Serviço de L2VPN SDN em Redes de *Backbone*: estudo de caso da REDECOMEP-Rio

Pedro Henrique Diniz da Silva, Natália Castro Fernandes, Nilton Alves Jr. e Márcio Portes de Albuquerque

***Resumo*— As redes OpenFlow introduzem o conceito de controladores centralizados para gerenciamento do comportamento do encaminhamento dos elementos de rede. Esse conceito permite que diversos serviços em redes de *backbone*, atualmente implementados através de soluções proprietárias, distribuídas pelos diversos elementos e de grande complexidade de operação, sejam simplificados facilitando a operação de rede. Esse trabalho propõe uma nova aplicação para provimento de conexões de Redes Privadas Virtuais de Camada 2 (L2VPN, do inglês *Layer 2 Virtual Private Network*) em redes de *backbone*, para caso de uso do *backbone* acadêmico, de pesquisa e de governo da cidade do Rio de Janeiro (REDECOMEP-Rio). São apresentados os algoritmos implementados através do protocolo OpenFlow e controlador Ryu. A solução proposta é avaliada através de cenários emulados por meio do emulador Mininet e de um estudo de caso executado na rede de produção da REDECOMEP-Rio.**

***Palavras-Chave*— L2VPN; Redes Definidas por Software; OpenFlow; Controlador Ryu.**

***Abstract*— OpenFlow networks introduce the concept of centralized controllers for managing the forwarding behavior of network elements. This concept allows multiple services in backbone networks, currently implemented through proprietary solutions, distributed by the various elements, and of highly complex operation, to be simplified to facilitate network operation. This paper proposes a new application for provision of Layer 2 Virtual Private Networks (L2VPN) in backbone networks for use case of academic, research, and government backbone network of the city of Rio de Janeiro (REDECOMEP-Rio). We present the algorithms implemented using the OpenFlow protocol and Ryu controller. We also evaluate the proposed solution through emulated scenarios using Mininet emulator and a case study implemented in the production network of Redecomep-Rio.**

***Keywords*— L2VPN; Software Defined Networks; OpenFlow; Ryu Controller.**

I. INTRODUÇÃO

A RedeRio Metropolitana (REDECOMEP-Rio) [1] é uma infraestrutura de fibras óticas próprias que formam uma rede de alta velocidade para as instituições de ensino, ciência, tecnologia, inovação e governo na cidade do Rio de Janeiro. Apesar de ser um *backbone* que atende instituições acadêmicas, a REDECOMEP-Rio, que opera como uma rede de núcleo IP tradicional puramente roteada, também atende a diversos outros tipos de instituições, as quais incluem empresas ligadas à prefeitura, ao estado e ao governo federal. Além de serviços de conectividade de alta velocidade à Internet e de alta confiabilidade, essas instituições também carecem de serviços que simulem redes privadas em cima dessa rede de núcleo compartilhada entre os diversos afiliados. Esse tipo de rede privada é também conhecido como Rede Privada Virtual (VPN, do inglês *Virtual Private Network*).

As VPNs mais comuns são as de camada 3 do modelo de referência TCP/IP (L3VPN) [2], ou seja, fazem com que dois elementos de rede pareçam estar diretamente conectados via uma rede roteada, ainda que existam diversos elementos de rede (roteadores) no meio do caminho. Entretanto, existem também as VPNs de camada 2. Esse tipo de rede virtual são conexões ponto-a-ponto que simulam um circuito físico de camada 2. Sua principal vantagem é a transparência do enlace virtual formado ponta-a-ponta, permitindo que qualquer protocolo, e não somente o IP, possa trafegar.

Apesar de cada uma das tecnologias apresentadas oferecer diversas vantagens e desvantagens uma em relação a outra, lidar com a variedade de protocolos para a criação de VPNs torna a gerência e operação da infraestrutura de rede mais complexa para os seus operadores. Essas tecnologias são difíceis de implementar, ou operacionalmente quase impossíveis de realizar em larga escala em redes IP puramente roteadas [3], como o caso da REDECOMEP-Rio, devido principalmente a complexidade de configuração dos equipamentos envolvidos. Alguns trabalhos, como [4] demonstram a utilização do protocolo OpenFlow para provimento de serviços de VPNs em redes MPLS (do inglês, *Multi Protocol Label Switching*), porém não é de conhecimento soluções de VPNs sobre redes IP tradicionais.

Com o enfoque em simplificar e otimizar o processo de operação de uma rede de núcleo puramente roteada, propõe-se uma solução de estabelecimento de circuitos L2VPN que evita a complexidade de ter protocolos de encapsulamento adicionais para o estabelecimento das conexões ponto-a-ponto, sem necessidade de configurações adicionais nos equipamentos envolvidos e centraliza a configuração das conexões em um ponto único da rede. Essa solução escala naturalmente ao passo que a mesma depende, basicamente, do processo de descoberta de topologia do protocolo OpenFlow.

A aplicação desenvolvida para o estabelecimento de L2VPNs instala regras de fluxos OpenFlow reativamente acada circuito que passa pelo *backbone*, baseado em um arquivo de configuração centralizado. Essa aplicação visa ser implementada na operação da REDECOMEP-Rio como um serviço na rede de produção.

Na próxima seção, é apresentada a arquitetura proposta de estabelecimento de L2VPNs. Apresenta-se, também, na Seção III, uma avaliação do protótipo e um estudo de caso

Pedro Henrique Diniz da Silva, Nilton Alves Jr. e Márcio Portes de Albuquerque¸ Coordenação de Atividades Técnicas, Centro Brasileiro de Pesquisas Físicas, Rio de Janeiro-RJ, Brasil, E-mails: phds@cbpf.br, naj@cbpf.br, mpa@cbpf.br. Natália Castro Fernandes¸ Departamento de Engenharia de Telecomunicações, Universidade Federal Fluminense, Niterói-RJ, Brasil, E-mail: nataliacf@id.uff.br.

na REDECOMEP-Rio. O artigo é concluído na Seção IV.

II. Características Principais da Aplicação Proposta

Uma rede de *backbone* IP consiste de múltiplos roteadores do tipo *Provider Edge (PE)*, que conectam o roteador de borda *Customer Edge* (CE) à rede de núcleo do provedor, e roteadores do tipo *Provider* (P), os quais funcionam como um roteador de trânsito na rede de núcleo entre os PE, conforme mostrado na Fig. 1.

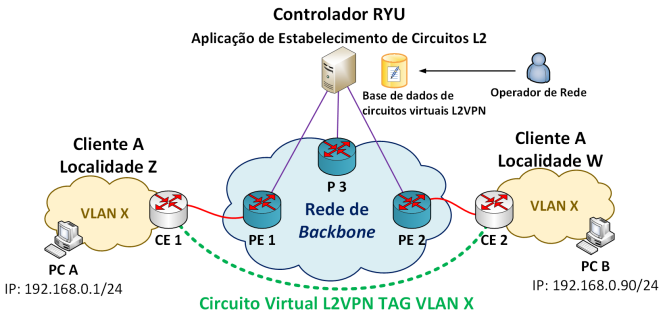


Fig. 1. Modelo básico da visão da aplicação de estabelecimento de circuitos L2.

Na solução proposta, os administradores de rede do cliente informam aos operadores de rede do provedor as *tags* de VLAN 802.1q [5] utilizadas em cada localidade nas quais desejam estabelecer um circuito L2VPN, e os operadores somente precisam armazenar as informações do circuito (como, por exemplo, *tag* de VLAN e roteadores PE envolvidos) em uma base de dados. A aplicação, então, é responsável por direcionar o tráfego de maneira adequada através da rede de núcleo, baseando-se nas informações obtidas através da base de dados e utilizando o algoritmo de escolha de menor caminho *Shortest Path First* (SPF) [6].

Nesta seção, primeiramente, são descritas as características principais do protocolo OpenFlow utilizadas nessa solução. Em seguida, é descrito como o operador de rede configura sua base de dados de estabelecimento de circuitos L2VPN. Em sequência, é descrito a utilização do algoritmo de menor caminho SPF para estabelecimento entre os roteadores PE.

A. Características Principais do Protocolo OpenFlow

O OpenFlow é um protocolo que permite que tabelas de fluxos em *switches* e roteadores sejam remotamente gerenciadas por um controlador. O protocolo define um fluxo como uma tupla com valores tais como: porta física de entrada, endereço MAC de origem e destino, campo tipo do cabeçalho *Ethernet*, identificador de VLAN, endereços IP de origem e de destino, campo protocolo do cabeçalho IP, porta de origem e destino do protocolo de camada de transporte. O *pipeline* de tabelas de fluxos em um *switch* OpenFlow mapeia a definição de um fluxo através dessa tupla em uma ação a ser tomada nos pacotes que pertencem a esse fluxo. Algumas dessas ações podem ser descartar os pacotes, encaminhá-los em uma porta física específica ou em um conjunto delas, ou enviar os pacotes para o controlador. Em caso de um pacote não corresponder a nenhuma das entradas da tabela de fluxos, o *switch* pode ser configurado para armazenar em um *buffer*, encapsular e enviar o pacote ao controlador para inspeção. Quando o controlador toma a decisão referente ao que fazer com todos os pacotes com aquela característica descrita pela tupla de campos do cabeçalho, ele adiciona uma entrada para esse fluxo na tabela de fluxos para armazenar essa decisão. Além disso, existe também a opção do controlador instalar entradas pró-ativamente para que esse processo não seja novamente repetido. Essas entradas são também conhecidas como regras.

O OpenFlow também implementa um conjunto de mensagens de sinalização e controle trocadas entre o controlador e o *switch*. Essas mensagens são responsáveis por realizar diversas ações como: verificação de características, configuração, modificação de estados, leitura de estados, envio de pacotes e mensagens de barreira. Dentre os tipos de mensagens mais relevantes estão o *packet_in*, o *packet_out* e o *flow_mod*. O *packet_in* é uma mensagem assíncrona enviada do *switch* para o controlador para notificar a chegada de um fluxo não classificado. O *packet_out* é uma mensagem enviada do controlador para o *switch*, em resposta a um *packet_in*, indicando qual ação deve ser tomada para aquele pacote. Já o *flow_mod* é também enviado do controlador para o *switch* para modificar o estado do mesmo, podendo realizar diversos comandos como adição e modificação de entradas na tabela de fluxos.

Em nossa solução de L2VPN, o controlador realiza algumas funções principais como: o procedimento de descoberta de topologia; verificação dos *endpoints* (*switches* PE) do circuito virtual L2VPN a ser estabelecido; escolha do menor caminho entre todos os *switches* na rede e encaminhamento do pacote na porta do *switch* associada ao caminho entre os *endpoints*; realização do VLAN *stitching*, garantindo a consistência do circuito fim-a-fim. Essas funções serão descritas em mais detalhes adiante.

B. Verificação dos Endpoints e VLAN Stitching

A aplicação proposta se baseia nos dados armazenados em duas bases de dados: (i) gerenciada pelo operador de rede para armazenar a configuração do circuito virtual; (ii) utilizada para o processo de VLAN *stitching* no núcleo da rede. No caso da base de configuração a cada cinco minutos, cada entrada de configuração é checada em sequência em busca de modificações.

Na base de dados de configuração, cada entrada é representada por uma tupla de seis valores: porta *switch* PE 1, *switch* PE 1, VLAN ID de acesso no *switch* PE 1, porta *switch* PE 2, *switch* PE 2 e VLAN ID de acesso no *switch* PE 2.

Para lidar com as situações em que um mesmo cliente utiliza *tags* de VLANs distintas nos locais em que é atendido, e para decisão de qual *tag* é utilizada no núcleo da rede sem que haja sobreposição, outra base de identificação é utilizada. Nela, cada circuito passa a ser identificado por uma tupla de quatro valores: *switch* PE 1, VLAN ID de acesso no *switch* PE 1, *switch* PE 2, VLAN ID de acesso no *switch* PE 2. A partir desses dados, seleciona-se sequencialmente em função das *tags* já em uso no núcleo, qual a próxima a ser utilizada e

armazena-se essa informação na base, conforme ilustrado na Fig. 2.

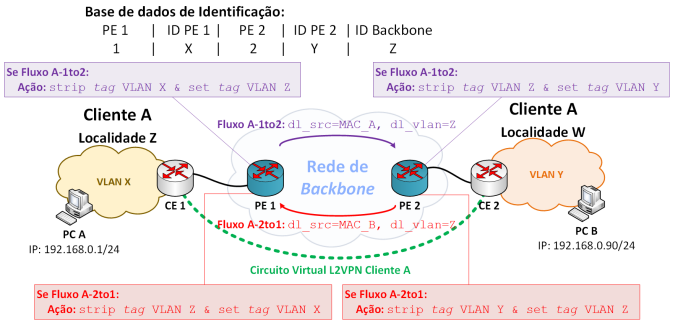


Fig. 2. Modelo básico da visão da aplicação de estabelecimento de circuitos L2 para processo de VLAN *stitching*.

C. Escolha do Menor Caminho e Estabelecimento do Circuito

A cada pacote do tipo *packet_in*, o controlador analisa as informações dos cabeçalhos do pacote, verificando para cada entrada de configuração se: (i) o *switch* de entrada do pacote é um dos *endpoints* do circuito; (ii) a porta de entrada do *switch* é referente a um dos circuitos; (iii) a *tag* de VLAN do pacote é igual ao VLAN ID de entrada de algum circuito. Caso essas informações se confirmem, o *switch* realiza as ações de associar o endereço MAC de origem do pacote ao *endpoint*, de substituir a *tag* de entrada pela ID de *backbone* do circuito e de encaminhar o pacote. Ao chegar no *endpoint* de destino a operação inversa é realizada. Caso o *switch* que recebe o pacote não seja um *endpoint*, mas a *tag* seja associada a um ID de circuito no *backbone* e o MAC de origem esteja vinculado a um dos *endpoints*, o pacote é encaminhado para o outro *endpoint*. Então, para estabelecer o caminho entre o *switch* que envia a mensagem de *packet_in* e o *switch* PE de destino do circuito, a aplicação utiliza o algoritmo de *Shortest Path First* [6]. Dessa forma, a aplicação é capaz de estabelecer o circuito L2VPN fim-a-fim reativamente a cada mensagem de *packet_in*, encaminhando o fluxo à porta de saída de cada *switch*, obtida em função do algoritmo SPF em conjunto com as bases de dados de configuração e de identificação de circuitos. Para evitar que ocorram *loops* na topologia utilizou-se também o protocolo *Spanning Tree* (STP).

A aplicação proposta instala uma regra com *soft-timeout* de 60 segundos para poder lidar com o processo de alteração da configuração de um circuito. Então, após uma mudança por parte do operador de rede, há a necessidade de aguardar: (a) um período de 60 segundos de inatividade para que a entrada da tabela de fluxo de cada *switch* no caminho seja deletada; (b) um período de 5 minutos para que a configuração possa ser verificada novamente e o novo circuito possa ser estabelecido a partir de novas mensagens de *packet_in* - podendo ambos os intervalos (a) e (b) ocorrerem concorrentemente.

III. Implementação e Avaliação do Funcionamento da Aplicação

Nesta seção, buscou-se avaliar o tempo de convergência da aplicação visto pelo *switch* CE, definido como o intervalo de tempo entre o início do funcionamento da aplicação e o momento em que os protocolos STP e SPF já convergiram, ou seja, quando os *switches* PE OpenFlow iniciam o encaminhamento de pacotes baseados nas mensagens de *packet_in* e *packet_out*. Para tal, os resultados foram subdivididos em dois cenários: topologias emuladas e estudo de caso da implantação no *Backbone* REDECOMEP-Rio. A avaliação de desempenho ilustra como o procedimento para o estabelecimento do circuito é eficaz e como o sistema escala adequadamente, aparentando ter pouco *overhead* em função da complexidade da rede. A complexidade, nesse caso, é considerada como o número total de nós, isto é, o número total de *switches* mais *hosts*.

Para validar a proposta nos cenários de topologias emuladas, construiu-se um protótipo utilizando o OpenVswitch (um *switch* OpenFlow via *software*) e o controlador OpenFlow Ryu, por meio do emulador de redes Mininet [7]. A aplicação Ryu instala as regras nos *switches* do caminho do circuito L2VPN por meio do protocolo OpenFlow v1.3. Nesse protótipo, todos os procedimentos de verificação do circuito e de escolha de menor caminho foram desenvolvidos conforme descritos nas Seções II-B e II-C.

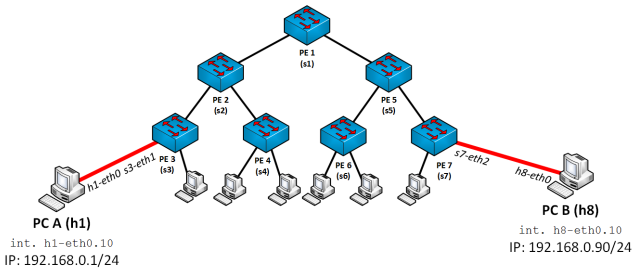
Para avaliar o funcionamento da aplicação, foi medido o tempo de convergência na rede, considerando o uso do algoritmo STP, disponibilizado por padrão no controlador Ryu, com o algoritmo SPF, desenvolvido para a aplicação, e com algoritmo proposto de estabelecimento de conexões L2VPN. Emulou-se o processo de estabelecimento de uma conexão entre dois *hosts* de um mesmo cliente, assumindo que o circuito já estava previamente descrito no arquivo de configuração. Foi medido o intervalo de tempo entre o primeiro pacote ARP *Request* enviado e sem resposta e o primeiro pacote ARP *Request* que recebe resposta. Os resultados são apresentados com um intervalo de confiança de 95%.

A. Cenários Emulados

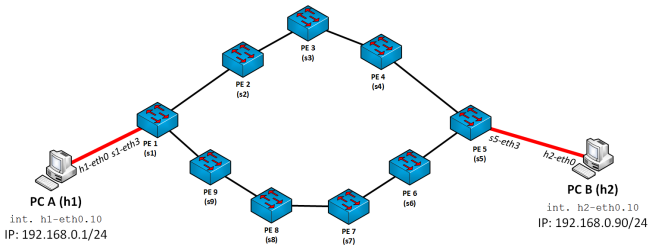
Para realizar a avaliação do funcionamento da aplicação, emulou-se três diferentes topologias. Utilizou-se duas topologias padrão do Mininet e uma customizada, conforme descritas em detalhes adiante. Para geração dos pacotes para estabelecimento de conexão e geração dos pacotes ARP, utilizamos a ferramenta FPING [8], com intervalos entre mensagens ICMP *Echo Request* de 10 ms. Os intervalos entre os pacotes ARP foram checados através da ferramenta *tcpdump* [9], capturando pacotes na interface do *host* de origem.

1. **Topologia Linear com 2 *Switches*:** Como avaliação inicial, utiliza-se a topologia mais simples disponível no emulador com dois *switches*. Os *switches* são conectados diretamente e dois *hosts* são conectados cada um a um dos *switches*. ***2.*** **Topologia *Fat-tree* com Três Níveis e Sete *Switches*:** A topologia *fat-tree* é um tipo de topologia voltada para *data centers*. Realizou-se a avaliação de funcionamento em uma topologia de três níveis com sete *switches* e oito *hosts* conectados de acordo com a Fig. 3(a). ***3.*** **Topologia Emulada do *Backbone* REDECOMEP-Rio:** Como a aplicação visa ser implementada na rede em produção da REDECOMEP-Rio, optou-se por realizar uma emulação em uma infraestrutura

similar à sua topologia real. A topologia de *backbone* da REDECOMEP-Rio, atualmente, é composta por nove Pontos de Presença (PoP, do inglês *Point-of-Presence*) espalhados por uma topologia em anel com múltiplas redundâncias. A Fig.3(b) exibe o *layout* da topologia da rede em produção simplificada. Foram conectados nove *switches* em anel com dois *hosts*. Nesse cenário, cada *host* faz o papel dos *switches* CE que devem ser conectados aos *switches* PE de *backbone*. Vale ressaltar que como a topologia é em anel, ocorrem *loops* do *Spanning Tree*.



(a) Fat-tree com sete *switches* e oito *hosts*.



(b) Emulação da REDECOMEP-Rio com 9 *switches* e 2 *hosts*.

Fig. 3. *Layout* das topologias emuladas.

Os resultados referentes aos cenários emulados podem ser visualizados na Fig. 4(a). O tempo de convergência é em grande parte dominado pelo tempo de convergência do STP, que para todos os casos apresentados é de 30s. Isso ocorre de acordo com o apresentado em [10]. A partir do instante em que o STP converge a aplicação demora somente 1,5s para iniciar o encaminhamento de pacotes. Assim como os casos em que não ocorrem *loops*, o tempo de convergência é em grande parte dominado pelo tempo de convergência do STP, cerca de 30s. Porém, após uma porta entrar em estado *BLOCKED* pelo STP, é necessário mais 13,5s para que o LLDP, implementado por padrão no Ryu, identifique que o enlace está bloqueado e a aplicação, a partir de então, possa recalcular os caminhos. Desde o momento em que o enlace passa a estar bloqueado, a aplicação leva cerca de 1s para encaminhar os pacotes.

Cabe ainda ressaltar que a partir do momento em que a aplicação já convergiu, o estabelecimento do circuito depende somente da latência entre os *switches* e do tempo em que as mensagens de *packet_out* e *flow_mod* demoram para serem processadas e enviadas pelo controlador. Como pode ser observado na Fig. 4(b), a função de distribuição acumulada (CDF) dos tempos de processamento dessas mensagens demonstra que esse intervalo representa cerca de 0,017% do tempo total de convergência para o pior dos casos de processamento de mensagens *flow_mod*, ou seja, a topologia emulada *fat-tree*. Logo, isso mostra que a partir da convergência da aplicação, o tempo para estabelecimento do circuito é muito pequeno.

B. Estudo de Caso da Implantação Inicial no Backbone REDECOMEP-Rio

Apresenta-se, a seguir, a avaliação do sistema protótipo na REDECOMEP-Rio. O estudo de caso foi executado em uma configuração com equipamentos híbridos, ou seja, realizando o encaminhamento dos pacotes da rede em produção no mesmo enlace que os pacotes SDN/OpenFlow. O cenário descrito foi avaliado utilizando os equipamentos: 2 (PE) x Cisco ASR9000 rodando IOS XR 5.1.3 equipados com pelo menos 6 GB DRAM; 1 (CE) x Cisco Catalyst WS-C3560G-24TS rodando IOS 12.2(50)SE5; 1 (CE) x Cisco Catalyst WS-C3750G-24TS-1U rodando 12.2(25)SEE2. A associação entre os roteadores Cisco ASR9000 e o controlador ocorre *in-band*. Todos os dispositivos são conectados com enlaces de 1 Gbps de capacidade, conforme mostrado na Fig. 5. A conexão entre o *switch* CE e o roteador PE ocorre através de uma interface do tipo tronco com permissão de tráfego das VLANs 802.1q 10 e 100.

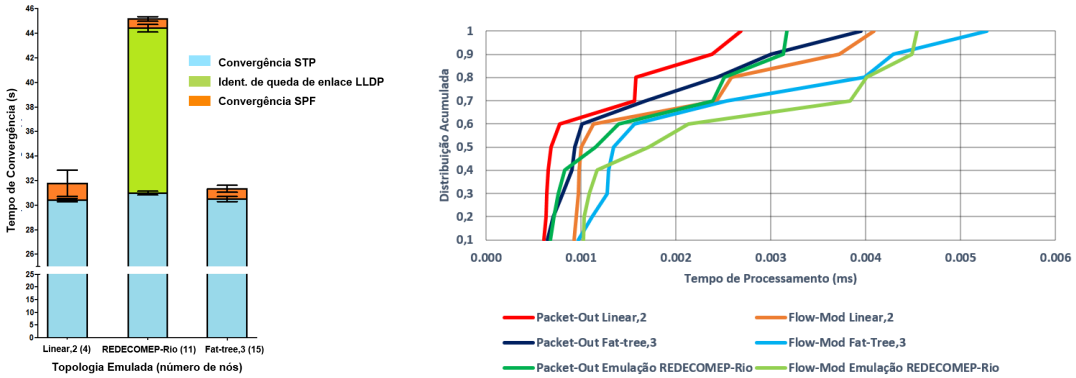
De acordo com o que pode ser observado na Fig. 5, fez-se necessário o estabelecimento de dois enlaces virtuais entre os roteadores PE para o correto funcionamento da aplicação. Além do transporte dos pacotes com *tags* de VLAN 802.1q para o estabelecimento de circuitos, também é necessário o transporte dos pacotes LLDP para o mapeamento da topologia. Desse modo, para cada tipo de enlace virtual foi permitido um determinado tipo de encapsulamento de pacotes. Isto é, para o transporte dos pacotes dos circuitos com encapsulamento dot1q configurou-se uma interface com *encapsulation dot1q any*, em vermelho na Fig. 5, enquanto que para os pacotes de controle LLDP sem encapsulamento foi configurado outro tipo de interface com *encapsulation default*, em laranja na Fig. 5.

Assim como o exposto na Seção III-A, o tempo de convergência da aplicação também foi avaliado, *i.e.*, o intervalo de tempo entre o pacotes ARP *Request* inicial e o que recebe resposta foi determinado. Entretanto, em vez de utilizar a ferramenta FPING foi utilizado o utilitário *ping* disponível por padrão nos *switches*, também com intervalo de 10ms entre ICMP *Echo Request*, e para análise dos resultados a ferramenta *tcpdump*. As requisições foram enviadas a partir do *switch* denominado Catalyst 3560 na Fig. 5.

A Fig.6 exibe o resultado comparativo entre as CDFs dos tempos de convergência do cenário emulado de topologia linear descrito na Seção III-A e do estudo de caso descrito na presente seção. Pode-se observar uma diferença de até 8s entre o menor valor para o cenário emulado e o maior valor do estudo de caso. Essa diferença ocorre, principalmente, devido ao intervalo entre tentativas de associação entre controlador e roteador ASR9000 serem fixadas em 8s, enquanto que o intervalo referente ao OpenVswitch ser de apenas 1s.

IV. Conclusão

Redes de *backbone* têm como uma de suas principais demandas o provimento de serviços de estabelecimento de redes virtuais privadas. Um dos principais tipos de redes são as VPNs de camada 2 ou L2VPNs. A solução proposta de



(a) Tempo total de convergência em relação ao número total de nós na rede. (b) CDF do tempo de processamento das mensagens de *packet_out* e *flow_mod*.

Fig. 4. Resultados obtidos para o cenários emulado.

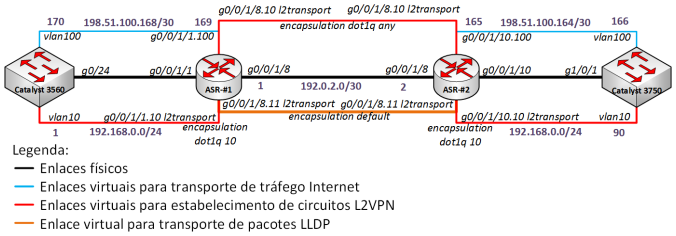


Fig. 5. Cenário de estudo de caso da avaliação do funcionamento da aplicação de estabelecimento de circuitos L2VPN no *backbone* da REDECOMEP-Rio.

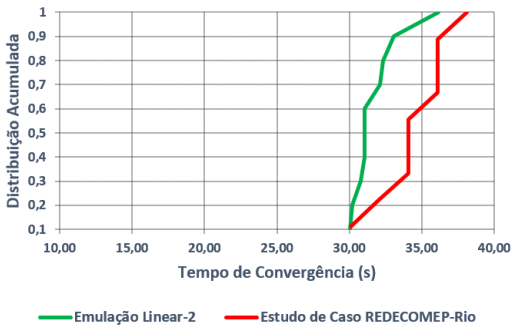


Fig. 6. CDF do tempo total de convergência da aplicação para os cenários emulado linear e de estudo de caso.

estabelecimento desse tipo de circuitos instala regras reativamente nos *switches* da rede utilizando o controlador OpenFlow Ryu. Essa solução se difere, principalmente, das disponíveis no mercado atualmente, pois não é necessário em nenhum momento que o operador tenha conhecimento do funcionamento do protocolo, nem sequer tenha que alterar as configurações dos equipamentos. As avaliações mostram que o sistema escala bem para redes com até quinze nós de *backbone*, dependendo em grande parte do tempo de convergência do protocolo *Spanning Tree*, e se há ou não *loops* na topologia. Esse efeito ocorre devido ao fato de que o protocolo LLDP leva cerca de 13s após a convergência do STP para detectar o bloqueio de um enlace. Além disso, a diferença encontrada entre o cenário emulado e o implantado, mostra que o tempo de associação entre o controlador e o *switch* OpenFlow tem um pequeno impacto no tempo total de convergência da aplicação. Ademais, o impacto dos atrasos entre os *switches*, e entre eles e o controlador no tempo de estabelecimento de circuito é pequeno, devido ao fato dos tempos de processamento de mensagens de *packet_out* e *flow_mod* somados a esses atrasos serem muito pequenos em relação ao tempo total de convergência. Assim, após o período de inicialização da rede, o tempo para estabelecimento automático da L2VPN é inferior a 10 ms, o que é um excelente tempo quando comparado ao processo tradicional para a configuração e estabelecimento desse tipo de circuito.

Portanto, em função dos resultados obtidos, a implementação da aplicação em todo o *backbone* em produção da REDECOMEP-Rio está em fase de finalização, onde o impacto de sua utilização nos usuários da rede deve ser avaliado, posteriormente.

REFERÊNCIAS

[1] V. Zepeda, "Instituicoes cientificas do estado ganham internet de altissima velocidade," *Arquivo de Noticias, FAPERJ*, 05 jun. 2014, Accessed: 2016-03-29. [Online]. Available: http://www.faperj.br/?id=2691.2.2

[2] P. Knight and C. Lewis, "Layer 2 and 3 virtual private networks: taxonomy, technology, and standardization efforts," *Communications Magazine, IEEE*, vol. 42, no. 6, pp. 124–131, 2004.

[3] E. D. Osborne and A. Simha, *Traffic engineering with MPLS*. Cisco Press, 2002.

[4] A. R. Sharafat, S. Das, G. Parulkar, and N. McKeown, "MPLS-TE and MPLS VPNs with OpenFlow," in *ACM SIGCOMM Computer Communication Review*, vol. 41, no. 4. ACM, 2011, pp. 452–453.

[5] *IEEE 802: Local and Metropolitan Area Network Standards*, IEEE Standard 802.1q, 2014. [Online]. Available: http://standards.ieee.org/getieee802/download/802-1Q-2014_mibs.zip

[6] T. J. Misa and P. L. Frana, "An Interview with Edsger W. Dijkstra," *Commun. ACM*, vol. 53, no. 8, pp. 41–47, Aug. 2010. [Online]. Available: http://doi.acm.org/10.1145/1787234.1787249

[7] B. Lantz, B. Heller, and N. McKeown, "A network in a laptop: Rapid prototyping for software-defined networks," in *Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks*, ser. Hotnets-IX. New York, NY, USA: ACM, 2010, pp. 19:1–19:6. [Online]. Available: http://doi.acm.org/10.1145/1868447.1868466

[8] D. Schweikert. (2011) Fping. [Online]. Available: http://fping.org/

[9] F. Fuentes and D. C. Kar, "Ethereal vs. tcpdump: a comparative study on packet sniffing tools for educational purpose," *Journal of Computing Sciences in Colleges*, vol. 20, no. 4, pp. 169–176, 2005.

[10] P. T. RYU. (2014) Ryubook. [Online]. Available: http://osrg.github.io/ryu-book/en/Ryubook.pdf