Universidade Federal de Juiz de Fora
Instituto de Ciências Exatas
Bacharelado em Sistemas de Informação

Solução SDN para gerência de tráfego na virtualização do P-GW no LTE.

Rafael Gonçalves Motta

JUIZ DE FORA
DEZEMBRO, 2018

Solução SDN para gerência de tráfego na virtualização do P-GW no LTE.

Rafael Gonçalves Motta

Universidade Federal de Juiz de Fora

Instituto de Ciências Exatas

Departamento de Ciência da Computação

Bacharelado em Sistemas de Informação

Orientador: Luciano Jerez Chaves

JUIZ DE FORA

DEZEMBRO, 2018

Solução SDN para gerência de tráfego na virtualização do P-GW no LTE.

Rafael Gonçalves Motta

MONOGRAFIA SUBMETIDA AO CORPO DOCENTE DO INSTITUTO DE CIÊNCIAS EXATAS DA UNIVERSIDADE FEDERAL DE JUIZ DE FORA, COMO PARTE INTEGRANTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE BACHAREL EM SISTEMAS DE INFORMAÇÃO.

Aprovada por:

Luciano Jerez Chaves
Mestre em Ciência da Computação - UNICAMP

Alex Borges Vieira
Doutor em Ciência da Computação - UFMG

Edelberto Franco Silva
Doutor em Ciência da Computação - UFF

JUIZ DE FORA
03 DE DEZEMBRO, 2018

Resumo

Resumo: Com o crescimento das redes de computadores e a necessidade de flexibilidade de seus equipamentos, as Redes Definidas por Software (SDN) surgem como forma de melhorar o gerenciamento ao centralizar o plano de controle e distribuir o plano de dados nos elementos da rede. O SDN se mostra como um habilitador para a Virtualização das Funções de Rede (NFV), que oferece flexibilidade ao software responsável pelo processamento das funções de rede ao separá-lo do hardware onde ele é executado. Este trabalho incorpora essas tecnologias e propõe uma abordagem de gerência de tráfego baseada na otimização do uso dos recursos dos *switches* num cenário de virtualização do plano de dados de um *Gateway* de Pacotes (P-GW) em uma rede *Long Term Evolution* (LTE). Com a implementação de um controlador especializado, a abordagem proposta mostra-se eficiente ao elevar os níveis de vazão da rede e reduzir as taxas de bloqueio de tráfego num cenário com *switches* OpenFlow de recursos heterogêneos.

Palavras-chave: Redes Definidas por Software; Protocolo OpenFlow; Virtualização das Funções de Rede; Simulação de redes; *Network Simulator 3* (ns-3).

Abstract

The necessity for flexibility in network hardware has grown and the Software Defined Networks (SDN) arise as a way to improve network management, centralizing the control plane and distributing the data plane through the components of the network. The use of SDN allows for the Network Function Virtualization (NFV) to offer flexibility when it decouples the software responsible for network functions from the hardware where it is implemented. This work incorporates these technologies to propose an approach to traffic management based on optimization of the switches' resources. The scenario used is the virtualization of the dataplane of a Packet Gateway in a Long Term Evolution (LTE) network. With the creation of a specialized controller, the proposed approach shows its efficiency by elevating the network throughput and reducing traffic block rates in a scenario with OpenFlow switches with heterogeneous resources.

Keywords: Software Defined Networks; OpenFlow Protocol; Network Function Virtualization; Network simulation; *Network Simulator 3* (ns-3)

Agradecimentos

Aos meus pais, pelo apoio incondicional e incentivo ao estudo desde cedo.

Aos amigos e familiares, pelo apoio e compreensão. Em especial ao amigos Jack e John, por nunca permitirem que a distância física impedisse de estarem presentes.

Ao professor Luciano Jerez Chaves, pela orientação, amizade, encorajamento e compreensão. Sua conduta é um exemplo que levo para a vida.

Aos professores do Departamento de Ciência da Computação, pelos ensinamentos que contribuíram para o meu enriquecimento pessoal e profissional.

"Don't you think we're extraordinary?

Believing, and seeing

Realizing the imaginary

Don't you? Don't you?

Yes, I think we're extraordinary"

Serj Tankian - Cornucopia

Conteúdo

Lista de Figuras

Lista de Abreviações

4G	Quarta geração
5G	Quinta geração
API	*Application Program Interface*
CPU	*Central Processing Unit*
DHCP	*Dynamic Host Configuration Protocol*
DPI	*Deep Packet Inspection*
eNB	*Evolved Node B*
EPC	*Evolved Packet Core*
ETSI	*European Telecommunication Standards Institute*
E-UTRAN	*Evolved Universal Terrestrial Radio Access Network*
GPL	*General Public License*
HSS	*Home Subscriber Server*
IP	*Internet Protocol*
LTE	*Long Term Evolution*
MME	*Mobility Management Entity*
MPCN	*Mobile Packet Core Network*
NFV	*Network Function Virtualization*
ns-3	*Network Simulator 3*
ONF	*Open Network Foundation*
OXM	*OpenFlow eXtensible Match*
PCRF	*Policy Control and Charging Rules Function*
PDN	*Packet Data Network*
P-GW	*PDN Gateway*
SDN	*Software Defined Network*
S-GW	*Serving Gateway*
TCAM	*Ternary Content-Addressable Memory*
TFT	*Traffic Flow Template*

TLS	*Transport Layer Security*
TLV	*Type-Length-Value*
UE	*User Equipment*
VNF	*Virtualized Network Function*
VoIP	*Voice Over IP*

1 Introdução

1.1 Motivação

A maioria das atividades da sociedade atual passa por uma ou mais redes de computadores. A tecnologia espalhou-se e encontra-se em vários ambientes como casas, empresas e escolas. A Internet é um recurso muito utilizado pela população e saber como usá-la tornou-se uma característica essencial ao profissional do século XXI.

Com a expansão e ampla adoção da Internet, a arquitetura de redes de computadores atingiu sua maturidade e tornou-se pouco flexível. Como forma de resolver este problema têm-se investido em abordagens que tornem as redes programáveis, permitindo que novas tecnologias possam ser estudadas e implementadas.

Cada vez mais os equipamentos de rede precisam oferecer versatilidade de configuração e uso. No cenário atual, roteadores usam *softwares* fechados em *hardwares* proprietários, permitindo alguma flexibilidade apenas com o uso de protocolos proprietários. Essas características resultam em soluções caras e sem suporte à inovação e experimentação (HAMPEL; STEINER; BU, 2013).

As Redes Definidas por *Software* (SDN, do Inglês *Software Defined Networks*) apresentam uma abordagem diferente, gerenciável, adaptável e com melhor custo-benefício (Open Networking Foundation, 2012). Elas permitem o gerenciamento centralizado e estratégico da rede, possibilitando a configuração dos fluxos de tráfego e a reação a eventos adversos, tudo a partir de uma interface de *software* e sem depender de alterações no *hardware* dos elementos da rede.

O OpenFlow é a tecnologia mais utilizada no desenvolvimento e em pesquisas sobre SDN (COSTA et al., 2017). Ele permite a programação das tabelas de encaminhamento do *hardware* para definir as ações a serem executadas sobre cada pacote. Assim, o encaminhamento mantém-se eficiente, pois continua sendo executado pelo *hardware*, mas a decisão sobre o processamento dos pacote é transferida para uma camada de controle, onde novas funcionalidades podem ser implementadas por um *software* centralizado.

Outra tecnologia importante na busca por flexibilidade da arquitetura de redes é a Virtualização das Funções de Rede (NFV, do Inglês *Network Function Virtualization*). Através da separação entre o *software* responsável pela função de rede e o *hardware* onde ele é executado, essa virtualização permite o aumento e a redução de uso dos recursos conforme demanda, além do uso de equipamentos genéricos, que tendem a ter custo inferior em relação a equipamentos especializados. Além disso, o NFV permite a configuração dinâmica da rede, com a possibilidade de instanciar as funções de rede em diferentes locais, de acordo com as necessidades de cada ambiente.

As redes *Long Term Evolution* (LTE) de 4ª geração (4G) são as redes para acesso móvel mais utilizadas atualmente. Uma das tendências na evolução do 4G para o 5G é a aplicação dos paradigmas de SDN e NFV na arquitetura, desassociando *hardware* de *software* e centralizando o plano de controle (NGUYEN; DO; KIM, 2015). Este trabalho tem como foco a aplicação destas tecnologias em um dos elementos da rede LTE: o *Gateway* de Pacotes (P-GW, do Inglês *Packet GateWay*). Este *gateway* é responsável por conectar a rede LTE à Internet e tem que processar o tráfego de todos os usuários da rede com alta disponibilidade.

1.2 Objetivos e contribuições

Conforme proposto em (CHAVES; GARCIA; MADEIRA, 2017), uma solução para a implementação do *gateway* de pacotes de maneira virtualizada pode ser feita usando *switches* OpenFlow. As regras dos *switches* representam o processamento necessário para cada um dos fluxos. Para substituir adequadamente este *gateway* é necessário escalabilidade dos recursos. A solução proposta inclui a utilização de vários *switches*, balanceando a carga entre eles. Entretanto, na proposta citada só foram discutidas situações onde os *switches* são homogêneos do ponto de vista de recursos disponíveis para uso (tamanho da tabela de fluxo e capacidade de processamento). Considerando a dinamicidade de um cenário real, onde equipamentos podem ser agregados à rede em diferentes momentos, é proposto um estudo de caso para avaliar o desempenho desse *gateway* virtualizado quando temos *switches* com recursos diferentes.

Para avaliar o comportamento do tráfego num cenário híbrido, este trabalho

compara duas políticas de gerência de tráfego: a *Política Arbitrária* e uma nova *Política Direcionada*. A *Política Arbitrária*, adotada na proposta citada, divide os tráfegos quantitativamente entre os *switches*, e foi idealizada considerando um cenário homogêneo. Já a nova *Política Direcionada* é a contribuição original deste trabalho, e baseia-se na distribuição dos tráfegos de maneira a otimizar o uso dos recursos dos *switches* em cenários heterogêneos, redirecionando tráfegos com maiores vazões para *switches* com melhor capacidade de processamento e mantendo os demais tráfegos nos *switches* com maiores tabelas de fluxos.

Durante a avaliação das políticas são analisadas duas estratégias de bloqueio de tráfego por sobrecarga de recursos dos *switches*. Para cada combinação de política de tráfego e de estratégia de bloqueio, foram feitas simulações com diferentes cargas de tráfego na rede. Com os resultados foi possível analisar comportamentos diferentes para várias métricas entre as duas políticas. Os resultados confirmam uma vantagem da *Política Direcionada* na melhor utilização dos recursos dos *switches*, tendo como consequência o aumento da vazão agregada da rede.

Os próximos capítulos deste trabalho estão organizados da seguinte maneira: No capítulo 2 são apresentados os fundamentos teóricos das tecnologias envolvidas; no capítulo 3 é apresentado o ambiente de avaliação, com a descrição do cenário e das políticas de gerência de tráfego adotadas; no capítulo 4 são apresentados os parâmetros usados na simulação e os resultados obtidos são apresentados e analisados; no capítulo 5 é apresentada a conclusão deste trabalho.

2 Fundamentação teórica

2.1 Virtualização das Funções de Rede

Os serviços tradicionais de rede baseiam-se em *hardware* proprietário com funções limitadas definidas pelo fabricante do equipamento. Esta situação dificulta o desenvolvimento de novos serviços e a consequente melhoria da rede, pois não é possível modificar o funcionamento dos equipamentos além dos ajustes de configurações.

Para resolver este problema, o *European Telecommunication Standards Institute* (ETSI) propôs a Virtualização das Funções de Redes (NFV, do Inglês *Network Function Virtualization*), como forma de virtualizar as funções de rede antes executadas por *hardware* dedicado e proprietário. O NFV pode ser definido como a separação do *software* responsável pelo processamento das funções da rede do *hardware* onde ele é executado (MIJUMBI J. SERRAT; BOUTABA, 2015). Separando *software* do *hardware*, o NFV permite flexibilidade de *software* numa infraestrutura de *hardware* otimizada e compartilhada. Essa otimização reduz o custo de operação e manutenção pois *hardware* e *software* proprietários podem ser substituídos por versões de menor custo (LI; CHEN, 2015).

A Figura 2.1 apresenta uma visão em alto nível da arquitetura do NFV. No

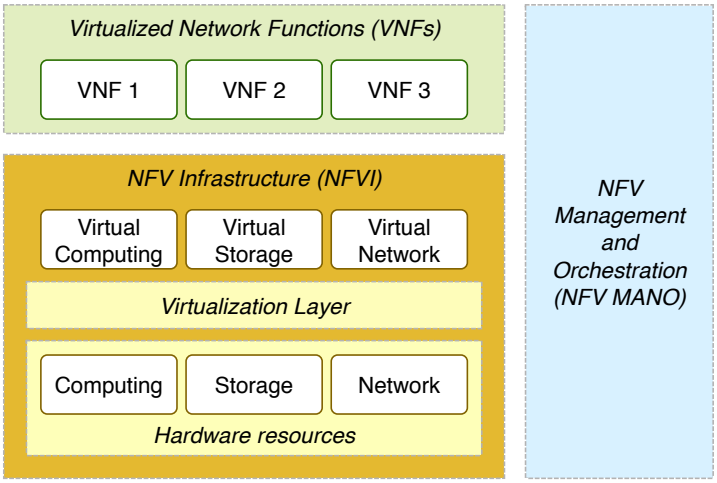


Figura 2.1: Arquitetura NFV. Adaptado de (ETSI NFV 002, 2014).

topo da arquitetura encontramos as *Virtualized Network Functions* (VNFs). Cada VNF representa um elemento da rede que pode ser virtualizado, como roteadores, *firewalls*, servidores DHCP, sistemas de detecção de intrusão, *Deep Packet Inspection* (DPI), entre outros (LI; CHEN, 2015). Logo abaixo encontramos a *NFV Infrastructure* (NFVI) que é responsável pela união de recursos de *software* e *hardware*, criando o ambiente que permita a instalação de uma VNF. Para orquestrar a arquitetura temos o *NFV Management and Orchestration* (NFV MANO), responsável por gerenciar os recursos e garantir que haja processamento, armazenamento e recursos de rede disponíveis para cada VNF (ABDELWAHAB et al., 2016).

2.2 Redes Definidas por Software

Redes Definidas por *Software* (SDN, do Inglês *Software Defined Networks*) é um paradigma criado para permitir que redes sejam flexíveis (Open Networking Foundation, 2012). Na arquitetura SDN os planos de dados e de controle são separados, o que permite um melhor entendimento de toda a estrutura da rede quando comparado com os algoritmos tradicionais, que são distribuídos e executados individualmente em cada nó da rede. Com isso o processo de tomada de decisão centralizado é melhorado (Open Networking Foundation, 2013).

A *Open Networking Foundation* (ONF) definiu a arquitetura SDN conforme ilustrado na Figura 2.2. A arquitetura é composta de três camadas acessíveis através de *Application Program Interfaces* (APIs) abertas. A camada de aplicação é composta pelas aplicações de negócio que usam os serviços de comunicação da SDN. A camada de controle é responsável pela supervisão do encaminhamento de pacotes. A camada de infraestrutura contém os dispositivos e elementos de rede responsáveis pelo encaminhamento de pacotes.

O controle da rede é feito através de um *software* controlador que pode analisar o estado da rede em tempo real e otimizar as rotas de encaminhamento. Isso permite que os operadores da rede tenham a flexibilidade de configurar, gerenciar e otimizar a rede de maneira centralizada.

Tanto o NFV quanto o SDN tem como objetivo aumentar a flexibilidade da rede, reduzir custos e aumentar a escalabilidade e velocidade de implementação de novos

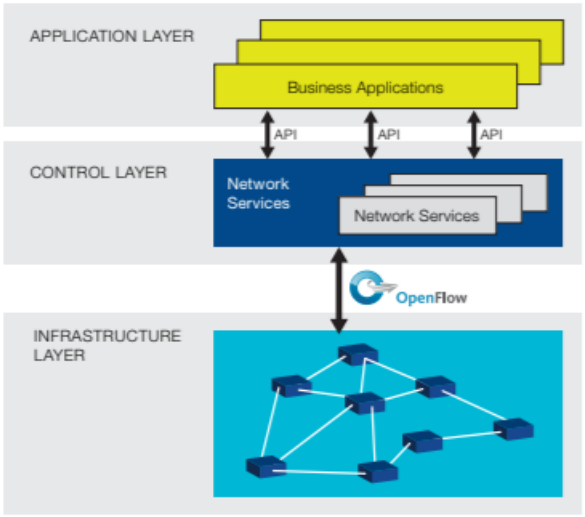


Figura 2.2: Arquitetura SDN (Open Networking Foundation, 2014)

serviços. São tecnologias independentes entre si mas que podem se beneficiar quando usadas em conjunto: o NFV pode virtualizar elementos do SDN, como o controlador, colocando-o como uma VNF; e o SDN permite programar com facilidade as conexões entre as VNFs (LI; CHEN, 2015).

2.2.1 Protocolo OpenFlow

O OpenFlow é um protocolo SDN criado para interligar a camada de controle à camada de infraestrutura (MCKEOWN et al., 2008). O protocolo é implementado tanto nos dispositivos da infraestrutura de rede quanto no *software* controlador. O OpenFlow usa o conceito de fluxos para identificar tráfego na rede baseado em regras que podem ser programadas pelo controlador nos *switches*.

A implementação atual do protocolo OpenFlow contempla os componentes e as funções básicas de um *switch*, conforme ilustra a Figura 2.3. O canal OpenFlow é a interface que conecta o *switch* OpenFlow ao controlador. Esta interface permite que o controlador configure e gerencie o *switch*, receba eventos e envie pacotes para a rede. O canal de controle do *switch* pode suportar um canal OpenFlow com um controlador ou vários canais OpenFlow, permitindo que vários controladores compartilhem o gerenciamento do *switch*. Todas as mensagens do canal OpenFlow devem seguir o formato especificado no protocolo, com criptografia opcional pelo protocolo *Transport Layer Security* (TLS).

O caminho de dados do *switch* OpenFlow é composto de uma ou mais tabelas

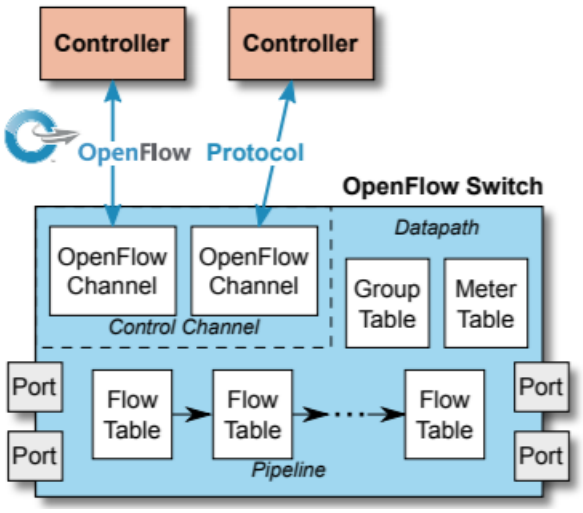


Figura 2.3: Componentes de um *switch* OpenFlow (OpenFlow 1.5.1, 2015)

de fluxo, uma tabela de grupo e uma tabela de *meter*. As tabelas de fluxo inspecionam e encaminham pacotes de acordo com as entradas de fluxos configuradas pelo controlador. Cada entrada é composta de campos *OpenFlow eXtensible Match* (OXM) do tipo *Type-Length-Value* (TLV), que identificam valores para os cabeçalhos dos pacotes de um fluxo de dados. Além desses campos, as regras de fluxo também possuem contadores para coleta de estatísticas e um conjunto de instruções a serem aplicadas aos pacotes que fazem parte do fluxo de dados.

O processamento dos pacotes começa na primeira tabela de fluxo, de acordo com a prioridade das regras instaladas, e pode continuar pelas tabelas subsequentes. Se uma correspondência é encontrada, as instruções associadas ao fluxo específico são executadas. Instruções podem conter ações ou modificar a sequência do processamento do pacote pelas tabelas do *pipeline*. As ações podem ser de encaminhamento de pacotes para portas de saída do *switch*, modificações no conteúdo do pacote ou direcionamento do pacote para processamento nas tabelas de *meter* e grupo.

Os grupos representam conjuntos de ações mais complexas de encaminhamento, que podem ser utilizados para inundação da rede, balanceamento de carga, recuperação de enlaces com falhas, etc. Já as *meters* são regras usadas para monitorar a vazão de cada fluxo e, eventualmente, descartar os pacotes excedentes para implementar os mecanismos limitadores de vazão.

2.3 Redes móveis LTE

As redes *Long Term Evolution* (LTE) de quarta geração (4G) são as redes mais usadas atualmente para comunicação sem fio de alta velocidade, atingindo uma fatia de mercado de 35% no fim de 2017 e com previsão de atingir 60% até 2022 (GLOBE NEWS, 2018). Além disso, com a iminente chegada da quinta geração das redes móveis (5G), os desafios serão cada vez maiores para oferecer altas taxas de transmissão de dados para um número cada vez maior de dispositivos e aplicações.

A arquitetura do *LTE* pode ser vista na Figura 2.4. Ela é dividida em duas partes: *Evolved Universal Terrestrial Radio Access Network* (E-UTRAN) e *Evolved Packet Core* (EPC). O *E-UTRAN* é composto de pelo *Evolved Node B* (eNB), que é responsável por fornecer a interface de comunicação via rádio com o plano de dados e plano de controle do dispositivo do usuário, representando na imagem pelo *UE* (*User Equipment*).

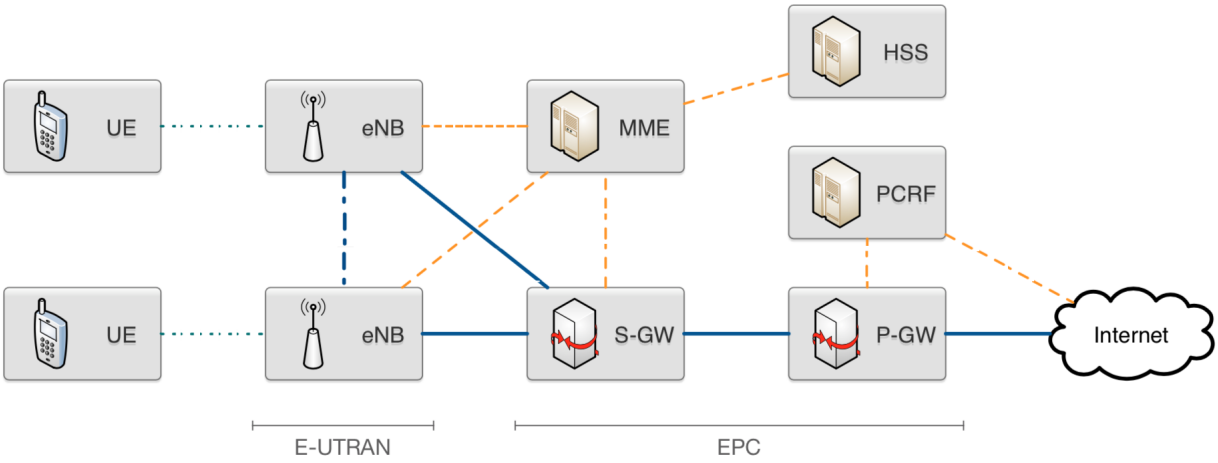


Figura 2.4: Arquitetura do LTE

O *EPC* oferece acesso somente ao domínio do *Packet Data Network* (PDN). Ele é formado por diferentes tipos de nós:

- *Mobility Management Entity* (MME): É o elemento chave do plano de controle do EPC. Suas funções principais estão relacionadas ao gerenciamento da conexão (segurança, escolha do gateway, mobilidade e entrega).
- *Policy Control and Charging Rules Function* (PCRF): Responsável pela tomada de decisão da política de controle. Também fornece a autorização de QoS que decide como um fluxo será tratado pelos elementos do plano de dados.

- *Home Subscriber Server* (HSS): Mantém uma base de informações como perfis de usuário, restrições em *roaming*, informações de segurança e localização e autorizações de acessos e serviços.

- *Serving Gateway* (S-GW): É o nó do plano de dados que conecta o EPC ao E-UTRAN. Também atua como âncora de roteamento durante a mobilidade dos usuários.

- *PDN Gateway* (P-GW): É o nó do plano de dados que conecta o *EPC* à Internet. É responsável pela alocação de endereços *IP* para um *UE* além de garantir a *QoS* estabelecida pelo *PCRF*. Para garantir *QoS* o *P-GW* usa *Traffic Flow Templates*, um conjunto de regras para fazer inspeção detalhada dos pacotes e classificar os tráfegos em túneis.

A arquitetura do LTE é complexa e inflexível. O *E-UTRAN* oferece alta vazão mas não possui uma coordenação eficiente entre os *eNBs*. Entretanto, o *EPC* depende de equipamentos especializados como o *S-GW* e o *P-GW* para prover mobilidade para os *UE* (ARSLAN; SUNDARESAN; RANGARAJAN, 2015). Como indicado em (HAMPEL; STEINER; BU, 2013), os *gateways* podem se tornar um ponto crítico de falha e por isso demandam alta confiabilidade, acarretando em alto custo. Os *gateways* são complexos, caros e incompatíveis entre versões de diferentes fabricantes.

2.4 NFV e SDN em redes móveis

Implementar NFV e SDN em redes móveis tem sido o objetivo de várias pesquisas recentes. O trabalho de (AKYILDIZ; LIN; WANG, 2015) faz uma avaliação qualitativa de várias propostas de pesquisa de uso de NFV e SDN para redes 5G. A maioria das soluções que envolvem o uso de NFV e SDN concentra-se no núcleo de pacotes da rede móvel (do inglês *Mobile Packet Core Network*, MPCN). (NGUYEN et al., 2017) e (NGUYEN; DO; KIM, 2015) apresentam uma revisão de trabalhos que usam virtualização nesse núcleo, classificando os trabalhos de acordo com diferentes aspectos. Uma arquitetura de redes móveis com SDN chamada *MobileFlow* é proposta em (PENTIKOUSIS; WANG; HU,

2013), onde o plano de controle é feito via *software* e o plano de dados é composto apenas de elementos simples com suporte a manipulação de túneis.

No contexto de virtualização do *gateway*, (BASTA et al., 2013) fez uma análise das funções de um *gateway* de uma rede móvel e mapeou-as em diferentes *frameworks* de implementação, apresentando as vantagens e desvantagens de cada implementação. Esse trabalho continuou em (BASTA et al., 2014), dividindo o problema da implementação das funções do *gateway* em duas categorias: virtualização do *gateway* e decomposição do *gateway*. A virtualização exige o direcionamento do tráfego de uma rede para um *data center*, o que implica no aumento da carga e do atraso do tráfego. Ao decompor as funções do *gateway*, somente o plano de controle é direcionado para o *data center* e o plano de dados pode ser processado com o uso de SDN. Para encontrar a melhor solução, os autores levaram em conta a carga do plano de controle e a latência do plano de dados e tentaram minimizar esses parâmetros. O trabalho de (AN; KIESS; PEREZ-CAPARROS, 2014) foca na virtualização do *gateway* e propõe uma arquitetura que aplica SDN para separar os planos de controle e de dados e aplica NFV para implantar a função de encaminhamento do plano de dados do *gateway* em *hardware* de custo reduzido. Um modelo de processamento em *cluster* é usado para superar as limitações de performance no uso de um servidor único. *Switches* OpenFlow e um controlador OpenFlow melhorado são usados para balanço de carga.

2.5 Network Simulator 3

O estudo e a adoção de novos protocolos na Internet tornou-se inviável devido ao risco de interrupção dos serviços para as quais ela se tornou essencial. Além disso, existe um alto custo de implementação de arquiteturas de redes de grande escala com fins experimentais. Dessa forma, o uso de ferramentas de *software* para pesquisa e desenvolvimento tem se mostrado como a melhor alternativa.

Tratando-se do OpenFlow, a ferramenta mais utilizada para estudo é o Mininet, um emulador *Open Source* que permite criar e avaliar uma SDN de forma fácil e rápida. Entretanto, o Mininet possui limitações: Sua largura de banda é limitada pelo desempenho do *hardware* e não há a possibilidade de dilatação de tempo, o que impede a realização de

emulações quando a demanda computacional for maior que a capacidade de processamento do *hardware* em tempo real. Os simuladores se apresentam como soluções para este tipo de limitação. *Softwares* como o Simulador de Redes 3 (ns-3, do Inglês *Network Simulator 3*) surgiram como ferramenta capaz de desenvolver simulações fiéis de ambientes reais sem a necessidade de implantar fisicamente os protocolos na rede.

O *ns-3* é um *software* livre para simulação de eventos discretos licenciado sob o GNU GPLv2 (*GNU General Public License*). O objetivo do ns-3 é oferecer um ambiente de simulação de redes em nível de pacote, que atenda as demandas de pesquisadores de ponta na academia e na indústria, sempre encorajando os usuários a contribuírem para a evolução do *software* (Network Simulator 3, 2018).

O *ns-3* é construído com foco em prover um ambiente sólido, bem documentado, de fácil utilização e que atenda às necessidades de todas as etapas da simulação, desde a configuração da simulação até a coleta e análise dos resultados. Além disso, sua infraestrutura permite o desenvolvimento de simulações realistas o suficiente para que ele seja usado como um emulador de rede em tempo real, conectado com o mundo real, o que permite a reutilização de implementações de protocolos reais dentro do *ns-3*.

Considerando o foco deste trabalho, é possível simular redes OpenFlow com o *ns-3*, avaliando o comportamento do tráfego quando os equipamentos de rede possuem restrições como tamanho de tabelas de fluxo ou capacidade de processamento limitadas.

2.5.1 Módulo OFSwitch13

É possível simular redes SDN com o ns-3 através do módulo OpenFlow que é distribuído juntamente com a árvore padrão de código do simulador (HURD, 2010). Entretanto, este módulo apresenta uma versão desatualizada do protocolo OpenFlow (versão 0.8.9, de 2008), o que o torna desinteressante para pesquisas de ponta nesta área.

Para adicionar as diversas funcionalidades introduzidas nas versões mais recentes do protocolo OpenFlow, foi desenvolvido um novo módulo para o ns-3: o OFSwitch13 (CHAVES; GARCIA; MADEIRA, 2016). Este módulo permite a realização de simulações com o OpenFlow versão 1.3.5, incrementando o simulador com os modelos do *switch* e do canal OpenFlow. Com este módulo é possível interconectar nós da simulação através

dos *switches* OpenFlow, que serão os responsáveis por receber e enviar os pacotes pela rede. Para gerenciar os *switches* existe uma classe base para o controlador, que pode ser estendida de forma a implementar a lógica de controle desejada.

A Figura 2.5 ilustra a visão geral deste módulo. Este módulo se baseia em uma biblioteca externa denominada *ofsoftswitch13*, que de fato implementa o caminho de dados do *switch* e as ferramentas necessárias para manipulação das mensagens OpenFlow. A comunicação entre a aplicação controladora da rede e o *switch* OpenFlow é realizada através dos protocolos, dispositivos de rede e canais padrão do ns-3.

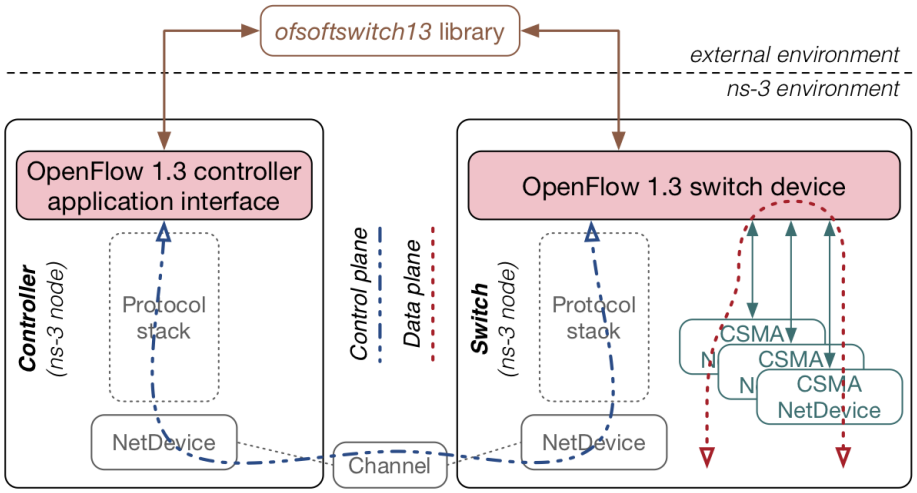


Figura 2.5: Arquitetura do módulo OFSwitch13

Os *switches* criados pelo OFSwitch13 podem ter os parâmetros do caminho de dados configurados de acordo com o cenário de simulação desejado. Dentre os parâmetros ajustáveis temos: o tamanho das tabelas de fluxo, de grupo e de *meter*; a capacidade de processamento do *switch*; e o tempo de execução de uma operação de busca em memória do tipo *Ternary Content-Addressable Memory* (TCAM). O tempo médio de processamento do pacote no *switch* depende da organização interna das tabelas na memória, sendo que no OFSwitch13 este tempo é proporcional ao logaritmo de base 2 do número de regras instaladas.

Uma das novas funcionalidades oferecidas pelo OFSwitch13 em relação ao módulo OpenFlow padrão do ns-3 é a fiel implementação do canal OpenFlow, permitindo a coleta de *traces* em formato PCAP, além da utilização de um controlador SDN real para gerenciar os *switches* no ambiente de simulação. Podemos destacar também outras funcionalidades do protocolo OpenFlow 1.3, que antes não eram suportadas no ns-3 e que agora podem

ser simuladas com o OFSwitch13:

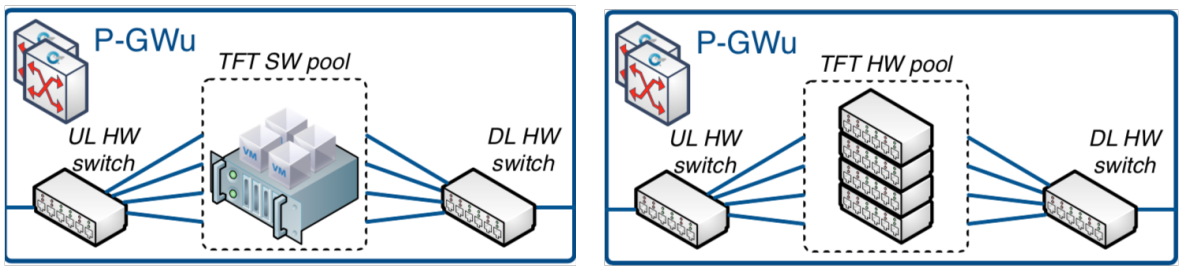
- Operações avançadas sobre pacotes recebidos e enviados pelo *switch* através de portas lógicas, como é o caso do encapsulamento em túneis;
- Suporte a regras com *match* em cabeçalhos IPv6;
- Processamento de pacotes em *pipeline* por múltiplas tabelas de fluxo, diminuindo o tamanho efetivo das tabelas e explorando o paralelismo no processamento;
- Criação de limitadores de fluxo através de entradas na tabela de *meter*, que controlam a taxa de pacotes de um determinado tráfego;
- Abstração de conjunto de regras de encaminhamento como um grupo, viabilizando operações avançadas no encaminhamento de pacotes;
- Compartilhamento da gerência de um *switch* por múltiplos controladores, através de canais OpenFlow independentes.

3 Ambiente de avaliação

3.1 Cenário de teste

O cenário adotado como motivação para este trabalho considera a aplicação dos princípios de NFV em redes *Long Term Evolution* (LTE). As redes LTE têm sido usadas como meio de comunicação sem fio de alta velocidade. Dentre os desafios enfrentados temos a demanda crescente por largura de banda e a disponibilidade para muitos usuários, o que é praticamente impossível de ser alcançado com a arquitetura de rede tradicional (Open Networking Foundation, 2012). Um dos pontos críticos da arquitetura do LTE são os *Gateways* de Pacotes (P-GWs) presentes no núcleo da rede LTE. Com *hardware* proprietário, sua função é lidar com o tráfego de milhares de usuários com alta disponibilidade. Por isso, tendem a ser complexos, caros e de baixa compatibilidade com equipamentos de outros fabricantes.

Em um *Gateway P-GW*, a classificação e o encaminhamento de pacotes para diferentes túneis é realizado pelos *Traffic Flow Templates* (TFT). Os TFT usam informações do cabeçalho IP, como endereços de origem e destino, além dos números das portas de entrada e saída para encaminhar os pacotes para os túneis corretos. Em (CHAVES et al., 2015) é apresentada uma proposta alternativa para a substituição destes *gateways*, que passam a ser virtualizados sobre uma rede de *switches* OpenFlow. Essa topologia, apresentada na Figura 3.1, é composta por um conjunto de *switches* OpenFlow denominados



(a) *Switches TFT* baseados em *software* (b) *Switches TFT* baseados em *hardware*

Figura 3.1: P-GW virtualizado sobre *switches* OpenFlow (CHAVES et al., 2015).

de *Switches TFT* que exercem a função de classificação dos tráfegos através das regras instaladas nas tabelas de fluxo. Como a quantidade de regras cresce proporcionalmente ao tamanho da rede, então um único *switch* não será capaz de realizar essa tarefa sozinho. Dessa forma, essa topologia conta com mais dois *switches* OpenFlow chamados aqui de *Uplink Switch* (UL HW *switch*) e *Downlink Switch* (DL HW *switch*). Esses *switches* são responsáveis por distribuir o tráfego entre os vários *Switches TFTs*, eventualmente balanceando a carga de trabalho entre eles. Essa decisão pode ser arbitrária ou pode ser tomada de acordo com métricas fornecidas pelo protocolo OpenFlow, como as demandas dos tráfegos e os recursos disponíveis nos *switches*. Atuando como pontos de entrada e saída do P-GW, os *Switches Uplink* e *Downlink* tendem a ter uma grande carga de tráfego e por isso os *hardwares* de alto desempenho são os mais indicados para eles (KIESS; AN; BEKER, 2015). Já os *Switches TFT* podem ser implementados via *hardware* OpenFlow ou virtualizados sobre *software* que simulam um *switch* OpenFlow, como é o caso do Open vSwitch (Open vSwitch, 2017).

Switches implementados via *software* ou *hardware* apresentam diferentes características. Em geral *switches* implementados em *software* possuem tabelas de fluxo com tamanhos maiores que switches implementados em *hardware*, além de serem mais fiéis às especificações do protocolo OpenFlow. Entretanto, o atraso no encaminhamento de pacotes num ambiente virtualizado pode ser até 10 vezes maior em relação ao atraso de um switch com *hardware* dedicado (COSTA et al., 2017).

No cenário adotado neste trabalho vamos utilizar uma abordagem híbrida, com os dois tipos de *switches* sendo usados em conjunto para implementar os *Switches TFTs*: um *Hardware Switch* e um *Software Switch*. O cenário adotado é apresentado pela Figura 3.2. Todos os *switches* são gerenciados pelo mesmo *Controlador*, que é responsável pela instalação e remoção de regras nas tabelas de fluxo dos *switches* e decisão sobre a gerência do tráfego nesta topologia. O funcionamento do *Controlador* será discutido em detalhes na Seção 3.2.

As redes LTE são baseadas no protocolo IP, focadas principalmente em aplicações multimídia (CHAVES et al., 2015). Para criar um cenário mais realístico, diferentes modelos de tráfego para aplicações cliente-servidor foram usados nas simulações: acesso

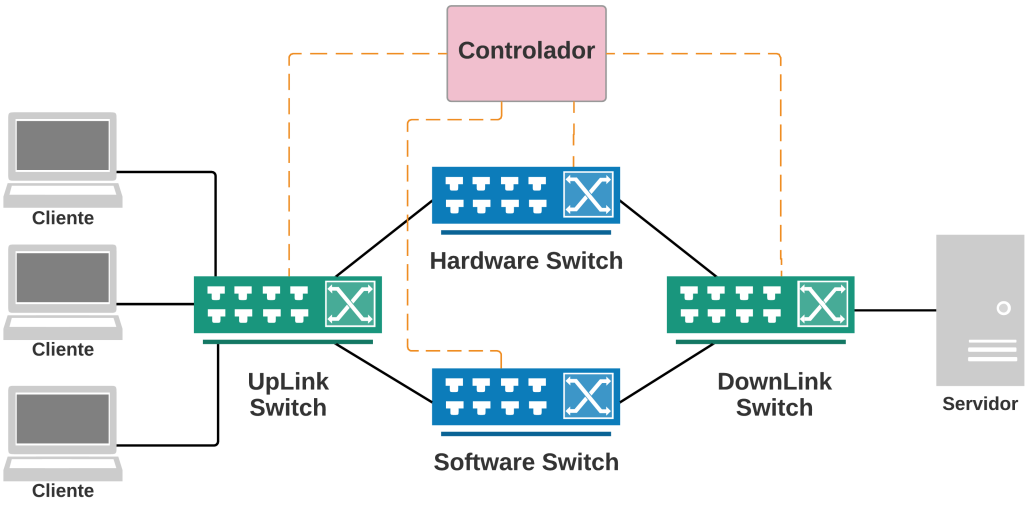


Figura 3.2: Topologia do cenário da simulação

a páginas web, *Voice Over IP* (VoIP), acesso a vídeos bufferizados, *streaming* de vídeos em tempo real, monitoramento remoto de localização, jogos on-line e comunicação entre veículos autônomos. Cada cliente possui uma ou mais aplicações, sendo o número de clientes um parâmetro configurável. O início das aplicações em cada cliente obedece a um processo de Poisson independente com valor médio configurável. Dessa forma, a taxa de chegada de novos tráfegos na rede também obedece a um processo de Poisson com taxa diretamente proporcional ao número de clientes ativos. A duração de cada tráfego depende do tipo de aplicação. Como exemplos, a duração média de uma chamada VoIP é de 100 segundos (MELO et al., 2010)(GUO; LIU; ZHU, 2007), enquanto a duração média de um vídeo bufferizado é de 90 segundos de conteúdo (YOUTUBE, 2015).

3.2 Gerência de tráfego

Antes do início de cada aplicação, o *Controlador* é requisitado para um processo de admissão de tráfego e instalação pró-ativa das regras. No início de uma simulação o *Controlador* instala no *UpLink Switch* e no *DownLink Switch* as regras de encaminhamento de pacotes de acordo com a política definida. Quando uma requisição de novo tráfego chega ao *Controlador*, ele decidirá qual *switch* (*Hardware Switch* ou *Software Switch*) deve receber o novo tráfego, de acordo com a política vigente. As políticas adotadas são discutidas nas Seções 3.2.1 e 3.2.2.

Antes da requisição ser aceita, o *Controlador* verifica se o *switch* de destino possui recursos disponíveis para uso. O primeiro recurso analisado é o espaço na tabela de fluxo. Se a razão do número de regras instaladas pelo tamanho da tabela exceder um limitante denominado *Limiar de bloqueio*, então o tráfego é bloqueado. Essa medida é necessária pois se não houver espaço na tabela para a instalação de novas regras, o *switch* não conseguirá lidar com aquele tráfego e todos os pacotes serão descartados. O segundo recurso analisado é a capacidade de processamento do *switch*. Se a razão entre o processamento atual do *switch* e sua capacidade total exceder o *Limiar de bloqueio* estabelecido, o *Controlador* pode então bloquear ou não o tráfego. A aceitação do tráfego pode acarretar em uma carga superior a 100% e eventual descarte aleatório de pacotes. O bloqueio garante uma melhor qualidade no processamento dos tráfegos instalados, apesar de restringir o número de aplicações ativas na rede. A decisão por bloquear ou não uma requisição quando há sobrecarga de processamento no *switch* é definida pela estratégia de *bloqueio de sobrecarga*. Uma vez que o tráfego é aceito pelo mecanismo de admissão, o *Controlador* instala no *switch* destino, *Hardware Switch* ou *Software Switch*, as regras que fazem o papel dos TFTs num *Gateway PG-W*.

No cenário avaliado neste trabalho são consideradas duas políticas para determinar o encaminhamento do tráfego pelo *Hardware Switch* ou pelo *Software Switch*: Uma política chamada de arbitrária, utilizando o IP do cliente no encaminhamento do tráfego, e outra política direcionada para o uso adequado dos recursos nos *switches*. Elas são descritas nas seções abaixo.

3.2.1 Política Arbitrária

Nesta política os tráfegos são balanceados entre o *Hardware Switch* e o *Software Switch* através da comparação do número final do endereço IP do *Cliente*. Tráfegos originados de um *Cliente* com número de IP com final par são enviados para o *Hardware Switch* enquanto os de final ímpar são enviados para o *Software Switch*, sendo essa uma escolha arbitrária. Esta política faz com que o uso dos recursos nos *switches* se equilibre, permitindo avaliar como as características de cada *switch* afetam o desempenho dos tráfegos.

A implantação da política acontece através da instalação de regras OpenFlow nas

tabelas de fluxo do *UpLink Switch* e *DownLink Switch*. Em ambos *switches* aplica-se o par IP/máscara 0.0.0.0/0.0.0.1 sobre o endereço IP do cliente para identificar um IP par e 0.0.0.1/0.0.0.1 para identificar um IP ímpar. Esta política demanda a instalação de apenas duas regras no *UpLink Switch* e *DownLink Switch*. Como esses *switches* são preferencialmente implementados em *hardware*, o uso de apenas duas regras vai ao encontro do tamanho reduzido da tabela de fluxo destes *switches*.

3.2.2 Política Direcionada

Esta política tem como objetivo aumentar o desempenho da rede, explorando melhor os recursos disponíveis no *Hardware Switch* e o *Software Switch*. Inicialmente somente uma regra é instalada no *UpLink Switch* e no *DownLink Switch*, encaminhando todos os tráfegos para o *Software Switch*, que possui uma tabela de fluxos grande o suficiente para acomodar todos os tráfegos da rede.

Porém, como citado anteriormente, *switches* baseados em *software* apresentam um atraso maior no encaminhamento de pacotes e uma capacidade total de processamento menor quando comparados à *switches* baseados em *hardware*, fazendo com que o limite da sua capacidade de processamento seja atingido rapidamente. Para lidar com essa limitação, usamos a seguinte abordagem: Periodicamente o *Controlador* constrói uma lista dos tráfegos ativos no *Software Switch*, ordenando-os de maneira decrescente em relação à vazão. Em seguida, ele verifica se há disponibilidade de recursos (capacidade de processamento e espaço na tabela de fluxo) no *Hardware Switch*. Se houver, o *Controlador* consulta a lista e move o tráfego de maior vazão do *Software Switch* para o *Hardware Switch*. O *Controlador* atualiza as métricas de recursos do *Hardware Switch* e repete o processo enquanto houver recursos disponíveis no *Hardware Switch*. Desta forma usamos a maior capacidade de processamento do *Hardware Switch* para os tráfegos que melhor aproveitam essa característica e reduzimos a carga de processamento no *Software Switch*.

A transferência do tráfego de um *switch* para outro é feita através da instalação no *Hardware Switch* de uma regra *TFT* exatamente igual a existente no *Software Switch*. Na sequência, é necessário que uma nova regra específica para aquele tráfego seja instalada no *UpLink Switch* e *DownLink Switch* para encaminhá-lo ao *Hardware Switch*. Esta regra

tem prioridade maior à regra instalada inicialmente, que encaminha todos os tráfegos para o *Software Switch*. Após a transferência do tráfego e atualização do *UpLink Switch* e *DownLink Switch*, a regra pode ser finalmente removida do *Software Switch*. A ordem dessas operações minimiza as perdas de pacotes durante a transição.

A *Política Direcionada* demanda a instalação de no mínimo uma regra no *UpLink Switch* e *DownLink Switch* e no máximo o mesmo número de regras instaladas no *Hardware Switch*. Isso garante que as tabelas de fluxo desses *switches* não excedam sua capacidade, viabilizando sua implementação com *switches* de *hardware*.

4 Resultados

4.1 Configuração do simulador

Neste capítulo são apresentadas as configurações usadas no simulador e os resultados obtidos nas simulações. Os experimentos apresentados neste capítulo foram realizados no *ns-3* utilizando o módulo *OFSwitch13*. O código implementado está disponível para acesso em ⟨https://www.github.com/rafaelgmotta/tccsi⟩ e os arquivos estão organizados da seguinte maneira:

- *applications*: Conjunto de classes que modelam tipos diferentes de aplicações;
- *application-helper e traffic-helper*: Classes responsáveis por automatizar a instalação e configuração das aplicações nos clientes;
- *custom-controller*: Implementação da lógica do controlador OpenFlow específica para o cenário avaliado;
- *main*: Configuração da topologia de rede para a simulação;
- *traffic-manager*: Responsável pelo início e fim dos tráfegos das aplicações, interagindo com o controle de admissão no *Controlador*;
- *traffic-statistics*: Responsável pelo monitoramento do tráfego e coleta de estatísticas.

Como já discutido, é esperado que *UpLink Switch* e *DownLink Switch* sejam implementados em *hardware* de alto desempenho e por os parâmetros indicados na Tabela 4.1 foram utilizados durante as simulações. Esses parâmetros oferecem para o *UpLink Switch* e *DownLink Switch* uma capacidade muito maior que a necessária para lidar com os tráfegos do experimento. Isso garante que eles não se tornem um gargalo e interfiram nos resultados da simulação, já que o interesse é a análise do desempenho do *Hardware Switch* e do *Software Switch*.

Tabela 4.1: Parâmetros para configuração do *UpLink Switch* e *DownLink Switch*.

Parâmetro	**UpLink Switch** e **DownLink Switch**
Capacidade de processamento	100 Gbps
Tamanho da tabela de fluxo	65.535 regras
Tempo de busca em TCAM	20 microsegundos

Para o *Hardware Switch* e *Software Switch* os parâmetros foram ajustados de maneira proporcional aos valores encontrados na literatura (CHAVES et al., 2015; COSTA et al., 2017), dimensionados adequadamente para o cenário avaliado neste trabalho. Os valores adotados estão indicados na Tabela 4.2.

Tabela 4.2: Parâmetros para configuração do *Hardware Switch* e *Software Switch*.

Parâmetro	**Hardware Switch**	**Software Switch**
Capacidade de processamento	2 Gbps	300 Mbps
Tamanho da tabela de fluxo	1.024 regras	8.192 regras
Tempo de busca em TCAM	20 microsegundos	160 microsegundos

Abaixo são listados outros parâmetros configurados nas simulações:

- *Limiar de bloqueio*: Limitante para ocupação das tabelas de fluxo e processamento dos *switches*. Neste trabalho foi adotado o valor de 90% por ser um valor próximo do máximo e ao mesmo tempo oferecer uma margem de segurança para eventuais picos de uso dos recursos que possam ocorrer durante o experimento;

- *Bloqueio de sobrecarga*: Estratégia de bloqueio do tráfego após a capacidade de processamento do *switch* ultrapassar o *Limiar de bloqueio*. Se configurado como falso, tráfegos que excedam a capacidade de processamento não serão bloqueados, o que pode ocasionar perda de pacotes em determinados momentos da simulação.

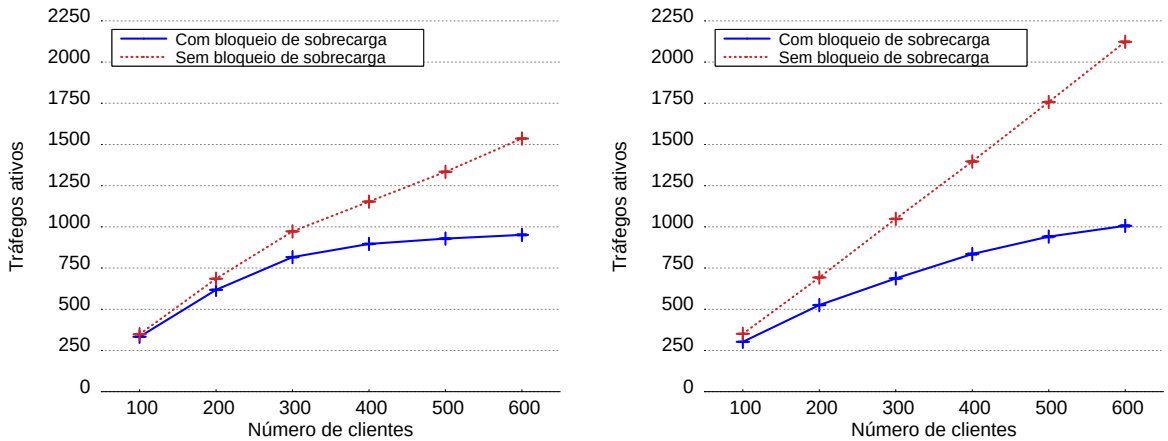
- *Intervalo de redirecionamento*: Periodicidade com que o *Controlador* analisa as cargas do *Software Switch* e *Hardware Switch* e, quando oportuno, redireciona o

tráfego de um para o outro. Adotamos o valor de 15 segundos para que essas operações aconteçam regularmente, mas sem sobrecarregar a rede.

4.2 Avaliação dos resultados

Para avaliação do cenário foram feitos experimentos considerando todas as combinações entre as políticas de gerência de tráfego (*Política Arbitrária* e *Política Direcionada*) e as estratégias de bloqueio de sobrecarga. Os gráficos desta seção apresentam os valores médios durante o período estável da rede, para diferentes quantidades de clientes: 100, 200, 300, 400, 500 e 600 clientes. O período estável da rede pode ser caracterizado pela equivalência entre o número de novas requisições de tráfego e o número de tráfegos que estão sendo encerrados num intervalo de tempo. As simulações foram configuradas para uma duração total de 500 segundos e os gráficos apresentam um intervalo de confiança de 95% para os resultados. As oito métricas seguintes foram analisadas: Número de tráfegos ativos, taxa de bloqueio dos tráfegos, atraso médio nos pacotes, *jitter* dos pacotes, uso de carga da CPU dos *switches*, uso da tabela de fluxo dos *switches*, perda de pacotes e vazão da rede. nota de rodapé –Para os gráficos x,y,z o intervalo de confiança de 95

A primeira métrica avaliada é o número de tráfegos ativos (Figura 4.1). Considerando a estratégia sem bloqueio de sobrecarga, podemos observar que o número de tráfegos ativos na *Política Arbitrária* tem seu crescimento afetado exclusivamente pelo limite no



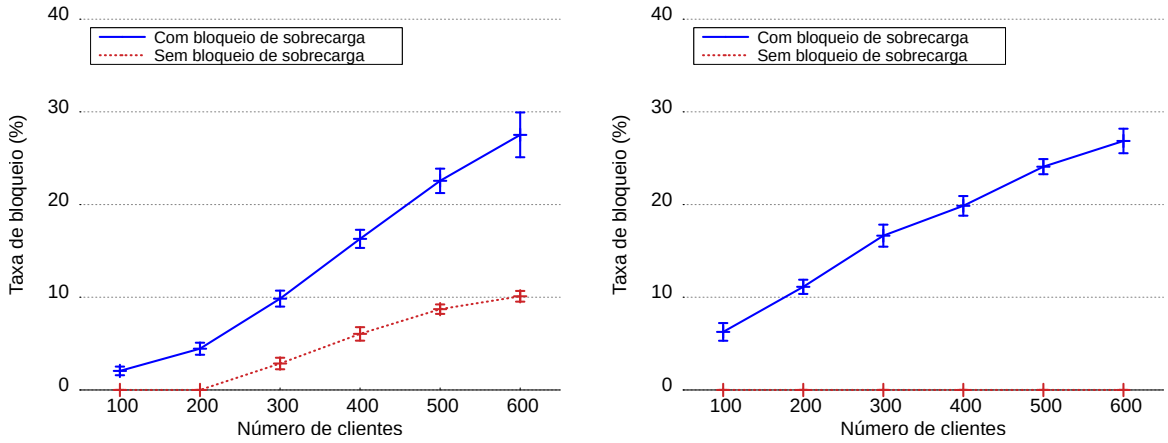
(a) Política Arbitrária (b) Política Direcionada

Figura 4.1: Número médio de tráfegos ativos na rede por número de clientes de acordo com as políticas de gerência de tráfego e estratégias de bloqueio adotadas.

tamanho da tabela de fluxo do *Hardware Switch*, que ao chegar a seu limite começa a bloquear os tráfegos. Já na *Política Direcionada*, todos os tráfegos são inicialmente direcionados para o *Software Switch*, cuja tabela de fluxos possui tamanho suficiente para acomodar a chegada dos tráfegos. Por isso vemos um crescimento linear para todas as quantidades de clientes.

Quando o bloqueio de sobrecarga é considerado, em ambas políticas o número de tráfegos ativos cresce e se estabiliza de forma semelhante por causa das restrições tanto no tamanho das tabelas como no processamento dos *switches*, conforme apresentado nas Figuras 4.5 e 4.6.

A segunda métrica considerada é a taxa de bloqueio dos tráfegos (Figura 4.2). Essa métrica se refere a porcentagem dos tráfegos que o *Controlador* bloqueou por falta de recursos nos *switches*. Quando não há bloqueio de sobrecarga, a limitação no tamanho da tabela de fluxo do *Hardware Switch* é a única responsável pelo crescimento da taxa de bloqueio na *Política Arbitrária*. Como a tabela de fluxo do *Software Switch* consegue acomodar regras para todos os tráfegos, não há bloqueio de tráfegos na *Política Direcionada*.



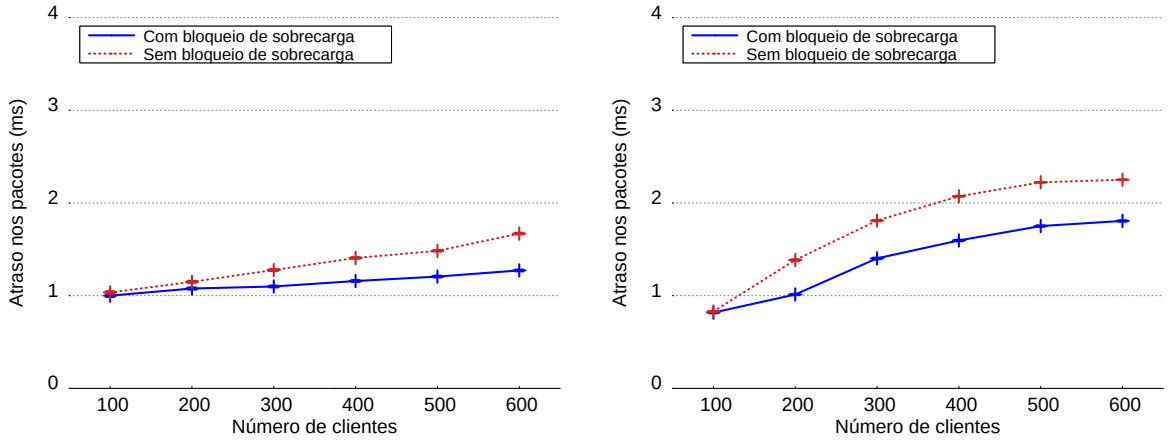
(a) Política Arbitrária (b) Política Direcionada

Figura 4.2: Bloqueio de tráfegos ativos por número de clientes de acordo com as políticas de gerência de tráfego e estratégias de bloqueio adotadas.

Quando o bloqueio de sobrecarga é considerado, a *Política Arbitrária* apresenta menor taxa de bloqueio nas simulações que consideram até 400 clientes. Isso pode ser explicado pelo fato da *Política Direcionada* fazer uso prioritário do *Software Switch* que,

com sua capacidade de processamento inferior ao *Hardware Switch*, pode ultrapassar o *Limiar de bloqueio* estabelecido entre duas operações de redirecionamento de tráfego consecutivas, aumentando a taxa de bloqueio. Porém, nas simulações com 500 e 600 clientes essa diferença na capacidade de processamento dos *switches* tem um impacto reduzido e as duas políticas começam a apresentar comportamento semelhante. Podemos observar ainda que a taxa de bloqueio na *Política Direcionada* tende a crescer mais lentamente do que na *Política Arbitrária*. Isso sugere que a *Política Direcionada* mostra-se mais eficiente apresentando uma taxa de bloqueio menor com o crescimento do número de clientes.

A terceira métrica avaliada é o atraso nos pacotes (Figura 4.3). Apesar de uma diferença pequena, podemos perceber que com o uso do bloqueio de sobrecarga o atraso é menor nas duas políticas, já que o bloqueio limita a chegada de novos tráfegos e garante que os recursos disponíveis nos *switches* sejam direcionados para os tráfegos existentes. Entre as duas políticas, a *Política Direcionada* apresenta um desempenho um pouco inferior devido a sua característica de uso predominante do *Software Switch*. Entretanto, esse aumento é pouco significativo no desempenho da rede.

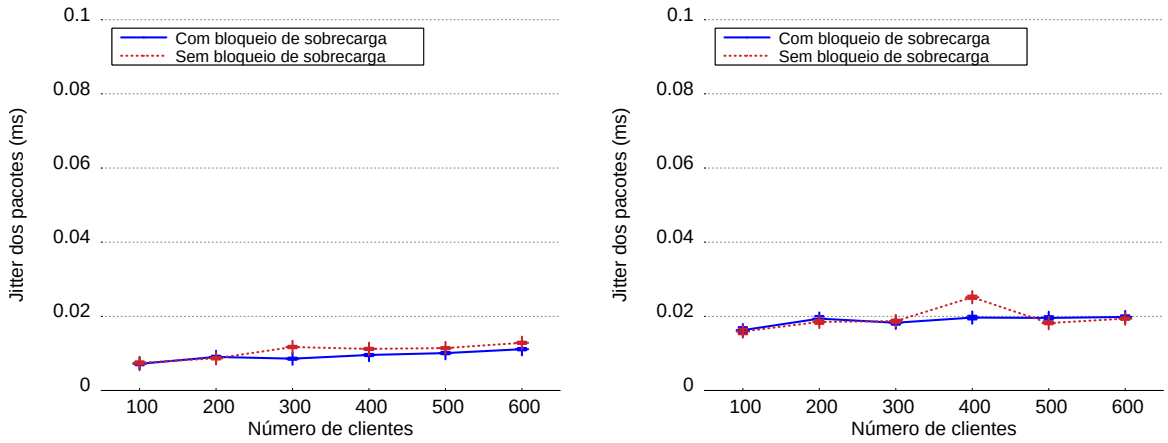


(a) Política Arbitrária (b) Política Direcionada

Figura 4.3: Atraso nos pacotes por número de clientes de acordo com as políticas de gerência de tráfego e estratégias de bloqueio adotadas.

A quarta métrica avaliada é o jitter dos pacotes (Figura 4.4). Considerando a escala ampliada do gráfico é possível observar que não há diferença significativa entre as quatro combinações avaliadas.

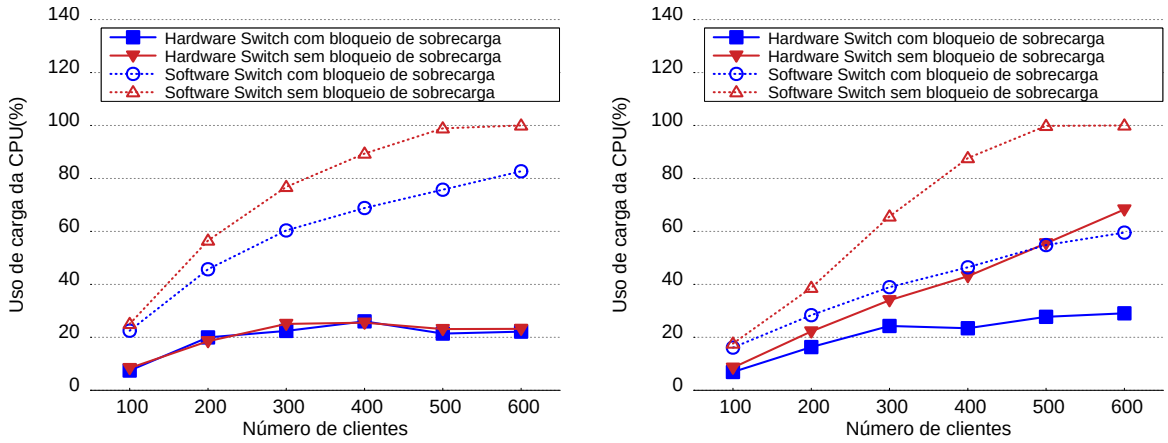
A quinta métrica avaliada é o uso de CPU dos *switches* (Figura 4.5). Os intervalos



(a) Política Arbitrária (b) Política Direcionada

Figura 4.4: Jitter dos pacotes por número de clientes de acordo com as políticas de gerência de tráfego e estratégias de bloqueio adotadas.

de confiança do gráfico, muito pequenos, foram omitidos para melhorar sua legibilidade. Como a capacidade de processamento do *Software Switch* é menor que do *Hardware Switch*, então o percentual de uso de sua CPU é maior nas duas políticas.



(a) Política arbitrária (b) Política Direcionada

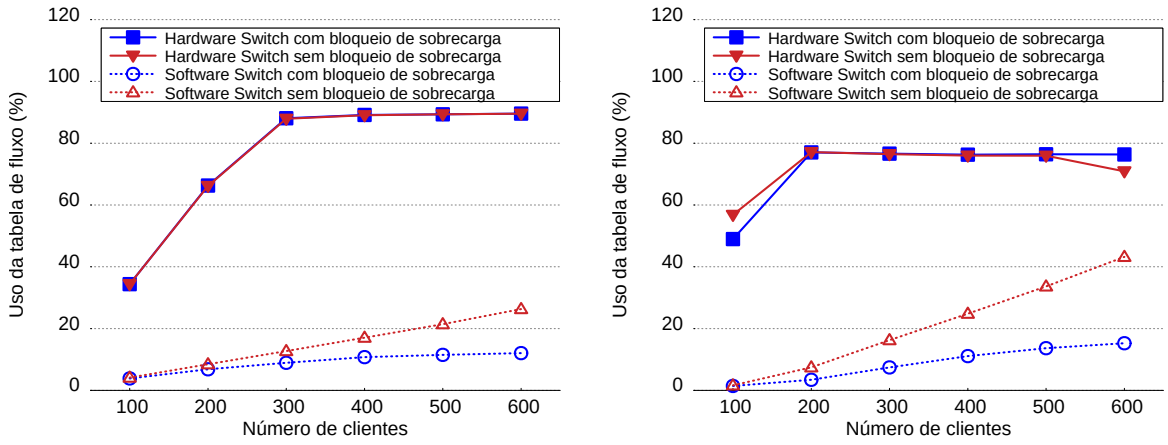
Figura 4.5: Uso de carga da CPU dos *switches* por número de clientes e tipo de *switch* de acordo com as políticas de gerência de tráfego e estratégias de bloqueio adotadas.

Podemos observar que na *Política Arbitrária* o uso ou não do bloqueio de sobrecarga não afeta de forma significativa o uso da CPU do *Hardware Switch*, que não ultrapassa 30%. Isso acontece pela falta de espaço da tabela de fluxo, que limita a chegada de novos fluxos antes que o uso da CPU cresça. Já para o *Software Switch* o bloqueio afeta diretamente o uso da CPU, que chega a 100% nas simulações sem bloqueio com 500 e 600 clientes. Como esperado, o uso de sua CPU cresce mais rápido quando comparado

ao *Hardware Switch*.

O uso de bloqueio de sobrecarga (linhas azuis) não afeta o uso de CPU do *Hardware Switch* nas duas políticas, mas afeta diretamente no *Software Switch*. O crescimento do uso de CPU é mais lento na *Política Direcionada* por ter um comportamento instável com momentos de alta carga, que geram bloqueio, e momentos de baixa carga com ociosidade de recursos, como é possível ver na Figura 4.7a. Quando o uso do bloqueio de sobrecarga é desconsiderado (linhas vermelhas), podemos observar um melhor aproveitamento do *Hardware Switch* na *Política Direcionada*. Quando o *Software Switch* atinge 100% de sua capacidade de processamento nas simulações com 500 e 600 clientes, o *Hardware Switch* ainda não atingiu sua capacidade máxima, porém não é possível explorá-la mais devido o esgotamento de sua tabela de fluxo.

A sexta métrica avaliada é a porcentagem de uso das tabelas de fluxo dos *switches* (Figura 4.6). Novamente os intervalos de confiança do gráfico, muito pequenos, foram omitidos para melhorar sua legibilidade. Como o *Hardware Switch* possui uma tabela de tamanho reduzido o uso de suas tabelas cresce rapidamente.



(a) Política Arbitrária (b) Política Direcionada

Figura 4.6: Uso das tabelas de fluxo por número de clientes de acordo com as políticas de gerência de tráfego e estratégias de bloqueio adotadas.

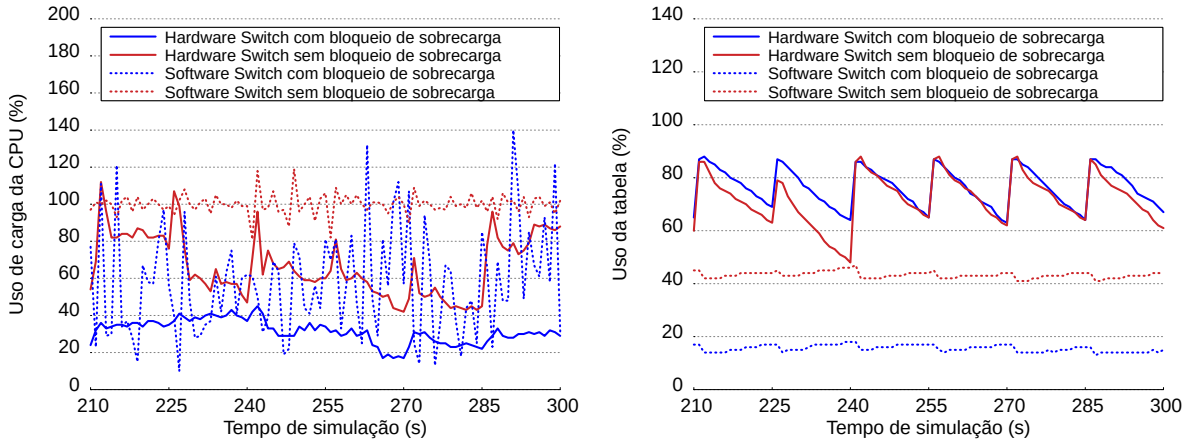
Podemos observar na *Política Arbitrária* que o uso da tabela de fluxo do *Hardware Switch* mantêm-se no limiar estabelecido de 90%, independente da estratégia de bloqueio de sobrecarga adotada. Devido a limitação da capacidade de processamento do *Software Switch*, o uso do bloqueio de sobrecarga afeta-o de forma mais significativa, pois reduz a

quantidade de tráfegos ativos na rede.

Pela proposta da *Política Direcionada* de instalar todos os tráfegos inicialmente no *Software Switch*, podemos observar um crescimento acelerado no uso de sua tabela de fluxo apenas quando sem bloqueio de sobrecarga: Com o bloqueio ativo, parte dos tráfegos acaba sendo bloqueada por excesso de carga, fazendo com que o uso da tabela não alcance seu máximo, de forma semelhante à *Política Arbitrária*.

O uso médio da tabela de fluxo do *Hardware Switch* mantêm-se na faixa de 80% e é pouco afetado pelo uso ou não do bloqueio de sobrecarga. Esse valor médio não atinge o limiar estabelecido de 90% pois a porcentagem do uso instantâneo da tabela varia consideravelmente entre as operação de redirecionamento de tráfego.

Para compreender melhor o efeito causado pelo processo de redirecionamento de tráfegos da *Política Direcionada* no uso de carga da CPU e no uso da tabela de fluxo, a Figura 4.7 mostra o comportamento instantâneos destas métricas ao longo do tempo, considerando o momento de estabilidade da rede para simulações com 500 clientes.



(a) Uso de carga da CPU

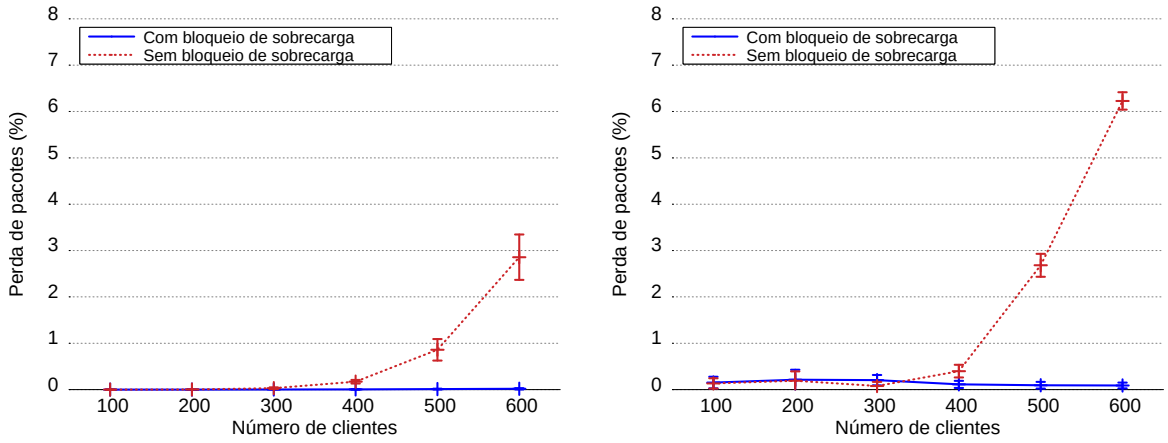
(b) Uso da tabela de fluxo

Figura 4.7: Uso instantâneo de carga da CPU e da tabela de fluxo ao longo do tempo para a *Política Direcionada*.

Podemos observar no uso da CPU sem bloqueio de sobrecarga (linhas vermelhas) que a cada *Intervalo de redirecionamento* (15 segundos) o *Hardware Switch* recebe novos tráfegos vindos do *Software Switch*, portanto, vemos um pico no uso de sua CPU. O *Software Switch* tem seu uso de CPU sempre próximo a 100%. Com o bloqueio de sobrecarga (linhas azuis) podemos ver o comportamento instável do uso de CPU do *Software Switch*,

explicado pelo fato de receber todo os tráfegos, processá-los e periodicamente direcionar alguns deles para o *Hardware Switch*. Podemos ver também que, independente da estratégia de bloqueio, o *Hardware Switch* mantêm um alto uso de sua tabela de fluxo devido ao seu tamanho reduzido. Sempre durante uma operação de redirecionamento de tráfegos, o *Controlador* preenche a tabela de fluxo do *Hardware Switch* até que sua ocupação da tabela ou a capacidade de processamento do *switch* atinja o *Limiar de bloqueio*.

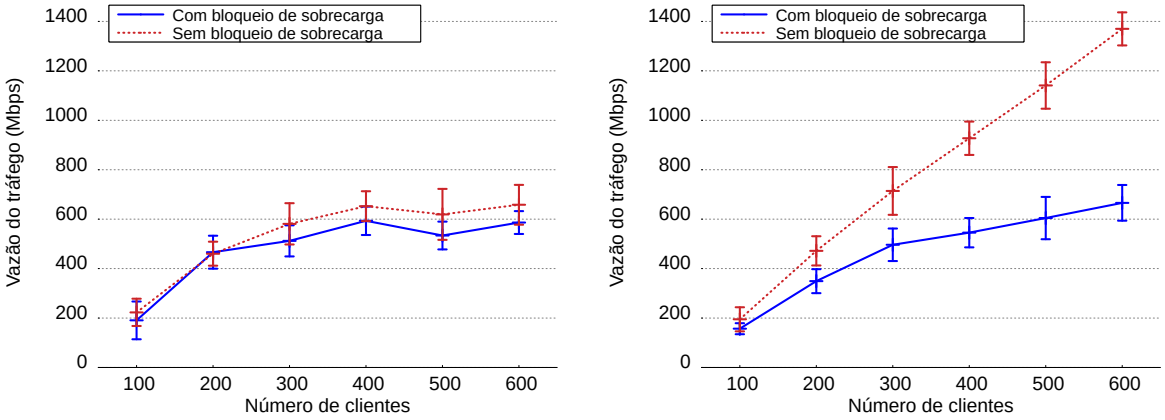
A sétima métrica avaliada é a porcentagem de perda de pacotes ocorridas exclusivamente por sobrecarga de tráfego nos *switches* OpenFlow (Figura 4.8). O bloqueio de sobrecarga garante uma perda quase nula em ambas políticas. Sem o bloqueio de sobrecarga, a *Política Direcionada* apresenta um crescimento maior nas perdas de pacotes que a *Política Arbitrária* em cenários sobrecarregados. Pode-se atribuir esse crescimento aos pacotes perdidos durante a operação de redirecionamento de tráfegos entre os *switches* e possíveis sobrecargas momentâneas no uso de CPU do *Software Switch* durante o intervalo entre as operações de redirecionamento de tráfego.



(a) Política Arbitrária (b) Política Direcionada

Figura 4.8: Perda de pacotes por número de clientes de acordo com as políticas de gerência de tráfego e estratégias de bloqueio adotadas.

A oitava e última métrica avaliada é a vazão agregada da rede (Figura 4.9). Na *Política Arbitrária* o bloqueio de sobrecarga não afeta a vazão de forma considerável, que demonstra um comportamento estável a partir de 300 clientes. Com o bloqueio de sobrecarga, a vazão é similar entre as duas políticas, apresentando uma tendência de crescimento um pouco maior na *Política Direcionada* em relação à *Política Arbitrária*.



(a) Política Arbitrária (b) Política Direcionada

Figura 4.9: Vazão por número de clientes de acordo com as políticas de gerência de tráfego e estratégias de bloqueio adotadas.

Na *Política Direcionada*, quando o bloqueio de sobrecarga não é considerado, a vazão tem um desempenho muito superior à *Política Arbitrária*. Isso acontece pelo melhor aproveitamento dos recursos onde cada *switch* se destaca: capacidade de processamento do *Hardware Switch* e tamanho da tabela de fluxo do *Software Switch*. Entretanto, esse ganho vem associado à uma perda maior de pacotes nesta configuração.

5 Conclusão

Este trabalho explorou o uso de SDN e NFV para propor uma abordagem de gerência de tráfego baseada na otimização de uso dos recursos dos *switches* OpenFlow. O uso dessas tecnologias permite a flexibilidade necessária para o crescimento das redes de computadores ao separar o *software* do *hardware* onde ele é executado. Esse *hardware*, geralmente de alto custo e funções limitadas, pode ser substituído por uma infraestrutura compartilhada e otimizada, reduzindo custos de operação e manutenção. O protocolo OpenFlow é o responsável por trazer a SDN ao nível de enlace da rede, sendo possível implementá-lo em *switches* físicos ou virtualizá-lo sobre *software*.

O estudo de caso deste trabalho foi a virtualização do plano de dados de um *gateway* P-GW de uma rede LTE utilizando *switches* OpenFlow. Em nosso cenário fizemos uso de uma combinação híbrida dos switches, usando tanto a versão implementada em *hardware* quanto a virtualizada em *software*.

Foram propostas duas políticas de gerência de tráfego, uma política que balanceia o tráfego entre os dois *switches* de forma arbitrária (*Política Arbitrária*) e uma *Política Direcionada*, focada no aproveitamento dos recursos abundantes de cada *switch*. A *Política Arbitrária* divide os tráfegos quantitativamente entre os *switches*, enquanto a *Política Direcionada* envolve periódicos redirecionamentos de tráfegos entre os *switches* para otimizar o uso dos recursos: tamanho da tabela de fluxo no *Software Switch* e capacidade de processamento no *Hardware Switch*.

Como mostram os resultados da Seção 4.2, a *Política Direcionada* explora melhor os recursos abundantes em cada tipo de *switch*, aumentando a vazão da rede. Porém, o aumento da vazão é associado com o aumento de perdas de pacotes, ocasionadas pela dinâmica do mecanismo de redirecionamento de tráfego. Através da análise dos resultados podemos observar que o uso da estratégia de bloqueio de sobrecarga causa ganhos pequenos na vazão e uma redução sutil na taxa de bloqueio para cenários sobrecarregados. Isso é influenciado pelos períodos entre os redirecionamentos de tráfego, o que sugere que o uso de *Intervalos de redirecionamento* menores podem gerar resultados melhores,

porém associados à uma sobrecarga de controle maior na rede. Quando o bloqueio por sobrecarga não é utilizado, mais tráfegos são acomodados nos *switches*, o que implica no aumento da vazão agregada da rede e também no aumento de perdas de pacotes em cenários sobrecarregados.

Dado as limitações encontradas nos resultados, entendemos que uma análise mais detalhada do impacto na rede durante o redirecionamento dos tráfegos de um *switch* para o outro pode trazer benefícios para ajustar melhor os parâmetros do simulador visando diminuir a perda de pacotes e a taxa de bloqueio em cenários sobrecarregados. Estas análises são consideradas como possíveis trabalhos futuros.

Bibliografia

ABDELWAHAB, S. et al. Network function virtualization in 5G. *IEEE Communications Magazine*, v. 54, n. 4, p. 84–91, Apr 2016.

AKYILDIZ, I. F.; LIN, S.-C.; WANG, P. Wireless software-defined networks (W-SDNs) and network function virtualization (NFV) for 5G cellular systems: An overview and qualitative evaluation. *Computer Networks*, v. 93, n. 1, p. 66–79, Dec 2015.

AN, X.; KIESS, W.; PEREZ-CAPARROS, D. Virtualization of cellular network EPC gateways based on a scalable SDN architecture. In: IEEE. *Global Communications Conference (GLOBECOM)*. [S.l.], 2014. p. 2295–2301.

ARSLAN, M. Y.; SUNDARESAN, K.; RANGARAJAN, S. Software-defined networking in cellular radio access networks: Potential and challenges. *IEEE Communications Magazine*, v. 53, n. 1, p. 150–156, Jan 2015.

BASTA, A. et al. A virtual SDN-Enabled LTE EPC architecture: A case study for S-/P-Gateways functions. In: IEEE. *SDN for Future Networks and Services (SDN3FNS)*. [S.l.], 2013. p. 1–7.

BASTA, A. et al. Applying NFV and SDN to LTE mobile core gateways, the functions placement problem. In: ACM. *Workshop on All Things Cellular: Operations, Applications, & Challenges (AllThingsCellular)*. [S.l.], 2014. p. 33–38.

CHAVES, L. J. et al. Integrating OpenFlow to LTE: some issues toward Software-Defined Mobile Networks. In: IEEE. *International Conference on New Technologies, Mobility and Security (NTMS)*. [S.l.], 2015. p. 1–5.

CHAVES, L. J.; GARCIA, I. C.; MADEIRA, E. R. M. OFSwitch13: Enhancing ns-3 with OpenFlow 1.3 support. *Proceedings of the 8th Workshop on ns-3 (WNS3), p. 33-40. ACM, 2016.*, 2016.

CHAVES, L. J.; GARCIA, I. C.; MADEIRA, E. R. M. An adaptive mechanism for LTE P-GW virtualization using SDN and NFV. *13th International Conference on Network and Service Management, CNSM 2017, Tokyo, Japan*, Nov 2017.

COSTA, L. C. et al. Performance evaluation of OpenFlow data planes. In: IFIP/IEEE. *Symposium on Integrated Network Management (IM)*. [S.l.], 2017. p. 470–475.

ETSI NFV 002. *Network Function Virtualisation (NFV) LTE; Architectural Framework*. [S.l.], 2014.

GLOBE NEWS. 2018. Online. Disponível em ⟨https://globenewswire.com/news-release/2018/03/22/1444749/0/en/LTE-Most-Widely-Used-Cellular-Technology-Worldwide-at-End-of-2017.html⟩.

GUO, J.; LIU, F.; ZHU, Z. Estimate the call duration distribution parameters in GSM system based on K-L divergence method. In: IEEE. *International Conference on Wireless Communications, Networking and Mobile Computing (WiCom)*. [S.l.], 2007. p. 2988–2991.

HAMPEL, G.; STEINER, M.; BU, T. Applying software-defined networking to the telecom domain. In: IEEE. *Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. [S.l.], 2013. p. 133–138.

HURD, B. *GSoC 2010 OpenFlow*. 2010. Online. Disponível em ⟨http://www.nsnam.org/wiki/GSOC2010OpenFlow⟩.

KIESS, W.; AN, X.; BEKER, S. Software-as-a-service for the virtualization of mobile network gateways. In: IEEE. *Global Communications Conference (GLOBECOM)*. [S.l.], 2015. p. 1–6.

LI, Y.; CHEN, M. Software-defined network function virtualization: A survey. *IEEE Access*, v. 3, p. 2542–2553, Dec 2015.

MCKEOWN, N. et al. OpenFlow: Enabling innovation in campus networks. *ACM SIGCOMM Computer Communication Review*, v. 38, n. 2, p. 69–74, Apr 2008.

MELO, P. O. S. V. d. et al. Surprising patterns for the call duration distribution of mobile phone users. In: *Machine Learning and Knowledge Discovery in Databases*. [S.l.]: Springer Berlin Heidelberg, 2010. (Lecture Notes in Computer Science, v. 6323), p. 354–369.

MIJUMBI J. SERRAT, J.-L. G. N. B. F. D. R.; BOUTABA, R. Network function virtualization: State-of-the-art and research challenges. *IEEE Communications Surveys Tutorials, vol. 18, no. 1, pp.236–262*, 2015.

Network Simulator 3. 2018. Online. Disponível em ⟨https://www.nsnam.org/about⟩.

NGUYEN, V.-G. et al. SDN/NFV-based mobile packet core network architectures: A survey. *IEEE Communications Surveys & Tutorials*, v. 19, n. 3, p. 1567–1602, 2017.

NGUYEN, V.-G.; DO, T.-X.; KIM, Y. SDN and virtualization-based LTE mobile network architectures: A comprehensive survey. *Wireless Personal Communications*, p. 1–38, Aug 2015.

Open Networking Foundation. *Software-Defined Networking: The New Norms for Networks*. 2012. ONF White Paper.

Open Networking Foundation. *OpenFlow Enabled Mobile and Wireless Networks*. 2013. ONF Solution Brief.

Open Networking Foundation. *OpenFlow-enabled Transport SDN*. 2014. ONF Solution Brief.

Open vSwitch. 2017. Online. Disponível em ⟨http://openvswitch.org⟩.

OpenFlow 1.5.1. *OpenFlow Switch Specification*. [S.l.], 2015.

PENTIKOUSIS, K.; WANG, Y.; HU, W. MobileFlow: Toward software-defined mobile networks. *IEEE Communications Magazine*, v. 51, n. 7, p. 44–53, Jul 2013.

YOUTUBE. *YouTube statistics*. 2015. Online. Disponível em ⟨https://www.youtube.com/yt/press/statistics.html⟩.