Perquirere, 17 (2): 227-240, maio/ago. 2020. *ISSN*: 1806-6399

https://revistas.unipam.edu.br/index.php/perquirere

Comparação da alta disponibilidade implementada no *PfSense* e no *Mikrotik*

Comparison of the high availability implemented in PfSense and Mikrotik

Omar Junio Antunes Vieira
Graduando do Curso de Sistemas de Informação (UNIPAM)
E-mail: omarjav@unipam.edu.br

José Corrêa Viana
Professor Orientador (UNIPAM)
E-mail: jcorrea@unipam.edu.br

Resumo: O presente artigo traçou um comparativo da implementação de alta disponibilidade entre dois sistemas operacionais, *PfSense* e *Mikrotik RouterOS*, visando a identificar qual forneceria melhor funcionalidade do sistema. Assim, foram propostas duas soluções para identificar qual teria melhor desempenho em suas funcionalidades, garantindo o melhor funcionamento dos serviços de rede, melhorando o tempo de resposta mediante a presença de falhas. Para o desenvolvimento do trabalho, foram utilizados os conceitos de redundância, *failover* e *load balance*. Todo o projeto foi implementado no *VirtualBox* e utilizou o *software Zabbix*. Concluiu-se que as duas soluções obtiveram resultados bem parecidos, porém o *PfSense* fornece um ambiente mais instável.
Palavras-chave: *PfSense*. *Mikrotik RouterOS*. *Load balance*. *Failover*. *Multi Wan*.

Abstract: This paper drew a comparison of the implementation of high availability between two operating systems, PfSense and Mikrotik RouterOS, in order to identify which would provide better system functionality. Thus, two solutions were proposed to identify which would perform better in its functionalities, guaranteeing the best functioning of the network services, improving the response time through the presence of failures. For the development of the work, the concepts of redundancy, failover and load balance were used. The entire project was implemented in VirtualBox and used the Zabbix software. It was concluded that the two solutions achieved similar results, but PfSense provides a more unstable environment.
Keywords: PfSense. Mikrotik RouterOS. Load balance. Failover. Multi Wan.

1 INTRODUÇÃO

A construção de modernos parques computacionais, com regras de negócios bem definidas, tarefas automatizadas e infraestrutura de comunicação de dados de alta velocidade, tem vários objetivos, dentre eles os principais são aumentar a lucratividade e a competitividade. Desde sistemas bancários até caixas de supermercados e padarias,

os computadores há tempos desenvolvem papéis fundamentais no cotidiano da sociedade moderna.

No entanto, como exemplo da inoperância do sistema computacional, é possível observar o caos em aeroportos devido a problemas no funcionamento do sistema que controla as aeronaves ou até mesmo a irritação de um cliente em uma fila parada de caixa de supermercado. Porém, isso ocorre não apenas em tais tipos de serviços, mas também em empresas cujo maior objetivo é exatamente a oferta de algum serviço computacional, como comércio eletrônico, notícias, hospedagem de sites Web, aplicações distribuídas etc.

Para tais ambientes, em que os sistemas de computação são um fator de grande relevância e caracterizam partes críticas do sistema global de uma organização, é necessário desenvolver plano de contingência para otimizar o desempenho das atividades do processo e da alta disponibilidade, garantindo, assim, melhor funcionamento desses sistemas e evitando eventuais problemas que acarretariam prejuízos materiais e financeiros.

A gerência de redes de computadores é um dos instrumentos utilizados para a elaboração de estratégias que visem à melhoria do nível das operações e execuções das atividades de sistemas computacionais críticos, pois inclui o oferecimento, a integração e a coordenação de elementos de hardware e software e dispõe de pessoas para monitorar e realizar ações, objetivando satisfazer às exigências operacionais de desempenho e de qualidade de serviço de sistemas computacionais.

Com isso, pode-se propor um balanceamento de carga para se obter melhor desempenho entre as interfaces de rede, melhorando o tempo de resposta, e a alta disponibilidade para se prover tolerância a falhas, garantindo, assim, a disponibilidade dos serviços de rede.

O objetivo deste trabalho foi propor duas soluções que prometem utilizar ferramentas para proporcionar a construção de um ambiente capaz de realizar balanceamento de carga, remodelando requisições entre as interfaces de rede e redundância, transferindo as tarefas de um sistema centralizado para um novo sistema computacional.

Posto isso, foi criado mecanismo de balanceamento de carga de requisições em serviços, dados e informações. Com a redundância, foi possível absorver as requisições de serviços dos clientes e distribuí-las de maneira coordenada para outra máquina, com melhoria do desempenho da prestação dos serviços envolvidos, o que garantiu a alta disponibilidade. Para alcançar esse objetivo, essas soluções abordaram os conceitos de *load balance*, *failover e* multi WAN.

Este trabalho apresenta a implementação de um ambiente de alta disponibilidade no *PfSense* e no *Mikrotik RouterOS*. As soluções foram implementadas por meio do sistema de virtualização, utilizando *Linux Ubuntu* e *Slackware*, sistema operacional e *software* livre, como clientes da rede, e *software Zabbix*, para monitoração de ativos de rede na testagem do desempenho de cada solução. Desta forma, pretendeu-se ter um *firewall* estável, seguro, com alta disponibilidade e baixo custo.

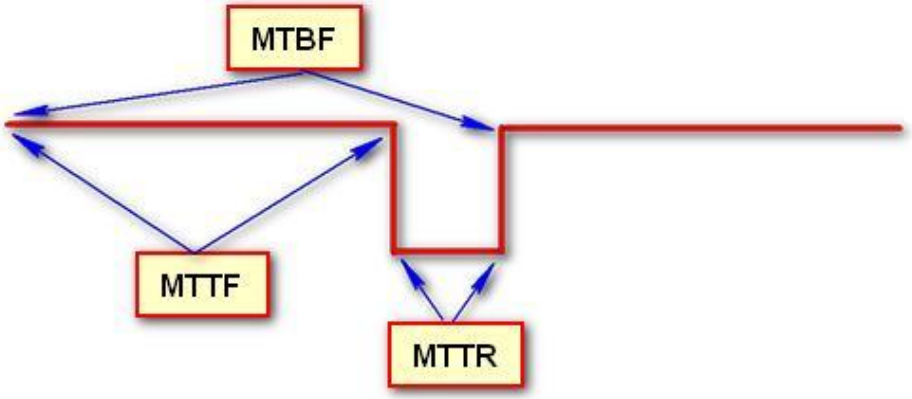
2 REVISÃO DA LITERATURA

Esta seção tem por objetivo apresentar os conceitos relacionados à alta disponibilidade, *PfSense, Mikrotik RouterOS, Zabbix*, Multi WAN, *failover* e *load balance*.

2.1 ALTA DISPONIBILIDADE

A disponibilidade de um sistema computacional é definida como sendo a probabilidade de um sistema estar funcionando e pronto para o uso em um dado momento. A Figura 1 apresenta a relação entre MTBF (*Mean Time Between Failures*), MTTR (*Maximum Time To Repair*) e MTTF (*Mean Time To Failures*) (ALDEVAN, 2013).

Figura 1 Tempo médio: falhas reparos e falhas



Fonte: ALDEVAN, 2013

O MTTR normalmente depende do SLA (*Service Level Agreement*) acordado. Nos exemplos aqui apresentados vamos utilizar um número apenas para uso didático. O Quadro 1 apresenta uma disponibilidade de 1 nove (90%) até 5 noves (99,999%) e os tempos de parada (indisponibilidade ou *downtime*) por ano (ALDEVAN, 2013).

Quadro 1 - Relação por tempo de indisponibilidade

Tipo de sistema	Disponibilidade	Tempo de Inatividade / Ano	Noves
Não Gerenciado	90%	52,560 minutos	1 noves
Gerenciado	99%	5,256 minutos	2 noves
Bem Gerenciado	99,9%	526 minutos	3 noves
Tolerantes a Falhas	99,99%	314 minutos	4 noves
Alta Disponibilidade	99,999%	314 segundos	5 noves

Fonte: Dados da pesquisa, 2019

Para chegar a esse tempo, foram utilizados o MTBF do equipamento e o tempo que leva para reparo dele. Nos equipamentos em que se exige alta disponibilidade, um bom número é o que se chama de 5 noves, já que o tempo de *downtime* é de apenas 314

segundos por ano. A Equação 1 a seguir apresenta a disponibilidade de um sistema, em que D é a disponibilidade do sistema (ALDEVAN, 2013):

$$\boldsymbol{D} = \frac{MTBF}{(MTBF+MTTR)} \qquad \text{Equação 1}$$

1.2 *PFSENSE*

O software *PfSense* é uma distribuição personalizada de código aberto do *FreeBSD*, especificamente adaptada para uso como um *firewall* e roteador que é inteiramente gerenciado via interface Web. Além de ser uma plataforma de roteamento e *firewall* poderosa e flexível, ela inclui uma longa lista de recursos relacionados e um sistema de pacotes que permite mais expansibilidade sem adicionar vulnerabilidades de segurança e inchadas à distribuição de base. O projeto *PfSense* é hospedado e desenvolvido pela *Rubicon Communications*, LLC (Netgate) (PFSENSE, s/d).

1.2.1 *CARP e pfSync*

A combinação de *CARP*, *pfsSync* e a sincronização de configuração fornece a funcionalidade de alta disponibilidade no *PfSense*. Configuram-se dois ou mais *firewalls* como um grupo de falha. Se o *firewall* primário ficar *off-line*, o secundário se torna ativo automaticamente. O *PfSense* inclui a capacidade de sincronização de configuração, ou seja, qualquer alteração de configuração feita no primário automaticamente é sincronizada com o *firewall* secundário. A tabela de estado do *firewall* é replicada para todos os *firewalls* de *failover* configurado. Isso significa que suas conexões existentes serão mantidas em caso de falha, o que é importante para se evitarem interrupções de rede, impactando o negócio da empresa (4LINUX, s/d).

1.3 *MIKROTIK*

É uma empresa da Letônia fabricante de equipamentos para redes de computadores. Sua frente de produção trabalha especialmente com produtos *wireless* e roteadores, muito utilizados por provedores de banda larga e empresas dos mais diversos segmentos (LEANDRO, 2018).

1.3.1 *RouterOS*

O principal produto da *Mikrotik* é o *RouterOS*, um sistema operacional baseado em *Linux*, que permite que uma plataforma x86 torne-se um roteador. Nessa condição, ele possui diversas funções, como Proxy, VPN (*Virtual Private Network*), *Firewall*,

Hotspots, QoS, Controle de Banda, e outras. O acesso às funções varia conforme a licença adquirida (LEANDRO, 2018).

É possível criar uma rede segura com o *RouterOS*, além de haver suporte de protocolos de roteamento BGP (*Border Gateway Proocol*), RIP (*Routing Information Protocol*), OSPF (*Open Shortest Pathe Fist*), MPLS (*Multiprotocol Label Switching*), etc. Existem alguns métodos para administrar esse ambiente, como o Console (CLI – linha de comando), *Winbox* (GUI), WEB (configuração em ambiente WEB) e Dude (LEANDRO, 2018).

1.4 *ZABBIX*

A ferramenta de monitoramento de redes *Zabbix* oferece uma interface 100% Web para administração e exibição de dados. O servidor Zabbix coleta dados para o monitoramento sem agentes e de agentes. Quando alguma anormalidade é detectada, alertas são emitidos visualmente, através de uso de sistemas de comunicação como e-mail e SMS. O servidor *Zabbix* mantém histórico dos dados coletados em banco de dados (*Oracle, MySQL* e *PostgreSQL*), de onde são gerados gráficos, painéis de acompanhamento e *slide-shows*, que mostram informações de forma alternada (4LINUX, s/d).

1.4.1 *Zabbix Agent*

O agente *Zabbix* é instalado nos hosts e permite coletar métricas comuns - específicas de um sistema operacional, como CPU (*Central Processing Unit*) e memória. Além disso, o agente *Zabbix* permite a coleta de métricas personalizadas, com uso de scripts ou programas externos, permitindo a coleta de métricas complexas e até tomada de ações diretamente no próprio agente *Zabbix* (4LINUX, s/d).

1.5 MULTI WAN

Uma WAN abrange uma grande área geográfica, ou seja, é semelhante a uma grande LAN (*Local Area Network*) cabeada. Normalmente, em uma WAN, os hosts e a sub-rede são proprietários e são administrados por diferentes pessoas (TANENBAUM, 2011). Já a multi WAN permite o uso de múltiplas conexões a essa grande área na Internet, com balanceamento de carga e/ou *failover*, para melhorar a disponibilidade de Internet e a distribuição do uso da banda (4LINUX, s/d).

1.6 *FAILOVER*

O *failover* é a capacidade de determinado sistema/serviço migrar automaticamente para outro servidor, sistema ou rede redundante ou que está em *standby* quando da ocorrência de falha ou do término anormal do servidor, do sistema ou da rede que estava ativa até aquele instante. O *failover* acontece sem intervenção humana e geralmente sem aviso prévio. Reciprocamente, o *failback* é o processo de restauração de um sistema/componente/serviço que se encontra em um estado de *failover* (ou seja, aquela máquina onde estava rodando o serviço que apresentou problemas) de volta a seu estado original que estava antes da falha (APPUNIX, 2011).

1.7 *LOAD BALANCE*

O objetivo do balanceamento de carga é criar um sistema que virtualize o trabalho dos servidores físicos que executam aqueles serviços. Uma definição mais básica é a de equilibrar a carga entre vários servidores físicos que atendem a uma demanda específica e, com isso, fazer com que eles trabalhem de tal forma que aparente ser um grande servidor para o mundo externo (AGILITY, 2016).

Há muitos motivos para se fazer isso, no entanto, os principais pontos podem ser resumidos em escalabilidade, alta disponibilidade e previsibilidade. A escalabilidade é a capacidade de adaptação fácil e dinâmica ao aumento da carga, sem impacto sobre o desempenho atual. A virtualização de serviços oferece uma oportunidade interessante para a escalabilidade. Se o serviço no ponto de contato do usuário estivesse separado do servidor, o reescalonamento do aplicativo significaria apenas adicionar mais servidores, que não seriam visíveis ao usuário final (AGILITY, 2016).

2 METODOLOGIA

O presente trabalho teve como base a implementação de uma infraestrutura de rede de alta disponibilidade, considerando, mesmo diante de falhas, a capacidade de manter o sistema e os serviços de rede funcionando normalmente, sem alterar o fluxo de trabalho de uma empresa.

Para isso, foi feita pesquisa sobre o assunto, realizado o planejamento de execução das tarefas, desenvolvidos os projetos e por fim implementada cada solução, além de um servidor de monitoramento para validação dos testes imposto em cada solução desenvolvida.

Para a etapa de desenvolvimento, foram utilizadas as subsequentes ferramentas:

- *VirtualBox*: programa de virtualização da *Oracle*, que permite instalar e executar diferentes sistemas operacionais em um único computador.
- *Ubuntu*: sistema operacional de código aberto, construído a partir do núcleo *Linux*, baseado no *Debian*, para instalação do software *Zabbix* e como cliente da rede.

- *Slackware*: sistema operacional de código aberto, uma das mais antigas e conhecidas distribuição *GNU/Linux*, como cliente da rede.
- *PfSense*: sistema operacional de código aberto baseado em *Unix FreeBSD*, adaptado para ser usado como um *firewall* e/ou roteador.
- *Draw.io Diagrams*: é um software de diagrama *on-line* gratuito, para fazer fluxogramas, diagramas de processo, organogramas, UML (*Unified Modeling Language*) e diagramas de rede.
- *Trello:* aplicação web para gerenciamento de projetos. Será utilizada para organização de todas as atividades, categorizando as que estão em fila, em execução e finalizadas.

Para levantamento das informações de cada solução, foi implementado um servidor *Zabbix* para monitorar e disponibilizar os resultados de cada teste imposto no projeto, implementados em cada ambiente, *PfSense* e *Mikrotik*.

3 RESULTADOS

No primeiro momento, foram realizadas pesquisas sobre balanceamento de carga, *failover*, e multi WAN e suas subáreas, para implementação no *PfSense* e no *Mikrotik RouterOS*, e cenários para aplicação desse modelo de configuração.

Os procedimentos iniciais se basearam em fazer diagrama do projeto a ser desenvolvido, precedendo de um levantamento de requisitos do que é necessariamente composta a infraestrutura para uma rede de alta disponibilidade, usando o *PfSense* e no *Mikrotik RouterOS*.

O desenvolvimento das soluções foi realizado no *VirtualBox*, um sistema capaz de instalar diferentes sistemas virtualizados. Para implementação da solução, foi preciso preparar a plataforma onde as máquinas virtuais seriam instaladas, e definir as configurações básicas de *hardware* para implementação do *PfSense* e do *Mikrotik RouterOS*. Como se pode ver no Quadro 2, foi feita a padronização de *hardware* para as duas soluções, baseada nos requisitos mínimos recomendados para instalação do *PfSense*, segundo informações oficias no próprio site.

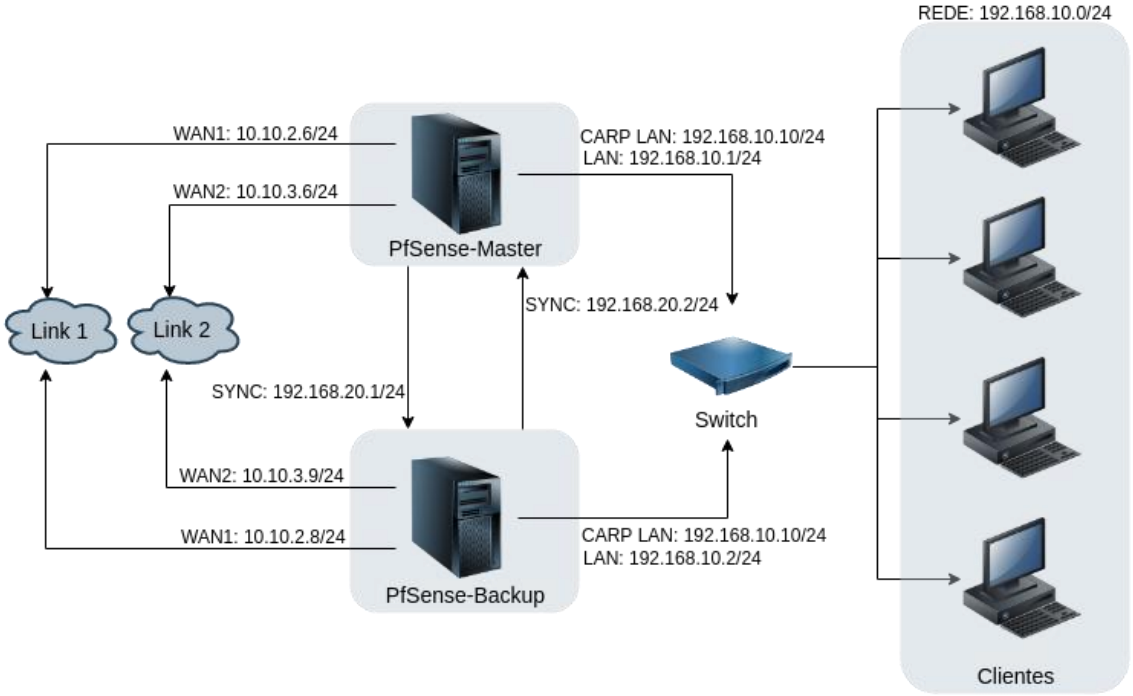
Quadro 2 – Definições de *hardware*

Sistema Operacional	Processador	Memória RAM	Placa de rede
PfSense 4.2 v2.4.4 amd64 *Mikrotik RouterOS* v6.36 x86	CPU –1Ghz	1GB	4

Fonte: Dados da pesquisa, 2019

A Figura 2 apresenta uma estrutura de rede implementada no *PfSense*, de alta disponibilidade caso venha ocasionar alguma falha. A sua implementação foi dividida em cinco fases.

Figura 2 – Diagrama de rede implementado no *PfSense*



Fonte: Elaborado pelos autores, 2019

Na primeira etapa, foram realizadas as definições das interfaces de rede WAN e LAN: duas interfaces de rede com descrições WAN1 e WAN2, para simulação dos links que dão acesso à navegação na internet para as estações da rede interna, e uma interface com descrição LAN, que faz a comunicação com a rede interna com atribuição de IP (*Internet Protocol*) via DHCP (*Dynamic Host Configuration Protocol*).

Na segunda etapa, fez-se a implementação da alta disponibilidade e redundância entre os dois *PfSense*. Para isso, foi utilizada uma interface de rede com a descrição SYNC, que faz a comunicação entre os dois *PfSense*, com base no conceito de IP Virtual (VIP) do tipo *CARP*.

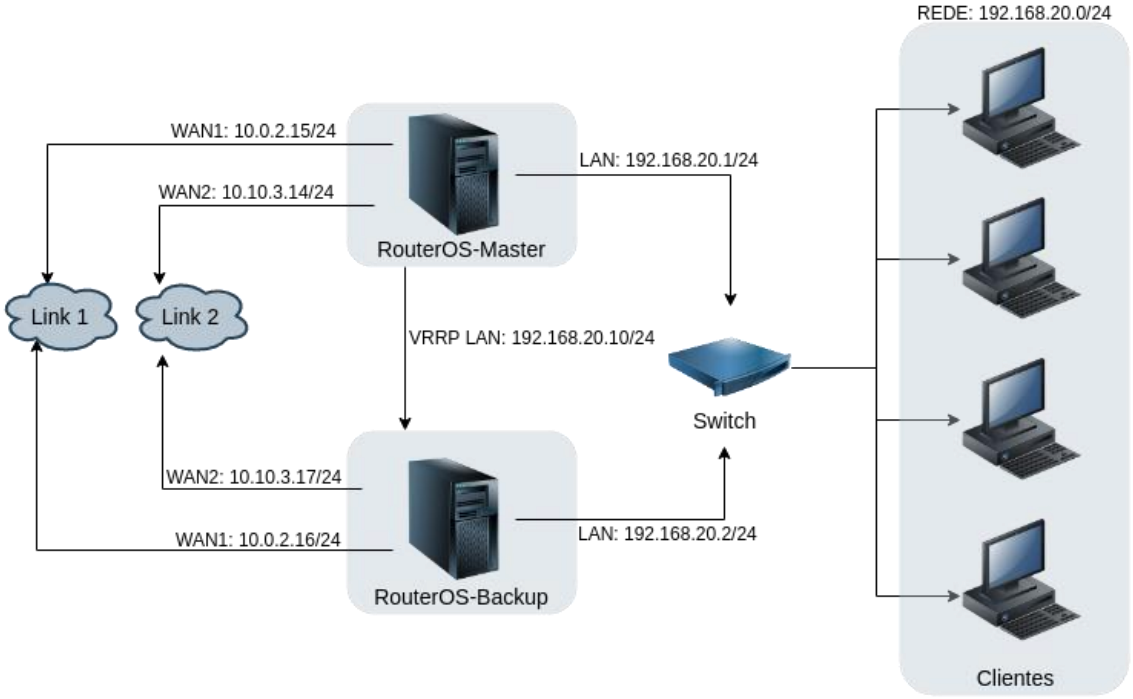
Na terceira etapa, foi feita a configuração de balanceamento de carga nas interfaces de rede WANs, criando-se um grupo de gateways com camada de prioridades iguais para as duas interfaces. Esse grupo foi utilizado como o gateway na interface de rede LAN, a fim de que as requisições para internet fossem balanceadas entre as interfaces de saída WAN.

Na quarta etapa, a configuração do *failover* foi realizada para as interfaces de rede WANs, criando-se um grupo de gateways com camada de prioridade diferente para as duas interfaces. Esse modelo foi utilizado para se determinar a prioridade entre as interfaces; utilizou-se uma interface de saída WAN. Caso ela caísse, a outra interface de rede assumiria as requisições para a internet.

Na quinta etapa, foi realizada a validação das configurações. Foram necessários ajustes de regras de *firewall* para comunicação entre os *PfSense*, acesso à internet e configuração de NAT (*Network Address Translation*) para que as máquinas da rede interna pudessem navegar na internet.

A Figura 3 apresenta uma estrutura de rede implementada no *Mikrotik RouterOS*, de alta disponibilidade caso haja alguma falha.

Figura 3 – Diagrama de rede implementado no *Mikrotik RouterOS*



Fonte: Elaborado pelos autores, 2019

A implementação no *Mikrotik RouterOS* teve etapas bem parecidas com as da implementação no *PfSense*, porém utilizaram-se conceitos e métodos diferentes em algumas etapas. Na implementação da alta disponibilidade e redundância entre os dois *Mikrotik RouterOS*, utilizou-se o conceito de vr (*Virtual Router Redundancy Protocol*), no qual o *RouterOS-Master* e *RouterOS-Backup* são membros de um *Mikrotik* VRRP (*Virtual Router Redundancy Protocol*) do mesmo VRID (*Virtual Router Identifier*), porém o *RouterOS-Master* tem prioridade maior do que o *RouterOS-Backup*.

Para o balanceamento de carga nas interfaces de rede WAN no *Mikrotik RouterOS*, utilizou-se o conceito de marcação de rotas que entra no roteador. Essa configuração foi realizada na secção *Firewall* do *Mikrotik RouterOS*, onde é configurado *Chain* do tipo *prerouting* com as marcações WAN e outra WAN2, fazendo com essa marcação seja atribuída nas configurações de rotas estáticas para as interfaces de saída WAN1 e WAN2, respectivamente.

Para a configuração do *failover*, utilizou-se o conceito de rotas, em que foi preciso configurar três rotas estáticas no *RouterOS-Master* e três *no RouterOS-Backup*: uma rota com o nome Principal designada a interface de rede WAN1, outra rota com o nome Backup designada a interface de rede WAN2 e a última que faz ping a cada minuto para o IP 8.8.8.8, saindo pela interface de rede WAN1, checando assim a disponibilidade dessa interface de rede. Caso ela não conseguisse acesso ao host com sucesso, ela trocaria a sua rota de Principal para a Backup, automaticamente.

As outras configurações como nome de interface de rede, regras de *firewall*, NAT, servidor DHCP na interface de rede LAN seguiram o mesmo procedimento. Após a implementação da alta disponibilidade nos dois sistemas, foram realizados quatro testes diferentes para saber qual solução teria maior desempenho. Para isso, utilizou-se o *Zabbix* para coleta e cálculo automático dos resultados.

O principal fator considerado nos resultados foi o SLA, que é o acordo de nível de serviços entre as partes interessadas e o *Service* HTTP (*HyperText Transfer Protocol*). O SLA é composto por dois itens: a disponibilidade do *host*, em que, a cada segundo, o *Zabbix* verificava se máquina estava ativa, e o Service HTTP, que é o resultado de um script escrito em Shell que faz um ping no IP do Google e checava se o código de estado HTTP era igual a 200, resposta de requisição bem-sucedida.

O Quadro 3 apresenta o resultado do primeiro teste, que simulava uma indisponibilidade de *hardware*, causando uma falha a cada 15 minutos durante 24 horas.

Quadro 3 – Resultado do teste: falha de *hardware*

	RouterOS-Master	RouterOS-Backup	PfSense-Master	PfSense-Backup
SLA	58,1568	98,6692	49,527	99,6075
Service HTTP	23,9s		3,66s	
Uptime Média	00h09m14s	-	00h07m34s	-

Fonte: Dados da pesquisa, 2019

Segundo esse resultado, o *PfSense* teve um SLA menor, porque, ao se desligar, ele demorava muito para ser ligado novamente, gerando assim uma média de *uptime* menor em relação ao *Mikrotik RouterOS*, porém ele obteve menor tempo de indisponibilidade no serviço Web, porque ele consegue detectar que houve uma falha mais rapidamente e passar assim as requisições para o servidor de backup.

O Quadro 4 apresenta o resultado do segundo teste, o *failover*, que, através de processo automático, simulava uma falha na interface LAN e WAN a cada 14 segundos de forma alternada, com um intervalo entre as falhas de 1 minuto, durante 24 horas.

Quadro 4 – Resultado do teste de *failover*

	RouterOS-Master	RouterOS-Backup	PfSense-Master	PfSense-Backup
SLA	52,1843	98,2449	52,7074	99,3885
Service HTTP	13,65s		2,38s	

Fonte: Dados da pesquisa, 2019

O *Mikrotik RouterOS* teve maior tempo de resposta a uma falha, segundo o resultado *Service* HTTP, isso porque ele demorou, no mínimo, 1 minuto para fazer uma requisição no IP 8.8.8.8, detectando assim a falha, migrando para o servidor que estava em *standby*.

No Quadro 5 é apresentado o resultado do terceiro teste, o *load balance*. Para esse teste, utilizaram-se duas máquinas clientes (*Ubuntu e Slackware*) que faziam três *downloads* em cada uma ininterruptamente durante 24 horas.

Quadro 5 – Resultado do teste de *load balance*

	RouterOS-Master	RouterOS-Backup	PfSense-Master	PfSense-Backup
Input WAN1	12,35Mbps	63,99bps	10.45Mbps	991.86bps
Output WAN1	174,37Kbps	117,49bps	170.39Kbps	1.09Kbps
Input WAN2	11,54Mbps	0,0948bps	9.77Mbps	921.5bps
Output WAN2	167,82Kbps	56,06bps	159.5Kbps	653.77bps

Fonte: Dados da pesquisa, 2019

Pôde-se observar que não houve diferença entre os seus balanceamentos entre os links de internet, mesmo usando métodos de aplicabilidade diferente. O resultado entre os dois é o mesmo tanto para entrada de dados (*input*) quanto para saída de dados (*output*).

O Quadro 6 apresenta o resultado do quarto teste, de performance. Para esse teste, utilizou-se uma máquina clientes (*Ubuntu*), que fazia três *downloads* ininterruptamente durante 24 horas. Nesse processo, o *Zabbix* media como cada servidor se comportava em diversos aspectos.

Quadro 6 – Resultado do teste de performance

	RouterOS-Master	RouterOS-Backup	PfSense-Master	PfSense-Backup
SLA	99,8530	99,8530	99,9803	99,9803
Service HTTP	1,32		1,13	
Used *memory*	19,69MB	19,5MB	148,79MB	109,13MB
Load average (1m)	13,01%	0%	24,40%	7,22%
LAN downtime	0	0	0	0
WAN *downtime*	0	0	0	0
ICMP *loss*	0,0507%	0%	0%	0%
ICMP *reponse time*	0,47ms	0,55ms	0,32ms	0,35ms
Input WAN1	18,41Mbps	56,13bps	19,45Mbps	962,56bps
Output WAN1	195,85Kbps	64,14bps	299,26Kbps	1,06Kbps
Input WAN2	0,0948bps	0,0632bps	932,9bps	931,68bps
Output WAN2	56,06bps	56,13bps	653,32bps	652,17bps
Input LAN	234,81bps	253,56bps	340,31Kbps	19,8Mbps
Output LAN	32bps	32,05bps	19,44Mbps	3,26Kbps

Fonte: Dados da pesquisa, 2019

Como se pôde observar, não houve diferença nas duas soluções. As duas apresentaram que, diante das operações normais do dia a dia, com o ambiente favorável, sem nenhuma tratativa de incidentes, conseguem entregar resultados esperados similares.

4 CONCLUSÃO

Esse trabalho teve como objetivo a implementação de uma infraestrutura de rede com alta disponibilidade, oferecendo aos provedores de serviços uma maior confiabilidade em seus sistemas, evitando-se assim transtornos e até perda de dinheiro por descumprimento do SLA.

Para a implementação dessa solução no *Mikrotik RouterOS*, exige-se maior conhecimento na parte de redes de computadores no geral. Ela apresenta ser específica e menos intuitiva, o que dificulta a implementação. Já o *PfSense* é uma plataforma mais intuitiva, ajudando na agilidade das configurações, e não exige tanto conhecimento da área.

Os testes impostos em cada solução, em sua maioria, não apresentaram muita diferença, apenas o *PfSense* que, no teste de *failover* e no teste de indisponibilidade de *hardware*, demostrou ser mais instável, detectando a falha e realizando o *failover* e o *failback* com maior agilidade; porém, no teste de indisponibilidade de máquina, ele teve um desempenho menor por demorar mais para ligar e desligar seu sistema.

Já nos demais testes, as duas soluções se mantiveram com os resultados praticamente idênticos, principalmente quando se trata do balanceamento de carga. Conclui-se que as duas soluções podem ser aplicadas em ambientes similares, porém o *PfSense* se destacou bem quando o assunto é instabilidade.

Para projetos futuros, pode-se implementar essa solução em ambiente real, com máquinas não virtualizadas e mais robustas, com dois links de internet e com redundância de energia para proporcionar um ambiente favorável. Levanta-se a hipótese de que, com máquinas melhores, seja possível minimizar o tempo de ligação do *Pfsense*, otimizando-se os resultados do uso destas tecnologias como um todo.

REFERÊNCIAS

4LINUX. **O que é PfSense.** s/d. Disponível em: https://www.4linux.com.br/o-que-e-pfsense.

4LINUX. **O que é Zabbix.** s/d. Disponível em: https://www.4linux.com.br/o-que-e-zabbix.

AGILITY. **A história do balanceamento de carga.** 2016. Disponível em: http://www.agilitynetworks.com.br/blogdaagility/a-historia-do-balanceamento-de-carga.

ALDEVAN. **Calcular disponibilidade e indisponibilidade dos ativos de rede.** 2013. Disponível em: https://under-linux.org/entry.php?b=3069**.**

APPUNIX. **O que é Failover, Failback e SwitchOver.** 2011. Disponível em: https://appunix.com.br/o-que-e-failover-failback-e-switchover.html.

LEANDRO. **O que é Mikrotik.** 2018. Disponível em: https://www.4infra.com.br/o-que-e-mikrotik/.

PFSENSE. **Faça um tour pelo pfSense.** s.d. Disponível em: https://www.pfsense.org/about-pfsense.

SIGNIFICADOS. **Significado de Kanban.** S.d. Disponível em: https://www.significados.com.br/kanban.

TANENBAUM, Andrew Stuart. **Redes de Computadores**. 5. ed. São Paulo: Person Education do Brasil, 2011.

VIEIRA, Omar Junio Antunes. **Comparação da alta disponibilidade implementada no PfSense e no Mikrotik**. 2019. Disponível em: https://github.com/omarjuav/Alta-disponibilidade-PfSense-e-Mikrotik.