CEEL - ISSN 2596-2221
Universidade Federal de Uberlândia
25 a 29 de novembro de 2019

ANÁLISE DE MÉTODO DE COMUNICAÇÃO PARA GERENCIAMENTO EM REDES DE LARGA ESCALA

Luiz Cláudio Theodoro*[1] e Raoni Exaltação Masson[2]

[1]FACOM - Universidade Federal de Uberlândia
[2]FEELT - Universidade Federal de Uberlândia

***Resumo* - O objetivo deste artigo é uma breve análise sobre a ferramenta de comunicação de dados chamada Telemetry. Com o aumento substancial da quantidade de elementos nas redes de computadores e a quantidade de métricas por eles geradas, protocolos tradicionais como o SNMP (*Simple Networking Management Procotocol*) e BGP (*Board Gateway Protocol*) que fazem o uso do modelo *pull*, impõem limites na escalabilidade dessas redes. A comunicação através da telemetria utiliza o modelo *push*, de modo a entregar dados de forma assíncrona eliminando a necessidade do *polling*, tornando o gerenciamento da rede escalável e eficiente, superando assim os limites impostos pelos protocolos tradicionais para as grandes redes.**

***Palavras-Chave*- Compactação de dados, gerenciamento de redes, métodos de comunicação, monitoramento de redes, protocolos de comunicação.**

COMMUNICATION METHOD ANALYSIS FOR LONG-SCALE NETWORK MANAGEMENT

***Abstract* - The purpose of this article is a brief review of the Telemetry data communications tool. With the substantial increase in the number of elements in the computer networks and the amount of metrics generated by them, traditional protocols such as Simple Networking Management Protocol (SNMP) and BGP (Board Gateway Protocol) use the pull model and directly limits scaling of these networks. Telemetry communication uses the push model to deliver data asynchronously, eliminating the need for polling, making network management scalable and efficient, thus overcoming the limits imposed by traditional protocols for large networks.**

***Keywords* - Communication methods, communication protocols, data compression, network management, network monitoring.**

I. INTRODUÇÃO

No inicio do desenvolvimento da Internet, a rede era composta por apenas uma pequena quantidade de elementos e, portanto, não demandava gerenciamento. A rede era relativamente simples e os problemas eram resolvidos manualmente através de algumas linhas de comandos [1]. Com o rápido avanço tecnológico ocorrido nas redes de computadores, sejam elas públicas (Internet) ou privadas, essas passaram rapidamente a serem utilizadas por centenas de milhões de pessoas, se tornando uma grande infraestrutura global.

A conexão de centenas ou até milhares de elementos aumenta substancialmente a complexidade de uma rede. Uma pequena falha de configuração, mal funcionamento de um equipamento ou quebra de segurança pode afetar milhares de usuários e sistemas distribuídos por uma cidade. A necessidade de gerenciar essa complexa infraestrutura se tornou vital. A ISO (*International Organization for Standardization*) criou um modelo de gerenciamento de redes bastante útil para alocar os cenários apresentador em um quadro mais estruturado. Foram definidas cinco áreas para o gerenciamento de redes, sendo essas [1]:

1. Gerenciamento de desempenho. O objetivo de gerenciar o desempenho da rede é obter, quantificar, medir, analisar e controlar a performance de diferentes elementos da rede. Esses elementos podem ser individuais como roteadores, *switchs*, links e *hosts* assim como conexões fim-a-fim através da rede;

2. Gerenciamento de falhas. O objetivo do gerenciamento de falhas é gerar *logs*, detectar e responder do modo mais rápido possível a eventuais falhas nos elementos da rede. Pode-se pensar no gerenciamento de falhas como ações mais imediatas para lidar com falhas transientes na rede;

3. Gerenciamento de configuração. Permite que o administrador de rede seja capaz que fiscalizar as configurações de hardware e software dos elementos da rede.

4. Gerenciamento de contas. O gerenciamento de contas permite que o administrador de redes consiga criar *logs*, especificar e controlar os acessos de usuários e elementos da rede;

5. Gerenciamento de segurança. O objetivo do gerenciamento de segurança é controlar o acesso à rede de

*luiz.theodoro@ufu.br

acordo com políticas definidas pela empresa ou administrador daquela rede.

Existem três principais componentes na arquitetura de redes: (1) a entidade gerenciadora; (2) os dispositivos de redes gerenciados; (3) e o protocolo de gerenciamento de redes. A entidade gerenciadora nada mais é do que uma aplicação que se situa, normalmente, numa central de gerenciamentos de redes. Ela é responsável por coletar, processar e analisar dados referentes aos elementos que compõem essa rede. Um dispositivo gerenciado nada mais é do que um dos elementos que fazem parte dessa rede gerenciada, ele pode ser um roteador, *switch*, *hub*, ou simplesmente um modem. O protocolo de gerenciamento é o que permite que a entidade gerenciadora monitore, configure e tenha consciência sobre a ocorrência de eventos como falhas e erros na rede [1].

Em paralelo com a evolução das redes, dezenas de protocolos foram desenvolvidos e progrediram para suprir as necessidades dos administradores de rede. Dois exemplo de protocolos amplamente conhecidos e utilizados na área de gerenciamento de redes são SNMP e BGP (uma breve explanação do funcionamento desses protocolos é feita nas Seção 2 e 3).

Entretanto, com a rápida expansão das redes e o aumento exponencial dos elementos que a compõem, protocolos como o SNMP e o BGP já não eram o suficiente para garantir a escalabilidade e permitir o desenvolvimento de ferramentas de gerenciamento e controle mais sofisticadas. Deste modo, novos métodos foram estudados e implementados de forma a facilitar o gerenciamento e monitoramento dessas grandes redes. Neste sentido, a telemetria veio como uma solução de comunicação para garantir uma alta escalabilidade, facilitar a implementação em redes de larga escala, e proporcionar a capacidade de desenvolvimento de ferramentas de gerenciamento que correspondem às necessidades que os administradores de redes tem para monitorar uma enorme quantidade de elementos.

O presente artigo tem como objetivo realizar uma análise sobre método de comunicação da telemetria através do seu estudo e implementação num roteador MX480 da JUNIPER. O projeto realizado visa automatização do monitoramento de roteadores de borda para otimização de seu gerenciamento, monitoramento, e agilidade por parte do administrador de redes na resposta à falhas, erros e eventos excepcionais que ocorram nessa rede.

A. *Organização do artigo*

O artigo está organizado da seguinte maneira: na Seção 2 estão explanados os pontos sobre o protocolo SNMP, amplamente conhecido e utilizado no gerenciamento de redes. Na Seção 3 é feita uma discussão sobre o BGP. Na Seção 4 os principais pontos sobre a telemetria são abordados (neste caso, em específico a telemetria dos equipamentos da *JUNIPER*). Na Seção 5 é detalhado o procedimento de implantação da telemetria. Na Seção 6, finalmente, é realizada a conclusão do artigo.

II. SNMP

O SNMP, ou *Simple Network Management Protocol*, é um protocolo de gerenciamento de redes amplamente conhecido e utilizado por administradores em todo o mundo. Seu desenvolvimento ocorreu junto com a evolução da Internet e, assim, ele traz um método bastante simples e prático de coletar informações dos elementos da rede.

O método que o SNMP coleta ou define essas informações nos elementos da rede é utilizando a MIB. A MIB (*Management Information Base*) nada mais é do que um banco virtual de informações que guarda valores que contemplam o estado atual de certos objetos MIB relativos aos elementos gerenciados. Os chamados objetos MIB podem conter: informações de estado, se uma das portas do roteador está ativa ou inativa; informações descritivas, sob qual domínio AS está o elemento; características físicas, quantidade de portas, sistema de hardware, ou até a cidade local na qual se encontra o dispositivo; ou informações detalhadas e especificas sobre diferentes tipos de protocolos como SNMP, BGP, OSPF, etc. Esses objetos são nomeados de forma hierárquica. Cada objeto é identificado por um nome e um número à si associado, este identificador é denominado OID (*Object Identifier*) [1].

Deste modo, o SNMP pode ser utilizado de dois modos diferentes. O primeiro e normalmente utilizado é comumente chamado de *request-response mode*. Este nome é dado pois é necessário enviar, através da entidade gerenciadora uma requisição para certo dispositivo da rede, e este irá responder ao pedido com a informação desejada. Esta requisição normalmente é um pedido de informação para monitorar ou definir algum objeto MIB associado àquele dispositivo. O segundo modo de se utilizar o SNMP é através da *trap-message*. Esta mensagem é "acionada"no dispositivo da rede e irá gerar uma resposta ao administrador de redes apenas na ocorrência de certos eventos. Funciona como uma *flag*. O administrador de redes normalmente utiliza o método de *trap* para descobrir de forma rápida, por exemplo, quando uma interface é derrubada, ou quando eventuais falhas e erros ocorrem em um dos dispositivos da rede [1].

Tabela 1: Tipos de PDU SNMP. (Tabela adaptada de KUROSE, 2013)

Tipos de PDU	**Descrição**
GetRequest	A entidade gerenciadora requisita a um dispositivo da rede o valor de um ou mais objetos MIB.
GetNextRequest	A entidade gerenciadora requisita a um dispositivo da rede o valor da próxima instância do objeto MIB. Utilizado para descobrir quais OIDs (identificadores) aquele dispositivo de rede possui.
GetBulkRequest	A entidade gerenciadora requisita a um dispositivo da rede os valores de uma grande quantidade de objetos MIB.
InformRequest	A entidade gerenciadora requisita a um dispositivo da rede informa à entidade gerenciadora valores de MIB que são remotos ao seu acesso.
SetRequest	A entidade gerenciadora requisita a um dispositivo da rede para mudar um valor de uma ou mais instâncias de objetos MIB.
Response	É a mensagem gerada em resposta a: GetRequest, GetNextRequest, GetBulkRequest, SetRequerst ou InformRequest.
Trap	Notificação assíncrona. O dispositivo da rede informa a entidade gerenciadora sobre a ocorrência de um certo tipo de evento.
Report PDU	Permite a comunicação entre agentes SNMP para reportar problemas no processamento de mensagens SNMP (implementado no SNMPv3).

O SNMP possuí três versões, sendo a primeira SNMPv1 teve sua RFC [RFC 1067] apresentada em 1988. O SNMPv1 foi duramente criticado por sua falta de segurança e as versões posteriores vieram para corrigir esse erro. Além do aprimoramento na segurança das mensagem, o SNMPv2, definido pelas RFC 1441 e RFC 1452 introduziu novos métodos de requisição de informação para coletar grande quantidade de dados. No geral o SNMP possui 7 tipos de mensagens, chamadas de PDU (*Protocol Data Units*), incluindo as introduzidas na versão 2. A Tabela 1 de forma resumida a descrição desses tipos de mensagens.

III. BGP

Para determinar rotas ótimas dentro de um mesmo AS são utilizados diversos protocolos, entre eles os mais utilizados são o RIP (*Routing Information Protocol*) e o OSPF (*Open Shortest Pathe First*). Um AS nada mais é do que um conjunto de elementos de redes sob a supervisão de uma mesma entidade gerenciadora, deste modo, protocolos como RIP e OSPF são denominados protocolos *Intra-AS*. O BGP (*Boarder Gateway Protocol*) é um protocolo *Inter-AS*, ou seja, utilizado para determinar os melhores caminhos de roteamento entre diferentes AS (*Autonomous System*) [1]. O BGP é um protocolo extremamente complexo e, neste artigo, é feito uma abordagem bem simples apenas para dar a ideia de sua importância no funcionamento da Internet.

No BGP, pares de roteadores trocam informações de roteamento por conexões TCP. Normalmente, há uma conexão TCP, denominada sessão BGP, para cada enlace que conecta diretamente dois roteadores em diferentes ASs. Como o TCP é um protocolo confiável, elimina-se, assim, a necessidade da implementação da confiabilidade no próprio BGP. A sessão BGP pode ser uma eBGP (sessão BGP externa), quando abrange dois ASs; ou uma iBGP (sessão BGP interna), abrange roteadores dentro do mesmo AS. Assim, o BGP permite que cada AS conheça os destinos que podem ser alcançados através de seus ASs vizinhos. Deste modo, no BGP os destinos não são *hosts*, mas sim prefixos, sendo que cada prefixo representará uma sub-rede ou um conjunto de sub-redes [1].

No BGP, cada AS tem um ASN (*Autonomous System Number*) exclusivo e é por ele identificado. Quando um roteador anuncia um prefixo para uma sessão BGP, incluí vários atributos BGP junto com o prefixo. Esse conjunto formado pelo prefixo com os atributos é denominado de rota. Dois atributos extremamente importantes são o *AS-PATH* e o *NEXT-HOP*. O *AS-PATH* contém a informação dos ASs pelos quais passou o anúncio para o prefixo. O *NEXT-HOP* é a interface do roteador que inicia o *AS-PATH*, e é utilizado pelos roteadores como um atributo para atualizar corretamente as suas tabelas de repasse. Além disso, os roteadores de bordas fazem o uso da política de importação para decidir se aceita ou não o anuncio de certa rota. Ele pode filtrar uma rota pois o AS não quer enviar tráfego por um dos ASs no AS-PATH, ou pois o roteador já possuí uma rota preferencial para o mesmo prefixo [1].

A quantidade de informações que um roteador de borda precisa processar e armazenar e a quantidade de decisões que precisa fazer é imensa. Esses roteadores são extremamente importantes na rede, pois fazem a conexão entre dois ou mais ASs, de modo que uma grande quantidade de dados irá trafegar através dele. Apesar de normalmente existir mais de uma rota para um prefixo, um erro, falha, ou sobrecarregamento no processamento de um roteador que esteja no caminho do seu enlace pode causar diversos problemas não apenas para a rede interna da operadora que provê o serviço de Internet, mas para todas as redes à ela conectadas.

IV. TELEMETRIA

A rápida expansão e desenvolvimento da Internet acarretou num aumento exponencial dos elementos da rede. Para que o funcionamento desta rede ocorra evitando grande problemas ou falhas, é necessário um gerenciamento rígido de diversas métricas desses elementos, o que ocasiona em um grande fluxo de dados gerados por protocolos como SNMP e BGP, por exemplo, limitando a sua eficiência e escalabilidade.

O JTI (*Juniper Telemetry Interface*) supera este limite aplicando o modelo *push*, eliminando assim a necessidade de *polling*. Neste modelo, um pedido ou requisição de transmissão é feito apenas uma vez e o elemento gerenciado passará a fazer o *stream* de dados periodicamente. Deste modo, o JTI é altamente escalável e suporta o gerenciamento de milhares de elementos de redes [2].

Uma das principais funções do JTI é a otimização no gerenciamento da rede através da utilização de *stream* de dados. Isso permite que um administrador de redes consiga monitorar variáveis desta rede e diagnosticar problemas, como congestionamento em um roteador em tempo real. Isso é uma característica amplamente desejada em uma ferramenta de monitoramento já que alguns problemas na rede devem ser imediatamente resolvidos pois podem afetar dezenas de milhares de usuários.

O JTI aplica dois métodos de exportação dos dados no formato gpb (*Google protocol buffer*)[3]:

1. *Native sensor*. Os sensores nativos conseguem exportar dados diretamente da *line card* ou NPU (*Network Processing Unit*), utilizando o protocolo UDP. O modelo de dados é definido pela própria *JUNIPER*, mas é um modelo aberto e extensível. Os dados são compactados através da utilização do gpb. É válido notar que apenas as versões mais recentes de roteadores da JUNIPER conseguem exportar dados BGP através do native sensor.

2. gRPC. O gRPC (*Google Remote Procedure Call*) é um *framework open-source* desenvolvido pela Google que utiliza os *protocol buffers* (um método de serialização de estrutura de dados) [4] através do *Routing Engine*. O modelo de dados é definido pelo *OpenConfig* e, do mesmo modo como no outro método, os dados são compactados utilizando o gpb.

O *Routing Engine* é um software da *JUNIPER* que roda em seu OS (*Operational System*). Este software é responsável por lidar com todos protocolos de roteamento, assim como o

processamento que controla as interfaces do roteador, os componentes do chassis e o sistema de gerenciamento. O processamento do software roda no topo do *kernel*, sendo que este interage com o PFE (*Packet Forwarding Engine*). Portanto, o *Routing Engine* é responsável pelo processamento de todo o sistema de roteamento e atualização da tabela de repasse de um roteador da *JUNIPER* [5].

Neste sentido, a grande vantagem da implementação dos native sensor é a utilização de uma arquitetura de distribuição da *JUNIPER* através do qual os dados gerados pelos sensores que forem configurados no elemento da rede são exportados diretamente da camada de dados [5]. Deste modo, não há utilização da camada de controle, conservando assim os recursos do roteador necessários para outras funções de maior urgência. É de extrema importância não onerar os roteadores para alcançar uma aplicação que possa realizar seu monitoramento em tempo real, já que os roteadores são elementos altamente requisitados no funcionamento da rede.

Uma desvantagem do native sensor é o envio dos dados de controle através de canais *in-band*. Isso significa que as informações de telemetria são enviadas no mesmo canal que é utilizado para transporte de dados como vídeo, voz, etc. Uma falha no canal ou um congestionamento no mesmo implicaria em atraso ou perda dos pacotes contendo as informações de controle. E neste caso, obviamente, há a necessidade da conexão *in-band* entre o elemento e o servidor de gerenciamento já que as informações do native sensor não podem ser enviadas por uma interface de controle, mas apenas por interfaces de dados.

A implementação da telemetria através do gRPC é um pouco mais complexa e de certo modo não traz os mesmo benefícios. Para tal, é necessário a instalação de um software chamado *Network Agent* nos roteadores. Este software roda no *Routing Engine* e funciona como um servidor gRPC, servindo também para encerrar as sessões gRPC iniciadas pelo sistema de gerenciamento do administrador de redes. Deste modo, há uma carga na capacidade de processamento do roteador. A conexão realizada por gRPC é através do protocolo TCP. Diferente do native sensor, é possível enviar os dados através da interface de controle, assim, pode-se garantir uma conexão mais estável e segura entre o elemento e o servidor de gerenciamento da rede.

V. DESCRIÇÃO DA IMPLEMENTAÇÃO

O projeto desenvolvido tem como objetivo a automatização do monitoramento de roteadores de borda para otimização de seu gerenciamento. Para alcançar tal objetivo, optou-se pela implementação da telemetria utilizando o método do native sensor, de modo a acrescentar uma carga mínima aos roteadores de borda e ainda assim ser capaz de desenvolver uma aplicação o mais próximo possível do tempo real.

Deste modo, foi utilizado um roteador MX480 da *JUNIPER* como roteador de teste para configuração e implementação da telemetria. O roteador foi hospedado no laboratório de homologação e teste da empresa e foi criado uma conexão *in-band* com o servidor utilizado para a coleta do *stream* de dados.

Para monitorar um recurso específico do elemento de rede é necessário a definição de um sensor. Para isto, é requerida a configuração de três componentes principais sejam feitas no elemento da rede:

1. *Sensor profile* - habilita o monitoramento dos recursos do elemento e permite que o administrador defina parâmetros relacionados como o servidor de destino do *stream* de dados. Cada *Sensor profile* pode gerar dados de apenas um sistema de recurso do elemento que será monitorado. Entretanto, é possível configurar mais de um *Sensor profile* para monitorar o mesmo recurso. Pode-se, ainda, como configuração opcional, utilizar expressões regulares para filtrar apenas as informações necessárias coletadas pelo sensor daquele recurso em específico. Caso nenhuma expressão regular seja utilizada o recurso do sistema será inteiramente monitorado [3].

2. *Export profile* - especifica os atributos para o processo de exportação dos dados coletados, assim como o tipo de protocolo utilizado para o transporte e o intervalo de tempo entre cada envio de dados. É necessário a configuração de ao menos um *Export profile*. Cada *Export profile* pode ser associado à múltiplos *Sensors profile*, mas apenas um *Export profile* pode ser associado à um *Sensor profile* em específico [3].

3. *Streaming server profile* - define os parâmetros do servidor que coletará o *stream* dos dados exportados pelo elemento de rede, assim como o endereço IP e o número da porta de destino. Por questões de redundância no envio e armazenamento das informações de telemetria é possível definir mais de um *Server profile* e associa-los à um Sensor profile em específico [3].

As definições desses parâmetros são bastantes simples e seguem uma estrutura básica. Após acessar o elemento de rede - o que usualmente é feito através da utilização do ssh (*Secure Socket Shell*) - utiliza-se o CLI (*Command-Line Interface*) para realizar essas configurações. O ssh nada mais é do que um protocolo de redes que dá ao administrador da rede um método seguro de acessar um elemento que a compõem. Já o CLI é um método de interação com um software onde o usuário (neste caso o administrador de redes) consegue emitir comandos para um programa na forma de texto em linhas sucessivas.

Como exemplo, abaixo serão expostos os comandos básicos para configurar um perfil de exportação com o nome de **TELEMETRIA**, endereço IP **10.100.100.100**, porta **21000**, intervalo de geração e exportação de dados **5 segundos**, formato da estrutura dos dados **gpb** e protocolo de transporte **UDP**:

1. Definindo um nome para o perfil de exportação
   *set services analytics export-profile **TELEMETRIA***

2. Definindo o endereço IP fonte dos pacotes exportados
   *set services analytics export-profile TELEMETRIA local-address **10.100.100.100***

3. Definindo o número da porta fonte dos pacotes exportados
   *set services analytics export-profile TELEMETRIA local-port **21000***

4. Definindo o intervalo, em segundo, no qual o sensor irá gerar os dados de telemetria
   *set services analytics export-profile TELEMETRIA reporting-rate **5***

5. Definindo o formato no qual os dados exportados serão estruturados
   *set services analytics export-profile TELEMETRIA format **gpb***

6. Definindo o protocolo de transporte que será utilizado para carregar os dados de telemetria na camada IP
   *set services analytics export-profile TELEMETRIA transport **UDP***

O mesmo método foi utilizado para configurar um *Streaming server profile* e os *Sensors profiles*. Após a configuração dos três componentes, foi realizado a configuração do servidor para a captura da stream de dados. Para esta etapa, assumindo que o servidor já tenha uma conexão *in-band* com o roteador, basta abrir a porta de destino que foi configurada no *Streaming server profile*. Deste modo, utilizou-se uma ferramenta de rede chamada *netcat*. Esta ferramenta é *open-source* e tem diversas utilidades como permitir, através de CLIs, que o administrador de rede consiga abrir portas e estabelecer conexões TCP e UDP. Supondo que a porta de destino utilizada no *Streaming server profile* seja a porta 30000, a sintaxe básica utilizada pelo *netcat* é dada por:

nc [-options] <host> port[s] [ports]

Para abrir uma conexão UDP no próprio servidor para a porta 30000:

nc -ul 0.0.0.0 30000 > data.gpb

A opção *-u* permitem ativar o modo UDP e a opção *-l* executa o *netcat* em modo de escuta, ou seja, ele vai "forçar"a conexão abrindo uma porta e ficar na escuta pela espera do *stream* de arquivos. O endereço de IP *0.0.0.0*, em um contexto de servidores, normalmente é utilizado para referenciar todos os endereços IPv4 na *host* local, assim, será possível alcançar o servidor mesmo que ele tenha mais de um endereço IP à ele associado. Por fim, a porta na qual o *netcat* foi configurada à escutar foi a 30000, e neste caso os dados capturados pelo *netcat* estão sendo redirecionados para a escrita no arquivo *data.gpb*.

Após a captura do *stream* de dados é necessário fazer a decodificação para poder realizar o tratamento das informações. Como já dito anteriormente, os dados são comprimidos em formato gpb que é um estrutura de serialização de dados da *Google* extremamente pequena, rápida e simples, proporcionando assim as características necessárias para que seja possível fazer o *stream* desses dados. Assim, para realizar a decodificação, são necessários uma série de arquivos disponibilizados pela *JUNIPER* no formato *.proto*. Após a aquisição desses arquivos, utiliza-se o compilador da *Google protoc* para fazer a decodificação da seguinte forma:

#protoc –decode arquivo.proto -I PATH < data.gpb > saída.txt

A opção *–decode* serve para indicar que será realizado a decodificação do arquivo *data.gpb*. Logo em seguida coloca-se o nome do arquivo *.proto* retirado do site da *JUNIPER*. A opção *-I PATH* se refere ao caminho de diretórios no qual o arquivo *.proto* estará disposto. Neste caso, como exemplo, o resultado da decodificação está sendo enviado para o arquivo *saída.txt* e será armazenado para posteriormente ser tratado e colocado em um banco de dados.

VI. CONCLUSÕES

As características presentes na telemetria traz consigo diversas consequências futuras. Por usar o modelo *push*, esse método de comunicação é altamente escalável, permitindo o monitoramento de milhares de dispositivos gerenciados. A implementação da desta ferramenta é simples e abre um leque de possibilidade para o administrador de redes. Com o seu emprego é possível automatizar ferramentas de monitoramento de redes com softwares mais responsivo, dinâmico e confiável do que os já conhecidos baseados em protocolos como o SNMP, por exemplo. Além disso, através da utilização do *native sensor*, a telemetria pode ser usada com o objetivo de aliviar a carga no processamento nos roteadores. Deste modo, a tecnologia se mostrou bastante promissora e promete ser uma grande ferramenta para automatização na área de gerenciamento e monitoramento de redes de larga escala.

REFERÊNCIAS

[1] J. F. Kurose, and K. W. Ross. *Computer networking: A top-down approach featuring the Internet*, Pearson, 6ª ed, 2013.

[2] JUNIPER NETWORKS *Overview of the Junos Telemetry Interface*. Acedido em 10 de junho de 2019, em: `https://www.juniper.net/documentation/en_US/junos/topics/concept/junos-telemetry-interface-oveview.html`.

[3] Juniper Networks Inc. *Junos Telemetry Interface Feature Guide*. Acedido em 05 de março de 2019, em: `https://www.juniper.net/documentation/en_US/junos/information-products/pathway-pages/junos-telemetry-interface/junos-telemetry-interface.pdf`.

[4] gRPC Documents *A high performance, open-source universal RPC framework*. Acedido em 20 de junho de 2019, em: `https://grpc.io/`.

[5] JUNIPER NETWORKS *Routing Engine Architecture*. Acedido em 20 de junho de 2019, em: `https://www.juniper.net/documentation/en_US/release-independent/junos/topics/concept/routing-engine-m7i-architecture.html`.