Capítulo 3

Virtualização de Funções de Rede na IoT: Um Panorama do Gerenciamento de Desempenho x Segurança

Guilherme Werneck de Oliveira, Jonathan Rangel Porto, Nelson Gonçalves Prates Jr., Aldri Luiz dos Santos, Michele Nogueira, Daniel Macêdo Batista

Abstract

This chapter presents open issues and the state-of-the-art related to the use of Network Function Virtualization (NFV) in the detection and mitigation of security threats in the Internet of Things (IoT). The main characteristics of NFV, the project, architectures, and protocols of networks to connect IoT devices are also discussed as basis to understand the main concepts. It focuses also on the NFV management specifications and how they affect performance and security in IoT. Next, the chapter presents the issues related to the performance of NFV applied to security in IoT. Then, it highlights a case study on the MCTIC/FAPESP MENTORED Testbed, an experimental environment being deployed for the community that provides realistic scenarios for simulating specific vulnerabilities and for evaluating network security mechanisms in a controlled manner. Finally, the chapter discusses the main open challenges in order to increase the interest of readers to conduct research on topics that bring together virtualization, IoT and network security.

Resumo

Este capítulo apresenta questões e o estado da arte relacionados ao uso da Virtualização de Funções de Rede (NFV – Network Function Virtualization) na detecção e mitigação de ameaças de segurança na Internet das Coisas (IoT – Internet of Things). Ele aborda as características relacionadas à NFV, assim como o projeto, as arquiteturas e os protocolos de redes que suportam a conexão dos dispositivos IoT. Consideram-se também as especificações de gerenciamento de NFV e como elas afetam os requisitos de desempenho e de segurança na IoT. Na sequência, são apresentadas as questões relacionadas ao desempenho de NFV quando esta é aplicada à garantia de segurança em IoT. Então, detalha-se um estudo de caso implementado no ambiente experimental MCTIC/FAPESP MENTORED, um ambiente em implantação que será aberto à comunidade e oferecerá cenários realísticos para simulação de vulnerabilidades específicas e avaliação de mecanismos de segurança de redes de forma controlada. Por fim, discutem-se os principais desafios em aberto nas pesquisas que unam virtualização, IoT e segurança na IoT.

3.1. Introdução

A Internet das Coisas (IoT – *Internet of Things*) tem passado por uma rápida popularização, alcançando uma grande diversidade de domínios de aplicações, como por exemplo cuidados da saúde, monitoramento ambiental, automação residencial, mobilidade inteligente e Indústria 4.0 [da Silva et al. 2021, Batista et al. 2016, Rosário et al. 2014]. Como consequência, cada vez mais dispositivos IoT[1] com características diversas são implantados em uma variedade de ambientes públicos e privados, tornando-se progressivamente objetos comuns da vida cotidiana. A IoT tem sido vista por muitos como o próximo estágio da Internet. Sua implementação possibilita a evolução na sociedade e indústria, como no caso das cidades inteligentes, pois com a IoT os sistemas de transporte, de controle de resíduos, de energia, entre outros, tornam-se mais eficientes e melhoram a qualidade de vida dos cidadãos. Por outro lado, a infraestrutura física de sistemas heterogêneos é complexa e exige soluções eficientes e dinâmicas para o gerenciamento e a configuração das redes num nível que permita a implantação padronizada e de fácil replicação em casas, prédios e cidades inteligentes [Alam et al. 2020, Chi et al. 2019].

A Figura 3.1 ilustra uma visão geral da IoT. Os dispositivos IoT, na camada mais baixa da figura, interagem com o ambiente físico nos papéis de sensores e atuadores. Esses dispositivos são necessários para o correto funcionamento das aplicações IoT, na camada mais alta da figura. As comunicações entre os vários componentes do cenário acontecem graças à infraestrutura de tecnologia da informação disponível na Internet por meio de recursos na borda e no núcleo da rede. Como ocorre em redes de telecomunicações de um modo geral, dois dos requisitos importantes em IoT são desempenho e segurança. No entanto, muitos dispositivos na IoT não oferecem suporte a mecanismos de segurança fortes e, portanto, podem ser alvos e, até mesmo, meios para uma série de ataques [Meneghello et al. 2019]. De acordo com a empresa Kaspersky [Kaspersky 2020, Kaspersky 2019], os ataques a dispositivos IoT têm se tornado frequentes. Nos primeiros seis meses de 2019, foram contabilizados 105 milhões de ataques a dispositivos IoT oriundos de 276 mil endereços IP exclusivos. Esses ataques são dos mais variados tipos e se aproveitam de diversas vulnerabilidades, como a descoberta em agosto de 2020 pela empresa Check Point [CheckPoint 2020] contra a assistente pessoal Alexa, que permitia a manipulação de seus *tokens* e a realização de ações em nome da vítima. Outro dado importante é que 28% das empresas que utilizam plataformas IoT reportam ter encontrado incidentes envolvendo dispositivos conectados. Com o grande volume de dados gerados por sensores e dispositivos inteligentes, inclusive dados sigilosos, as consequências de tais incidentes podem ser graves.

Uma abordagem que vem ganhando espaço quando o intuito é responder às ameaças na IoT consiste no uso de recursos virtualizados de rede por meio da Virtualização de Funções de Rede (NFV – *Network Functions Virtualization*) [Alam et al. 2020, Farris et al. 2019, Zarca et al. 2018]. A NFV apresenta um novo grau de flexibilidade e escalabilidade criando recursos de rede virtuais sob demanda, como *firewalls*, sistemas de detecção de intrusão (IDS – *Intrusion Detection Systems*) e sistemas de inspeção profunda

[1]No decorrer deste capítulo, o termo "dispositivos IoT" será usado para descrever dispositivos de hardware voltados para aplicações de Internet das Coisas, como assistentes pessoais, relógios inteligentes, sensores e atuadores de um modo geral.

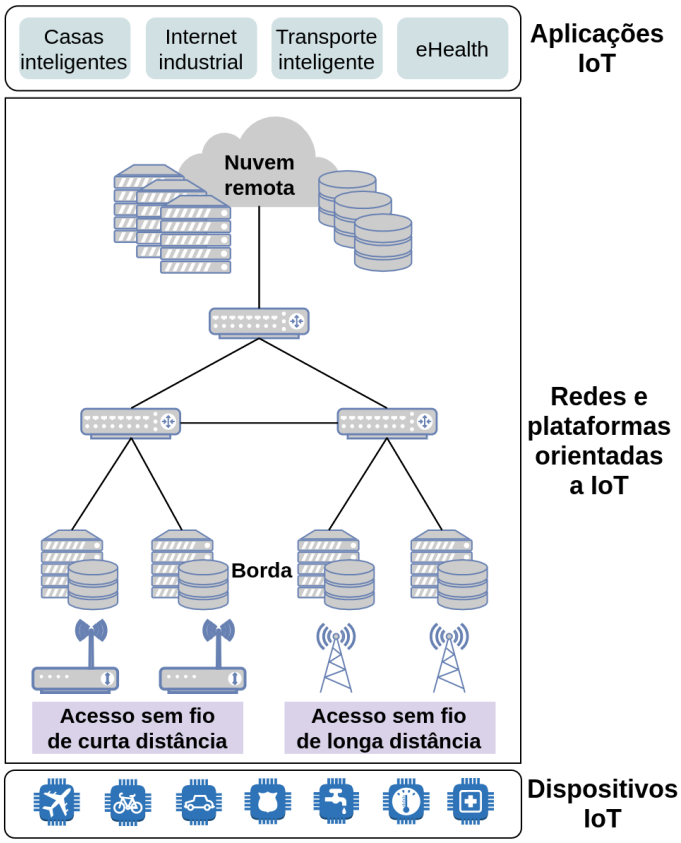


Figura 3.1. Uma visão geral da IoT [Farris et al. 2019].

de pacotes (DPI – *Deep Packet Inspection*). A abordagem de virtualização permite que várias instâncias de um mecanismo de detecção ou mitigação específico possam ser implementadas em diferentes locais na rede, considerando as restrições impostas pela ocorrência de eventos maliciosos [Farris et al. 2019] e pelo poder limitado de processamento e armazenamento de muitos dispositivos IoT.

Além da localização das funções virtualizadas, limitada pela capacidade dos dispositivos IoT e pelas particularidades dos ataques, há outras questões relacionadas ao uso de NFV. Uma delas diz respeito ao gerenciamento do desempenho dos recursos virtualizados. Para [Laghrissi and Taleb 2019], nesse cenário, o verdadeiro desafio é produzir *software* eficiente e escalável para orquestrar as redes virtuais, a fim de permitir que essas redes sejam facilmente configuradas e tenham seus ciclos de vida gerenciados. Em [Mijumbi et al. 2016], além da gestão e da orquestração de recursos, os autores também consideram a eficiência energética, a própria eficiência do recurso provisionado, a modelagem e a alocação de recursos como desafios em aberto. Tudo isso se relaciona com o gerenciamento de desempenho em ambientes virtualizados, tópico que tem levantado diversos desafios de pesquisa, como orquestração de funções de rede virtualizadas (VNF – *Virtual Network Function*), otimização da cadeia de funções (SFC – *Service Function Chaining*), posicionamento de VNFs em localidades distintas, entre outros [Zarca et al. 2020a, Gupta et al. 2019, Zhang et al. 2016].

Este capítulo apresenta um apanhado do estado da arte relacionado ao desempenho de NFV na detecção e mitigação de ameaças na IoT, enfatizando as questões de pesquisa em aberto. O foco na utilização da tecnologia NFV busca não só considerar aspectos ligados à redução de custos, devido à sua utilização sob demanda, mas também à redução de forma eficaz dos riscos e dos prejuízos causados por ataques na IoT. Além disso, por se tratar de um tópico recente, espera-se que sua divulgação por meio do capítulo tenha um papel importante no avanço do estado da arte. O capítulo apresenta características relacionadas à NFV, bem como à construção, arquiteturas e protocolos de redes que possuam dispositivos IoT. O texto também considera as especificações de gerenciamento de NFV, assim como requisitos de desempenho e segurança na IoT. Em seguida, são apresentadas questões de pesquisa relacionadas ao desempenho de NFV quando esta é aplicada à detecção e à mitigação de ameaças na IoT. Na sequência, será apresentado um estudo de caso sobre o MENTORED *Testbed*, ferramenta aberta à comunidade que disponibiliza cenários realísticos para simulação de vulnerabilidades específicas e avaliação de mecanismos de segurança de redes de forma controlada. Também são apresentadas uma análise e uma discussão sobre os principais desafios em aberto na área.

Este capítulo está organizado da seguinte forma: a Seção 3.2 apresenta a definição e as características da tecnologia NFV, descreve aspectos relacionados à camada física nos dispositivos IoT, redes e suas aplicações, e detalha as características da camada de orquestração de NFV; ainda, a Seção 3.2 relaciona as questões de desempenho de NFV às camadas física, virtualização, serviços e orquestração, aplicadas à detecção e à mitigação de ameaças na IoT; a Seção 3.3 apresenta a revisão da literatura sobre desempenho da NFV quando empregada para a detecção e mitigação de ameaças na IoT; a Seção 3.4 descreve um estudo de caso preparado sobre o MENTORED *Testbed*, ambiente de experimentação aberto à comunidade que disponibiliza cenários realísticos para simulação de vulnerabilidades específicas e para avaliação de mecanismos de segurança de redes de forma controlada. Por fim, o capítulo traz uma análise e uma discussão na Seção 3.5 sobre os principais desafios em aberto na área.

3.2. Conceitos Básicos

A Figura 3.2 ilustra um cenário de virtualização de funções de rede na IoT. Os dispositivos reais, na camada **física**, utilizam os serviços das VNFs disponíveis na camada de **serviços**. As VNFs são instanciadas graças à infraestrutura NFV, na camada de **virtualização**, e são gerenciadas pelo gerenciador de VNF e pelo gerenciador de infraestrutura virtualizada na camada de **orquestração**. Em um cenário ideal, algumas das VNFs fornecem serviços de segurança e nenhuma das VNFs possui vulnerabilidades. O foco deste capítulo é na utilização de VNFs que implementem mecanismos de segurança da rede, como *firewall*, IDS e DPI. Às leitoras e aos leitores com interesse no tópico de detecção de vulnerabilidades nas implementações de VNFs, recomenda-se a leitura de [De Benedictis and Lioy 2019, Marku et al. 2020].

As subseções seguintes apresentam os conceitos básicos para que os componentes apresentados na Figura 3.2 sejam compreendidos: a Subseção 3.2.1 apresenta a definição e as características de NFVs, relacionando-as com as camadas de virtualização, serviços e orquestração. A Subseção 3.2.2 define a IoT e apresenta aplicações nesse contexto, relacionando com a camada física. A Subseção 3.2.3 descreve conceitos presentes no ge-

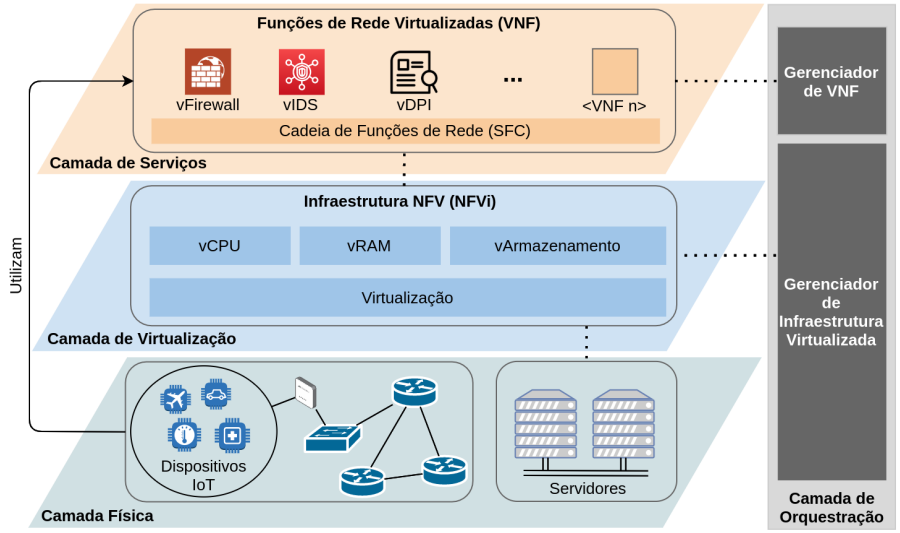


Figura 3.2. Visão geral da virtualização de funções de rede na IoT.

renciamento de redes que utilizam NFV, relacionando-os com a camada de orquestração; e a Subseção 3.2.4 apresenta conceitos que ajudam a compreender o *tradeoff* entre desempenho e segurança quando utiliza-se NFV para detecção e mitigação de ameaças de segurança na IoT, relacionando as camadas **física**, de **virtualização**, de **serviços** e de **orquestração**.

3.2.1. Virtualização de Funções de Rede (NFV)

As funções de rede como filtragem de pacotes, tradução de endereços e balanceamento de carga são essenciais em muitas redes de computadores. Mesmo em uma rede residencial de pequeno porte, em que os usuários locais agem como clientes, acessando servidores externos localizados na Internet, os modems fornecidos pelo provedor de Internet costumam já vir com algumas destas funções embutidas de fábrica. Em uma rede de grande porte, como as existentes em *campi* universitários, as funções de rede costumam ser implementadas em hardware dedicado e são configuradas de modo a formar um encadeamento de funções de rede ou cadeias de funções (SFCs). Um exemplo de SFC aplicada a um fluxo de pacotes entrando em uma rede para acesso a um serviço web está ilustrado pelos dispositivos interconectados da Figura 3.3. As funções de rede realizadas pelos dispositivos na figura são:

1. **Filtragem de pacotes:** para verificar se as características do fluxo (endereço IP de origem, endereço IP de destino, porta origem, protocolos) coincidem com alguma regra que represente um potencial ataque. Caso represente, os pacotes são descartados. Caso não represente, eles seguem para a próxima função de rede;

2. **Tradução de endereços:** para repassar os pacotes da requisição para uma rede interna com endereços IP não roteáveis na Internet, onde o serviço *web* será fornecido por um conjunto de servidores;

3. **Balanceamento de carga:** para distribuir as requisições entre o conjunto de servidores *web*, reduzindo a chance de sucesso de um ataque de negação de serviço.

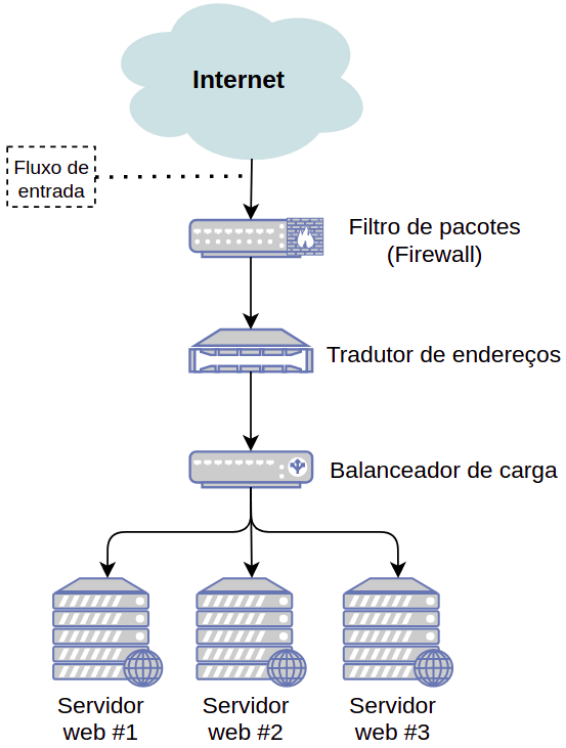


Figura 3.3. Exemplo de um encadeamento de funções em uma rede tradicional.

Uma rede como a ilustrada na Figura 3.3, em que as funções de rede são fornecidas por equipamentos de hardware dedicados conhecidos como *appliances* ou *middleboxes*, será chamada no decorrer deste capítulo de uma **rede tradicional**. Nessa rede, se o operador tiver que adicionar uma nova função, como um sistema de detecção de intrusão (IDS), por exemplo, depois do *firewall* (filtro de pacotes), ele precisará adquirir o *appliance* específico e fazer sua instalação física e configuração na rede local, considerando possíveis mudanças em rotas nos *appliances* existentes.

A NFV permite a criação de uma arquitetura de redes que flexibiliza o processo de aquisição e reconfiguração de funções de rede por meio da virtualização destas funções. Diferente de uma rede tradicional, em uma rede baseada em NFV, as VNFs são fornecidas como *software* e instanciadas em hardware de propósito geral. Neste caso, o operador poderia contratar um serviço de IDS, que estaria implementado completamente em *software* e instanciar o mesmo em um servidor localizado no *data center* de um provedor de nuvem ou em computadores na névoa ou na borda. Possíveis mudanças nas configurações de *appliances* existentes poderiam ser necessárias, assim como no caso da rede tradicional, mas todas as ações relacionadas à aquisição do equipamento específico e sua instalação física na rede local passam a ser desnecessárias. Neste caso da rede baseada em NFV, caso o operador resolva passar a utilizar outra implementação de IDS, bastaria alterar o *software* instanciado na nuvem/névoa/borda, sem necessidade de aquisição de um novo hardware. Além disso, em momentos de pico, em que o IDS precisasse realizar

tarefas intensivas em memória e CPU, bastaria solicitar ao provedor que ampliasse esses recursos ou migrasse a VNF para um recurso com maior capacidade. Essas facilidades nas mudanças da implementação das VNFs e da capacidade das mesmas tornam a NFV ideal para ambientes heterogêneos, com constantes mudanças e com recursos limitados, como é o caso da IoT [Alam et al. 2020].

A Figura 3.4 ilustra uma possível implementação da SFC da Figura 3.3 em uma arquitetura de NFV. Neste caso, o fluxo de pacotes alcança a primeira função de rede, o *firewall*, que está instanciado na nuvem, na névoa ou na borda. O fluxo percorre as demais funções, também instanciadas na nuvem/névoa/borda, até alcançar os servidores web. Vale notar que neste exemplo, o filtro de pacotes e o tradutor de endereços estão compartilhando uma mesma máquina física, mas essa configuração não precisa ser estática. Em um momento em que o tráfego na rede esteja relativamente alto, as instanciações podem ser feitas em máquinas separadas para que cada uma possa tirar proveito de todos os recursos físicos de cada um dos servidores. O compartilhamento de um mesmo servidor físico seria interessante em momentos de tráfego baixo, permitindo que menos servidores permaneçam ligados, levando a uma economia em termos de energia elétrica. Além disso, também não é necessário que todas as funções estejam virtualizadas. Seria possível, por exemplo, instanciar apenas o *firewall*.

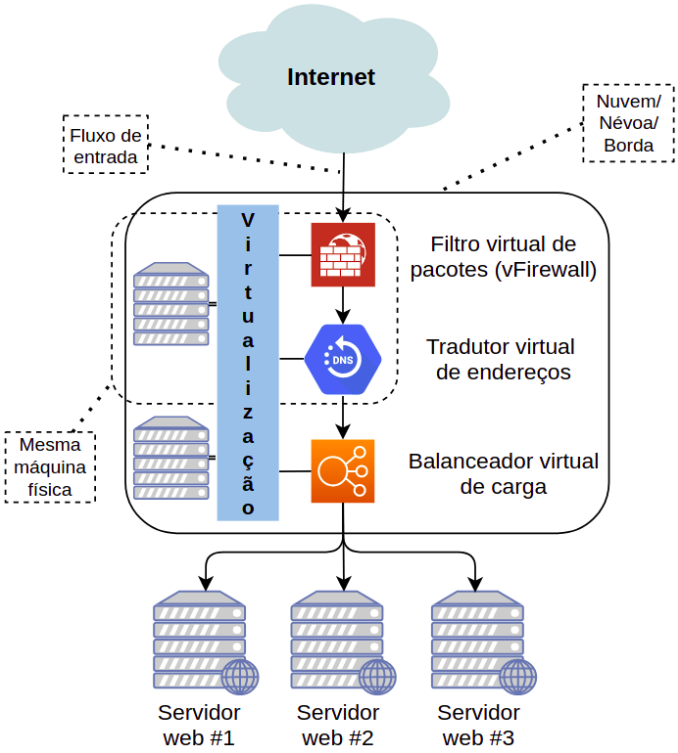


Figura 3.4. Exemplo de um encadeamento de funções de rede com NFV.

Para que a NFV funcione corretamente é necessário que haja a interação de diversos componentes, que podem ser agrupados em camadas, como apresentado na Figura 3.2. As VNFs precisam ser implementadas como *software* (contêineres ou máquinas virtuais, por exemplo) e mantidas em algum repositório, compondo a camada de servi-

ços. A camada de serviços também pode fornecer SFCs inteiras. Sempre que uma VNF é requisitada, ela deve ser instanciada em alguma máquina física. A infraestrutura da NFV (NFVi), na camada de virtualização, é responsável por abstrair os recursos da infraestrutura física, que possivelmente estarão em um ou mais provedores, permitindo que equipamentos de diferentes fabricantes possam oferecer os recursos físicos necessários para a instanciação das VNFs. A NFVi também cuida da interconexão das VNFs entre si e com o mundo exterior, permitindo a interação com a Internet. Para garantir que os recursos da infraestrutura sejam alocados da melhor forma possível, é necessário gerenciar o ciclo de vida de cada uma das VNFs. O gerenciador de VNF na camada de orquestração tem a responsabilidade de cuidar desse ciclo de vida, instanciando, atualizando e encerrando as VNFs. Ainda na camada de orquestração, o gerenciador de infraestrutura virtualizada tem o papel de manter um controle do mapeamento dos recursos físicos para os recursos virtuais.

3.2.2. Redes IoT e Aplicações

A IoT é um paradigma de comunicação com o objetivo de conectar diversos tipos de objetos à Internet [Meneghello et al. 2019]. Dentre esses objetos há, por exemplo, relógios, veículos, prédios, sensores e atuadores de um modo geral. Adicionar suporte à IoT em um objeto consiste na inclusão de recursos como bateria, memória, processador e interface de rede sem fio. Estes recursos são utilizados para execução de protocolos e aplicações da IoT. Com esse suporte, é comum que os objetos passem a ser chamados de "inteligentes". Este adjetivo também é usado para descrever aplicações executadas nos objetos. No cenário da Figura 3.2, os objetos inteligentes estão localizados na camada física.

A IoT não é uma rede à parte da Internet, mas sim uma extensão da mesma que permite a conexão de diversos tipos de objetos do cotidiano. Diferente de um computador convencional, seja ele um computador pessoal ou um servidor, é comum que um dispositivo IoT seja exposto a ambientes críticos para equipamentos eletroeletrônicos, com altas temperaturas e umidade, por exemplo. A depender dos ambientes e das aplicações que serão executadas nos objetos, eles têm restrições nas dimensões físicas. As limitações impostas pelo ambiente e pela dimensão física impedem que dispositivos IoT tenham um alto poder computacional, exigindo que seja dada preferência pela execução de protocolos e aplicações leves, que garantam a qualidade de experiência dos usuários sem que haja um consumo elevado dos recursos do dispositivo, principalmente da bateria.

A Figura 3.5 ilustra a típica arquitetura em três camadas da IoT [Lin et al. 2017]. A **camada de percepção** está na parte inferior da arquitetura. Ela é responsável por interagir com os dispositivos e componentes físicos dos objetos inteligentes, obtendo dados e enviando comandos. A **camada de rede** integra diversos dispositivos de interconexão, como *switches* e pontos de acesso, a várias tecnologias de comunicação, como Bluetooth e Wi-Fi. Nesta camada são determinadas as rotas das comunicações. A **camada de aplicação** é a camada no topo da arquitetura e é onde as aplicações são propriamente implementadas. Aplicativos de cidades inteligentes, indústrias inteligentes e casas inteligentes estão nesta camada. O ideal é que estes aplicativos utilizem protocolos leves da camada de aplicação da Arquitetura Internet, como o MQTT, o CoAP e o AMQP [Naik 2017].

A limitação de recursos dos dispositivos conectados à IoT, unida às configurações

Figura 3.5. Arquitetura em camadas da IoT

padrão dos dispositivos, que não levam em conta alguns aspectos básicos de segurança como criptografia e a obrigatoriedade de senhas fortes [Giaretta et al. 2019], têm tornado a IoT alvo de ataques de segurança. Parte desses ataques é realizada para obter o controle remoto dos dispositivos, que passam a fazer parte das chamadas *botnets*, redes criadas para disparar ataques de larga escala contra a Internet [Schwengber et al. 2020]. No decorrer deste capítulo, o termo Redes IoT será usado para se referir a redes de comunicação que possuam dispositivos IoT.

3.2.3. Gerenciamento de Redes com NFV

A ausência de dispositivos com recursos abundantes para a execução de funções de rede relacionadas à segurança tornam a arquitetura de NFV interessante para prover segurança na IoT. Neste caso, as VNFs de segurança podem ser instanciadas em dispositivos externos, fornecidos por provedores de nuvem, de névoa ou na borda. Para que isso seja possível, os atores da camada de orquestração (Figura 3.2) têm papel fundamental [Medhat et al. 2017].

O gerenciador de infraestrutura virtualizada está presente em cada um dos domínios de infraestrutura virtual. No caso de uma infraestrutura mantida pela plataforma OpenStack[2], por exemplo, o gerenciador utilizado é o `neutron`. Um gerenciador de infraestrutura virtualizada como o `neutron` é responsável por cuidar do endereçamento entre os dispositivos virtualizados, da topologia virtual, e da potencial conexão entre diversas infraestrutura por meio de redes privadas virtuais (VPNs - *Virtual Private Networks*), entregando "rede como um serviço".

O gerenciador de VNF é responsável por gerenciar o ciclo de vida das VNFs, como instanciação, atualização, mudança de configuração, escala vertical (mudança na quantidade de recursos físicos disponibilizados por um computador hospedeiro para uma NF), escala horizontal (mudança na quantidade de computadores hospedeiros usados para instanciar uma NF) e finalização.

[2]OpenStack é uma plataforma de código aberto desenvolvida para gerenciar múltiplos recursos virtualizados. `https://www.openstack.org`.

3.2.4. Questões de Desempenho x Segurança com NFV na IoT

Um fator a ser considerado no uso de qualquer recurso computacional é o seu desempenho. Este caracteriza-se por dois fatores: *métricas* e *medições*. As métricas são definições padrões de quantidades produzidas em um experimento, com uma utilidade pretendida. Elas devem ser cuidadosamente especificadas para transmitir o significado exato de um valor medido. A medição remete-se a um conjunto de operações com o objetivo de determinar o valor de uma métrica [IETF 2011, IETF 1998]. Dessa forma, os indicadores de desempenho computacionais estão diretamente relacionados às métricas dos recursos que demonstram o grau de funcionalidade de um sistema computacional e, com isso, representam de forma abstrata o estado desse sistema. Por exemplo, o tempo necessário para o carregamento de um aplicativo baseado na *web* é o indicador mais perceptível de que o sistema está funcionando no nível esperado. Assim, tempos de carregamento mais longos do que o normal indicam que o sistema pode estar com problemas, exigindo atuações de seus administradores. Esses indicadores mudam ao longo do tempo e são fontes iniciais de informação sobre a saúde do sistema [Moghaddam et al. 2019].

Do ponto de vista de NFV, a Tabela 3.1 apresenta métricas de desempenho relacionadas à velocidade, acurácia e confiabilidade de acordo com as categorias referentes à orquestração, operação de máquina virtual, estabelecimento e operação de redes e componentes de tecnologia como serviço. Essas métricas foram definidas pelo Instituto Europeu de Normas de Telecomunicações (ETSI – *European Telecommunications Standards Institute*) [ETSI 2014a, ETSI 2014b] para entidades que fornecem VNFs e que gerenciam infraestruturas virtualizadas, no intuito de garantir que os recursos oferecidos tenham o desempenho necessário de acordo com os requisitos de qualidade previstos. Por exemplo, uma rede que está sendo altamente demandada por requisições benignas, ou seja, que não são ataques, e possui a latência de provisionamento de máquina virtual (VM – *Virtual Machine*) alta para o serviço virtualizado de *firewall*, poderá gerar grandes atrasos nas respostas às requisições, o que afeta diretamente a qualidade de experiência do usuário final do serviço.

Na mesma linha do ETSI, a Força-Tarefa de Engenharia da Internet (IETF – *Internet Engineering Task Force*) [IETF 2017a] designou um grupo específico, chamado Grupo de Trabalho de Metodologia de *Benchmarking* (BMWG – *Benchmarking Methodology Working Group*), para produzir uma série de recomendações sobre as principais características e análise de desempenho de dispositivos, sistemas e serviços de rede. Nele a Metodologia de *Benchmarking* para Desempenho de Virtualização de Redes foi desenvolvida, considerando como métricas de desempenho a **taxa de transferência**, a **taxa de perda de quadros**, o **consumo de CPU**, o **consumo de memória** e a **latência**.

É possível encontrar, também, métricas relacionadas à NFV em outros trabalhos. Em [Gupta et al. 2019], por exemplo, os autores abordam questões relacionadas à tolerância a falhas de NFV e apresentam métricas específicas para o problema, tais como: perda de pacotes, taxa de transferência média de pacotes, congestionamento de pontos de interconexão, entre outras. Em [Kim et al. 2015], os autores apresentam métricas de comparação de desempenho para controladores de redes definidas por *software* (SDN – *Software Defined Network*) e NFV. As métricas apresentadas sobre NFV são concentradas em vCPU e vMemória (associadas à computação), latência e taxa de transferência

Tabela 3.1. Resumo das métricas de qualidade de serviço em NFV [ETSI 2014b].

Categoria da métrica	Velocidade	Acurácia	Confiabilidade
Primeira etapa da orquestração (por exemplo, alocação de recursos, configuração e instalação)	Latência de provisionamento de VM	Política de posicionamento de VM e conformidade	Confiabilidade de provisionamento de VM
Operação de VM	Interrupção da VM (duração e frequência do evento) e Latência de programação da VM	Erro de *clock* da VM	Taxa de liberação prematura de VM
Estabelecimento de rede virtual (VN – *Virtual Network*)	Latência de provisionamento da VN	Conformidade de diversidade de VN	Confiabilidade de provisionamento de VN
Operação de rede virtual	Atraso de pacote, *jitter* (variação no atraso) e taxa de transferência de pacotes entregues	Taxa de perda de pacotes	Conexão interrompida
Segunda etapa da orquestração (por exemplo, liberação de recursos)	–	–	Taxa de liberação de VM com falha
Componente de tecnologia como serviço (TcaaS)	Latência do serviço TcaaS	–	Confiabilidade TcaaS (por exemplo, relação de transação defeituosa) e interrupção do TcaaS

(associadas à comunicação), taxa de E/S e tempo de recuperação de VNFs (associadas ao armazenamento). Em [Zhang et al. 2016], os autores utilizam métricas como uso de CPU, taxa de transferência de pacotes e latência para apresentar como o efeito do acesso não uniforme a memória (NUMA – *Non-Uniform Memory Access*) e o posicionamento da cadeia de serviços impactam no desempenho de NFV.

Quando as características de dispositivos IoT são consideradas na implementação de uma rede, como baixo consumo de energia e recursos computacionais limitados, deve-se observar principalmente o comportamento da comunicação entre os dispositivos e os serviços disponibilizados a eles. Por exemplo, um determinado dispositivo IoT acoplado a um semáforo deve solicitar a um servidor remoto a resposta de um serviço de cálculo, baseado nas informações de volume de automóveis coletadas, para que o semáforo tenha seus tempos de abertura e fechamento alterados de acordo com o tráfego de veículos. Se essa comunicação não levar em consideração métricas como latência entre o dispositivo IoT e o servidor, o tempo de resposta para a requisição fará com que o serviço prestado não tenha a qualidade pretendida em relação à otimização da gestão do tráfego de veículos. Além disso, é necessário observar o comportamento dos serviços disponibilizados na rede IoT. Nesse contexto, a Figura 3.6 apresenta um cenário onde o serviço de IDS, provido por uma VNF, é consumido sob demanda de acordo com o volume de solicitações das câmeras, independentemente do *hardware* que o sustenta. A Figura 3.7 ilustra não só o consumo do serviço virtualizado de *firewall* feito pelo dispositivo IoT A em movimento, mas também a migração do estado em que ele se encontra entre os diferentes provedores

de VNF, nesse caso, Nuvem #1 e Nuvem #2. Um exemplo prático desse cenário seria um sistema de transporte urbano de uma cidade inteligente onde os ônibus poderiam ser representados pelos dispositivos IoT e as nuvens pelos diferentes provedores de computação disponíveis em localidades geográficas distintas. Para os dois cenários ilustrados, a RFC 8172 [IETF 2017b] adiciona outras métricas ao desempenho de NFV, como tempo para implantar e migrar VNFs, que devem ser levadas em consideração para que o tempo de resposta em uma rede IoT não seja afetado negativamente.

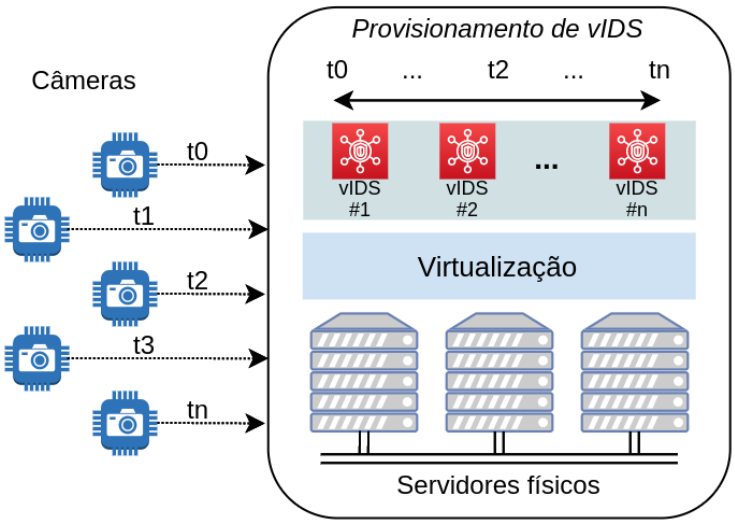


Figura 3.6. Escalando recursos de VNFs de segurança em redes IoT [Farris et al. 2019].

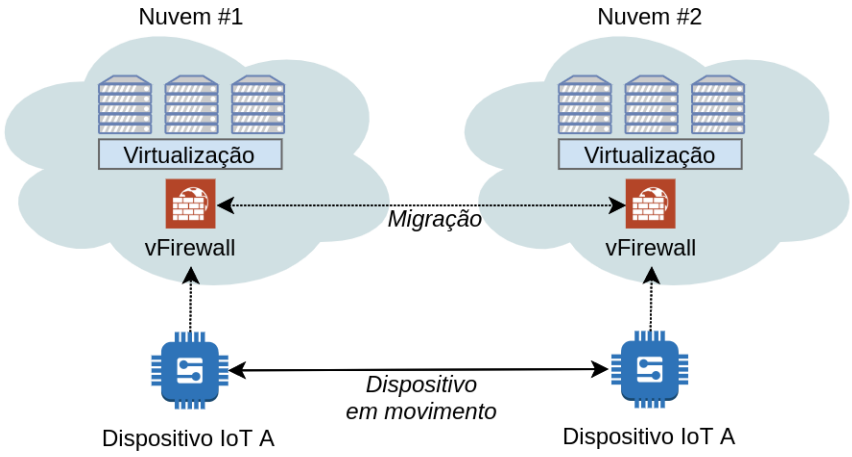


Figura 3.7. Migrando recursos de VNFs de segurança em redes IoT [Farris et al. 2019].

A aplicação de NFV especificamente para segurança em redes IoT é outro fator a ser considerado, abordagem que vem ganhando espaço atualmente [Alam et al. 2020, Farris et al. 2019, Zarca et al. 2018]. Nesse caso, além das justificativas para as métricas apresentadas na Tabela 3.1, a própria funcionalidade de segurança fornecida pela VNF

pode ter requisitos mais rígidos, principalmente em termos de tempo de resposta, para garantir que a detecção e a mitigação de um ataque sejam realizadas antes que seja tarde demais. Por exemplo, um IDS instalado na rede local onde os dispositivos de IoT estão conectados pode agir dentro do tempo esperado pelo fato de ele estar localizado no exato local onde os fluxos de interesse estão trafegando. Repassar esse tráfego para um ambiente computacional remoto, mesmo que esse ambiente tenha alta capacidade de processamento, pode tornar o IDS inútil caso a latência entre a rede local e o ambiente remoto seja muito alto. Esse é um caso em que não há diferentes gradientes para a qualidade do serviço, como no caso de um serviço que, mesmo se estiver "lento", ainda é útil. O não atendimento de um tempo mínimo, permitindo que o ataque tenha sucesso, significa que o serviço requisitado não foi entregue.

De um ponto de vista da camada de serviços, ilustrada na Figura 3.2, o trabalho de [Farris et al. 2019] apresenta abordagens para a aplicação de NFV voltadas à segurança de redes IoT. Isto possibilita a abrangência e a customização de serviços de proteção à rede oferecidos como, por exemplo, segurança como serviço (SECaaS – *Security as a Service*). Os autores classificam os trabalhos da literatura de acordo com quatro aspectos: separação do *software* de segurança do hardware; escalabilidade sob demanda e tolerância a falhas para VNF de segurança; suporte à mobilidade de VNF de segurança; e encadeamento de serviços de segurança de rede. Estendendo essa classificação proposta pelos autores, é possível considerar também o uso de VNF para implementação de modelos de aprendizado de máquina no intuito de detectar e mitigar ataques na rede, denominando a proposta de VNF para modelos de aprendizado de segurança.

Sobre a separação do *software* de segurança do hardware, três trabalhos se destacam. Em [Bremler-Barr et al. 2014], os autores apresentam uma abordagem de virtualização de serviços de DPI cujo desempenho, medido em taxa de transferência de pacotes, foi melhorado quando comparado a soluções implementadas em *middleboxes*. Em [Montero et al. 2015], os autores propõem virtualizar as aplicações de segurança na borda da rede baseadas em domínios virtuais de confiança (TVD – *Trusted Virtual Domain*), um contêiner lógico instanciado na rede composto por aplicativos de segurança do usuário e dados de controle de acesso a outras TVDs. Em [Yu et al. 2015], os autores propõem uma arquitetura de segurança chamada IoTSec, a qual contempla *micro-middleboxes* customizados (*µboxes*) que atuam como *gateways* de segurança para cada dispositivo IoT e cuja implementação pode se dar por meio de VNF. A IoTSec é composta por um controlador centralizado que monitora os contextos dos dispositivos e o ambiente operacional para gerar uma visão global e aplicar políticas entre os dispositivos. Com base nessa visualização, ele instancia e configura *µboxes* individuais e os mecanismos de encaminhamento necessários para rotear pacotes para elas.

Sobre a escalabilidade sob demanda e tolerância a falhas para VNF de segurança, em [Cao et al. 2015], os autores apresentam uma solução chamada de NFV-VITAL. Seu objetivo é determinar a configuração que produz o maior desempenho de uma VNF, ou seja, ela computa a carga de trabalho máxima que uma VNF suporta antes que a qualidade do serviço reduza, usando diferentes tamanhos de implantação e opções de virtualização.

A análise de soluções de IDS virtualizadas, como Snort[3] e Suricata[4], demonstrou os benefícios de selecionar o dimensionamento e as configurações ideais para os mecanismos de segurança. Em [Hafeez et al. 2016], os autores abordam a manutenção de réplicas com reconhecimento de estado de *middleboxes* virtuais com o objetivo de, em caso de falhas, instanciar novos serviços.

Sobre o suporte à mobilidade de VNF de segurança, é possível citar uma estrutura de suporte à migração de instâncias virtuais de segurança perto dos dispositivos do usuário final, como proposto por [Montero and Serral-Gracià 2016]. A abordagem aproveita o uso de tecnologias de virtualização e de SDN para gerenciar a migração de aplicativos de segurança na extremidade da rede, enquanto minimiza a interrupção das conexões em andamento. A solução é composta por quatro componentes: um contêiner virtual de segurança, cuja função é fornecer aplicativos como *firewall* ou alguma composição entre eles; um controlador de rede, que é responsável por configurar o direcionamento do tráfego de rede para os aplicativos de segurança; um migrador de recursos, que realiza a função de mover o estado de um aplicativo de segurança específico para o novo local do usuário; e um orquestrador que coordena a alocação dos recursos, a configuração de rede e a migração dos contêineres virtuais de segurança.

A respeito do encadeamento de serviços de segurança de rede, em [Gember-Jacobson et al. 2014], os autores apresentam o OpenNF, uma solução que implementa um plano de controle dedicado para garantir o controle coordenado dos estados da VNF e dos estados de encaminhamento da rede. Em [Qazi et al. 2013], os autores desenvolveram uma abordagem baseada na aplicação de políticas de SDN para simplificar o encaminhamento de tráfego entre *middleboxes*. A solução, chamada de SIMPLE (*Software-defIned Middlebox PoLicy Enforcement*), é composta pelo processamento de políticas nas quais os administradores de rede configuram suas lógicas de processamento, abstraindo informações sobre onde esse processamento ocorre ou como o tráfego precisa ser roteado. Assim, o SIMPLE traduz as políticas configuradas para a infraestrutura física considerando os dados da topologia e do tráfego da rede. Por fim, a solução também considera restrições de dispositivos como CPU, memória, aceleradores para diferentes *middleboxes* e quantidade de memória TCAM (*Ternary Content Addressable Memory*) disponível para instalar regras de encaminhamento nos *switches* SDN. O trabalho de [Wang et al. 2021] não se restringe a serviços de segurança. Nele os autores propõem uma abordagem baseada em aprendizagem por reforço profundo (do inglês, *deep reinforcement learning*) para posicionar as cadeias de serviços de rede de forma adaptativa. O algoritmo extrai características da rede física em tempo de execução por meio de um grafo de rede convolucional (GCN – *Graph Convolutional Network*) e gera estratégias de posicionamento de serviços através de um modelo de sequência a sequência (Seq2Seq – *Sequence-to-Sequence*), no qual aprende a tomar decisões de posicionamento de cadeias de serviços observando o desempenho correspondente de decisões anteriores.

Dentre os trabalhos da literatura relacionados ao uso de VNF para modelos de aprendizado de segurança, é possível citar [Bülbül and Fischer 2020], onde é proposta

[3]Snort é um sistema de prevenção de intrusão (IPS – *Intrusion Prevention System*) baseado em código aberto. `https://www.snort.org`.

[4]Suricata é um IDS e um IPS também baseado em código aberto. `https://suricata.io`.

uma abordagem de mitigação de ataques de negação de serviço distribuído (DDoS – *Distributed Denial of Service*). Este utiliza SDN e NFV por meio de um algoritmo de aprendizado de máquina baseado em generalização e sumarização e LMP (do inglês, *Longest Matching Prefix*), que derivam padrões para descrever ataques na rede. Após a identificação do ataque, o módulo de geração de padrões deriva uma regra OpenFlow[5] para os demais *switches* da rede, solicitando a filtragem do tráfego. Caso o ataque extrapole a rede local, as regras de filtros de tráfego também podem ser propagadas para roteadores externos à rede local.

Em [Jia et al. 2020], os autores têm como objetivo propor técnicas de defesa contra ataques DDoS de IoT e apresentam um esquema de defesa centrado na borda da rede, denominado FlowGuard. Este consiste na composição de dois fluxos, o de filtro e o de manipulação. O fluxo de filtro utiliza uma tabela que define as regras de filtragem de fluxo para diferentes tipos de ataque DDoS. Também é composto por um algoritmo que detecta anomalias no tráfego de dispositivos IoT que passa por servidores de borda. Uma vez identificada a anomalia na rede, o fluxo de manipulação é ativado e a classificação do ataque DDoS ocorre utilizando técnicas de aprendizado de máquina, como memória de curto prazo longo (LSTM – *Long Short-Term Memory*) e rede neural convolucional (CNN – *Convolutional Neural Network*), que podem ser implementadas por meio de VNFs.

Pode-se dizer que métricas e medições são as bases para o processo de monitoramento de qualquer tipo de rede de computadores, sendo utilizadas para determinar o estado dos componentes e, também, para servir de insumos para o gerenciamento de desempenho. Em redes tradicionais, o desempenho da rede leva em conta questões como transferência de dados, o tempo de resposta da rede, a perda de pacotes, a porcentagem do uso de recursos, a utilização da conexão de dados, entre outras. Contudo, quando um contexto de rede envolve dispositivos IoT juntamente com abordagens de segurança baseadas em NFV, outras questões devem ser incluídas para nortear a análise de seu comportamento e o impacto que elas trazem para o seu desempenho. Alguns exemplos são: no domínio de IoT, tipos de dispositivos e protocolos de comunicação utilizados por eles; em NFV, provisionamento, escalabilidade, conexões de rede e orquestração de VNFs; e no contexto de segurança, ferramentas, algoritmos, tempo de detecção e mitigação de ameaças.

3.3. Gerenciamento de Desempenho e Segurança em IoT com NFV

O gerenciamento de desempenho parte do princípio de como utilizar os indicadores ou métricas coletadas da rede para provisionar de forma eficiente recursos necessários para a manutenção da qualidade dos serviços entregues aos seus usuários. Por isso, para que a operação de um recurso de NFV seja realizada de forma eficiente, os indicadores de desempenho devem ser bem definidos e aferidos a fim de demonstrarem o comportamento e a saúde do sistema. Por serem recursos virtualizados, um fator a ser ponderado por meio da medição dos indicadores é a capacidade máxima da infraestrutura física em que estão hospedados, ou seja, o orquestrador de VNFs não deve permitir a hiper alocação de recursos em uma infraestrutura compartilhada, fazendo com que outros recursos

[5]OpenFlow é um protocolo de comunicação que permite o acesso ao plano de encaminhamento de um *switch* ou roteador, possibilitando o direcionamento e até mesmo o descarte de pacotes da rede. `https://opennetworking.org`.

disponibilizados virtualmente tenham seu desempenho afetado. Nessa mesma linha, o orquestrador também não deve manter alocados recursos que não têm utilidade em um determinado contexto que a rede contempla. Em outras palavras, os indicadores de desempenho contribuem para a tomada de decisão no uso ou não de um determinado recurso para um propósito identificado. Por exemplo, um recurso virtualizado de DPI pode ser alocado a partir do momento em que a rede começa a receber requisições de origens que ela geralmente não recebe, fazendo com que os pacotes sejam inspecionados de forma profunda, com o objetivo de detectar códigos maliciosos. Uma vez não identificado algum tipo de ameaça, a desconexão com o endereço que originou a análise ou até mesmo após a identificação e mitigação do ataque, o recurso virtualizado de DPI deve ser desalocado, liberando recursos computacionais às demais necessidades da rede. Assim, a perspectiva de **gerenciamento de desempenho** apresentada nesta seção, analisa como as métricas e medições de desempenho, apresentadas na Seção 3.2.4, foram abordadas por diversos autores em suas soluções.

Além disso, ao se tratar de dispositivos IoT, é preciso investigar a fundo suas características antes mesmo de propor qualquer tipo de infraestrutura ou mecanismos de segurança dedicados a eles [Lin et al. 2017]. Assim, a perspectiva **IoT** leva em consideração as seguintes características para a análise dos trabalhos da literatura apresentada nesta seção:

- **Dispositivos** utilizados no experimento. A variedade de dispositivos contidos no guarda-chuva de IoT é grande e tende a aumentar. Segundo a Corporação Internacional de Dados (IDC – *International Data Corporation*) [IDC 2020], em 2025, 75% dos mais de 55 bilhões de dispositivos previstos em todo o mundo estarão conectados a uma plataforma IoT. Além disso, a corporação estima que os dados gerados a partir de dispositivos IoT conectados estejam na casa dos 73 ZB (73×10^{21} Bytes), sendo a maioria proveniente de segurança e vigilância por vídeo, além de aplicações industriais. É possível dizer que grande parte dos dados gerados e transitados nessas redes envolve informações privativas de usuários, tais como imagens, documentos, localizações, preferências, entre outros. Por isso, identificar como esses dispositivos estão sendo utilizados auxilia no reconhecimento de possíveis limitações de segurança que podem ser supridas por recursos de NFV;

- **Protocolo** abordado. A aplicação de padrões da Internet já consolidados a dispositivos inteligentes - como o IP - pode simplificar a integração dos cenários previstos em IoT. No entanto, os protocolos de comunicação convencionais da Internet precisam ser modificados ou estendidos para oferecer suporte a necessidades específicas de aplicações IoT. Nessa linha, [Sobin 2020] contempla em seu trabalho um estudo dos protocolos e tecnologias de comunicação dedicadas às camadas: física, contida pelo protocolo IEEE 802.12.4; camada MAC, por IEEE802.15.4E, ZigBee e TSMP (do inglês, *Time Synchronized Mesh Protocol*); camada de convergência, contida por 6LoWPAN; camada de transporte, por UDP (do inglês, *User Datagram Protocol*), IPv6, uIP (*micro* IP), ROLL (do inglês, *Routing Over Low power and Lossy networks*), RPL (do inglês, *IPv6 Routing Protocol for Low Power and Lossy Networks*) e DTLS (do inglês, *Datagram Transport Layer Security*); e camada de aplicação, por CoAP (do inglês, *Constrained Application Protocol*), MQTT (do in-

glês, *Message Queuing Telemetry Transport*), DDS (do inglês, *Data-Distribution Service*), AMQP (do inglês, *Advanced Message Queuing Protocol*), SMCP (do inglês, *Secure Mobile Crowdsensing Protocol*) e LLAP (do inglês, *Lightweight Local Automation Protocol*). Diferentemente de [Sobin 2020], [Lin et al. 2017] consideram que a camada de percepção é dedicada a tecnologias como RFID (do inglês, *Radio Frequency Identification*) e demais sensores de redes IoT. Os protocolos IEEE 802.15.4, 6LoWPAN, ZigBee, Z-Wave, MQTT, CoAP, DDS, AMQP e XMPP (do inglês, *Extensible Messaging and Presence Protocol*) são categorizados na camada de rede, pois a camada de aplicação refere-se ao contexto de negócio do uso de IoT como, por exemplo, cidades e transportes inteligentes. Outros trabalhos [Meneghello et al. 2019, Neshenko et al. 2019, Deogirikar and Vidhate 2017, Yang et al. 2017, Gao et al. 2013] também fazem estudos de camadas e protocolos de comunicação a fim de direcionar soluções, desafios e limitações no contexto de IoT. Portanto, identificar e relacionar os protocolos de comunicação ajuda a compreender de que forma a NFV tem contribuído com esses desafios. Baseado nisso, a Tabela 3.2 traz um consenso entre as camadas e os protocolos de comunicação IoT abordados nos trabalhos analisados;

Tabela 3.2. Protocolos referentes às camadas IoT.

Camada	Protocolo
Aplicação	HTTP(S), MQTT, CoAP, AMQP, LLAP, SMCP, DDS, XMPP
Rede	TCP, UDP, UIP, ROLL, RPL, DTLS, Zigbee, Z-Wave, IPV6, 6LoWPAN, LoRaWAN, BLE
Percepção	IEEE 802.15.4, IEEE 802.15.4E, TSMP

- **Arquitetura** apresentada. Existem diversas vantagens em considerar a utilização da computação em nuvem para aplicações tradicionais, como a virtualização de recursos, escalabilidade, redundância, pagamento de acordo com o uso (do inglês, *pay-as-you-go*), entre outros. No entanto, para aplicações IoT nas quais os dispositivos possuem capacidades limitadas, o processamento de determinadas tarefas deve ser dividido e realizado entre os demais nós da rede em diferentes localidades, e quando esse cenário abrange milhares de dispositivos interconectados, é preciso pensar em outras estratégias de infraestrutura para que o desempenho das aplicações não seja afetado negativamente. Por essa razão, outros paradigmas, como a computação em névoa ou *fog computing*, surgiram para suportar tal realização [Yousefpour et al. 2019, Lin et al. 2017, Singh et al. 2016]. Diferentes abordagens podem ser encontradas no intuito de aproximar o processamento computacional dos dispositivos IoT e reduzir a latência entre o provisionamento de serviços. Por exemplo, a Figura 3.8 ilustra a quantificação dos atributos quando próximos ou distantes da nuvem, além de diferentes níveis de processamento, partindo do nível mais capacitado (nuvem) para o menos capacitado (borda), contemplando um nível intermediário (névoa). Dessa forma, entender as características de infraestrutura consideradas pelos autores dos trabalhos analisados afeta diretamente o desempenho do uso de recursos NFV.

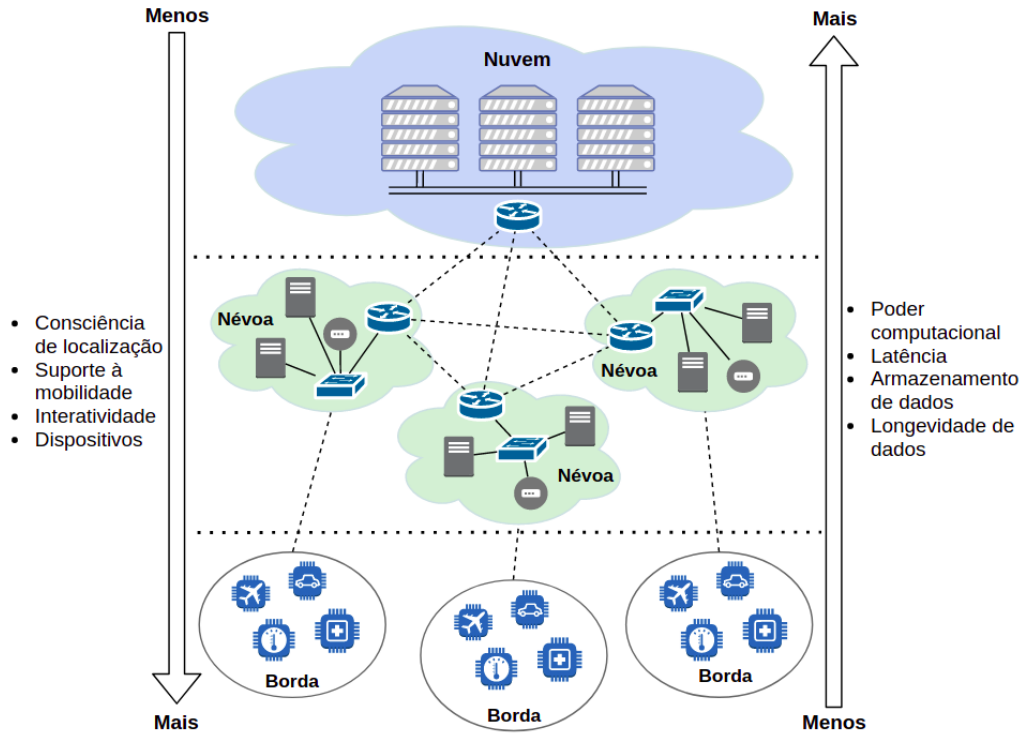


Figura 3.8. Níveis de processamento e capacidades em diferentes arquiteturas distribuídas [Yousefpour et al. 2019, Lin et al. 2017].

É fato que dispositivos IoT possuem características específicas e são desenvolvidos para atuar em cenários bem definidos, ou seja, não possuem um propósito geral de atuação, como computadores pessoais. Por exemplo, equipamentos hospitalares são desenvolvidos visando alcançar a máxima redução de tempo entre a coleta dos sinais vitais de uma pessoa e sua exibição no monitor. Em um contexto industrial, sensores são utilizados para determinar a abertura ou fechamento de comportas de acordo com o volume de determinado líquido. Na gestão de tráfego urbano, dispositivos inteligentes podem coletar informações referentes ao número de carros nas vias e determinar o tempo de fechamento de um ou mais semáforos. Os exemplos são extensos. No entanto, baixo consumo de energia, protocolos de comunicação otimizados e suporte à conexão entre dispositivos heterogêneos são alguns dos requisitos em comum que devem ser observados durante o desenvolvimento de mecanismos de segurança. Assim, do ponto de vista de **segurança**, a análise dos trabalhos da literatura apresentada nesta seção foi baseada nos seguintes itens:

- **Abordagem** da solução proposta. O propósito de um determinado mecanismo de segurança deve ser definido logo na etapa de planejamento de um projeto, sempre levando em conta os fundamentos de confidencialidade, integridade e disponibilidade. Quando em pleno funcionamento, o mecanismo deve objetivar duas principais ações: a detecção e a mitigação de ameaças à rede. A ação de detectar ameaças pode ser considerada uma etapa preliminar do processo de resposta a incidentes de segurança. Do mesmo modo, a mitigação pode ser vista como uma etapa subsequente, cujo objetivo é desabitar ou neutralizar a ameaça presente na rede. Por

exemplo, uma vez identificado um ataque DDoS pelo mecanismo de detecção, o sistema de mitigação pode entrar em ação no intuito de negar ou redirecionar os pacotes maliciosos para o descarte. Entender e avaliar esses mecanismos ajuda a compreender em que nível de completude e maturidade do uso de NFV a solução proposta está;

- **Ataque** considerado. Qualquer computador conectado via rede pode ser alvo de um ataque, da mesma forma que pode participar de um ataque e isso não é diferente em plataformas IoT. Neste contexto, o trabalho realizado por [Meneghello et al. 2019] apresenta vulnerabilidades em aberto de tecnologias como Zigbee, BLE (do inglês, *Bluetooth Low Energy*), 6LoWPAN e LoRaWAN da camada de rede. [Yang et al. 2017] pontua as principais limitações de dispositivos IoT, além de apresentar e analisar questões de segurança classificadas pelas camadas de percepção, rede, transporte e aplicação. No trabalho de [Deogirikar and Vidhate 2017], além de considerar os ataques a redes IoT em camadas física, de rede e de *software*, são pontuados ataques específicos de encriptação, tais como: *side-channel*, *cryptanalysis of a simple modular exponentiation*, *ciphertext only*, *known plaintext*, *chosen plaintext*, *chosen ciphertext* e homem no meio (MITM – *Man-in-the-Middle*). [Neshenko et al. 2019] vai além da classificação feita pelas camadas baseadas em dispositivo, rede e *software*, considerando o impacto que os ataques têm em relação à confidencialidade, integridade, disponibilidade e responsabilização. Para isso, os autores consideram que para a camada baseada em dispositivo, as vulnerabilidades identificadas são referentes à deficiência física ou captação de energia insuficiente. As vulnerabilidades de autenticação inadequada, encriptação inapropriada e portas desnecessariamente abertas são citadas para a camada baseada em rede e, para a camada de *software*, são considerados itens como controle de acesso insuficiente, gerenciamento impróprio de atualizações de correção, práticas inseguras de desenvolvimento de código e mecanismos insuficientes de autenticação. A Tabela 3.3 apresenta uma descrição dos diferentes tipos de ataques em redes IoT e auxilia na compreensão das ameaças que têm sido detectadas e mitigadas pelo uso de NFV.

[Farris et al. 2017] apresentam uma proposta de um *framework* cujo objetivo é gerenciar e orquestrar políticas de segurança em redes IoT, na qual a heterogeneidade de dispositivos físicos e virtuais prevalece. A arquitetura do *framework* é composta pelos planos de usuário, de orquestração, de aplicação de segurança e de gerenciamento. O plano de usuário implementa e oferece uma interface para a definição de políticas de segurança, tais como autenticação e autorização, filtragem e encaminhamento e proteção de canal. Após as políticas de segurança serem definidas pelo usuário, o plano de orquestração as interpreta e as incorpora no processo de execução. Além disso, o plano de orquestração também inclui os módulos de monitoramento, reação, orquestração de segurança e ativadores, que são responsáveis, respectivamente, por coletar as informações em tempo real dos componentes da rede; interpretar as informações recebidas do módulo de monitoramento e selecionar as contramedidas de acordo com as políticas de segurança definidas; selecionar os ativadores de acordo com a contramedida; e aplicar as políticas de segurança. Os ativadores, neste caso, são considerados os recursos provisionados via VNFs. O plano de aplicação de segurança é contido pelos domínios de gerenciamento e controle,

Tabela 3.3. Tipos de ataques a redes IoT e suas descrições.

Ataque	Descrição
Vírus, *worms*, cavalo de tróia, *spyware* e *adware*	O invasor pode alterar o comportamento do sistema, roubar dados ou invalidar acessos usando esses códigos maliciosos.
Scripts maliciosos	Ao injetar um *script* malicioso, o invasor obtém acessos privilegiados ao sistema.
Phishing	O invasor obtém as informações confidenciais como nome de usuário e senhas através da divulgação de *links* via e-mail, redirecionando a vítima para páginas falsas.
Negação de serviço (DoS) ou negação de serviço distribuída (DDoS)	O invasor faz com que o sistema alvo pare de fornecer seus serviços por meio do consumo de largura de banda da rede.
Sinkhole	O invasor compromete um nó dentro da rede e executa o ataque a partir desse nó. O nó comprometido envia informações de roteamento falsas para os nós vizinhos de que tem o caminho mínimo e, em seguida, atrai o tráfego para ele.
Roteamento	O invasor modifica e envia informações de roteamento anormais. Isso pode resultar no descarte de pacotes, encaminhamento de dados errados ou particionamento da rede.
Homem no meio (MITM)	O invasor intercepta a comunicação entre dois nós da rede e obtém acesso aos pacotes trocados por eles.
Sybil	O nó malicioso assume ilegalmente várias identidades e age de acordo com elas.
Captura de tráfego	O invasor intercepta e examina as mensagens para obter informações da rede.
Esgotamento de bateria	O invasor consome mais energia do que o previsto e, consequentemente, desativa o dispositivo.
Interferências de rádio frequência (RF) em RFIDs	O invasor envia sinais com ruídos via radio frequência a fim de clonar a etiqueta do dispositivo alvo para obter acesso não autorizado ou desabilitar a comunicação.

infraestrutura e virtualização, VNF e IoT. O domínio de gerenciamento e controle supervisiona o uso dos recursos em tempo de execução, além de orquestrar as VNFs sobre a infraestrutura virtualizada seguindo as recomendações de gerenciamento e orquestração da ETSI. O domínio de infraestrutura e virtualização é responsável por disponibilizar, de forma virtualizada, os recursos de CPU, memória e elementos de redes. O domínio de VNF provê os recursos virtualizados e utilizados sob demanda para a rede, tais como *firewall*, IDS, DPI, entre outros. Por fim, o domínio de IoT inclui os dispositivos que fazem parte da rede a ser gerenciada.

Para demonstrar a aplicabilidade do *framework*, os autores descrevem dois estudos de caso e experimentam seu desempenho em uma prova de conceito em [Zarca et al. 2018]. O primeiro estudo de caso considera um cenário MEC (do inglês, *Multi-access Edge Computing*), no qual o principal objetivo do *framework* é suportar a capacidade de prover VNFs mais próximas dos dispositivos IoT, fazendo com que os requisitos de qualidade da rede sejam atendidos. O segundo estudo de caso apresenta um cenário relacionado a prédios inteligentes, BMS (do inglês, *Building Management System*), onde a proteção de dispositivos distintos, como computadores e aparelhos de ar condicionado, pode se dar por políticas de segurança que diferem os canais de comunicação. A

prova de conceito foi focada na aplicação de políticas para provisionar mecanismos de segurança em SDN. Os experimentos consideraram a comparação entre três abordagens. A primeira e a segunda foram implementadas pelos controladores SDN *OpenDaylight* (ODL) e *Open Network Operating System* (ONOS), respectivamente, e a terceira abordagem por um componente de VNF de *firewall* baseado em NETCONF e iptables. Também consideraram um ambiente sob ataque do tipo MITM já detectado, utilizando como estratégia de mitigação o isolamento dos sensores comprometidos. Sobre os dispositivos IoT, foi utilizada uma máquina virtual executando um emulador Cooja Contiki, contido por sensores IoT conectados por meio do protocolo 6LoWPAN.

Os resultados foram aferidos de acordo com o tempo de reação necessário para cada abordagem implementada em um intervalo de 1 a 1000 aplicações de políticas de segurança para a mitigação do ataque. Em comparação com os controladores SDN ODL e ONOS, a implementação do *firewall* em VNF teve um desempenho inferior, principalmente na etapa de aplicação das políticas. Segundo os autores, o tipo de comunicação foi o principal motivo para essa perda de desempenho, ou seja, enquanto que o SDN estabelece inicialmente uma conexão TLS e a mantém aberta durante a seção em que está ativo, o NETCONF realiza a abertura de um canal SSH para cada solicitação.

A evolução do *framework* proposto inicialmente por [Farris et al. 2017] e experimentado por [Zarca et al. 2018] teve sua continuação em outros trabalhos. Por exemplo, em [Zarca et al. 2019b], os autores apresentam um abordagem para gerenciar políticas de autenticação, autorização e responsabilização (AAA – *Authentication, Authorization e Accounting*) na IoT através de uma VNF chamada de vAAA. Ela se estabelece em nível de névoa juntamente com políticas de segurança, sendo executadas em um controlador SDN em nível de nuvem. Os autores também propõem realizar a comunicação entre dispositivos IoT e serviços da rede por meio de canais virtuais protegidos (vChannel-Protection), em específico, utilizando o DTLS. Para a experimentação, foram utilizados dispositivos IoT em uma versão personalizada do Contiki OS 2.7 e do servidor erbium CoAP, juntamente com o protocolo 6LoWPAN.

O desempenho dos componentes propostos foi avaliado de acordo com o processo de comunicação segura entre os dispositivos IoT e o *broker*[6], que consiste nas etapas de autenticação, autorização e canal protegido a partir das ações de refinamento, tradução e aplicação da política de segurança. Sob essa dimensão, os valores apresentados para as etapas de autenticação, autorização e canal protegido se mantiveram constantes entre 0 e 0,1 segundos para todas as ações, exceto para a de aplicação da política na etapa de canal protegido, que consumiu 0,35 segundos. Os autores também avaliaram o consumo de tempo para a aplicação de políticas, CPU e memória RAM de acordo com a escalabilidade de dispositivos IoT em um intervalo de 1 a 500. Os resultados mostraram que o tempo médio para as ações de refinamento, tradução e aplicação das políticas foi em média de 30 segundos para 500 dispositivos IoT na rede. O consumo de CPU tornou-se constante, próximo dos 100%, a partir de 100 dispositivos IoT e o consumo de memória RAM se manteve estabilizado para as ações de refinamento e tradução de políticas, em torno de 40 MB, enquanto que para a ação de aplicação da política o consumo chegou aos 120 MB.

[6]Um *broker* é um elemento de software usado em protocolos do tipo *publish-subscribe* que recebe as mensagens dos publicadores e as envia para os assinantes.

Em [Zarca et al. 2019a], os autores apresentam uma arquitetura referente ao gerenciamento de segurança para o *framework*. Nessa etapa, os autores desenvolveram um módulo de gerenciamento composto por um componente chamado de filtragem e pré-processamento de dados, que realiza a captura de informações da rede e as encaminha ao detector de incidentes. Este último executa a análise de segurança, procurando possíveis ataques na rede. O módulo também contém um banco de dados de assinaturas de ataques e um outro componente baseado em inteligência artificial, que aplica técnicas de aprendizado de máquina para detectar anomalias de comportamento, componente esse desenvolvido por [Bagaa et al. 2020]. Os experimentos implementados consideraram os dois cenários citados anteriormente, MEC e BMS, e os mesmos dispositivos de [Zarca et al. 2019b]. Além disso, foram realizados ataques DDoS no cenário MEC e a simulação de um dispositivo IoT infectado por um *malware* a fim de demonstrar a capacidade do módulo em detectar e mitigar a ameaça na rede.

Os resultados apresentados na análise do desempenho foram relacionados à detecção e mitigação dos ataques. Para a detecção, a solução mostrou um tempo médio de 460 milissegundos em 100 testes realizados. Para a mitigação, ou seja, a aplicação de políticas de segurança na rede, o tempo médio entre tradução e aplicação foi de 406 milissegundos também para as 100 iterações. Portanto, a execução do ciclo completo de resposta a ataques de DDoS levou menos de um segundo.

Seguindo a mesma linha de evolução do *framework*, o trabalho de [Zarca et al. 2020b] traz uma nova abordagem de mitigação de ataques à rede IoT por meio da implementação automatizada de *honeynets* baseadas em VNFs. A Figura 3.9 ilustra a criação de uma rede de dispositivos IoT virtualizados (vIoTHoneynet) sem nenhuma função para que, após a detecção de um ataque, o redirecionamento do tráfego seja feito para a vIoTHoneynet através de regras atualizadas no controlador SDN. Os experimentos foram baseados em dois cenários BMS e os experimentos foram executados 100 vezes para o mesmo caso de teste.

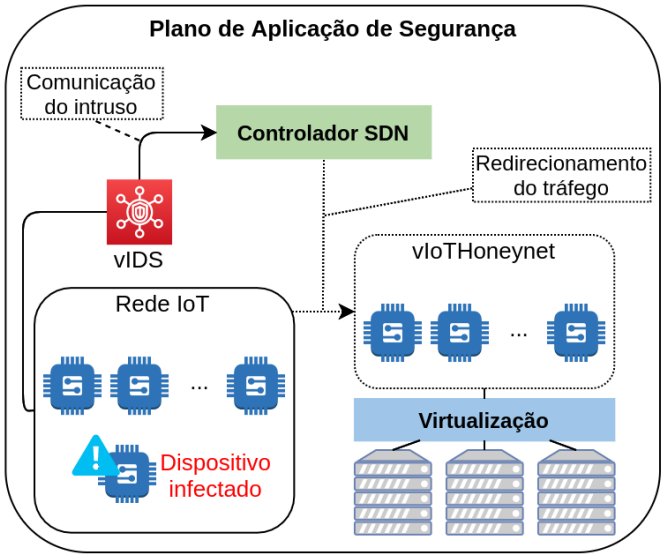


Figura 3.9. Arquitetura vIoTHoneynet [Zarca et al. 2020b].

Considerando que a vIoTHoneynet é instanciada sob demanda, ou seja, apenas quando um ataque é detectado na rede IoT, os experimentos consideraram dois tipos de tempos para compor o tempo total de mitigação. O primeiro tempo foi referente à configuração da vIoTHoneynet composta pelos tempos de compilação e carregamento dos códigos dos novos dispositivos IoT, além do tempo de configuração da rede para esses dispositivos. O segundo tempo foi referente ao estabelecimento da vIoTHoneynet na rede, por exemplo, pela configuração do roteamento das conexões do atacante através da implementação de novas regras no controlador SDN e pelo tempo de aplicação das políticas de segurança em si.

O primeiro cenário de experimentação contou com 20 e o segundo com 50 dispositivos IoT remotamente acessíveis, distribuídos em sensores de umidade, temperatura, luz, CO2, RFIDs para a abertura de portas e roteadores RPL. No entanto, os cenários não se diferenciaram apenas pela quantidade de dispositivos, mas também pela versão do sistema operacional Contiki utilizado para a simulação dos dispositivos IoT, respectivamente, 2.7 e 3.1. Os autores também consideraram a implantação de diferentes tamanhos de vIoTHoneynets para cada cenário. Esses tamanhos de teste foram de no mínimo 10 dispositivos a no máximo 50. Os resultados mostraram que para o primeiro cenário, o tempo médio para a configuração da vIoTHoneynets foi de 15 segundos. Para o segundo cenário, o tempo médio de configuração foi de 45 segundos. Quando os tempos de configuração da vIoTHoneynets foram somados ao tempo de aplicação das políticas de segurança para mitigar o ataque, o tempo médio de mitigação do ataque para o primeiro cenário ficou em torno de 51,5 segundos, e para o segundo cenário, 160 segundos. Os autores também aferiram o consumo de CPU e memória RAM das vIoTHoneynets criadas. Em ambos os cenários, o consumo de CPU se manteve em 100% para todos os tamanhos de teste, enquanto que o consumo de memória RAM no primeiro cenário cresceu de acordo com o aumento dos testes se mantendo no limite de 20% de uso, diferentemente do segundo cenário, que alcançou um consumo máximo de 25%.

Em relação à aplicação de aprendizagem de máquina para a detecção de anomalias em redes IoT, o trabalho de [Bagaa et al. 2020] propõe um modelo baseado em aprendizagem supervisionada desenvolvido sobre o *framework* proposto por [Farris et al. 2017]. Nele, os autores implementam um agente de inteligência virtual que recebe as informações do módulo de monitoramento da rede e as analisa no intuito de detectar padrões referentes a ataques como DDoS, *Probing*, U2R (do inglês, *User to Root*) e R2L (do inglês, *Remote to Local*). Os modelos de aprendizagem *Random Forest*, J48, *Bayesian Network* e *Hoeffding Tree* foram desenvolvidos e treinados a partir de conjuntos de dados que incluem os tipos de ataque citados anteriormente. Os experimentos foram baseados na coleta de dados de sensores de temperatura e CO2 de salas distintas e subdivididos em quatro categoria: (SV), conjunto de dados contento apenas os valores de medição e data; (P5V), conjunto de dados que inclui data, valor, valor precedente, valor 2º precedente, valor 5º precedente; (PD3V), que inclui data, valor, valor precedente diferente, 2º precedente diferente, 3º precedente diferente; e (CR), conjunto de valores cruzados entre as salas sendo sua estrutura data, sala 1, sala 2, sala 3, sala 4, etiqueta. Os resultados apresentados mostraram que para os dados de sensores de temperatura, o modelo desenvolvido teve uma acurácia de 99,71% para SV e P5V, de 99,28% para PD3V e de 99,23% para CR. Para o dados de sensores de CO2, o modelo mostrou uma acurácia de 98,86%

para SV, de 99,24% para P5V e de 99,13% para PD3V.

No mesmo contexto referente à aplicação de aprendizagem de máquina, o trabalho de [Sairam et al. 2019] apresenta uma solução chamada NETRA, que disponibiliza VNFs na borda da rede IoT no intuito de melhorar sua segurança. A solução foi desenvolvida para ser executada em dispositivos de baixa capacidade de processamento computacional. Por isso, utiliza uma estrutura baseada em contêineres Docker[7] para suportar as VNFs de segurança. A Figura 3.10 ilustra como os componentes da solução são distribuídos. A camada de núcleo da rede é composta pela conexão com o provedor de Internet (ISP – *Internet Service Provider*) e pelo Docker Hub[8], responsável por armazenar as imagens utilizadas em cada VNF. A camada de borda contém o *gateway* IoT desenvolvido em Raspberry Pi 3, onde se implementa as VNFs para aquele domínio de segurança. A camada de dispositivos IoT é composta pelos dispositivos inteligentes utilizados na rede. Todo o tráfego de rede realizado na borda é analisado pelo componente vAnalytics que implementa um modelo de aprendizado de máquina baseado em *Random Forest* para classificar os ataques a partir de anomalias identificadas. O modelo utilizado foi treinado a fim de detectar e mitigar ataques do tipo DDoS.

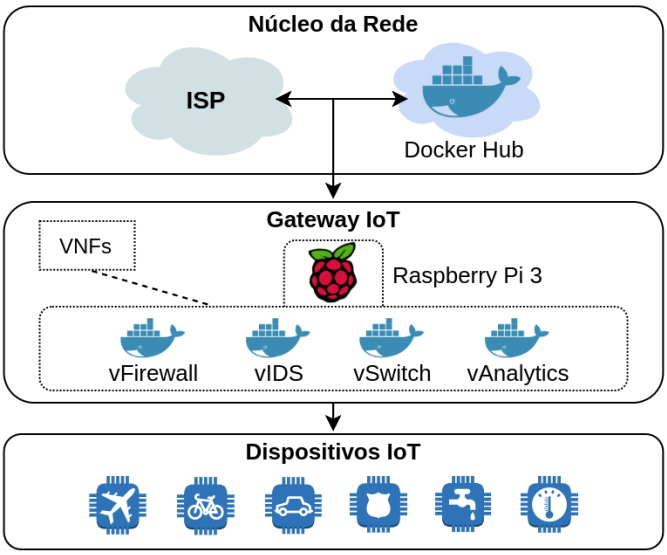


Figura 3.10. Arquitetura NETRA [Sairam et al. 2019].

Os experimentos foram realizados sob duas perspectivas de implementação, a de máquinas virtuais e a de contêineres, com o objetivo de avaliar os indicadores de desempenho: armazenamento, memória, latência, taxa de transferência e escalabilidade. Sobre o desempenho de armazenamento, as máquinas virtuais requisitaram 72 GB de espaço, enquanto que os contêineres já instanciados com as VNFs requisitaram 465 MB. Sobre o consumo de memória, as máquinas virtuais tiveram um pico de 140 MB, e os contêineres de 110 MB, para quatro instâncias em execução. Os valores apresentados de latência

[7]Docker é uma plataforma que utiliza a virtualização para executar aplicações em pacotes chamados contêineres. Os contêineres são isolados uns dos outros e agrupam suas próprias aplicações, bibliotecas e arquivos de configuração `https://www.docker.com`.

[8]Docker Hub é o principal repositório de imagens de aplicações disponibilizadas em contêineres `https://hub.docker.com`.

entre VNFs e entre dispositivos IoT e VNFs foram, em média, 0,83 milissegundos para a implementação baseada em máquinas virtuais e 0,53 milissegundos para a implementação em contêineres. Para a taxa de transferência, os resultados mostraram que os contêineres podem suportar até 16,8 Mbit/s para pacotes TCP e aproximadamente 1 Mbit/s para pacotes UDP. Sobre a escalabilidade, a perspectiva baseada em máquinas virtuais apresentou um tempo aproximado de 20 minutos para configurar uma VNF, e de cinco minutos para iniciá-la/pará-la. A perspectiva baseada em contêineres apresentou um tempo aproximado de quatro minutos para configurar uma VNF pela primeira vez, e a partir desse momento, um tempo de nove segundos. Além disso, os experimentos mostram que o tempo para iniciar uma VNF em um contêiner é menor que um segundo. Os autores também apresentam uma avaliação do desempenho do modelo de aprendizagem para a detecção e mitigação de ataques DDoS. Nesse caso, o modelo apresentou uma acurácia de 94,4% e um tempo médio de resposta da VNF para mitigação menor que um segundo.

O trabalho de [Boudi et al. 2019] também apresenta uma abordagem baseada em contêineres para arquiteturas de computação de borda. Nele, os autores realizaram a comparação de desempenho onde o Suricata foi executado em um dispositivo Raspberry Pi 3, que contempla dois cenários diferentes: o primeiro em máquina física (SoBM) e o segundo, em um contêiner Docker (SoDC). Os indicadores de desempenho utilizados na comparação foram: número de pacotes processados, taxa de transferência, consumo de memória RAM e CPU e número de pacotes descartados. Os resultados mostraram que, em relação ao número de pacotes processados, taxa de transferência e consumo de memória RAM, os valores foram muito próximos, podendo considerar um empate entre as duas implementações. No entanto, os indicadores de consumo de CPU e número de pacotes descartados apontaram que a abordagem SoDC possui uma vantagem sobre SoBM, onde o consumo de CPU foi 10% menor e o número de pacotes descartados foi de 4,5% menor.

Em [Guizani and Ghafoor 2020], os autores utilizam das vantagens da escalabilidade providas pela NFV para implementar um IDS baseado em modelos de aprendizagem de máquina para mitigar *malwares* em redes IoT. A solução apresentada é composta por um modelo RNN-LSTM, executando em uma rede centralizada, que detecta os ataques na rede, e cria zonas de vigilância por meio de diferentes NFVs para realizar a aplicação de *patches* de segurança nos dispositivos. Para demonstrar o desempenho da abordagem, os autores realizaram experimentos considerando o modelo epidêmico SEIR, usado na área de epidemiologia. Nesse modelo, há indivíduos suscetíveis (S), expostos (E), infectados (I) e resistentes (R). No caso de [Guizani and Ghafoor 2020], os indivíduos são os dispositivos IoT. A partir da disseminação de um *malware* na rede IoT, o desempenho da abordagem foi aferido de acordo com a escalabilidade da rede, ou seja, a criação de zonas de vigilância em NFV e a quantidade de vírus na rede. Foram consideradas diferentes quantidades de dispositivos na rede, dentre 1000, 10.000 e 100.000. Os resultados mostraram que a criação de zonas de vigilância virtualizadas se mantém constante para 1000 dispositivos na rede. Nesse caso cinco zonas foram criadas. Já o aumento do número de zonas de vigilância, para 50, iniciou-se a partir de 10.000 dispositivos na rede. No entanto, se manteve constante até os 100.000.

Levando em consideração a computação em névoa, [Zhou et al. 2019] propõem a implementação de VNFs para a realização da detecção de ataques DDoS em redes IIoT (do inglês, *Industrial Internet of Things and Services*). A abordagem de detecção de

anomalias proposta pelos autores considera o uso do Snort para implementar máquinas de estado modeladas a partir de protocolos de comunicação TCP e proprietários utilizados em sistemas de controle industrial, nesse caso, o *Modbus*. Os resultados apresentados mostraram que em uma arquitetura que considera o processamento do sistema de detecção próximo aos dispositivos IoT, há um tempo médio de 137 milissegundos e uma taxa de detecção de 93,93%, enquanto que o tempo médio de resposta em uma arquitetura local é de aproximadamente 240 milissegundos e uma taxa de detecção de 84,95%.

A complexidade no gerenciamento de segurança de redes IoT também é uma questão a ser abordada quando há mais de um provedor de serviços na rede. As cidades inteligentes são alguns dos exemplos. Nelas, diversos provedores de computação em nuvem podem disponibilizar diferentes serviços aos vários dispositivos IoT conectados pela cidade. No entanto, a oferta desses serviços deve levar em consideração as características de redes IoT, como a utilização de *gateways* de borda e até mesmo domínios específicos para este tipo de conexão. Por isso, [Massonet et al. 2017] apresentam uma extensão de uma arquitetura de segurança em nuvem federada para redes IoT, baseada na aplicação de políticas de segurança. A segurança em uma arquitetura de nuvem federada é composta por um componente gerenciador central que realiza a gestão das políticas de segurança da rede e as disponibiliza por meio dos demais gerenciadores de nuvens, que são federadas. A proposta apresentada pelos autores considera que a implementação das políticas de segurança de rede, feita por cada provedor de nuvem, seja realizada por meio das cadeias de VNF de segurança e pelo roteamento do tráfego via regras no controlador SDN. No contexto de IoT, a arquitetura proposta se estende para a utilização de *gateways* de borda, onde se realiza a conexão direta com os dispositivos IoT. Além disso, a proposta também considera que o processamento das VNFs de segurança deve ser realizado na borda, devido ao baixo poder computacional e ao possível aumento da latência devido à distância entre o provedor de serviço e os dispositivos. Os autores não avaliaram o desempenho dos recursos propostos.

Em um contexto de soluções domésticas inteligentes, [Al-Shaboti et al. 2018] apresentam uma proposta da utilização de controladores SDN para reforçar o controle de acesso estático e dinâmico à rede IoT. Além disso, apresentam a implementação de uma VNF para mitigar ataques de falsificação ARP (do inglês, *Address Resolution Protocol*). A implementação do servidor ARP foi realizada de duas formas, uma utilizando a biblioteca Scapy[9] (SC), e a outra utilizando o *Data Plane Development Kit* (DPDK). Esse servidor tem como objetivo ser uma entidade confiável que realiza a gestão das requisições ARP na rede. Desta forma, todos os pacotes ARP na rede são filtrados pelo controlador SDN e são encaminhados ao servidor para resolução. Para demonstrar a abordagem de mitigação, os experimentos foram executados utilizando o Mininet[10], de acordo com diferentes quantidades de máquinas presentes na rede, começando em dez e finalizando em 50. Os resultados mostraram que o tempo de resposta da VNF implementada em SC foi de mais de 70 milissegundos para dez dispositivos, e de aproximadamente um segundo para 50 dispositivos. Para a implementação em DPDK, o tempo de resposta

[9]Scapy é uma biblioteca do Python específica para a criação e manipulação de pacotes de redes. `https://scapy.net`.

[10]Mininet é um emulador de redes que possibilita a criação virtual de máquinas, *switches*, controladores e conexões. `http://mininet.org`.

Tabela 3.4. Panorama do desempenho de NFV para o contexto de segurança.

Trabalho	Desempenho avaliado	Segurança	
		Abordagem	Ataque
[Farris et al. 2017]	—	Prevenção	—
[Massonet et al. 2017]	—	Prevenção	—
[Al-Shaboti et al. 2018]	Tempo de resposta ARP	Prevenção e mitigação	MITM
[Zarca et al. 2018]	Tempo de aplicação de políticas de segurança	Prevenção e mitigação	MITM
[Boudi et al. 2019]	Número de pacotes processados, número de pacotes descartados, taxa de transferência, consumo de CPU e de memória RAM	Detecção e mitigação	—
[Sairam et al. 2019]	Armazenamento, consumo de memória RAM, latência, taxa de transferência, escalabilidade, tempo de implantação de ambiente e acurácia do modelo	Detecção e mitigação	DDoS
[Zarca et al. 2019a]	Tempo de detecção e de aplicação de políticas de segurança	Detecção e mitigação	DDoS e *malware*
[Zarca et al. 2019b]	Tempo de aplicação de políticas de segurança, consumo de CPU e de memória RAM	Prevenção e mitigação	MITM
[Zhou et al. 2019]	Tempo de detecção	Detecção e mitigação	DDoS
[Afek et al. 2020]	—	Prevenção	—
[Bagaa et al. 2020]	Acurácia do modelo	Detecção	DDoS, Probing Attack, U2R e R2L
[Guizani and Ghafoor 2020]	Escalabilidade	Detecção e mitigação	*Malware*
[Zarca et al. 2020b]	Tempo de implantação de ambiente e de aplicação de políticas de segurança	Detecção e mitigação	DDoS

médio foi de 0,57 milissegundo. Em relação à mitigação do ataque de falsificação ARP, os autores apresentam apenas a taxa de 100% de descarte para os pacotes maliciosos sem contabilizar o tempo de execução do descarte.

A descrição de um dispositivo IoT e seu comportamento na rede podem ser utilizados para a criação de melhorias em mecanismos de segurança. Dessa forma, a [IETF 2020] orienta os fabricantes de dispositivos IoT e de sistemas de comunicação sobre a implementação da MUD (do inglês, *Manufacturer Usage Description*). Armazenadas em arquivos, as MUDs consistem em listas de permissões que descrevem domínios e *endpoints* para cada tipo de dispositivo na rede. Assim, logo ao conectar, os dispositivos IoT solicitam a MUD ao servidor dedicado a esse propósito para, então, se integrar com os demais componentes e serviços.

Nesse sentido, o trabalho de [Afek et al. 2020] apresenta uma proposta de uti-

Tabela 3.5. Panorama do desempenho de NFV para o contexto de IoT.

Trabalho	Desempenho avaliado	IoT		
		Protocolo	Arquitetura	Dispositivo
[Farris et al. 2017]	—	—	—	—
[Massonet et al. 2017]	—	—	Névoa	Dispositivos em geral
[Al-Shaboti et al. 2018]	Tempo de resposta ARP	—	—	Dispositivos em geral
[Zarca et al. 2018]	Tempo de aplicação de políticas de segurança	6LoWPAN	—	Dispositivos em geral
[Boudi et al. 2019]	Número de pacotes processados, número de pacotes descartados, taxa de transferência, consumo de CPU e de memória RAM	—	Névoa	Raspberry Pi 3
[Sairam et al. 2019]	Armazenamento, consumo de memória RAM, latência, taxa de transferência, escalabilidade, tempo de implantação de ambiente e acurácia do modelo	—	Névoa	Raspberry Pi 3 e câmeras IP
[Zarca et al. 2019a]	Tempo de detecção e de aplicação de políticas de segurança	CoAP e 6LoWPAN	Névoa	Dispositivos em geral
[Zarca et al. 2019b]	Tempo de aplicação de políticas de segurança, consumo de CPU e de memória RAM	CoAP e 6LoWPAN	Névoa	Dispositivos em geral
[Zhou et al. 2019]	Tempo de detecção	TCP e Modbus	Névoa	Dispositivos em geral
[Afek et al. 2020]	—	—	—	Dispositivos que suportam MUD
[Bagaa et al. 2020]	Acurácia do modelo	—	—	Sensores de temperatura e CO2
[Guizani and Ghafoor 2020]	Escalabilidade	—	—	Dispositivos em geral
[Zarca et al. 2020b]	Tempo de implantação de ambiente e de aplicação de políticas de segurança	CoAP, RPL e LoWPANs	—	Sensores de umidade, temperatura, luz, CO2 e RFID

lização de VNFs em nível de ISP para a aplicação de políticas de segurança em redes

domésticas em larga escala baseadas em MUD. O objetivo da proposta é garantir que os pacotes que trafegam na rede de ou para um dispositivo IoT sejam analisados e confrontados com suas regras contidas no arquivo MUD. Este mecanismo consiste em dois componentes, o de monitoramento (WLM – *whitelist monitoring*) e o de aplicação (WLE – *whitelist/MUD enforcement*). O componente WLM é responsável por analisar se os pacotes estão ou não de acordo com as regras. O componente WLE recebe as informações da WLM e aplica as políticas de exclusão ou de manutenção da conexão. A proposta dos autores é de que o WLM seja desenvolvido como uma VNF, sendo solicitada pelo WLE que está no caminho principal dos pacotes ou roteadores. Nesse trabalho, os autores apresentam uma prova de conceito. Porém, não analisaram o desempenho dos componentes nem do mecanismo implementados.

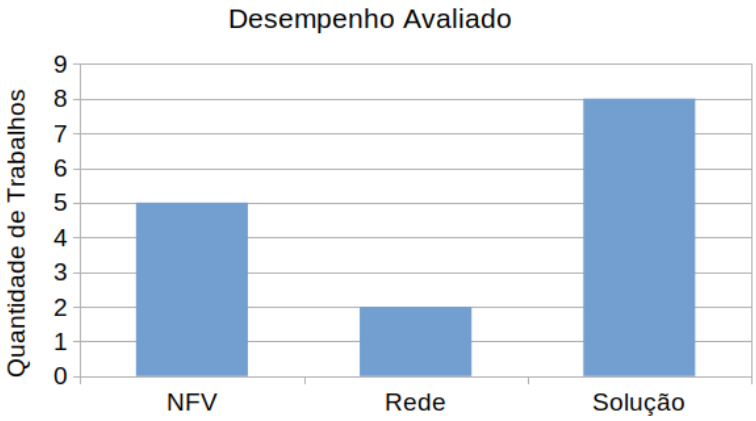
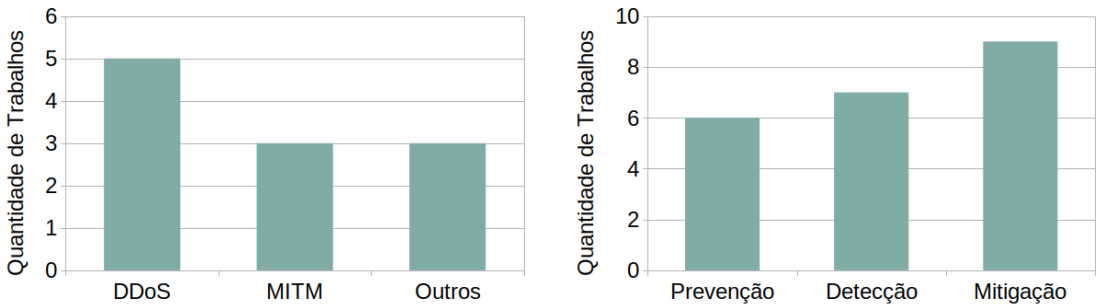


Figura 3.11. Métricas utilizadas na avaliação de desempenho.



(a) Ataque (b) Abordagem

Figura 3.12. Tipos de ataques considerados e abordagens apresentadas.

A Tabela 3.4 traz um panorama do desempenho de NFV para o contexto de segurança e a Figura 3.11 contabiliza a quantidade de trabalhos e os tipos de métricas utilizadas na avaliação de desempenho dos recursos virtualizados. É possível observar que as métricas relacionadas aos componentes implementados por meio da virtualização (NFV), como consumo de CPU, memória RAM, armazenamento, escalabilidade e tempo de implantação de ambiente, estão presentes em 50% dos trabalhos. Já as métricas relacionadas à rede (Rede), como latência, taxa de transferência e número de pacotes

processados/descartados, estão em 20% dos trabalhos. As que mais se destacam são as métricas que apresentam a eficácia da solução proposta (Solução), como tempo de resposta, de aplicação de políticas, de detecção e mitigação de ataques e acurácia do modelo de aprendizado de máquina, que estão em 80% dos trabalhos. A Figura 3.12 contabiliza os ataques e as abordagens apresentadas nos trabalhos analisados. O ataque mais utilizado para demonstrar o uso da NFV é o DDoS (55,55%), a frente do MITM (33,33%) e os demais (33,33%). Além disso, também é possível observar que 46,15% dos trabalhos levam em consideração a NFV para a prevenção de ataques na rede IoT, 53,85% para detecção e 69,23% para mitigação. Para o contexto de IoT, a Tabela 3.5 mostra que 46,15% dos trabalhos abordam as características de computação em névoa para a aplicação de NFV. No entanto, apenas [Zhou et al. 2019] apresentam uma comparação entre o desempenho de detecção e mitigação para arquiteturas distintas.

3.4. Estudo de Caso: MENTORED Testbed

O principal objetivo desta Seção é divulgar os esforços que estão sendo realizados no escopo do projeto MCTI/FAPESP MENTORED[11] em direção à criação de um ambiente controlado para experimentação em cibersegurança e introduzir a utilização de ferramentas de experimentação para avaliar soluções de redes como aquelas baseadas em NFV que foram apresentadas nas seções anteriores deste capítulo. Para este fim, apresenta-se um estudo de caso focado em uma ferramenta para detecção de *botnets* baseada em uma NFV de monitoramento de tráfego. Neste estudo de caso, empregaram-se ferramentas que permitem a reprodução do experimento conduzido pelo público. Assim, a Subseção 3.4.1 detalha o *testbed* em desenvolvimento e a Subseção 3.4.2 ilustra o emprego de VNFs em um cenário experimental.

3.4.1. MENTORED Testbed

O projeto MENTORED tem como objetivos identificar, modelar e avaliar comportamentos maliciosos associados à IoT de forma a auxiliar na construção de soluções avançadas para possibilitar: prevenção, predição, detecção e mitigação de ataques DDoS. O projeto se divide em quatro pacotes de trabalho (WPs – *Work Packages*), sendo os dois primeiros pacotes de trabalho (WP1 e WP2) dedicados à prevenção e à predição de *botnets* e ataques DDoS, respectivamente; o terceiro pacote (WP3) é dedicado à detecção e à mitigação de ataques DDoS; e o quarto pacote (WP4) foca em criar um ambiente, denominado MENTORED *Testbed*, para a experimentação de sistemas de cibersegurança.

O MENTORED *Testbed* é um ambiente acadêmico experimental, em desenvolvimento, que oferece aos pesquisadores a possibilidade de demonstrar a viabilidade de suas soluções de cibersegurança em cenários de rede com escala realista. Este ambiente vem sendo definido tomando como base a Infraestrutura Definida por *Software* da RNP (IDS-RNP) e as demandas dos outros pacotes de trabalho do projeto. A infraestrutura IDS-RNP se baseia na plataforma de orquestração de contêineres Kubernetes[12] e funciona como um ambiente externo para o desenvolvimento de projetos de rede e nuvem.

[11] `https://mentoredproject.org/`

[12] Kubernetes é uma plataforma de orquestração de contêineres que automatiza a implantação, o dimensionamento e a gestão de aplicações em contêineres. `https://kubernetes.io`.

A Figura 3.13 apresenta a visão geral dos servidores dispostos fisicamente em 13 Pontos de Presença (do inglês, *Point of Presence* - PoP) da RNP, sendo cinco servidores localizados nas regiões norte e nordeste e oito localizados nas regiões sul, sudeste e centro-oeste. Os circuitos de rede do nordeste e uma conexão com o sudeste possuem 100 Gbit/s. Os demais circuitos possuem entre 10 Gbit/s e 40 Gbit/s de capacidade. Cada nó do IDS-RNP possui um *switch* gerenciável e um servidor de virtualização equipados minimamente com dois processadores Intel® Xeon® E5-2630, 64GB de memória RAM e disco com 4TB de armazenamento.

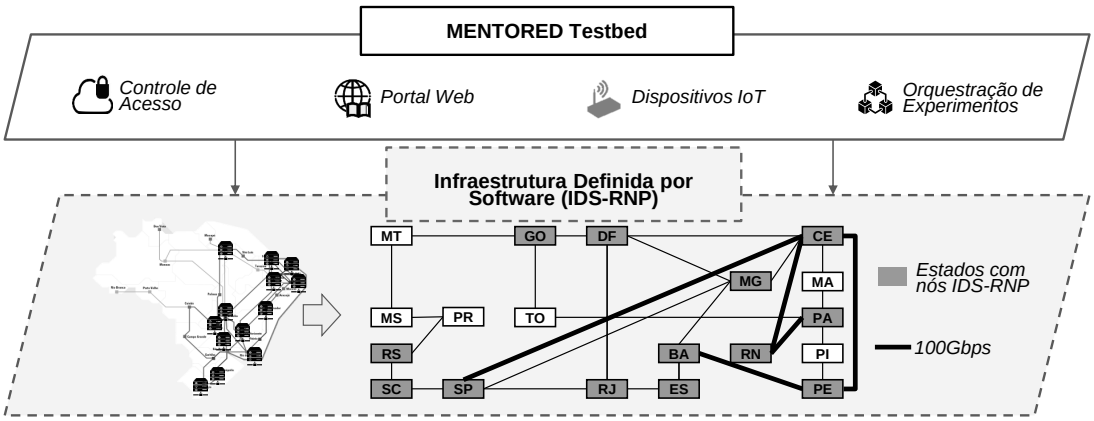


Figura 3.13. Visão geral do ambiente.

O WP4 do projeto MENTORED tem como objetivo a implantação de um ambiente usando tecnologias e metodologias avançadas para pesquisa experimental sob demanda em cibersegurança com foco em cenários realísticos da IoT. Portanto, a equipe do projeto realizou um levantamento de requisitos considerando os requisitos do *testbed* de cibersegurança DETERLab [Benzel 2011], do *testbed* de IoT FIT-IoT Lab [Adjih et al. 2015] e as definições apresentadas pelos pesquisadores dos outros pacotes de trabalho do projeto. O grupo escolheu analisar estes *testbeds*, pois estes são pioneiros em experimentação em redes com acesso aos dispositivos físicos e porque possibilitam que o próprio usuário defina o comportamento dos agentes em cada cenário experimental, diferentemente dos demais *testbeds*, que emulam os dispositivos ou são baseados em aplicações específicas da IoT. Assim, os seguintes requisitos para o MENTORED *Testbed* foram definidos:

- **Fidelidade**: se refere à capacidade de obter precisão suficiente na reprodução dos fenômenos específicos, sendo estudados em um experimento particular, dentro de um ambiente que pode ou não corresponder a qualquer rede real existente. Assim, além da representação física de uma rede real, o *testbed* pode também considerar e empregar modelos acurados para realizar avaliações fiéis;

- **Validade**: as limitações do próprio *testbed* não devem distorcer acidentalmente os resultados de um experimento. O *testbed* deve identificar e denunciar as violações dessas condições experimentais exigidas, alertando o usuário para possíveis falhas de validade do experimento;

- **Escalabilidade**: o *testbed* deve ser capaz de prover suporte a experimentos representativos para capturar efeitos complexos dos ataques relacionados ao tráfego maciço de dados na escala da Internet;
- **Segurança**: exige que o *testbed* garanta que nenhum código ou usuários maliciosos obtenham o acesso indevido ou prejudiquem outras infraestruturas de rede, informações ou códigos do próprio *testbed* ou da Internet em geral;
- **Reprodutibilidade**: garante que um experimento, uma vez executado, possa ser exportado e, em seguida, executado em um ambiente idêntico em um momento posterior, para produzir resultados idênticos;
- **Transparência**: o *testbed* deve possibilitar o monitoramento em tempo real e não intrusivo do tráfego de rede e dos recursos computacionais, além de empregar ferramentas de visualização destes recursos tanto de forma gráfica quanto pela linha de comandos;
- **Perspectiva centrada no usuário**: o *testbed* deve oferecer liberdade para os usuários desenvolverem novas classes de ferramentas que facilitam as pesquisas experimentais, além das funções tradicionais da pesquisa experimental, como a configuração de experimentos e o monitoramento do tráfego;
- **Acesso em Tempo Real**: o *software* de orquestração deve fornecer acesso em tempo real aos dispositivos, para que um usuário possa redefinir, reprogramar e monitorar o estado de cada dispositivo durante a execução dos experimentos.

Como mencionado, o MENTORED *Testbed* vem sendo construído sobre a infraestrutura IDS-RNP, a qual está fundamentada na plataforma de orquestração de contêineres Kubernetes. A plataforma Kubernetes gerencia o ciclo de vida de aplicativos e serviços em contêineres, usando métodos que fornecem monitoramento, escalabilidade e alta disponibilidade. Cada contêiner é uma área isolada dentro do sistema operacional e possui seu próprio sistema de arquivos, compartilhamento de CPU, memória e espaço de processo para executar serviços ou aplicações individualmente. Estes contêineres são logicamente alocados em *pods*, conjuntos de contêineres que compartilham a mesma área do sistema operacional e de rede. Um conjunto de *pods* se comunica por meio de uma rede e forma um *cluster*. Os servidores físicos (*Workers*) executam o sistema operacional, os *pods* e os módulos de controle. A plataforma Kubernetes emprega um módulo *Master* com uma API (do inglês, *Application Programming Interface*) que administra a comunicação de rede entre os *Workers* para criar *clusters* em múltiplos *Workers* remotos. O Kubernetes *Master* recebe *scripts* de configuração e controla localmente cada *Worker* para executar serviços e aplicações de forma portátil em *clusters*.

Os *Workers* executam três módulos locais básicos: o *Kubelet*, *Proxy* e de *Networking*. O *Kubelet* é um agente local da plataforma Kubernetes para administrar os *pods* hospedados localmente em cada *Worker*. O módulo de *Proxy* administra as regras locais de rede, intermediando as conexões e controlando os acessos. O módulo de *Networking* distribui as regras para cada *Proxy* e estabelece a comunicação entre os contêineres. Então, um usuário da plataforma Kubernetes pode definir como seus aplicativos devem ser

executados e como eles devem interagir com outros aplicativos ou com a Internet. Estas características permitem o desenvolvimento modular do *testbed*. Além disso, a plataforma Kubernetes compreende primitivas de *software* livre, para o desenvolvimento de novas versões capazes de cumprir os requisitos estudados sobre redes heterogêneas de longa distância.

O IDS-RNP auxilia na concepção do MENTORED *Testbed* ao fornecer esta estrutura de nível nacional e a equipe técnica que auxilia na implementação dos novos requisitos. Esta infraestrutura permite criar experimentos em escala com redes de longa distância, recursos computacionais e *softwares* bem conceituados tanto na academia quanto no mercado. O MENTORED *Testbed* utiliza estas características do IDS-RNP para configurar automaticamente experimentos e para testar mecanismos de cibersegurança, com cenários de rede realísticos, escaláveis e com segurança suficiente para não prejudicar a disponibilidade da infraestrutura. A Figura 3.14 ilustra a arquitetura do MENTORED *Testbed*, composta por três componentes principais: as *Ilhas MENTORED*, as *Redes Virtuais de Controle e Teste* e o *MENTORED Master*.

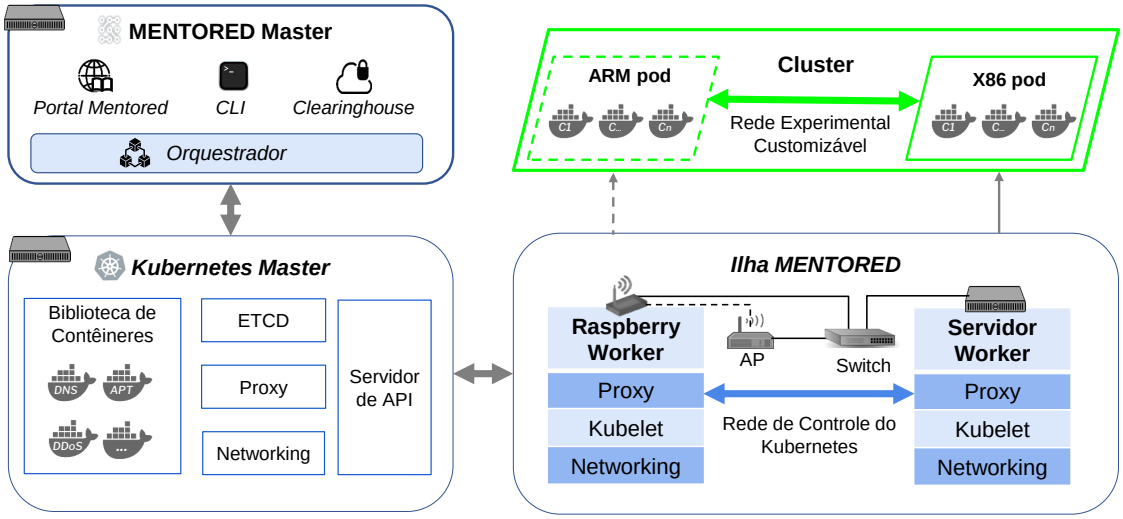


Figura 3.14. Arquitetura do MENTORED *Testbed*.

As **Ilhas MENTORED** constituem-se de ilhas do IDS-RNP que foram acrescidas de um Ponto de Acesso sem fio (AP – *Access Point*) e um conjunto de placas Raspberry PI 4 para formarem uma rede local sem fio. Esses novos dispositivos poderão ser usados para aumentar a heterogeneidade de equipamentos (ARM e X86) e meios de transmissão e, com isso, prover a fidelidade do *testbed* ao criar redes IoT. Para suportar tal heterogeneidade, utiliza-se a plataforma Kubespray[13], um módulo Kubernetes que administra *clusters* multi-arquiteturais. As placas Raspberry PI 4 também possuem conexão de rede cabeada, porém, a mesma só é usada para o tráfego de controle e gerência dos dispositivos. Ou seja, o tráfego dos experimentos só trafega pela interface de rede sem fio.

As **Redes Virtuais de Controle do Kubernetes e Experimental Customizável** servem para trafegar a comunicação entre os componentes administrativos, responsáveis

[13] `https://kubespray.io`.

pela configuração dos componentes do *testbed* e a carga de trabalho gerada pelos experimentos, respectivamente. Estas redes podem ser configuradas por diferentes módulos baseados nas CNIs (do inglês, *Conteiners Networks Interfaces*) do Kubernetes. Diversas CNIs podem ser implantadas em conjunto e permitem que o *testbed* defina os melhores módulos de rede ou que seja desenvolvido um próprio conforme as especificidades do projeto. Além disso, permite-se a criação de experimentos em redes com tecnologias heterogêneas.

O **MENTORED Master** é uma camada de *software* sobre a API do Kubernetes que administra a interação entre os quatro módulos de *software*. Ele controla os demais componentes do *testbed*: o *Portal MENTORED*, a *CH (ClearingHouse)*, a *Interface de Linha de Comandos (do inglês, Command-Line Interface - CLI*) e o *Orquestrador*. O *Portal MENTORED* e a *CH* lidam com a interação com o usuário, gestão de identidade e controle de acesso mediante uma interface gráfica para o usuário (GUI – *Graphical User Interface*). A CLI fornece comandos específicos para utilizar o orquestrador *MENTORED* e controlar os componentes do *testbed*. O *Portal MENTORED* será um site para os usuários se cadastrarem e, por meio da *CH*, adquirirem as credenciais necessárias para estabelecerem uma conexão remota e segura com a CLI. O Orquestrador faz a interface entre os demais módulos e a API do Kubernetes para fornecer as funções específicas de criação de experimentos, controle e monitoramento dos recursos. A API do Orquestrador simplifica a configuração dos cenários de rede e emprega as ferramentas necessárias automaticamente para cumprir os requisitos definidos. Assim, o usuário autenticado reserva os recursos de hardware, configura os contêineres a serem executados e, por fim, cria e submete um *script* de descrição do experimento pretendido conforme os comandos específicos da API. O Orquestrador interpreta o *script* e configura automaticamente a infraestrutura com base no Kubernetes para disponibilizar os recursos necessários e alocar os módulos para cada *namespace*.

Este conjunto de componentes é coordenado a fim de permitir que o MENTORED *Testbed* atinja os requisitos do ambiente experimental e aloque serviços pré-estabelecidos em uma biblioteca de contêineres. Então, caso um usuário precise de qualquer NFV para executar experimentos, o *testbed* possui um contêiner com um conjunto de configurações prontas para executar avaliações sobre este tipo de serviço na biblioteca. O requisito *Perspectiva Centrada no Usuário* compreende que as metodologias de pesquisa experimental são, elas mesmas, uma área de pesquisa ativa e os avanços nesta área permitem a criação de novas metodologias de pesquisa [Benzel 2011]. Neste sentido, tanto os pesquisadores do WP4, quanto os usuários gerais podem criar conjuntos de contêineres com experimentos configurados para colaborar com novas metodologias de pesquisa e com a biblioteca de contêineres.

3.4.2. Estudo de Caso

Esta Subseção detalha um estudo de caso projetado para uso no ambiente de experimentação, tendo como objetivo geral apresentar à comunidade acadêmica uma introdução na criação de cenários experimentais de redes e atrair novos pesquisadores para o desenvolvimento de metodologias de pesquisa experimentais em cibersegurança. O estudo de caso compreende operações simples de rede, que envolvem uma VNF de análise do tráfego de rede e detecção de *bots* (*i.e.*, dispositivos infectados na rede). Tanto o MENTORED *Test-*

bed quanto a IDS-RNP estão em desenvolvimento e ainda não possuem suporte para os usuários finais. Portanto, este estudo de caso é descrito de uma forma que um usuário comum possa simular um *cluster* semelhante ao do IDS-RNP e reproduzir os experimentos localmente. Os arquivos de apoio estarão disponíveis no github do projeto[14].

Avaliar propostas de detecção de *bots* de forma eficiente exige que o *testbed* possua suporte a diferentes topologias, protocolos e aplicações. Os trabalhos de detecção de intrusão geralmente monitoram pontos estratégicos do tráfego e enviam sinais customizados para a ferramenta de defesa tomar as providências necessárias. Isto ocorre principalmente quando consideram-se as redes de dispositivos IoT, que possuem desafios relacionados à heterogeneidade e ao volume de dados. O estudo de caso compreende avaliar uma ferramenta de detecção em um cenário de rede customizável e fácil de configurar.

A plataforma Kubernetes oferece um conjunto de procedimentos para o desenvolvimento de módulos que agregam funcionalidades na orquestração de contêineres. Neste contexto, os cenários criados para desenvolver este estudo de caso utilizam estes módulos de CNI para organizar a comunicação de rede entre os contêineres. A principal ferramenta considerada neste estudo para desenvolver os cenários de rede é o Knetlab[15], uma CNI desenvolvida pela equipe da RNP para simular topologias de rede entre os *pods* do Kubernetes. Além disso, a ferramenta Kind[16] monta o *cluster* Kubernetes com os servidores *Workers* e o Kubernetes *Master* conteinerizados localmente.

A Figura 3.15 ilustra a arquitetura do cenário de experimentação. Toda esta estrutura será executada sobre um computador *host*. A ferramenta Kind simula, com contêineres Docker, um *cluster* Kubernetes com múltiplos *Workers* e *Masters*. Cada *Worker* representa uma Ilha do IDS-RNP e recebe uma etiqueta de identificação baseada nas localizações do país. O Kubernetes *Master* instala os módulos de CNI como o Knetlab, o Multus e o OpenVSwitch (OVS) nos *Workers* distribuídos. O Knetlab emprega outras CNIs, como o Multus, que cria múltiplas interfaces de rede em cada *pod* e as regras OVSDB (do inglês, *Open vSwitch Database Management Protocol*), que implementam *switches* virtuais e criam *bridges* de VLANs (do inglês, *Virtual Local Area Network*) entre os *pods* ou as VXLANs (do inglês, *Virtual Extensible LAN*) entre contêineres *Masters* e *Workers*.

Esta estrutura permite criar *pods* que se comunicam através de uma rede controlada. Eles implementam VNFs sobre qualquer camada de rede. Cada *pod* na topologia customizada representa um dispositivo em rede e executa funções específicas para o experimento. Neste estudo de caso, considera-se que a rede possui duas classes de dispositivos, os dispositivos de borda e os de rede. Os dispositivos de borda atuam gerando três modelos de tráfego: normal, atacante e de servidor alvo. Os dispositivos de rede atuarão como OpenVSwitch, para o encaminhamento de tráfego, ou como uma *middlebox*, executando a VNF que pretendemos avaliar. A VNF classifica aqueles dispositivos que estiverem gerando tráfego atacante e executa o comando para bloquear este tráfego.

O Knetlab emprega o conceito de laboratórios para executar diferentes experimentos de forma isolada. Cada laboratório recebe um *namespace* e uma série de *scripts yaml*

[14] `https://github.com/mentoredtestbed`.
[15] `https://git.rnp.br/cnar/knetlab/knetlab-doc`.
[16] `https://kind.sigs.k8s.io`.

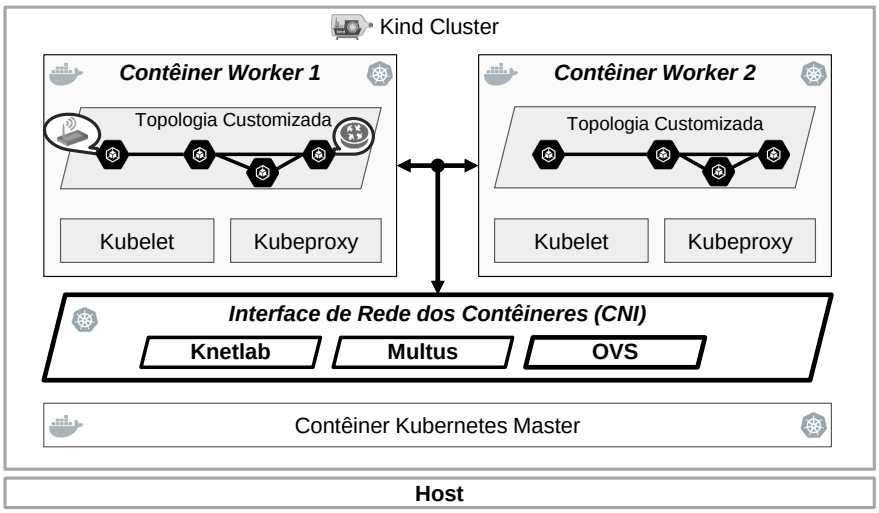


Figura 3.15. Arquitetura do experimento.

com as descrições de cada tipo de dispositivo, as configurações específicas de cada *pod* e a topologia de rede. Neste estudo de caso, o cenário inteiro pode ser criado através da execução de um *script* de inicialização, desde que no *host* esteja instalada a ferramenta de comandos da plataforma Kubernetes (*kubectl*), o Kind e o Docker.

O cenário criado automaticamente apenas inicializa as interfaces, os dispositivos e as conexões entre os *pods*. A topologia considerada neste estudo está ilustrada na Figura 3.16. Cada *Worker* pode simular uma topologia inteira sozinho, no entanto, esta topologia mostra que o Knetlab pode também usar as conexões de longa distância do IDS-RNP e suportar diferentes serviços e protocolos de rede entre as rotas virtuais.

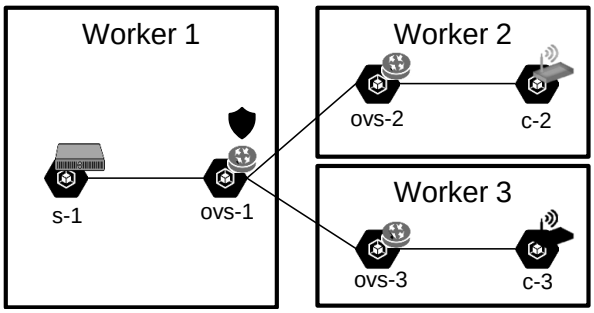


Figura 3.16. Topologia do estudo de caso.

Os *pods* que desempenham o papel de dispositivos de rede (OVS-1, OVS-2 e OVS-3) possuem um terminal linux e o OVS instalados por padrão. Os dispositivos de borda executam os comandos conforme as pré-configurações definidas para cada modelo. Portanto, os usuários podem desenvolver os experimentos como se fossem uma estrutura de rede real apenas com os cabos de rede conectando os dispositivos. Neste contexto, os *pods* contidos na topologia executam os serviços para desempenhar as funções de rede. Os *pods* podem executar contêineres específicos disponibilizados no repositório público do MENTORED *Testbed*, com exemplos práticos que executam aplicações pré-configuradas.

O *script* de configuração dos *pods* insere os recursos necessários para a execução do estudo de caso. Os *pods* que executam os dispositivos de rede recebem e instalam os *scripts* de configuração das VLANs. Os *pods* de borda executam contêineres com um servidor *web* nginx, um *script* gerador de tráfego no modo cliente e outro gerador de tráfego no modo ataque. O *switch* OVS-1 também executará o *script* representando uma VNF que monitora o tráfego a fim de detectar *botnets* e bloqueia, com um comando simples, os IPs supostamente infectados. O *script* extrai informações dos cabeçalhos dos pacotes de rede, realiza uma análise simples para determinar qual o IP responsável por gerar o tráfego atacante e envia um sinal ao OVS para bloqueá-lo - lembrando que o objetivo maior deste estudo de caso é demonstrar a metodologia de experimentação empregada para avaliar a VNF e não a solução de detecção em si.

Os *pods* também rodam um serviço agente para determinar quando o experimento será inicializado. O serviço agente envia sinais através da rede administrativa, que é configurada por padrão pela plataforma Kubernetes na interface `eth0`. Desta forma, o serviço recebe o sinal para começar e determina um horário para os demais *pods* executarem as funções especificadas e salvarem os resultados obtidos, ou seja, se a ferramenta de detecção foi eficiente e se o bloqueio da entrada do tráfego aliviou a carga no servidor.

Este cenário permite a execução de experimentos com topologias específicas e emprega VNFs em diferentes contextos de rede. Assim, cada usuário deve implementar os próprios dispositivos, topologias e serviços. O MENTORED *Testbed* visa simplificar a criação destes experimentos, alimentando uma biblioteca de cenários, que podem ser criados automaticamente e reconfigurados conforme as demandas do usuário, como foi apresentado. Desta forma, os usuários gastam menos tempo com as configurações complexas dos cenários realísticos de rede e ficam mais tempo focados no problema de pesquisa em si.

3.5. Principais Desafios e Limitações

Aprimorar os mecanismos de segurança levando em consideração a heterogeneidade de dispositivos IoT é um grande desafio a ser encarado. Dessa forma, a NFV tem se mostrado uma alternativa viável para auxiliar no cumprimento dos requisitos de segurança em redes IoT. No entanto, existem desafios e limitações em aberto que precisam de atenção para que a aplicação dessa tecnologia seja cada vez mais abrangente e eficaz.

Como visto na Seção 3.3, a definição e a aplicação de políticas de segurança na IoT têm sido uma das abordagens suportadas pela NFV. Expressar requisitos de segurança para controlar sistemas IoT distribuídos representa uma tarefa desafiadora, especialmente quando diferentes domínios administrativos e tecnológicos estão envolvidos. Isso significa que redes IoT possuem características dinâmicas que exigem que as políticas de segurança sejam estendidas, incluindo informações de contextos dinâmicos de rede. Assim, as políticas de segurança também podem depender das interações entre objetos inteligentes localizados no mesmo ambiente. Todos esses aspectos exigem capacitar os modelos de política para incluir as peculiaridades dos sistemas de IoT, bem como o potencial de novos mecanismos de segurança orientados por *software*.

A flexibilidade na seleção de mecanismos virtualizados de segurança pode afetar diretamente a qualidade dos serviços da rede, especialmente quando aplicações IoT com

requisitos rigorosos em termos de latência e confiabilidade são consideradas. Nessa linha, surgem desafios, como: quais serviços de segurança implementar, onde colocá-los e como configurá-los, resultando em um problema complexo de otimização para determinar a melhor alocação de serviços virtualizados. Além disso, três fatores podem aumentar ainda mais a complexidade da decisão. O primeiro é relacionado ao ambiente *multicloud*, ou seja, um ambiente composto por mais de um provedor de nuvem, pública ou privada, como Amazon AWS, Google GCP, Red Hat OpenStack/OpenShift, entre outros. Nesse tipo de ambiente, diversas VNFs disponibilizadas por dois ou mais provedores de nuvem podem ser selecionadas para a composição da SFC, o que aumenta as variáveis de decisão do orquestrador de VNFs. O segundo é sobre a execução em tempo real do otimizador de seleção das VNFs, uma vez que a carga de trabalho atual da rede deve ser considerada na reconfiguração dos mecanismos de segurança, especialmente para os recursos dinâmicos dos sistemas IoT. O terceiro fator são os requisitos de segurança que englobam o tempo de reação a ameaças e os custos envolvidos para a implementação de seus mecanismos.

Além da baixa latência e do aumento na largura de banda da rede, a tecnologia 5G também traz um recurso chamado de fatiamento da rede ou *network slicing* para atender, principalmente, os aspectos de redes IoT. Essa abordagem permite a criação de recursos personalizados de redes de acordo com requisitos de funcionalidade, isolamento, desempenho, entre outros. Em um contexto de NFV, para aprimorar a segurança em redes IoT, o fatiamento da rede pode ser implementado de acordo com o encadeamento de funções, levando em consideração a compensação entre a flexibilidade, desempenho e custo. O desafio aqui está relacionado à granularidade da fatia, que contém os recursos de rede, podendo ser fornecida por domínio de negócio, fluxo de tráfego, tipos de dispositivos e, até mesmo, por dispositivo unitário. Além disso, outro desafio nesse contexto é criar e gerenciar de forma dinâmica as fatias de recursos de rede.

Do ponto de vista de segurança inerente às estruturas lógicas da NFV ETSI, é possível citar o desafio do isolamento dos ambientes virtualizados sobre uma mesma infraestrutura física. Por exemplo, uma falha não corrigida de isolamento no *hypervisor* pode ser explorada por um superusuário do ambiente (*root*), possibilitando seu acesso às VNFs e explorando seus dados privados. Esse desafio pode ser ampliado quando o provisionamento de VNFs for realizado por uma entidade diferente da provedora de NFVi, por exemplo, provedores de computação em nuvem distintos. Além disso, o gerenciamento das imagens que são utilizadas para implementar as VNFs deve ser observado de forma cuidadosa. As imagens contêm informações de configuração das VNFs que podem ser interceptadas por um atacante e redefinidas para que se comportem de acordo com suas necessidades. Outro desafio é a composição de políticas de autenticação e autorização para a utilização de VNFs e suas cadeias por domínios de redes distintos e isolados, responsabilidade do componente de gerenciamento e orquestração NFV MANO (do inglês, *Network Function Virtualization Management and Orchestration*).

3.6. Considerações Finais

A introdução de recursos de rede virtualizados no ecossistema IoT traz diversas vantagens que aumentam o valor agregado da rede, principalmente pelo desacoplamento de funções de segurança de dispositivos com capacidades limitadas. Além disso, a escalabilidade é um dos aspectos mais fortes da virtualização quando comparada a recursos estáticos

em uma rede tradicional, permitindo, assim, a alocação dinâmica de funções de rede de acordo com o contexto requisitado, como o volume de tráfego, quantidade de dispositivos, ataques, entre outros. No entanto, o comportamento de uma tecnologia nova em um ambiente de redes deve ser observado para que seu desempenho não interfira de forma negativa na qualidade dos serviços prestados aos usuários finais. Este capítulo apresentou questões e o estado da arte relacionados ao uso de NFV na detecção e mitigação de ataques em redes IoT. Foram apresentados os conceitos iniciais de IoT, características relacionadas à NFV, questões de desempenho quando aplicadas à segurança de IoT, o estado da arte e um estudo de caso implementado no MENTORED *Testbed*, além dos desafios e limitações que essa tecnologia apresenta.

A análise da literatura, Seção 3.3, permite concluir que os indicadores designados para avaliar o desempenho da virtualização de NF são menos explorados que os indicadores utilizados para demonstrar a eficácia dos métodos propostos pelos diversos autores. Um exemplo claro disso é o tempo aferido para a implantação de uma ou mais máquinas virtuais que é encontrado apenas em dois dos trabalhos que analisam o desempenho da abordagem proposta. Outro ponto importante é que o desempenho da rede também é pouco explorado, mesmo em trabalhos que consideram a implementação de uma arquitetura em névoa. O desempenho da rede é observado pelos autores em apenas dois dos trabalhos analisados. Avaliar o comportamento de recursos virtualizados por meio dessas métricas é essencial, pois impacta diretamente no gerenciamento de desempenho da rede e se torna ainda mais importante quando funções de segurança são implementadas por meio de NFV.

Ademais, a NFV tem suportado a criação customizada de ambientes para a realização de simulações de ataques em redes IoT, como mostra o MENTORED *Testbed*. O estudo de caso apresentado na Seção 3.4 demonstrou na prática operações, que envolvem VNFs, em diferentes contextos de redes. Isto não só possibilita a simplificação da criação de experimentos em redes IoT, mas também a redução de custos relacionados com a implantação de ambientes dinâmicos e complexos.

Por fim, gerenciar o desempenho da rede torna-se uma atividade ainda mais complexa quando desafios como a aplicação de políticas de segurança de forma dinâmica, a seleção de serviços e onde implementá-los, a granularidade do fatiamento da rede e questões inerentes às estruturas de provisionamento de NFV são incluídos na rede IoT. Isso faz com que pesquisas relacionadas a esses contextos tendam a aumentar no decorrer do tempo, fazendo com que a virtualização de funções de rede alcance níveis de aplicabilidade cada vez mais abrangentes.

Agradecimentos

O presente trabalho foi realizado com apoio do Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) e da Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP). Agradecemos as agências de pesquisa CNPq pelo auxílio financeiro 130762/2021-0 e a FAPESP pelos processos 18/23098-0, 14/50937-1 e 18/22979-2.

Referências

[Adjih et al. 2015] Adjih, C., Baccelli, E., Fleury, E., Harter, G., Mitton, N., Noel, T., Pissard-Gibollet, R., Saint-Marcel, F., Schreiner, G., Vandaele, J., and Watteyne, T. (2015). FIT IoT-LAB: A Large Scale Open Experimental IoT Testbed. In *Anais do IEEE World Forum on Internet of Things*, Milan, Italy.

[Afek et al. 2020] Afek, Y., Bremler-Barr, A., Hay, D., Goldschmidt, R., Shafir, L., Avraham, G., and Shalev, A. (2020). NFV-based IoT Security for Home Networks using MUD. In *Anais do NOMS 2020 - 2020 IEEE/IFIP Network Operations and Management Symposium*, pages 1–9.

[Al-Shaboti et al. 2018] Al-Shaboti, M., Welch, I., Chen, A., and Mahmood, M. A. (2018). Towards Secure Smart Home IoT: Manufacturer and User Network Access Control Framework. In *Anais da 2018 IEEE 32nd International Conference on Advanced Information Networking and Applications (AINA)*, pages 892–899.

[Alam et al. 2020] Alam, I., Sharif, K., Li, F., Latif, Z., Karim, M. M., Biswas, S., Nour, B., and Wang, Y. (2020). A Survey of Network Virtualization Techniques for Internet of Things Using SDN and NFV. *ACM Computing Surveys*, 53(2):1–40.

[Bagaa et al. 2020] Bagaa, M., Taleb, T., Bernabe, J. B., and Skarmeta, A. (2020). A Machine Learning Security Framework for Iot Systems. *IEEE Access*, 8:114066–114077.

[Batista et al. 2016] Batista, D. M., Goldman, A., Hirata, R., Kon, F., Costa, F. M., and Endler, M. (2016). InterSCity: Addressing Future Internet research challenges for Smart Cities. In *Anais da 7th International Conference on the Network of the Future (NOF)*, pages 1–6.

[Benzel 2011] Benzel, T. (2011). The Science of Cyber Security Experimentation: The DETER Project. In *Anais da 27th Annual Computer Security Applications Conference*, ACSAC '11, page 137–148, New York, NY, USA. Association for Computing Machinery.

[Boudi et al. 2019] Boudi, A., Farris, I., Bagaa, M., and taleb, T. (2019). Assessing Lightweight Virtualization for Security-as-a-Service at the Network Edge. *IEICE Transactions on Communications*, E102.B(5):970–977.

[Bremler-Barr et al. 2014] Bremler-Barr, A., Harchol, Y., Hay, D., and Koral, Y. (2014). Deep Packet Inspection as a Service. In *Anais da 10th ACM International Conference on Emerging Networking Experiments and Technologies (CoNEXT '14)*, page 271–282. Association for Computing Machinery.

[Bülbül and Fischer 2020] Bülbül, N. S. and Fischer, M. (2020). SDN/NFV-based DDoS Mitigation via Pushback. In *Anais da ICC 2020 - 2020 IEEE International Conference on Communications (ICC)*, pages 1–6.

[Cao et al. 2015] Cao, L., Sharma, P., Fahmy, S., and Saxena, V. (2015). NFV-VITAL: A Framework for Characterizing the Performance of Virtual Network Functions. In *Anais da 2015 IEEE Conference on Network Function Virtualization and Software Defined Network (NFV-SDN)*, pages 93–99.

[CheckPoint 2020] CheckPoint (2020). Keeping the gate locked on your IoT devices: Vulnerabilities found on Amazon's Alexa. Disponível em: `https://research.checkpoint.com/2020/amazons-alexa-hacked`. Acessado em 14 de Junho de 2021.

[Chi et al. 2019] Chi, Z., Li, Y., Sun, H., Yao, Y., and Zhu, T. (2019). Concurrent Cross-Technology Communication among Heterogeneous IoT Devices. *IEEE/ACM TON*, 27(3):932–947.

[da Silva et al. 2021] da Silva, M. d. V. D., Rocha, A., Gomes, R. L., and Nogueira, M. (2021). Lightweight Data Compression for Low Energy Consumption in Industrial Internet of Things. In *Anais da IEEE 18th Annual Consumer Communications Networking Conference (CCNC)*.

[De Benedictis and Lioy 2019] De Benedictis, M. and Lioy, A. (2019). A Proposal for Trust Monitoring in a Network Functions Virtualisation Infrastructure. In *Anais da 2019 IEEE Conference on Network Softwarization (NetSoft)*.

[Deogirikar and Vidhate 2017] Deogirikar, J. and Vidhate, A. (2017). Security Attacks in IoT: A Survey. In *Anais da International Conference on IoT in Social, Mobile, Analytics and Cloud*, pages 32–37.

[ETSI 2014a] ETSI (2014a). Network Functions Virtualisation (NFV): NFV Performance & Portability Best Practises. Disponível em: `https://www.etsi.org/deliver/etsi_gs/NFV-PER/001_099/001/01.01.01_60/gs_NFV-PER001v010101p.pdf`. Acessado em 13 de Maio de 2021.

[ETSI 2014b] ETSI (2014b). Network Functions Virtualisation (NFV): Service Quality Metrics. Disponível em: `https://www.etsi.org/deliver/etsi_gs/NFV-INF/001_099/010/01.01.01_60/gs_NFV-INF010v010101p.pdf`. Acessado em 13 de Maio de 2021.

[Farris et al. 2017] Farris, I., Bernabe, J. B., Toumi, N., Garcia-Carrillo, D., Taleb, T., Skarmeta, A., and Sahlin, B. (2017). Towards provisioning of SDN/NFV-based security enablers for integrated protection of IoT systems. In *Anais da 2017 IEEE Conference on Standards for Communications and Networking (CSCN)*, pages 169–174.

[Farris et al. 2019] Farris, I., Taleb, T., Khettab, Y., and Song, J. (2019). A Survey on Emerging SDN and NFV Security Mechanisms for IoT Systems. *IEEE COMST*, 21(1):812–837.

[Gao et al. 2013] Gao, Y., Peng, Y., Xie, F., Zhao, W., Wang, D., Han, X., Lu, T., and Li, Z. (2013). Analysis of security threats and vulnerability for cyber-physical systems. In *Anais da 2013 3rd International Conference on Computer Science and Network Technology*, pages 50–55.

[Gember-Jacobson et al. 2014] Gember-Jacobson, A., Viswanathan, R., Prakash, C., Grandl, R., Khalid, J., Das, S., and Akella, A. (2014). OpenNF: Enabling Innovation in Network Function Control. In *Anais da 2014 ACM Conference on SIGCOMM*, pages 163–174.

[Giaretta et al. 2019] Giaretta, A., Dragoni, N., and Massacci, F. (2019). IoT Security Configurability with Security-by-Contract. *Sensors*, 19(19).

[Guizani and Ghafoor 2020] Guizani, N. and Ghafoor, A. (2020). A Network Function Virtualization System for Detecting Malware in Large IoT Based Networks. *IEEE Journal on Selected Areas in Communications*, 38(6):1218–1228.

[Gupta et al. 2019] Gupta, L., Salman, T., Zolanvari, M., Erbad, A., and Jain, R. (2019). Fault and Performance Management in Multi-Cloud Virtual Network Services using AI: A Tutorial and a Case Study. *Computer Networks*, 165:106950.

[Hafeez et al. 2016] Hafeez, I., Ding, A. Y., Suomalainen, L., Kirichenko, A., and Tarkoma, S. (2016). Securebox: Toward Safer and Smarter IoT Networks. In *Anais do 2016 ACM Workshop on Cloud-Assisted Networking*, pages 55–60.

[IDC 2020] IDC (2020). IoT Growth Demands Rethink of Long-Term Storage Strategies, says IDC. Disponível em: `https://www.idc.com/getdoc.jsp?containerId=prAP46737220`. Acessado em 12 de Junho de 2021.

[IETF 1998] IETF (1998). RFC 2330: Framework for IP Performance Metrics. Disponível em: `https://datatracker.ietf.org/doc/html/rfc2330`. Acessado em 13 de Maio de 2021.

[IETF 2011] IETF (2011). RFC 6390: Guidelines for Considering New Performance Metric Development. Disponível em: `https://datatracker.ietf.org/doc/html/rfc6390`. Acessado em 13 de Maio de 2021.

[IETF 2017a] IETF (2017a). Benchmarking Methodology for Virtualization Network Performance. Disponível em: `https://tools.ietf.org/id/draft-huang-bmwg-virtual-network-performance-03.html`. Acessado em 13 de Maio de 2021.

[IETF 2017b] IETF (2017b). RFC 8172: Considerations for Benchmarking Virtual Network Functions and Their Infrastructure. Disponível em: `https://datatracker.ietf.org/doc/html/rfc8172`. Acessado em 13 de Maio de 2021.

[IETF 2020] IETF (2020). RFC 8520: Manufacturer Usage Description Specification. Disponível em: `https://datatracker.ietf.org/doc/rfc8520`. Acessado em 21 de Junho de 2021.

[Jia et al. 2020] Jia, Y., Zhong, F., Alrawais, A., Gong, B., and Cheng, X. (2020). FlowGuard: An Intelligent Edge Defense Mechanism Against IoT DDoS Attacks. *IEEE Internet of Things Journal*, 7(10):9552–9562.

[Kaspersky 2019] Kaspersky (2019). Brasil é o segundo país com mais ataques a dispositivos IoT. Disponível em: `https://www.kaspersky.com.br/about/press-releases/2019_brasil-e-o-segundo-pais-com-mais-ataques-a-dispositivos-iot`. Acessado em 9 de Março de 2021.

[Kaspersky 2020] Kaspersky (2020). Quase 30% das empresas que usam IoT sofreram incidentes de segurança. Disponível em: `https://www.kaspersky.com.br/blog/empresas-iot-seguranca-dicas/14869`. Acessado em 9 de Março de 2021.

[Kim et al. 2015] Kim, T., Koo, T., and Paik, E. (2015). SDN and NFV Benchmarking for Performance and Reliability. In *Anais do 2015 17th Asia-Pacific Network Operations and Management Symposium (APNOMS)*, pages 600–603.

[Laghrissi and Taleb 2019] Laghrissi, A. and Taleb, T. (2019). A Survey on the Placement of Virtual Resources and Virtual Network Functions. *IEEE COMST*, 21(2):1409–1434.

[Lin et al. 2017] Lin, J., Yu, W., Zhang, N., Yang, X., Zhang, H., and Zhao, W. (2017). A Survey on Internet of Things: Architecture, Enabling Technologies, Security and Privacy, and Applications. *IEEE Internet of Things Journal*, 4(5):1125–1142.

[Marku et al. 2020] Marku, E., Biczok, G., and Boyd, C. (2020). Securing Outsourced VNFs: Challenges, State of the Art, and Future Directions. *IEEE Communications Magazine*, 58(7):72–77.

[Massonet et al. 2017] Massonet, P., Deru, L., Achour, A., Dupont, S., Croisez, L.-M., Levin, A., and Villari, M. (2017). Security in Lightweight Network Function Virtualisation for Federated Cloud and IoT. In *Anais da 2017 IEEE 5th International Conference on Future Internet of Things and Cloud (FiCloud)*, pages 148–154.

[Medhat et al. 2017] 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.

[Meneghello et al. 2019] Meneghello, F., Calore, M., Zucchetto, D., Polese, M., and Zanella, A. (2019). IoT: Internet of Threats? A Survey of Practical Security Vulnerabilities in Real IoT Devices. *IEEE Internet of Things Journal*, 6(5):8182–8201.

[Mijumbi et al. 2016] Mijumbi, R., Serrat, J., Gorricho, J.-L., Bouten, N., Turck, F. D., and Boutaba, R. (2016). Network Function Virtualization: State-of-the-Art and Research Challenges. *IEEE COMST*, 18(1):236–262.

[Moghaddam et al. 2019] Moghaddam, S. K., Buyya, R., and Ramamohanarao, K. (2019). Performance-Aware Management of Cloud Resources: A Taxonomy and Future Directions. *ACM Computing Surveys*, 52(4).

[Montero and Serral-Gracià 2016] Montero, D. and Serral-Gracià, R. (2016). Offloading Personal Security Applications to the Network Edge: A Mobile User Case Scenario. In *Anais da 2016 International Wireless Communications and Mobile Computing Conference (IWCMC)*, pages 96–101.

[Montero et al. 2015] Montero, D., Yannuzzi, M., Shaw, A., Jacquin, L., Pastor, A., Serral-Gracia, R., Lioy, A., Risso, F., Basile, C., Sassu, R., Nemirovsky, M., Ciaccia, F., Georgiades, M., Charalambides, S., Kuusijarvi, J., and Bosco, F. (2015). Virtualized Security at the Network Edge: a User-Centric Approach. *IEEE Communications Magazine*, 53(4):176–186.

[Naik 2017] Naik, N. (2017). Choice of Effective Messaging Protocols for IoT Systems: MQTT, CoAP, AMQP and HTTP. In *Anais do 2017 IEEE International Systems Engineering Symposium (ISSE)*, pages 1–7.

[Neshenko et al. 2019] Neshenko, N., Bou-Harb, E., Crichigno, J., Kaddoum, G., and Ghani, N. (2019). Demystifying IoT Security: An Exhaustive Survey on IoT Vulnerabilities and a First Empirical Look on Internet-Scale IoT Exploitations. *IEEE Communications Surveys Tutorials*, 21(3):2702–2733.

[Qazi et al. 2013] Qazi, Z. A., Tu, C.-C., Chiang, L., Miao, R., Sekar, V., and Yu, M. (2013). SIMPLE-Fying Middlebox Policy Enforcement Using SDN. *SIGCOMM Comput. Commun. Rev.*, 43(4):27–38.

[Rosário et al. 2014] Rosário, D., Zhao, Z., Santos, A., Braun, T., and Cerqueira, E. (2014). A Beaconless Opportunistic Routing based on a Cross-layer Approach for Efficient Video Dissemination in Mobile Multimedia IoT Applications. *Computer Communications*, 45:21–31.

[Sairam et al. 2019] Sairam, R., Bhunia, S. S., Thangavelu, V., and Gurusamy, M. (2019). NETRA: Enhancing IoT Security Using NFV-Based Edge Traffic Analysis. *IEEE Sensors Journal*, 19(12):4660–4671.

[Schwengber et al. 2020] Schwengber, B. H., Vergütz, A., Prates, N. G., and Nogueira, M. (2020). A Method Aware of Concept Drift for Online Botnet Detection. In *Anais da 2020 IEEE Global Communications Conference (GLOBECOM)*, pages 1–6.

[Singh et al. 2016] Singh, J., Pasquier, T., Bacon, J., Ko, H., and Eyers, D. (2016). Twenty Security Considerations for Cloud-Supported Internet of Things. *IEEE Internet of Things Journal*, 3(3):269–284.

[Sobin 2020] Sobin, C. (2020). A Survey on Architecture, Protocols and Challenges in IoT. *Wireless Personal Communications*, 112:1383–1429.

[Wang et al. 2021] Wang, T., Fan, Q., Li, X., Zhang, X., Xiong, Q., Fu, S., and Gao, M. (2021). Drl-sfcp: Adaptive service function chains placement with deep reinforcement learning. In *Anais da ICC 2021 - 2021 IEEE International Conference on Communications (ICC)*.

[Yang et al. 2017] Yang, Y., Wu, L., Yin, G., Li, L., , and Zhao, H. (2017). A Survey on Security and Privacy Issues in Internet-of-Things. *IEEE Internet of Things Journal*, 4(5):1250–1258.

[Yousefpour et al. 2019] Yousefpour, A., Fung, C., Nguyen, T., Kadiyala, K., Jalali, F., Niakanlahiji, A., Kong, J., and Jue, J. P. (2019). All one needs to know about fog computing and related edge computing paradigms: A complete survey. *Journal of Systems Architecture*, 98:289–330.

[Yu et al. 2015] Yu, T., Sekar, V., Seshan, S., Agarwal, Y., and Xu, C. (2015). Handling a Trillion (Unfixable) Flaws on a Billion Devices: Rethinking Network Security for the Internet-of-Things. In *Anais do 14th ACM Workshop on Hot Topics in Networks (HotNets-XIV)*.

[Zarca et al. 2020a] Zarca, A. M., Bagaa, M., Bernal Bernabe, J., Taleb, T., and Skarmeta, A. F. (2020a). Semantic-Aware Security Orchestration in SDN/NFV-Enabled IoT Systems. *Sensors*, 20(13).

[Zarca et al. 2018] Zarca, A. M., Bernabe, J. B., Farris, I., Khettab, Y., Taleb, T., and Skarmeta, A. (2018). Enhancing IoT security through network softwarization and virtual security appliances. *International Journal of Network Management*, 28(5):e2038. e2038 nem.2038.

[Zarca et al. 2020b] Zarca, A. M., Bernabe, J. B., Skarmeta, A., and Alcaraz Calero, J. M. (2020b). Virtual IoT HoneyNets to Mitigate Cyberattacks in SDN/NFV-Enabled IoT Networks. *IEEE Journal on Selected Areas in Communications*, 38(6):1262–1277.

[Zarca et al. 2019a] Zarca, A. M., Bernabe, J. B., Trapero, R., Rivera, D., Villalobos, J., Skarmeta, A., Bianchi, S., Zafeiropoulos, A., and Gouvas, P. (2019a). Security Management Architecture for NFV/SDN-Aware IoT Systems. *IEEE Internet of Things Journal*, 6(5):8005–8020.

[Zarca et al. 2019b] Zarca, A. M., Garcia-Carrillo, D., Bernal Bernabe, J., Ortiz, J., Marin-Perez, R., and Skarmeta, A. (2019b). Enabling Virtual AAA Management in SDN-Based IoT Networks. *Sensors*, 19(2).

[Zhang et al. 2016] Zhang, W., Hwang, J., Rajagopalan, S., Ramakrishnan, K. K., and Wood, T. (2016). Performance Management Challenges for Virtual Network Functions. In *2016 IEEE NetSoft Conference and Workshops*, pages 20–23.

[Zhou et al. 2019] Zhou, L., Guo, H., and Deng, G. (2019). A fog computing based approach to DDoS mitigation in IIoT systems. *Computers & Security*, 85:51–62.

