Arquitetura de Provisão de Qualidade de Serviço para Aplicações Distribuídas de Alto Desempenho em Redes Definidas por Software

Alexandre T. Oliveira[1], Bruno José C. A. Martins[1], Antônio Tadeu A. Gomes[2], Artur Ziviani[2], Marcelo F. Moreno[1], Alex B. Vieira[1]

[1] Departamento de Ciência da Computação – Universidade Federal de Juiz de Fora (UFJF)

[2]Laboratório Nacional de Computação Científica (LNCC)

{alexandre.tavares,brunomartins,moreno}@ice.ufjf.br,

{atagomes,ziviani}@lncc.br,alex.borges@ufjf.edu.br

Abstract. *The specification of quality of service (QoS) requirements such as delay, jitter and throughput in traditional data networks is limited by the high administrative cost of these environments. On the other hand, the software-defined networks (SDN) paradigm, with its global vision and its higher level of programmability, simplifies the management of the whole infrastructure of the data networks. In this sense, SDN can provide simple and effective QoS provisioning mechanisms in these networks in order to meet the needed requirements. Thus, in this paper, we propose a QoS provision architecture exploiting the capabilities of the SDN model. The approach consists in providing software components that allow the specification of classes of service, in addition to negotiating the QoS requirements between applications and the network controller. In this platform, the SDN controller monitors the environment and adjusts the network through resource reservation and traffic prioritization. A proof-of-concept has been deployed and our results show that the additional routines present low overhead, whereas a reduction of up to 47% in transfer times is observed.*

Resumo. *A especificação de requisitos de qualidade de serviço (QoS) como atraso, jitter e vazão em redes de dados tradicionais é limitada pelo alto custo administrativo desses ambientes. Por outro lado, o paradigma de redes definidas por software (SDN), com sua visão global e seu maior nível de programabilidade, simplifica o gerenciamento de toda a infraestrutura das redes de dados. Nesse sentido, SDN pode fornecer mecanismos simples e efetivos de provisionamento de QoS nessas redes a fim de atender aos requisitos exigidos. Assim, neste artigo, propomos uma arquitetura de provisão de QoS explorando as capacidades do modelo SDN. A abordagem consiste em disponibilizar componentes de software que permitam a especificação de classes de serviço, além da negociação dos requisitos de QoS entre as aplicações e o controlador de rede. Nessa plataforma, o controlador SDN monitora o ambiente e ajusta a rede através de reserva de recursos e priorização de tráfego. Uma prova de conceito foi implementada e nossos resultados mostram que as rotinas adicionais apresentam baixa sobrecarga, mostrando também redução de até 47% nos tempos de transferência.*

1. Introdução

Atualmente há uma demanda crescente por aplicações distribuídas de alto desempenho que exigem requisitos rígidos das infraestruturas de redes sobre as quais elas operam. Sistemas de saúde à distância, serviços de *streaming* de conteúdo multimídia, aplicações interativas em tempo real como jogos *online* e aplicações de processamento paralelo em *data centers* (ambientes HPC) são alguns dos exemplos de plataformas que requerem garantias de parâmetros de qualidade de serviço (*Quality of Service* – QoS). O atendimento desses requisitos de maneira fim-a-fim impõe enormes desafios às redes de dados atuais. Tendo em vista que a maioria dessas redes já implantadas são "ossificadas" e possuem recursos escassos, torná-las mais flexíveis e adaptáveis a sistemas distribuídos de alto desempenho é de grande importância para melhorar a eficiência dessas plataformas e aprimorar a qualidade de experiência (*Quality of Experience* – QoE) de seus usuários.

As possibilidades de especificação de requisitos de QoS em redes tradicionais são limitadas devido ao controle complexo desses ambientes [Humernbrum et al. 2014]. A modificação da infraestrutura existente possui um alto custo administrativo, onde cada subsistema componente do serviço deve ter seus recursos gerenciados no intuito de garantir aos usuários o nível de qualidade solicitado por cada um deles. Cabe ao provedor de serviços, por exemplo, a tarefa de implementar os parâmetros e categorias de serviços, configurar reservas de largura de banda nos elementos de comutação e roteamento, entre outras funções. Em contraponto, Redes Definidas por *Software* (SDN – acrônimo de *Software-Defined Networks*) constituem um paradigma emergente que facilita a criação e a introdução de novas abstrações na rede, simplificando o gerenciamento e facilitando a sua evolução [Kreutz et al. 2015].

Uma das características fundamentais do conceito SDN é o desacoplamento dos planos de dados e de controle dos ativos (comutadores e roteadores). Com isso, a inteligência da rede é transferida para um elemento controlador logicamente centralizado que, através de protocolos de comunicação seguros como o OpenFlow [McKeown et al. 2008], gerencia os dispositivos de encaminhamento de dados. Dessa forma, o modelo SDN proporciona uma visão global e um maior nível de programabilidade da rede de dados. Através dessas capacidades e, com o suporte do protocolo OpenFlow, o controlador da rede pode opcionalmente fornecer mecanismos simples e efetivos de provisionamento de QoS, como técnicas de moldagem de tráfego, através dos quais é capaz de garantir requisitos como retardo máximo, variação estatística do retardo (*jitter*) e vazão (média e máxima) às aplicações sensíveis a esses parâmetros.

Neste artigo, nós apresentamos uma arquitetura de provisão de QoS para aplicações distribuídas de alto desempenho explorando as capacidades do paradigma SDN. O modelo proposto tem o objetivo de oferecer uma plataforma de comunicação eficiente, mediante reserva de recursos e priorização de tráfego de dados. A abordagem consiste em disponibilizar componentes de *software* que são invocados pelas aplicações distribuídas através de chamadas de funções, de onde então estabelecem a negociação dos parâmetros dos requisitos particulares de QoS entre os componentes da arquitetura (entidades comunicantes e controladores SDN). As classes de serviços e suas respectivas especificações de tráfego são predefinidas em estruturas de dados geradas pelo administrador da rede e acessadas por alguns elementos que compõem o ambiente. O processo de troca de informações de solicitação e confirmação de reserva é inspirado

no protocolo RSVP (*Resource ReSerVation Protocol* [Braden et al. 1997]), utilizado na arquitetura IntServ (*Integrated Services* [Braden et al. 1994]). Esse procedimento é monitorado pelo controlador SDN que, ao final, modifica a rede alocando filas de QoS aos fluxos individuais de pacotes nas interfaces dos dispositivos de encaminhamento, ao longo de todo o caminho entre o prestador e o solicitador do serviço.

Como exemplo de viabilidade da proposta, nós desenvolvemos os protótipos dos componentes da arquitetura de provisão de QoS para um cenário específico de solução de transmissão de arquivos de dados. Para avaliar os resultados do trabalho, nós montamos um ambiente SDN no emulador Mininet[1] em conjunto com um controlador SDN POX[2]. No cenário, entidades em redes diferentes conectam-se por rotas através de três roteadores. A partir de medições com simulações de fluxos concorrentes, nós constatamos que a inclusão das rotinas de *software* adicionais ocasionou uma baixa sobrecarga média face ao tempo total de transferência. Os resultados mostraram também que tal arquitetura pode proporcionar um desempenho superior das aplicações em relação às implementações das mesmas em redes sem estabelecimento de contratos de serviço, com tempos de transferência até 47% inferiores, mesmo considerando condições de sobrecarga no cenário avaliado.

O projeto de arquitetura aqui apresentado destaca-se por sua implementação simples, através da manipulação de campos de cabeçalhos de pacotes e do emprego de ações nativas da especificação do protocolo OpenFlow sobre o Open vSwitch[3], de forma que o controlador toma ciência dos requisitos das aplicações sem a necessidade de metodologias custosas de classificação de tráfego, como aquelas baseadas em inspeção profunda de pacotes (DPI – *Deep Packet Inspection*) ou aprendizado de máquina (*Machine Learning*) [Qazi et al. 2013]. A interface funcional de alto nível oferecida às aplicações distribuídas incorpora um método simplificado de provisionamento de QoS com a granularidade de fluxos individuais de dados, de maneira oposta a certas arquiteturas tradicionais, tais como DiffServ (*Differentiated Services* [Blake et al. 1998]), que tratam agregados de fluxos. Conforme será discutido na Seção 2, a proposta também evidencia-se por seu propósito genérico, ao contrário de trabalhos na mesma área que focam em soluções para aplicações específicas.

O restante deste artigo está estruturado da seguinte forma. A Seção 2 discute os trabalhos relacionados. Na Seção 3, nós apresentamos os conceitos por trás da arquitetura proposta. A Seção 4, por sua vez, explora a arquitetura em si, seus componentes e interações. A descrição da avaliação, com o detalhamento do cenário considerado, da metodologia e dos resultados, é apresentada na Seção 5. Por fim, a Seção 6 mostra as conclusões deste artigo e os trabalhos futuros.

2. Trabalhos Relacionados

Ao longo dos anos, o crescente consumo de aplicações sensíveis a requisitos de QoS motivou muitos trabalhos de pesquisa nesse campo. Os esforços resultaram em várias propostas de mecanismos e arquiteturas como IntServ, DiffServ, MPLS (*MultiProtocol Label Switching* [Rosen et al. 2001]), entre outros. Infelizmente, as dificuldades de

[1] http://mininet.org

[2] https://github.com/noxrepo/pox

[3] http://openvswitch.org

administração e a configuração pouco flexível das redes tradicionais sempre dificultaram a implantação generalizada desses modelos. Naturalmente, com o amadurecimento do paradigma SDN, o estudo em torno do provisionamento de QoS nessas novas redes ganhou força [Karakus and Durresi 2017], de forma que muitas dessas novas pesquisas se relacionam de alguma forma com a proposta e o objetivo deste trabalho.

Nesse âmbito, há na literatura soluções altamente dependentes do contexto ou da aplicação. Por exemplo, [Yu et al. 2015] propõem uma solução para aplicações de *streaming* de vídeo que utiliza roteamento adaptativo para melhorar a qualidade desses fluxos sobre SDN. Nessa abordagem, chamada ARVS, pacotes da camada base e da camada de aprimoramento dos fluxos de *bits* do vídeo são tratados separadamente em dois níveis de fluxos de QoS. Dependendo das restrições de *jitter* do caminho mais curto, os diferentes fluxos podem ser isolados entre si e rerroteados para novos caminhos, de modo que os pacotes da camada base prioritariamente ocupem o caminho mais curto ou um novo caminho que ofereça largura de banda disponível. Sendo assim, alternativamente ao que propõe o nosso trabalho, os níveis de QoS são garantidos através da seleção de novas rotas viáveis. Entretanto, a escolha de caminhos alternativos nem sempre é possível, seja pela topologia ou pela própria política de roteamento.

Existem algumas outras soluções em SDN da literatura que são amarradas a contextos/aplicações específicos. Por exemplo, [Diorio and Timóteo 2016] apresentam e discutem o emprego de recursos para o encaminhamento de fluxos multimídia com suporte à QoS. Na rede, esses recursos são fornecidos por um *gateway* multimídia que atua como componente complementar ao controlador OpenFlow e como *gateway* de acesso à rede dos sistemas finais. Esse novo elemento é capaz de identificar e classificar múltiplos fluxos multimídia, possibilitando, por exemplo, que tais fluxos recebam um tratamento diferenciado quanto ao seu processamento e direcionamento. Apesar do método de identificação e classificação dos pacotes empregado nesse trabalho se assemelhar ao que é proposto no nosso estudo (análise do campo ToS – *Type of Service* – dos pacotes IPv4), a provisão de QoS se baseia em técnicas de engenharia de tráfego, o que também pode restringir a sua adoção em determinados ambientes.

[Humernbrum et al. 2014] também apresentam uma proposta destinada a uma aplicação específica. Os autores criam uma API *Northbound* SDN (entre o controlador e as aplicações de rede) para que aplicações interativas *online* em tempo real (*Real-Time Online Interactive Applications* – ROIA) especifiquem seus requisitos dinâmicos e os tenham atendidos em tempo real. Aplicações como jogos *online* com múltiplos jogadores possuem altas demandas de QoS devido às interações intensivas e dinâmicas, onde as respostas das ações dos usuários devem ocorrer, virtualmente, de forma imediata. A solução nesse caso é semelhante aos dois trabalhos anteriores [Yu et al. 2015, Diorio and Timóteo 2016], onde o controlador SDN reconfigura a rede na tentativa de acomodar os requisitos de qualidade de tráfego dessas aplicações, transmitindo os fluxos de dados sensíveis por uma conexão mais rápida.

No âmbito de redes de *data center*, [Afaq et al. 2015] utilizam técnicas de limitação de taxa para garantir QoS aos chamados fluxos "camundongo". Esses fluxos curtos sensíveis à latência, como VoIP, concorrem com os chamados fluxos "elefante", mais longos e gerados por rotinas de *backup*, por exemplo. Os autores usam um *framework* baseado em amostragem para detectar esses fluxos demorados e submetê-los a

um módulo QoS administrado por um controlador SDN que os roteiam a caminhos onde são aplicadas taxas de vazão restritas. A abordagem de provisão de QoS desse artigo adota técnicas de moldagem de tráfego somente sobre os fluxos "elefante". Nossa proposta, por sua vez, possibilita que sejam criados e moldados diversos níveis para diferentes fluxos, não se restringindo ao problema relatado pelos autores e sendo aplicável em outros inúmeros cenários de uso. Além disso, nossa arquitetura prevê que o mapeamento entre um determinado tráfego e sua respectiva categoria de qualidade de serviço, quando houver, seja realizado pelos *hosts* finais.

Por fim, poucas propostas tratam QoS sem a dependência de uma aplicação alvo. Por exemplo, [Tomovic et al. 2014] apresentam o projeto original de um ambiente de controle SDN/OpenFlow que fornece garantias de largura de banda para fluxos prioritários através de limitações de taxas. O controlador executa o cálculo das rotas e a reserva de recursos, porém a criação das filas nas interfaces dos dispositivos de encaminhamento é estática, sendo realizada no início da execução do controlador SDN. Dessa forma, como será evidenciado na Seção 4, essa proposta pode ser considerada menos flexível que a idealizada na nossa arquitetura.

3. *Background*

A qualidade de serviço (QoS) é uma característica que varia entre serviços, sendo, em geral, uma função do tipo de aplicação e da mídia transmitida [Colcher et al. 2005]. Segundo [Gomes and Soares 1999], pode-se enumerar dois princípios básicos relacionados ao fornecimento de QoS: (i) especificação dos requisitos dos usuários; e (ii) provisionamento dos mecanismos que garantam os requisitos dos usuários. Os requisitos geralmente são especificados em termos de retardo máximo, variação estatística do retardo (*jitter*), vazão (*throughput*), taxa de erros e taxa de perdas. Prover mecanismos que garantam tais requisitos significa permitir que os parâmetros especificados sejam garantidos às aplicações ao longo de sua execução, de maneira fim-a-fim.

Com base nesses princípios, [Colcher et al. 2005] também definem algumas fases genéricas de provisão de QoS: (i) solicitação de serviços; (ii) estabelecimento de contratos de serviço; (iii) manutenção de contratos de serviço; e (iv) encerramento de contratos de serviço. Na fase de solicitação de serviços podem ser empregados mecanismos estáticos, que impõem uma carga administrativa custosa, ou dinâmicos, como os que usam protocolos de sinalização. Feita a solicitação, um contrato implícito entre a aplicação e a rede é estabelecido. Nele, a rede assegura o transporte dos fluxos da aplicação, desde que a aplicação se comprometa a gerá-los de acordo com o especificado. Esse contrato é mantido por mecanismos apropriados até o seu encerramento, que pode ser feito também de forma estática ou dinâmica, quando os recursos reservados são então liberados.

Considerando esses princípios e fases, [Gomes and Soares 1999] descrevem as funcionalidades necessárias a um *framework* genérico de fornecimento de QoS: (i) a parametrização de serviços especifica parâmetros e categorias de serviço; (ii) o compartilhamento de recursos aborda mecanismos de alocação, classificação e escalonamento; (iii) a orquestração de recursos considera funções de negociação, controle de admissão e sintonização; e (iv) a adaptação de serviços altera o comportamento dos mecanismos e funções acima.

A configuração de funções e mecanismos de provisão de QoS em cada

equipamento de comutação ou roteamento em uma rede de dados convencional impõe um alto custo administrativo. Redes Definidas por *Software* (SDN – *Software-Defined Networks*), por sua vez, simplificam a gerência dessas redes. O termo SDN surgiu primeiramente no âmbito da Universidade de Stanford, nos EUA. Ele refere-se a uma arquitetura de rede fundamentada em quatro pilares: (i) desacoplamento dos planos de dados e de controle; (ii) decisões de encaminhamento baseadas em fluxos; (iii) transferência da lógica de controle para um elemento controlador externo logicamente centralizado; e (iv) programação da rede através de aplicações de *software* executando no topo do modelo. A Figura 1 retrata uma arquitetura típica SDN [Kreutz et al. 2015].

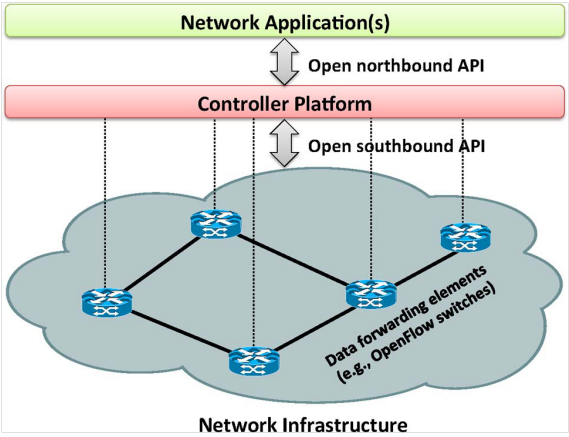


Figura 1. Visão simplificada de uma arquitetura SDN [Kreutz et al. 2015].

A interface entre o controlador SDN e os dispositivos da infraestrutura de rede (*southbound* API) são fornecidos por protocolos seguros como o OpenFlow [McKeown et al. 2008]. Ao contrário da API *northbound*, até hoje não padronizada, o OpenFlow é considerado atualmente um padrão *de facto*, sendo utilizado na imensa maioria das plataformas SDN atuais. Assim, em um ambiente SDN típico, os elementos de encaminhamento de dados (comutadores e roteadores) são habilitados com o OpenFlow, através dos quais podem ser gerenciados por um controlador compatível. As implementações do protocolo podem ser feitas em *hardware* ou em *software*. Para o último caso, uma implementação bastante comum é o Open vSwitch.

A API do OpenFlow, desde sua versão 1.0, oferece funções nativas que permitem o desenvolvimento de mecanismos de provisão de QoS. Juntamente com o Open vSwitch ou outros dispositivos compatíveis, é possível implementar facilmente funcionalidades de alocação, classificação e escalonamento de pacotes. A granularidade no trato de fluxos individuais e a visão global do SDN também viabilizam o fornecimento de mecanismos de negociação, controle de admissão e sintonização de QoS. De modo prático, essas capacidades são exploradas em soluções de QoS que utilizam rerroteamento de pacotes e balanceamento de tráfego de fluxos sensíveis. Entretanto, nem sempre esses métodos são viáveis. Em muitos casos, a política de roteamento aplicada à rede impede o estabelecimento de caminhos alternativos. Em outros, pode não haver rotas disponíveis, ou essas rotas podem estar simplesmente congestionadas. Nesse sentido, técnicas de moldagem de tráfego a partir de reservas de largura de banda fim-a-fim e limitação de taxas são soluções simples e efetivas de provisão de QoS, capazes de garantir os requisitos de qualidade das aplicações.

4. Arquitetura Proposta

A arquitetura proposta neste trabalho leva em conta os princípios, fases e funcionalidades relacionados na Seção 3, norteando diretamente a concepção dos componentes e o desenvolvimento dos protótipos utilizados na avaliação. Aqui, porém, é importante ressaltar que a elaboração da arquitetura objeto deste estudo se restringe ao fornecimento de níveis de qualidade no subsistema de rede de comunicação, sem preocupar-se com a provisão de QoS nos *hosts* finais (dispositivos e *buffers* de entrada/saída, processos de SO, pilha de protocolos, entre outros), conforme propõem [Moreno and Soares 2002]. A Figura 2 mostra uma representação, em alto nível, dessa arquitetura.

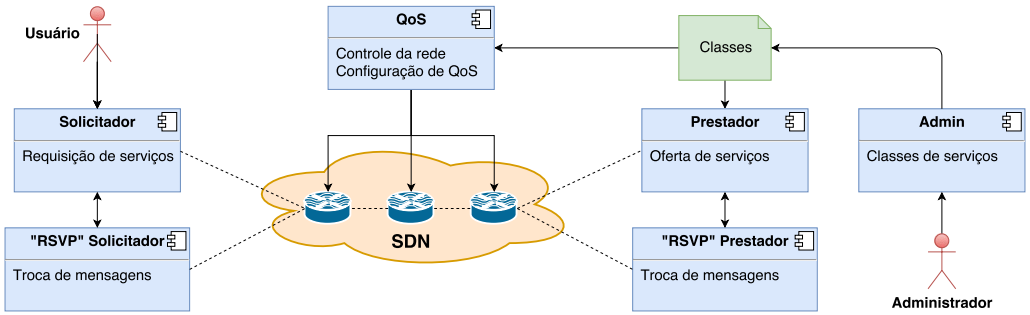


Figura 2. Diagrama da arquitetura de provisão de QoS.

A arquitetura é composta pelos módulos das entidades comunicantes (**Solicitador** e **Prestador**), os quais representam as aplicações distribuídas propriamente ditas. A fim de prover a QoS necessária a essas aplicações, são adicionados ao ambiente módulos de trocas de mensagens (**"RSVP" Solicitador** e **"RSVP" Prestador**), um módulo controlador SDN (**QoS**), um módulo de administração (**Admin**) e uma estrutura de dados (**Classes**). Na rede, a aplicação que exerce o papel de Solicitador requisita serviços à aplicação que exerce o papel de Prestador. Na arquitetura proposta, o Prestador, além de ofertar os respectivos serviços, assume a função de verificar a possibilidade de ofertá-los com garantias de qualidade. Para tanto, o Prestador examina a estrutura de dados que contém as categorias de serviço (por exemplo VoIP, vídeo, melhor esforço) e seus parâmetros (vazão média e máxima, por exemplo), mapeando elementos da mensagem de requisição (por exemplo nomes de arquivos, portas de serviços) às identificações das classes de serviço especificadas naquela estrutura. O módulo de administração, disponibilizado ao ente administrativo da rede, possui a tarefa de gerar essa base de dados de classes de serviço. O administrador deve considerar as necessidades das aplicações, os dados trafegados e as condições de rede. Conceitualmente, esse módulo implementa a funcionalidade de parametrização de serviços. A estrutura de dados deve seguir um formato que permita o relacionamento entre os serviços solicitados e as classes disponibilizadas pelo administrador às aplicações. O módulo de administração também executa a configuração inicial das políticas de QoS na rede, tendo em vista que o OpenFlow não possui essa capacidade. Os módulos de trocas de mensagens, invocados pelas aplicações através de chamadas de funções, são os responsáveis pelos procedimentos de solicitação e confirmação de reserva de recursos. A partir do monitoramento das mensagens trocadas, o módulo controlador realiza o controle de admissão das reservas e os ajustes na rede, a partir dos quais são atendidas as funções de alocação, classificação e escalonamento.

Para o atendimento das tarefas definidas pela arquitetura, a mesma prevê algumas interfaces entre as aplicações distribuídas e alguns dos módulos adicionais. A partir da consulta à estrutura de dados, o Prestador interage com o Solicitador informando-o da previsão de fornecimento de QoS. Assim, no momento oportuno, ambos iniciam as chamadas às funções correspondentes dos módulos de solicitação/confirmação de reserva. Em especial, na chamada à função do módulo "RSVP Prestador", o Prestador indica a categoria de serviço relativa a aplicação. As modificações das aplicações distribuídas seguem as diretivas específicas de implementação das interfaces dos módulos de trocas de mensagens, da base de dados das classes de serviço e das próprias aplicações. Não obstante, as interações previstas na arquitetura presumem poucas modificações por parte dos desenvolvedores dessas aplicações.

A troca de mensagens de solicitação e confirmação de reserva é inspirada no protocolo RSVP e resumida na Figura 3. O procedimento é acompanhado pelo controlador SDN que, a partir da verificação dos pacotes que contêm as respectivas mensagens, implementa os mecanismos de provisionamento de QoS. Inicialmente, (1) o processo do lado da aplicação prestadora envia uma mensagem (aqui identificada como "PATH") ao lado da aplicação solicitadora, juntamente com uma identificação numérica correspondente à classe do serviço que será fornecido. Ao perceber cada pacote contendo essa mensagem, as rotinas do módulo do controlador instalam as regras de encaminhamento nos comutadores/roteadores, estabelecendo o caminho fim-a-fim. (2) O processo do lado da aplicação solicitadora, ao receber a mensagem "PATH", reconhece o identificador da classe e gera uma mensagem de solicitação de reserva (aqui identificada como "RESV"), inserindo no campo ToS do cabeçalho do pacote um numeral relativo ao identificador. Ao analisar esse pacote em cada elemento de encaminhamento de dados na rota entre as entidades comunicantes, as rotinas de QoS do módulo controlador realizam o controle de admissão da reserva, calculando em cada porta do enlace a disponibilidade de largura de banda necessária àquela classe de serviço requerida, obtida pelo exame do campo ToS. O recebimento da mensagem "RESV" no lado da aplicação prestadora indica que a reserva é viável em toda a rota. Assim, (3) o processo do lado da aplicação prestadora gera um novo pacote com uma mensagem de confirmação de reserva (aqui identificada como "RESVCONF"), com o campo ToS identicamente marcado. Desse modo, à medida que o módulo do controlador analisa tal pacote, ele aplica as políticas de QoS nas interfaces dos dispositivos através dos quais os fluxos sensíveis irão trafegar, e configura as respectivas filas nas regras desses fluxos (mediante a diretiva ENQUEUE do OpenFlow). Com isso, as funcionalidades de alocação, classificação e escalonamento de pacotes são garantidas. Finalmente, (4) a recepção da mensagem "RESVCONF" no lado da aplicação solicitadora confirma a reserva de recursos no caminho, o que habilita o envio de uma mensagem de encerramento (aqui identificada como "FIN"). A partir do recebimento da mensagem "FIN" no lado da aplicação prestadora, o Prestador inicia o envio do fluxo de dados solicitado, o qual terá sua largura de banda garantida.

Na lógica descrita acima, não foram consideradas situações alternativas, como indisponibilidade de largura de banda no momento da reserva, que devem ser tratadas com mensagens de erro e de cancelamento de reserva/caminho, como aquelas presentes no RSVP. Também não foram examinadas funcionalidades de provisão de QoS mais complexas, como mecanismos de sintonização e adaptação de serviços.

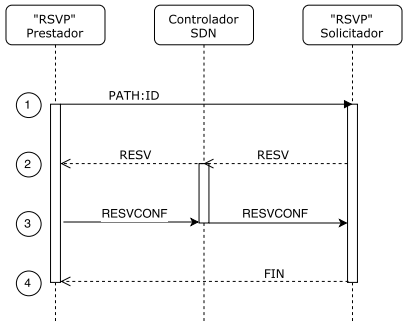


Figura 3. Diagrama de sequência de mensagens de reserva.

5. Avaliação

Para fins de avaliação da viabilidade da proposta, foram desenvolvidos os protótipos dos componentes da arquitetura[4]. Esta seção descreve os detalhes dessa avaliação.

5.1. Cenário Considerado

O cenário considerado na Figura 4 baseia-se em um ambiente SDN onde são oferecidos serviços de transferência de dados, composto por dois *hosts* (h1 e h2), dois servidores (srv1 e srv2), três roteadores (r1 a r3) e um controlador de rede (c0). Os roteadores r1 e r2 atuam como *gateways* para duas diferentes redes (10.0.1.0/24 e 10.0.11.0/24). Na Figura 4 também é possível observar a topologia de rede empregada e a associação de cada *host*/servidor/roteador a seus respectivos endereços de rede e enlaces/portas.

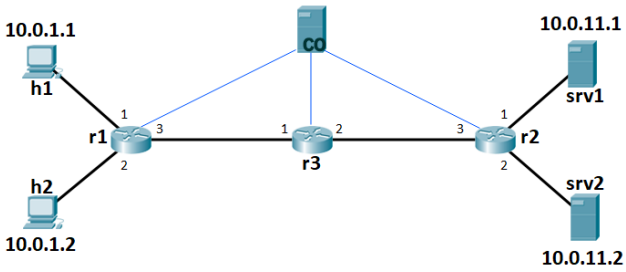


Figura 4. Cenário de avaliação considerado.

Esse cenário retrata um ambiente de rede de dados reduzido que, de acordo com análises prévias, captura todos os aspectos relevantes à nossa avaliação. Nele, as aplicações distribuídas demandam requisitos de qualidade a um provedor de serviços que provê a QoS necessária sob o controle de um ente administrativo. Dada essa premissa, a solução pode ser utilizada em inúmeros cenários, sob os mais diversos contextos, não importando quantos *hosts* compõem a rede ou onde eles estão localizados. As funções de prestador e solicitador de serviços podem ser assumidas por quaisquer dos *hosts*, do mesmo modo que em uma arquitetura sem QoS. A topologia com um único caminho entre estações representa uma rede onde métodos de provisão de QoS por rerroteamento ou balanceamento de tráfego não são aplicáveis. O número de roteadores é justificado pela distância média em saltos entre quaisquer *hosts* em redes menores, como aquelas encontradas em universidades ou pequenos *data centers*, o que torna o cenário realista.

[4]Os códigos-fontes dos protótipos dos componentes de *software* e os dados resultantes das simulações estão disponíveis em https://github.com/netlabufjf/qos-sdn

5.2. Metodologia de Avaliação

O ambiente de rede descrito na Subseção 5.1 foi implementado no emulador Mininet [Lantz et al. 2010] versão 2.3.0d1. Os *hosts* e servidores foram incluídos com as mesmas características, apesar de serem representados graficamente de maneiras diferentes. Da mesma forma, os roteadores foram adicionados à topologia como comutadores executando o Open vSwitch 2.0.2. Como controlador SDN foi empregado o POX *carp* utilizando o OpenFlow 1.0. Todo o ambiente foi montado em um Ultrabook 64 bits com CPU Intel Core i5-3317U 1,70 GHz, memória RAM de 4,0 GB e SO Ubuntu 14.04. Os recursos computacionais empregados foram suficientes para a avaliação e não influenciaram os resultados, tendo em vista que os consumos de memória e de CPU durante todos os testes permaneceram em torno de 30% e 25%, respectivamente.

O modelo de testes foi preparado visando reproduzir um ambiente capaz de realçar os impactos da provisão de QoS sobre fluxos concorrentes: (i) servidor srv1 é responsável pela transmissão de dados; (ii) *host* h1 (ponta de prova) exerce um papel de Solicitador, executando requisições de dados a srv1; (iii) tráfego constante unidirecional é gerado entre srv2 e h2; e (iv) largura de banda dos enlaces é limitada a 1 Gbps. Nesse contexto, o tráfego entre srv2 e h2 simula um fluxo de dados sem garantia de QoS. Esse fluxo é gerado através da ferramenta iPerf[5] com uma carga padrão (TCP à taxa máxima). Por sua vez, o fluxo de dados entre srv1 e h1 é tratado como sensível. Dessa forma, a partir de uma requisição de h1, o tráfego de resposta de srv1 é classificado de acordo com um nível de qualidade previamente estabelecido. Assim, durante a transferência entre srv1 e h1, ambos os fluxos se tornam concorrentes na rota r2-r3-r1, de modo que o tráfego proveniente de srv2 tem sua largura de banda reduzida a um nível mínimo, enquanto o fluxo sensível tem sua largura de banda garantida. Nas simulações sem QoS, como será visto adiante, ambos os fluxos compartilham a largura de banda máxima dos enlaces.

Para implementar o modelo de testes, foram desenvolvidas duas aplicações (Solicitador e Prestador) para a transferência de dados em rede sobre os protocolos TCP/IP. O uso do TCP justifica-se pelo fato de ser esse o protocolo de transporte predominante da Internet. Atualmente, até mesmo aplicações sensíveis ao retardo/*jitter* e tolerantes a perdas, como é o caso das soluções atuais de *streaming* multimídia, estão sendo implantadas sobre esse protocolo, principalmente em plataformas de *streaming* adaptativo fornecidas por provedoras como Netflix. Também foram implementados os componentes de *software* que permitem a especificação dos parâmetros e das categorias de serviço na rede. Embora a arquitetura forneça grande liberdade na definição das classes de QoS, no contexto do modelo de testes foram necessários somente dois níveis, que variaram em relação às vazões garantidas. Eles poderiam ser comparados, por exemplo, aos típicos níveis de *carga controlada* e *serviço garantido*. Para a troca de mensagens de parâmetros de qualidade entre as entidades comunicantes, foram também desenvolvidos os respectivos módulos. Por fim, o módulo de encaminhamento de pacotes *l3_learning*[6] do POX foi modificado com a inclusão das rotinas de provisão de QoS. Toda a programação foi feita utilizando a linguagem Python[7].

[5]https://iperf.fr

[6]O módulo *l3_learning* original permite que um comutador atue também como um roteador, encaminhando pacotes entre redes diferentes.

[7]https://www.python.org

Para o projeto das simulações foram definidas duas variáveis de resposta, ambas medidas a partir do ponto de vista do Solicitador: (i) sobrecarga adicionada ao tempo de transferência – causada pelo processamento das rotinas de provisão de QoS; e (ii) tempo total de transferência de dados. Para as simulações das transferências foram utilizados arquivos de vídeo cujo tamanho foi estabelecido como fator dos experimentos: (i) pequeno (44 MB); (ii) médio (128 MB); e (iii) grande (435 MB). Esses valores correspondem aos tamanhos relativos a três resoluções diferentes de um mesmo vídeo. A partir desse planejamento, foram executadas três sessões de testes, onde para cada sessão somente um tamanho de arquivo foi usado. Em cada sessão foram realizadas três rodadas de requisições, nas quais a rede se comporta sob três condições diferentes: (1) controlador SDN com módulo original (sem provisão de QoS); (2) controlador SDN com módulo modificado e reserva de 40% de largura de banda para o fluxo sensível; e (3) controlador SDN com módulo modificado e reserva de 60% de largura de banda para o fluxo sensível. Nos testes com provisão de QoS (casos 2 e 3) o tráfego sem garantia foi fixado em 10% da largura de banda, de modo que o fluxo de dados entre srv2 e h2 foi enquadrado nessa categoria. Com o intuito de determinar o impacto do erro experimental, em cada rodada foram efetuadas 30 requisições do mesmo arquivo utilizando as aplicações de transferência desenvolvidas, com intervalos de 120 segundos entre cada requisição.

5.3. Resultados

Os resultados da análise das 180 requisições com provisão de QoS mostram que as rotinas adicionais de *software* criadas para esse provisionamento apresentam baixa sobrecarga, com média de 1,056 s, com intervalos de confiança (95%) muito próximos: 1,052 s a 1,060 s. A Figura 5 mostra a função de distribuição de probabilidade acumulada dessa sobrecarga. Além do próprio processo de troca de mensagens na fase de solicitação de reserva, essa medida está associada ao número de saltos na rede, ou seja, possui relação com a quantidade de roteadores no caminho entre o Solicitador e o Prestador (aqui neste caso, três), tendo em vista os procedimentos de controle de admissão e reserva de recursos executados pelo controlador na chegada dos respectivos pacotes de mensagens em cada elemento de encaminhamento da rede. Para o cenário utilizado, esse valor médio obtido tem grande impacto, principalmente nas transferências mais curtas, sendo minimizado nas situações onde as transmissões são mais demoradas. Nesse aspecto, é possível avaliar que em ambientes onde há arquivos grandes ou cargas pesadas, como *streaming* de vídeo (cuja duração pode alcançar dezenas de minutos), a sobrecarga é compensada pela garantia de QoS dos usuários. Nesses cenários, há uma diminuição dos tempos de transmissão.

De fato, como mostrado na Figura 6, a análise das medições dos tempos totais de transferência de dados mostra que a provisão de QoS a partir da arquitetura proposta permite uma redução significativa do tempo de duração da transmissão, dependendo dos parâmetros e categorias de serviço estabelecidos. Na sessão de testes realizada com o arquivo pequeno (Figura 6(a)), a execução da aplicação de transferência com o módulo de controle sem QoS resultou em menores tempos totais, com média de 1,818 s. Conforme discutido anteriormente, a sobrecarga adicionada pelas rotinas de provisionamento influenciou no pior desempenho das demais simulações, com tempos médios de 2,715 s (QoS com reserva de 40%) e 2,398 s (QoS com reserva de 60%). Na sessão realizada com o arquivo de tamanho médio (Figura 6(b)), a configuração da reserva

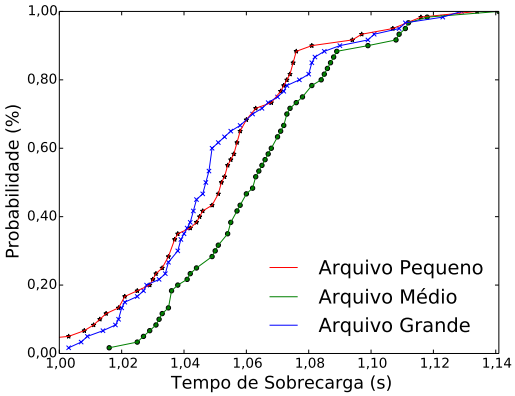
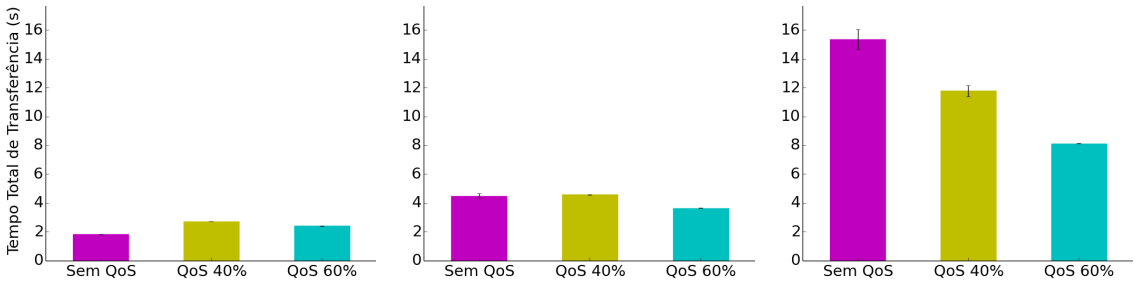


Figura 5. Distribuição de sobrecarga adicionada pela arquitetura.

de 60% para o fluxo de interesse foi capaz de alcançar menores tempos de transmissão, com média de 3,637 s, em uma redução de praticamente 19% se comparado aos testes sem QoS, que apresentaram uma média de 4,489 s. Nessa sessão, a rodada de requisições com reserva de 40% se mostrou equivalente à rodada sem QoS (intervalo de confiança de 95%), com média de 4,567 s. Por fim, a sessão de testes efetuada com o arquivo grande (Figura 6(c)) atingiu tempos visivelmente inferiores, com médias de 11,765 s (QoS a 40%) e 8,125 s (QoS a 60%). Em comparação às simulações sem QoS, com tempo de transferência médio de 15,346 s, as médias foram aproximadamente 23% e 47% menores. Nesse caso, apesar da sobrecarga adicional, a arquitetura de provisão de QoS mostrou-se consideravelmente melhor, o que corrobora ainda mais sua viabilidade.



(a) Arquivo Pequeno (b) Arquivo Médio (c) Arquivo Grande

Figura 6. Médias dos tempos totais de transferência de dados.

Como avaliação complementar, foi realizada uma análise comparativa por observações pareadas entre a abordagem sem QoS e com reservas de recursos. Para tanto, foram computados os intervalos de confiança para as médias das diferenças dos resultados. Os valores com 95% de confiança estão descritos na Tabela 1. As medidas negativas dos intervalos de confiança na comparação "Sem QoS/QoS 40%" e na comparação "Sem QoS/QoS 60%", utilizando o arquivo pequeno, indicam que a abordagem tradicional é superior àquela com provisão de QoS. Na sessão de simulações com o arquivo médio, o valor 0,0 incluso no intervalo da comparação "Sem QoS/QoS 40%" aponta que, para essa configuração, as alternativas são equivalentes. Ao aumentar a reserva para 60%, a aplicação da arquitetura com QoS já se mostra vantajosa,

conforme sinaliza o intervalo de valores positivos. Por fim, as comparações a partir das medições com o arquivo grande mostram intervalos positivos em ambos os casos, o que evidencia a superioridade da abordagem de provisionamento de QoS, sobretudo para arquivos maiores onde o ganho em desempenho compensa a sobrecarga introduzida. Essa metodologia de avaliação certifica a análise anterior. Naturalmente, os ganhos são dependentes das aplicações implementadas e das especificações de QoS aplicadas na rede.

Tabela 1. Análise comparativa por observações pareadas – intervalos de confiança de 95%.

	Sem QoS/QoS 40%	Sem QoS/QoS 60%
Arquivo Pequeno	(-0,923 , -0,872)	(-0,606 , -0,554)
Arquivo Médio	(-0,251 , 0,096)	(0,680 , 1,025)
Arquivo Grande	(2,894 , 4,268)	(6,541 , 7,902)

6. Conclusões e Trabalhos Futuros

Este estudo apresentou uma arquitetura de provisão de QoS para aplicações distribuídas de alto desempenho explorando as capacidades do paradigma SDN. Componentes de *software* habilitam um processo de solicitação e confirmação de reserva de recursos, o qual permite a um controlador de rede configurar filas de QoS a fluxos individuais no caminho entre entidades comunicantes dessas aplicações. Uma avaliação em cenário reduzido com protótipos desenvolvidos mostrou que a solução é viável, apresentando baixa sobrecarga sobre o tempo total de transferência de dados. Os resultados mostraram também que a aplicação adequada de parâmetros de classes de serviço nesse cenário reduziu o tempo de transmissão significativamente, melhorando o sistema existente.

Como trabalhos futuros, pretende-se desenvolver mecanismos adicionais de provisão de QoS, como funções de renegociação e adaptação. Espera-se também poder avaliar essa arquitetura no contexto de cenários mais complexos, analisando a escalabilidade da solução e o impacto dos tamanhos das rotas, além da utilização de dispositivos físicos (não simulados). Planeja-se, ainda, examinar os aspectos de qualidade de experiência dos usuários durante a utilização de aplicações como *streaming* de vídeo.

Agradecimentos

Os autores agradecem o apoio de CAPES, CNPq, FAPEMIG, FAPERJ e FAPESP.

Referências

Afaq, M., Rehman, S., and Song, W. (2015). Visualization of elephant flows and qos provisioning in sdn-based networks. In *Network Operations and Management Symposium (APNOMS), 2015 17th Asia-Pacific*, pages 444–447. IEEE.

Blake, S., Black, D., Carlson, M., Davies, E., Wang, Z., and Weiss, W. (1998). An architecture for differentiated services. RFC 2475, RFC Editor. `http://www.rfc-editor.org/rfc/rfc2475.txt`.

Braden, R., Clark, D., and Shenker, S. (1994). Integrated services in the internet architecture: an overview. RFC 1633, RFC Editor. `http://www.rfc-editor.org/rfc/rfc1633.txt`.

Braden, R., Zhang, L., Berson, S., Herzog, S., and Jamin, S. (1997). Resource reservation protocol (rsvp) – version 1 functional specification. RFC 2205, RFC Editor. http://www.rfc-editor.org/rfc/rfc2205.txt.

Colcher, S., Gomes, A., Silva, A., Filho, G., and Soares, L. (2005). *VoIP: Voz sobre IP*. Rio de Janeiro: Elsevier.

Diorio, R. and Timóteo, V. (2016). Per-flow routing with qos support to enhance multimedia delivery in openflow sdn. In *Proceedings of the 22nd Brazilian Symposium on Multimedia and the Web*, pages 167–174. ACM.

Gomes, A. and Soares, L. (1999). Um framework para provisão de qos em ambientes genéricos de processamento e comunicação. Master's thesis, Pontifícia Universidade Católica do Rio de Janeiro, Departamento de Informática.

Humernbrum, T., Glinka, F., and Gorlatch, S. (2014). A northbound api for qos management in real-time interactive applications on software-defined networks. *Journal of Communications*, 9(8):607–615.

Karakus, M. and Durresi, A. (2017). Quality of service (qos) in software defined networking (sdn): A survey. *Journal of Network and Computer Applications*, 80(Supplement C):200–218.

Kreutz, D., Ramos, F., Veríssimo, P., Rothenberg, C., Azodolmolky, S., and Uhlig, S. (2015). Software-defined networking: A comprehensive survey. *Proceedings of the IEEE*, 103(1):14–76.

Lantz, B., Heller, B., and McKeown, N. (2010). A network in a laptop: Rapid prototyping for software-defined networks. In *Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks*, page 19. ACM.

McKeown, N., Anderson, T., Balakrishnan, H., Parulkar, G., Peterson, L., Rexford, J., Shenker, S., and Turner, J. (2008). Openflow: Enabling innovation in campus networks. *ACM SIGCOMM Computer Communication Review*, 38(2):69–74.

Moreno, M. and Soares, L. (2002). Um framework para provisão de qos em sistemas operacionais. Master's thesis, Pontifícia Universidade Católica do Rio de Janeiro, Departamento de Informática.

Qazi, Z., Lee, J., Jin, T., Bellala, G., Arndt, M., and Noubir, G. (2013). Application-awareness in sdn. *ACM SIGCOMM Computer Communication Review*, 43(4):487–488.

Rosen, E., Viswanathan, A., and Callon, R. (2001). Multiprotocol label switching architecture. RFC 3031, RFC Editor. http://www.rfc-editor.org/rfc/rfc3031.txt.

Tomovic, S., Prasad, N., and Radusinovic, I. (2014). Sdn control framework for qos provisioning. In *Telecommunications Forum Telfor (TELFOR), 2014 22nd*, pages 111–114. IEEE.

Yu, T., Wang, K., and Hsu, Y. (2015). Adaptive routing for video streaming with qos support over sdn networks. In *Information Networking (ICOIN), 2015 International Conference on*, pages 318–323. IEEE.