Faculdade de Tecnologia de Americana "Ministro Ralph Biasi"

CONTROLE DE ACESSO UTILIZANDO O SERVIDOR DE PROXY SQUID

ACCESS CONTROL USING THE SQUID PROXY SERVER

Lucas Martinelli Borelli, Fatec Americana, lucas.borelli@fatec.sp.gov.br
Clerivaldo José Roccia, Fatec Americana, clerivaldo.roccia@fatec.sp.gov.br

Resumo

O objetivo deste artigo é discutir sobre a importância de servidores de *proxy* em como item de controle de acesso, e demonstrar a maneira em que sua implementação garante maior segurança dentro de uma rede. Temas como segurança da informação, controle de acesso e servidores de *proxy*, serão abordados e definidos ao decorrer deste trabalho. Em complemento, é apresentado o cenário em que ocorre a implementação de um servidor de *proxy* Squid em um ambiente de rede interna, a fim de garantir controle do que é acessado na internet. Ao final deste artigo espera-se que seja elucidada a importância do controle de acesso e bem como é o processo de sua configuração em uma determinada rede.

Palavras-chave: Servidor de *proxy*, Controle de acesso, Segurança da informação.

Abstract

The objective of this article is to discuss the importance of proxy servers as an access control item and demonstrate the way in which their implementation guarantees greater security within the network. Topics such as information security, access control and proxy servers will be addressed and defined throughout this work. In addition, the scenario is presented in which the implementation of a Squid proxy server in an internal network environment, in order to guarantee control over what is accessed on the internet. At the end of this article, it is expected that the importance of access control will be clarified as well as the process of configuring it in a given network.

Keywords*: Proxy server, Access control, Information security.*

1. Introdução

Nos últimos anos, a crescente necessidade de segurança e controle de acesso tem se tornado uma prioridade para organizações e instituições públicas. Com a rápida expansão da internet, as empresas se deparam com um ambiente cada vez mais complexo e vulnerável a ameaças cibernéticas. Nesse contexto, o controle de acesso é uma estratégia fundamental para proteger os recursos digitais e garantir a integridade das informações.

Um dos principais desafios enfrentados pelas organizações é a implementação de soluções eficazes de controle de acesso que sejam flexíveis, escaláveis e robustas o suficiente para acompanhar as demandas em constante evolução. Nesse sentido, o servidor de *proxy* Squid é uma ferramenta poderosa e versátil para gerenciar o acesso à internet e proteger os sistemas corporativos contra ameaças externas e internas.

Squid é um servidor de *proxy* amplamente utilizado em ambientes corporativos e institucionais para controlar e monitorar o tráfego de rede. Sua capacidade de *caching*, filtragem de conteúdo e autenticação de usuários o tornam uma escolha popular para implementações de controle de acesso em larga escala.

Neste contexto, este trabalho explora a utilização do servidor de *proxy* Squid como uma solução eficaz para o controle de acesso em ambientes corporativos e educacionais. São abordados aspectos como a configuração do Squid, suas funcionalidades de filtragem de conteúdo e autenticação de usuários, visando fornecer uma visão abrangente sobre sua implementação e seus benefícios para a segurança da informação. Ao final, compreende-se a importância deste serviço para profissionais de TI que procuram fortalecer as defesas cibernéticas de suas organizações e instituições de ensino.

2. Referencial Teórico

2.1. Segurança da Informação

De acordo com Hintzbergen *et al.* (2018), a segurança da informação consiste em proteger os dados contra uma variedade de ameaças, visando assegurar a continuidade operacional, reduzir os riscos empresariais e potencializar os benefícios e oportunidades de investimento.

Segundo Peltier (2001 *apud* Mascarenhas Neto; Araújo, 2019, p. 26):

> A segurança da informação compreende o uso de controles de acesso físicos e lógicos, com o intuito de proteger os dados contra modificações acidentais ou não autorizadas, destruição, quebra de sigilo, perda ou dano aos ativos informacionais.

Como observado por Sêmola (2013), podemos também interpretá-la como a implementação de estratégias de gestão de riscos relacionados a incidentes que possam afetar os três pilares fundamentais da segurança da informação: confidencialidade, integridade e disponibilidade.

Confidencialidade: Este termo abrange dois conceitos, a confidencialidade dos dados, ou seja, garantir que as informações confidenciais e privadas não estejam ao alcance de pessoas não autorizadas, e a privacidade, assegura que os indivíduos possam controlar quais informações suas são coletadas. Quando ocorre uma violação de confidencialidade, significa que informações foram divulgadas sem autorização (Stallings; Brown, 2014).

Integridade: Sobre este conceito, Sêmola (2013) afirma que é fundamental preservar a integridade de toda informação conforme fornecida pelo seu detentor original, a fim de protegê-la contra quaisquer modificações não autorizadas, sejam elas intencionais ou acidentais.

Disponibilidade: A disponibilidade assegura que os sistemas permaneçam operacionais e prontos para uso quando necessário. Além disso, este princípio

garante que os serviços de segurança necessários pelo profissional da área estejam plenamente operacionais (Hintzbergen *et al*., 2018).

O valor da informação está intrinsecamente ligado às suas características. Se uma característica da informação se altera, seu valor geralmente diminui, embora em alguns casos possa aumentar. Certas características têm um impacto maior no valor percebido da informação pelos usuários do que outras. Isso pode variar dependendo das circunstâncias; por exemplo, a atualidade da informação pode ser crucial, já que a informação perde sua utilidade quando é entregue com atraso (Whitman; Mattord, 2011).

2.2. Controle de acesso

Conforme citado por Brasil (2012, p.16):

> Os controles de acesso, físicos ou lógico, têm como objetivo proteger equipamentos, aplicativos e arquivos de dados contra perda, modificação ou divulgação não autorizada. Os sistemas computacionais, bem diferentes de outros tipos de recursos, não podem ser facilmente controlados apenas com dispositivos físicos, como cadeados, alarmes ou guardas de segurança.

Os controles de acesso lógico consistem em uma série de métodos e estratégias destinados a salvaguardar informações, *softwares* e sistemas contra tentativas de acesso não autorizado realizadas por indivíduos ou outros *softwares* (Brasil, 2012).

Os principais propósitos da segurança cibernética são evitar que indivíduos não autorizados alcancem acesso aos recursos, garantir que usuários legítimos não acessem os recursos de forma não permitida e possibilitar que usuários autorizados obtenham acesso aos recursos de forma apropriada (Stallings; Brown, 2014).

Os sistemas de controle de acesso gerenciam a entrada de usuários em áreas seguras da organização, incluindo o acesso digital aos sistemas de informação e o acesso físico às instalações. Esses controles são mantidos por meio

de um conjunto de diretrizes, procedimentos para implementar essas diretrizes e tecnologias que aplicam tais diretrizes (Macedo, 2021).

Os controles de acesso lógico não apenas determinam quem ou o que pode acessar um recurso específico do sistema, mas também especificam o tipo de acesso permitido. Esses controles podem ser incorporados diretamente ao sistema operacional, integrados em programas de aplicativos ou utilitários essenciais (como sistemas de gerenciamento de banco de dados ou comunicação), ou instalados por meio de pacotes de segurança adicionais (NIST, 2017).

2.2.1. Autorização

Segundo Hintzbergen *et al.* (2018), uma autorização é composta por um conjunto de direitos concedidos. Essas permissões podem ser básicas, como acesso para leitura de um arquivo específico ou modificação de um registro em um banco de dados. Por outro lado, também podem ser mais complexas, como as permissões exigidas para realizar pagamentos em faturas bancárias.

Em empresas onde há rigorosas políticas de conformidade, as permissões costumam ser dadas pelo indivíduo encarregado do recurso, tipicamente um gerente. Além disso, em algumas circunstâncias, pode ocorrer de usuários individuais concederem acesso a outros usuários a recursos como dados ou programas (Hintzbergen *et al*., 2018).

No contexto padrão de um sistema Unix, a conta *root* recebe privilégios automáticos. Essa conta tem permissão para executar todas as ações no sistema. Por outro lado, os usuários têm acesso restrito, permitindo-lhes fazer *login*, acessar apenas certos arquivos e executar apenas os aplicativos autorizados (Rhodes-Ousley, 2013).

2.2.2. Autenticação

Autenticação é o procedimento para verificar se um usuário (ou outra

entidade) deve ser permitido a acessar um sistema. Por definição, os usuários que foram autenticados podem acessar os recursos do sistema. Contudo, é comum que esses usuários não tenham acesso total e irrestrito a todos os recursos. Por exemplo, apenas um usuário com privilégios elevados, pode ser autorizado a instalar *software* em um sistema (Stamp, 2011).

Em geral, considera-se que senhas seguras são aquelas que contêm uma combinação de letras (maiúsculas e minúsculas), números e símbolos misturados, com mais de seis caracteres. No entanto, para ser verdadeiramente segura, uma senha deve ser difícil de ser adivinhada por outras pessoas, mas ao mesmo tempo fácil de ser lembrada, evitando assim a necessidade de anotá-la em algum lugar (Brasil, 2012).

Nos sistemas Unix mais recentes, os nomes de usuário são registrados no arquivo */etc/passwd*, mas as senhas são guardadas em um arquivo distinto, chamado arquivo de senhas sombreadas, encontrado em */etc/shadow.* Este arquivo contém as senhas criptografadas e não é acessível para leitura por qualquer usuário comum. O acesso é limitado aos administradores do sistema, dificultando ataques provenientes de contas de usuário regulares (Rhodes-Ousley, 2013).

Normalmente, a autenticação inicia quando o usuário insere a senha no dispositivo. Em seguida, a senha é encaminhada para o servidor de autenticação e passa por um algoritmo de *hash*, resultando em um *hash* único para essa senha. Esse algoritmo de *hash* possui a característica de produzir um *hash* distinto para diferentes senhas, porém não é viável reconstruir a senha a partir do *hash* gerado (Tipton; Krause, 2007).

2.3. Servidor de *Proxy*

Servidores *proxy* são intermediários que realizam transações em nome de um cliente. Sem um servidor *proxy* da *web*, os clientes HTTP se comunicam diretamente

com os servidores HTTP. Com um servidor *proxy* o cliente, em vez disso, se comunica com o proxy, que por sua vez se comunica com o servidor em nome do cliente (Gourley; Totty, 2002).

Em configurações mais avançadas, um servidor *proxy* tem a capacidade de examinar solicitações usando diferentes critérios e somente autorizar a comunicação quando as solicitações se alinham com as regras estabelecidas. Essas regras normalmente levam em consideração o endereço IP do cliente ou servidor de destino, o protocolo utilizado, o tipo de conteúdo dos documentos da *web*, e outros aspectos relacionados ao conteúdo da *web* (Saini, 2011).

Ocasionalmente, um servidor *proxy* pode alterar os pedidos ou respostas, ou ainda guardar as respostas do servidor de destino em sua própria memória para atender à mesma solicitação mais tarde, seja para o mesmo cliente ou outros. Esse armazenamento local de respostas é chamado de *cache*. O *cache* é uma estratégia amplamente adotada por servidores *proxy* para reduzir o uso de largura de banda, fortalecer os servidores *web* e aprimorar a experiência de navegação dos usuários (Saini, 2011).

O uso de *caches web* pode significativamente diminuir o tráfego na conexão de uma instituição com a internet. Com essa redução no tráfego, a instituição, como uma empresa ou universidade, não precisa aumentar sua largura de banda com tanta urgência, resultando em economia de custos (Kurose; Ross, 2010).

Os servidores *proxy* têm uma variedade de funções benéficas. Eles podem aprimorar a segurança online, otimizar o desempenho de redes e reduzir os custos operacionais. Além disso, ao ter acesso ao tráfego HTTP, eles têm a capacidade de monitorar e ajustar esse tráfego para oferecer uma série de serviços *web* valiosos e úteis (Gourley; Totty, 2002).

2.3.1. Squid

O Squid funciona como um servidor *proxy* e *cache*, possibilitando o compartilhamento do acesso à *web* entre computadores na rede e melhorando a velocidade de acesso por meio do armazenamento em cache. O Squid é rico em recursos, incluindo autenticação de usuários, restrições de acesso, auditoria, entre outros, sendo uma solução completa para conceder acesso à Internet aos funcionários de uma grande empresa sem perder o controle sobre a utilização (Morimoto, 2004).

De acordo com Wessels (2004), o Squid roda em todos os sistemas Unix populares, assim como no Microsoft Windows. Os requisitos de *hardware* do Squid geralmente são modestos. A memória frequentemente é o recurso mais importante. Uma escassez de memória causa uma degradação drástica no desempenho. O espaço em disco é, naturalmente, outro fator importante. Mais espaço em disco significa mais objetos em cache.

A sintaxe do arquivo de configuração documentado do Squid é semelhante a muitos outros programas para Linux/Unix. Geralmente, há algumas linhas de comentários contendo documentação útil relacionada antes de cada diretiva usada no arquivo de configuração. Normalmente, é necessário ler os comentários e usar as opções apropriadas disponíveis para uma diretiva específica (Saini, 2011).

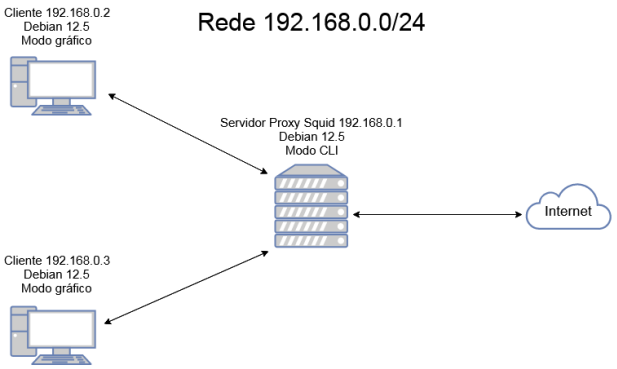
3. Desenvolvimento

3.1. Cenário de teste

O cenário de teste apontado na figura 1 e desenvolvido para a elaboração deste artigo consiste em uma rede interna com três máquinas, duas delas atuando como clientes e uma como o servidor de *proxy*. A máquina utilizada como servidor conta com duas interfaces de rede, uma delas em modo NAT conectada a internet, e a outra em modo de rede interna. As duas máquinas clientes possuem apenas uma interface de rede cada uma, em modo de rede interna.

A garantia de acesso à internet pelas máquinas clientes é feita através de regras de redirecionamento de portas utilizando o *firewall* Iptables na máquina servidor, onde o tráfego de internet da interface NAT é redirecionado para a interface de rede interna, onde as máquinas clientes estão conectadas.

Figura 1 – Diagrama da rede desenvolvida



Fonte: Autoria própria através do *software* VirtualBox

O sistema operacional escolhido para o experimento é o Debian em sua versão 12.5, uma distribuição GNU/Linux *open source*. Para questões de desempenho e maior facilidade de configuração, a máquina utilizada como servidor funciona com seu modo gráfico desabilitado, e as máquinas clientes com o modo gráfico ativado.

3.2. Configurações do servidor Squid

Todas as configurações descritas nas subseções seguintes foram elaboradas na máquina servidor e estão localizadas dentro do arquivo */etc/squid/squid.conf*.

3.2.1 Configurações de controle de acesso cliente 192.168.0.2

Figura 2 – Parâmetros de controle de acesso máquina 192.168.0.2

```
# config controle de acesso cliente 192.168.0.2
acl ip_origem1 src 192.168.0.2
acl dominios_bloqueados dstdomain "/etc/squid/redesociais_ia.txt"
http_access deny dominios_bloqueados ip_origem1
acl horario_uso time MTWHF 08:00-19:00
http_access deny !horario_uso
```

Fonte: Autoria própria através do *software* VirtualBox

A figura 2 apresenta as configurações utilizadas para o controle de acesso do cliente especificado:

Acl ip_origem1 src 192.168.0.2: Uma ACL denominada com o nome de ip_origem1 é criada e atribuída ao IP 192.168.0.2 utilizando o parâmetro src, este que é utilizado quando se deseja especificar um endereço IP ou sub-rede de onde as solicitações são originadas.

Acl domínios_bloqueados dstdomain "/etc/squid/redesociais_ia.txt": Essa regra de configuração cria uma ACL chamada dominios_bloqueados que contém os domínios listados no arquivo /etc/squid/redesociais_ia.txt. O parâmetro dstdomain específica os domínios de destino, que estão contido no arquivo de texto, representado na figura 3.

Figura 3 – Arquivo de texto que contém os domínios que são bloqueados

Fonte: Autoria própria através do *software* VirtualBox

Http_access deny dominios_bloqueados ip_origem1: A diretiva http_access, junto com o parâmetro *deny*, bloqueia o acesso HTTP para os domínios listados na ACL dominios_bloqueados para quando as solicitações serem originadas de ip_origem1.

Acl horario_uso time MTWHF 08:00-19:00 : Esta linha define o nome da ACL como horario_uso. O parâmetro time indica que a política de controle de acesso é baseada em tempo, e é aplicada de segunda-feira a sexta-feira(MTWHF) no intervalo de tempo das 08:00 às 19:00.

Http_access deny !horario_uso: A diretiva http_access, utilizada em conjunto com a opção *deny*, bloqueio o acesso a rede do cliente 192.168.0.2 em horários que não sejam entre 08:00hs e 19:00hs, e dias que não forem de segunda a sexta-feira, o sinal ! utilizado na configuração explicita negação.

3.2.2 Configurações de controle de acesso cliente 192.168.0.3

Figura 4 – Configurações de controle de acesso da máquina 192.168.0.3

```
# config controle de acesso cliente 192.168.0.3
acl ip_origem2 src 192.168.0.3
acl expressoes_bloqueadas url_regex -i "/etc/squid/expressoes_bloqueadas.txt"
http_access deny expressoes_bloqueadas ip_origem2
acl horario_uso2 time MTWHF 08:00-19:00
http_access deny !horario_uso2
```

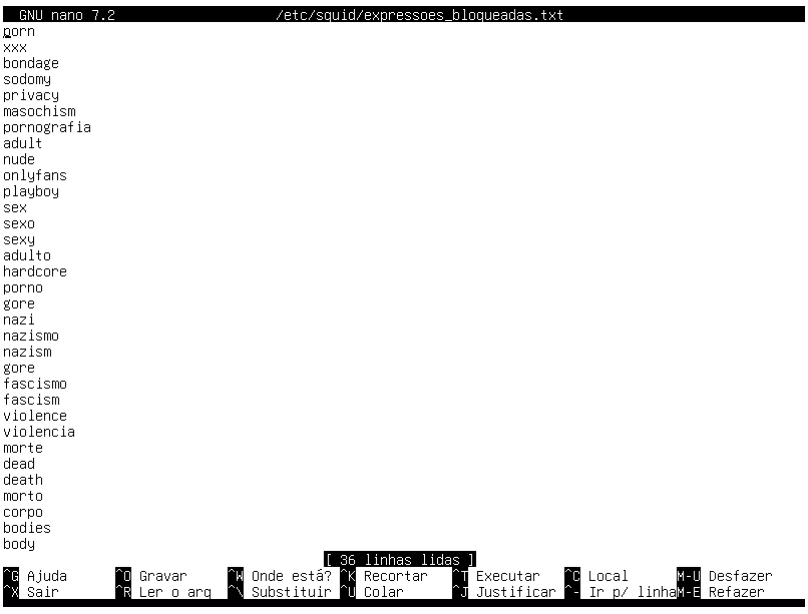
Fonte: Autoria própria através do *software* VirtualBox

Detalhamento sobre cada regra escolhida e aplicada para a máquina cliente 192.168.0.3, representado na figura 4:

Acl ip_origem2 src 192.168.0.3: Uma ACL denominada com o nome de ip_origem2 é criada e atribuída ao IP 192.168.0.2 através do parâmetro src.

Acl expressões_bloqueadas url_regex -i "/etc/squid/expressões_.txt": Esta ACL chamada expressões_bloqueadas lista uma série de palavras contidas no arquivo /etc/squid/expressões_bloqueadas.txt, em uso junto do parâmetro url_regex, que bloqueia palavras inapropriadas que estiverem na URL de um site. A opção -i possibilita que não haja distinção entre caracteres maiúsculos e minúsculos. A figura 5 exibe o arquivo criado.

Figura 5 – Arquivo de texto com as expressões bloqueadas



Fonte: Autoria própria através do *software* VirtualBox

Http_access deny expressoes_bloqueadas ip_origem2: Esta diretiva,

utilizada junto ao deny, nega acesso HTTP para as URLs que contém expressões listadas na ACL expressoes_bloqueadas para quando as solicitações forem do host ip_origem2.

Acl horario_uso2 time MTWHF 00:00-08:00: A ACL horario_uso2, junto da opção time indicam que o tempo é o tipo de controle de acesso aplicado, tendo vigência de segunda-feira a sexta-feira (MTWHF) entre as 0:00hs e as 08:00hs.

Http_access deny !horario_uso2: A diretiva http_access, junto com a opção deny, bloqueia o acesso a rede do cliente 192.168.0.3 em horários e dias que não forem os descritos na ACL horário_uso2, utilizando o símbolo de negação ! para isso.

3.2.3 Configurações de portas seguras

A configuração, exibida na figura 6, define as portas que o Squid irá considerar seguras para conexões é baseada em ACLs.

Figura 6 – Trecho da configuração de portas seguras contida no arquivo /etc/squid/squid.conf

```
 GNU nano 7.2                          /etc/squid/squid.conf
# config de portas seguras, metodo de conexao e porta que o squid ira operar
acl SSL_ports port 443
acl Safe_ports port 80
acl Safe_ports port 21
acl Safe_ports port 443
acl CONNECT method CONNECT
http_access deny !Safe_ports
http_access deny CONNECT !SSL_ports
http_port 192.168.0.1:3128
```

Fonte: Autoria própria através do *software* VirtualBox

Acl SSL_ports port 443: Esta linha cria uma ACL denominada SSL_ports, através da opção port na porta 443, que é a porta utilizada pelo protocolo HTTPS.

Acl Safe_ports port 80: Uma ACL chamada Safe_ports foi definida utilizando a porta 80, porta que o protocolo HTTP faz o uso.

Acl Safe_ports port 21: Mais uma ACL foi incluida na lista de portas seguras, na porta 21, utilizada pelo protocolo FTP.

Acl Safe_ports port 443: Uma ACL utilizando a porta 443 do protocolo HTTPS foi adicionada à lista de portas seguras.

Acl CONNECT method CONNECT: Define uma regra ACL chamada CONNECT que se aplica ao método HTTP CONNECT.

Http_access deny !Safe_ports: Esta configuração bloqueia conexão HTTP através do parâmetro deny, a opção ! significa negação, indicando que se negue conexão para o que não estiverem na lista de portas seguras.

Http_access deny CONNECT !SSL_ports: Esta linha diz que todas as requisições que correspondem à ACL CONNECT, todas as requisições que tentem usar o método CONNECT, são negadas, exceto se estiverem destinadas a portas que são permitidas para tráfego SSL.

Http_port 192.168.0.1:3128: Esta diretriz declara que o Squid irá operar no IP do servidor 192.168.0.1 através da porta 3128.

3.2.4 Configurações para permitir tráfego apenas na rede interna

A figura 7 mostra os parâmetros para permitir filtragem da rede de origem e declarar que o Squid irá operar apenas na rede interna criada.

Figura 7 – Regras de configuração da rede de origem que o Squid irá operar

```
# config acl para permitir trafego apenas da rede interna
acl Rede_Interna src 192.168.0.0/24
http_access allow Rede_Interna
http_access deny all
```

Fonte: Autoria própria através do *software* VirtualBox

Acl Rede_Interna src 192.168.0.0/24: A ACL Rede_Interna foi atribuída ao endereço de rede do cenário que foi criado através do parâmetro src.

Http_access allow Rede_Interna: Esta linha garante através do parâmetro allow, que todo o tráfego HTTP será permitido dentro da rede interna.

Http_access deny all: Esta regra sinaliza através da opção deny a negação de todo tráfego que não estiver dentro das regras descritas no arquivo de configuração do Squid.

3.2.5 Arquivo de *logs* de acesso do Squid

A diretiva access_log indica que o caminho utilizado para gravar o arquivo de logs de acesso será o /var/log/squid/access.log padrão utilizado pelo Squid, como demonstrado na figura 8.

Figura 8 – Linha indicando o caminho onde estará o arquivo de logs de acesso

```
# config arquivo de log
access_log /var/log/squid/access.log
```

Fonte: Autoria própria através do *software* VirtualBox

A figura 9 mostra o arquivo aberto. Nele é possível observar o bloqueio de acesso a domínios que são considerados bloqueados como Telegram, Youtube, ChatGPT e Instagram para a máquina cliente 192.168.0.2.

Figura 9 – Arquivo acess.log

```
debian:~# tail -f /var/log/squid/access.log
1717340655.053      0 192.168.0.2 TCP_DENIED/403 4213 GET http://detectportal.firefox.com/canonical.
html - HIER_NONE/- text/html
1717340655.058      1 192.168.0.2 TCP_DENIED/403 4213 GET http://detectportal.firefox.com/canonical.
html - HIER_NONE/- text/html
1717340655.063      0 192.168.0.2 TCP_DENIED/403 4213 GET http://detectportal.firefox.com/canonical.
html - HIER_NONE/- text/html
1717340655.066      0 192.168.0.2 TCP_DENIED/403 4213 GET http://detectportal.firefox.com/canonical.
html - HIER_NONE/- text/html
1717340655.068      0 192.168.0.2 TCP_DENIED/403 4213 GET http://detectportal.firefox.com/canonical.
html - HIER_NONE/- text/html
1717340655.071      0 192.168.0.2 TCP_DENIED/403 4213 GET http://detectportal.firefox.com/canonical.
html - HIER_NONE/- text/html
1717340665.679      0 192.168.0.2 TCP_DENIED/403 3973 CONNECT web.telegram.org:443 - HIER_NONE/- tex
t/html
1717340668.170      0 192.168.0.2 TCP_DENIED/403 4223 GET http://chat.openai.com/ - HIER_NONE/- text
/html
1717340668.289      0 192.168.0.2 TCP_DENIED/403 4259 GET http://debian:3128/squid-internal-static/i
cons/SN.png - HIER_NONE/- text/html
1717340670.619      0 192.168.0.2 TCP_DENIED/403 3958 CONNECT youtube.com:443 - HIER_NONE/- text/htm
l
1717340692.865      0 192.168.0.2 TCP_DENIED/403 4000 CONNECT push.services.mozilla.com:443 - HIER_N
ONE/- text/html
1717340692.989      0 192.168.0.2 TCP_DENIED/403 3973 CONNECT web.telegram.org:443 - HIER_NONE/- tex
t/html
```

Fonte: Autoria própria através do *software* VirtualBox

3.2.6 Configurações para autenticação de usuários no Squid

As diretrizes utilizadas para autenticação estão demonstradas na figura 10.

Figura 10 – Parâmetros de configuração para autenticação

```
# config autenticacao
auth_param basic program /usr/lib/squid/basic_ncsa_auth /etc/squid/squid_password
auth_param basic realm Servidor de Proxy
auth_param basic credentialsttl 10 hours
auth_param basic casesensitive on
acl usersquid proxy_auth REQUIRED
http_access allow usersquid
```

Fonte: Autoria própria através do *software* VirtualBox

Auth_param basic program /usr/lib/squid/basic_ncsa_auth /etc/squid/squid_password: Esta diretiva aponta um programa *helper* que irá auxiliar a realizar a autenticação de usuários e realizar a validação das credenciais, neste caso o utilizado é o basic_ncsa_auth, um *helper* baseado em usuário e senha. O caminho /etc/squid/squid_password indica ao programa o caminho onde está localizado o arquivo que contém os usuários e as senhas, que será utilizado pelo *helper* verificar se as credenciais são válidas.

Auth_param basic realm Servidor de Proxy: Este parâmetro indica que

nome será exibido para os usuários no *prompt* de autenticação, neste caso o nome escolhido foi "Servidor de Proxy".

Auth_param basic credentialsttl 10 hours: Define o *time-to-live(TTL)*, tempo de vida de vida das credenciais em *cache* por 10 horas, após a expiração deste tempo, o usuário precisará se autenticar novamente.

Auth_param basic casesensitive on: Esta diretiva aponta que a autenticação será *case sensitive*, ou seja ela será sensível ao diferenciar letras maiúsculas e minúsculas nos nomes de usuários e senhas.

Acl usersquid proxy_auth REQUIRED: Uma ACL com o nome usersquid foi criada, proxy_auth indica que o tipo de ACL corresponde a usuários autenticados, REQUIRED aponta que a autenticação é obrigatória para tornar esta ACL válida.

Http_access allow usersquid: Esta diretiva permite acesso através da opção *allow* para todos os usuários que se autenticarem com sucesso.

3.3. Criando usuários e senhas para autenticação no Squid

A criação de usuários e senhas para utilização na autenticação do servidor de *proxy* Squid é feita através do utilitário *htpasswd*. Esta ferramenta é encontrada através do *download* do pacote *apache2-utils*, um conjunto de ferramentas disponibilizadas pelo projeto Apache HTTP Server.

3.3.1. Criando um usuário e senha através do *htpasswd*

Para criar um usuário e sua senha utilizando o utilitário *htpasswd* é necessário digitar o seguinte comando no terminal do Debian:

htpasswd /etc/squid/arquivo-onde-as-credenciais-ficarao NomeDeUsuario

Ao abrir o arquivo, exibido na figura 11, utilizado para gravar as credenciais é possível encontrar o nome de usuário definido e a senha criptografada.

Figura 11 – Arquivo com as credenciais criadas

```
debian:~# cat /etc/squid/squid_password
usuario1:$apr1$WsTCdWvL$OQKy0J1NYtQ6d2tRLwZgu0
usuario2:$apr1$FYFPJuu6$pOB48omKSIHoupYXJvi8Q.
UsuarioTeste:$apr1$xDeNTzL0$H/S8fzulkYQXs/RS/DppK/
debian:~# _
```

Fonte: Autoria própria através do *software* VirtualBox

3.4. Testando as configurações de controle de acesso dos clientes

As subseções a seguir apresentam os resultados dos testes feitos utilizando as regras de controle de acesso nas máquinas clientes da rede interna.

3.4.1. Teste de configurações cliente 192.168.0.2

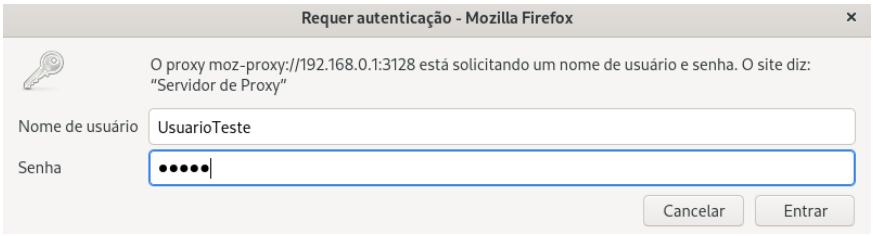
É necessário configurar o navegador manualmente para detecção do servidor de *proxy*, indicando o endereço do servidor e a porta em que serviço está funcionando, conforme demonstrado na figura 12.

Figura 12 – Configuração do navegador para receber o servidor de proxy

Fonte: Autoria própria através do *software* VirtualBox

A figura 13 apresenta o navegador sendo aberto na máquina cliente, um *prompt* é levado a tela solicitando as credenciais de autenticação do usuário. Ele também exibe a mensagem "Servidor de Proxy" configurada no arquivo de configuração do Squid.

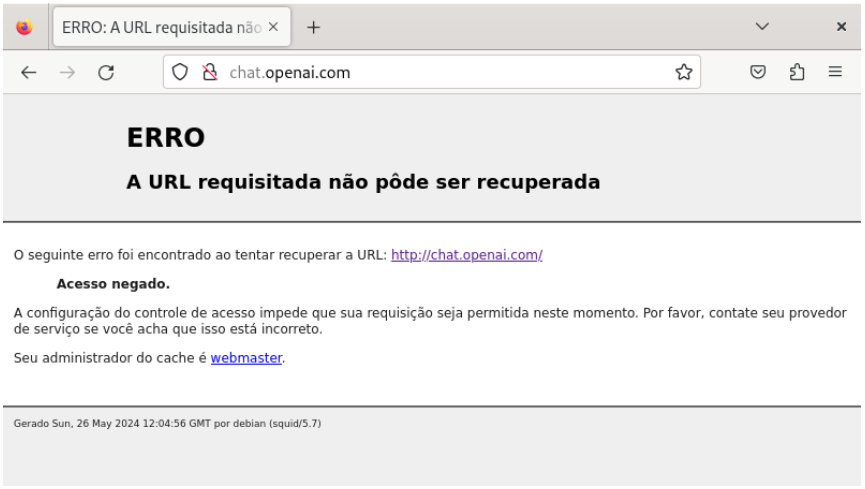
Figura 13 – *Prompt* de autenticação



Fonte: Autoria própria através do *software* VirtualBox

Como é possível observar na figura 14, o acesso a um dos domínios listados para ser bloqueado ocorre, surgindo um aviso na tela de acesso negado.

Figura 14 – Tela de aviso de acesso negado



Fonte: Autoria própria através do *software* VirtualBox

As regras de controle de acesso por tempo e horário impedem que o acesso a internet seja feito aos fins de semana, mesmo que seja a um site não listado na lista de bloqueio, como exibido na figura 15.

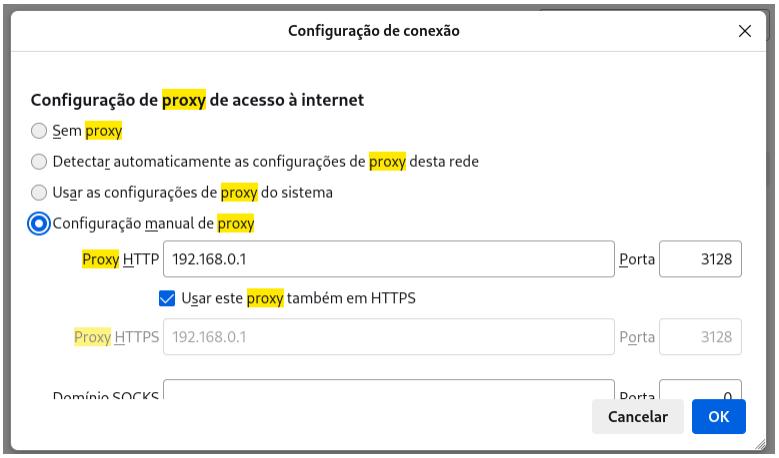
Figura 15 – Tela informação a requisição negada

Fonte: Autoria própria através do *software* VirtualBox

3.4.2. Teste de configurações cliente 192.168.0.3

A figura 16 aponta o ajuste manual do manualmente para detectar o servidor de *proxy*, especificando o endereço do servidor e a porta onde o serviço está ativo.

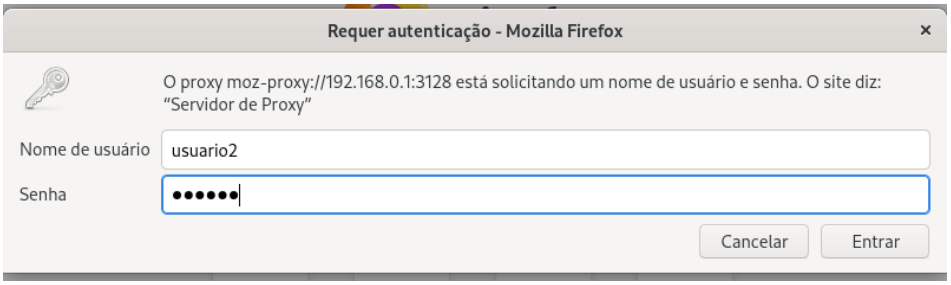
Figura 16 – Configuração no navegador para o servidor de *proxy*



Fonte: Autoria própria através do *software* VirtualBox

Logo após o navegador ser iniciado, um *prompt* de autenticação é exibido exigindo as credenciais de usuário, como exibido na figura 17.

Figura 17 – *Prompt* exigindo as credenciais de usuário



Fonte: Autoria própria através do *software* VirtualBox

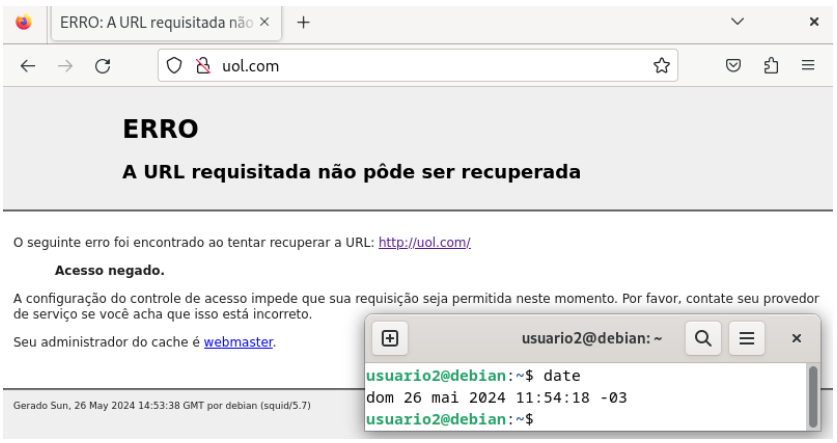
Na figura 18 é demonstrada a tentativa de acessar um site que contém um termo da lista de palavras inapropriadas em sua URL, onde o acesso é negado.

Figura 18 – Mensagem informando acesso negado

Fonte: Autoria própria através do *software* VirtualBox

Ao tentar acessar a internet num dia que não consta na ACL de horário e dia de uso, a conexão é recusada, como exibido na figura 19.

Figura 19 – Acesso negado



Fonte: Autoria própria através do *software* VirtualBox

4. Conclusão

Este trabalho de conclusão de curso teve como objetivo implementar e avaliar a eficácia do Squid como ferramenta de controle de acesso à internet, utilizando métodos de bloqueio de domínios, palavras na URL e restrições baseadas em dias e horários. Após a execução do projeto e a demonstração dos resultados, é possível concluir que o Squid se mostrou uma solução robusta e flexível para a administração de políticas de acesso à web em ambientes corporativos e educacionais.

O Squid permitiu a configuração detalhada de regras de bloqueio, o que possibilitou a criação de um ambiente de navegação mais seguro e produtivo. A capacidade de bloquear domínios específicos foi particularmente útil para restringir o acesso a sites não relacionados às atividades profissionais ou educacionais. Além disso, o bloqueio de URLs com base em palavras-chave demonstrou ser uma ferramenta eficiente para impedir o acesso a conteúdos inadequados ou que pudessem comprometer a segurança da rede.

As restrições baseadas em dias e horários forneceram um controle adicional, permitindo que a navegação fosse ajustada apenas para quando se fosse necessário. Essa funcionalidade mostrou-se essencial para otimizar o uso da banda de internet e para garantir que os recursos da rede fossem utilizados de forma adequada.

Referências

BRASIL. TRIBUNAL DE CONTAS DA UNIÃO. **Boas práticas em segurança da informação**. 4. ed. Brasília: [*s.n.*], 2012. E-book. Disponível em: https://portal.tcu.gov.br/lumis/portal/file/fileDownload.jsp%3FfileId%3D8A8182A24F0A728E014F0B226095120B&ved=2ahUKEwiM84awidmFAxViqZUCHRmyBXAQFnoECA4QAQ&usg=AOvVaw316E4O6UZ-v7IOPt8YHdqI. Acesso em: 21 maio 2024.

GOURLEY, David; TOTTY, Brian. **HTTP:** The definitive guide. 1. ed.

Sebastopol/Estados Unidos: O'reilly Media, 2002.

HINTZBERGEN, Jule et al. **Fundamentos da segurança da informação:** com base na IS0 27001 e na ISO 27002. Traduzido por: Alan de Sá. 3. ed. Rio de Janeiro: Brasport, 2018.

KUROSE, Jim; ROSS, Keith. **Redes de computadores:** uma abordagem top-down. Traduzido por: Opportunity Translations. 5. ed. São Paulo: Pearson Education, 2010.

MACEDO, Marcus. **Introdução à segurança da informação corporativa.** 1. ed. Pernambuco: [*s.n.*], 2021. E-book. Disponível em: http://www.portais.pe.gov.br/c/document_library/get_file?p_l_id=30580954&folderId=61591351&name=DLFE-508005.pdf. Acesso em: 21 maio 2024.

MASCARENHAS NETO, Pedro Tenório; ARAÚJO, Wagner Junqueira. **Segurança da informação:** uma visão sistêmica para implantação em organizações. João Pessoa: Editora UFPB, 2019. E-book. Disponível em: http://www.editora.ufpb.br/sistema/press5/index.php/UFPB/catalog/view/209/75/905-1. Acesso em: 21 maio 2024.

MORIMOTO, Carlos E. **Entendendo e dominando o linux**. 3. ed. [*S.l.: S.n.*], 2004.

NIST. NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY. **An introduction to information security**. [*S.l. ; S.n.*], 2017. Disponível em: https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-12r1.pdf. Acesso em: 26 maio. 2024.

RHODES-OUSLEY, Mark. **Information security:** the complete reference. 2. ed. Nova York/Estados Unidos: McGraw Hill, 2013.

SAINI, Kulbir. **Squid proxy server 3.1:** begginer's guide. Birmingham/Inglaterra: Packt Publishing, 2011.

SÊMOLA, Marcos. **Gestão de segurança da informação:** uma visão executiva. 2. ed. [*S.l.*]: GEN LTC, 2013. E-book. Disponível em: https://www.livrodeseguranca.com/. Acesso em: 26 maio. 2024.

STALLINGS, William; BROWN, Lawrie. **Computer security**: principles and practice**.** 3. ed. Londres/Inglaterra: Pearson, 2014.

STAMP, Mark. **Information security:** principles and practice. 2. ed. Hoboken/Estados Unidos: Wiley, 2011.

TIPTON, Harold F.; KRAUSE, Micki. **Information security management handbook.** 6. ed. Boca Raton/Estados Unidos: CRC Press, 2007.

WESSELS, Duane. **Squid:** the definitive guide. Sebastopol/Estados Unidos: O'Reilly, 2004.

WHITMAN, Michel E.; MATTORD, Herbert J. **Principles of information security.** 4. ed. Boston/Estados Unidos: Cengage Learning, 2011.