Uma Avaliação de Desempenho de Segurança Definida por Software através de Cadeias de Funções de Rede*

Igor J. Sanz[1], Igor D. Alvarenga[1], Martin Andreoni[13], Leopoldo A. F. Mauricio[1], Diogo M. F. Mattos[1], Marcelo G. Rubinstein[2] e Otto Carlos M. B. Duarte[1]

[1]Universidade Federal do Rio de Janeiro – GTA/UFRJ/PEE-COPPE – Brasil
[2]Universidade do Estado do Rio de Janeiro – FEN/DETEL/PEL – Brasil
[3]Université Pierre et Marie Curie/Sorbonne Universités – LIP6/CNR – França

Abstract. *Network Function Virtualization enables the provisioning and composition of an on-demand network function chain appropriated tailored to meet requirements of an application or service. This feature is necessary to provide security for enterprise networks and critical infrastructures, which depend on the processing in the core of the network. This paper proposes and develops virtual network security functions and evaluates their performance on the Open Platform for Network Function Virtualization (OPNFV). A prototype using the software defined networking technology and compliant with the Network Service Header (NSH) protocol has been designed. Our prototype shows a smart chaining of an intrusion and prevention detection system with a firewall that presents high flexibility without compromising the end-to-end delay.*

Resumo. *A Virtualização de Funções de Rede (Network Function Virtualization – NFV) possibilita o provisionamento e a composição sob demanda de uma cadeia de função de rede criada sob medida para atender requisitos de uma aplicação ou serviço. Esta funcionalidade é necessária para prover segurança de redes empresariais e infraestruturas críticas que dependem de processamento no núcleo da rede. Este artigo propõe e desenvolve funções virtuais de segurança de rede e avalia seus desempenhos na plataforma aberta para funções virtuais de rede, denominada OPNFV. Um protótipo usando a tecnologia de redes definidas por software e compatível com o protocolo Network Service Header foi projetado e desenvolvido. O protótipo realiza o encadeamento inteligente de um sistema de detecção e prevenção de intrusão com um firewall e apresenta alta flexibilidade sem comprometer o atraso fim-a-fim.*

1. Introdução

A tecnologia de Virtualização de Funções de Rede (*Network Function Virtualization* – NFV) tem como objetivos principais reduzir os gastos com *hardware* e com a implantação de novos serviços, reduzir a complexidade de operação e gerenciamento e melhorar a eficiência e o desempenho da rede através da otimização do consumo de recursos [Pattaranantakul et al. 2016]. Nesse novo modelo as funções de rede, tais como, tradução de endereços de rede (*Network Address Translation* – NAT), *firewall*, sistema de detecção e prevenção de intrusão (*Intrusion Detection and Prevention System* – IDPS)

*Este trabalho foi realizado com recursos do INCT INTERNET DO FUTURO, do CNPQ, da CAPES e da FAPERJ.

e balanceador de carga, deixam de ser executadas em dispositivos de *hardware* dedicados para serem executadas em *softwares* que rodam em computadores de uso geral. Assim, os serviços de rede podem ser controlados de forma centralizada, migrados e implantados dinamicamente na rede para atender sob medida as necessidades de cada aplicação. Devido a essa flexibilidade, é possível definir zonas de segurança, redefinir a direção do tráfego para isolar elementos de rede comprometidos e impedir propagação de *malware* [Pattaranantakul et al. 2016, Reynaud et al. 2016]. O encadeamento de funções de serviço (*Service Function Chaining* – SFC) é o habilitador para o gerenciamento flexível do tráfego de um serviço ou aplicação [Medhat et al. 2017].

A tecnologia de Virtualização de Funções de Rede possui enormes desafios, pois requer uma execução eficiente do encadeamento das funções virtuais de rede (*Virtual Network Functions* – VNFs), além da correta ordenação das funções e a localização da função virtual na máquina física apropriada [Quinn and Nadeau 2015, Medhat et al. 2017]. No que se refere às funções de segurança, uma alta latência ou uma ordem incorreta destas funções pode resultar na não aplicação de uma política adequada sobre os pacotes, ocasionando vulnerabilidades e incidentes de segurança. Portanto, é importante identificar os principais gargalos de desempenho das plataformas de NFV, assim como das ferramentas de encadeamento de funções [Luizelli et al. 2017]. Além disso, a correta definição dos requisitos de recursos é necessária para evitar o sobre ou o subprovisionamento de recursos [Andreoni Lopez et al. 2016].

Este artigo provê o encadeamento de funções de serviço padronizado pelo *Internet Engineering Task Force* (IETF) [Halpern and Pignataro 2015], sobre uma arquitetura de virtualização de funções de rede descrita pelo *European Telecommunications Standards Institute* (ETSI) [ETSI 2014] e implementada através da *Open Platform for Network Function Virtualization* (OPNFV). O protótipo de segurança definida por *software* desenvolvido atende às especificações do protocolo de cabeçalho de serviço de rede (*Network Service Header* – NSH) e se serve da tecnologia de redes definidas por software (SDN) para rapidamente bloquear ataques. Foram desenvolvidos como funções virtuais de segurança de rede um sistema de detecção e prevenção de intrusão baseado em aprendizado de máquina com processamento em nuvem e um *firewall* configurável através de uma interface *RESTFull*. O protótipo implementado pelo encadeamento de funções virtuais de rede usando a OPNFV é pioneiro no Brasil e os resultados da avaliação de desempenho ressaltam importantes aspectos que se mostram os principais gargalos a serem vencidos. O desempenho da versão atual da plataforma mostra que o encadeamento de diversas funções de rede implica um forte decaimento da banda e um aumento linear da latência fim-a-fim, independentemente da topologia de realização física das funções, o que impõe restrições de serviço.

O restante do artigo está organizado da seguinte forma. A Seção 2 apresenta os trabalhos relacionados. A Seção 3 aborda a virtualização de funções de rede e o estado da arte do encadeamento de funções de rede. A Seção 4 descreve o protótipo proposto e as funções virtuais de segurança. A Seção 5 expõe os detalhes da implementação e os resultados da avaliação de desempenho. Por fim, a Seção 6 conclui o artigo.

2. Trabalhos Relacionados

A arquitetura de encadeamento, assim como a implementação das funções de serviço e o encapsulamento da cadeia de funções de serviço (SFC) variam entre diversas propostas. A proposta FlowTags é de um encapsulamento de SFC para que os sistemas intermediários sejam capazes de introduzir rótulos no cabeçalho dos pacotes e passar informações de contexto para os próximos sistemas intermediários em sequência, possibilitando a aplicação de políticas sobre o tráfego [Fayazbakhsh et al. 2013]. Já a proposta StEERING direciona os pacotes entre os sistemas intermediários sem adicionar rótulos [Zhang et al. 2013]. Para isso, a proposta utiliza múltiplas tabelas presentes no protocolo *OpenFlow* 1.0 para criar regras de encaminhamento hierárquicas que, a cada etapa do processamento, adicionam metadados ao tratamento do pacote e definem o próximo destino do encadeamento. A ferramenta ESCAPE se baseia na arquitetura de virtualização de funções de rede padronizada pelo ETSI para o desenvolvimento de protótipos de funções virtuais de rede. A ferramenta utiliza o NETCONF para o gerenciamento das funções e o controlador de rede SDN POX para fazer o encadeamento das funções [Csoma et al. 2014]. Paralelamente, a proposta Cloud4NFV apresenta uma arquitetura de virtualização de funções de redes baseada em quatro planos: infraestrutura, gerenciamento de infraestrutura virtual, orquestração e serviço [Callegati et al. 2015]. A ESCAPE e a Cloud4NFV são as propostas que mais se aproximam do padrão de arquitetura previsto pelo ETSI, mas ainda não empregam a definição de encadeamento de funções de rede do IETF [Halpern and Pignataro 2015].

Quanto ao desempenho do encadeamento de funções, Kulkarni *et al.* argumentam que o encapsulamento SFC com o cabeçalho NSH pode ter seu desempenho aprimorado utilizando uma nova semântica de identificação do SFP (*Service Function Path*) [Kulkarni et al. 2017]. Essa semântica se baseia em utilizar campos do cabeçalho NSH para definir uma sequência lógica entre tipos de VNFs, sem especificidade, e assim o controlador SDN pode decidir dinamicamente o melhor encadeamento disponível entre VNFs do mesmo tipo. A proposta NetBricks, por sua vez, desenvolve um encaminhamento de pacotes em funções virtuais de rede com zero cópia de memória. Para isso, usa-se memória compartilhada com passagens de referência pela área de memória [Panda et al. 2016]. Essa proposta traz consideráveis ganhos de desempenho das funções virtuais em termos de banda e latência. O desempenho do encadeamento de funções virtuais ainda é avaliado analiticamente e considerado nas restrições de problemas de otimização, como o da localização das VNFs na infraestrutura física. Luizelli *et al.* formalizam um modelo para encadeamento de funções usando programação linear inteira e propõem uma heurística que torna o atraso fim-a-fim da SFC comparável ao de sistemas intermediários físicos com uma alocação eficiente de recursos [Luizelli et al. 2015]. Lopez *et al.*, por sua vez, argumentam que a localização das funções virtuais sobre a infraestrutura física segue uma relação de compromisso entre aceitar um maior número de VNFs e o atraso do encadeamento em nós físicos mais distantes [Andreoni Lopez et al. 2016].

Outros trabalhos de avaliação de desempenho existentes na literatura também se relacionam com o encadeamento de funções de rede. Emmerich *et al.* avaliam o desempenho de comutadores virtuais e concluem que otimizações em configurações no núcleo do sistema operacional e o uso de CPU dedicada para a placa de rede são essenciais para aumentar o desempenho dos comutadores virtuais [Emmerich et al. 2014]. Callegati *et al.* avaliam o desempenho da virtualização de rede e dos principais componen-

tes do sistema operacional de nuvem *OpenStack* [Callegati et al. 2014] e Bonafiglia *et al.* comparam o desempenho de diferentes tecnologias de virtualização de funções de rede [Bonafiglia et al. 2015]. Bonafiglia *et al.* consideram configurações com comutadores virtuais e com comutadores com suporte a DPDK (*Data Plane Development Kit*), que transfere as operações de rede em *software* do ambiente virtual diretamente para serem executadas em *hardware* na interface física. Os autores comparam ainda o desempenho do encadeamento de funções virtuais de rede executadas em contêineres versus uma execução com virtualização completa. Os resultados mostram que ao dedicar núcleos de processamento para máquinas virtuais e usar comutadores com suporte a DPDK, o desempenho alcançado por máquinas virtuais é superior ao desempenho dos contêineres. No entanto, o encadeamento efetuado por Bonafiglia *et al.* não segue as normas da ETSI para NFV, nem da IETF para SFC, ou qualquer encapsulamento SFC.

Este artigo, diferentemente de todos os trabalhos citados acima, avalia o desempenho de funções virtuais de segurança encadeadas conforme a arquitetura de encadeamento de funções de rede proposta pelo IETF [Halpern and Pignataro 2015], usando o protocolo de cabeçalho de serviço de rede (NSH) e a plataforma aberta OPNFV. A plataforma OPNFV é compatível com arquitetura de virtualização de rede do ETSI [Rosa et al. 2014] e possui o controlador de redes definidas por software *OpenDayLight*. Portanto, toda função virtual implementada é NSH-consciente, permitindo a passagem de contexto entre funções. Além disso, este artigo avalia o desempenho de um protótipo de segurança definida por *software* composto pelo encadeamento de funções virtuais de segurança.

3. A Virtualização e o Encadeamento de Funções de Rede

A tecnologia de virtualização das funções de rede é a aplicação dos conceitos de computação em nuvem no domínio de redes de operadoras de telecomunicações [Medhat et al. 2017]. Para alcançar desempenho similar aos requisitos de sistemas intermediários em *hardware*, avanços recentes em computação em nuvem, como múltiplos hipervisores, virtualização assistida por *hardware*, sistemas operacionais de nuvem e comutadores por *software* eficientes, têm contribuído na implementação em *software* de funções virtuais de rede. Por sua vez a tecnologia de redes definidas por *software* (SDN) desacopla o plano de controle do plano de dados subjacente e consolida funções de controle em um controlador logicamente centralizado [Mattos and Duarte 2016]. O NFV e o SDN são tecnologias complementares, visto que, o gerenciamento das funções de rede se beneficia do controle SDN logicamente centralizado para configurar o plano de dados e encadear funções de rede coerentes. A Figura 1 mostra como a infraestrutura de virtualização de funções de rede (*NFV Infrastructure* – NFVI), em acordo com a arquitetura de gerência e orquestração das funções virtuais de rede (*Network Function Virtualization MANagement and Orchestration* – NFV-MANO) [ETSI 2014], pode compor microsserviços fim-a-fim sob medida para cada aplicação. A NFVI fornece as abstrações de processamento, armazenamento e acesso à rede às funções virtuais. Além disso, o controle do encaminhamento de pacotes e a consequente abstração da infraestrutura em um grafo de encadeamento de funções virtuais podem ser realizados de forma flexível pelo controle SDN [Han et al. 2015].

Funções de serviço e funções de rede são definidos como sinônimos neste trabalho, e o encadeamento coerente destas funções para cumprir um objetivo é denominado um serviço de rede. A ideia básica do encadeamento de funções de rede é oferecer funcio-

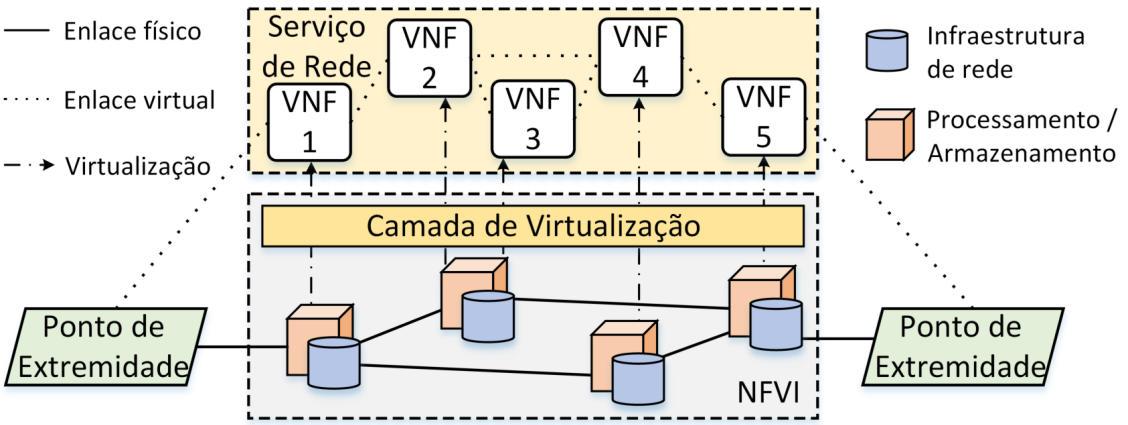


Figura 1. Infraestrutura de virtualização de funções de rede que possibilita a criação de enlaces virtuais para microsserviços fim-a-fim através do encadeamento de funções virtuais de rede (VNF).

nalidades através do encadeamento de funções virtuais que executam funções específicas da camada rede, tais como detecção de intrusão, *firewall*, *proxy* etc. As funções virtuais de rede substituem os inúmeros elementos intermediário de *hardware* que existem hoje (*middleboxes*). A arquitetura básica do encadeamento de funções de rede (*Service Function Chaining* – SFC) conforme a RFC 7665 é apresentada na Figura 2. Nesta arquitetura, o caminho virtual de um serviço, que envolve uma sequência ordenada de VNFs, é definido como caminho de funções de serviço (*Service Function Path* – SFP). Para que o tráfego de um determinado serviço percorra o SFP desejado, um classificador de fluxos é configurado com regras específicas determinando qual SFP cada fluxo deve percorrer. Dessa forma, tráfegos de diferentes microsserviços podem percorrer simultaneamente as mesmas VNFs, mas seguindo caminhos lógicos distintos e isolados em função do identificador de caminho de cada fluxo. As VNFs também podem estar hospedadas em nós físicos diferentes. Por isso, o encaminhador de funções de serviço (*Service Function Forwarder* – SFF) é um elemento necessário em cada nó da NFVI que fornece os enlaces virtuais para suas VNFs hóspedes. Por sua vez, as VNFs podem ser conscientes ou não do encapsulamento da SFC. As VNFs que não são conscientes do encapsulamento podem operar normalmente contanto que sejam precedidas por um elemento que faça o desencapsulamento dos pacotes. Este elemento é chamado de *Proxy* SFC. Na implementação deste artigo, os elementos da arquitetura SFC são construídos baseados em regras através do comutador virtual *Open vSwitch* usando o protocolo *OpenFlow*. Essas regras são gerenciadas pelo controlador SDN *OpenDaylight* aliado a um gerenciador e orquestrador de VNFs denominado *Tacker*.

Para orientar o encaminhamento dinâmico de fluxos em diferentes caminhos virtuais de serviço, como por exemplo para balancear carga ou para conseguir um controle granular e preciso de políticas, é necessário fazer um encapsulamento para marcar os pacotes. A plataforma OPNFV especifica o protocolo de cabeçalho de serviço de rede (*Network Service Header* – NSH) para o encapsulamento da SFC [Quinn and Elzur 2017]. O NSH fornece visibilidade do caminho virtual do serviço de ponta-a-ponta sobre o protocolo de transporte UDP e permite classificar e reclassificar o fluxo em cada função de serviço. Este direcionamento do tráfego é feito a partir de dois campos principais do protocolo. O primeiro, responsável por identificar unicamente cada caminho SFP, é o identificador de

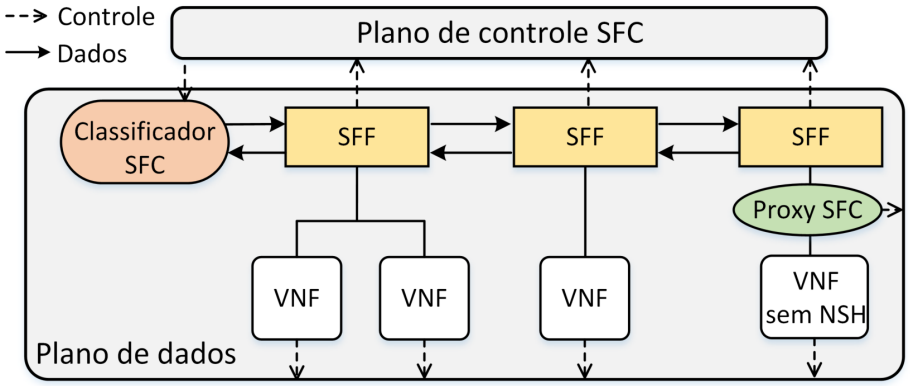


Figura 2. Elementos básicos da arquitetura de Encadeamento de Funções de Rede conforme a RFC 7665.

caminho de serviço (*Service Path Identifier* – SPI), enquanto o segundo, que identifica a posição na cadeia, é o índice de função de serviço (*Service Index* – SI). Este índice é um iterador de 8 bits que decresce uma unidade de 255 a cada salto nas funções de serviço, até atingir um valor referente ao tamanho final da cadeia. Outra vantagem do encadeamento com o NSH é o intercâmbio de metadados entre as VNFs, que é possível devido à existência de campos de contexto. Este aspecto é positivo para os casos de uso de internet móvel ou de provimento de acesso para usuários fixos domésticos em que o contexto pode conter o identificador do assinante ou do inquilino, de forma a permitir políticas baseadas em contratos [Kulkarni et al. 2017].

4. O Protótipo de Segurança Definida por *Software*

Um dos principais usos para a virtualização de funções de rede corresponde à garantia da aplicação correta, coerente e consistente de políticas ao tráfego de rede [Zhang et al. 2013]. No protótipo implementado, as duas principais funções de segurança são o sistema de detecção e prevenção de intrusão e o *firewall*. O primeiro é responsável por analisar os pacotes e gerar alarmes, enquanto o segundo implementa regras de filtragem de pacotes para gerar perímetros de segurança na rede. O encadeamento inteligente destas funções virtuais, doravante denominado como segurança definida por *software*, fornece flexibilidade para instanciar mecanismos complexos de segurança em tempo real e em qualquer ponto da rede. A Figura 3 mostra a arquitetura do protótipo, cuja a ideia central é associar o processamento provido por uma nuvem de processamento por fluxo *Spark* com a flexibilidade de encadeamento das funções de segurança e ainda possibilitar uma ação de resposta em tempo real ao tráfego malicioso detectado. Sua principal vantagem é poder escalar a tarefa de processamento dos fluxos com o espelhamento e o tratamento dos fluxos em nuvem, portanto, de forma independente da infraestrutura de virtualização de funções de rede.

O elemento principal da arquitetura é o sistema de detecção e prevenção de intrusão que utiliza uma tecnologia de processamento por fluxo para fazer a análise do tráfego em tempo real [Andreoni Lopez et al. 2017]. Os pacotes são capturados, através do espelhamento de tráfego, pelo módulo do IDPS que atua diretamente na cadeia. Estes pacotes são abstraídos em fluxos, que são definidos como uma sequência de pacotes com a mesma quíntupla IP de origem, IP de destino, porta de origem, porta de destino e protocolo, durante uma janela de tempo. Ao todo, 46 características de fluxo são extraí-

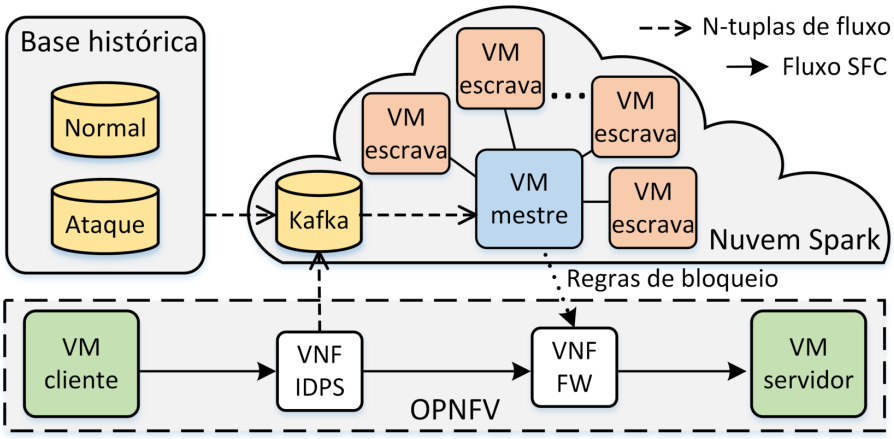


Figura 3. Arquitetura do protótipo de Segurança Definida por *Software* através do encadeamento de um sistema de detecção e prevenção de intrusão e de um *firewall* virtuais.

das e publicadas em um serviço produtor/consumidor de mensagens *Apache Kafka*. Este serviço opera como um sistema de fila e manipulação de fluxo de dados a baixa latência, onde as características da fila são consumidas pelo módulo de classificação. O módulo de classificação, por sua vez, é instanciado em uma nuvem dedicada para a classificação e possui um núcleo principal de processamento, o *Apache Spark*. O *Spark* é implementado em um aglomerado de máquinas no modelo mestre/escravo, onde os escravos possuem a capacidade de expansão e redução de recursos, tornando o sistema escalável.

O IDPS implementado utiliza a arquitetura lambda [Marz and Warren 2013], composta por três camadas: a camada de processamento de fluxo de dados, a camada de processamento em lotes e a camada de serviço. A camada de processamento de fluxo de dados trata o fluxo de dados em tempo real. A camada de processamento em lotes analisa grande quantidade de dados armazenados através de técnicas de computação distribuída como *map-reduce*. Finalmente, a camada de serviço agrega as informações obtidas das duas camadas anteriores para criar as saídas dos dados analisados. O IDPS classifica os fluxos em maliciosos ou benignos através de algoritmos de aprendizado de máquina (*Machine Learning* – ML) baseados em árvores de decisão. Uma base histórica com dados de tráfegos marcados como ataques ou normais é utilizada treinar o algoritmo ML de maneira *offline*. Os parâmetros calculados no processamento *offline* com os dados históricos servem para ajustar o modelo de processamento em tempo real. Assim, o sistema possui uma característica adaptativa, pois os parâmetros podem ser atualizados, se ajustando a novos padrões de uso. Após o treino, o IDPS permite uma classificação do tráfego em tempo real e, ao detectar fluxos maliciosos, é capaz de enviar regras de bloqueio para o *firewall* implementado.

A VNF *firewall* implementada nesse trabalho possui um módulo capaz de encapsular e desencapsular o pacote NSH baseado na aplicação *Python* de código aberto *vxlan_tool*[1]. Esta aplicação é estendida para conter um dicionário *Python* que armazena regras de bloqueio baseadas nas quíntuplas do fluxo. Antes do encaminhamento do pacote para a próxima VNF da cadeia, a quíntupla do pacote é verificada quanto ao conjunto de regras existentes. Uma interface *RESTful* é implementada ainda para possibilitar entradas

[1]Disponível em https://github.com/opendaylight/sfc/blob/master/sfc-test/nsh-tools/vxlan_tool.py.

e remoções de regras no dicionário. Assim, a adição ou remoção de regras de bloqueio no *firewall* é feita por chamadas REST em tempo real pelo IDPS.

5. Implementação e Resultados

Um protótipo de Segurança Definida por *Software* construído através do encadeamento das funções IDPS e *firewall* é implementado conforme a Figura 3 e seu desempenho e eficiência são avaliados para diferentes topologias e configurações.

A plataforma utilizada para implementação dos testes e do protótipo é a plataforma aberta *Open Platform for Network Function Virtualization* (OPNFV) na versão Danube 2.0[2]. Esta plataforma implementa a arquitetura de referência NFV-MANO usando como base o sistema operacional de nuvem *OpenStack*[3] A plataforma OPNFV é construída através do instalador *Fuel* conforme no cenário que fornece a arquitetura de Encadeamento de Funções de Rede referenciada na RFC 7665. Este cenário utiliza ainda o controlador SDN *OpenDaylight*, responsável pelo gerenciamento da camada de enlace, o gerenciador e orquestrador de VNFs *Tacker* e o comutador virtual *Open vSwitch* com suporte ao protocolo NSH. As versões utilizadas desses softwares são ramificações que foram adaptadas para integração à plataforma OPNFV e que estão disponíveis publicamente na versão Danube 2.0 da plataforma. Deve ser ressaltado que essa plataforma ainda se encontra em fase de desenvolvimento, com documentação precária, e muito erros tiveram que ser depurados e corrigidos pela equipe. Este protótipo de cadeia de funções de rede usando NSH é com certeza uma implementação pioneira no Brasil.

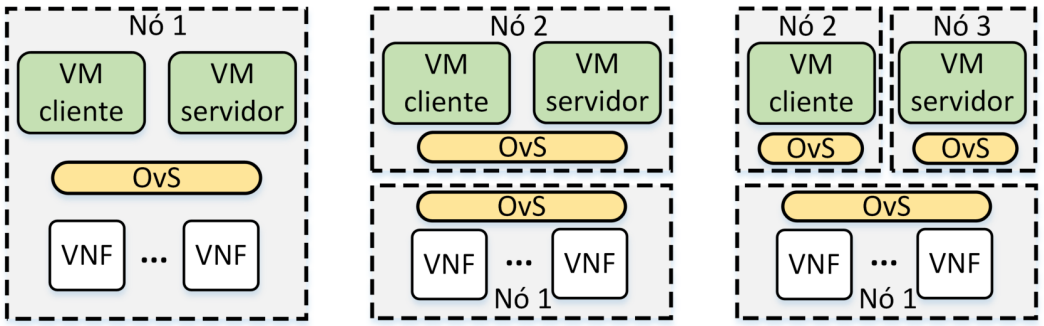
As máquinas usadas para criação do ambiente de testes consistem em um nó controlador Intel 8-Core i7-4770 CPU, 3,40 GHz com 32 GB de memória, e três nós de processamento, Intel Xeon CPU X5570 2,93 GHz com 96 GB de memória (nó 1), Intel 8-Core i7-6700 CPU, 3.40 GHz com 64 GB de memória (nó 2) e Intel 8-Core i7-2600 CPU, 3,40 GHz com 32 GB de memória (nó 3). Todas as máquinas são interligadas através de um comutador topo de *rack* por interfaces de rede de 1 Gb/s que englobam as cinco redes VLANs necessárias para a instalação da nuvem OPNFV: pública, privada, de gerenciamento, de armazenamento e de *Preboot Execution Environment*.

A avaliação de desempenho do encadeamento de funções virtuais de segurança é baseada na arquitetura da RFC 7665 usando o protocolo NSH, como mostrado na Figura 2. Para isso, são analisados diversos casos de uso que impactam o desempenho do encadeamento das funções virtuais de segurança. As medidas estudadas neste trabalho incluem a posição das máquinas virtuais do cliente e do servidor e a cadeia de VNFs instanciados na nuvem, a sobrecarga que cada função de rede introduz na cadeia e a quantidade de núcleos virtuais de processamento demandados pela VNF. Além disso, é avaliado o comportamento na cadeia de uma VNF *firewall* ao adicionar um número alto de regras de bloqueio. Por fim, o protótipo implementado que encadeia a função IDPS e a função *firewall* é avaliado quanto à acurácia de classificação *online* na cadeia em relação a classificação *offline*, fora da cadeia, e à eficiência na detecção em tempo real e no bloqueio automático de fluxos maliciosos.

As topologias escolhidas para avaliar o encadeamento são ilustradas na Figura 4. A primeira topologia, ilustrada na Figura 4(a), utiliza apenas um nó da nuvem, uma má-

[2]Disponível em https://www.opnfv.org.

[3]Disponível em https://www.openstack.org.



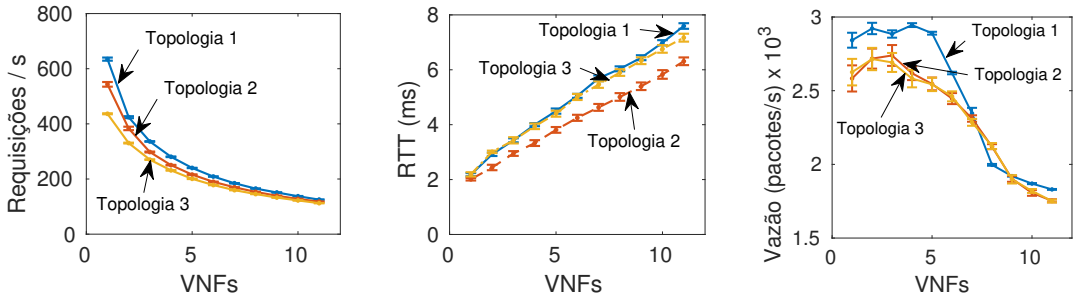
(a) Topologia 1. (b) Topologia 2. (c) Topologia 3.

Figura 4. Topologias utilizadas para avaliar o desempenho do encadeamento de funções virtuais de rede: a) cliente, servidor e cadeia de VNFs no mesmo nó; b) cliente e servidor em um nó separado do nó da cadeia; c) cliente, servidor e cadeia em três nós separados.

quina física, para instanciar a cadeia de funções virtuais e as VMs do pontos de extremidade. Dessa forma, as máquinas virtuais da extremidade da cadeia e a própria cadeia de VNFs competem por recursos de rede do mesmo nó, ao mesmo tempo que não há saltos entre os enlaces físicos dos nós. A Figura 4(b) define a Topologia 2, em que as VMs da extremidade são instanciadas em um mesmo nó, mesma máquina física, e a cadeia de VNFs se encontra em outro nó, uma outra máquina física. A terceira topologia, mostrada na Figura 4(c), a VM servidor, a VM cliente e a cadeia de VNFs se encontram em três nós distintos. Dessa forma, as Topologias 2 e 3 demandam o uso de mais comutadores virtuais, uma vez que cada nó possui um comutador virtual *Open vSwitch* local para controlar seus recursos de rede.

A Figura 5 mostra os resultados de impacto no desempenho obtidos em cada topologia em função do número de VNFs da cadeia, considerando um intervalo de confiança de 95%. Vale ressaltar que as VNFs usadas neste cenário são funções de rede extremamente simples que apenas encaminham os pacotes recebidos para a próxima VNF da cadeia. Este encaminhamento é feito desencapsulando o pacote do protocolo NSH, em que o campo *Network Service Index* do cabeçalho é decrescido de 1 e, assim, o pacote é retornado ao *Open vSwitch* que o envia à próxima função virtual. Assim, o processamento e o atraso é praticamente somente devido ao desencapsulamento e reencapsulamento com o protocolo NSH. A aplicação responsável por estas operações é uma ferramenta escrita em *Python* (*vxlan_tool*) para testes com o cabeçalho NSH. A Figura 5(a) compara as três topologias em relação a taxa de requisições HTTP feitas a partir de uma VM cliente para uma VM servidor que atravessa a cadeia. Nota-se que a Topologia 1 fornece a melhor taxa de requisições HTTP para cadeias curtas de VNFs. No entanto, essa diferença se torna desprezível quando o comprimento da cadeia ultrapassa 8 VNFs. Isso demonstra que, para cadeias curtas, a sobrecarga introduzida por separar o cliente, servidor e as VNFs em diferentes nós é o fator limitante do desempenho. No entanto, cadeias longas introduzem uma sobrecarga que ultrapassa este fator. Já a Figura 5(b) mostra que o tempo de ida e volta em todas as topologias cresce linearmente com o aumento do tamanho da cadeia. A Topologia 2, entretanto, apresentou um aumento de latência significantemente menor que as outras duas. Isto pode ser explicado em função de que na Topologia 2 as VMs cliente e servidor estão no mesmo nó físico, o que diminui o tempo da volta do pacote, e ainda, este nó não compartilha recursos com outras VNFs. Pode-se concluir então que o au-

mento de saltos entre enlace físicos dos nós, assim como o compartilhamento de recursos de um mesmo nó, são fatores que comprometem a latência. Deste modo, a Topologia 2 encontra-se ao meio termo no compromisso desses fatores. O gráfico da Figura 5(c), por sua vez, mostra a vazão máxima medida em pacotes por segundo para cada topologia ao aumentar o tamanho da cadeia. O tamanho de pacote usado nos experimentos deste trabalho é 1334 bytes, que representa o tamanho máximo do pacote *Ethernet* enviado pelo cliente, para que ao ser encapsulado com NSH não haja fragmentação IP. A Topologia 1 apresentou uma melhor vazão em relação às demais para o encadeamento de poucas VNFs por evitar saltos pelos enlaces físicos entre os nós. Percebe-se, no entanto, que o aumento de VNFs em todas as topologias gera competição de recursos do nó que hospeda a cadeia, o que compromete consideravelmente a vazão.



(a) Taxa máx. de req. HTTP em função do comprimento da cadeia.

(b) Tempo de ida e volta em função do comprimento da cadeia.

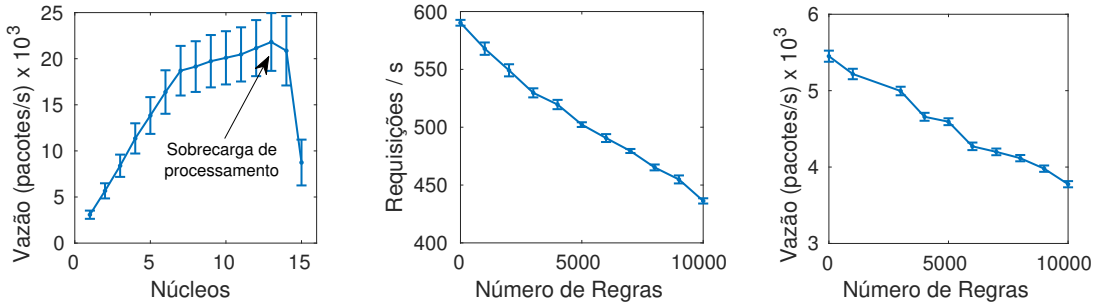
(c) Vazão máxima em função do comprimento da cadeia.

Figura 5. Impacto no desempenho de cadeias de funções de rede ao aumentar o número de VNFs encadeadas avaliado pelas métricas: a) taxa de requisições HTTP suportada; b) tempo de ida e volta; e c) vazão máxima da cadeia.

O maior fator limitante da vazão nestes resultados é a ferramenta *vxlan_tool* que desencapsula os pacotes NSH. Essa aplicação, por padrão, opera de forma sequencial em apenas um núcleo de processamento, mas foi estendida para operar paralelamente em vários núcleos. O efeito desta mudança é observado ao analisar a Figura 6(a), que mostra o aumento da vazão de uma VNF em uma cadeia de tamanho unitário ao alocar-se mais núcleos virtuais dedicados do hipervisor. Dessa forma, a VNF passa a reter mais poder de processamento e consegue efetuar mais operações com pacotes por segundo até atingir o limite de processamento do hipervisor. A plataforma OPNFV, no entanto, já prevê em suas próximas versões que esta função de desencapsulamento seja implementada no núcleo do sistema operacional para ganhos em desempenho.

Para avaliar o desempenho de diferentes funções virtuais de segurança, são implementadas as VNFs *firewall* e IDPS descritas na Seção 4 em cadeias individuais, considerando a Topologia 1. As Figuras 6(b) e 6(c) caracterizam as taxas máximas de requisições HTTP e a vazão máxima da cadeia da VNF *firewall* ao aumentar o número de regras de bloqueio. Observa-se uma redução linear tanto da vazão máxima quanto do número máximo de requisições com o aumento da quantidade de regras impostas. No entanto, essa redução impacta o desempenho em apenas 1% para um uso considerável de 500 regras no *firewall*. O aumento do número de regras também não interferiu significativamente na latência do pacote que atravessa o *firewall*, que manteve-se a uma média próxima de 2 ms.

A segunda função de segurança avaliada é o IDPS, que nesta implementação é



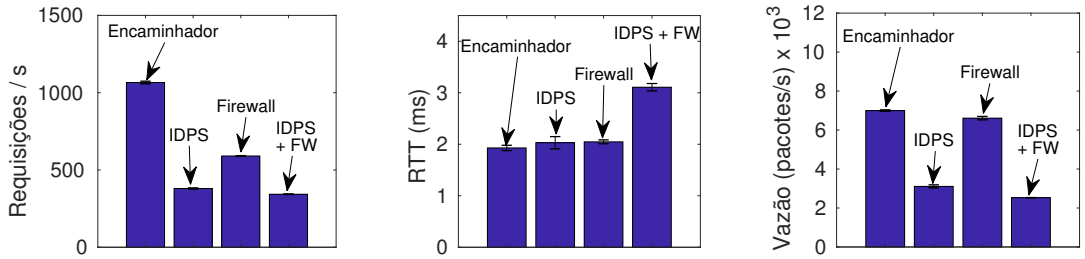
(a) Vazão máxima em função do n^o núcleos virtuais atribuídos.

(b) Taxa máx. de requis. HTTP em função do n^o regras do *FW*.

(c) Vazão máxima em função do n^o de regras do *FW*.

Figura 6. (a) Acréscimo da vazão ao aumentar os núcleos dedicados do hipervisor para a VNF. Impacto no desempenho da (b) taxa máxima de requisições HTTP e da (c) vazão máxima em função do número de regras da VNF *firewall*.

composto por dois módulos descritos na Seção 4. O primeiro módulo atua diretamente na cadeia e realiza duas funções básicas no tráfego, extrair características dos fluxos em janelas de dois segundos e, ao mesmo tempo, encaminhar os pacotes para a próxima VNF. Além disso, a VNF que implementa o IDPS publica as características extraídas no sistema de fila e manipulação de fluxo de dados *Kafka*, para serem lidos pelo segundo módulo de processamento que opera fora da cadeia.



(a) Taxa máxima de requisições HTTP em cada cadeia.

(b) Tempo de ida e volta médio em cada cadeia.

(c) Vazão máxima suportada em cada cadeia.

Figura 7. Impacto no desempenho introduzido por cada função virtual de segurança e pelo encadeamento das duas funções em relação à: (a) taxa máxima de requisições HTTP; (b) tempo de ida e volta do pacote; e (c) vazão máxima.

A Figura 7 mostra o desempenho de cadeias simples com diferentes funções virtuais de rede e ainda do encadeamento composto por duas funções virtuais de segurança. Em todas as cadeias é utilizada a Topologia 1 como referência, fornecendo apenas um núcleo virtual para cada VNF e sem nenhuma regra introduzida no *firewall* na etapa de caracterização. Pode-se observar que o desempenho da VNF que somente encaminha pacotes é superior ao das outras funções virtuais e por isso é usado como patamar inicial de referência. A Figura 7(a) e a Figura 7(c) mostram resultados similares em relação à taxa máxima de requisições HTTP e à vazão suportada de cada cadeia. A VNF *firewall* apresentou melhores resultados que a VNF IDPS nestas duas métricas, por isso, o encadeamento do IDPS com o *firewall* tem seu desempenho limitado pelo IDPS, com uma pequena sobrecarga devido ao encadeamento de duas VNFs. No entanto, a Figura 7(b) mostra que a sobrecarga em latência introduzida por cada função virtual de segurança é bem pequena, mantendo-se a um tempo similar ao patamar inicial. Já o encadeamento

composto pelas duas funções de segurança aumenta em 50% no tempo de ida e volta do pacote, o que já era previsto a partir da Figura 5(b). No entanto, este valor é 1,33 vezes menor do caso em que as funções *firewall* e IDPS atuassem no tráfego separadamente.

Por fim, o protótipo é testado quanto à classificação em tempo real de um conjunto de dados de usuários de uma operadora de telecomunicações brasileira. O algoritmo de aprendizado de máquina implementado é primeiro treinado por uma fração do tráfego que foi rotulada como ataque ou normal pelo IDS Suricata[4]. O restante do tráfego é injetado a partir do cliente em direção ao servidor e atravessa as duas funções virtuais de segurança. A Tabela 1 mostra os resultados da classificação[5] comparando os fluxos que atingiram o servidor com os que foram bloqueados no meio do caminho pelo *firewall*. Os resultados mostram uma acurácia de 72,7% na classificação e que 0,02% dos fluxos maliciosos foram bloqueados antes de serem iniciados no servidor. No entanto, como o *firewall* proposto é reativo, há uma redução real de 15% do volume em bytes do tráfego malicioso total que chega ao servidor. Tal fato é resultado do tempo necessário para análise do fluxo até que seja caracterizado um comportamento malicioso. Fluxos curtos (camundongos) de ataque têm duração menor que a janela de análise e, portanto, não são possíveis de serem bloqueados em tempo real. Isto mostra que o protótipo foi eficaz contra ataques com grande volumes de dados, como os de negação de serviço.

Tabela 1. Matriz de confusão da classificação e bloqueio de fluxos em tempo real. Os fluxos que atingem o servidor são representados como Normal, enquanto os bloqueados no meio do caminho como Ataque.

	VP	**FP**	**VN**	**FN**
Ataque	430	2277	4412795	1658973
Normal	4412795	1658973	430	2277

6. Conclusão

Este trabalho apresentou uma avaliação de desempenho do encadeamento de funções virtuais de segurança conforme as normas da arquitetura de referência NFV-MANO da ETSI, da arquitetura de encadeamento de funções de rede da RFC 7665 e do protocolo *Network Service Header*. O ambiente de testes utilizado para criação das cadeias foi a plataforma de código aberto OPNFV, que usa como base o sistema operacional de nuvens *OpenStack*. Esse artigo propôs ainda duas funções virtuais de segurança em rede: um sistema de detecção e prevenção de intrusão baseado no processamento de dados na nuvem e um *firewall* configurável por uma interface RESTful. Foi avaliado o impacto no desempenho em termos de vazão, tempo de ida e volta e taxa de requisições HTTP causado por diferentes topologias de implementação da cadeia, núcleos virtuais fornecidos às VNFs, configurações das VNFs e sobrecarga acrescentada por cada função virtual. Ressalta-se que tanto os saltos entre enlaces físicos dos nós, quanto a competição de recursos em um mesmo nó, são fatores que impactam o desempenho do encadeamento destas funções. Ainda, a vazão está diretamente condicionada à quantidade de núcleos fornecidos à função virtual, que define a quantidade de pacotes que cada VNF é capaz de processar. A implementação realizada na plataforma OPNFV usando encadeamento de funções de rede (SFC) com o protocolo de cabeçalho de serviço de rede (NSH) é seguramente pioneira no Brasil. A versão atual do protocolo NSH executado pela ferramenta *vxlan_tool*

[4]Disponível em https://suricata-ids.org.

[5]VP: Verdadeiros Positivos; FP: Falsos Positivos; VN: Verdadeiros Negativos; FN: Falsos Negativos.

em *Python* é certamente o principal fator limitante da plataforma. Esta limitação deve ser eliminada nas próximas versões com a sua implementação em *kernel*.

Como trabalhos futuros, pretende-se avaliar o desempenho do encadeamento de funções em novas topologias, como alternar a localização de cada VNF da cadeia e ainda implementar e avaliar novas funções virtuais de segurança, como funções que criptografam e descriptografam o tráfego.

Referências

Andreoni Lopez, M., Lobato, A. G. P., Mattos, D. M. F., Alvarenga, I. D., Duarte, O. C. M. B., and Pujolle, G. (2017). Um Algoritmo Não Supervisionado e Rápido para Seleção de Características em Classificação de Tráfego. In *SBRC'2017*, Belém/PA.

Andreoni Lopez, M., Mattos, D. M. F., and Duarte, O. C. M. B. (2016). Evaluating allocation heuristics for an efficient virtual network function chaining. In *7th International Conference on the Network of the Future (NoF)*, pages 1–5.

Bonafiglia, R., Cerrato, I., Ciaccia, F., Nemirovsky, M., and Risso, F. (2015). Assessing the performance of virtualization technologies for NFV: A preliminary benchmarking. In *2015 Fourth European Workshop on Software Defined Networks*, pages 67–72.

Callegati, F., Cerroni, W., Contoli, C., and Santandrea, G. (2014). Performance of network virtualization in cloud computing infrastructures: The openstack case. In *IEEE 3rd International Conference on Cloud Networking (CloudNet)*, pages 132–137.

Callegati, F., Cerroni, W., Contoli, C., and Santandrea, G. (2015). Dynamic chaining of virtual network functions in cloud-based edge networks. In *1st IEEE Conference on Network Softwarization (NetSoft)*, pages 1–5.

Csoma, A., Sonkoly, B., Csikor, L., Németh, F., Gulyas, A., Tavernier, W., and Sahhaf, S. (2014). ESCAPE: Extensible service chain prototyping environment using Mininet, Click, NETCONF and POX. In *ACM Conference on SIGCOMM*, SIGCOMM '14, pages 125–126, New York, NY, USA. ACM.

Emmerich, P., Raumer, D., Wohlfart, F., and Carle, G. (2014). Performance characteristics of virtual switching. In *IEEE 3rd International Conference on Cloud Networking (CloudNet)*, pages 120–125.

ETSI (2014). ETSI GS NFV-MAN 001: Network functions virtualisation; management and orchestration. Technical report.

Fayazbakhsh, S. K., Sekar, V., Yu, M., and Mogul, J. C. (2013). Flowtags: Enforcing network-wide policies in the presence of dynamic middlebox actions. In *II ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking*, HotSDN '13, pages 19–24, New York, NY, USA. ACM.

Halpern, J. and Pignataro, C. (2015). Service Function Chaining (SFC) architecture. RFC 7665, RFC Editor. `http://www.rfc-editor.org/rfc/rfc7665.txt`.

Han, B., Gopalakrishnan, V., Ji, L., and Lee, S. (2015). Network function virtualization: Challenges and opportunities for innovations. *IEEE Communications Magazine*, 53(2):90–97.

Kulkarni, S., Arumaithurai, M., Ramakrishnan, K. K., and Fu, X. (2017). Neo-NSH: Towards scalable and efficient dynamic service function chaining of elastic network functions. In *20th Conference on Innovations in Clouds, Internet and Networks (ICIN)*, pages 308–312.

Luizelli, M. C., Bays, L. R., Buriol, L. S., Barcellos, M. P., and Gaspary, L. P. (2015). Piecing together the NFV provisioning puzzle: Efficient placement and chaining of virtual network functions. In *IFIP/IEEE International Symposium on Integrated Network Management (IM)*, pages 98–106.

Luizelli, M. C., Raz, D., Sa'ar, Y., and Yallouz, J. (2017). The actual cost of software switching for nfv chaining. In *2017 IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*, pages 335–343.

Marz, N. and Warren, J. (2013). *Big Data: Principles and Best Practices of Scalable Realtime Data Systems*. Manning Publications Co., Greenwich, CT, USA, 1st edition.

Mattos, D. M. F. and Duarte, O. C. M. B. (2016). Authflow: authentication and access control mechanism for software defined networking. *Annals of Telecommunications*, 71(11):607–615.

Medhat, A. M., Taleb, T., Elmangoush, A., Carella, G. A., Covaci, S., and Magedanz, T. (2017). Service function chaining in next generation networks: State of the art and research challenges. *IEEE Communications Magazine*, 55(2):216–223.

Panda, A., Han, S., Jang, K., Walls, M., Ratnasamy, S., and Shenker, S. (2016). Netbricks: Taking the v out of NFV. In *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI'16)*, pages 203–216. USENIX Association.

Pattaranantakul, M., He, R., Meddahi, A., and Zhang, Z. (2016). SecMANO: Towards network functions virtualization (nfv) based security management and orchestration. In *IEEE Trustcom/BigDataSE/ISPA*, pages 598–605.

Quinn, P. and Elzur, U. (2017). Network service header. Internet-Draft draft-ietf-sfc-nsh-12, IETF Secretariat. `http://www.ietf.org/internet-drafts/draft-ietf-sfc-nsh-12.txt`.

Quinn, P. and Nadeau, T. (2015). Problem statement for service function chaining. RFC 7498, RFC Editor. `http://www.rfc-editor.org/rfc/rfc7498.txt`.

Reynaud, F., Aguessy, F. X., Bettan, O., Bouet, M., and Conan, V. (2016). Attacks against network functions virtualization and software-defined networking: State-of-the-art. In *IEEE NetSoft Conference and Workshops (NetSoft)*, pages 471–476.

Rosa, R., Siqueira, M., Barea, E., Marcondes, C., and Rothenberg, C. (2014). Network function virtualization: Perspectivas, realidades e desafios. In *Minicursos do XXX Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos - SBRC 2014*.

Zhang, Y., Beheshti, N., Beliveau, L., Lefebvre, G., Manghirmalani, R., Mishra, R., Patneyt, R., Shirazipour, M., Subrahmaniam, R., Truchan, C., and Tatipamula, M. (2013). Steering: A software-defined networking for inline service chaining. In *2013 21st IEEE International Conference on Network Protocols (ICNP)*, pages 1–10.

