidaputa
```

ATENÇÃO: Pela nossa senhora do capacete vermelho, atente pela ordem das suas regras!!!

Uma outra maneira de se fazer ambos os bloqueios é definir o IP/login do seu chefe em uma ACL e na hora em que for declarar o http_access, definir acima das ACL's de bloqueio, assim:

```
http_access allow fidaputa
http_access deny blocked !unblocked
```

Peãozada só vê site de banco

Este exemplo também é muito interessante e é bem similar ao exemplo acima, só que ao invés de especificar os sites negados, você vai especificar o site que ele vai poder acessar.

Este exemplo também pode ser adaptado para outros sites, não somente para o site de bancos e caso sejam muitos sites, você pode especificá-los em uma lista também.

Caso você utilize proxy convencional, sem autenticação, faça assim:

Primeiramente, especifique os sites que eles irão poder acessar:

```
acl bancos url_regex -i "/etc/squid/lists/bancos"
```

Adicione os endereços dos bancos na lista e especifique também o IP do computador do usuário:

```
acl peao src 192.168.1.24
```

Então junte os dois em um único http_access, desta maneira:

```
http_access deny !bancos peao
```

Interpretando a regra, fica: Vamos bloquear todos os sites, **COM EXCESSÃO DOS SITES ESPECIFICADOS NA LISTA DE BANCOS** para o IP 192.168.1.24.

Caso você utilize proxy autenticado, o pensamento é praticamente o mesmo, mas como é sempre bom praticar, vamos explicar todo o procedimento novamente:

Primeiramente, especifique o login do usuário por meio de uma ACL:

```
acl piao proxy_auth josefa
```

Depois especifique os endereços dos sites que você quer que a Josefa tenha acesso, desta maneira:

```
acl bancos url_regex -i "/etc/squid/lists/bancos"
```

Insira os endereços que você deseja que a Josefa acesse na lista e não se esqueça de dar as permissões de leitura para o arquivo, hein?

Depois, junte as duas ACL's com um único http_access:

```
http_access deny !bancos piao
```

ATENÇÃO DE NOVO!!! Pela minha nossa senhora do capacete vermelho, presta atenção na ordem das ACL's!!!

Softwares da Caixa (Conectividade Social)

Hummm, está aí uma coisa que dá muita dor de cabeça para a maioria dos administradores de redes que eu encontrei por aí e eu mesmo trombei com este problema.

O negócio é o seguinte. Vamos conhecer o problema: O Conectividade Social é um software da Caixa Econômica Federal que utiliza para a transmissão de dados um protocolo de criptografia que não passa por proxies, havendo uma corrupção do pacote.

A manha para contornar o problema é editar a sua regra de IPTables para o proxy transparente, fazendo com que todos os pacotes, tirando esses com destino à Caixa Econômica Federal, sejam redirecionados para o Squid. Então esses pacotes irão passar apenas pelo NAT.

Para isto, você pode editar a sua regra de proxy transparente e deixá-la deste modo:

```
# iptables -t nat -A PREROUTING -d ! 200.201.174.0/24 -p tcp --dport 80 -j REDIRECT  --to-port 3128
```

NOTA: Você pode estar um pouco perdido com relação à regra acima, mas não fique, tudo isso será esclarecido no capítulo "7.6.2. Regras de IPTables".

No meu caso esta regra resolveu meu problema completamente, mas caso você tenha problemas relacionados à isto, você pode me escrever ou dar uma pesquisada no Google, que os resultados serão satisfatórios.

Bloqueando MSN

Bem, aqui está outro problema pelo qual alguns administradores de redes possam ter dificuldades. O MSN a partir da versão 6.x (in)felizmente passou a fazer conexões utilizando tunneling por http. Ou seja: Ele se conecta pela porta 80 também!

Sendo assim, é muito difícil fazer o bloqueio do MSN se não utilizados softwares como o layer-7 ou o Squid, que tratam as requisições vindas pela porta 80. O layer-7 é um software muito interessante também, trabalhando na camada de aplicação da tabela OSI (Para quem realmente é administrador de rede, vai entender do que eu estou falando), por isso seu nome.

O bloqueio do Squid é simples de ser feito, bastando adicionar o termo "gateway.dll" na sua lista de palavras negadas.

Bloqueando extensões e downloads de determinados

Mais um exemplo interessante. Eu acho incrível o número de pessoas que nos dias atuais se infectam com vírus contidos em arquivos de extensões conhecidas por conter programas maliciosos, como o .bat e o .pif.

Mas por incrível que pareça, você pode bloquear extensões que os seus usuários baixam no computador por meio de HTTP ou de FTP e de quaisquer outros protocolos que o Squid suporte, fazendo os seguintes passos:

Primeiramente, você deve criar a velha listinha e setar as permissões corretas:

```
# touch /etc/squid/lists/extensoes
# chmod 755 /etc/squid/lists/extensoes
```

Feito isto, você deve escrever as extensões que você quer bloquear da seguinte maneira no arquivo:

```
\.mp3$
\.wav$
\.pif$
\.bat$
```


NOTA: O "\" é um eliminador de metacaracteres e serve para cancelar a função do ".". Já o "$" serve para que seja analizado até o final de string.

Agora nós vamos adicionar a ACL no Squid que vai bloquear as extensões efetivamente, juntamente com o seu http_access:

```
acl extensoes urlpath_regex -i "/etc/squid/lists/extensoes"
http_access deny extensoes
```

ATENÇÃO: PELA MINHA NOSSA SENHORA DO CAPACETE VERMELHO, PRESTA ATENÇÃO NA (*&%$&()(DA ORDEM DAS SUAS ACL'S!!!

Note que ao invés do **url_regex**, eu utilizei o **urlpath_regex**. Caso você precise de mais informações sobre isso, consulte a seção "4.1. Tipos de ACL's".

Alterando configurações sem reiniciar o serviço

Bom, para os apressadinhos, talvez isso ajude em algo e talvez você, caro leitor, fique com raiva de mim agora.

Acontece que o Squid, assim como os softwares mais atuais que trabalham como daemons, conta com um método de "recarregamento das regras sem a reinicialização do serviço". Traduzindo isto quer dizer que se você muda uma coisinha no squid.conf, você não precisa reiniciar o serviço, só as regras.

Antes, você fazia isso:

```
# /etc/init.d/squid restart
```

Agora, você fará isso:

```
# squid -k reconfigure
```

Segurança em um servidor Squid

Todo servidor, independente de plataforma ou serviço que executa, precisa de certas configurações de segurança. Ele pode ser um servidor NFS, Samba, Apache, Squid, NIS, tanto faz, ele sempre irá precisar de um ou outro macete de segurança para que o mesmo não sofra ataques.

Ao longo dos anos em que eu mexo com GNU/Linux, eu aprendi que o mesmo é um sistema extremamente seguro, mas que não é muito à prova de burros. Computadores podem ser tanto canalizadores de soluções nas mãos das pessoas certas, quanto canalizadores de problemas nas mãos das pessoas erradas.

Neste capítulo eu irei detalhar um pouco dos principais erros de segurança que as pessoas deixam passar quando estão configurando o Squid.

Erro I: Definição de relay do servidor Squid

Assim como a maioria dos servidores, o Squid também possui suas configurações de relay. Entenda por relay o "lado para qual o servidor está rodando, se é para a internet ou para a rede interna".

Claro que é interessante que você monte um servidor para a sua rede interna. Sendo assim, você deveria se preocupar com o relay do seu servidor Squid, pois geralmente os usuários costumam disfarçar seus IP's utilizando servidores proxy abertos pela internet, prática conhecida como IP Spoofing.

Você pode detectar se existe alguém utilizando o seu proxy de duas maneiras: A primeira é deixar o seu servidor rodando e tirar o log de 1 dia e sentir um verdadeiro orgasmo com o resultado. Com certeza algo aparecerá, seja um endereço IP que não é da rede interna ou um endereço de host totalmente nada a ver. A segunda é monitorar a ações do seu proxy com o "tail", em tempo real, para quem gosta de emoções mais fortes.

Por outro lado, temos também duas maneiras de resolver este problema. Você escolhe a que melhor lhe couber.

1ª Maneira: http_port

Você pode editar esta tag para que o Squid só escute requisições vindas da rede interna, desta maneira:

Troque o valor-padrão da tag:

```
http_port 3128
```

Para:

```
http_port 192.168.1.1:3128
```

Se você resolver por aqui, o seu problema com relay morreu. Eu prefiro esta maneira por ser mais prática e por não ter que ficar mexendo com ACL's e se algum dia um abelhudo passar um portscanner na sua máquina, como o nmap, ele não irá apresentar o serviço Squid rodando.

2ª Maneira: acl all src 192.168.0.0/24

Você pode permitir o http_port como mencionado acima, mas deverá definir a sua rede interna quando for utilizar as suas ACL's, desta maneira:

Troque a ACL all de:

```
acl all src 0.0.0.0/0
```

Para:

```
acl all src 192.168.0.0/24
```

O Squid somente irá permitir conexões vindas da rede interna.

Erro II: http_access deny all

Coloque esta linha no final das declarações das suas ACL's. O motivo disto é que se o Squid não encontra uma ocasião pela qual encaixar uma ACL, ele simplesmente libera a sua utilização.

Erro III: Usuários

Legal. Muito interessante. Você sempre pensou nos ataques externos e esqueceu dos internos e esqueceu que 70% dos ataques à servidores vêm de dentro ou possuem alguma ajuda de dentro.

Isto porque os usuários conhecem o sistema e encontram mais facilmente meios de burlarem o mesmo.

Geralmente usuários removem as configurações de proxy do navegador para navegarem sem restrições e somente pelo NAT ou procuram por proxies abertos na internet e configuram os navegadores para utilizarem estes endereços ou ainda vão em páginas como o "anonymizer" ou similares e navegam por lá. Por isto, é importante que você faça o bloqueio de tais páginas.

Eu também tomo uma atitude um tanto drástica, bloqueando a porta 80 para a internet. Um contratempo disto é que alguns softwares de anti-vírus não conseguem atualizar as definições de vírus, mas é muito interessante utilizar a regra abaixo neste caso e no caso de proxy autenticado.

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

Erro IV - Proxy Transparente sem ser no gateway da rede

Também é comum eu receber e-mails de pessoas perguntando como fazer isso. Não é uma idéia muito segura, a menos que você possua uma DMZ.

Isso porque os usuários podem alterar as configurações de rede e fazer com que o gateway da rede aponte para o roteador final, fazendo um caminho alternativo para os pacotes e fazendo com que os mesmos não passem pelo servidor Squid.

Relatórios

O Squid possui vários programas externos para uma melhor interpretação do access.log, que é o arquivo onde são registrados os acessos ao proxy. Entre eles estão o Calamaris, o SARG e o Squid Graph.

Podemos também criar scripts simples com o uso de shell-scripts, filtrando os dados que nos interessam e mandá-los para o nosso e-mail.

Neste capítulo, serão abordados a utilização de scripts simples e o uso do SARG. Escolhi abordar o SARG por dois motivos:

1 - Foi feito por um brasileiro;
2 - É o melhor de todos os geradores de relatório que eu já vi.

Criando um script de relatório simples

Para a criação de um shell-script que gere o relatório que pegue as últimas requisições negadas pelo proxy, devemos criar um arquivo em branco:

```
# touch /usr/bin/squidreport
```

Com o conteúdo abaixo:

```
#!/bin/bash
cat /var/log/squid.log | grep DENIED | tail -n 60 | mail root
```

Mude as permissões do arquivo para:

```
# chmod 700 relatoriosquid.sh
```

E agende o relatório para ser gerado dentro do CRON, com a seguinte linha:

```
# crontab -e
30        21    *   *         *         squidreport
```

Pronto! Todos os dias às nove e meia da noite, será gerado o relatório de acessos negados pelo Squid e eles vão estar dentro da caixa de e-mails do administrador do sistema.

Utilizando o SARG

Desenvolvido pelo brasileiro Pedro Orso, ele gera páginas em HTML à partir do arquivo de log do Squid.

Instalação

Para instalar o SARG, você pode tanto puxá-lo do site oficial (http://sarg.sourceforge.net) ou instalá-lo através de pacotes. Caso você esteja compilando-o, execute os seguintes comandos:

```
# tar xvfz sarg-x.y.tar.gz
# cd sarg-x.y
# ./configure
# make
# make install
```

Configuração

Por padrão, o SARG é instalado em /usr/local/sarg. Neste diretório encontra-se o arquivo de configuração do SARG, o sarg.conf. Dentre várias opções, recomendo setar as seguintes:

language Portuguese
access_log /var/log/squid/access.log
title "Relatório de uso da internet"
temporary_dir /tmp
output_dir /var/www/squid-reports
resolve_ip no
user_ip no
topuser_sort_field BYTES reverse
topsites_num 100
max_elapsed 28800000

Sendo importante destacar:

access_log - Indica o arquivo de log do Squid;

output_dir - Indica onde será gerado o seu relatório. É recomendável que seja um diretório onde o seu httpd possa acessar;
resolve_ip - Habilita/desabilita a resolução de nomes;
user_ip - Se você estiver utilizando um proxy autenticado, coloque **yes**. Caso contrário coloque **no**;
topsites_num - Quantidade de sites que você deseja que apareçam como os TOP de acessos.

Gerando relatórios

Esta é a parte mais simples. Para gerar os relatórios, execute o comando:

```
# sarg
```

Assim como o shell-script que nós criamos, também podemos colocar este comando para ser executado uma vez por dia, no CRON, seguindo o mesmo procedimento.

Você também tem a opção de possuir diversos arquivos de configuração para os relatórios. O SARG lê um arquivo de configuração por padrão, quando não é apontado nenhum arquivo de configuração, mas você pode especificar outro arquivo de configuração utilizando o parâmetro **-f**, da seguinte maneira:

```
# sarg -f $ARQUIVODECONFIGURACAO
```

Dica

Os relatórios do SARG ocupam um enorme espaço em disco, principalmente por não possuir um rotate ou comprimir os HTML's gerados. Para contornar esse problema, nós podemos colocar os seguintes comandos para serem executados em um determinado horário no crontab.

```
# find /var/www/squid-reports/ -name "*.html" -type f -mtime +30 -exec bzip2 {} \
    find /var/www/squid-reports/ -name "*.bzip2" -type f -mtime +180 -exec rm -rf {} \;
```

Obtida de "http://under-linux.org/wiki/Projetos/Squid-Doc"