Plataforma para Monitoramento de Métricas de Nível de Serviço em Redes Definidas por Software

Pedro H. A. Rezende, Paulo R. S. L. Coelho, Luís F. Faina, Lasaro Camargos, Rafael Pasquini

Faculdade de Computação (FACOM)
Universidade Federal de Uberlândia (UFU)

pedroh@mestrado.ufu.br

{paulocoelho, faina, lasaro, rafael.pasquini}@ufu.br

Resumo. *Este artigo apresenta e avalia uma plataforma para monitoramento de métricas de nível de serviço em Redes Definidas por Software (SDN) OpenFlow, denominada SDNMon. Por meio de monitoramentos frequentes e em diferentes granularidades, incluindo fluxos individuais, a SDNMon mantém medições precisas da vazão e atraso experimentados. As avaliações incluem duas formas de levantamento de dados, uma proposta baseada em consultas explícitas (polling) a contadores OpenFlow mantidos pelos elementos de rede e uma alternativa baseada em amostragens feitas com o protocolo sFlow.*

Abstract. *This paper presents and evaluates a platform for monitoring service-level metrics in OpenFlow Software Defined Networks (SDN), named SDNMon. The platform supports frequent network observation, at different granularity levels, including per-flow observation, being able to present accurate on-the-fly statistics regarding throughput and delay. The evaluation considered two statistical data gathering approaches, a proposed polling-based mechanism which collects information kept by OpenFlow counters at the network elements, and an alternate sampling-based mechanism which uses sFlow.*

1. Introdução

A utilização de um plano de controle desacoplado dos elementos de rede, conforme promovido no cenário de Redes Definidas por Software (SDN), estabelece um rico cenário para pesquisas em diversas áreas. A implementação deste plano de controle ocorre através de pelo menos um controlador, capaz de atuar em todos os elementos de rede presentes no plano de dados, programando seu comportamento de encaminhamento de tráfego. Esta atuação por parte do controlador é geralmente baseada em uma visão completa da rede, e utiliza métricas convencionais para alimentar algoritmos de caminho mais curto, tais como o número de saltos [Kim and Feamster 2013].

Embora muitas sejam as pesquisas relacionadas ao plano de controle SDN [Kreutz et al. 2014, Guedes et al. 2012, dos Passos Silva et al. 2013], apenas recentemente trabalhos investigam o monitoramento do plano de dados em SDN. Ademais, os cenários explorados são bem específicos, tais como, monitoramento de transmissões TCP [Suh et al. 2014], monitoramento dos elementos de entrada e saída da rede [van Adrichem et al. 2014] e monitoramento de latência utilizando sondas (*probes*) em cada enlace [Phemius and Bouet 2013].

A principal contribuição deste artigo é uma plataforma de monitoramento, denominada SDNMon, que provê um arcabouço para a implementação de diferentes técnicas de monitoramento, e portanto mais geral que os trabalhos anteriores. A SDNMon é independente da aplicação sendo transportada na rede e capaz de produzir uma visão detalhada do plano de dados, em diferentes granularidades. Além de unificar e melhorar o conjunto de ferramentas disponíveis aos administradores de rede, a SDNMon pode atuar em conjunto com outros módulos do controlador, por exemplo, enriquecendo a visão sobre o plano de dados, permitindo a definição de rotas que consideram a maximização da taxa de transmissão ou a minimização do atraso.

Como uma segunda contribuição deste trabalho, é proposto um mecanismo de monitoramento baseado em consultas explícitas (*polling*) aos contadores mantidos em cada um dos elementos de rede, definidos na especificação do OpenFlow [ONF 2013]. Basicamente, o mecanismo de monitoramento proposto baseado em *polling* é implementado no arcabouço promovido pela plataforma SDNMon. O mecanismo de *polling* é capaz de adaptar-se a situação de carga da rede, produzindo uma visão detalhada em diferentes níveis de granularidade, incluindo a condição dos elementos de rede presentes no plano de dados, das portas que os compõem, das aplicações utilizando a rede e dos fluxos individuais sendo encaminhados na SDN.

Para validar a SDNMon, a seção de experimentos contempla medições efetuadas utilizando duas diferentes abordagens de monitoramento: a primeira é o mecanismo proposto baseado em *polling*; a segunda utiliza o protocolo sFlow [sFlow 2014, Phaal et al. 2001], efetuando amostragens de pacotes sendo transmitidos na rede. Os experimentos apresentados neste artigo investigam o funcionamento de ambas as abordagens e, também, avaliam o monitoramento considerando três aplicações legadas, incluindo transmissões UDP feitas a partir do gerador de tráfego *Iperf* [Iperf 2014], transmissões TCP feitas utilizando a aplicação *scp* para a cópia de um arquivo, e *streaming* de vídeo utilizando o VLC [VLC 2014].

As demais seções estão organizadas da seguinte forma: A Seção 2 introduz os principais trabalhos de monitoramento presentes na literatura, oferecendo uma breve comparação com a plataforma proposta. A Seção 3 detalha a plataforma SDNMon e o mecanismo de monitoramento proposto baseado em *polling*. A Seção 4 apresenta os resultados das avaliações. A Seção 5 conclui o artigo e apresenta os trabalhos futuros.

2. Trabalhos Relacionados

Em [van Adrichem et al. 2014], os autores utilizam um mecanismo de *polling* para coletar dados referentes aos fluxos, permitindo apresentar medidas de largura de banda consumida por fluxo e da taxa de pacotes perdidos. Ainda, utilizando um mecanismo de *push*, os autores conseguem analisar o atraso fim-a-fim experimentado. O principal diferencial entre o trabalho apresentado em [van Adrichem et al. 2014] e a plataforma SDNMon está na quantidade de detalhes capturados pelo módulo de monitoramento baseado em *polling*. Em [van Adrichem et al. 2014], leituras são executadas apenas nos elementos de rede de ingresso e egresso, enquanto na SDNMon as leituras são feitas em todos os elementos de rede que compõem o plano de dados, permitindo o monitoramento dos fluxos em cada um dos enlaces por onde são encaminhados.

O protocolo sFlow [sFlow 2014, Phaal et al. 2001] trabalha na metodologia de

push, utilizando agentes presentes em cada um dos elementos de rede, responsáveis por amostrar pacotes e enviá-los para um nó central denominado coletor. As amostragens do sFlow são realizadas apenas na granularidade das portas dos elementos de rede, através da configuração de uma taxa de amostragem N. Por exemplo, utilizando uma taxa de amostragem $N = 100$, o agente sFlow irá amostrar um pacote a cada cem pacotes transmitidos em determinada porta. O coletor recebe uma cópia dos cabeçalhos dos pacotes amostrados, permitindo estimar o total de bytes transmitidos e a taxa de transmissão.

O sFlow é modificado em [Suh et al. 2014] para inspecionar o número de sequência do TCP presente nos pacotes, como forma de aumentar a precisão das amostragens. De fato, o mecanismo apresentado em [Suh et al. 2014] melhora o funcionamento do sFlow, porém restringe as amostragens apenas a transmissões TCP. O principal diferencial do sFlow, ou mesmo da modificação publicada em [Suh et al. 2014], frente ao mecanismo de *polling* proposto neste artigo, refere-se a atuação em diferentes níveis de granularidade (portas, filas, fluxos, elementos de rede), na metodologia de *pull* para coletar informações mantidas pelos contadores previstos no OpenFlow.

Em [Phemius and Bouet 2013], os autores utilizam mensagens OpenFlow como forma de medir a latência experimentada em cada enlace. Basicamente, propõem a injeção de mensagens via controlador em um *switch* da malha de elementos de rede e a posterior recuperação destas mensagens em um *switch* adjacente. Através do monitoramento do tempo necessário ao retorno das mensagens ao controlador, os autores apresentam os valores de latência. Na SDNMon não são utilizadas mensagens injetadas pelo controlador; as medições são feitas considerando os dados pertencentes aos fluxos monitorados, comparando o volume de dados encaminhado em cada um dos *switches*.

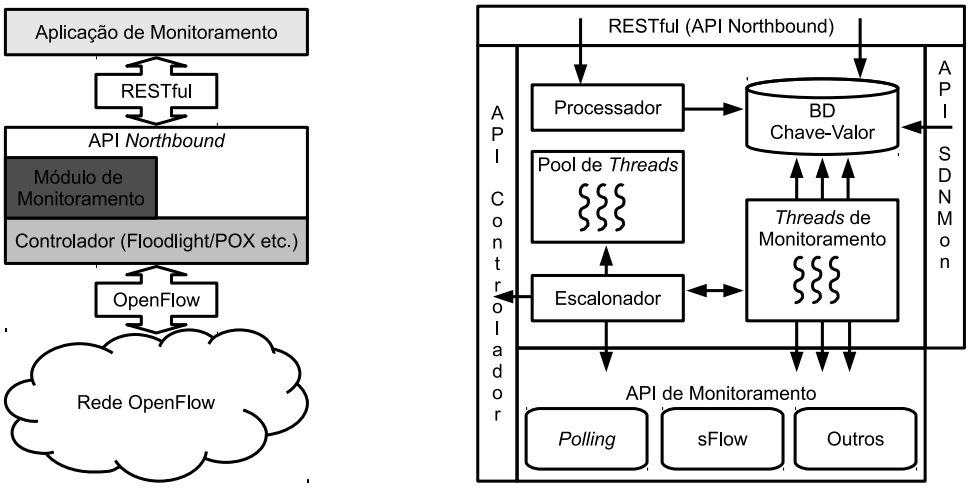
3. Plataforma de Monitoramento SDNMon

Conforme ilustra a Figura 1(a), o desenvolvimento da plataforma de monitoramento SDNMon é feito sob o cenário de SDN, composta por elementos de rede OpenFlow, sendo os elementos de rede responsáveis pela composição do plano de dados e ao menos um controlador responsável pela composição do plano de controle.

A SDNMon é desenvolvida como uma extensão ao controlador, utilizando informações presentes neste para auxiliar no monitoramento, tais como informações sobre a topologia do plano de dados. Todas as informações coletadas pelo módulo de monitoramento estão disponíveis utilizando o protocolo RESTful, através de uma extensão à API *northbound* do controlador. Nesta interface, a Aplicação de Monitoramento consegue interagir com o Módulo de Monitoramento, obtendo dados para apresentação em uma interface gráfica e, também, indicando qual o nível de granularidade desejado nas ações de monitoramento do mecanismo de *polling*.

A Figura 1(b) detalha a arquitetura do Módulo de Monitoramento proposto na SDNMon, cuja característica principal é a adaptabilidade ao cenário, em reação a variações no volume de tráfego na rede, baseada na utilização de um conceito de *pool* de *threads*. A Seção 3.1 detalha esta característica de adaptabilidade da SDNMon, implementada pelo mecanismo de *polling* proposto.

O Escalonador utiliza informações referentes à topologia do plano de dados obtidas no controlador (através da API do Controlador) para instanciar o módulo de monitoramento, provendo inicialmente uma *thread* para monitorar integralmente cada um dos



(a) Cenário considerado.

(b) Arquitetura do Módulo de Monitoramento.

Figura 1. Plataforma SDNMon.

elementos de rede. As ações de monitoramento ocorrem através da API de Monitoramento, que pode contemplar diferentes abordagens. A Figura 1(b) apresenta as opções *Polling* e sFlow, utilizadas nas experimentações deste artigo, mas também seria possível utilizar outras abordagens, como a proposta em [Suh et al. 2014] descrita na Seção 2.

Todos os dados coletados são armazenados na Base de Dados do tipo chave-valor pelas *threads*. A chave refere-se a um elemento de rede, uma porta, uma fila ou a um fluxo. No caso de um elemento de rede, das portas ou das filas, o campo valor armazena dados referentes ao volume agregado de tráfego sendo experimentado neste componente. No caso de um fluxo, o valor do objeto é o conjunto de arestas do caminho percorrido pelo fluxo; para cada aresta há uma outra entrada na base de dados, cuja chave associa o fluxo ao par de *switches* que compõe a aresta e o valor é o conjunto de informações relativas à mesma. Desta forma, as *threads* podem atualizar as arestas concorrentemente.

Especificamente no caso dos monitoramentos utilizando o mecanismo proposto de *polling*, os dados coletados estão em conformidade com os contadores previstos na especificação do OpenFlow [ONF 2013]. Alguns dos dados não precisam de nenhum tipo de processamento para serem disponibilizados para a Aplicação de Monitoramento, tais como informações sobre a capacidade dos enlaces e o número de portas dos elementos de rede. Porém, alguns dos valores precisam de um grau variado de processamento para serem apresentados. Desta forma, o módulo de monitoramento contempla um Processador, responsável pelos cálculos. As Seções 3.2 e 3.3 descrevem o uso do processador.

Os dados referentes à taxa de transmissão obtidos pelo sFlow são pré-processados pelo coletor do sFlow, utilizando as amostras enviadas a ele pelos agentes. Neste caso, a opção sFlow implementada pelo módulo de monitoramento recupera estes dados junto ao coletor e armazena na base de dados. O Processador também é utilizado para adaptar os dados ao formato adequado para exibição na Aplicação de Monitoramento.

Finalmente, é importante destacar que a plataforma de monitoramento SDNMon é o primeiro componente na direção de uma plataforma para o provisionamento de quali-

dade de serviço que encontra-se em desenvolvimento. Através da API SDNMon ilustrada na Figura 1(b), os módulos da plataforma de qualidade de serviço podem obter os dados armazenados na base de dados de monitoria, observando se acordos de nível de serviço estão sendo honrados. A plataforma de qualidade de serviço está fora do escopo deste trabalho e será detalhada em um trabalho futuro.

3.1. Adaptabilidade do *Pool* de *Threads* no Mecanismo de *Polling*

Conforme mencionado anteriormente, o Escalonador consulta informações topológicas através da API do Controlador e instancia uma *thread* de monitoramento para cada um dos elementos de rede compondo a SDN. Na sequência, a partir das informações coletadas via mecanismo de *polling* em cada um dos elementos de rede, as *threads* de monitoramento realimentam o escalonador, levando a adaptações no número de *threads* instanciadas.

Os dados são coletados através do envio de mensagens de *Statistic Request* a partir de cada uma das *threads* aos respectivos elementos de rede. O teor das solicitações enviadas nas mensagens é definido pelo escalonador de acordo com o escopo de monitoramento atribuído a cada uma das *threads*. Uma vez que inicialmente há uma *thread* para cada elemento de rede, as mensagens utilizam coringas (*wildcards*) em todos os campos previstos na estrutura da mensagem de *Statistic Request*. Ao receber uma requisição como esta, o elemento de rede devolve uma mensagem do tipo *Statistic Reply*, contendo todos os valores contabilizados.

Ao receber o retorno de um elemento de rede, a *thread*, além de armazenar as informações na base de dados, analisa o volume de informações retornado e comunica ao escalonador para que decida sobre a necessidade de novas *threads* para eliminar gargalos de monitoramento. Utilizando desta técnica de *feedback*, o escalonador é capaz de instanciar novas *threads* e definir o escopo de monitoramento atuando nos campos solicitados nas mensagens de *Statistic Request*, ou seja, o escalonador determina o escopo de monitoramento através da definição dos coringas em cada uma das *threads* instanciadas.

A partir da especificação atual dos coringas presente no OpenFlow, a SDNMon consegue instanciar *threads* responsáveis pelo monitoramento na granularidade de elementos de rede, portas individuais dos elementos de rede, filas ou fluxos individuais. Como investigação futura, um dos itens de interesse é o suporte por parte do OpenFlow para a definição de coringas que representam intervalos, permitindo a instanciação de *threads* responsáveis, por exemplo, pelo monitoramento de um conjunto de portas dos elementos de rede ou um conjunto de filas. Também investigaremos aspectos relacionados ao sincronismo entre as *threads*, e o impacto que o sincronismo causa no volume de tráfego de controle e a sobrecarga de processamento aplicada nos elementos compondo o plano de dados da SDN.

3.2. Calculando a Taxa de Transmissão

A Figura 2 apresenta o método proposto neste trabalho para o cálculo da taxa de transmissão a partir do total de dados transmitidos. O método pode ser aplicado independentemente da granularidade dos dados obtidos pelo mecanismo de *polling*. Considerando a coleta de dados referente a um fluxo em um mesmo elemento de rede, temos que a Série 1, apresentada na Figura 2, corresponde ao total de dados transmitidos no fluxo em questão para este elemento de rede. Através da coleta de dois pontos consecutivos é possível obter a taxa de transmissão referente ao fluxo neste elemento de rede.

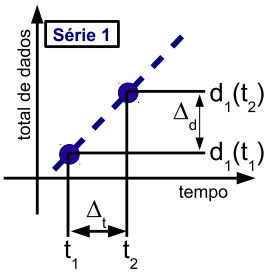


Figura 2. Cálculo da taxa de transmissão.

Conforme ilustrado na Figura 2, é possível obter o total de dados transmitidos entre os dois pontos coletados utilizando a fórmula $\Delta_d = d_1(t_2) - d_1(t_1)$, onde $d_x(t_y)$ representa o total de dados transmitidos na Série x em um determinado instante de tempo y. Ainda considerando estes dois pontos, é possível obter o intervalo de tempo entre eles, utilizando a fórmula $\Delta_t = t_2 - t_1$, onde t_y corresponde ao instante no qual as respectivas medições foram efetuadas. A partir destes valores, é possível obter a taxa de transmissão $T^1_{t1,t2} = \Delta_d/\Delta_t$ da Série 1 no intervalo t_1, t_2.

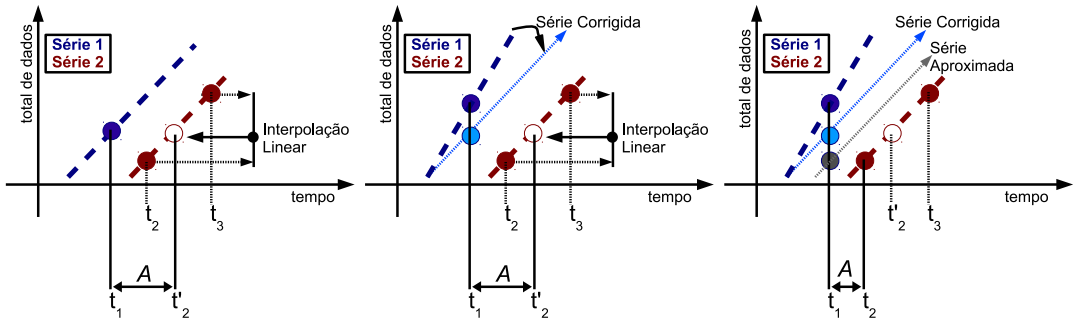
3.3. Calculando o Atraso

É comum encontrarmos trabalhos na literatura, por exemplo [Phemius and Bouet 2013] apresentado na Seção 2, que injetam sondas (pacotes de *probe*) na rede para determinar o atraso sendo experimentado. Este tipo de abordagem é bastante simples de ser implementada, mas pode não representar precisamente o atraso sendo experimentado por um determinado fluxo. Atualmente, é comum utilizarmos diversas filas de encaminhamento em uma única porta de um elemento de rede, fato este que exigiria a inserção de diversas sondas, específicas para cada uma das possíveis filas ao longo do caminho.

Em nossa abordagem, propomos a utilização de informações referentes ao total de dados transmitidos em cada um dos elementos de rede, de tal forma a efetuar o cálculo de atraso em diferentes granularidades, de acordo com os valores obtidos nos contadores especificados pelo OpenFlow. A Figura 3(a) apresenta o método proposto em um cenário livre de interferências, sem a ocorrência de descartes de dados, consistindo essencialmente em estimar o intervalo de tempo necessário para que cada elemento receba todos os dados enviados pelo anterior.

Considerando o cálculo de atraso para um determinado fluxo, a Figura 3(a) apresenta as Séries 1 e 2, coletadas em dois elementos de rede adjacentes, através dos quais o fluxo em questão é encaminhado. O método consiste em determinar t_1, t_2 e t_3, tal que $d_2(t_2) \leq d_1(t_1) \leq d_2(t_3)$. O próximo passo é determinar $T^2_{t2,t3}$, conforme definido na Seção 3.2. Finalmente, determinar t'_2, tal que $d_1(t_1) = d_2(t'_2)$, utilizando uma interpolação linear, para obter o atraso $A = t'_2 - t_1$.

O contador de total de bytes transmitidos do OpenFlow, mantido pelos elementos de rede, contabiliza o total bruto de bytes transmitidos, incluindo os bytes referentes aos pacotes efetivamente transmitidos e, também, os bytes referentes aos pacotes descartados. Desta forma, a Figura 3(b) ilustra como esta condição interfere no cálculo do atraso experimentado. Para determinar o atraso real, é preciso subtrair da Série 1 o total de dados descartados, obtendo o total de dados efetivamente transmitidos no instante de tempo em



(a) Séries sem a ocorrência de descarte de dados. (b) Utilizando um contador de total de dados descartados. (c) Sem a utilização de um contador de dados descartados.

Figura 3. Cálculo do atraso.

questão, representado pela Série Corrigida na Figura 3(b). Após a correção da série de dados, aplica-se o mesmo método descrito anteriormente para obter o valor de atraso.

Entretanto, a especificação do OpenFlow contempla apenas o contador referente aos bytes descartados por porta, não sendo possível obter informações referentes aos bytes descartados por fila ou fluxo. Sendo assim, a Figura 3(c) apresenta um cenário alternativo, onde o ponto da Série 1 obtido no instante t_1 é aproximado para o correspondente em total de dados do ponto da Série 2 obtido no instante t_2, considerando $d_{ap}(t_1) \approx d_2(t_2)$. Neste caso, descarta-se a interpolação linear, sendo possível obter o atraso $A = t_2 - t_1$. Entretanto, como pode ser observando na Figura 3(c), esta metodologia compromete a precisão do cálculo de atraso. A Seção 4 apresenta uma análise sobre o nível de precisão perdido neste caso.

Em um trabalho futuro, estenderemos uma implementação do OpenFlow, incluindo contadores referentes aos dados descartados em todos os níveis de granularidade (portas, filas e fluxos), em termos de total de pacotes descartados e total de bytes descartados. A inserção destes novos contadores permitiria análises mais precisas sobre o funcionamento da rede.

4. Experimentos

A Figura 4 apresenta o cenário utilizado para as análises da plataforma SDNMon. Foram consideradas a comunicação entre dois hospedeiros (H1 e H2) através de uma rede OpenFlow composta por cinco elementos de rede (SW1, SW2, SW3, SW4 e SW5). A infraestrutura de rede foi instanciada no Mininet versão 2.1.0, executando o Open vSwitch versão 2.0.2 nos elementos de rede e o controlador utilizado para a implementação do módulo de monitoramento é o Floodlight versão 0.90. Todos os experimentos foram efetuados em um Intel Core i3-2310M CPU 2.10GHz x 4 com 4 GB de memória RAM e sistema operacional Ubuntu 12.04.

Conforme mencionado anteriormente, o módulo de monitoramento da SDNMon foi implementado contando com dois mecanismos de monitoramento. O primeiro é o mecanismo proposto de *polling*, que efetua a coleta de valores armazenados pelos contadores previstos na especificação OpenFlow, através de consultas explícitas usando as mensagens de *Statistic Request* e *Statistic Reply*. O segundo mecanismo utiliza o protocolo sFlow, baseado em agentes executados nos elementos de rede que enviam as amostragens a um

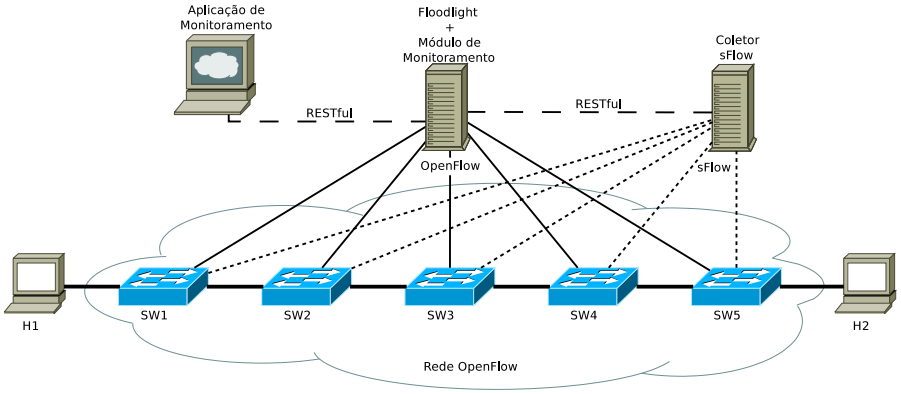
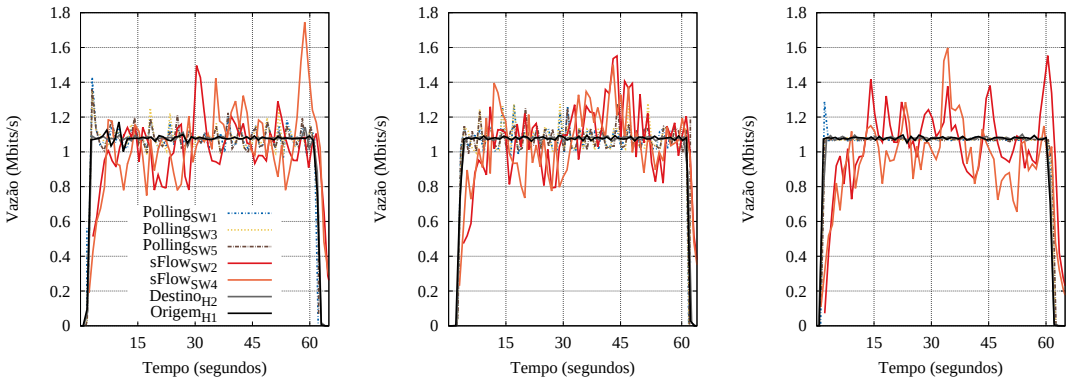


Figura 4. Cenário utilizado nas avaliações.

coletor central. As amostragens consideram um pacote dentro de um conjunto N de pacotes transmitidos, na granularidade das portas. O módulo de monitoramento comunica-se com o protocolo na versão sFlow-RT via RESTful.

4.1. Análise do Mecanismo de *Polling*

A Figura 5 apresenta uma análise referente à variações na taxa de coletas do módulo de monitoramento no caso do mecanismo de *polling*, objeto principal da análise efetuada nesta seção. O fluxo analisado foi gerado utilizando o *Iperf*, transmitindo UDP à taxa de 1 Mbps durante aproximadamente um minuto e utilizando pacotes com MTU de 1500 bytes. Para permitir uma visão completa sobre o mecanismo de monitoramento em todos os elementos de rede e permitir uma análise comparativa, são apresentados os dados de monitoramento do mecanismo de *polling* coletados nos elementos de rede SW1, SW3 e SW5 e os dados referentes ao sFlow coletados nos elementos de rede SW2 e SW4. Estes dados do mecanismo de monitoramento são comparados à transmissão real observada diretamente nas placas de rede dos hospedeiros H1 e H2, sendo H1 o hospedeiro de origem do tráfego analisado.



(a) 10 milissegundos. (b) 100 milissegundos. (c) 1 segundo.

Figura 5. Variação de intervalos de coleta do mecanismo de *polling*.

A metodologia foi variar a frequência entre leituras, considerando 10 milissegundos (Figura 5(a)), 100 milissegundos (Figura 5(b)), 1 segundo (Figura 5(c)) e armazenando apenas os valores observados quando havia mudança nos contadores. Em todas

as amostragens da Figura 5, o mecanismo sFlow efetuou a amostragem considerando $N = 10$, ou seja, eram considerados para amostragem um pacote a cada dez transmitidos.

A Tabela 1 apresenta os dados consolidados referentes à média e o desvio das análises apresentadas na Figura 5. Os valores apresentados na Tabela 1 consideram as observações feitas em todos os elementos de rede (SW1 a SW5) para ambas as metodologias de *polling* e sFlow. Como pode ser observado, o mecanismo de *polling* apresenta maior precisão em todos os cenários avaliados, apresentando sempre média próxima aos valores de H1 e H2 com um baixo desvio padrão.

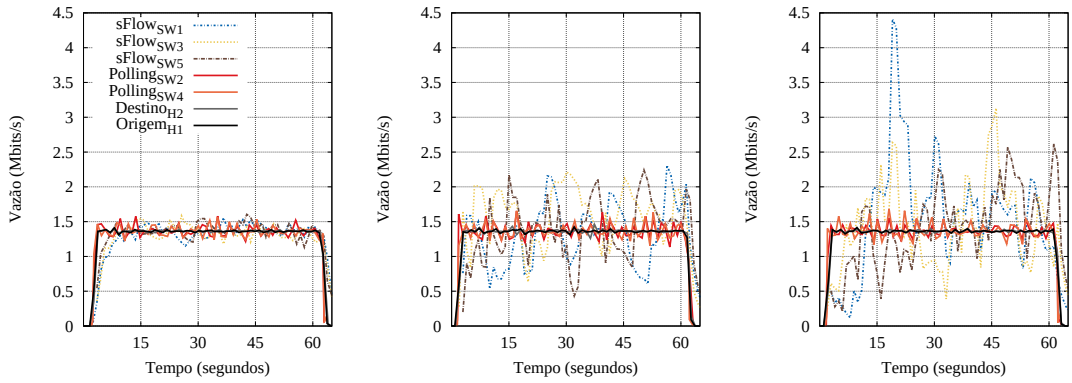
Tabela 1. Dados consolidados referentes à análise do mecanismo de *polling*.

Análise	Média e Desvio Padrão (em Mbits/s)			
	$Origem_{H1}$	$Destino_{H2}$	$Polling_{SW1..5}$	$sFlow_{SW1..5}$
Fig. 5(a) (10ms)	1,073 +/- 0,044	1,075 +/- 0,029	1,086 +/- 0,069	1,055 +/- 0,191
Fig. 5(b) (100ms)	1,072 +/- 0,048	1,070 +/- 0,041	1,065 +/- 0,155	1,081 +/- 0,176
Fig. 5(c) (1s)	1,072 +/- 0,048	1,068 +/- 0,057	1,068 +/- 0,029	1,060 +/- 0,170

4.2. Análise do Mecanismo de Amostragem sFlow

A Figura 6 analisa variações na taxa N de amostragem do sFlow. Por se tratar de um mecanismo de amostragem, o principal problema desta abordagem é ser utilizada em situações onde os pacotes amostrados possuem tamanho variado, uma característica comum nas redes de comutação por pacotes. Desta forma, a metodologia aplicada nestes testes foi a geração de 15 fluxos UDP entre H1 e H2 a uma taxa de 100 kbps. Cada fluxo possui MTU distinta, variando entre 100 e 1500 bytes, em incrementos de 100 bytes.

Neste caso, como o sFlow é o objeto principal de avaliação, são apresentados os dados de monitoramento da amostragem sFlow referentes aos elementos de rede SW1, SW3 e SW5 e, para comparação, são apresentados os dados referentes ao mecanismo de *polling* coletados nos elementos de rede SW2 e SW4. Em todas as análises da Figura 6, o mecanismo de *polling* efetuou medições em intervalos de um segundo.



(a) $N = 10$. (b) $N = 100$. (c) $N = 256$.

Figura 6. Variação da taxa de amostragem N do sFlow.

Como pode ser observado nos gráficos da Figura 6, mudanças na taxa de amostragem N comprometem a precisão do protocolo sFlow no caso de transmissões com pacotes

de tamanho variado. Além disso, a taxa de amostragem, conforme mencionado, é associada à porta dos elementos de rede, não possuindo caráter adaptativo, ou seja, não reage à mudanças no volume de tráfego. Por exemplo, uma taxa de amostragem $N = 256$ é o valor recomendado para portas de 10 Mbps, $N = 512$ para 100 Mbps, $N = 1024$ para 1 Gbps, $N = 2048$ para 10 Gbps e $N = 8192$ para 100 Gbps [sFlow 2014]. Embora o valor de amostragem da Figura 6(c) seja o recomendado para portas de 10 Mbps, a incidência de um volume inferior de tráfego também ocasiona perda de precisão.

Considerando as características dos fluxos UDP transmitidos, temos uma taxa de geração superior a quatrocentos pacotes por segundo, levando o agente sFlow a amostrar mais de quarenta pacotes por segundo no cenário com $N = 10$. Neste mesmo cenário, a coleta de dados dos contadores no mecanismo de *polling* está ocorrendo à frequência de uma coleta por segundo. Nestes experimentos, o mecanismo de *polling* está atuando na granularidade da porta de rede, da mesma forma como o sFlow atua. Pode-se ajustar o mecanismo de *polling* para atuar na granularidade de fluxos, exibindo dados individuais referentes a cada um dos quinze fluxos do experimento, uma característica não possível de ser implementada pelo sFlow.

A Tabela 2 apresenta a consolidação dos dados referentes ao sFlow. Especificamente sobre os resultados apresentados na Figura 6(a), é possível notar que a precisão do sFlow está bem próxima aos valores apresentados pelo mecanismo de *polling*, mas ainda assim indicando vantagem do mecanismo de *polling*.

Tabela 2. Dados consolidados referentes à análise do sFlow.

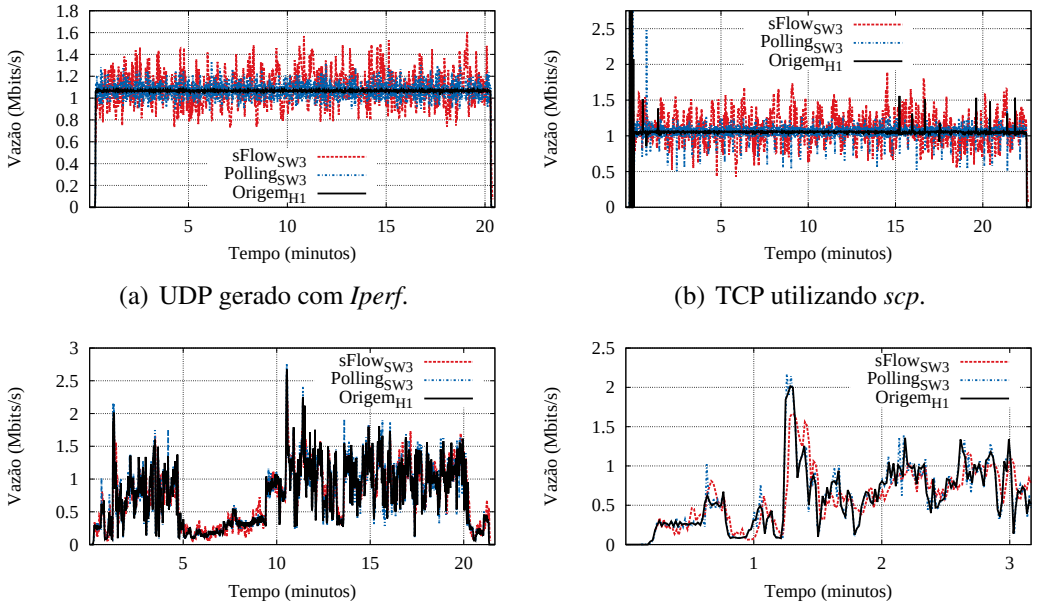
Análise	Média e Desvio Padrão (em Mbits/s)			
	$Origem_{H1}$	$Destino_{H2}$	$Polling_{SW1..5}$	$sFlow_{SW1..5}$
Fig. 6(a) ($N = 10$)	1,356 +/- 0,045	1,352 +/- 0,072	1,359 +/- 0,109	1,327 +/- 0,185
Fig. 6(b) ($N = 100$)	1,360 +/- 0,023	1,362 +/- 0,027	1,360 +/- 0,123	1,431 +/- 0,436
Fig. 6(c) ($N = 256$)	1,354 +/- 0,064	1,357 +/- 0,042	1,357 +/- 0,122	1,493 +/- 0,636

4.3. Análise da Taxa de Transmissão em Aplicações Legadas

A Figura 7 apresenta uma análise com três aplicações legadas, incluindo transmissão UDP feita a partir do gerador de tráfego *Iperf* [Iperf 2014] à taxa de 1 Mbps, transmissão TCP feita utilizando a aplicação *scp* limitada a vazão de 1Mbps durante a cópia de um arquivo, e transmissão de vídeo utilizando o VLC [VLC 2014] sem qualquer controle referente à taxa de transmissão. Em todas as três aplicações legadas avaliadas, o período de análise considera um intervalo aproximado de vinte minutos de transmissão. Nestes experimentos, os mecanismos de monitoramento foram configurados da seguinte forma: a) mecanismo de *polling* efetuando coletas em intervalos de cem milissegundos e b) sFlow com taxa de amostragem $N = 10$. Ambas as taxas de leitura elevadas, de tal forma a melhorar a precisão das leituras efetuadas.

Por se tratar de experimentos distintos, as escalas dos gráficos foram definidas individualmente, de tal forma a permitir uma melhor avaliação das séries de dados. Devido ao longo tempo de duração, as séries foram plotadas utilizando uma ordem que permitia uma melhor visualização das sobreposições, includindo apenas a série observada no hos-

pedeiro H1 (origem do tráfego) e as séries do mecanismo de *polling* e sFlow observadas no elemento de rede SW3, localizado no centro do cenário avaliado.



(a) UDP gerado com *Iperf*.

(b) TCP utilizando *scp*.

(c) *Streaming* de vídeo utilizando VLC.

(d) Zoom na coleta do *streaming*.

Figura 7. Análise da taxa de transmissão com aplicações legadas.

Nas Figuras 7(a) e 7(b) é possível observar a maior precisão do mecanismo de *polling* frente ao mecanismo de amostragem do sFlow. A Tabela 3 apresenta os dados consolidados referentes às transmissões UDP e TCP, evidenciando no caso do mecanismo de *polling* observações de taxa média similares aos valores de H1 e H2, além de apresentar um valor de desvio padrão inferior ao sFlow.

Tabela 3. Dados consolidados referentes às transmissões UDP e TCP.

Análise	Média e Desvio Padrão (em Mbits/s)			
	$Origem_{H1}$	$Destino_{H2}$	$Polling_{SW1..5}$	$sFlow_{SW1..5}$
Figura 7(a)	1,064 +/- 0,038	1,064 +/- 0,041	1,067 +/- 0,090	1,076 +/- 0,168
Figura 7(b)	1,045 +/- 0,102	1,045 +/- 0,104	1,054 +/- 0,126	1,086 +/- 0,237

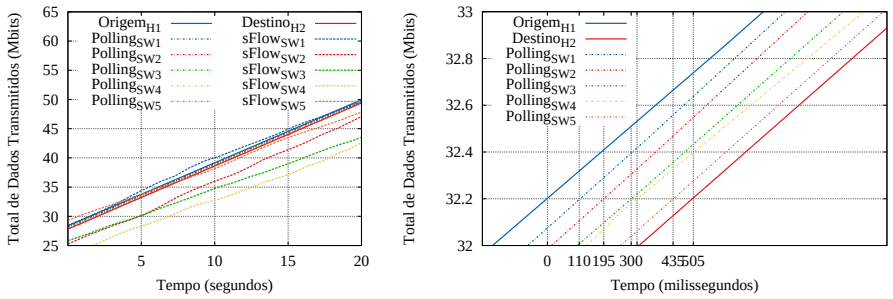
Especificamente para o *streaming* de vídeo apresentado na Figura 7(c), a natureza da aplicação possui variação frequente na taxa de transmissão, não justificando as análises de média e desvio padrão. Desta forma, a Figura 7(d) apresenta um intervalo de aproximadamente três minutos, a contar do inicio do experimento, onde fica caracterizado comportamento similar ao observado nas análises com UDP e TCP, evidenciando a maior precisão no caso do mecanismo de *polling*.

4.4. Análise do Atraso

A Figura 8 apresenta as séries temporais referentes ao total de dados transmitidos, coletadas durante uma transmissão UDP à taxa de 1 Mbps feita pelo gerador de tráfego *Iperf*, em um cenário sem a ocorrência de descarte de pacotes. As medições do mecanismo de

polling ocorreram em intervalos de cem milissegundos e a taxa de amostragem utilizada para o sFlow foi $N = 10$.

Neste experimento, o Mininet foi configurado para aplicar um atraso de cem milissegundos em cada um dos enlaces entre os elementos de rede OpenFlow, ou seja, espera-se que o tráfego experimente em torno de seiscentos milissegundos de atraso fim-a-fim. De maneira geral, o Mininet não conseguiu aplicar o tempo esperado em cada um dos enlaces, realizando um atraso fim-a-fim da ordem de quinhentos milissegundos. Esta configuração foi utilizada para demonstrar a metodologia de cálculo de atraso proposta, uma vez que sem esta configuração o Mininet apresenta atraso próximo a zero.



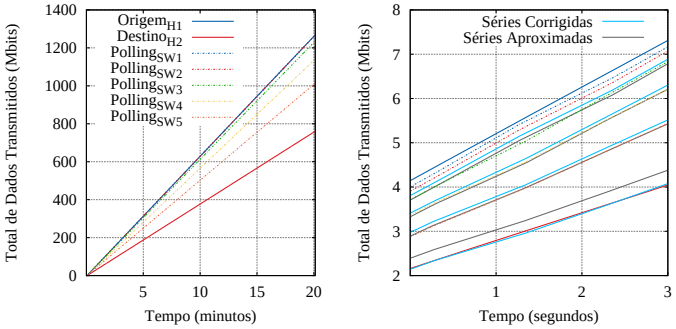
(a) Todas as séries temporais. (b) Zoom nas séries do mecanismo de *polling*.

Figura 8. Análise do atraso sem a ocorrência de descarte de pacotes.

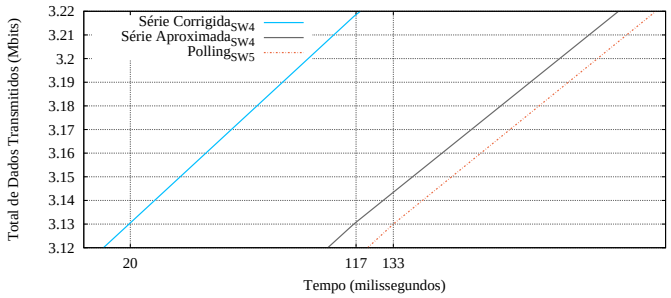
A Figura 8(a) evidencia a inviabilidade do mecanismo de amostragem do sFlow para efetuar a análise de atraso apresentada na Seção 3.3, uma vez que o total de Mbits apresentado nas séries temporais do sFlow são estimativas obtidas através das amostras enviadas ao coletor pelos agentes. Porém, na Figura 8(b) é possível observar o sequenciamento das séries temporais coletadas pelo mecanismo de *polling*, partindo do hospedeiro de origem H1, atravessando sequencialmente os elementos de rede de SW1 a SW5 até atingir o hospedeiro de destino H2. Os valores destacados no eixo X permitem visualizar o atraso entre as séries temporais observadas na altura de 32.2 Mbits transmitidos, sendo 110 ms, 85 ms, 105 ms, 10 ms, 125 ms, 70 ms, respectivamente.

A Figura 9 apresenta as séries temporais coletadas em um cenário onde há descarte de pacotes. Basicamente, foram transmitidos pacotes com MTU de tamanho variado e foram definidos no Mininet enlaces com MTU decrescente entre SW1 (1500 bytes) e SW5 (1496 bytes) para forçar o descarte de pacotes. Foram enviados cinco fluxos UDP com o gerador de tráfego *Iperf*, de tal forma que não haveria nenhum descarte no SW1, uma pequena ocorrência de descarte no SW2, gradualmente aumentando até permitir a vazão de um volume de tráfego no SW5. As medições do mecanismo de *polling* ocorreram em intervalos de cem milissegundos.

A Figura 9(a) apresenta todas as séries temporais durante a transmissão de aproximadamente vinte minutos, deixando evidente o distanciamento entre as séries ocasionado pelos descartes. A Figura 9(b) apresenta as séries corrigidas e aproximadas observadas em um período de três segundos, conforme discutido na Seção 3.3. A Figura 9(c) apresenta ambas as abordagens, detalhando a série temporal coletada do SW5 e as séries corrigida e aproximada referentes ao SW4. É possível observar a significativa perda de precisão do mecanismo aproximado, apresentando um atraso na ordem de 16 milissegundos, sendo



(a) Todas as séries temporais. (b) Séries Corrigidas e Aproximadas.



(c) Comparação entre as séries corrigida e aproximada para o cálculo do atraso.

Figura 9. Análise do atraso quando há a ocorrência de descarte de pacotes.

que a série corrigida apresenta um atraso na ordem de 113 milissegundos.

A coleta referente aos dados descartados foi feita utilizando o contador *Transmit Drops* previsto no OpenFlow apenas para a granularidade em nível de portas. Este contador mantém apenas o número de pacotes descartados. Os resultados da Figura 9 reforçam a sugestão feita na Seção 3.3, para que seja feita a inserção de contadores de total de bytes e total de pacotes descartados para todos os níveis de granularidade em uma futura alteração na especificação do OpenFlow [ONF 2013].

5. Conclusões e Trabalhos Futuros

Este artigo apresentou a plataforma de monitoramento SDNMon, capaz de utilizar diferentes abordagens de monitoramento na obtenção de dados referentes a situação do plano de dados em uma rede OpenFlow. Estes dados são processados e disponibilizados para acesso em outros módulos, tais como a aplicação de monitoramento desenvolvida.

O mecanismo de *polling* proposto neste artigo é capaz de enriquecer a visão da rede mantida pelos controladores OpenFlow, através de monitoramentos precisos e adaptativos à carga experimentada no plano de dados. Este enriquecimento da visão da rede é a base para a plataforma de provimento de qualidade de serviço sendo desenvolvida.

O desenvolvimento deste trabalho permitiu identificar possíveis extensões à especificação OpenFlow, sugerindo a inserção de novos contadores em todos os níveis de granularidade e a possibilidade de especificação de intervalos nas mensagens de *Statistic Request* para permitir uma melhor adaptabilidade por parte do mecanismo proposto.

Finalmente, propomos uma investigação sobre a utilização da abordagem de *push* no envio dos dados dos contadores mantidos pelos elementos de rede do plano de dados OpenFlow na direção do controlador. Ao invés de utilizar amostragens dos cabeçalhos dos pacotes conforme sFlow, o cenário a ser investigado seria o *push* das informações mantidas pelos contadores, evitando as frequentes consultas. A hipótese é que esta abordagem aumentaria a precisão dos monitoramentos e que todas estas medidas contribuiriam para o desenvolvimento de uma ferramenta capaz de prover uma tomografia da rede.

Agradecimento

Os autores gostariam de agradecer CAPES, CNPq e FAPEMIG pelo suporte ao desenvolvimento deste trabalho.

Referências

dos Passos Silva, D., Pontes, A. B., Avelar, E. A. M., and Dias, K. L. (2013). Uma Arquitetura para o Aprovisionamento de QoS Interdomínios em Redes Virtuais baseadas no OpenFlow. *31º Simp. Brasileiro de Redes de Computadores e Sistemas Distribuídos*.

Guedes, D., Vieira, L., Vieira, M., Rodrigues, H., and Nunes, R. (2012). Redes Definidas por Software: uma abordagem sistêmica para o desenvolvimento de pesquisas em Redes de Computadores. *Minicursos do Simpósio Brasileiro de Redes de Computadores-SBRC 2012*, 30(4):160–210.

Iperf (2014). `http://iperf.fr`.

Kim, H. and Feamster, N. (2013). Improving network management with software defined networking. *Communications Magazine, IEEE*, 51(2):114–119.

Kreutz, D., Ramos, F. M. V., Veríssimo, P., Rothenberg, C. E., Azodolmolky, S., and Uhlig, S. (2014). Software-Defined Networking: A Comprehensive Survey. *CoRR*, abs/1406.0440.

ONF (2013). OpenFlow Switch Specification - Version 1.4.0.

Phaal, P., Panchen, S., and McKee, N. (2001). InMon Corporation's sFlow: A Method for Monitoring Traffic in Switched and Routed Networks. RFC 3176 (Informational).

Phemius, K. and Bouet, M. (2013). Monitoring latency with OpenFlow. In *Proceedings of the 9th International Conference on Network and Service Management, CNSM 2013, Zurich, Switzerland, October 14-18, 2013*, pages 122–125.

sFlow (2014). `http://sflow.org`.

Suh, J., Kwon, T. T., Dixon, C., Felter, W., and Carter, J. B. (2014). Opensample: A low-latency, sampling-based measurement platform for commodity SDN. In *IEEE 34th International Conference on Distributed Computing Systems, ICDCS 2014, Madrid, Spain, June 30 - July 3, 2014*, pages 228–237.

van Adrichem, N. L. M., Doerr, C., and Kuipers, F. A. (2014). OpenNetMon: Network monitoring in OpenFlow Software-Defined Networks. In *2014 IEEE Network Operations and Management Symposium, NOMS 2014, Krakow, Poland, May 5-9, 2014*.

VLC (2014). `http://www.videolan.org/vlc/`.