Universidade do Estado do Rio de Janeiro
Centro de Tecnologia e Ciências
Faculdade de Engenharia

Rodrigo da Silva Amancio

Métodos de Resiliência em Redes Definidas por Software

Rio de Janeiro
2019

Rodrigo da Silva Amancio

Métodos de Resiliência em Redes Definidas por Software

Dissertação apresentada, como requisito parcial para obtenção do título de Mestre em Engenharia Eletrônica, ao Programa de Pós-Graduação em Engenharia Eletrônica, da Universidade do Estado do Rio de Janeiro. Área de concentração: Redes de Telecomunicações.

Orientador: Prof. D.Sc. Marcelo Gonçalves Rubinstein
Orientador: Prof. D.Sc. Rodrigo de Souza Couto

Rio de Janeiro
2019

CATALOGAÇÃO NA FONTE

UERJ / REDE SIRIUS / BIBLIOTECA CTC/B

A484 Amancio, Rodrigo da Silva.
Métodos de resiliência em redes definidas por software / Rodrigo da Silva Amancio. – 2019.
50f.

Orientadores: Marcelo Gonçalves Rubinstein, Rodrigo de Souza Couto.
Dissertação (Mestrado) – Universidade do Estado do Rio de Janeiro, Faculdade de Engenharia.

1. Engenharia eletrônica - Teses. 2. Redes remotas (Redes de computadores) - Teses. 3. Falhas de sistemas de computação - Teses. 4. Recuperação de dados (Computação) - Teses. 5. Computadores digitais - Confiabilidade - Teses. 6. Tolerância a falha (Computadores) - Teses. I. Rubinstein, Marcelo Gonçalves. II. Couto, Rodrigo de Souza. III. Universidade do Estado do Rio de Janeiro, Faculdade de Engenharia. IV. Título.

CDU 004.052.3

Bibliotecária: Júlia Vieira – CRB7/6022

Assinatura

Data

Rodrigo da Silva Amancio

Métodos de Resiliência em Redes Definidas por Software

Dissertação apresentada, como requisito parcial para obtenção do título de Mestre em Engenharia Eletrônica, ao Programa de Pós-Graduação em Engenharia Eletrônica, da Universidade do Estado do Rio de Janeiro. Área de concentração: Redes de Telecomunicações.

Aprovada em 18 de Dezembro de 2019.
Banca Examinadora:

Prof. D.Sc. Marcelo Gonçalves Rubinstein (Orientador)
PEL/UERJ

Prof. D.Sc. Rodrigo de Souza Couto (Orientador)
PEE/COPPE/UFRJ

Prof. D.Sc. Alexandre Sztajnberg
PEL/UERJ

Prof. D.Sc. Igor Monteiro Moraes
IC/UFF

Rio de Janeiro

2019

DEDICATÓRIA

Dedico esta dissertação à minha avó Lenita (*in memoriam*) que nos deixou no decorrer dessa dissertação e que foi imprescindível na minha criação, além de deixar como exemplo a sua simplicidade, carinho e amor ao próximo.

AGRADECIMENTOS

Agradeço primeiramente a Deus por me guiar, iluminar e me dar tranquilidade para seguir em frente com os meus objetivos e não desanimar com as dificuldades. Agradeço aos meus pais e avós por terem me dado educação e valores. Agradeço à minha namorada Jaqueline Souto por todo o carinho, respeito, compreensão e paciência durante todo esse processo. Agradeço aos amigos da Logicalis RJ, principalmente ao Marcio Vieira e Bruno Hodge, por todo o apoio, tornando possível minha ausência em vários momentos e batalhando comigo em mais essa etapa. Agradeço ao apoio dos colegas de classe, que foram sempre prestativos e companheiros durante todos os dias de aula e estudo. Agradecimento especial aos professores e orientadores Rodrigo S. Couto e Marcelo G. Rubinstein, pelo tempo dedicado, pelas orientações, pela disposição de sempre me ajudar com minhas limitações técnicas e teóricas e principalmente pela paciência. Agradeço ao Programa de Pós-Graduação em Engenharia Eletrônica da Universidade do Estado do Rio de Janeiro pela oportunidade de poder participar do programa de Pós-Graduação. Por fim, um agradecimento a todos que de certa forma participaram, mesmo que indiretamente, desse momento único em minha vida. Muito obrigado!

A tarefa não é tanto ver aquilo que ninguém viu, mas pensar o que ninguém ainda pensou sobre aquilo que todo mundo vê.

Arthur Schopenhauer

RESUMO

AMANCIO, R. S. A. *Métodos de Resiliência em Redes Definidas por Software*. 2019. 50 f. Dissertação (Mestrado em Engenharia Eletrônica) – Faculdade de Engenharia, Universidade do Estado do Rio de Janeiro, Rio de Janeiro, 2019.

As Rede Definidas por *Software* (*Software Defined Networks* - SDNs) facilitam o gerenciamento e a configuração dinâmica da rede; porém, a capacidade de responder prontamente a falhas em um curto período de tempo também é essencial. Tendo em vista que as redes estão cada vez mais utilizando o conceito de SDN, é importante estudar os métodos de resiliência neste tipo de rede. O processo de detectar a falha de enlace, comunicá-la ao controlador e recalcular os novos caminhos mais curtos pode resultar em um longo tempo de recuperação. Esse tempo de recuperação deve ser pequeno de forma a não comprometer os serviços oferecidos. Nesta dissertação analisam-se métodos de resiliência que permitem a diminuição do atraso, do *jitter* e da perda de pacotes em caso de falhas. Os métodos analisados utilizam fluxos pré-configurados, Detecção de Encaminhamento Bidirecional (*Bidirectional Forwarding Detection* - BFD) e a agregação de enlaces por meio da implantação do LACP (*Link Aggregation Control Protocol*). Com esses mecanismos, é possível aumentar a confiabilidade da rede diminuindo em até oito vezes a perda de pacotes. Além disso, melhora-se a estabilidade da rede com uma diminuição em torno de 0,8 ms do *jitter* e 3,2 ms do RTT em comparação a cenários sem os métodos de resiliência implementados.

Palavras-chave: SDN. Resiliência. Falhas. Recuperação.

ABSTRACT

AMANCIO, R. S. A. *Resilience Methods in Software Defined Networks*. 2019. 50 f. Dissertação (Mestrado em Engenharia Eletrônica) – Faculdade de Engenharia, Universidade do Estado do Rio de Janeiro, Rio de Janeiro, 2019.

Software Defined Networking (SDN) facilitates the management and the dynamic configuration of the network, however, the ability to quickly respond to failures in a short period of time is also essential in large networks. Considering that networks are increasingly using the SDN concept, it is important to study the methods of resilience in this type of network. The process of detecting link failure, communicating it to the controller, and recalculating the new shortest paths can result in a long recovery time. This recovery time should be short so as not to compromise the services offered. In this dissertation, we analyze methods of resilience that allow the reduction of delay, jitter and the packet loss in case of failures. The methods analyzed use preconfigured flows, Bidirectional Forwarding Detection (BFD) and link aggregation through the LACP (Link Aggregation Control Protocol) deployment. With the aforementioned mechanisms it is possible to increase network reliability by reducing packet loss by up to eight times. In addition, the stability of the network is increased by reducing about 0.8 ms of the jitter and 3.2 ms of the RTT in comparison to scenarios without the implemented resilience methods.

Keywords: SDN. Resilience. Failures. Recovery.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

API	*Application Programming Interface*
BFD	*Bidirectional Forwarding Detection*
CPU	*Central Processing Unit*
IP	*Internet Protocol*
LACP	*Link Aggregation Control Procol*
LAG	*Link Aggregation Group*
NFV	*Network Functions Virtualization*
RAM	*Random Access Memory*
RSVP-TE	*Resource Reservation Protocol - Traffic Engineering*
SDN	*Software-Defined Networking*
MPLS	*Multi-Protocol Label Switching*
UDP	*User Datagram Protocol*
VNF	*Virtualized Network Function*
WDM	*Wavelength Division Multiplexing*

SUMÁRIO

INTRODUÇÃO

A capacidade de responder prontamente a falhas em um curto período de tempo é essencial em redes de grande porte, como *datacenters* e provedores de serviço de Internet. Tendo em vista um cenário altamente competitivo, a busca por melhor desempenho nos serviços oferecidos e a necessidade de redes cada vez mais estáveis e resilientes, arquiteturas SDN (*Software Defined Networking*) têm se tornado as principais opções de implementação. Isso ocorre pois as redes IP tradicionais são mais difíceis de gerenciar, se comparadas às redes SDNs, já que precisam ser configuradas e analisadas individualmente (KREUTZ et al., 2015).

A maioria das soluções atuais de comutação e roteamento integra o plano de dados com o plano de controle. O plano de dados executa o encaminhamento por pacote com base em tabelas de pesquisa localizadas na memória do comutador ou roteador, enquanto o plano de controle é usado para definir as regras. Devido às altas demandas de desempenho de rede e crescente complexidade de configuração, o plano de controle tornou-se excessivamente complicado, inflexível e difícil de gerenciar. Para resolver este problema, foi necessário um novo paradigma de rede, compatível com as técnicas utilizadas de comutação como a *Ethernet* e roteamento IP. A solução foi encontrada em técnicas que permitem a separação dos planos de controle e de dados e a SDN adotou esse paradigma (KREUTZ et al., 2015).

A arquitetura SDN ganhou importância no campo de redes porque permite um desacoplamento entre o plano de controle e o plano de dados. Nas redes SDN, os elementos de rede são responsáveis pelo encaminhamento dos pacotes de dados com base nas regras de encaminhamento que são calculadas e instaladas por um controlador logicamente centralizado. Como o controlador pode ser programado em linguagens de alto nível, isso acelera a inovação da rede, pois permite implantar novos serviços rapidamente (VESTIN; KASSLER; AKERBERG, 2015).

Um dos benefícios da entidade controladora central introduzida na SDN é sua possibilidade de monitorar a rede quanto ao desempenho e funcionalidade, assim como sua capacidade de reprogramar os fluxos quando necessário. O controlador pode monitorar a integridade geral da rede e observar as características de fluxos, como a vazão, o atraso e a perda de pacotes. Além disso, o controlador pode detectar falhas nos enlaces com base nas estatísticas de perdas de pacotes e calcular caminhos alternativos para os fluxos afetados. A tarefa mais básica do controlador é configurar como é realizado o encaminhamento de tráfego dos fluxos da rede (SHENKER et al., 2011). Portanto, quando um enlace é interrompido, o controlador precisa reconfigurar a rede para restaurar ou manter a conectividade. Todavia, o tempo de restauração de um caminho quebrado, além do tempo de detecção da quebra, inclui o atraso introduzido pelo tempo de propagação da notificação

do evento ao controlador e o atraso da reconfiguração da rede. Todas essas etapas, consequentemente, atrasam a execução dos métodos de resiliência da rede, aumentando o tempo de recuperação (ADRICHEM; ASTEN; KUIPERS, 2014).

Motivação e Objetivos

Como exposto anteriormente, caso ocorra uma falha em rede SDN, pode haver uma demora no cálculo em busca de um caminho alternativo, causando instabilidade na rede. Nesse caso, é relevante considerar alguns métodos que possam manter a rede mais estável e definir de forma rápida e eficiente um caminho alternativo em caso de falhas.

Esta dissertação apresenta uma análise de desempenho de métodos de resiliência em redes SDN. Esses métodos utilizam funcionalidades que combinam caminhos primários e secundários pré-configurados. A implementação da detecção de falha por enlace é realizada por meio da Detecção de Encaminhamento Bidirecional (*Bidirectional Forwarding Detection* - BFD) (KATZ; WARD, 2010), um protocolo que detecta falhas por meio da análise da perda de pacotes em fluxos frequentes de mensagens de controle. Além disso, o mecanismo de agregação de enlaces, que utiliza o Protocolo de Controle de Agregação de Enlace (*Link Aggregation Control Protocol* - LACP) (IEEE, 2000), também é implementado em redes resilientes e o seu desempenho é analisado quando implementado na borda da rede.

Os resultados obtidos mostram que é possível notar uma melhora de desempenho considerável em cenários que ocorrem falhas e utilizam os métodos de resiliência e seus mecanismos analisados nesta dissertação, como no caso dos cenários com fluxo pré-configurado, fluxo pré-configurado com BFD, controlador com BFD e o cenário que utiliza o LACP. A perda de pacotes, por exemplo, pode diminuir cerca de oito vezes em comparação com cenários que não utilizam estes métodos, ou seja, que utilizam apenas o controlador como resiliência. Além disso, o *jitter* e o RTT (*Round-Trip Time*) também reduzem seus respectivos valores melhorando o desempenho e a estabilidade da rede. O jitter diminui cerca de 0,8 ms com a utilização dos métodos de resiliência analisados nesta dissertação e o RTT pode diminuir até 3,2 ms em comparação com o cenário que utiliza apenas o controlador como método de resiliência. Já o cenário que utiliza o LACP consegue manter o jitter e o RTT estatisticamente igual a de um cenário sem queda.

Organização do Texto

A dissertação está organizada da seguinte forma. O Capítulo 1 apresenta os principais conceitos de resiliência em redes de computadores. O Capítulo 2 descreve a arquite-

tura SDN. No Capítulo 3 são apresentados os métodos de resiliência e suas classificações. Na sequência, o Capítulo 4 apresenta trabalhos que se relacionam com esta dissertação e mostra como ela está posicionada na literatura. No Capítulo 5, os resultados obtidos da análise de desempenho são apresentados e analisados. Finalmente, a conclusão e as possibilidades de trabalhos futuros são apresentadas no Capítulo 6.

1 RESILIÊNCIA EM REDES DE COMPUTADORES

Para facilitar o entendimento da abordagem desta dissertação, são apresentados neste capítulo os principais conceitos de resiliência. Conhecer as definições e características dos métodos de resiliência é fundamental para a discussão sobre a análise de desempenho realizada. A Figura 1 ilustra os esquemas de resiliência que são abordados neste capítulo. Os esquemas são divididos em dois paradigmas: proteção e restauração.

1.1 Proteção como técnica de resiliência

A proteção é uma técnica que define um caminho secundário antes de ocorrer uma falha. A abordagem de proteção é dividida em dois segmentos: a proteção do caminho e a proteção do enlace.

Na técnica de resiliência utilizando a proteção do caminho, os nós de origem e de destino de cada conexão reservam estaticamente os caminhos secundários de uma extremidade a outra durante a configuração (RAMAMURTHY; MUKHERJEE, 1999).

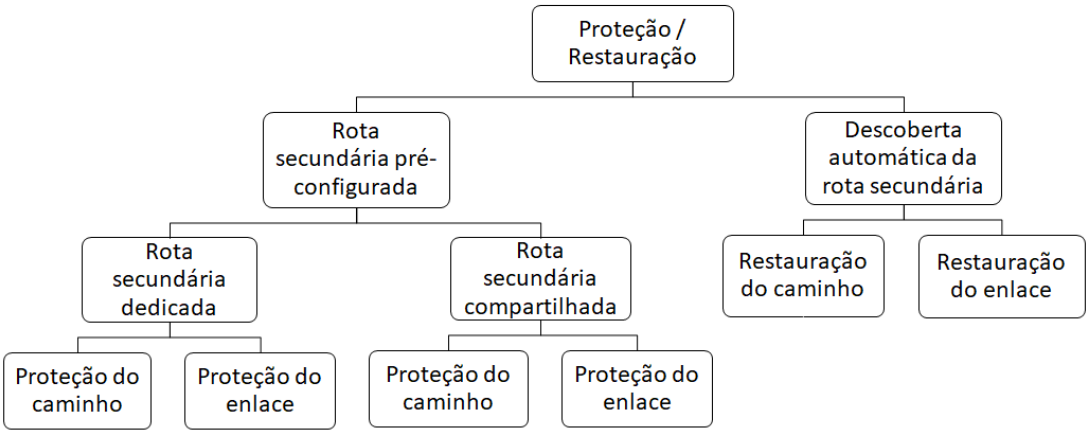
A Figura 2 mostra um exemplo de estratégia de resiliência da rede por meio da proteção do caminho. Supõe-se que o caminho principal, 1-2-3-6, foi afetado com a queda do enlace entre os nós 2 e 3. Como existe um caminho secundário, o nó de origem 1 poderia utilizar o caminho 1-4-5-6 para se comunicar com o nó de destino 6.

Existem duas formas de proteção do caminho: a proteção do caminho dedicado e a proteção do caminho compartilhado. Em ambos os tipos de proteção, no momento da configuração para o caminho primário, também é definido um caminho secundário. A proteção do caminho dedicado é também chamada de caminho 1+1, no qual o caminho definido como secundário não é compartilhado com outros caminhos. Já na proteção do caminho compartilhado, o caminho secundário pode ser utilizado normalmente por outros fluxos. Em resumo, nessa abordagem espera-se que os caminhos secundários sejam utilizados em diferentes momentos de falha, e portanto, a proteção do caminho compartilhado é mais eficiente em comparação à proteção do caminho dedicado já que enlaces do caminho secundário são compartilhados (RAMAMURTHY; MUKHERJEE, 1999).

Já na técnica de resiliência utilizando a proteção do enlace todas as conexões que atravessam o enlace com falha são redirecionadas em torno desse enlace, como mostra a Figura 3. Durante a configuração, os caminhos secundários são reservados em torno de cada enlace do caminho primário (RAMAMURTHY; MUKHERJEE, 1999).

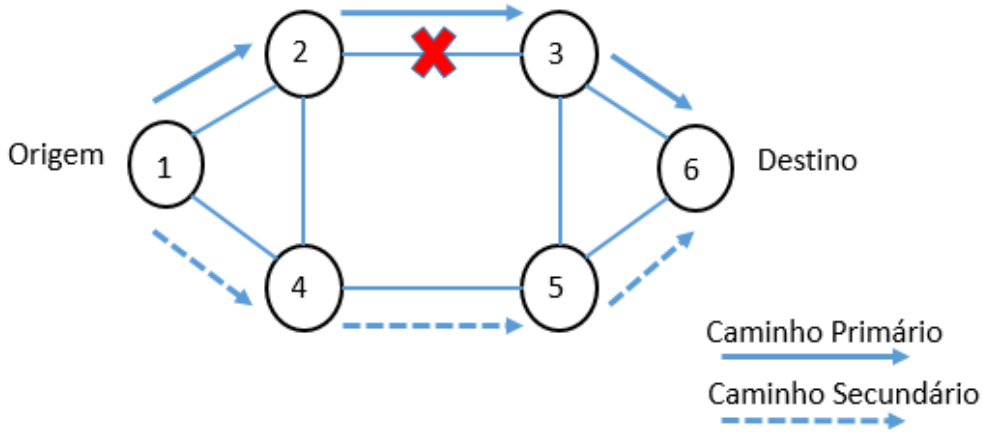
Existem duas formas de proteção do enlace: proteção do enlace dedicado e a proteção do enlace compartilhado. Na proteção do enlace dedicado, no momento da configuração, para cada enlace do caminho primário, um caminho secundário dedicado é

Figura 1 - Esquemas de sobrevivência a falhas.



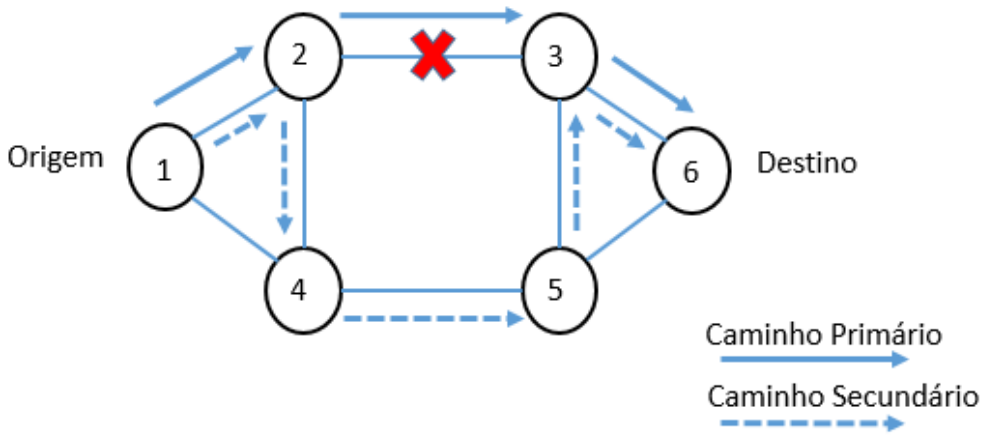
Fonte: Adaptado de (RAMAMURTHY; MUKHERJEE, 1999).

Figura 2 - Proteção do Caminho.



Fonte: Adaptado de (RAMAMURTHY; MUKHERJEE, 1999).

Figura 3 - Proteção do Enlace.



Fonte: Adaptado de (RAMAMURTHY; MUKHERJEE, 1999).

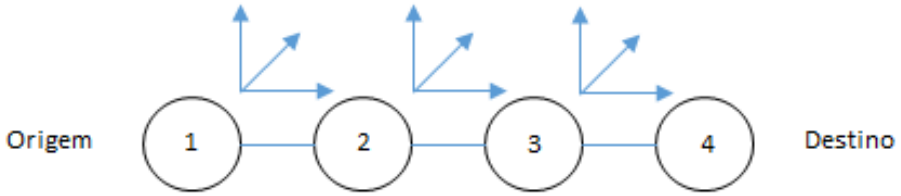
reservado em torno desse enlace. É bom ressaltar que nem sempre é possível alocar um caminho secundário dedicado em torno de cada enlace. Já na proteção do enlace compartilhado, diversos caminhos secundários podem usar um mesmo enlace, tornando-se mais eficiente em termos de capacidade (RAMAMURTHY; MUKHERJEE, 1999).

Em (CALLE; MARZO; URRA, 2004) aborda-se a alocação de recursos como um dos mecanismos de proteção existentes e exemplifica-se o seu funcionamento em uma rede MPLS (*Multi-Protocol Label Switching*). A alocação de recursos é pré-estabelecida quando os recursos de rede são alocados antes da falha. Quando os caminhos principal e secundário são selecionados, para pré-estabelecer e pré-reservar recursos, um protocolo de sinalização é utilizado com o objetivo de permitir a reserva necessária. O protocolo RSVP-TE (*Resource Reservation Protocol - Traffic Engineering*) é um exemplo de protocolo de sinalização. Além disso, em uma rede MPLS, um Caminho de Comutação de Rótulos (*Label Switch Path* - LSP) é criado distribuindo os rótulos apropriados sobre cada nó, que tem a função de encaminhar os pacotes baseados apenas no rótulo e reservar os recursos solicitados (CALLE; MARZO; URRA, 2004).

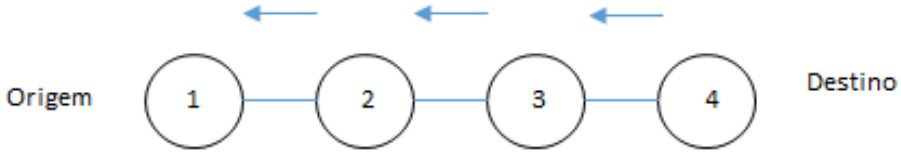
1.2 **Restauração como técnica de resiliência**

A restauração é uma técnica que só calcula um novo caminho após uma falha. Os protocolos de restauração de rede têm sido amplamente pesquisados na literatura. Em esquemas de restauração, o caminho secundário é determinado dinamicamente, a partir

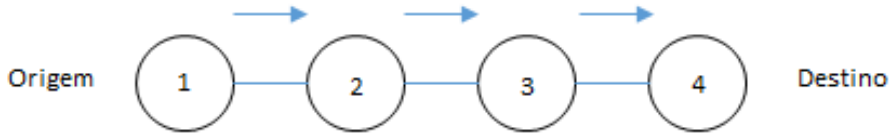
Figura 4 - Processo de Restauração.



(a) Transmissão para todos os enlaces e reserva



(b) Reconhecimento e configuração



(c) Confirmação

Fonte: Adaptado de (MUKHERJEE et al., 1999)

da capacidade disponível (MUKHERJEE et al., 1999).

Como mostra a Figura 4(a) o nó de origem que procura um caminho secundário envia mensagens de transmissão para todos os enlaces com capacidade disponível. Quando uma mensagem de transmissão atinge o nó de destino, Figura 4(b), o destino envia uma mensagem de confirmação ao longo do caminho que foi atravessado pela mensagem de transmissão e configura simultaneamente conexões cruzadas ao longo do caminho. Quando a mensagem de confirmação atinge o nó de origem, Figura 4(c), é enviada uma mensagem de confirmação para o destino, completando a configuração de conexão no caminho secundário (MUKHERJEE et al., 1999).

A abordagem de restauração também é dividida em dois segmentos: a restauração do caminho e a restauração do enlace.

Na restauração do caminho, após uma falha os nós de origem e de destino de cada conexão descobrem dinamicamente uma rota secundária de uma extremidade a outra. Em (MUKHERJEE et al., 1999) é abordado o funcionamento de um algoritmo na restauração

do caminho em uma rede WDM, que pode ser usado como exemplo para um melhor entendimento da restauração do caminho e suas características. A multiplexação por divisão de comprimento de onda (*Wavelength Division Multiplexing* - WDM) divide a largura de banda de uma fibra em muitos comprimentos de onda que não se sobrepõem, chamados de canais WDM. Cada canal pode ser operado de forma assíncrona e em paralelo a velocidade desejável. Nesse caso, os nós adjacentes ao enlace que falhou enviam mensagens de falha a todos os nós de origem e de destino de todas as conexões que atravessam o enlace com falha. À medida que estas mensagens se propagam, o comprimento de onda alocado para essa conexão pode ser liberado para uso por outras conexões. Quando um nó de origem de uma conexão recebe uma mensagem de falha de enlace, ele inicia uma pesquisa de caminho de restauração e se um caminho de restauração for encontrado, a conexão é configurada no caminho restaurado. Se for encontrado mais de um caminho de restauração para uma conexão, o primeiro encontrado é utilizado e os outros são liberados.

Na restauração do enlace, ainda usando uma rede WDM como exemplo, os nós de origem e de destino buscam dinamicamente uma rota ao redor do enlace com falha, para cada comprimento de onda que atravessa o enlace. Quando um caminho de restauração para uma conexão é encontrado, a conexão é trocada para o caminho restaurado. Geralmente, a restauração de enlace tem um melhor tempo de restauração em comparação com a restauração do caminho já que está restrita para encontrar enlaces secundários em torno do enlace com falha. Em contrapartida, a restauração do caminho realiza uma busca de um caminho secundário fim a fim, definindo um caminho totalmente novo a ser utilizado (MUKHERJEE et al., 1999).

É função dessa dissertação avaliar o desempenho de alguns métodos de resiliência em redes SDN, essas redes são apresentadas de uma forma reduzida no capítulo seguinte. No Capítulo 3, são apresentados os métodos de resiliência avaliados.

2 REDES DEFINIDAS POR *SOFTWARE (SOFTWARE-DEFINED NETWORKING, SDN)*

A arquitetura SDN separa o plano de dados do plano de controle. Dessa forma, os comutadores de rede tornam-se dispositivos de encaminhamento simples e a lógica de controle é implementada em um controlador logicamente centralizado, simplificando a aplicação de políticas, configuração, reconfiguração e evolução das redes. Essa separação é a principal característica que possibilita a flexibilidade desejada, simplificando o gerenciamento de rede e facilitando a evolução e a inovação da rede. A arquitetura SDN é composta por quatro pilares (KREUTZ et al., 2015):

- os planos de controle e dados são desacoplados. A funcionalidade de controle é removida dos dispositivos de rede que se tornam elementos de encaminhamento simples;

- em redes SDN as decisões de encaminhamento são baseadas em fluxo, em vez de baseadas no destino. Um fluxo é amplamente definido por um conjunto de valores contendo filtros e instruções. No contexto SDN/*OpenFlow*, um fluxo é uma sequência de pacotes entre uma origem e um destino. Todos os pacotes de um fluxo recebem políticas de serviço, ou seja, um grupo de regras, que neste caso são idênticas nos dispositivos de encaminhamento. A programação de fluxo permite flexibilidade, limitada às capacidades das tabelas de fluxo implementadas;

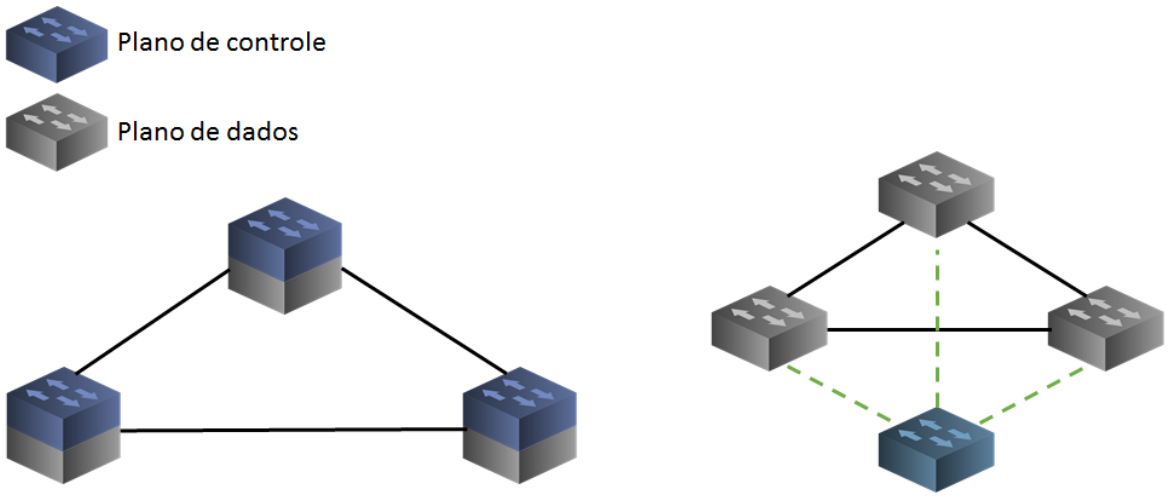
- a lógica de controle é movida para uma entidade externa, chamada de controlador. O controlador é uma plataforma de *software* que fornece os recursos e abstrações essenciais para facilitar a programação de dispositivos de encaminhamento com base em uma visão de rede abstrata, logicamente centralizada. Sua finalidade é, portanto, semelhante à de um sistema operacional tradicional;

- a rede é programável por meio de aplicativos de *software* executados sobre o controlador que, por sua vez, interage com os dispositivos de plano de dados subjacentes.

Novas arquiteturas de rede, como SDN, permitem que as redes sejam executadas perto de sua capacidade total e permitem que os enlaces carreguem mais tráfego sem a necessidade de excessivas reconfigurações na rede. Por outro lado, a otimização de encaminhamento pode ser interrompida pelo aumento repentino de tráfego, aumentando a necessidade de gerenciamento dos enlaces, principalmente em situações de falhas (GAY; HARTERT; VISSICCHIO, 2017).

Em relação ao gerenciamento da rede, a arquitetura SDN permite introduzir novas ideias na rede através de um programa de *software*, pois é mais fácil de alterar e manipular

Figura 5 - Diferença entre redes tradicionais e redes SDN.



(a) Rede tradicional com plano de controle acoplado (b) Rede SDN com plano de controle desacoplado

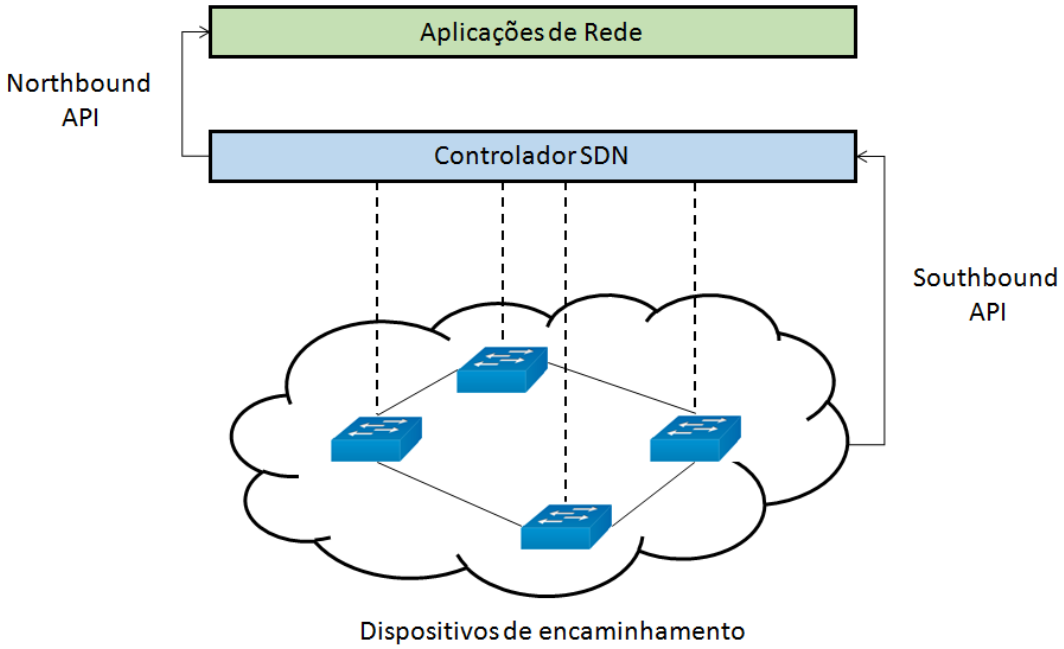
Fonte: Adaptado de (KREUTZ et al., 2015)

do que usar um conjunto fixo de comandos em dispositivos de rede proprietários. Além disso, a arquitetura SDN apresenta os benefícios de uma abordagem centralizada à configuração de rede, na qual os operadores não precisam configurar todos os dispositivos de rede individualmente para fazer mudanças no comportamento da rede. No entanto estes tomam decisões de encaminhamento de tráfego em toda a rede em um local logicamente único, com conhecimento global do estado da rede (KIM; FEAMSTER, 2013).

Em redes tradicionais, os planos de controle e dados são fortemente acoplados, incorporados nos mesmos dispositivos de rede e toda a estrutura é altamente descentralizada. Esse *design* foi considerado pois demonstrava ser a melhor maneira de garantir a resiliência da rede, que era um objetivo crucial do projeto. No entanto, o resultado é uma arquitetura muito complexa e relativamente estática. É também a razão fundamental pela qual as redes tradicionais são rígidas e complexas para gerenciar e controlar (KREUTZ et al., 2015). A Figura 5 ilustra a diferença entre redes tradicionais, com o acoplamento do plano de controle, e redes SDN com o plano de controle desacoplado.

A Figura 6 ilustra a arquitetura SDN e seus principais componentes. A separação do plano de dados e do plano de controle pode ser realizada por meio de uma interface de programação bem definida entre os comutadores e o controlador SDN. O controlador exerce controle direto sobre os elementos do plano de dados através de uma API (*Application Programming Interface*). Um comutador que implementa *OpenFlow*, que é um exemplo de API, possui uma ou mais tabelas de regras de manipulação de pacotes, que também podem ser chamadas de tabelas de fluxo. Cada regra corresponde a um subconjunto do tráfego e executa certas ações como, por exemplo, descartar, reencaminhar

Figura 6 - Arquitetura SDN.



Fonte: Adaptado de (KREUTZ et al., 2015)

e modificar o tráfego. Dependendo das regras instaladas, um comutador *OpenFlow* pode ser instruído pelo controlador a operar como um roteador, *firewall* ou até mesmo executar outras funções, como balanceador de carga (KREUTZ et al., 2015).

Existem diversos tipos de controladores SDN, como por exemplo NOX, Floodlight, Beacon, entre outros. O controlador *OpenDaylight* foi lançado com o objetivo de padronizar o desenvolvimento em SDN, apresentando uma nova arquitetura de controlador com base no conceito de camada de abstração de serviços (*Services Abstraction Layer* - SAL) (KHATTAK; AWAIS; IQBAL, 2014). Por ser utilizado nesta dissertação, o controlador OpenDaylight é apresentado a seguir.

2.1 Controlador *OpenDaylight*

O *OpenDaylight* é um projeto de código aberto suportado pela IBM, Cisco, Juniper, VMWare e vários outros grandes fornecedores de redes. O *OpenDaylight* é uma plataforma de controlador SDN implementada em *Java*. Como tal, pode ser implementado em qualquer plataforma de *hardware* e sistema operacional que suporte *Java* (KHATTAK; AWAIS; IQBAL, 2014). Este tipo de controlador possui três camadas:

- camada superior - Consiste em aplicativos de lógica de negócios e de rede. Esses aplicativos usam o controlador para coletar informações da rede, executar algoritmos

de análise e, em seguida, orquestrar as novas regras, se houver, em toda a rede;

- camada intermediária - É a estrutura que viabiliza a abstração SDN e hospeda as APIs. Além disso, também é formada pela camada de abstração de serviço (SAL) que determina como atender o serviço solicitado, independentemente do protocolo subjacente usado entre o controlador e os dispositivos de rede;

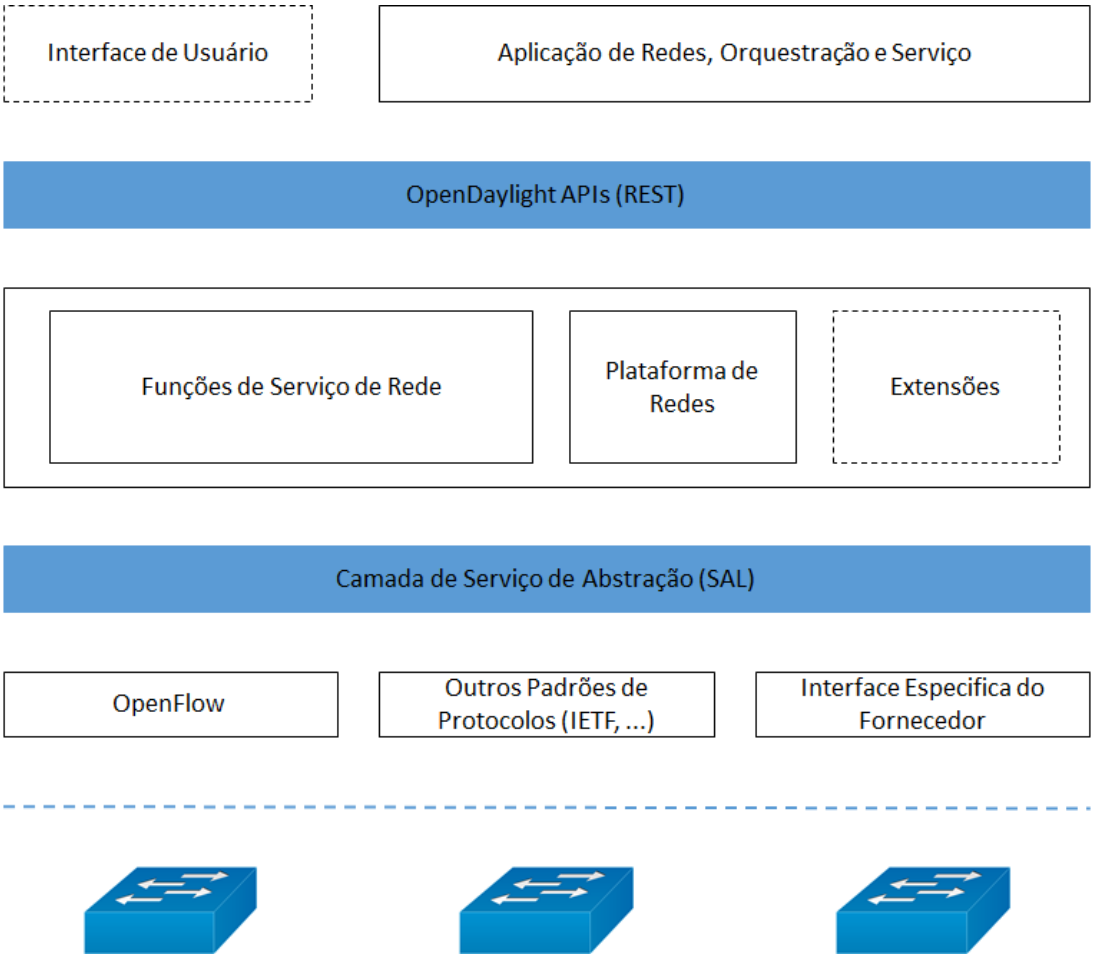
- camada inferior - Consiste em dispositivos físicos e virtuais como, por exemplo, comutadores e roteadores, que possibilitam a comunicação entre os elementos da rede. É capaz de suportar múltiplos protocolos como OpenFlow e BGP-LS.

A arquitetura de *software* do *OpenDaylight* define padrões de criação e interação de aplicativos e serviços subjacentes como, por exemplo, roteamento de mensagens, formatação e armazenamento de dados. Isso possibilita a criação de novas ferramentas de desenvolvimento (MEDVED et al., 2014). A arquitetura detalhada do controlador *OpenDaylight* com seus respectivos componentes e definições é ilustrada na Figura 7, no qual é possível visualizar também a segmentação das funções do controlador por meio de suas diferentes camadas.

As aplicações de rede podem criar regras para programar a rede através do controlador OpenDaylight, utilizando as interfaces *Northbound*. O núcleo do controlador traduz as solicitações dos serviços internos do controlador e dos aplicativos externos para os protocolos de rede implementados e conectados à interface *Southbound*. A interface *Southbound* é composta por um conjunto de *plugins* que implementam diferentes protocolos de controle e gerenciamento, para configurar dispositivos de rede física (*hardware*), como *OpenFlow*, *Netconf*, *Border Gateway Protocol* (BGP) entre outros (KHATTAK; AWAIS; IQBAL, 2014).

O controlador e o ambiente de desenvolvimento *OpenDaylight* expandem e ampliam a premissa básica da SDN, pois permitem um conjunto diversificado de serviços e aplicativos. Além disso, ele aproxima os aplicativos da rede e permite que os desenvolvedores de aplicativos e os pesquisadores da rede se concentrem nas APIs SDN, em vez dos protocolos usados para se comunicar com dispositivos de rede (MEDVED et al., 2014). Nesta dissertação utiliza-se o *OpenDaylight* como controlador SDN a fim de analisar e comparar o desempenho de alguns métodos de resiliência apresentados a seguir.

Figura 7 - Arquitetura OpenDaylight.



Fonte: Adaptado de (KHATTAK; AWAIS; IQBAL, 2014)

3 MÉTODOS DE RESILIÊNCIA AVALIADOS

Conforme a classificação apresentada no Capítulo 1, os mecanismos de resiliência podem ser divididos em métodos de proteção do caminho e métodos de proteção do enlace. A análise de desempenho realizada nesta dissertação engloba os resultados dos métodos de proteção do caminho, como os fluxos pré-configurados e os fluxos pré-configurados com o protocolo BFD. Além disso, também são apresentados resultados utilizando o LACP que é um caso especial de proteção do enlace, no qual provisiona-se um enlace redundante para o enlace protegido. A resiliência no controlador OpenDaylight também é analisada e comparada com os métodos implementados. Este capítulo explica detalhadamente o funcionamento de cada mecanismo utilizado nos métodos de proteção analisados.

3.1 Resiliência pelo Controlador OpenDaylight

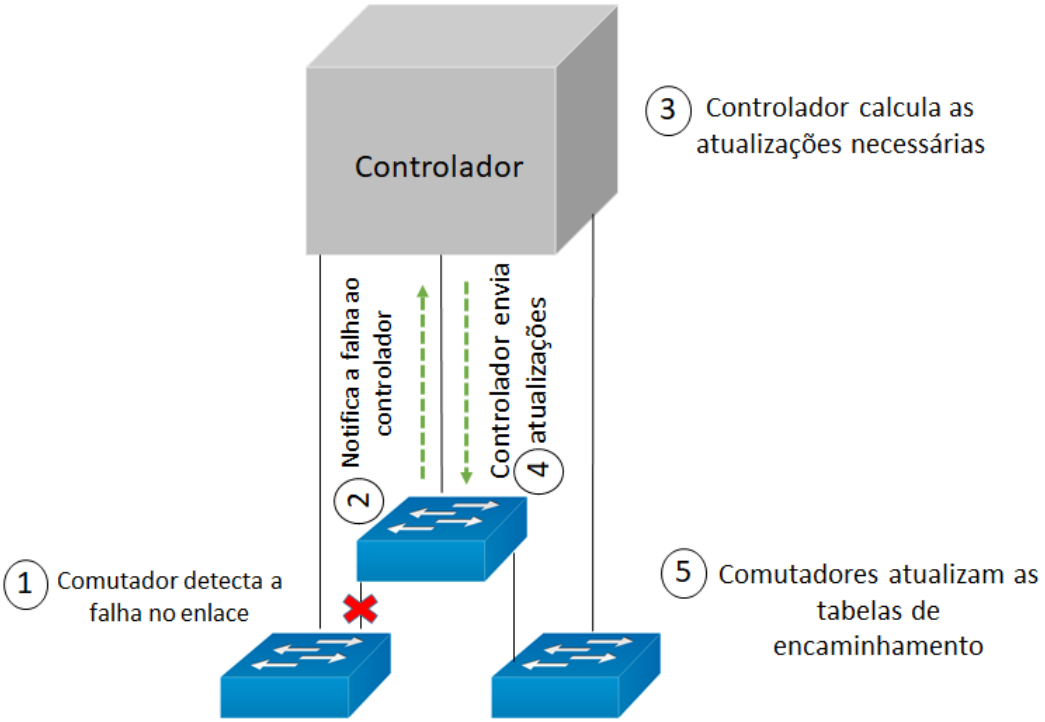
Quando ocorre uma falha de enlace na rede, o controlador precisa reconfigurá-la para manter a conectividade entre os nós. O controlador *OpenFlow* monitora constantemente toda a rede e calcula os caminhos de cada comutador intermediário para fornecer as redundâncias necessárias, ou seja, define quais caminhos podem ser utilizados em caso de falha. Além disso, uma vez que o controlador é informado do mau funcionamento de um enlace secundário que está sendo utilizado, ele pode reconfigurar a rede para substituir o caminho secundário atual por um caminho mais adequado. A resiliência no controlador não é estática , ou seja, o caminho secundário que está sendo utilizado após uma falha, pode ser alterado por um outro caminho caso não tenha a performance esperada.

Como ilustra a Figura 8, o processo de resiliência funciona de forma que o comutador detecta uma alteração e então notifica o controlador. Após a notificação, o controlador calcula as ações de reparo e envia atualizações para os elementos afetados que, por sua vez, atualizam suas tabelas de encaminhamento (YEGANEH; TOOTOONCHIAN; GANJALI, 2013). Todavia, todo esse processo inclui o atraso introduzido pelo tempo de propagação da notificação ao controlador e o atraso da reconfiguração da rede (ADRICHEM; ASTEN; KUIPERS, 2014).

3.2 Fluxos pré-configurados

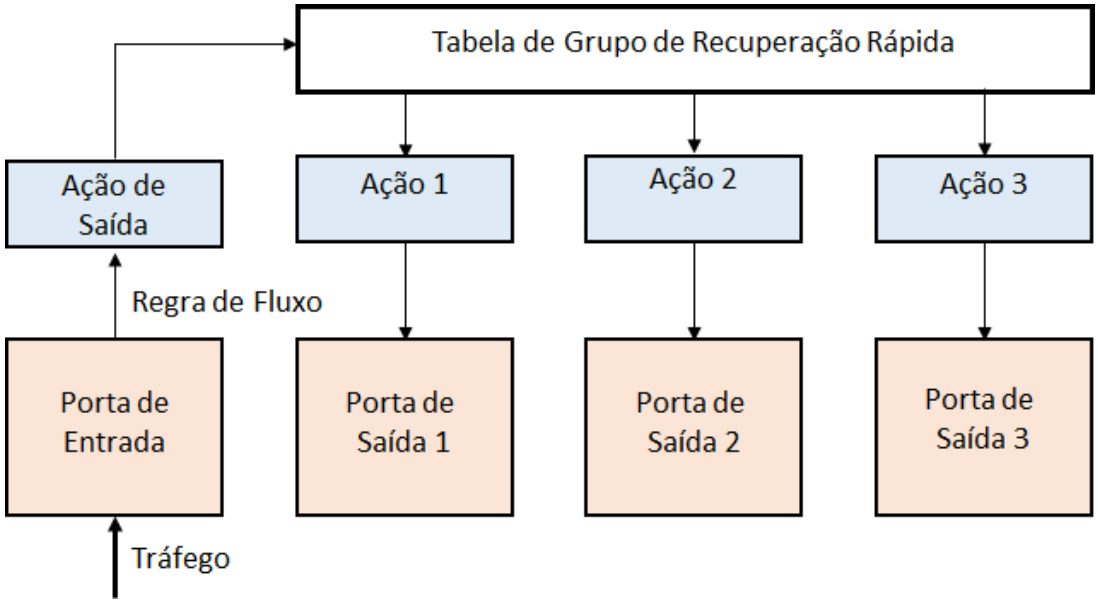
Quando ocorre uma falha na rede, é importante que a reconfiguração da rede para restaurar a conectividade seja realizada o mais rápido possível. Em uma abordagem baseada em SDN, uma forma reativa pode ser usada para reconfigurar a rede, como por exem-

Figura 8 - Processo de resiliência no controlador.



Fonte: Adaptado de (YEGANEH; TOOTOONCHIAN; GANJALI, 2013)

Figura 9 - Exemplo de Grupo de Recuperação Rápida.



Fonte: Adaptado de (ADRICHEM; ASTEN; KUIPERS, 2014).

plo, a resiliência pelo controlador. No entanto, é necessária uma quantidade significativa de tempo para recalcular os caminhos, uma vez que é necessário envolver o controlador SDN, adicionando o tempo de ida e volta para o caminho ser restabelecido (VESTIN; KASSLER; AKERBERG, 2015).

Com a implantação dos fluxos pré-configurados, um caminho secundário é preestabelecido no plano de dados. Caso ocorra uma falha, este caminho é utilizado, evitando o envio de solicitações para o controlador e eliminando o atraso de recuperação de ida e volta entre o plano de dados e o plano de controle (XIE et al., 2014). Além disso, esse tipo de mecanismo pode evitar uma sobrecarga no controlador causada por um grande número de eventos de controle gerados por uma rede de grande escala (YEGANEH; TOOTOONCHIAN; GANJALI, 2013).

Os fluxos pré-configurados podem ser implantados por meio da Tabela de Grupo de Recuperação Rápida, que é uma funcionalidade suportada a partir da versão 1.1 do protocolo *OpenFlow*. Essa funcionalidade pode ser configurada para monitorar o status de portas, interfaces e para alternar ações de encaminhamento independentemente do controlador (ADRICHEM; ASTEN; KUIPERS, 2014). Como mostra a Figura 9, após um tráfego ser recebido pela porta de entrada, uma ação de saída é definida. A Tabela de Grupos monitora continuamente um conjunto de portas de saída que podem encaminhar o tráfego caso seja necessário. Se o enlace de saída principal falhar, a Tabela de Grupos automaticamente encaminhará o tráfego para umas das portas que já estão sendo monitoradas e estão à disposição para serem utilizadas.

Nenhum método de fluxos pré-configurados é usado por padrão em uma rede que utiliza o controlador OpenDaylight.

Outro método avaliado nesta dissertação e que pode proporcionar uma melhora na resiliência, assim como os fluxos pré-configurados, é o protocolo BFD (*Bidirectional Forwarding Detection*).

3.3 Detecção de Encaminhamento Bidirecional (BFD)

Uma característica cada vez mais importante dos equipamentos de rede é a rápida detecção de falhas de comunicação entre sistemas adjacentes, a fim de estabelecer caminhos alternativos mais rapidamente. No entanto, existem meios que podem não detectar certos tipos de falhas no caminho como, por exemplo, falhas de interfaces. As redes usam mecanismos de *"Hello"* relativamente lentos, geralmente em protocolos de roteamento, para detectar falhas quando não há sinalização de *hardware* para ajudar. O tempo de detecção de falhas disponível nos protocolos existentes é lento para algumas aplicações e representa, consequentemente, uma grande quantidade de dados perdidos. É até possível diminuir o período de *Hello* que pode ser configurado em segundos, porém o protocolo BFD pode ser configurado com um tempo de detecção em milissegundos, possibilitando uma melhora no tempo da identificação da falha (KATZ; WARD, 2010).

O objetivo do protocolo BFD é fornecer detecção de falhas de baixa sobrecarga e de curta duração no caminho entre mecanismos de encaminhamento adjacentes. Um objetivo adicional é fornecer um mecanismo único que possa ser usado para detecção de atividade em qualquer meio, em qualquer camada de protocolo, com uma ampla variedade de tempos de detecção e sobrecarga, para evitar a proliferação de métodos diferentes (KATZ; WARD, 2010).

Uma sessão BFD separada é criada para cada enlace de comunicação entre dois sistemas, como por exemplo, dois nós de rede. Cada sistema comunica seu estado de sessão por meio de um pacote de controle BFD. Esse estado recebido, em combinação com o estado de sessão local, define o estado definitivo entre os sistemas. Existem quatro estados possíveis para uma sessão BFD (KATZ; WARD, 2010):

- estado *Down* - Significa que a sessão está inativa ou acabou de ser criada. Um sistema pode manter uma sessão no estado *Down* indefinidamente, simplesmente se recusando a avançar o estado da sessão. Isso pode ser feito por razões operacionais ou administrativas, entre outras;

- estado *Init* - Significa que o sistema remoto está se comunicando e o sistema local deseja "elevar a sessão", ou seja, passar para o próximo estado, mas o sistema remoto ainda não a percebe;

- estado *Up* - Significa que a sessão BFD foi estabelecida com sucesso e que a conectividade entre os sistemas está funcionando. A sessão permanecerá no estado *Up* até que a conectividade falhe, voltando para o estado *Down*, ou a sessão seja removida administrativamente, passando para o estado *AdminDown*;

- estado *AdminDown* - Significa que a sessão está administrativamente inativa. Isso faz com que o sistema remoto entre no estado *Down* e permaneça até que o sistema local saia do estado *AdminDown*.

Os valores de tempo usados para determinar os intervalos de transmissão de pacotes BFD e o tempo de detecção da sessão são continuamente negociados e, portanto, podem ser alterados a qualquer momento. Os valores de negociação e tempo são independentes em cada direção para cada sessão. Cada sistema relata no pacote de controle BFD com que frequência gostaria de transmitir pacotes BFD, bem como a frequência que está preparado para recebê-los. Isso permite que o sistema determine unilateralmente o intervalo mínimo em ambas as direções (KATZ; WARD, 2010).

O BFD implementa um mecanismo de controle de mensagens de eco para verificar se os enlaces estão ativos. Cada nó transmite mensagens de controle com o estado atual do enlace. Um nó que recebe uma mensagem de controle responde com uma mensagem de eco contendo o seu respectivo status de sessão. Após esse processo, as mensagens de controle, que são enviadas de forma frequente, confirmam a ausência de uma falha no enlace. Sob algumas condições, os sistemas podem negociar para não enviar pacotes BFD periódicos para reduzir a sobrecarga (ADRICHEM; ASTEN; KUIPERS, 2014). A Figura 10 ilustra todo esse processo de detecção de falha, no qual é possível notar que o nó A aguarda uma mensagem de resposta do nó B por um determinado período. Porém, se essa mensagem de resposta não é recebida dentro da janela de detecção, o status do enlace é alterado e a falha é confirmada.

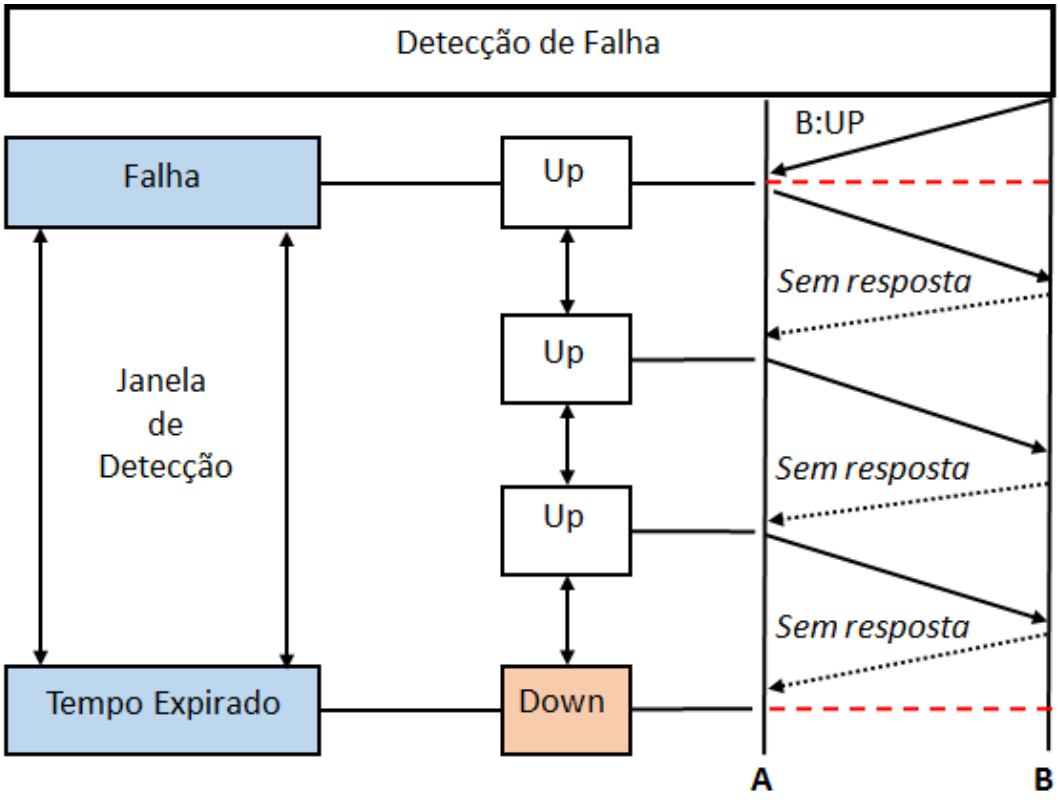
Nesta dissertação, o protocolo BFD, quando utilizado, é configurado no *Open vSwitch* (PFAFF et al., 2009), uma implementação popular de comutador *OpenFlow* (RODRIGUEZ; CAMPELO, 2013). O intuito é analisar as melhorias proporcionadas pelo seu rápido e bem definido sistema de detecção de falhas.

3.4 **Protocolo de Controle de Agregação de Enlace (LACP)**

O Protocolo de Controle de Agregação de Enlace, definido no padrão IEEE 802.3ad, é um recurso de configuração automática que permite que várias interfaces físicas agregadas tornem-se um único enlace lógico, chamado de Grupo de Agregação de Enlace (*Link Aggregation Group* - LAG).

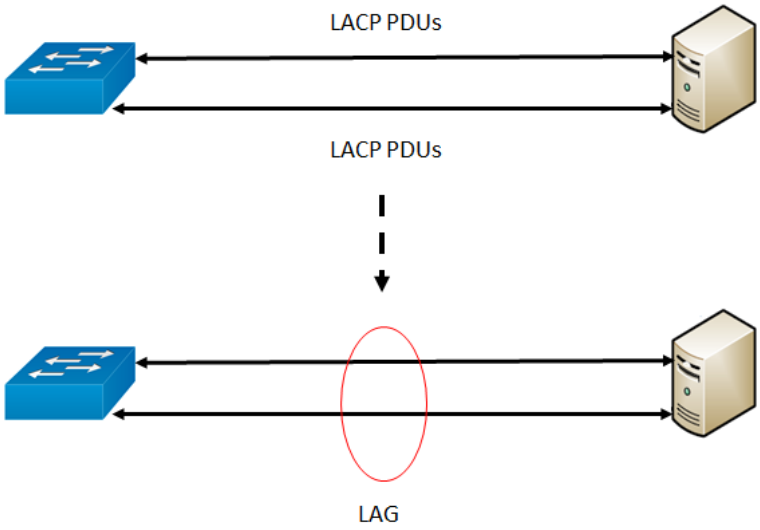
O LACP envia quadros denominados LACP PDUs para todos os enlaces que pos-

Figura 10 - Detecção de falha com a utilização do BFD.



Fonte: Adaptado de (ADRICHEM; ASTEN; KUIPERS, 2014).

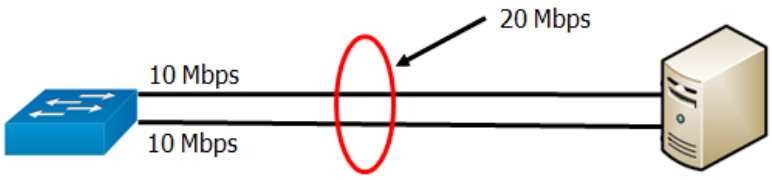
Figura 11 - Troca de LACP PDUs entre os elementos e formação do LAG.



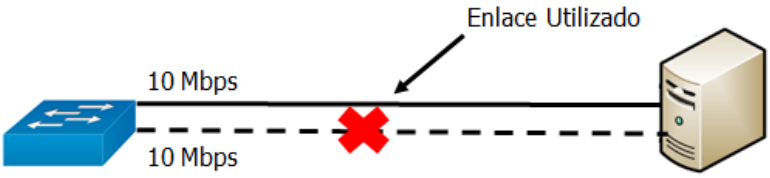
suem o protocolo habilitado. A troca de LACP PDUs entre dispositivos permite que duas unidades detectem vários enlaces entre eles e os combinem em um único enlace lógico, como ilustra a Figura 11. O LACP pode ser configurado em um dos dois modos: ativo ou passivo. No modo ativo, ele sempre enviará quadros ao longo dos enlaces configurados. No modo passivo, no entanto, ele apenas responde ao receber quadros da outra extremidade. Nesse modo, é necessário que a outra extremidade esteja configurada como modo ativo (IEEE, 2000).

Além de proporcionar um aumento da vazão devido à agregação de diversos enlaces, o LACP também ajuda na convergência rápida em caso de falhas, pois reconfigura os enlaces de forma rápida e automática. Dessa forma, o protocolo LACP pode ser utilizado para proteção do enlace. A Figura 12(a) mostra o aumento da largura de banda devido à agregação de dois enlaces entre dispositivos. Já a Figura 12(b) mostra como ocorre a tolerância a falhas em caso de queda de um dos enlaces, assegurando a comunicação confiável por meio de múltiplos enlaces físicos que são usados como secundários em caso de falhas (IRAWATI; HADIYOSO; HARIYANI, 2017).

Figura 12 - Função da Agregação de Enlaces.



(a) Aumento da largura de banda



(b) Tolerância a falhas

Fonte: Adaptado de (IRAWATI; HADIYOSO; HARIYANI, 2017)

4 TRABALHOS RELACIONADOS

Com o objetivo de situar esta dissertação em relação aos trabalhos relacionados disponíveis na literatura, as seções a seguir apresentam artigos que têm como foco o estudo dos métodos de resiliência em SDN e artigos que combinam a implementação de tecnologias diversas com SDN.

4.1 Resiliência em SDN

Os artigos relacionados que estudam resiliência em SDN são divididos em duas situações distintas de falhas: falhas em nós e enlaces e falha no controlador.

4.1.1 Falhas em nós e enlaces

Métodos de resiliência em SDN são objetos de estudos em diversos trabalhos da literatura. Van Adrichem *et al.* propõem um método de recuperação rápida baseado na identificação de falhas de enlaces com a utilização do BFD, combinando caminhos primários e secundários configurados por um controlador central (ADRICHEM; ASTEN; KUIPERS, 2014). A proposta é dividir o processo de recuperação em duas etapas. A primeira etapa consiste em uma recuperação rápida iniciada pelo comutador com base em regras de encaminhamento pré-configuradas que garantem conectividade de ponta a ponta. O segundo passo envolve o controlador calculando e configurando novos caminhos ótimos. Com a implementação desta abordagem proposta no artigo, é possível alcançar uma recuperação abaixo de 50 ms.

Já em (VESTIN; KASSLER; AKERBERG, 2015) é abordada uma arquitetura resiliente baseada em SDN para redes de controle industrial que combina várias tecnologias rápidas de recuperação baseadas em SDN. Essas tecnologias usam o protocolo BFD e caminhos primários e secundários pré-configurados. O BFD é implementado para detectar rapidamente falhas nos enlaces, trocando um fluxo rápido de mensagens de controle que acionam localmente o comutador para usar o caminho secundário. Já os caminhos primário e secundário pré-configurados são implementados utilizando tabelas de grupo de recuperação rápida, no qual monitora o status de portas, interfaces e executa ações de encaminhamento independentemente do controlador. Além disso, é utilizado a duplicação de pacotes, orquestrada por um controlador SDN, para aumentar a resiliência contra a perda de pacotes. Os pacotes duplicados são recebidos e descartados com base no número de sequência. Com a utilização dessas três técnicas, é possível reduzir significativamente a

latência de controle e fornecer garantias de desempenho mais rigorosas, mesmo em enlaces com perdas.

Em (PARIS; PASCHOS; LEGUAY, 2016) é abordada a possibilidade de adoção de diversos mecanismos de roteamento que reagem instantaneamente a variações inesperadas de tráfego e falhas de rede. Baseado nisso, é apresentado um sistema de roteamento dinâmico, concentrado no plano de controle, no qual esses mecanismos se esforçam para resolver uma instância de otimização em caso de falhas de rede e reparos. Esse tipo de método, que utiliza como *backup* os caminhos calculados pelo controlador no momento da falha, tem uma convergência mais demorada, já que inclui o tempo de propagação da notificação de falha ao controlador e o tempo da reconfiguração da rede. Em contrapartida esses métodos podem utilizar diversos parâmetros de configuração, como banda disponível e latência.

A ideia de pré-instalar entradas de fluxo redundantes no plano de dados foi abordada em (XIE et al., 2014), no qual propõe-se o aprimoramento do plano de dados por meio de rotas pré-configuradas. Esta abordagem possibilita que 60% dos pedidos de reconexão, que ocorrem em função de falhas na rede, possam ser manipulados neste plano. Ademais, como apenas uma fração das solicitações são enviadas para o plano de controle, ocorre uma diminuição na sobrecarga de controle.

Em (CASCONE et al., 2015) também são propostas formas de resiliência em SDN utilizando o plano de dados por meio da criação de algoritmos de recuperação local, que têm como objetivo preservar a conectividade sob falhas arbitrárias de enlace. A proposta é projetar um esquema de proteção capaz de se recuperar de falhas sem o envolvimento direto do controlador. Esses algoritmos podem ser implementados diretamente dentro dos padrões modernos do *OpenFlow*. Isso permite obter um melhor tempo de recuperação e superar problemas como falta de resposta do controlador, devido a atrasos no caminho ou inacessibilidade.

Outra solução analisada como um método para recuperação rápida em caso de falhas corresponde à implementação da agregação de enlaces por meio do uso do protocolo LACP (IRAWATI; HADIYOSO; HARIYANI, 2017), sendo possível integrar várias portas físicas em conjunto para criar um único enlace lógico de comunicação.

Artigos relacionados como (ADRICHEM; ASTEN; KUIPERS, 2014), (XIE et al., 2014), (CASCONE et al., 2015), (VESTIN; KASSLER; AKERBERG, 2015) e (IRAWATI; HADIYOSO; HARIYANI, 2017) analisam o tempo de recuperação dos métodos de resiliência e de seus respectivos mecanismos, porém não fazem uma análise de desempenho da rede de forma mais profunda, ou seja, não analisam métricas importantes que precisam ser verificadas em redes resilientes. Assim, nesta dissertação é realizada uma análise de desempenho da rede quantificando o *jitter*, o RTT e a perda de pacotes em cenários com falha. Além disso, é realizada uma comparação entre cenários que utilizam os métodos de resiliência analisados nesta dissertação com cenários sem a implementação desses métodos.

4.1.2 Falhas no controlador

Em (FONSECA et al., 2013) cita-se que a maioria das arquiteturas SDN usa uma abordagem centralizada de gerenciamento de rede. No entanto, tal abordagem levanta, entre outros problemas, a questão de um único ponto de falha, que pode comprometer o bom funcionamento da rede. Com base nisso, é abordada a técnica de replicação do controlador SDN como um método para atingir um nível alto de resiliência de rede. Foi investigado como diferentes técnicas de replicação se relacionam entre si e como cada uma desempenha a tarefa de fornecer resiliência em SDN. Foram definidas as técnicas de replicação que são principalmente classificadas em dois tipos: replicação passiva e ativa. No caso de replicação passiva, o cliente se conecta a apenas um controlador que processa as solicitações e atualiza os outros controladores. Na replicação ativa, o cliente se conecta a vários controladores que processam cada solicitação. Em função dos resultados obtidos chegou-se à conclusão que a replicação é uma forma adequada e simples de aumentar a resiliência em SDN, já que não será necessário que o controlador secundário reinicie todo o processo de configuração da rede em caso de falha, evitando uma maior indisponibilidade de serviço e recurso. Além disso, a replicação oferece a garantia de que as informações de rede permaneçam consistentes, aumentando a confiabilidade do sistema.

Segundo (YEGANEH; TOOTOONCHIAN; GANJALI, 2013), em um *data center*, por exemplo, existem dezenas de milhares de comutadores e isso pode crescer a um ritmo acelerado. O grande número de eventos de controle gerados em qualquer rede dessa escala é suficiente para sobrecarregar qualquer controlador centralizado. Uma maneira de enfrentar esse problema é utilizar controladores distribuídos. Dada a baixa latência em tais redes, a configuração de estado e fluxo de controladores distribuídos para se manterem com a mesma configuração tem uma latência mínima e aceitável para a maioria das aplicações. Além dos controladores distribuídos, o artigo ainda aborda a instalação de regras de forma proativa nos comutadores, eliminando a maioria dos pedidos de controle e consequentemente diminuindo a sobrecarga.

Em (SRIDHARAN; GURUSAMY; TRUONG-HUU, 2017) também é estudada a utilização de controladores distribuídos como método de resiliência. É desenvolvido um esquema que mapeia um comutador para vários controladores e distribui solicitações de configuração de fluxo entre eles minimizando o tempo de configuração geral do fluxo na rede. Além disso, é possível proporcionar justiça em termos de tempo de configuração do fluxo a cada comutador individual e garantindo a restrição de resiliência. Juntamente com a introdução da resiliência, foi modelado matematicamente o tempo de resposta do controlador usando a teoria das filas.

O posicionamento do controlador é um dos problemas críticos no projeto de SDN, que estuda como selecionar os melhores nós em SDN para executar os controladores a fim de otimizar uma função objetivo. Os problemas de posicionamento aparecem em muitos

contextos e receberam extensos estudos na literatura. Além disso, a maioria desses problemas estuda o posicionamento para melhorar o desempenho, como a minimização da latência. O estudo realizado em (GUO; BHATTACHARYA, 2013) faz a análise do impacto do posicionamento do controlador na resiliência em SDN, define uma nova métrica de resiliência baseada na análise de falhas e propõe uma abordagem que foca no posicionamento do controlador para melhorar a resiliência.

Não faz parte do escopo dessa dissertação o tratamento de falhas que ocorrem no controlador.

4.2 Trabalhos que combinam a implementação de tecnologias diversas com SDN

O estudo feito em (SAVAS et al., 2016) menciona que em redes de comunicação de alta capacidade, especialmente em redes de multiplexação por divisão de comprimento de onda (WDM), falhas na rede podem levar a grandes perdas de dados, causando falhas em grande escala. Devido a esses problemas foi realizado um estudo de resiliência em redes ópticas utilizando SDN, no qual é possível realizar a configuração dos caminhos simultaneamente em todos os comutadores, por meio de mensagens de configuração de fluxo enviadas pelo controlador. Essas novas funções em SDN, que levam a uma maior e mais fácil dinâmica na rede, podem ser exploradas para fornecer melhor capacidade de sobrevivência contra desastres. Usando o plano de controle centralizado e a visão geral da rede que o SDN oferece, foi proposto um esquema que executa continuamente o reprovisionamento de *backup* em cada mudança de estado da rede. Por fim, foi criada uma heurística para avaliar a vulnerabilidade da rede a diferentes tipos de desastres em diferentes locais.

Em (YANG et al., 2014) foi proposto um novo algoritmo de resiliência integrada de recursos globais (*Global Resources Integrated Resilience* - GRIR) para uma arquitetura de interconexão de *data center* definida por *software* baseada em redes ópticas. O algoritmo proposto fornece resiliência usando os recursos das camadas IP e óptica e aprimora a capacidade de resposta da resiliência do serviço de *data center* para as demandas dinâmicas de recuperação de ponta a ponta em caso de falha. Os resultados das simulações indicam que o algoritmo proposto alcança melhorias significativas em termos de probabilidade de bloqueio de caminho, latência da rede e taxa de ocupação de recursos, em comparação com outros algoritmos de resiliência.

No trabalho (MACHADO; GRANVILLE; SCHAEFFER-FILHO, 2016) é apresentada uma arquitetura que combina recursos de NFV (*Network Functions Virtualization*) e SDN para criar estratégias sofisticadas de resiliência de rede. Cada VNF (*Virtualized Network Function*) compreende um conjunto de ações corretivas dinâmicas, como o enca-

minhamento do tráfego para uma VNF específica. Porém, quando somente as VNFs não conseguem lidar com as anomalias, a arquitetura proposta explora os recursos de SDN para monitorar e analisar o comportamento da infraestrutura de rede, indicando se parte de uma estratégia de resiliência existente pode ser reconfigurada para obter resultados mais satisfatórios ou se uma estratégia de resiliência inteira precisa ser adicionada ou substituída. Além disso, a arquitetura pode rapidamente identificar e lidar com anomalias, como por exemplo, sobrecarga de tráfego de rede e interrupção de serviço distintos em diferentes cenários, indicando que a reconfiguração e a implantação de estratégias de resiliência podem ser realizadas em tempo real.

Esta dissertação avalia o desempenho de métodos de resiliência em redes SDN. Essa avaliação é realizada através de experimentos apresentados a seguir.

5 AVALIAÇÃO DE DESEMPENHO

Para analisar o desempenho dos métodos apresentados, foram realizados experimentos envolvendo uma máquina Intel Core i5 com 4 GB de RAM e CPU de 2,50 GHz. O sistema operacional utilizado foi o Ubuntu 18.04.5 LTS.

A topologia da Figura 13 foi utilizada para realizar as análises dos métodos de proteção. Essa topologia foi criada com diversos caminhos entre H1 e H2 para possibilitar a realização de testes de resiliência com utilização dos métodos analisados. Todos os enlaces são de 10 Mbps e possuem uma latência de 1 ms. Os testes são realizados utilizando o *Ping* e o *Iperf*, com tráfego UDP à taxa de 10 Mbps. Este valor foi utilizado por restrições do hardware no qual a rede foi emulada; entretanto, o objetivo foi escolher uma taxa de envio que saturasse a capacidade dos enlaces da rede. O controlador SDN utilizado para os experimentos é o *OpenDaylight* e a emulação dos comutadores e estações é realizada pelo *Mininet*.

Foram realizados experimentos envolvendo métodos de proteção do caminho e de proteção do enlace. A Tabela 1 ilustra os cenários utilizados para a realização da análise de desempenho dos métodos de proteção do caminho. Já a Tabela 2 especifica os cenários utilizados no método de proteção do enlace para a realização dos experimentos.

5.1 Métricas de Desempenho

Os experimentos deste trabalho utilizam o *jitter*, o RTT (*Round Trip Time*), a perda de pacotes e a vazão como métricas para analisar os métodos de resiliência. O *jitter* é a variação do atraso, ou seja, a variação do tempo decorrido entre o momento em que um pacote é gerado na fonte e o momento em que é recebido no destinatário (KUROSE; ROSS, 2010). Esta métrica é medida por meio do comando *Iperf*, que possibilita analisar detalhadamente a variação do atraso. Já o RTT é o atraso entre o momento em que

Tabela 1 - Cenários de análise dos métodos de proteção do caminho.

Método de Proteção do Caminho
Cenários
Sem Queda
Rota Pré-Configurada
Rota Pré-Configurada + BFD
Rota Controlador
Rota Controlador + BFD

Figura 13 - Topologia de rede utilizada nos testes.

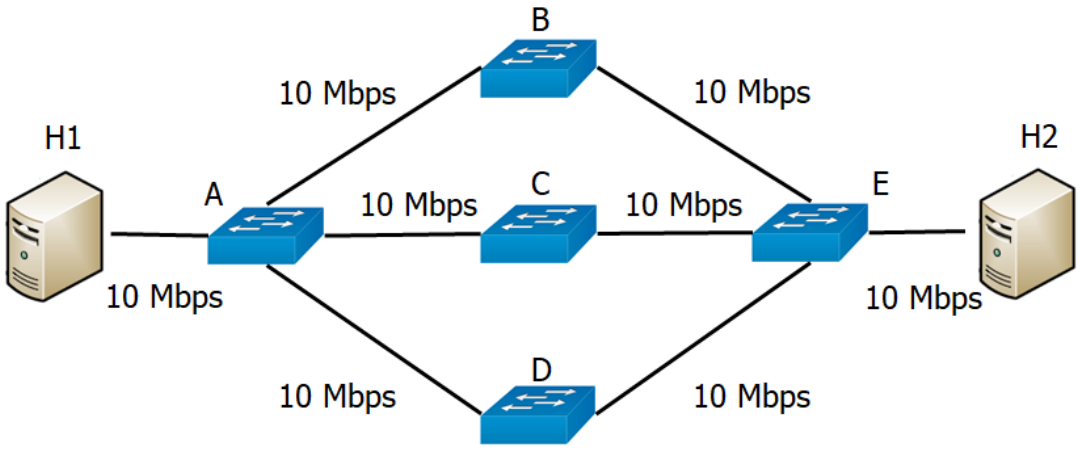


Tabela 2 - Cenários de análise dos métodos de proteção do enlace.

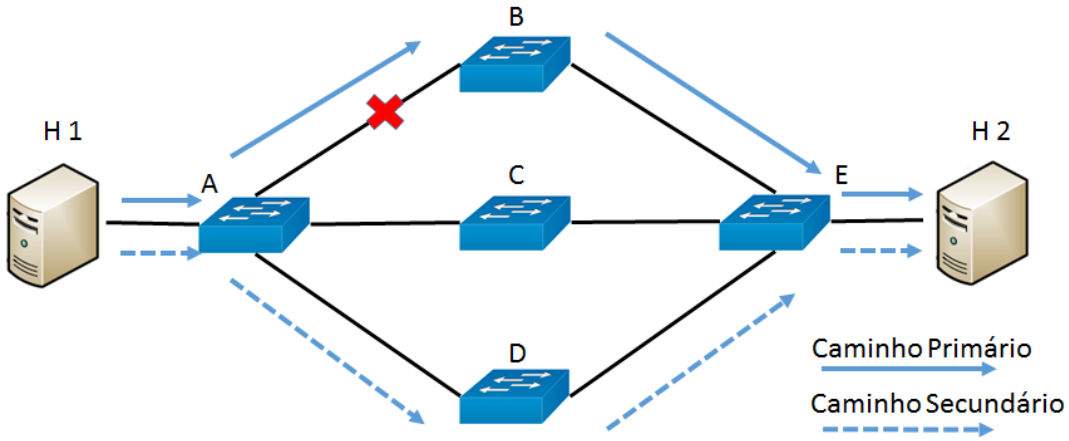
Método de Proteção do Enlace
Cenários
Sem Queda
Agregação LACP

o cliente envia a mensagem e a recebe de volta, ou seja, é o tempo de viagem de ida e volta (KUROSE; ROSS, 2010). O comando *Ping*, além de testar a acessibilidade aos dispositivos da rede, também é utilizado nesta dissertação para medir o valor do RTT. Na realização dos experimentos e medições das métricas, o *Ping* e o *Iperf* não foram executados ao mesmo tempo. Através destas métricas, também se pode determinar o nível de estabilidade da rede com a utilização dos métodos de resiliência analisados nesta dissertação. Além disso, para provedores de serviços, está se tornando cada vez mais importante monitorar e gerenciar essas métricas devido a aplicativos altamente interativos e sensíveis (ZANDER; ARMITAGE, 2013).

A métrica de perda de pacotes, que também é medida utilizando o *Iperf*, é utilizada para medir a quantidade de pacotes que não são entregues, sendo possível definir o desempenho da rede com a utilização dos métodos analisados. A análise desta métrica também permite inferir o congestionamento da rede que pode afetar a qualidade da comunicação ou até mesmo inviabilizá-la (KUROSE; ROSS, 2010).

Este trabalho utiliza também a métrica da vazão. A vazão é a quantidade de dados por segundo que pode ser transferido entre dois sistemas finais (KUROSE; ROSS, 2010). Com exceção dos gráficos que mostram a vazão ao longo do tempo, todas as métricas foram apresentadas com médias e intervalos de confiança de 95%.

Figura 14 - Topologia com caminhos primário e secundário no núcleo da rede.



Com relação aos parâmetros utilizados nas configurações dos métodos analisados, no caso dos fluxos pré-configurados, são definidas no grupo de recuperação rápida as portas de destino que podem ser utilizadas em caso de falha. No BFD o protocolo é configurado em todas as interfaces assim como o encaminhamento de mensagens de controle que permanece ativo enquanto houver pacotes recebidos, ou seja, enquanto o Rx for verdadeiro. Além disso, no LACP a configuração é realizada utilizando todos os parâmetros padrões do protocolo.

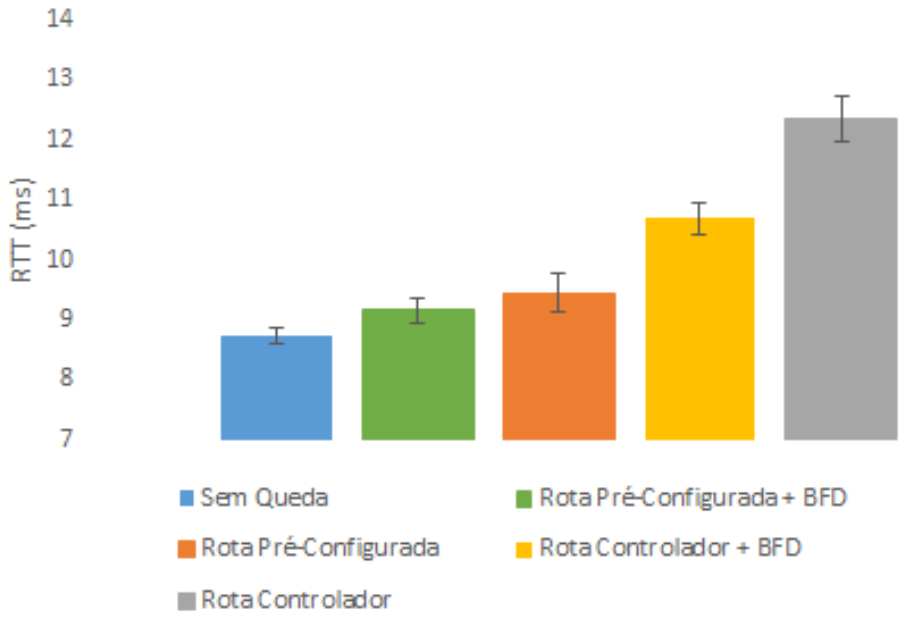
5.2 Resultados dos Métodos de Proteção do Caminho

Para realizar as análises dos métodos de proteção do caminho, ou seja, dos fluxos pré-configurados e dos fluxos pré-configurados com BFD, são definidos caminhos primários e secundários como mostra a Figura 14. Na topologia, o caminho primário é o A-B-E e o caminho secundário é o A-D-E. Para a realização dos testes, o enlace A-B é "derrubado" por meio de uma linha de comando aplicada no *Mininet*.

A Figura 15 ilustra os resultados do RTT para os mecanismos de proteção do caminho em todos os cenários. Nota-se que ocorre uma diminuição do RTT em um cenário que existe uma rota pré-configurada em comparação com uma rota definida pelo controlador. O RTT é cerca de 3 ms menor em um cenário com rota pré-configurada. Essa diferença ocorre já que as solicitações tratadas no plano de dados não serão enviadas para o plano de controle, eliminando assim o tempo de comunicação que seria necessário entre o plano de dados e o plano de controle e consequentemente melhorando o desempenho da métrica utilizada, de acordo com (XIE et al., 2014).

Com a implementação do BFD é possível observar uma melhora do RTT nos

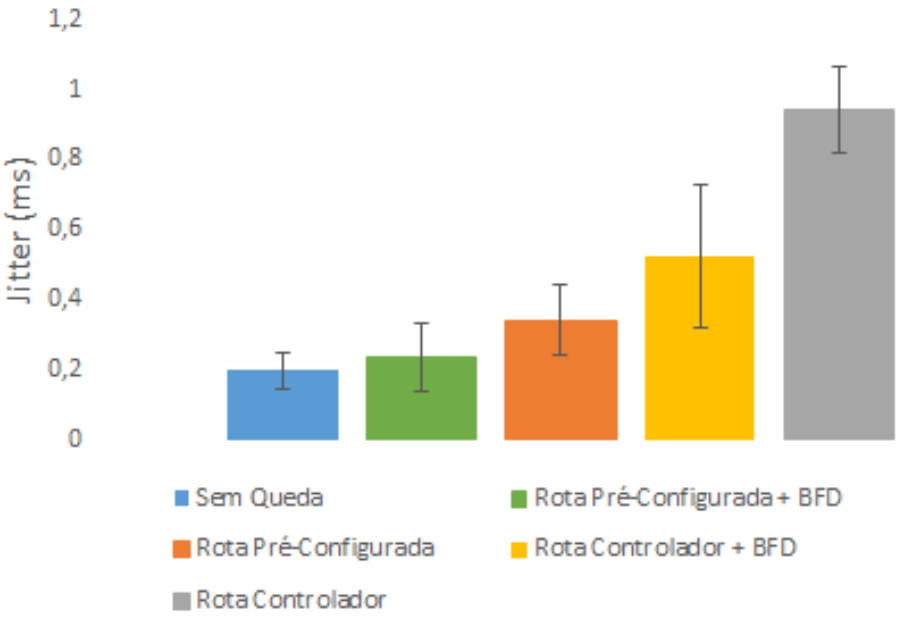
Figura 15 - RTT - Mecanismos de proteção do caminho.



cenários com rota pré-configurada e rota definida pelo controlador. No cenário com rota pré-configurada ocorre uma queda de cerca de 0,3 ms após a implementação do BFD nos enlaces, aproximando-se ainda mais do valor de RTT no cenário em que não ocorre queda. Já no cenário com rota definida pelo controlador, o RTT diminui cerca de 2 ms. Os melhores resultados em relação ao RTT ocorrem devido à troca de mensagens de eco proporcionada pelo BFD, que propicia a vantagem de se testar frequentemente o caminho percorrido entre dois nós. Isso pode permitir um menor tempo de detecção, além de potencialmente detectar falhas que poderiam não ser detectadas, de acordo com (KATZ; WARD, 2010). Consequentemente, o RTT diminui. É possível observar que apenas com a implementação da rota pré-configurada os valores de RTT se aproximam muito do valor de um cenário sem queda. Como consequência disso, temos um ganho menor com a implementação do BFD neste cenário em comparação com o cenário que utiliza como resiliência apenas o controlador. O RTT de um cenário que utiliza o controlador como resiliência é maior e possibilita uma margem de ganho melhor.

A Figura 16 mostra os resultados do *jitter* para cada um dos mecanismos de proteção do caminho. O *jitter* em um cenário em que existe uma rota pré-configurada no comutador é cerca de 0,6 ms menor em comparação com uma rota definida pelo controlador. É possível verificar também uma melhora no *jitter* após ser acrescentado aos mecanismos de fluxo pré-configurado e rota definida pelo controlador, a configuração do BFD nos enlaces. Nota-se uma diminuição do *jitter* de cerca de 0,5 ms em uma rota definida pelo controlador e 0,2 ms para uma rota pré-configurada, aproximando-se ainda mais do *jitter* do cenário sem queda.

Figura 16 - Jitter - Mecanismos de proteção do caminho.



A Figura 17 mostra os resultados de perda de pacotes para os mecanismos de proteção do caminho. O cenário com a rota pré-configurada perde cerca de oito vezes menos pacotes em comparação com o cenário em que a decisão é tomada apenas pelo controlador. O cenário sem queda foi omitido nessa figura, já que em todas as rodadas a taxa de perda de pacotes é zero. Para ilustrar melhor os momentos das quedas, a Figura 18 mostra a vazão na linha do tempo para uma das amostras dos experimentos com a rota pré-configurada e com a rota do controlador. Essa figura mostra a estabilidade quando o fluxo pré-configurado é utilizado.

Ainda na Figura 17, é possível notar uma diminuição de 2,1 pontos percentuais no número de pacotes perdidos na rota definida pelo controlador e cerca de 0,25 pontos percentuais na rota pré-configurada quando o BFD é utilizado em conjunto. Na Figura 19 é ilustrada a vazão em função do tempo ao se usar o BFD em conjunto, tornando-se possível concluir que, mesmo com a implantação do BFD, a rota pré-configurada continua sendo melhor que a do controlador.

5.3 Resultados dos Métodos de Proteção do Enlace

Para realizar as análises do método de proteção do enlace, ou seja, das agregações de enlaces utilizando o LACP, a topologia anterior foi modificada de acordo com a Figura 20. Todos os enlaces da topologia são de 10 Mbps, porém entre H1 e A existem dois enlaces agregados, ou seja, um LAG. Para a realização dos testes de proteção do enlace,

Figura 17 - Perda de Pacotes - Mecanismos de proteção do caminho.

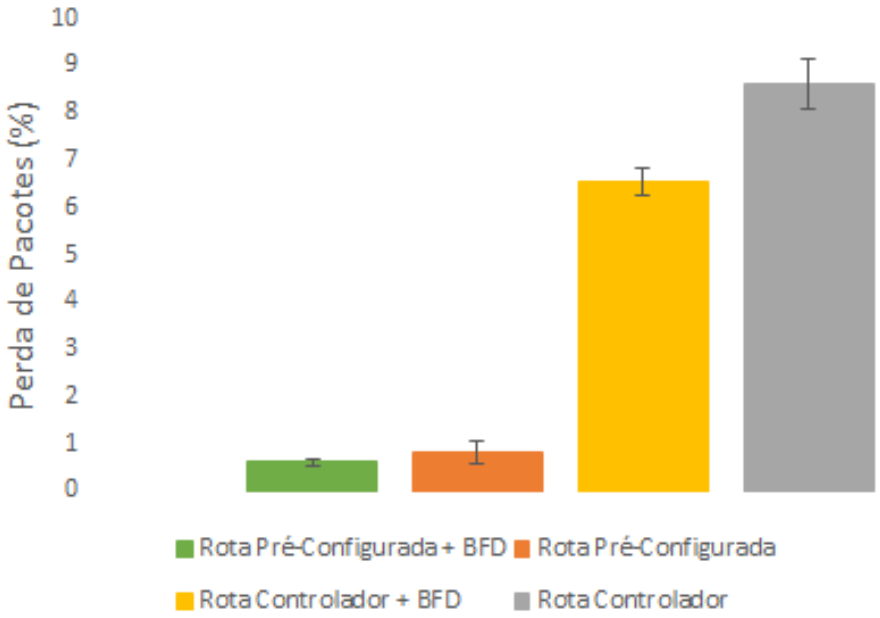


Figura 18 - Vazão ao longo do tempo - Rota Pré-configurada e Rota Controlador.

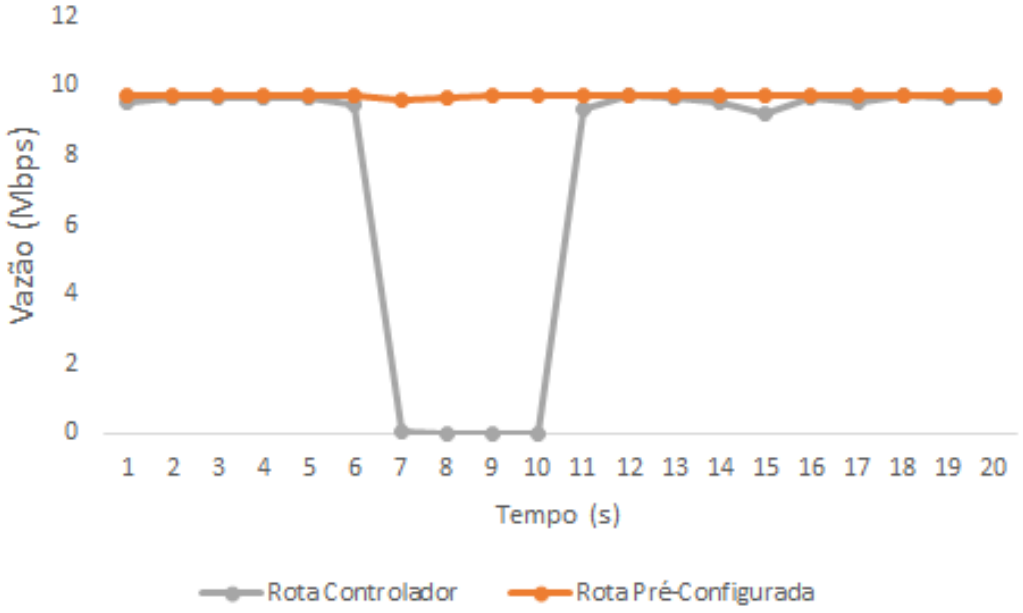
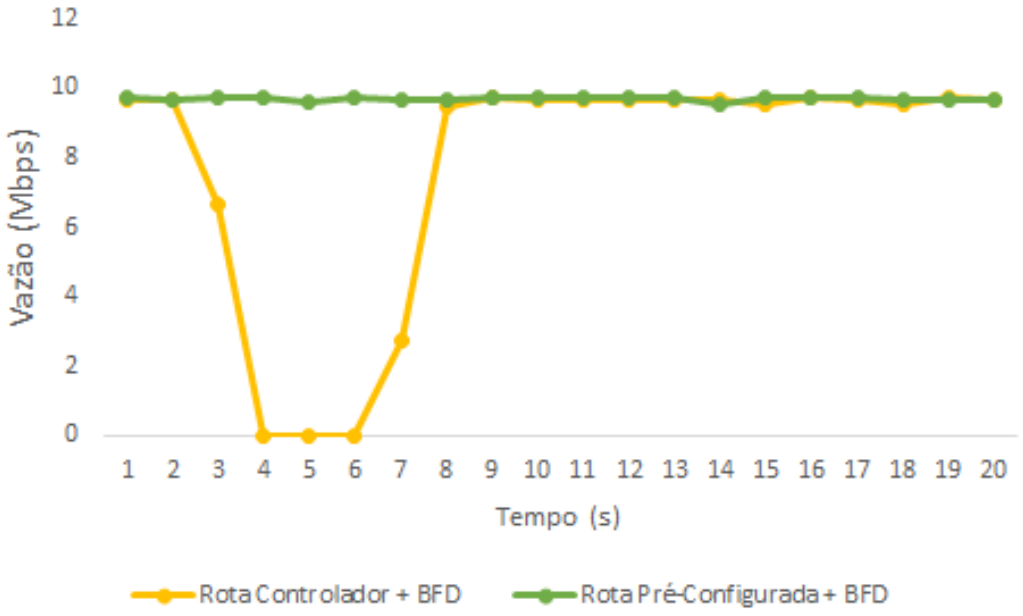


Figura 19 - Vazão ao longo do tempo - Rota Pré-configurada + BFD e Rota Controlador + BFD.



um dos enlaces físicos do LAG é derrubado.

A Figura 21 mostra o *jitter* para duas situações: uma sem a queda do enlace e a outra com a queda do enlace. É possível analisar que mesmo com uma queda em um dos enlaces físicos do LAG, o *jitter* é estatisticamente igual em comparação com um cenário sem falhas na rede.

A Figura 22 ilustra o RTT dos dois cenários. Assim como no jitter, mesmo em caso de falhas, a agregação de enlaces formada pelo LACP também mantém o RTT estatisticamente igual em comparação a um cenário no qual no ocorrem falhas.

Com relação à perda de pacotes, a taxa de perdas foi zero mesmo com a queda de um dos enlaces físicos, pois a taxa de transmissão no enlace é de 10 Mbps. Além disso, em função da vazão ao longo do tempo para o caso de agregação usando o LAG (Figura 23), é possível notar que a resiliência ocorre de forma automática mantendo a transferência de pacotes. Com base nos resultados encontrados, é possível concluir que a transferência de dados apenas não será realizada em caso de falha no LAG, ou seja, no enlace lógico formado por todos os enlaces físicos, conforme apresentado em (SEVCIK et al., 2009).

Figura 20 - Topologia com enlaces primário e secundário na borda da rede.

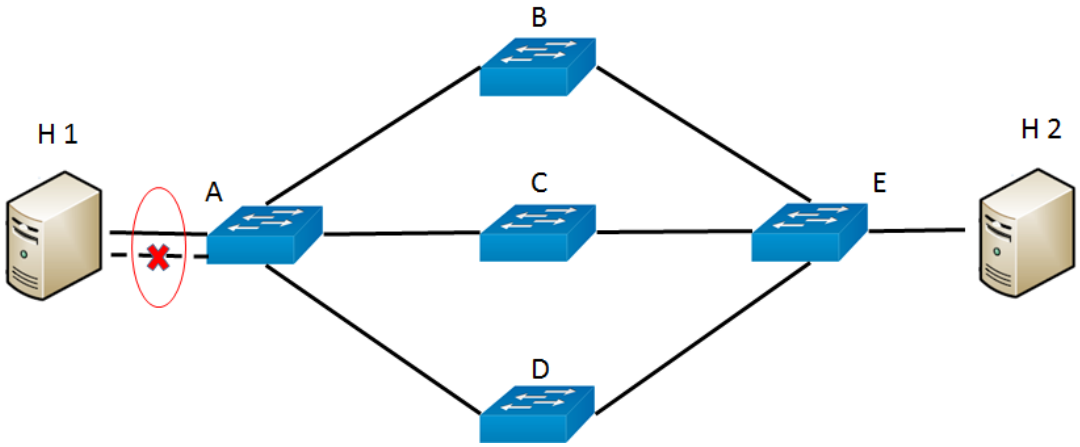


Figura 21 - Jitter - Agregação de enlace formado pelo LACP.

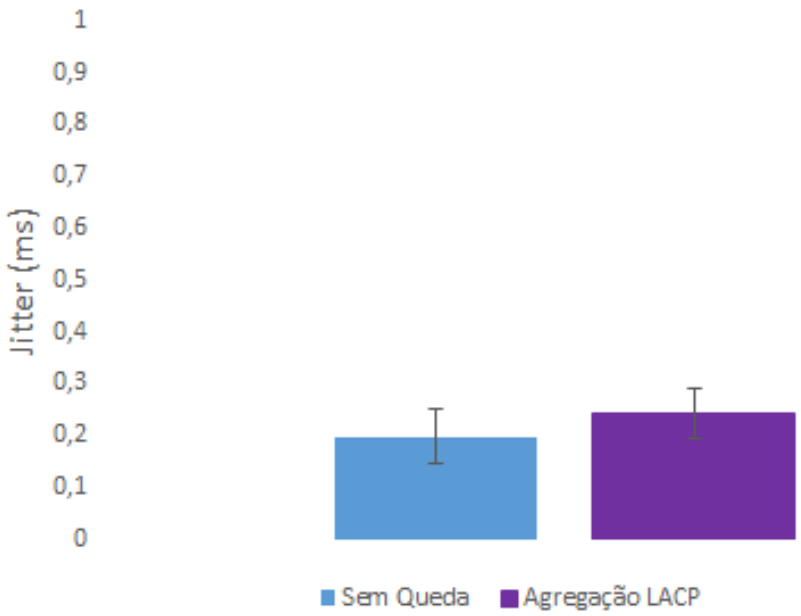


Figura 22 - RTT - Agregação de enlace formado pelo LACP.

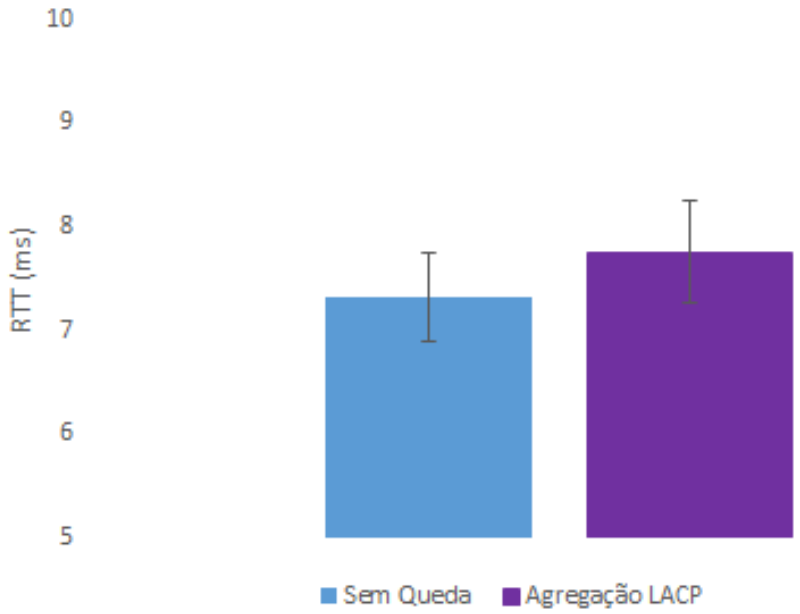
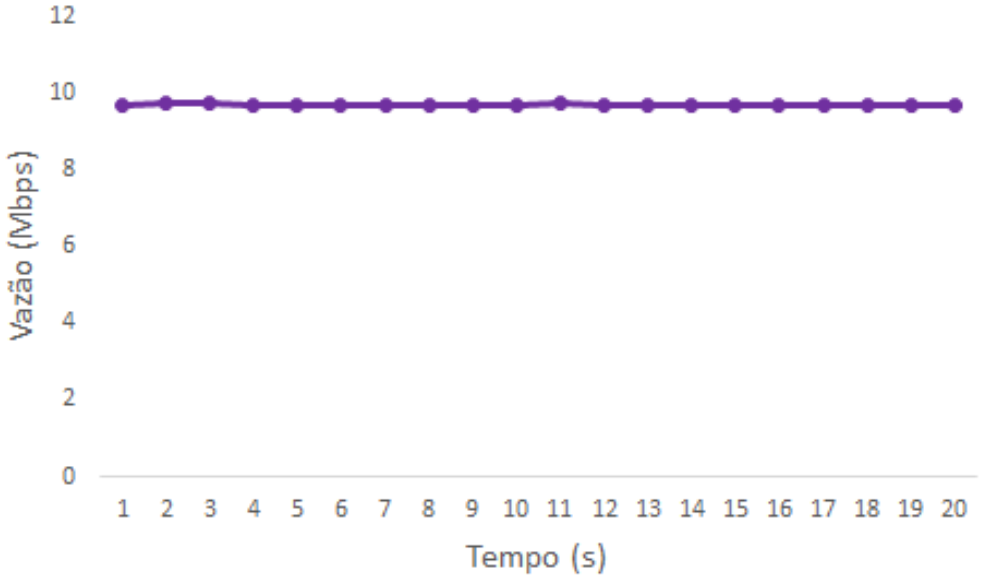


Figura 23 - Vazão ao longo tempo - Agregação de enlace formado pelo LACP.



6 CONCLUSÃO

A necessidade de redes cada vez mais estáveis e resilientes tem transformado arquiteturas que utilizam SDN em uma das principais opções de implementação. No entanto, redes SDN têm um controlador centralizado e recalcular caminhos após uma falha pode não ser a melhor opção. Com base nisso, os métodos de proteção do caminho e proteção do enlace podem ser implementados para alcançar um melhor desempenho em redes SDN resilientes.

Esta dissertação fez uma análise de desempenho de métodos de resiliência que podem ser implementados em uma rede SDN. Esta análise foi realizada com base nas métricas de desempenho *jitter*, RTT, a perda de pacotes e vazão. Com relação ao método de proteção do caminho, que foi implementado no núcleo da rede, foi possível concluir que a funcionalidade de rota pré-configurada diminui consideravelmente o *jitter*, o RTT e a perda de pacotes em comparação com uma rota definida pelo controlador. Ao adicionar a implementação do BFD nos enlaces, foi possível melhorar ainda mais o desempenho no cenário com rota pré-configurada. Além disso, a rota controlador também obteve melhora relevante após a implementação do BFD. Isso se dá devido à capacidade do BFD de testar frequentemente se os enlaces entre os nós estão ativos.

O método de proteção do enlace, que foi testado na borda da rede, também mostrou-se eficaz em caso de falhas. Com a implementação da agregação de enlaces realizada pelo LACP, o *jitter* e o RTT mantiveram-se estatisticamente iguais à de uma rede sem falhas e não houve perda de pacotes mesmo com a queda de um dos enlaces do LAG. Além disso, foi possível concluir que o tráfego não é interrompido devido à falha no enlace.

O gerenciamento centralizado de redes SDN simplifica a tarefa de gerenciar os serviços e reduz o nível de dificuldade de execução, no entanto, levanta entre outras questões, o problema da vulnerabilidade da rede. Isso ocorre porque, no caso de falha no controlador, toda a rede pode ser comprometida, uma vez que todos os aplicativos e serviços dependem dele. O protocolo *OpenFlow* oferece a possibilidade de configurar um ou mais controladores de *backup* que podem assumir o controle da rede em caso de falha, evitando a perda de configurações dos componentes e informações coletadas anteriormente. Sendo assim, como trabalhos futuros pretende-se analisar a resiliência em redes com mais de um controlador, no intuito de evitar a existência de um único ponto de falha que pode comprometer o bom funcionamento da rede. Além disso, pode ser estudado como os métodos de resiliência de proteção do caminho e proteção do enlace, analisados nesta dissertação, podem melhorar a estabilidade e confiabilidade da rede neste cenário.

REFERÊNCIAS

ADRICHEM, N. L. V.; ASTEN, B. J. V.; KUIPERS, F. A. Fast recovery in software-defined networks. In: *Software Defined Networks (EWSDN), Third European Workshop on Software Defined Networks, IEEE*. [S.l.: s.n.], 2014. p. 61–66.

CALLE, E.; MARZO, J. L.; URRA, A. Protection performance components in mpls networks. In: *Computer Communications*. [S.l.: s.n.], 2004. v. 27, n. 12, p. 1220–1228.

CASCONE, C. et al. Traffic management applications for stateful SDN data plane. In: *Software Defined Networks (EWSDN), Fourth European Workshop on Software Defined Networks, IEEE*. [S.l.: s.n.], 2015. p. 85–90.

FONSECA, P. et al. Resilience of SDNs based on active and passive replication mechanisms. In: *Global Communications Conference (GLOBECOM), IEEE*. [S.l.: s.n.], 2013. p. 2188–2193.

GAY, S.; HARTERT, R.; VISSICCHIO, S. Expect the unexpected: Sub-second optimization for segment routing. In: *INFOCOM Conference on Computer Communications, IEEE*. [S.l.: s.n.], 2017. p. 1–9.

GUO, M.; BHATTACHARYA, P. Controller placement for improving resilience of software-defined networks. In: *Networking and Distributed Computing (ICNDC), Fourth International Conference on Networking and Distributed Computing, IEEE*. [S.l.: s.n.], 2013. p. 23–27.

IEEE. *IEEE Standard for Ethernet Link Aggregation:IEEE802.3ad*. 2000. Disponível em: ⟨http://www.ieee802.org/3/ad/⟩. Acesso em: 2019.

IRAWATI, I. D.; HADIYOSO, S.; HARIYANI, Y. S. Link aggregation control protocol on software defined network. In: *International Journal of Electrical and Computer Engineering*. [S.l.: s.n.], 2017. v. 7, n. 5, p. 2706.

KATZ, D.; WARD, D. *Bidirectional forwarding detection (BFD)*. [S.l.], 2010.

KHATTAK, Z. K.; AWAIS, M.; IQBAL, A. Performance evaluation of opendaylight sdn controller. In: *20th IEEE international conference on parallel and distributed systems (ICPADS), IEEE*. [S.l.: s.n.], 2014. p. 671–676.

KIM, H.; FEAMSTER, N. Improving network management with software defined networking. In: *Communications Magazine, IEEE*. [S.l.: s.n.], 2013. v. 51, n. 2, p. 114–119.

KREUTZ, D. et al. Software-defined networking: A comprehensive survey. In: *Proceedings of the IEEE*. [S.l.: s.n.], 2015. v. 103, n. 1, p. 14–76.

KUROSE, J. F.; ROSS, K. W. *Redes de Computadores e a Internet (5ª Edição)*. [S.l.]: São Paulo, SP: Pearson Addison Wesley, 2010.

MACHADO, C. C.; GRANVILLE, L. Z.; SCHAEFFER-FILHO, A. Answer: Combining nfv and sdn features for network resilience strategies. In: *Computers and Communication (ISCC), Symposium on Computers and Communication, IEEE*. [S.l.: s.n.], 2016. p. 391–396.

MEDVED, J. et al. Opendaylight: Towards a model-driven sdn controller architecture. In: *World of Wireless, Mobile and Multimedia Networks (WoWMoM), 15th International Symposium on a World of Wireless, Mobile and Multimedia Networks, IEEE*. [S.l.: s.n.], 2014. p. 1–6.

MUKHERJEE, B. et al. Survivable wdm mesh networks part 2 restoration. In: *International Conference on Communications, IEEE*. [S.l.: s.n.], 1999. p. 23–30.

PARIS, S.; PASCHOS, G. S.; LEGUAY, J. Dynamic control for failure recovery and flow reconfiguration in sdn. In: *Design of Reliable Communication Networks (DRCN), 12th International Conference, IEEE*. [S.l.: s.n.], 2016. p. 152–159.

PFAFF, B. et al. Extending networking into the virtualization layer. In: *Hotnets*. [S.l.: s.n.], 2009.

RAMAMURTHY, S.; MUKHERJEE, B. Survivable wdm mesh networks. part i-protection. In: *INFOCOM. Eighteenth Annual Joint Conference of the Computer and Communications Societies. Proceedings, IEEE*. [S.l.: s.n.], 1999. v. 2, p. 744–751.

RODRIGUEZ, F. L.; CAMPELO, D. R. Rede SDN-openflow para o caso de um ISP: Desafios e oportunidades. *SBrT*, 2013.

SAVAS, S. S. et al. Backup reprovisioning with partial protection for disaster-survivable software-defined optical networks. In: *Photonic Network Communications*. [S.l.]: Springer, 2016. v. 31, n. 2, p. 186–195.

SEVCIK, B. et al. Network layer based redundancy for time-critical voip applications. In: *AFRICON, IEEE*. [S.l.: s.n.], 2009. p. 1–5.

SHENKER, S. et al. The future of networking, and the past of protocols. In: *Open Networking Summit*. [S.l.: s.n.], 2011. v. 20, p. 1–30.

SRIDHARAN, V.; GURUSAMY, M.; TRUONG-HUU, T. On multiple controller mapping in software defined networks with resilience constraints. In: *Communications Letters, IEEE*. [S.l.: s.n.], 2017. v. 21, n. 8, p. 1763–1766.

VESTIN, J.; KASSLER, A.; AKERBERG, J. Resilient software defined networking for industrial control networks. In: *Information, Communications and Signal Processing (ICICS), 10th International Conference on Information, Communications and Signal Processing, IEEE*. [S.l.: s.n.], 2015. p. 1–5.

XIE, A. et al. Designing a disaster-resilient network with software defined networking. In: *Quality of Service (IWQoS), 22nd International Symposium of Service (IWQoS), IEEE*. [S.l.: s.n.], 2014. p. 135–140.

YANG, H. et al. Global resources integrated resilience for software defined data center interconnection based on ip over elastic optical network. In: *Communications Letters, IEEE*. [S.l.: s.n.], 2014. v. 18, n. 10, p. 1735–1738.

YEGANEH, S. H.; TOOTOONCHIAN, A.; GANJALI, Y. On scalability of software-defined networking. In: *Communications Magazine, IEEE*. [S.l.: s.n.], 2013. v. 51, n. 2, p. 136–141.

ZANDER, S.; ARMITAGE, G. Minimally-intrusive frequent round trip time measurements using synthetic packet-pairs. In: *38th Annual Conference on Local Computer Networks, IEEE*. [S.l.: s.n.], 2013. p. 264–267.