INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
DO RIO GRANDE DO SUL
CAMPUS RESTINGA

BRAÇO FORTE – INTERFACE GRÁFICA PARA SERVIÇOS DE REDE INTEGRADOS

MÁRCIO JOSÉ DE ÁVILA GOMES

Porto Alegre
2018

MÁRCIO JOSÉ DE ÁVILA GOMES

BRAÇO FORTE

Trabalho de Conclusão de Curso apresentado, junto ao Curso de Análise e Desenvolvimento de Sistemas do Instituto Federal Educação, Ciência e Tecnologia do Rio Grande do Sul, como requisito parcial para a obtenção do grau de Tecnólogo em Análise e Desenvolvimento de Sistemas.

Orientador: Prof. Rodrigo Lange

Porto Alegre
2018

MÁRCIO JOSÉ DE ÁVILA GOMES

BRAÇO FORTE

Trabalho de Conclusão de Curso apresentado como requisito parcial para a obtenção do grau de Tecnólogo em Análise e Desenvolvimento de Sistemas.

Orientador: Prof. Rodrigo Lange

Aprovado em MÊS, ANO.

_____

Prof. Rodrigo Lange

_____

Membro da Banca - Professor José da Silva – Instituição

_____

Membro da Banca – Professora Maria da Silva – Instituição

_____

Membro da Banca – Professora Ana dos Santos – Instituição

Aos meus país, minha noiva, amigos e colegas que persistiram na jornada.

“Wyrd bið ful ãræd – O destino é inexorável.”

(Bernard Cornwell)

RESUMO

Este trabalho de conclusão de curso tem por objetivo ser uma solução para problemas encontrados nas Forças Armadas. A partir de 2003, as instituições governamentais passaram a adotar o plano de migração de software livre. Esse plano visa diminuir os gastos e dependência com softwares proprietários e o legado que eles deixam, além de prover uma forma mais segura de auditoria em segurança nos softwares adquiridos. As Forças Armadas adotaram o mesmo plano de migração e sofrem a falta de mão de obra qualificada para trabalhar com as novas tecnologias implantadas. Com isso, este trabalho foi desenvolvido como uma ferramenta de interface gráfica para administração de redes, utilizando softwares livres disponíveis nas distribuições Linux. Neste trabalho é apresentado soluções para criação de Firewall, Proxy, DHCP, DNS e Domínio, onde é possível realizar as configurações desses serviços sem a necessidade de estudo prévio ou familiaridade no uso de terminal, apenas em poucos cliques, sendo mais uma alternativa entre as ferramentas disponíveis no mercado.

PALAVRAS-CHAVE: Firewall, Rede, Software Livr

ABSTRACT

This course completion work is intended to be a solution to problems encountered in the Armed Forces. From 2003, government institutions began to adopt the free software migration plan. This plan aims to reduce spending and dependence on proprietary software and the legacy they leave, as well as providing a more secure way to audit the security of purchased software. The Armed Forces have adopted the same migration plan and suffer the lack of skilled labor to work with the new technologies deployed. With this, this work was developed as a graphical interface tool for network administration, using free softwares available in Linux distributions. This work presents solutions for the creation of Firewall, Proxy, DHCP, DNS and Domain, where it is possible to configure the services without prior study or familiarity in the use of the terminal in just a few clicks. tools available in the market.

KEYWORDS: Firewall, Free Software, Network.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1. INTRODUÇÃO

Para qualquer instituição moderna, um dos seus bens mais importantes é a informação. Devido a isso, é essencial que ela seja protegida e gerenciada. Com a evolução dos meios de comunicação, as organizações se modernizam para que tenham maior eficiência e agilidade nas decisões, sendo assim quase nula a chance de as empresas não utilizarem sistemas de informação. Por isso é de vital importância se valer de mecanismos de segurança para preservar sua sobrevivência.

É sabido que com o alto grau de interconexão das organizações, aumentaram os riscos de, entre outros, vazamento de dados críticos ou comprometimento de recursos computacionais devido a invasões ou instalações de aplicativos maliciosos. É fácil concluir que, neste contexto, é vital que qualquer instituição utilize mecanismos que protejam seus sistemas de informações.

Como qualquer outra organização, o Exército Brasileiro (EB) se preocupa com a preservação da informação e elabora a mais de duas décadas planos de segurança da informação. O principal encarregado pela elaboração desses planos é o Departamento de Ciência e Tecnologia (DCT). Em consonância com o Governo Federal, que em 2003 aprovou o “Planejamento Estratégico de Implementação de Software Livre no Governo Federal”, que originou o Primeiro Guia Livre (Referência de Migração para Software Livre do Governo Federal / Organizado por Grupo de Trabalho Migração de Software Livre. Brasília 2005), coordenado pelo Comitê Técnico de Implementação de Software Livre (CISL), Secretária de Logística e Tecnologia da Informação (SLTI) e Comitê Técnico de Sistemas Legados e Licenças de Software (CTSLL), as Forças Armadas também criaram diretrizes de migração para Software Livre (SL).

De acordo com o Guia Livre, em um experimento na migração para software livre realizado no Ministério do Desenvolvimento Agrário, foi realizada a migração de 95% dos servidores e de 30% das máquinas dos usuários, foi gerada uma economia estimada de R$ 721.380,00. A adoção de soluções baseadas em SL teve não apenas impacto financeiro devido à economia gerada, mas também trouxe garantia de segurança devido ao código aberto dos softwares relacionados. Softwares de código-aberto podem ser completamente auditados, garantindo sua segurança.

A implantação no Exército se deu a partir de 2004, com o primeiro plano de migração (Boletim do Exército N° 47 de 18 de novembro de 2004). Esse plano tinha por finalidade regular a estratégia de implementação de SL nos escalões do EB e objetivos de apresentar uma reformulação nos processos de aquisição e utilização de software, trazer uma potencial

economia de custos com a aquisição e manutenção de softwares, criar um Núcleo de Estudos de Software Livre (NESOL) e um Centro de Desenvolvimento de Sistemas (CDS), diminuir a dependência do legado baseado em tecnologia proprietária e priorizar a compra de hardware compatível às plataformas livres.

Contudo a implementação dos SL nas Forças Armadas trouxe problemas, como por exemplo a pouca capacidade de formação de mão de obra capacitada para trabalhar com os novos softwares, a dificuldade de padronizar as ferramentas usadas nas Organizações Militares, e a dificuldade de trabalhar com o sistema operacional Linux, especialmente no tocante a ferramentas para o gerenciamento de redes de computadores.

Para enfrentar esse problema, este trabalho propõe o desenvolvimento e a implementação de uma interface gráfica que permita a integração de diversas ferramentas que são empregadas em redes de pequeno porte. Tais ferramentas são pacotes integrantes dos sistemas operacionais baseados em Linux.

1.1 MOTIVAÇÃO

Esse trabalho tem a motivação de ajudar sanar algumas das adversidades encontradas por diversos militares pelo Brasil que são responsáveis pela Tecnologia da Informação das Organizações Militares e não possuem formação na área, usando as ferramentas de Software Livre. A implementação do sistema ajudará no compartilhamento de conhecimento das ferramentas que por muitas vezes assustam os usuários mais leigos, que não encontram por onde começar a estudar.

Devido à formação geral dos militares envolvidos no gerenciamento de redes na maioria das pequenas unidades do Exército, um grande desafio em usar ferramentas como iptables, Squid, Sarg, Samba e outros é a dificuldade de encontrar material completo em português. Por muitas vezes as soluções encontradas em fóruns brasileiros não atendem a necessidade do usuário. Para isso o sistema dará um início mais confortável ao novo usuário para entender como funcionam os softwares.

1.2 OBJETIVOS

1.2.1 OBJETIVO GERAL:

Implementar interface gráfica para interação de leigos em Linux com as ferramentas de segurança e manutenção de rede, agregando conhecimento aos usuários, sendo uma porta de

entrada para mais entusiastas no mundo do Software Livre.

1.2.2 OBJETIVOS ESPECÍFICOS:

a) Configuração de um sistema de rastreamento de estações de rede;
b) Estudar e configuração um Firewall (iptables) funcional;
c) Configuração um sistema Proxy (Squid) e monitoramento de acesso (Sarg);
d) Configuração de compartilhamento e domínio via Samba;
e) Configuração de servidor DNS; e
f) Configuração de servidor DHCP.

1.3 ORGANIZAÇÃO DO TEXTO

No Segundo capítulo será abordado o tema do Software Livre, sua origem e criação, migração de Software Livre no Governo brasileiro e nas Forças Armadas. No terceiro capítulo será abordado as ferramentas utilizadas neste estudo com uma breve descrição e funcionalidades. No quarto capítulo será apresentado softwares semelhantes e uma tabela comparativa do estudo desenvolvido e os softwares abordados. No quinto capítulo será apresentado o desenvolvimento do projeto e será apresentado as telas funcionais do sistema. No sexto capítulo é apresentado os resultados de testes das funcionalidades. O último capítulo apresenta as conclusões do projeto apresentado.

2. FERRAMENTAS INTEGRADAS PARA GERENCIAMENTO DE REDE

Neste capítulo são apresentadas soluções integradas para gerenciamento de rede semelhantes à proposta apresentada neste trabalho. Essas soluções podem incluir uma ou mais ferramentas descritas no capítulo anterior e por isso foram escolhidas como parâmetro de comparação.

No final deste capítulo, será apresentada uma tabela comparando as soluções existentes apresentadas com a solução proposta neste trabalho.

2.1 PFSENSE

O pfSense (BUECHLER, Chris; ULLRICH, Scott. Eletric Sheep Fencing, 2004) é open-source (código aberto) com licença BSD license e baseado em FreeBSD. É adaptado para fazer o papel de firewall e roteador de rede, possuindo vários recursos, vindos de pacotes de software livre de terceiros, tais como Snort, Suricata, Squid, ClamWin e outros. Além de ser gratuito é considerando uma central unificada de gerenciamento de ameaças, possui geradores de relatórios em gráficos, modelagem de tráfego e filtragem. Os recursos são disponíveis via interface web.

Além da interface web é possível fazer o acesso via SSH ou até mesmo via porta serial, é bastante leve, rodando inclusive em máquinas bem ultrapassadas com apenas 128MB de memória RAM. É bastante robusto, permitindo ser usado mesmo por empresas com centenas de máquinas e diversas sub-redes.

Entre as principais funcionalidades, podemos listar:

a) Firewall;
b) Servidor (Internet, DHCP, NTP, Proxy, Web e mais);
c) Antivírus;
d) Antispyware;
e) Antispam;
f) Filtragem de conteúdo; e
g) Detecção de intrusos.

O pfSense é uma solução gratuita que rivaliza com diversas soluções tradicionais do mercado, é bastante estável e tem uma equipe que mantem o projeto atualizado. É usado em diversas organizações governamentais por ser gratuito e completo, com comunidade bastante

participativa nos fóruns e documentação bastante completa.

2.2 BRAZILFW

BrazilFW (BRAZILFW, 2005) é uma mini distribuição Linux que se destina a ser um Firewall e Roteador. Desenvolvido para exigir pouco hardware, pode rodar em máquinas bastante antigas, com boa performance. Possui interface web para configuração totalmente em português, já que é desenvolvida por brasileiros. A ferramenta é gratuita, desenvolvida por uma empresa privada sem fins lucrativos.

Entre as funcionalidades é possível listar:

a) Webadmin em PHP;
b) Servidor DNS (BIND);
c) Proxy;
d) Gerador de relatórios;
e) Webserver;
f) Banco de Dados MySQL; e
g) DNS dinâmico com o IpUpdate.

O BrazilFW é um software brasileiro conhecido por muitos, utilizado em escolas. Possui uma comunidade pequena e pouca documentação para avaliar. Ainda está em desenvolvimento.

2.3 OPENWRT

O OpenWrt (PROJECT, OpenWrt. 2004) é uma distribuição Linux embarcada para roteadores pessoais. Originalmente foi criado para o roteador Linksys WRT54G, mas ganhou suporte para outros modelos, ainda que os dois mais famosos sejam o WRT54G e o Asus WL500G. Principalmente configurado por linha de comando, também dispõe de uma interface gráfica web ainda em desenvolvimento.

Entre as funcionalidades é possível listar:

a) Suporte a IPv4 e IPv6;
b) Roteamento de rede;
c) Topologia da rede via B.A.T.M.A.N;
d) Firewall;
e) Controle de invasões, via Snort;

f) VPN;
g) DNS e DHCP; e
h) Compartilhamento de arquivos via Samba.

2.4 ENDIAN FIREWALL

O Endian Firewall (Endian S.R.L. 2005) é um projeto open source baseado no IPCop. É uma solução para servidor de Internet, com diversas funcionalidades e interface gráfica web.

Entre as funcionalidades é possível listar:

a) Servidor Proxy;
b) Servidor DHCP e NTP;
c) Controle de Tráfego;
d) Firewall;
e) Antivírus;
f) VPN; e
g) Detecção de Intrusos.

2.5 COMPARAÇÃO DE SOFTWARES

A tabela 1 mostra a comparação entre os projetos apresentados anteriormente e o estudo desenvolvido, levando em consideração as funcionalidades disponíveis, método de utilização, tecnologia empregada e tipo de licença conforme tabela 1.

Os softwares semelhantes ao do trabalho proposto são bastante completos, mas são sistemas completos, sendo necessário a instalação do sistema operacional em uma máquina separada, não podendo utilizar uma máquina já em produção. O trabalho proposto tem por diferencial, ser uma solução que possa ser implementada em uma máquina pessoal ou servidor com sistema operacional já instalado e operar paralelamente com outros serviços dessa máquina.

	Proposta	pfSense	BrazilFW	OpenWrt	Endian
Roteamento	X	X	X	X	X
Firewall	X	X	X	X	X
Proxy	X	X	X	X	X
DHCP	X	X	X	X	X
Antivírus	-	X	X	X	X
AntiSpam	-	X	X	X	X
IDS	-	X	X	X	X
Compartilhamento de pastas	X	-	-	X	-
VPN	-	X	X	X	X
DNS	X	X	X	X	X
Relatórios de acesso	X	X	X	-	-
Sistema próprio	-	X	X	X	X
Licença	-	Apache	GLP	GLP	Várias
Linguagem	PHP			Lua	

Tabela 1. Comparativo de Softwares. Autoria própria.

O sistema tem restrições de uso conforme as distribuições dos Linux ao qual o trabalho proposto foi baseado, Linux Ubuntu versões 14.04 a 16.10 e Debian 8. Também é necessário atenção nas versões das ferramentas utilizadas, é possível incompatibilidade devido a modificações na localização dos arquivos de configuração ou mudanças na configuração do mesmo.

3. SOFTWARE LIVRE

Software Livre é um software que permite aos usuários o controle de execução, adaptação da computação e de processamento de dados às suas necessidades, também dá liberdade social, permitindo que os usuários cooperem sem a necessidade de nenhuma permissão. Assim os usuários podem executar, copiar, distribuir, estudar, mudar e melhorar o software, mas sem restringir que esse software possa ser vendido.

Há duas principais fundações responsáveis pela proteção e promoção do Software Livre e Software de Código Aberto, a Free Software Foundation (FSF) e a Open Source Iniciative (OSI). Há diferenças entre Software Livre e Código Aberto, um software ter seu código fonte aberto não garante que o software é livre, para um Software ser livre é necessário que permita que o software possa ser lido, modificado e redistribuídos conforme os quatro conceitos de Liberdade do GNU (GNU is not Unix) GPL (General Public License):

a) A liberdade de executar o programa, para qualquer propósito (liberdade nº 0)
b) A liberdade de estudar como o programa funciona e adaptá-lo para as suas necessidades (liberdade nº 1). O acesso ao código-fonte é um pré-requisito para esta liberdade.
c) A liberdade de redistribuir cópias de modo que você possa ajudar ao seu próximo (liberdade nº 2).
d) A liberdade de aperfeiçoar o programa, e liberar os seus aperfeiçoamentos, de modo que toda a comunidade se beneficie deles (liberdade nº 3). O acesso ao código-fonte é um pré-requisito para esta liberdade.

Além disso, um programa de Software Livre apenas é considerado GPL somente se todos os seus componentes forem GPL. A partir de restrições como essa foi criada a OSI, com objetivo de flexibilizar a criação de softwares de código aberto. Os aplicativos de código aberto podem ser modificados e redistribuídos, mas o desenvolvedor tem direito de impor algumas restrições. Um exemplo de aplicativo de código aberto é o Mozilla, que permite que o aplicativo seja modificado, mas o software modificado deve receber um nome diferente de Mozilla Firefox.

As maiores vantagens do Software Livre veem do compartilhamento do código-fonte, isso implica na simplificação de desenvolvimento de novas aplicações que não precisam ser iniciadas do zero, impactando diretamente na redução dos gastos e do esforço necessário na construção dos mesmos.

Outro ponto bastante significativo é o emprego de um grande número de

desenvolvedores e/ou usuários que auxiliam na correção de bugs em um tempo menor e com um número muito maior de usuários que podem gerar situação diferentes as testadas em ambiente controlado. Também gera uma maior competição no desenvolvimento de software, melhorias no suporte e redução de gastos extra software (Manuais, Mídias, etc). Muito significativo é a criação de empresas menores para atender as demandas locais, deixando de haver uma dependência tão grande das tradicionais gigantes de software.

Hoje temos disponíveis vários projetos muito importantes que são Software Livre ou Open Source, tais como:

a) Firefox, da Mozilla, um dos navegadores mais populares do mundo;
b) Android, sistema mais usado em smartphones;
c) GNU/Linux, principal projeto SL, um dos sistemas mais usados em servidores do mundo pela sua estabilidade e segurança e ainda é a base do Android;
d) Gnome, interface gráfica que roda em sistemas baseados em UNIX, é um dos principais do mundo;
e) Servidor Apache, servidor web mais usado do mundo, é multiplataforma;
f) Eclipse, uma das IDEs mais usadas no mundo, sendo a ferramenta oficial de diversas plataformas.

3.1 SOFTWARE LIVRE NO GOVERNO

O primeiro estado do Brasil a ter alguma legislação que garantia a preferência de software livre na administração pública ocorreu no Rio Grande do Sul, a lei 11.871/2002. A lei enfrentou diversas batalhas e apenas em 2015 conseguiu ser de fato liberada.

No Brasil, a lei veio logo em seguida, em 2003, com as diretrizes da implementação do software livre no governo federal e com o primeiro plano de migração para software livre em 2004.

Com o principal objetivo de reduzir custos, a implementação do SL na administração pública ganhou força e tinha por objetivos audaciosos de substituir 100% dos servidores departamentais, 80% das soluções de correio eletrônico, 50% dos sistemas operacionais das estações de trabalho, isso tudo até o fim de 2006, de acordo com o Plano de Padronização do Ambiente e Migração para Software Livre (Ministério do Planejamento, Orçamento e Gestão. Versão 1.2, 23 de maio de 2005).

Hoje o SPB (Software Público Brasileiro) distribuí mais de 50 programas de software livre destinados para a administração pública. Eles foram criados para atender as demandas da

administração em todas as esferas, com objetivo de diminuir custos e a dependência de fornecedores e de tecnologia (DANIEL, 2011), não gera dependência com empresas especializadas que consistirá em altos valores de modificação de software (BASIC, 2003). O SPB distribui o código-fonte, fomentando a competição entre distribuidores e melhorando a qualidade dos serviços e redução de custos (AMADEU, 2006).

A segurança dos softwares livres também é garantida pela fácil auditoria de código-fonte, permitindo que falhas, backdoors e trechos duvidosos sejam descartados. Como consequência direta, garante a segurança (AMADEU, 2006). O compartilhamento do conhecimento é um dos pontos do SPB, permitindo que melhorias sejam adotadas por qualquer um. Assim o conhecimento passa a ser disseminado, ajudando principalmente pequenas e médias empresas (AMADEU, 2006). A forte comunidade que se forma em torno dos softwares públicos traz um grande compartilhamento de conhecimentos, dando a sociedade inteira acesso, não importando se é um pequeno município ou grande órgão do governo federal (DANIEL, 2011).

3.2 SOFTWARE LIVRE NAS FORÇAS ARMADAS

Seguindo as diretrizes do governo federal, as Forças Armadas passaram a dar mais atenção ao Software Livre e também foram criados planos de migração. O Exército já se preocupava antes do governo federal já tendo criado planos de migração no fim da década de 90. No início dos anos 2000, já haviam planos para planejamento de migração. Em 2004 foi elaborado a primeira diretriz de migração para software livre (Boletim do Exército N° 47). Hoje grande parte das Organizações Militares tem em seus parques computacionais sistemas operacionais baseados em Linux. Apenas é autorizado ter Windows em parques de produção onde não haja solução em Software Livre.

No 1º Centro de GeoInformação, Organização Militar destinada na produção cartográfica da região sul do Brasil, é uma dessas organizações que produzia em software proprietário que não é disponível usar em Linux. A partir de 2014, a OM junto com o 2º Centro de GeoInformação, em Brasília passaram a trabalhar em conjunto para elaboração de plugins ao software QGis, software livre de cartografia, substituindo o software proprietário ArcGis. Com resultados bastante satisfatórios, o QGis se tornou software padrão nas duas OMs e está sendo implantado nas outras três OM de cartografia do país, 3°, 4° e 5° Centro de GeoInformações.

No próximo capítulo serão descritas soluções de Software Livre que podem ser

utilizadas no gerenciamento de redes de computadores, e que são relevantes para este trabalho.

4. APLICATIVOS DE SERVIÇO DE REDE

Nesta seção serão apresentados os serviços de rede, que são ferramentas necessárias para fornecer funcionalidades para rede interna e externa. Algumas dessas ferramentas são bem conhecidas do mundo da Tecnologia da Informação: Servidor Apache (Servidor HTTP), Servidor Samba (Servidor de arquivos) e Squid (Servidor Proxy). Essas ferramentas são úteis não apenas para interligar redes locais à Internet, mas também para proteger a mesma de invasões e ataques maliciosos.

4.1 SISTEMA OPERACIONAL LINUX

O Sistema Operacional Linux (TORVALDS, Linus. GNU/Linux, versão 0.02, 1992) foi desenvolvido pelo finlandês Linus Torvalds, sendo inspirado no Minix. Com código-fonte disponível sob licença GPL, permite ao usuário utilizar, modificar, alterar, estudar e distribuir livremente de acordo com as regras da licença. Foi desenvolvido de forma colaborativa e foi um dos berços do Software Livre e do código-aberto.

O Linux se popularizou por ser uma alternativa ao UNIX, SO padrão utilizado pelas universidades, que podia rodar em máquinas pessoais mais baratas. Também como alternativa ao Microsoft Windows e o Mac OS, sendo muito utilizado para Web e servidores de dados. Teve como início com a criação da GNU, projeto iniciado por Richard Stalmann em 1984, com o objetivo de impulsionar a criação de softwares livres. A primeira versão do Linux foi lançada em 1991, ainda sob licença pessoal de Linus.

Com ponto forte sendo a segurança e diversas distribuições possíveis, o Linux pode atender a todos públicos, desde os mais práticos e simples, como por exemplo o Linux Minti, quanto ao tradicional Debian e Slackware, quanto ao Red Hat, voltado para o público empresarial.

4.2 IPTABLES

Para criação de um firewall para este trabalho de conclusão de curso, foi escolhido o iptables (NETFILTER, Time. 2001). Foi escolhido devido a ser um aplicativo nativo do Linux e que dispõe de recursos suficientes para garantir a segurança da rede. Tem por objetivo proteger a máquina contra acessos e tráfegos indesejados, proteger serviços da máquina e bloquear a passagem pelo servidor. É um firewall muito flexível na programação de regras permitindo ao administrador ter controle de tráfego independentemente do tráfego da rede local ou entre

interfaces devido a organização das etapas de roteamento.

O iptables é um firewall de nível de pacote e funciona baseado no endereço/porta de origem/destino e trabalha com prioridades. Ele faz comparação de regras para saber se o pacote tem permissão de passar. Entre as funcionalidades estão:

a) Manipular e monitorar o tráfego de rede;
b) Fazer NAT (masquerading, source nat, destination nat, etc);
c) Redirecionamento de pacotes;
d) Modificar a prioridade dos pacotes;
e) Contagem de bytes;
f) Dividir o tráfego entre as máquinas;
g) Proteções anti-spoofing;
h) Proteções contra syn flood;
i) Entre inúmeras outras funções.

Por ser muito flexível o iptables fornece a quantidade de segurança necessária de acordo com a situação, pode oferecer segurança para máquinas isoladas ou para grandes organizações que necessitem de um controle rigoroso do tráfego, tudo depende da criatividade e conhecimento do administrador da rede. Acredita-se que 95% das invasões de rede são devido a falhas humanas.

O firewall funciona com regras, que são comandos para determinar as ações de acordo com endereço/porta, e origem/destino do pacote, e são armazenadas em chains. Chains são os locais onde as regras do usuário são armazenadas para que o firewall opere. Existe chains embutidas, que são do sistema e as criadas pelo usuário. As chains por sua vez são armazenadas em tabelas junto com regras com características em comum. Existem três tabelas disponíveis e suas chains:

a) Filter – INPUT, OUTPUT e FORWARD;
b) NAT – PREROUTING, OUTPUT e POSTROUTING; e
c) Mangle – INPUT, OUTPUT, FORWARD, PREROUNT e POSTROUTING.

4.2.1 TABELA FILTER

A tabela filter contém as regras que determinam a aceitação de um pacote, é a tabela básica, cuja as regras podem ser usadas de modo geral. Possui três chains: INPUT, utilizada quando o destino do pacote é a própria máquina, OUTPUT, todos os pacotes que foram gerados na máquina do firewall e que tem destino a rede e FORWARD, todos os pacotes que passam

pelo servidor em destino a outra máquina.

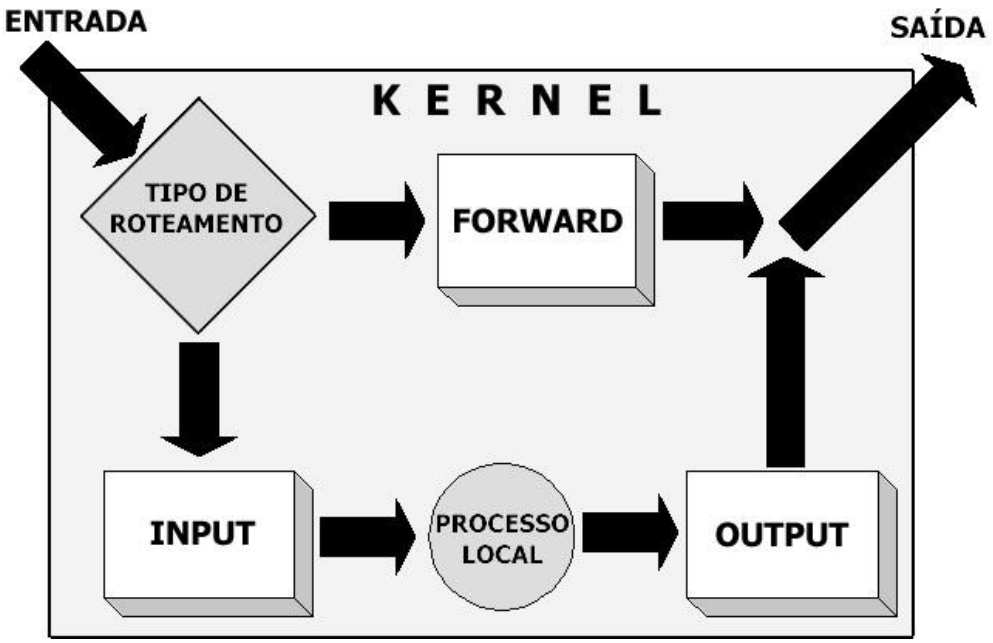


Figura 1. Tabela Filter, disponível em: http://eriberto.pro.br/iptables/

4.2.2 TABELA NAT

A tabela NAT (Network Address Translation) realiza a tradução dos endereços que passam pelo servidor. É uma função útil envolvendo os endereços IP, podendo alterar as características de origem e destino. Também possui três chains: PREROUTING, usada para analisar e modificar os pacotes que estão entrando no firewall, utilizada para fazer o DNAT (Dynamic Network Address Translation); O POSTROUTING, como o PREROUTING, modifica os pacotes na saída do firewall, realiza SNAT (Secure Network Address Translatio), modifica o endereço de origem do pacote, tornando a navegação mais segura e anônima, e OUTPUT, usada para fazer NAT com os pacotes originados no próprio servidor, sendo possível realizar DNAT com esses pacotes.

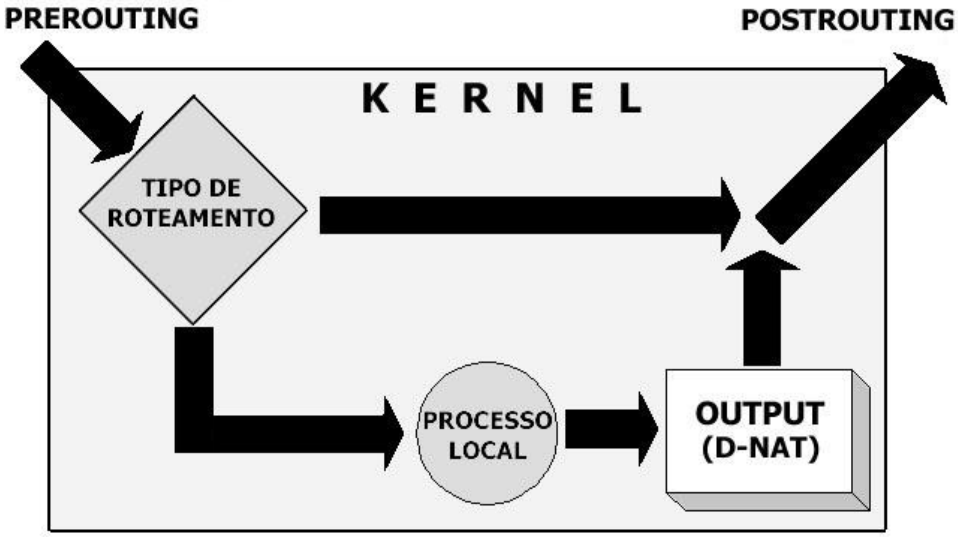


Figura 2. Tabela NAT, disponível em: http://eriberto.pro.br/iptables/

4.2.3 TABELA MANGLE

A tabela mangle tem funções especiais que são aplicadas no tráfego que passa pelas cadeias. Essas ações ocorrem antes da chains das tabelas Filter e NAT. Para melhor entendimento é melhor pensar que cada chain da tabela mangle corresponde a chain das outras tabelas. É possível visualizar conforme a Figura 3.

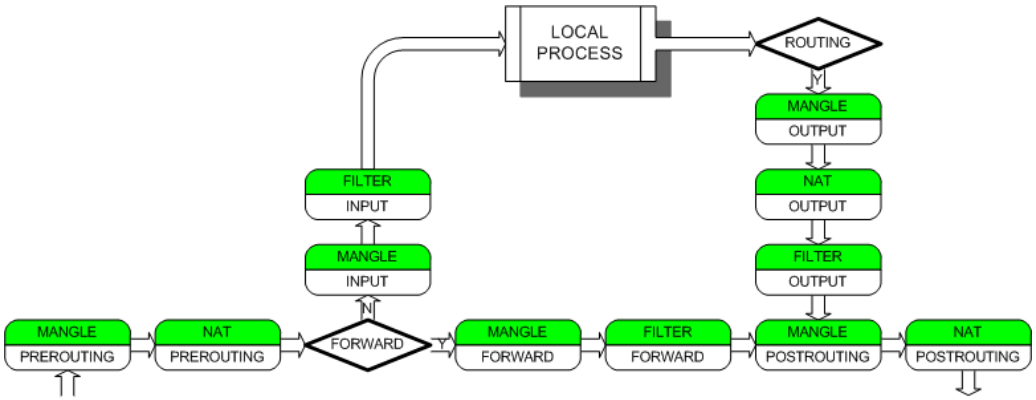


Figura 3. Lista de ações do iptables, disponível em: http://ebtables.netfilter.org/br_fw_ia/br_fw_ia.html

4.3 SQUID

O Squid (WESSELS, Duane et al. NSF, 1996) é um servidor proxy que tem suporte a vários protocolos, como por exemplo HTTP, HTTPS, FTP e outros. Com ele é possível reduzir o uso da conexão e melhora nos tempos da resposta por meio de cache em requisições frequentes de páginas web. No cache são armazenados dados da Internet disponíveis em

protocolos HTTP, FTP e Gopher. Esse recurso é utilizado por provedores no mundo todo para assegurar melhor velocidade de navegação. Pode também ser utilizado na segurança pois se trata de mais um intermediário para acesso aos objetos da rede. Contudo ele também pode fazer logs de registro de todas as conexões da rede, com as URLs, data e horas de acesso e quem acessou. Normalmente usados por empresas para controlar os acessos dos funcionários.

O Squid tem diversas funcionalidades, tais como proxy transparente, ao qual não necessita de configuração do navegador, pode tornar as conexões anônimas e pode alterar ou desabilitar campos do cabeçalho dos pedidos HTTP do cliente. Em alguns países do mundo, principalmente na Europa, é obrigatório que a empresa comunique aos funcionários que a rede está sendo monitorada por motivos de preservar a privacidade.

4.4 SARG

SARG ou Squid Analysis Report Generator (ORSO, Pedro. 2000), é um gerador de relatórios com informações dos usuários do Squid. É possível gerar gráficos e o relatório é alimentado com muitas informações de acesso dos usuários, tais como quando e o que acessou, quanto tempo e quanto de dados utilizou. Junto com o Squid é um excelente monitor de rede e sua interface simples e agradável permitem diversas configurações e personalizações.

Para utilizá-lo é necessário gerar logs com a periodicidade necessária para seu uso, sendo possível criar scripts para que seja diário, semanal ou mensal, ou todos juntos. Não é necessário conhecimentos avançados, se tornando uma ferramenta muito utilizado pelo mundo inteiro, inclusive há tradução para português.

4.5 SAMBA

O Samba (TRIDGELL, ANDREW. 1992), é um importante componente integrante do Linux/Unix para ambientes do Active Directory e serve como controlador ou membro de domínio.

O Samba tem diversas funcionalidades, as quais se destacam:

a) Compartilhamento de arquivos, impressoras, diretórios, etc, com máquinas Windows;
b) Controle de acesso e privilégios;
c) Resolução de nomes (DNS); e
d) Configuração a partir de ambientes remotos.

Uma das maiores vantagens do Samba é seu custo-benefício, já que ele simula um

Windows Server, com estabilidade, poucos bugs e com segurança sem nenhum custo, rodando em um GNU/Linux. A configuração é bastante simples, se dando em apenas um arquivo, o smb.conf, onde é possível criar os compartilhamentos e criar os usuários e grupos.

O Servidor Samba é poderoso e possui muitos recursos eficientes e é considerado por muitos até mais seguro que as soluções em Windows. De fácil aplicação e documentação extensa, é uma solução eficiente para qualquer tipo de ambiente.

Para informações detalhadas sobre o Samba, pode-se consultar o site oficial disponível em (https://www.samba.org/samba/docs/).

4.6 DHCP

O DHCP (Dynamic Host Configuration Protocol), é um protocolo de serviço TCP/IP que oferece configuração dinâmica para os terminais, sendo IP de host, máscara de sub-rede, gateway padrão e servidor DNS. É o sucessor do BOOTP que se tornou limitado para redes atuais. A grande vantagem de usar DHCP é poder amarrar o endereço MAC ao IP, tornando possível controlar os acessos de apenas máquinas cadastradas.

O servidor DHCP mais usado no Linux é o ISC DHCP (Internet Systems Consortium, Inc, 1994), desenvolvido por uma instituição sem fins lucrativos que desenvolve serviços de infraestrutura para Internet, e possui outros pacotes bastante usados, como o BIND e o NTPD.

4.7 DOMÍNIO

Com recursos do Samba é possível ter um DNS dinâmico, um servido LDAP, recursos de um Active Directory e um servidor de autenticação Kerberos. Uma boa alternativa ao Active Directory da Microsoft (MICROSOFT, Windows Server 2000, 2000), sendo possível inclusive utilizar o Active Directory Domain Controller após a empresa ter testado a interoperabilidade. Assim o Samba possibilita se valer das funcionalidades de um Windows Server sem pagar nada, podendo ter uma rede com máquinas Windows e Linux lado a lado segura e robusta.

4.8 BANCO DE DADOS MYSQL

MySQL (AXMARK, David; LARSSON, Allann; WIDENIUS, Michael. MySQL AB, 1995) é um sistema gerenciador de dados relacional de código aberto. Usa linguagem SQL (Structure Query Language – Linguagem de Consulta Estruturada), sendo hoje a linguagem

mais popular para manipular um banco de dados. Em aplicações web gratuitas, o MySQL é largamente utilizado, junto com o Linux, Apache, MySQL e Perl/PHP/Python, que juntos foram o acrônimo LAMP. Esse conjunto de aplicações consiste em um sistema operacional, um servidor Web, um sistema gerenciador de banco de dados e uma linguagem de programação. O MySQL começou em 1995 e hoje faz parte da Oracle, que restringiu parcialmente seu uso, sendo assim os desenvolvedores criaram o MariaDB(MariaDB Corporation Ab, MariaDB Foundation, WIDENIUS, Michael 'Monty'. 2009), como uma versão gratuita, que deverá ser a compatibilidade com as versões lançadas pela empresa.

Parte do sucesso do MySQL se dá pela fácil integração com o PHP. Várias grandes empresas usam o MySQL em suas aplicações. Possui várias vantagens, como portabilidade, compatibilidade, fácil manuseio, pouco exigente com hardware, bom desempenho e estabilidade, entre várias outras.

4.9 LARAVEL FRAMEWORK

Laravel (OTWELL, Taylor. 2011) é um framework para acelerar e estruturar a programação em PHP. É de código aberto e um dos mais usados no mundo, sendo o mais usado no GitHub e o framework mais procurado no Google. Possui um slogan que fala muito sobre ele “O framework PHP para artesão da web”, pois deixa seu código muito bem.

O Laravel usa o Composer (ADERMANN, Nils, BOGGIANO, Jordi) como gerenciador suas dependências, como a maioria das aplicações PHP modernas fazem. Com ele é possível de forma muito fácil gerenciar pacotes de terceiros na sua aplicação.

A documentação do Framework é bastante intuitiva e de fácil visualização, tornando muito fácil realizar buscas e se aprofundar nos recursos disponíveis. O sistema de rotas é bastante prático e organizado, fazendo o mapeamento do Url digitada no browser. É possível criar parâmetros, agrupamentos de rotas de maneira bem simples.

O sistema de templates usa o Blade, que é bastante flexível e não restringe o uso do PHP puro misturado com a sintaxe do template. O Blade tem por objetivo reduzir o código PHP na página no meio do HTML e aumentar o reuso, assim disponibiliza várias diretivas que são inseridas no código HTML. Os principais benefícios do uso de Blade é usar herança e as seções permitindo que se use o conceito de “master page”.

O Eloquent é o ORM (Mapeamento objeto-relacional) do Laravel, ele aplica o Design Patter ActiveRecord onde as tabelas do banco de dados são representadas em código através de

uma classe Modelo que é utilizada para interagir com a tabela. Nesses Modelos é possível fazer todas as operações com o banco sem o uso de SQL puro no código. Há uma série de métodos e recursos disponíveis na documentação.

O QueryBuilder é uma alternativa para não trabalhar com o Eloquent, é um construtor de querys usado para a maioria das operações no banco de dados.

Para facilitar o desenvolvimento da aplicação o Laravel usa o Artisan Console, é uma interface de linha de comando para criar os arquivos necessários.

5. SOLUÇÃO PROPOSTA

Este capítulo abordará os principais aspectos em relação à implementação, interfaces e diagramas envolvendo o software desenvolvido nesse projeto. Os próximos subcapítulos servirão para demonstrar as funcionalidades do sistema.

5.1 DIAGRAMA DE CASO DE USO

A figura 4 mostra o diagrama de casos de uso do Braço Forte, software desenvolvido neste projeto, onde é possível: configurar o iptables com regras pré-definidas e listas de portas liberadas ou bloqueadas; liberar ou bloquear o acesso das máquinas da rede à internet; configurar ferramentas como Squid, Sarg; DHCP; Domínio; compartilhar pastas com usuários pelo Samba; e a criação e alteração de usuários para a administração do sistema.

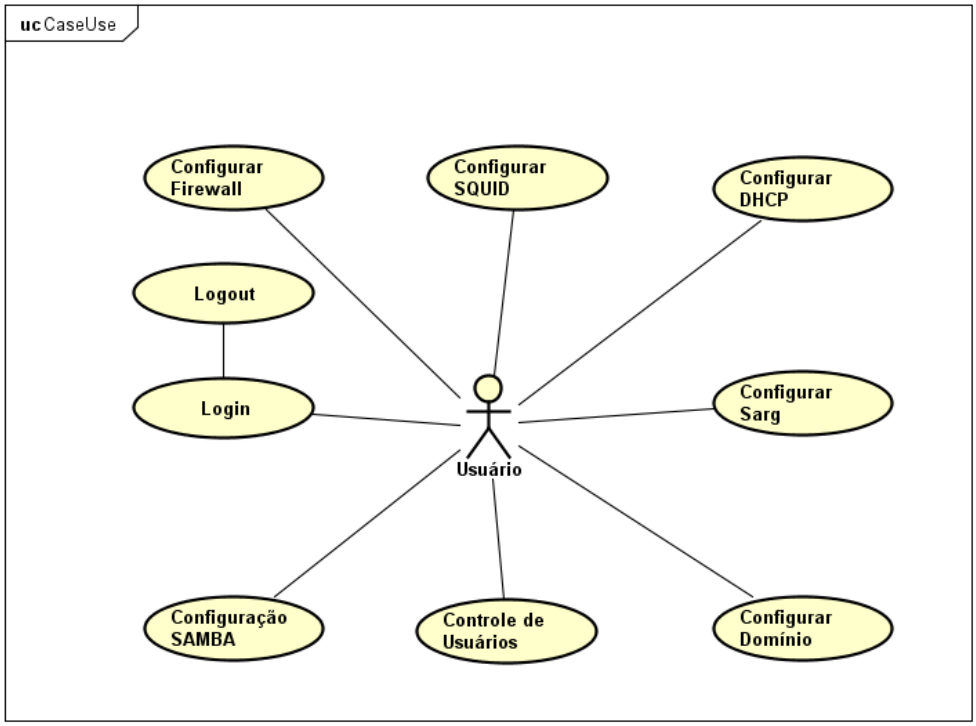


Figura 4. Casos de uso do Braço Forte. Autoria própria.

Ao fazer o login, o usuário poderá acessar o menu principal com todas as funcionalidades à disposição. Ao instalar o software, será criado uma senha default para o primeiro acesso, sendo necessário que seja trocada para garantir a segurança da rede e impedir o acesso de pessoas não autorizadas. As funcionalidades estão listadas como:

a) Firewall – Configurações de Firewall;
b) Máquinas na rede – Possibilidade de ativar ou desativar o acesso à internet das máquinas na rede;
c) Squid – Configuração do Squid;
d) Sarg – Configuração do Sarg;
e) DHCP – Configuração do DHCP;
f) Usuários – Controle dos usuários do sistema;
g) Domínio – Configuração de Domínio; e
h) Samba – Compartilhamento de pastas entre usuários.

5.3 IMPLEMENTAÇÃO

Neste subcapítulo será apresentado os pontos mais importantes na implementação do software e apresentar as páginas deste estudo desenvolvido.

5.3.1 LINGUAGENS

As linguagens empregadas para o desenvolvimento desse software.

5.3.1.1 PHP

O PHP que conhecemos se deu a partir do PHP 3.0, que se deu por início do seu desenvolvimento ainda em 1997. Andi Gutmans, Zeev Suraski e Rasmus, decidiram por desenvolver uma nova e independente linguagem de programação. Os pontos mais fortes desta versão do PHP vieram a ser os fortes recursos de extensibilidade, fornecimento de uma interface robusta para vários bancos de dados e APIs e a facilidade de estender a linguagem atraiu dezenas de desenvolvedores que criaram uma grande variedade de módulos. Essa foi a chave do sucesso do PHP 3.0. Também foi incluído o suporta a orientação a objetos e melhorias na sintaxe de linguagem.

A gama de opções que o PHP possibilita é imensa. É uma linguagem de programação back-end, voltada para rodar no lado do servidor. Graças a ela que é possível acessar os dados do banco e tratar as informações que são necessárias. O PHP também é usado para aplicativos móveis, mesmo não sendo a mais indicada para isso, ainda que consiga resultados significativos.

Diversas vantagens também acompanham a linguagem, o custo reduzido para

infraestrutura, uma linguagem consolidada no mercado a mais de 20 anos e uma curva de aprendizado menor que a maioria das linguagens.

Hoje o PHP está na versão 7.2.3, que trouxe grande ganho em performance, novas funcionalidades e também implementou e fortificou novos recursos de orientação a objetos. Está presente em diversas grandes aplicações do mercado, tais como: Facebook, Joomla e WordPress. É o concorrente direto da tecnologia ASP, da Microsoft, com a vantagem de ser um software livre.

A escolha do PHP se deu pela experiência na linguagem, a mais utilizada nos sistemas do Exército Brasileiro, além como já mencionado, uma linguagem consolidada no mercado com uma cursa de aprendizado rápida. O PHP não requer hardware potente e roda a partir do Apache, ferramenta gratuita do Linux.

5.3.1.2 JAVASCRIPT E JQUERY

O JavaScript é uma linguagem para o lado do cliente, utilizado para manipular o comportamento da página controlando o HTML e o CSS. Com ela é possível executar script pelo lado do cliente sem a necessidade de passar pelo servidor. Entre seus maiores usos estão a validação de campos de formulário, recursos de janelas de diálogo e animações de tela.

O JQuery é uma biblioteca fantástica que facilita a produção de reutilização de código, nada mais útil que “escrever menos e fazer mais”, para quem tem pouco tempo produzir um sistema. Torna o uso do Ajax mais fácil, tornando a página fácil de ser manipulada.

O JavaScript possibilita uma grande variedade de validações e manipulações nas páginas, sendo a forma mais simples de tornar as páginas mais interativas, é uma linguagem em expansão e não possui dependências para serem instaladas no servidor, por isso de sua escolha.

5.3.2 FRAMEWORK LARAVEL

Para implementação, foi escolhido como IDE o PhpStorm, ferramenta criada pela JetBrains, reconhecida por desenvolver ferramentas de implementação para diversas linguagens. O PhpStorm foi desenvolvido para projetos em HTML, CSS, PHP e JavaScript, é compatível com a maioria dos grandes frameworks do mercado. Possui muitos recursos, como ajudante de código, rápido e seguro refatoramento, debug fácil e ferramentas de teste. É a ferramenta mais recomendada aliada ao Laravel Framework.

5.3.3 ORGANIZAÇÃO

Por mais que muitos confundam que o Laravel é um framework MVC, isso não é totalmente verdade. O que define um MVC são as três camadas: Model, View e Controller conforme apresentado na figura 5. O Laravel até usa a estrutura de um MVC, mas acaba usando muito mais camadas, como a de rotas, middlewares e requests.

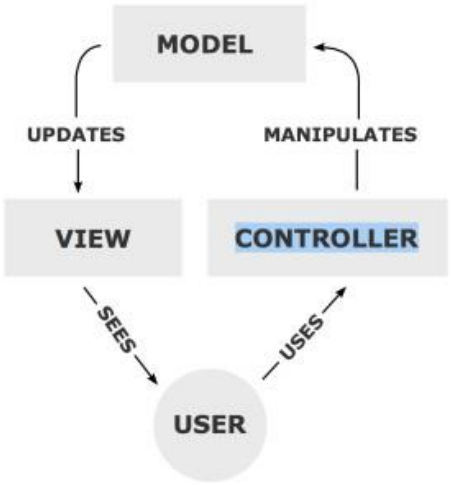


Figura 5. Modelo MVC. Fonte Wikipédia.

A partir do Laravel 5 não há mais pasta “Model” e os controladores apenas são uma pequena parte na pasta “Http”. A pasta com as Views não tem mais “resources”, mas sim a “resources” possui a “views” e outros componentes, como “assets” e “lang”. Isto não está acontecendo apenas no Laravel, está virando uma tendência, o CodeIgniter também já mudou. Robert Martin afirma que MVC não é uma arquitetura, que ela se perdeu nos anos conforme é mostrada na figura 6.

Não há mais como desenvolver uma aplicação sem uso de outros recursos. Talvez não se use repositórios ou se implemente middlewares, mas é provável que seja necessário usar interfaces e inclusão de dependências ou pelo menos Facades.

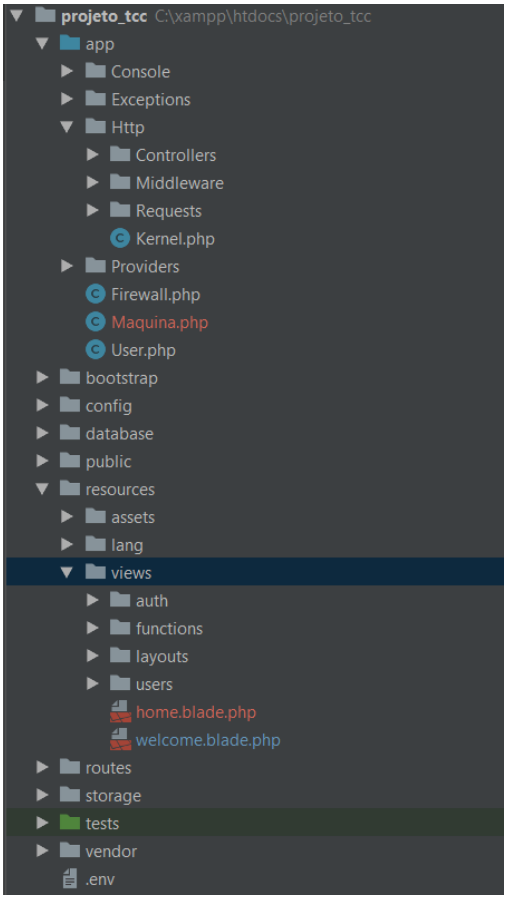


Figura 6. Organização da implementação. Autoria própria.

A estrutura deste estudo ficou estruturada conforma a Figura 6, onde se destacam:

Vendor – Pasta gerada automaticamente pelo composer contendo as bibliotecas do Laravel;

Public – Pasta que recebe todas as requisições do usuário, onde estarão todos nossos códigos javascript, ccs, imagens, etc.

Config – Contém os arquivos de configuração da aplicação.

Storage – Armazena os arquivos de uso interno do Laravel, como logs e cache.

App – Pasta principal da aplicação, onde a maior parte do desenvolvimento se encontra.

5.3.4 ROTAS

As rotas no Laravel foram feitas para facilitar a criação de serviços RESTful, assim quando é criado uma variável, objeto, array ou qualquer outro tipo de dado no Controller, é automaticamente convertido para JSON. Também é possível passar um JSON da View para o

Controller pelas rotas.

O sistema do Laravel permite registrar método de rotas para todos os verbos do HTTP, sendo eles:

a) GET – Utilizado para retornar valores, por exemplo listas;
b) POST – Utilizado no envio de dados da View para o Controller;
c) PUT ou PATH – Editar registros;
d) DELETE – Para deletar;
e) OPTIONS

O Laravel implementa proteção CSRF (Cross-site request forgery- Falsificação de solicitação entre sites) automática nas rotas tipo POST. Ataques CSRF tem por objetivo enviar um número muito grande de requisições para o sistema. Sem proteção, um sistema poderia ter seu banco de dados derrubado por excesso de dados. Para fazer essa proteção é necessário passar pelo formulário da requisição um token único para validar a requisição. Há diversas formas de utilizar o token, abaixo na figura 7 as duas formas utilizadas neste estudo.

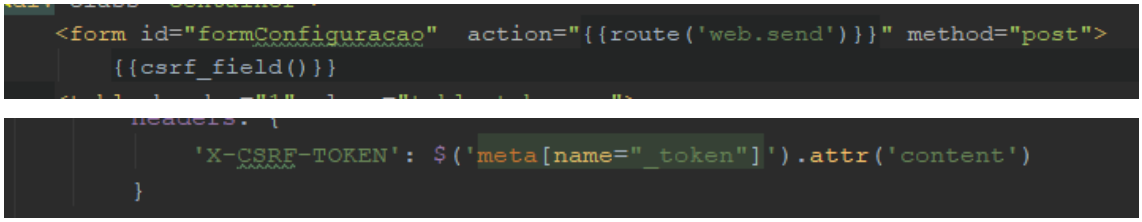


Figura 7. Métodos de inserção do CSRF-Token. Autoria própria.

Nomear as rotas é algo extremamente útil, simplificando a forma de acessar e facilitar a manutenção. É possível usar os nomes das rotas nos formulários de requisição e nos retornos dos Controllers na Figura 8.

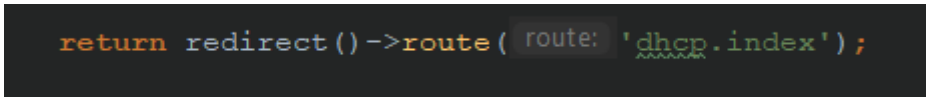


Figura 8. Redirecionamento de rota no Controller. Autoria própria

Os grupos de rotas é um recurso para aplicar regras em conjunto. O Prefix é um recurso para reduzir o caminho na edição das rotas. É possível agregar todas as páginas que possuem o mesmo caminho e abreviar o destino na regra da rota. É possível usar um prefix dentro de outro prefix. O namespace funciona da mesma maneira que o prefix para os controllers, sendo possível defini-lo diretamente na rota.

É importante que uma rota não tenha um retorno, ela deve passar a responsabilidade para o Controller, que é o encarregado de delegar as ações.

5.3.5 MIDDLEWARES

Middlewares são empregados como filtros sendo possível aplicar em um conjunto de rotas de uma única vez. Neste projeto, uma rota middleware foi utilizada para permitir que apenas usuários autenticados tenham permissão de acessar as rotas. O Laravel provém por padrão o Middleware “auth”, sendo necessário apenas implementar na rota como mostra a figura 9.

```
Route::group(['middleware' => ['auth']], function (){
    Route::prefix('admin')->namespace('Admin')->group(function (){
        Route::prefix('functions')->group(function (){
        Route::get('firewall', 'FirewallController@index')->name('firewall.index');
        Route::post('firewall', 'FirewallController@send')->name('firewall.send');
        Route::get('dhcp', 'DhcpController@index')->name('dhcp.index');
        Route::post('dhcp', 'DhcpController@send')->name('dhcp.send');
        Route::get('squid', 'SquidController@index')->name('squid.index');
        Route::post('squid', 'SquidController@send')->name('squid.send');
        Route::get('sarg', 'SargController@index')->name('sarg.index');
        Route::post('sarg', 'SargController@send')->name('sarg.send');
        Route::get('domain', 'DomainController@index')->name('domain.index');
        Route::post('domain', 'DomainController@send')->name('domain.send');
        Route::get('rede', 'WebController@index')->name('web.index');
        Route::post('rede', 'WebController@send')->name('web.send');

        });
        Route::prefix('users')->group(function (){
            Route::get('/', 'UserController@index')->name('user.index');
            Route::get('new', 'UserController@new')->name('user.new');
            Route::post('store', 'UserController@store')->name('user.store');
            Route::get('edit/{user}', 'UserController@edit')->name('user.edit');
            Route::post('update/{id}', 'UserController@update')->name('user.update');
            Route::get('remove/{id}', 'UserController@delete')->name('user.remove');

        });
    });
});
```

Figura 9. Arquivo de Rotas. Autoria própria.

5.3.6 CONTROLLER

Através do padrão MVC, toda funcionalidade deverá ter um Controller para ter a responsabilidade das ações. É importante ressaltar que os métodos dos controllers são chamados pelas Views através das rotas. É no Controller que toda a ação ocorrerá, e por ele que é feito todas as consultas, inserções, atualizações ou remoções no banco de dados. Os controladores agrupam toda a lógica de manipulação de solicitações relacionadas em uma única classe.

Os controladores servem de intermédio entre a View e a Modal e há métodos de facilitar a vida do desenvolvedor, deixando o código limpo, organizado e manutenível. Por meio do Eloquent é possível realizar as ações do Model de forma bastante organizada e reduzida. É uma solução de implementação ActiveRecord para trabalhar com banco de dados através da classe Model. De maneira simples é possível recuperar uma lista numa tabela do banco em apenas uma linha e passar para View por uma array. Na Figura 10 é um do exemplo das várias aplicações do método neste estudo.

```
for($i = 1; $i < 12; $i++) {
    $search[] = Firewall::select('comand', 'active', 'rule')
        ->where('rule',$i)
        ->first();
}
return view( view: 'functions.firewall', compact( varname: 'search'));
```

Figura 10. Retorno do Controller. Autoria própria.5.3.7 Views

Nas Views ficam o código HTML com o front-end que o usuário visualizará. Neste estudo foi empregado o Blade, ele permite o reaproveitamento de código a partir da criação de páginas mestre alterando apenas o conteúdo variável de cada página. Para isso é necessário que na página mestre a inserção da tag "@yield", onde o conteúdo dinâmico será inserido. É importante o uso do Blade e da página mestre, deixando a codificação limpa e não havendo a necessidade de importar as bibliotecas em toda página conforme exemplo na figura 11.

```
@extends('layouts.app')

@section('content')
```

Figura 11. Funcionalidade do Blade. Autoria própria.

Outra grande vantagem do emprego do Blade não restringe o uso da linguagem nas páginas. Cada View é compilada e armazenada em cache até que sofra uma modificação, deixando assim os templates muito mais leves.

5.3.8 IPTABLES

Neste subcapítulo será descrita a integração do iptables, que foi apresentado no capítulo 3.2, suas funções e como é executada as regras do firewall neste estudo desenvolvido.

5.3.8.1 FUNÇÕES

Para este estudo foi necessário fazer uma pesquisa entre os fóruns de segurança de rede sobre as principais funções recomendadas para os firewalls. No estudo ficou determinado que colocar regras fixas seriam mais interessantes que apenas usar ferramentas para criação das regras, já que se destina a usuários com pouca ou nenhuma experiencia. Entre as regras inclusas as mais básicas estão na figura 12.

Figura 12. Funcionalidades base do firewall. Autoria própria.

Na primeira opção é habilitado o redirecionamento de pacotes no arquivo “ip_forward”, é necessário para que os pacotes passem de uma interface para outra. A segunda ativa a comunicação interna do servidor, a terceira lê a lista de portas criada pelo usuário e libera o uso das mesmas e a última bloqueia lista de sites criada pelo usuário conforme a figura 13.

Figura 13. Principais problemas na rede. Autoria própria

Para essas regras foi necessário o estudo de segurança de redes e tomado por base o livro Segurança máxima para Linux (Anonymus, 2000), que define uma série de preocupações que devemos ter com a nossa rede, tais como:

Scanners de rede

Os scanners de rede os hosts de rede como um cracker, investigando os serviços e portas disponíveis, procurando fraquezas mais conhecidas que os invasores possam explorar. Com scanners é possível colher informações do servidor e para que isso não seja uma vulnerabilidade do nosso sistema é necessário bloquear no nosso firewall, se dando a partir da regra na figura 14.

```
#Bloqueio de scanners ocultos (Shealt Scan)
$IPTABLES -A FORWARD -p tcp --tcp-flags SYN,ACK, FIN, -m limit --limit 1/s -j ACCEPT
```

Figura 14. Shealt Scan. Disponível em Fórum Viva Linux.

Death Ping ou Ping da Morte

É um dos ataques a rede mais antigos, que antigamente visava criar um protocolo IP com mais de 64kb e ser enviado para um sistema sem proteção, provocando falhas no sistema. Atualmente os sistemas não são mais vulneráveis a esse tipo de ataque, mas é possível a partir de um script manter um fluxo contínuo de pacotes menores que 64kb deixando a rede instável.

Para realizar o bloqueio do ping da morte a partir do seguinte comando da figura 15.

```
#Bloqueio ping da morte
echo "0" > /proc/sys/net/ipv4/icmp_echo_ignore_all
$IPTABLES -N PING-MORTE
$IPTABLES -A INPUT -p icmp --icmp-type echo-request -j PING-MORTE
$IPTABLES -A PING-MORTE -m limit --limit 1/s --limit-burst 4 -j RETURN
$IPTABLES -A PING-MORTE -j DROP
```

Figura 15. Ping da Morte. Disponível em Fórum Viva Linux.

Syn Flood

O ataque Syn Flood é um ataque de rede por saturação explorando o mecanismo de aperto de mão em três tempos do protocolo TCP. Quando o usuário estabelece uma conexão com um servidor, o usuário envia um pedido Syn, com o servidor respondendo com um Syn/Ack e o cliente responde de volta com um Ack. Uma conexão TCP só se concretiza a partir desses três passos. O ataque Syn consiste em enviar um número muito grande de requisições Syn de um IP inexistente ou inválido, assim o servidor não recebe o pacote Ack de volta, então coloca em uma pilha as requisições sem resposta trazendo instabilidade ao servidor. Para fechar a vulnerabilidade é incluso a regra na figura 16.

```
#bloquear ataque do tipo SYN-FLOOD
echo "0" > /proc/sys/net/ipv4/tcp_syncookies
$IPTABLES -N syn-flood
$IPTABLES -A INPUT -i $WAN -p tcp --syn -j syn-flood
$IPTABLES -A syn-flood -m limit --limit 1/s --limit-burst 4 -j RETURN
$IPTABLES -A syn-flood -j DROP
```

Figura 16. Bloqueio Syn-Flood. Disponível em Fórum Viva Linux.

SSH por força bruta

É um ataque de força bruta testando as possibilidades de senhas uma a uma para descobrir a senha. Esse tipo de ataque diminui a performance do servidor e pode levar a instabilidades, além do risco da invasão. Para isso há ferramentas para esse controle, ou podemos apenas restringir as tentativas de acesso, para que o servidor não se sobrecarregue.

```
#Bloqueio de ataque ssh de força bruta
$IPTABLES -N SSH-BRUT-FORCE
$IPTABLES -A INPUT -i $WAN -p tcp --dport 22 -j SSH-BRUT-FORCE
$IPTABLES -A SSH-BRUT-FORCE -m limit --limit 1/s --limit-burst 4 -j RETURN
$IPTABLES -A SSH-BRUT-FORCE -j DROP
```

Figura 17. Bloqueio SSH por força bruta. Disponível em Fórum Viva Linux.

Anti-Spoofing

É um ataque que mascara os pacotes IP utilizando endereços remetentes falsificados. Há mais de um tipo de ataque spoofing: Arp, DNS e IP. No ataque por IP, o invasor usa um IP de fonte confiável para enviar mensagens, no ataque DNS, o atacante usa o DNS para redirecionar a vítima para outro endereço a fim de roubar informações e o ataque Arp, o invasor captura todos os pacotes que iriam chegar ao roteador. Para isso é indispensável que o firewall bloqueie pacotes que não iniciem com o IP da rede conforme regras na figura 18.

```
#bloqueio Anti-Spoofings
$IPTABLES -A INPUT -s 10.0.0.0/8 -i $WAN -j DROP
$IPTABLES -A INPUT -s 127.0.0.0/8 -i $WAN -j DROP
$IPTABLES -A INPUT -s 172.16.0.0/12 -i $WAN -j DROP
$IPTABLES -A INPUT -s 192.168.1.0/16 -i $WAN -j DROP
```

Figura 18. Anti-Spoofings. Disponível em Fórum Viva Linux.

Para o funcionamento das regras é montado um script a partir das regras que o usuário irá habilitar, as regras estão armazenadas em um banco de dados e irão fazer parte do script. Quando o usuário mandar salvar as alterações realizadas nas regras, salvará em banco de as regras estão ativas ou inativas, o controller irá verificar as regras que estão ativas e as colocará no script que será iniciado. O script é montado a partir da gravação de um arquivo, que coleta as informações das escolhas do usuário e insere as opções no script. Utilizado a função “fopen” de acordo com a figura 19.

```php
$array = Firewall::where('active',"on")
    ->pluck('comand')
    ->toArray();


$filename = '/etc/init.d/firewall';
if (is_writable($filename)) {
    if (!$handle = fopen($filename, mode: "w")) {
        return "Cannot open file ($filename)";
        exit;
    }
    $fim = '';
    // insere o valor do array com quebra de linha!
    foreach ($array as $teste => $value){
        $fim .= ($value.PHP_EOL);
    }
    if (fwrite($handle, $fim) === FALSE) {
        return "Cannot write to file ($filename)";
        exit;
    }
      return "Success, wrote ($write1) to file ($filename)";

    fclose($handle);

} else {
    return "The file $filename is not writable";
}
```

Figura 19. Método de escrita do script. Autoria própria

Esse script é salvo na pasta “etc/init.d”, nas distribuições Ubuntu 14.04 à 18.04 e Debian 8, ganha permissões de execução e cria um link simbólico para iniciar junto com o sistema. Isso é necessário, já que as chains são gravadas em memória e são perdidas todas as vezes que o servidor for reiniciado. Os comandos de tornar o script executável e de agendamento no boot são vistos na figura 20.

```php
shell_exec( cmd: 'chmod +x /etc/init.d/firewall');
shell_exec( cmd: 'ln -s /etc/init.d/firewall /etc/rc2.d/S99firewall');
```

Figura 20. Execução de comandos do sistema. Autoria própria

A sequência de funcionamento do sistema é visualizada no UML da Figura 21, a View chama a rota com o método requerido, acessa o controller que processa a requisição da rota, acessa o banco de dados, retorna os dados para serem apresentados na View. O usuário fará as alterações na View, serão enviadas pela rota via POST, o controller receberá, salvará as informações no banco de dados, gerará o script com as regras determinadas pelo usuário, executa o script pelo terminal do Linux e retorna para View mensagem com sucesso da atualização das regras.

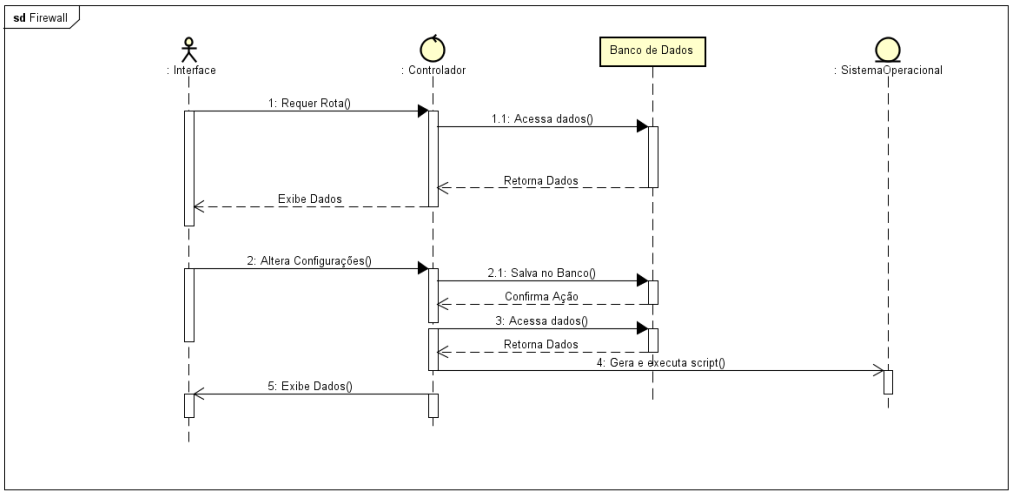


Figura 21. Diagrama de Sequência do Firewall. Autoria própria.

5.3.9 SQUID

Squid é um servidor proxy para vários protocolos, é de fácil configuração e é possível fornecer segurança e anonimato ao usuário. É muito útil para fazer controle de acesso dos usuários e é mais uma ferramenta de segurança para complementar o firewall. Toda a configuração do Squid se dá no arquivo “squid.conf”, onde é necessário adequar para a realidade das necessidades do usuário. As configurações mais importantes devem ser preenchidas, conforme o formulário criado e representado na figura 22.

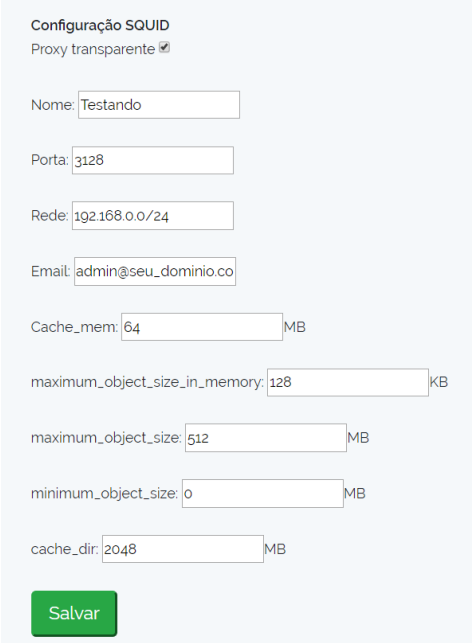


Figura 22. Tela configuração Squid. Autoria própria.

O formulário já puxa as informações contidas no squid.conf e recupera para o usuário, a partir do controller. Dessa forma o usuário pode alterar apenas um campo sem perder nenhum dado. No controller varre o arquivo de configurações do Squid enviando à View as informações necessárias. Conforme a figura 23, é possível visualizar o método de pesquisa de string procurando equivalências no texto e armazenando em um array que será retornado via "compact".

```
public function index(){

    $lines = file( filename: 'C:\xampp\htdocs\projeto_tcc\public\squid.txt');
    $pesquisas = ['http_port', 'visible_hostname', 'acl rede_local src', 'cache_mgr','cache_mem', 'in_memory',
        'maximum_object_size', 'minimum_object_size', 'cache_dir ufs /var/spool/squid', 'transparent'];
    $flag = 0;

    foreach ($pesquisas as $pesquisa) {
        foreach ($lines as $line) {
            if (strpos($line, $pesquisa) !== false) {

                if($pesquisa == 'acl rede_local src' || $pesquisa == 'cache_dir ufs /var/spool/squid'){
                    $test = preg_split( pattern: '/ /', $line, limit: 5);
                    $limpa[] = str_ireplace( search: '"', replace: '', $test[3]);
                }else {
                    if($pesquisa == 'transparent'){
                        $limpa[] = 'on';
                    }else {

                        if($pesquisa == 'maximum_object_size' && $flag < 1) {
                            $flag ++;
                        }else {
                            $test = preg_split( pattern: '/ /', $line, limit: 5);
                            $limpa[] = str_ireplace( search: '"', replace: '', $test[1]);
                        }
```

Figura 23. Método de varrer o arquivo squid.conf. Autoria própria.

Passado pelo controller, o array com as informações é recuperada pela View conforme a figura 24, sendo exibida em tela em um "input text". Como auxílio ao usuário, há uma pequena informação explicando o que o campo significa, isso é para agilizar a configuração, sem a necessidade da leitura da documentação da ferramenta.

```
<label data-toggle="tooltip" data-placement="top" title="Faixa de endereço de IP da sua rede. Ex: 10.0.0.0/24">Rede: </label>
<input type="text" name="rede" value="{{$limpa[2]}}"><br><br>
```

Figura 24. Recuperação das informações na View. Autoria própria.

Para apresentação da dica foi utilizado um recurso do Bootstrap, um "data-toggle" com o texto explicativo. Visualização é limpa e bastante prática, conforme segue na Figura 25.

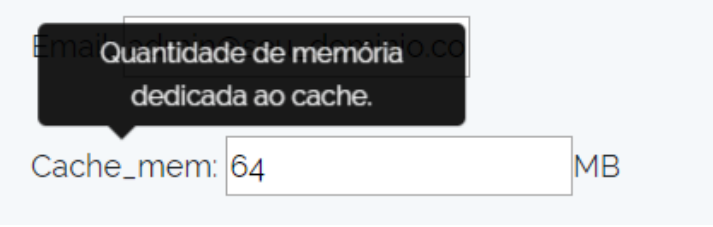


Figura 25. Um tooltip contendo as informações do campo. Autoria própria.

5.3.10 SARG

O Sarg (Squid Analisys Report Generator), é um interpretador de logs do Squid, onde é possível acompanhar os registros de acessos dos usuários e controlar os abusos que ocorrem. Também serve como verificador das regras implementadas, de modo que seja possível incrementar o trabalho realizado com o Squid e o iptables em regular o acesso a conteúdo indevido.

Os acessos são organizados por usuário, caso o Squid esteja requerendo autenticação, ou por IP das máquinas de acesso, mostrando o conteúdo acessado, tempo de permanência na página, quantidade de dados transmitidos e ainda as tentativas de acesso bloqueadas pelos filtros.

A configuração do Sarg é bastante simples, sendo necessário apenas a atenção para onde os logs do Squid estão sendo salvos e direcionar que o Sarg leia esses arquivos. Para que os logs sejam gerados é necessário aplicar o comando “sarg” no terminal como usuário root, mas há formas de gerar automaticamente os logs inserindo no crontab um script com os comandos necessários.

No estudo apresentado, a configuração do Sarg se resume em configurar o nome que da organização na página de apresentação dos logs e escolher qual será a frequência que os logs serão gerados, conforme a figura 26.

Figura 26. Página de configuração do SARG. Autoria própria.

O usuário após salvar enviará os dados ao controller, que efetuará a escrita do sarg.conf e também irá gerar o arquivo com o script a ser executado com a frequência escolhida para confecção dos logs do Squid. Na figura 27, representa um exemplo do comando acrescentado ao crontab e do script que ele executará todos dias as 21 horas.

```
if (is_writable($crontab)) {
    if (!$handle = fopen($crontab, mode: "a+")) {
      return "Cannot open file ($filename)";
        exit;
    }
    if (fwrite($handle, $rotina) === FALSE) {
        return "Cannot write to file ($crontab)";
        exit;
    }
  return "Success, wrote ($write) to file ($filename)";

    fclose($handle);

} else {
  return "The file $filename is not writable";
}
$sdiario =
            'INICIO=$(date --date \"0 days ago\" +%d/%m/%Y)' . "\r\n" .
            'FIM=$(date --date \"0 day ago \" +%d/%m/%Y)' . "\r\n" .
            'sarg -f /etc/sarg/sarg.conf -d $INICIO-$FIM -p -x -z' . "\r\n";
```

Figura 27. Comando adicionado ao crontab e o script a ser executado.

5.3.11 DHCP

É uma ferramenta de rede indispensável nos dias de hoje, pois com ela é possível atribuir endereços IP às máquinas ligadas no servidor para que eles possam se comunicar. O DHCP (Dynamic Host Configuration Protocol) é um protocolo de rede a nível aplicativo que fornece automaticamente informações de endereçamento de rede, endereço IP, máscara de rede, gateway padrão, servidores de nomes e outras informações ao host.

Funciona de maneira bem simples, ele responde aos pacotes de broadcast das máquinas, enviando um pacote com IP disponível e as demais informações da rede. Esses pacotes de broadcast são endereçados de maneira global à todas as portas de um switch, diferentemente de pacotes que possuem um endereço específico. Periodicamente o DHCP verifica se o usuário está ativo para renovação de IP, isso ajuda para que os IP da rede não se esgotem. Para isso é necessário a configuração da ferramenta, conforme formulário na figura 28.

O servidor utilizado neste estudo é o ISC DHCP, é o mais usado pelas distribuições Linux e foi desenvolvido pela Internet System Consortium, também desenvolvedora de outras ferramentas, como BIND e o NTPD.

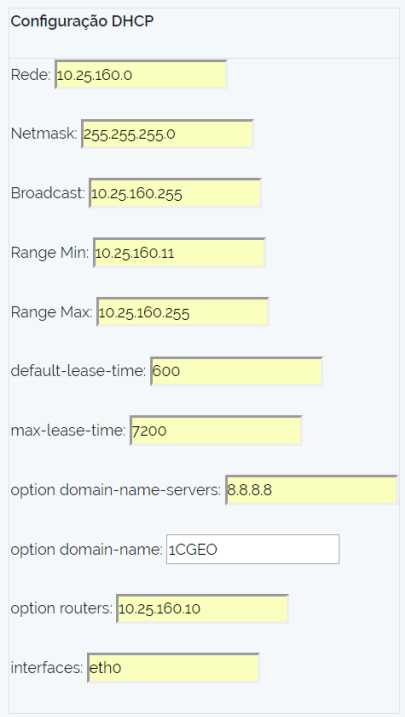


Figura 28. Página de configuração DHCP, exemplo de configuração de rede. Autoria própria.

Para configuração do servidor DHCP é necessário identificar qual será a sub-rede, equivalente ao campo “Rede” do formulário, o Netmask representa a máscara de rede, configuração que determina a qual rede o computador pertence. O campo broadcast determina o último da IP da rede, range máximo e mínimo restringem os IPs que serão distribuídos pelo servidor, assim é possível salvar alguns IPs para configuração de serviços da organização, como por exemplo o host de sistemas ou outros servidores.

A opção de “default-lease-time” controla o tempo de renovação dos IP, o padrão vem em 600 segundos, no caso a cada dez minutos o servidor faz a verificação se a máquina está ativa. O campo seguinte “max-lease-time” controla o tempo máximo que uma máquina pode usar um IP quando a rede estiver congestionada.

As opções de “option domain-name” e “option routers” contêm os servidores DNS que serão usados, devem ser separados por vírgula caso haja mais de um endereço e o outro campo indica o gateway da rede, que não necessariamente é o mesmo servidor onde está rodando o DHCP.

O envio dos dados segue a configuração das ferramentas anteriores, os dados do formulário são passados via POST pela rota ao controller, que recebe os dados e cria o arquivo de configuração contendo as informações colhidas no formulário. Na figura 29 demonstra como o controller escreve o arquivo de configuração.

```
$test = "#Domínio configurado no BIND \r\n".
        "#option domain-name ".$domainname.";\r\n".
        "default-lease-time ". $dleasetime ."; #controla o tempo de renovação do IP\r\n".
        "max-lease-time ".$mleasetime ." #determina o tempo que cada máquina pode usar um determinado IP. \r\n".
        "log-facility local7; \r\n \r\n".

        "subnet ".$rede." netmask ".$netmask." { #Define sua sub-rede 192.168.1.0 com a máscara 255.255.255.0 \r\n".
            "range ".$rangemin." ".$rangemax.";  #faixa de IPs que o cliente pode usar.\r\n".
             "option routers ".$router."; #Este é o gateway padrão (neste caso).\r\n".
             "option broadcast-address ".$broadcast." ; #Essa linha é o endereço de broadcast (neste caso).\r\n \r\n".

        "#Aqui você coloca os servidores DNS de terceiros ou seu DNS próprio configurado no BIND. Nesse caso coloquei o DNS do Google.\r\n".
            "option domain-name-servers ".$domainservers."; \r\n".
           "}\r\n \r\n";
```

Figura 29. Configuração no controller do dhcpd.conf.

Após a configuração completa é necessário determinar qual placa de rede fará a escuta da rede local para o DHCP, essa configuração acontece no arquivo “/etc/default/isc-dhcp-server”, onde é necessário inserir a interface da placa de rede. No estudo em questão, tal configuração é realizada pelo comando “dhcpd -cf /etc/dhcpd.conf -d $interfaces” e o servidor DHCP é reiniciado automaticamente para que as alterações tenham efeito.

5.3.12 DNS

Um servidor DNS (Domain Name System) faz a resolução de nomes em sua máquina ou nas máquinas da rede, então todas as máquinas da rede com o DNS configurado para um determinado servidor irá perguntar a ele o endereço IP do site que você quer navegar, ou em outra máquina da rede. Para redes de organizações é importante o DNS interno para que seja possível resolver os nomes das máquinas e servidores da rede, senão os usuários terão que decorar os IPs das máquinas ao invés de apenas nomes.

Para essa solução será utilizado o Bind9, servidor de DNS mais popular das distribuições Linux. São necessários a configurações de alguns arquivos e para isso será usado uma página de configuração semelhante as ferramentas anteriores, conforme a figura 30 apresenta.

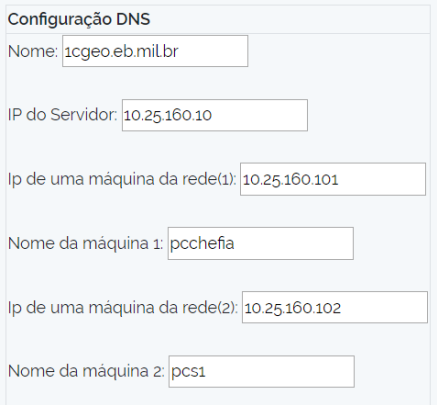


Figura 30. Formulário de configuração DNS. Autoria própria.

O nome do será o domínio a ser configurado, deverá ser seguido pelo nome complementado por ".com.br" como no exemplo, e será inserido no "named.conf". Na etapa seguinte é necessário criar e popular o arquivo de zona que foi determinado anteriormente, com o nome do domínio seguido por ".dns", onde deverá ser inserido parâmetros para o servidor e adicionar os registros com os IP e nomes das máquinas.

5.3.13 USUÁRIOS

O Sistema permite a criação e edição de usuários conforme a necessidade, não há hierarquias de usuários, pois o sistema deverá ser usado apenas pelo administrador da rede. A criação do usuário é feita de maneira simples, apenas preencher o formulário com nome, e-mail, senha e confirmação da senha. O e-mail é uma chave única e não pode ser repetido e a senha é sofre encriptação para ser armazenada no banco de dados via método "bcrypt" do Laravel, que transforma a senha em uma hash. O formulário é demonstrado na figura 31.

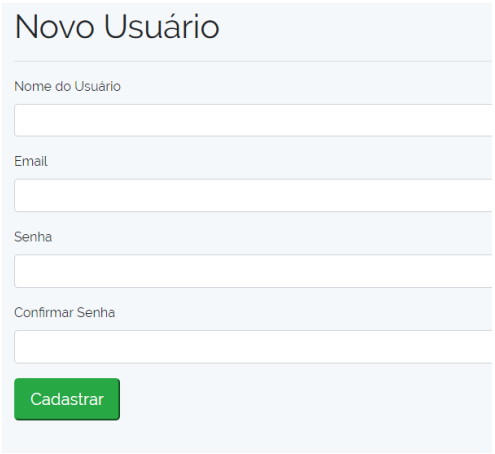


Figura 31. Formulário de cadastro de novo usuário. Autoria própria.

Os campos do formulário são validados no back-end, via “Request”, que valida se os campos atendem as especificações mínimas, como por exemplo se o e-mail é único, se a senha possui um número mínimo de caracteres e se o nome não excede o número máximo de caracteres. É possível passar vários parâmetros como regras a examinar os dados antes de serem adicionados ao banco, conforme é possível identificar na figura 32. Além de criar novos usuários é possível também editar ou deletar os usuários existentes.

```php
public function rules()
{
    return [
        'name' => 'required|string|max:255',
        'email' => 'required|string|email|max:255|unique:users',
        'password' => 'required|string|min:6|confirmed',
    ];
}
```

Figura 32. Exemplo de regras de validação back-end. Autoria própria.

6. CASOS DE USO

Para demonstração e teste do trabalho desenvolvido, foi aplicado na rede local do 1º Centro de GeoInformação, em um servidor Dell R720, que faz roteamento da rede e serve como servidor de aplicações. Para efeito de estudo, apenas as regras de iptables foram aplicadas, pois não havia a possibilidade de interromper os outros serviços. As outras ferramentas foram utilizadas apenas gerando os arquivos de configuração, sem alterar os originais.

Para testes foi utilizado o Debian 8 como sistema operacional e as ferramentas nas seguintes versões: iptables 1.6.2, de 02 de fevereiro de 2018; Squid 3.5, versão estável para Debian Streech; ISC DHCP versão 4.4.1; SARG atualizado em 14 de janeiro de 2018; pacote Samba 4.7.6 de 13 de março de 2018; e MariaDB 10.1.18 de 30 de setembro de 2016.

Para a realização dos testes foram empregados dois soldados recrutas com pouca experiencia e conhecimento em administração de redes. As instruções dadas foram seguir um questionário para utilização das ferramentas e após utilização o preenchimento para um feedback para futuras atualizações.

No questionário foi pedido que realizassem configurações por conta própria e que fosse tirado printscreen das páginas e dos arquivos gerados, conforme as figuras 33 e 34.

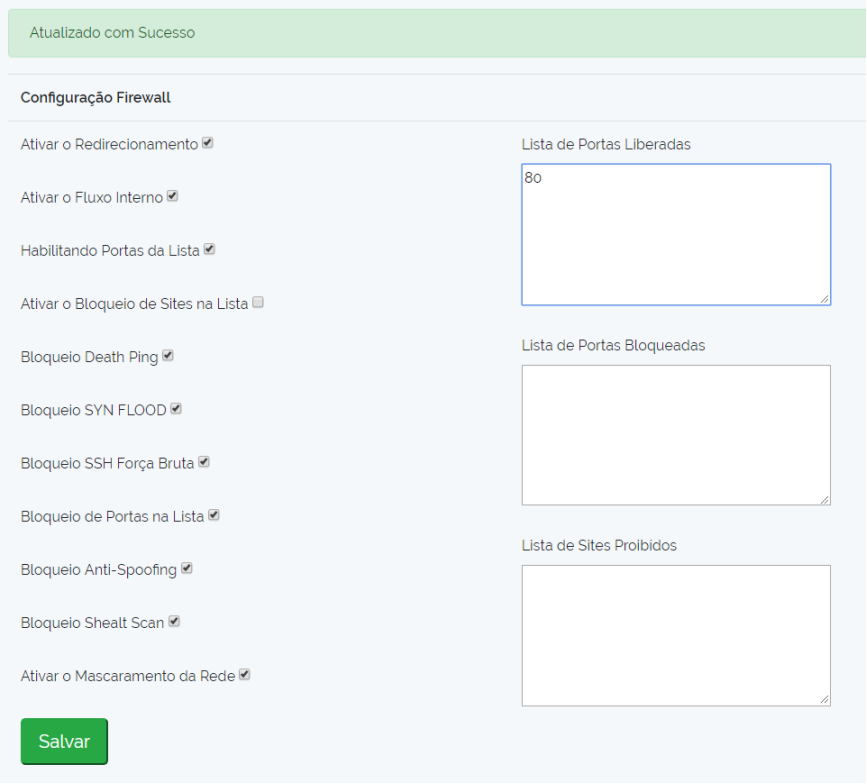
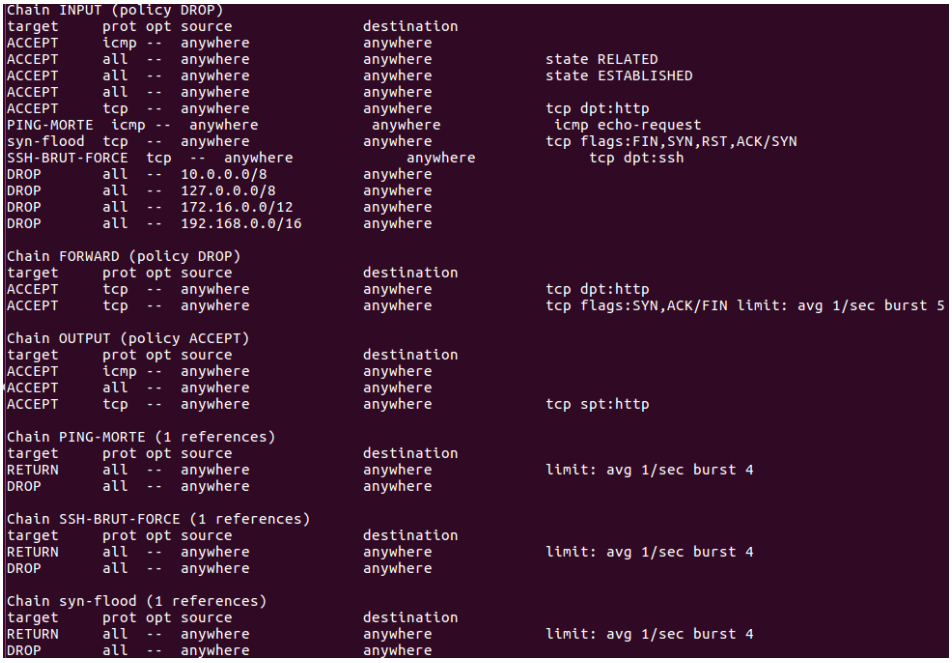


Figura 33. Configuração realizada. Autoria própria.

```
$IPTABLES -F
$IPTABLES -F INPUT
$IPTABLES -F OUTPUT
$IPTABLES -F FORWARD
$IPTABLES -t mangle -F
$IPTABLES -t nat -F
$IPTABLES -X
$IPTABLES -P INPUT DROP
$IPTABLES -P OUTPUT ACCEPT
$IPTABLES -P FORWARD DROP
echo "1" > /proc/sys/net/ipv4/ip_forward
$IPTABLES  -I INPUT -i lo -j ACCEPT
$IPTABLES -I OUTPUT -o lo -j ACCEPT
for i in `cat $PORTSLIB`; do
$IPTABLES -A INPUT -p tcp --dport $i -j ACCEPT
$IPTABLES -A FORWARD -p tcp --dport $i -j ACCEPT
$IPTABLES -A OUTPUT -p tcp --sport $i -j ACCEPT
done
$IPTABLES -I INPUT -m state --state ESTABLISHED -j ACCEPT
$IPTABLES -I INPUT -m state --state RELATED -j ACCEPT
$IPTABLES -I OUTPUT -p icmp -o $WAN -j ACCEPT
$IPTABLES -I INPUT -p icmp -j ACCEPT
echo "0" > /proc/sys/net/ipv4/icmp_echo_ignore_all
$IPTABLES -N PING-MORTE
$IPTABLES -A INPUT -p icmp --icmp-type echo-request -j PING-MORTE
$IPTABLES -A PING-MORTE -m limit --limit 1/s --limit-burst 4 -j RETURN
$IPTABLES -A PING-MORTE -j DROP
echo "0" > /proc/sys/net/ipv4/tcp_syncookies
$IPTABLES -N syn-flood
$IPTABLES -A INPUT -i $WAN -p tcp --syn -j syn-flood
$IPTABLES -A syn-flood -m limit --limit 1/s --limit-burst 4 -j RETURN
$IPTABLES -A syn-flood -j DROP
$IPTABLES -N SSH-BRUT-FORCE
$IPTABLES -A INPUT -i $WAN -p tcp --dport 22 -j SSH-BRUT-FORCE
```

Figura 34. Parte do script gerado a partir das configurações. Autoria própria.

Após enviado o formulário com as configurações selecionadas, o script foi gerado e executado pelo sistema, retornando o resultado apresentado na figura 35, no firewall, a partir do comando iptables -L.



```
Chain INPUT (policy DROP)
target     prot opt source               destination
ACCEPT     icmp --  anywhere             anywhere
ACCEPT     all  --  anywhere             anywhere             state RELATED
ACCEPT     all  --  anywhere             anywhere             state ESTABLISHED
ACCEPT     all  --  anywhere             anywhere
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:http
PING-MORTE  icmp --  anywhere             anywhere             icmp echo-request
syn-flood  tcp  --  anywhere             anywhere             tcp flags:FIN,SYN,RST,ACK/SYN
SSH-BRUT-FORCE  tcp  --  anywhere             anywhere             tcp dpt:ssh
DROP       all  --  10.0.0.0/8           anywhere
DROP       all  --  127.0.0.0/8          anywhere
DROP       all  --  172.16.0.0/12        anywhere
DROP       all  --  192.168.0.0/16       anywhere

Chain FORWARD (policy DROP)
target     prot opt source               destination
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:http
ACCEPT     tcp  --  anywhere             anywhere             tcp flags:SYN,ACK/FIN limit: avg 1/sec burst 5

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     icmp --  anywhere             anywhere
ACCEPT     all  --  anywhere             anywhere
ACCEPT     tcp  --  anywhere             anywhere             tcp spt:http

Chain PING-MORTE (1 references)
target     prot opt source               destination
RETURN     all  --  anywhere             anywhere             limit: avg 1/sec burst 4
DROP       all  --  anywhere             anywhere

Chain SSH-BRUT-FORCE (1 references)
target     prot opt source               destination
RETURN     all  --  anywhere             anywhere             limit: avg 1/sec burst 4
DROP       all  --  anywhere             anywhere

Chain syn-flood (1 references)
target     prot opt source               destination
RETURN     all  --  anywhere             anywhere             limit: avg 1/sec burst 4
DROP       all  --  anywhere             anywhere
```

Figura 35. Resultado após execução do script do firewall. Autoria própria.

A segunda etapa consiste em liberar as máquinas na rede para que acesse a rede externa, conforme a figura 36 demonstra que por default elas estão todas desativadas e é necessário que o administrador faça a ativação. Por motivos de segurança, os MAC Address foram protegidos, pois se tratam de máquinas reais em produção. É criada uma lista contendo o Mac das máquinas que terão permissão e é reiniciado o firewall.

Máquinas na Rede

IP	MAC	Autorização
10.25.163.30	[redacted]	Ativado
10.25.162.101	[redacted]	Desativado
10.25.163.55	[redacted]	Desativado
10.25.163.1	[redacted]	Desativado
10.25.160.7	[redacted]	Desativado
10.25.163.148	[redacted]	Desativado
10.25.163.173	[redacted]	Desativado

Figura 36. Acesso para a rede externa. Autoria própria.

A Terceira etapa consiste na configuração do Squid. Para essa configuração foi usada uma máquina virtual para exemplificação, como explicado não havia a possibilidade de parar a

rede para testes. Após a configuração ser salva, é alterado o squid.conf e é necessária a reinicialização da ferramenta e como resultados temos após executar o comando “service squid status” o que demonstra a figura 37, que o servidor está devidamente funcionando.

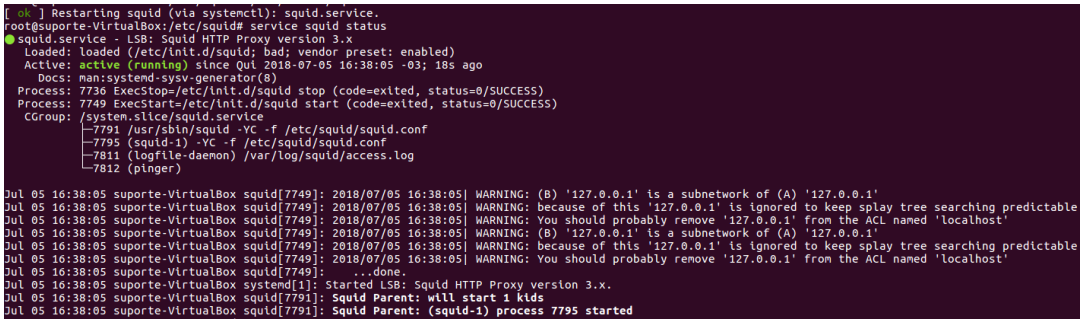


Figura 37. Representando status do Squid. Autoria própria

Como complemento para monitorar os logs do Squid, está sendo usado a ferramenta Sarg, uma ferramenta de leitura de logs do Squid. Nela é possível identificar diversas informações de acesso, quanto tempo o usuário ficou na página, o quanto de transmissão de dados realizou e outros detalhes. A figura 38 representa um exemplo realizado neste teste, onde o usuário configurou o proxy no navegador e acessou o Google. O relatório fica acessível

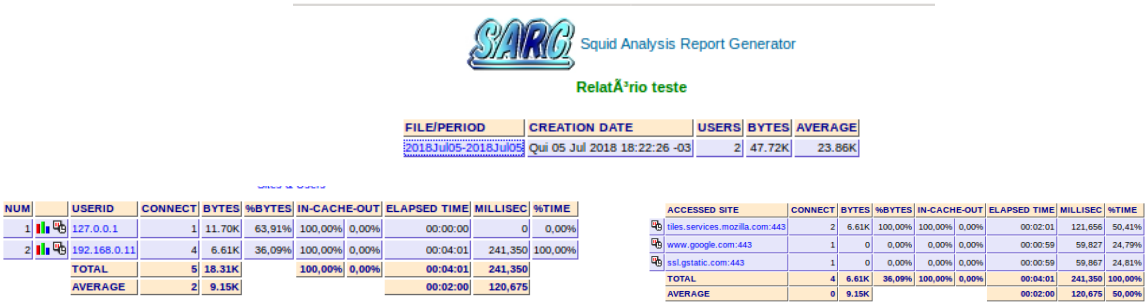


Figura 38. Exemplo de relatório do Sarg. Autoria própria.

A próxima etapa foi a configuração do servidor DHCP, foi instalado em uma máquina virtual o servidor isc-dhcp, o mais usado entre as distribuições Linux e configurado conforme a vontade do usuário de teste conforme figura 39.

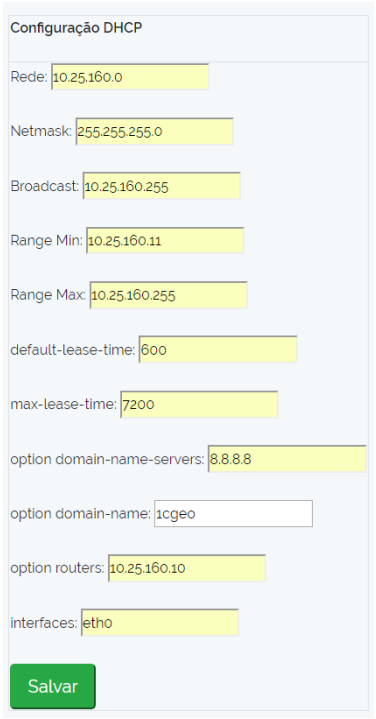


Figura 39. Exemplo de configuração. Autoria própria.

Como resultado o arquivo de configuração foi atualizado e o servidor DHCP reiniciado. Houve erros na configuração do servidor e servidor não funcionou corretamente, sendo um dos bugs com necessidade de concerto. É possível verificar os erros na figura 40.

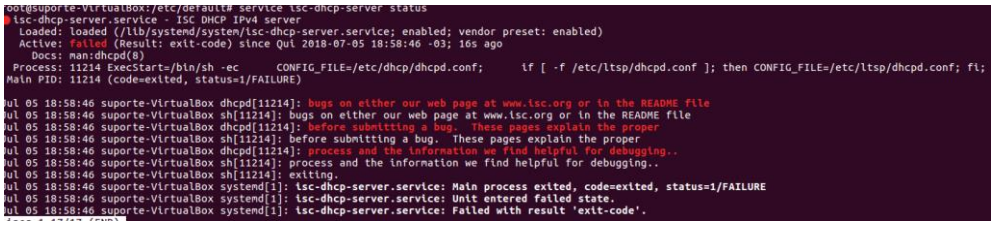


Figura 40. Erro na configuração do servidor DHCP. Autoria própria.

A última etapa se dá na criação de usuários para utilização do sistema. O cadastro ocorre de forma bastante simples conforme demonstra a figura 41.

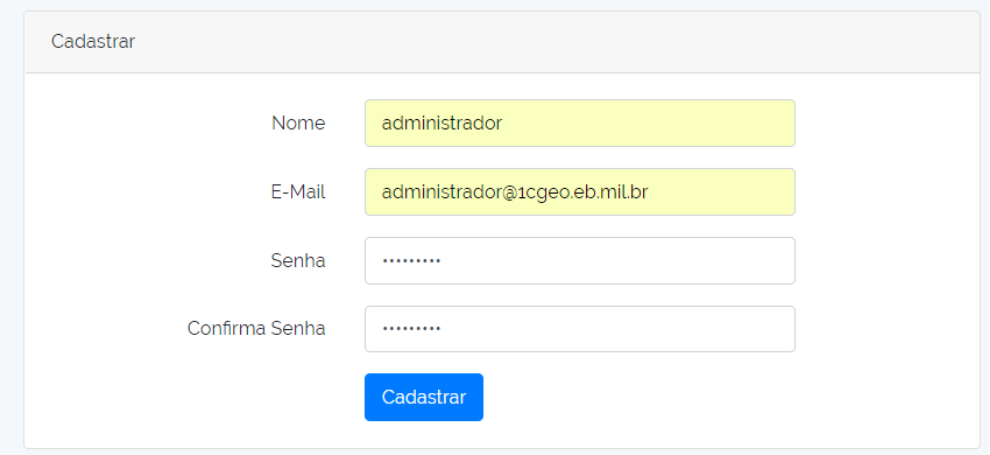


Figura 41. Formulário de cadastro de usuário. Autoria própria.

Após o cadastro o usuário retorna para tela de lista de usuários e recebe a notificação que o usuário foi criado com sucesso, conforme a figura 42.

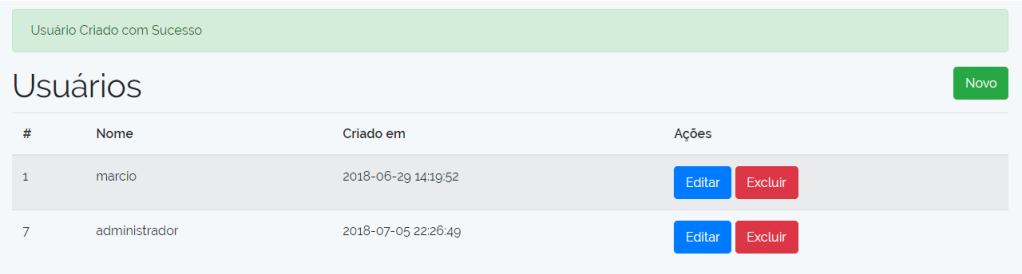


Figura 42. Lista de usuários. Autoria própria

Após os testes houve uma breve conversa com os usuários sobre a experiência com o sistema. A recepção foi positiva, de forma que os usuários conseguiram configurar ferramentas suficientes para administrar rede sem conhecimento prévio das ferramentas. Houve retorno de sugestões de posicionamento dos formulários e botões e alterações nas cores dos campos. Foi encontrado falhas na configuração do servidor DHCP, resultando em falha seu uso.

Foi levantada a questão da instalação das ferramentas, ponto que não foi necessário no teste, pois as ferramentas já estavam instaladas, tornando a falta de um manual um ponto menos importante no teste, mas que será necessário para novas instalações.

7. CONCLUSÃO

De acordo com o apresentado neste trabalho de conclusão de curso, há diversos benefícios no uso do sistema desenvolvido e possui potencial para ser uma boa solução para uma rede de pequeno porte para um usuário com pouca experiencia em administrar a rede.

Este trabalho permitiu o aprofundamento no conhecimento das ferramentas de segurança empregadas como por exemplo o iptables e o Squid, além do conhecimento adquirido na programação com PHP aliada com Framework Laravel.

Há uma necessidade cada vez mais crescente em manter a informação segura e para isso é necessário o uso correto das ferramentas de controle da rede, tais como um firewall e um proxy, mas é um assunto que requer uma experiencia maior do que a maioria dos usuários possui e para isso esse sistema foi construído, compartilhar conhecimento.

A partir do teste realizado, pode-se observar que o Braço Forte atende as necessidades primárias do usuário, diminuindo a quase zero a necessidade de intervenção em um terminal de comando para se configurar uma rede. Assim o software se torna uma alternativa para um público que não se interessa pelas soluções tradicionais, onde é necessário a instalação de um servidor dedicado para servir de roteador e máquina de firewall.

Para implementações futuras, pretende-se que haja formas mais flexíveis de inserção de regras no iptables, como em ferramentas existentes no mercado, onde é possível bloquear os pacotes a partir de uma origem/porta à um destino/porta. Implementação de um log para registrar as alterações realizadas pelos usuários, como forma de auditoria da administração da rede. A inserção de módulos para identificação de invasão, como por exemplo o Snort, concluir o módulo do Samba para compartilhamentos de pastas entre os usuários, criação de um servidor DNS a partir do BIND9, realizar a configuração de firewall para as máquinas da rede e o fechamento do sistema como um pacote Debian, facilitando na hora de instalação, com a resolução da instalação das dependências automaticamente.

8. REFERÊNCIAS BIBLIOGRÁFICAS

ALSHANETSKY, I. Usage Statistics. Dezembro 2010. Disponível em: <http://webadvent.org/2010/usage-statistics-by-ilia-alshanetsky.html> Acessado em: 02 de julho de 2018, as 14:00.

ANONIMOUS. Segurança Máxima pra Linux Rio de Janeiro: Campus, 2000.

DAPPER, Marcelo. História do JavaScript. Novembro 2016. Disponível em: <https://devheroes.io/javascript-s01e01-historia-javascript/> Acessado em: 02 de julho de 2018, as 14:00.

Documentação Mozilla. JavaScript. Outubro 2017. Disponível em: <https://developer.mozilla.org/pt-BR/docs/Web/JavaScript> Acessado em: 02 de julho de 2018, as 14:00.

DELFINO, Pedro. pfSense – Principais vantagens e recursos dessa poderosa ferramenta firewall. Disponível em: <http://e-tinet.com/linux/pfsense-vantagens/> Acessado em: 02 de julho de 2018, as 14:00

Documentação Oficial pfSense. Disponível em: <https://doc.pfsense.org/index.php/Main_Page#Welcome_to_the_pfSense_Documentation_site> Acessado em: 02 de julho de 2018, as 14:00.

Endian Company. Disponível em: < https://www.endian.com> Acessado em: 02 de julho de 2018, as 16:00.

Fórum Viva Linux. Disponível em: <https://www.vivaolinux.com.br> Acessado em: 02 de julho de 2018, as 14:30.

IMASTERS, Redação. JavaScript é a linguagem mais popular do GitHub. Novembro de 2014. Disponível em: <https://imasters.com.br/noticia/javascript-e-linguagem-mais-popular-github/> Acessado em: 02 de julho de 2018, as 14:00.

Internet System Consortium. Disponível em: <https://www.isc.org/> Acessado em: 02 de julho de 2018, as 15:00.

Laracasts. Disponível em: < https://laracasts.com/skills/laravel> Acessado em: 02 de julho de 2018, as 15:00.

MACUZO, Vitor. 10 motivos para considerar o pfSense como gateway da sua rede. Julho 2015. Disponível em: <https://www.escolalinux.com.br/blog/10-motivos-para-considerar-o-pfsense-como-o-gateway-da-sua-rede> Acessado em: 02 de julho de 2018, as 14:00

Manual Oficial Samba. Disponível em: <https://www.samba.org/samba/docs/SambaIntro.html> Acessado em: 02 de julho de 2018, as 15:00.

MELO, Reginaldo. Documentação Oficial BrazilFW. Outubro de 2013. Disponível em: <https://wiki.brazilfw.com.br/doku.php?id=origin_and_characteristics#software_servicos_nativos> Acessado em: 02 de julho de 2018, as 14:00.

MORIMOTO, Carlos E. Redes e Servidores Linux. Porto Alegre: Sul Editoras 2005.

MUSTAFA, Eduardo. JavaScript, 20 anos de história e construção da web. Janeiro de 2016. Disponível em: <https://imasters.com.br/front-end/javascript/javascript-20-anos-de-historia-e-construcao-da-web/?trace=1519021197&source=single> Acessado em: 02 de julho de 2018, as 14:00.

OpenWrt Project. Disponível em: < https://openwrt.org/> Acessado em: 02 de julho de 2018, as 16:00.

Plano de Migração de Software Livre. Disponível em: < https://www.governodigital.gov.br/documentos-e-

arquivos/Plano_de_Migracao_de_Software_Livre_no_MP-V1-2.pdf> Acessado em: 02 de julho de 2018, as 15:00.

RODRIGUES, Renan. jQuery: A biblioteca do JavaScript que usa a filosofia “write less, do more”! Agosto de 2010. Disponível em: <https://rcrrodrigues.wordpress.com/2010/08/19/jquery-o-framework-que-usa-a-filosofia-write-less-do-more/> Acessado em: 02 de julho de 2018, as 14:00

SILVA, Maurício Samy. JQuery. A Biblioteca do programador JavaScript. São Paulo: Novatec Editora, 2008.

Site Oficial Bootstrap. Disponível em: <http://getbootstrap.com.br/> Acessado em: 02 de julho de 2018, as 14:00.

Site Oficial MySQL. Disponível em: < https://www.mysql.com/> Acessado em: 02 de julho de 2018, as 16:30.

Site Oficial Samba. Disponível em: < https://www.samba.org/> Acessado em: 02 de julho de 2018, as 16:00.

SOUZA, Diogo. Explorando o jQuery UI. Disponível em: <https://www.devmedia.com.br/explorando-o-jquery-ui/30316> Acessado em: 02 de julho de 2018, as 14:00.

The State of the Octoverse 2017. Janeiro 2018. Disponível em: <https://octoverse.github.com/> Acessado em: 02 de julho de 2018, as 14:00

TIOBE. Index for April 2018. Disponível em: <https://www.tiobe.com/tiobe-index/> Acessado em: 02 de julho de 2018, as 14:00.