dx.doi.org/10.7437/NT2236-7640/2014.02.003
Notas Técnicas, v. 4, n. 2, p. 1–13, 2014

Ferramenta IPERF: geração e medição de Tráfego TCP e UDP

IPERF tool: generation and evaluation of TCP and UDP data traffic

Pedro Henrique Diniz da Silva* e Nilton Alves Júnior†
Centro Brasileiro de Pesquisas Físicas,
Rua Dr. Xavier Sigaud 150,
Rio de Janeiro, RJ - 22290-180, Brasil

Resumo: Este trabalho demonstra a utilização da ferramenta de medição e geração de tráfego de dados *Transport Control Protocol* (TCP) e *User Datagram Protocol* (UDP) conhecida como IPERF. Uma ferramenta útil para engenheiros e administradores de rede, do tipo cliente/servidor, desenvolvida com código livre e gratuita. Ela permite, como uma de suas principais vantagens, a alteração de parâmetros TCP, tal como o tamanho da janela TCP, além de exibir relatórios de banda nos modos TCP e UDP, e relatórios de *jitter* e perda de pacotes no modo UDP. Neste documento, são exibidos alguns casos em que a ferramenta é útil e como ela pode ser utilizada para auxiliar na análise de desempenho de redes TCP/IP.

Palavras-chave: Redes de computadores; Medição de desempenho de redes de computadores; Geração de tráfego TCP e UDP.

Abstract: This work demonstrates the use of Transport Control Protocol (TCP) and User Datagram Protocol (UDP) data traffic measurement and generation tool known as IPERF. This tool is useful for network engineers and administrators, developed based on the client/server, open source and free models. It allows you, as one of its main advantages, to modify TCP parameters such as the TCP window size, in addition to view reports of bandwidth in TCP and UDP modes, and reports of jitter and packet loss in UDP mode. In this document, we report some cases where the IPERF tool is useful and how it can be used to assist in analyzing the performance of TCP/IP networks.

Keywords: Computer networks; Performance evaluation of computer networks; TCP and UDP traffic generation.

1. INTRODUÇÃO

Os grandes experimentos de física necessitam cada vez mais de uma infraestrutura de computação de ponta e dedicada. Atualmente, todos esses experimentos compartilham quantidades enormes de dados para se beneficiar de uma capacidade descentralizada de processamento e armazenamento computacional. O meio físico para que esses compartilhamentos aconteçam são as redes de computadores.

Esses grandes experimentos sempre estiveram presentes na pesquisa e desenvolvimento da área de computação (em especial é possível citar o protocolo http/www e mais recentemente as pesquisas em tecnologias para operação de uma rede a 100 Gbps [1; 2]).O Centro Brasileiro de Pesquisas Físicas (CBPF/MCTI), por meio de seu *Grupo de Computação e Instrumentação*, tem tido uma atuação de destaque nesta linha, seja através da coordenação da RedeRio de Computadores (a rede acadêmica para pesquisa e ensino do Estado do Rio de Janeiro) e da liderança técnica da REDECOMEP-Rio (que foi inaugurada recentemente com um *backbone* de 10 Gbps sobre um suporte tecnológico de até 1.9Tbps em Multiplexação Densa por Divisão de Comprimento de Onda - DWDM, do inglês *Dense Wavelength Division Multiplexing*[3], em mais de 300 km de fibras óticas instaladas)[1],[2][4; 5; 6; 7; 8]. No entanto, os desafios para as redes acadêmicas de alta velocidade, em especial como suporte à instrumentação científica remota em física, são ainda imensos. Apesar da infraestrutura local ter melhorado significativamente e as redes terem atingido um desempenho de altíssima velocidade em seu ponto final, obter um ótimo desempenho entre as redes envolvidas fim a fim é ainda um desafio. É necessário pesquisar, entender e desenvolver tecnologias e ferramentas de análise de desempenho de redes com o envolvimento de diversos protocolos em várias camadas de

*Electronic address: phds@cbpf.br
†Electronic address: naj@cbpf.br

[1] Em junho de 2014, foi inaugurada a RedeRio Metropolitana (REDECOMEP-Rio), uma parceria da RedeRio/FAPERJ com a RNP/MCTI. Esta é uma infraestrutura de fibras óticas próprias que formam uma rede de alta velocidade para as instituições de ensino, ciência, tecnologia, inovação e de governo na cidade do Rio de Janeiro. A REDECOMEP-Rio interconecta 86 pontos, pertencentes a 51 instituições acadêmicas na região metropolitana do Rio de Janeiro estendendo-se por mais de 300 quilômetros em um backbone de 10Gbps em tecnologia DWDM atingindo uma banda agregada de até 1,9 Tbps. A coordenação e o projeto técnico da REDECOMEP-Rio ficaram sob a responsabilidade do CBPF (para mais informações vide: http://www.redecomep.rnp.br/?consorcio=2).

[2] Atualmente, os projetos RedeRio/FAPERJ e REDECOMEP Rio de Janeiro estão sob coordenação de operações da equipe do Coordenação de Engenharia de Operações (CEO) da RedeRio/FAPERJ. A operação das redes está sob a responsabilidade da Coordenação de Atividades Técnicas (CAT) do Centro Brasileiro de Pesquisas Físicas (CBPF/MCTI).

comunicação fim a fim. Pesquisar e avaliar o desempenho de redes de computadores, especialmente aquelas dedicadas a grandes experimentos colaborativos de física, envolve a análise de métricas que caracterizem seu comportamento, desde a sua topologia física (fim a fim) quanto todas as especificações operacionais e projetos lógicos. Para isto devem ser feitas escolhas corretas e análises de diversas métricas nos diferentes níveis funcionais de uma rede. Essas métricas podem ser obtidas através de técnicas de medição ativa(vide seção 2) que consistem na injeção de pacotes de controle na rede, ou passivas[3], em que são apenas observados os pacotes que trafegam pela rede. As características da rede que são geralmente medidas e analisadas pelas ferramentas para a análise de desempenho de rede são:

1. Largura de banda e taxa de uso dos canais de comunicação envolvidos (*throughput* ou vazão): a largura de banda indica a capacidade máxima de transmissão nominal de uma conexão, enquanto que o *throughput* ou vazão trata-se da quantidade de dados transferidos de um ponto a outro da rede em um determinado período de tempo.

2. Perda de pacotes: trata-se do número de pacotes enviados por um emissor, mas não recebidos pelo receptor, sendo, geralmente, obtido como um percentual de perda de pacotes na rede. Esse percentual é estimado com base no número total de pacotes perdidos dividido pelo total de pacotes enviados. Para uma boa análise dessa métrica deve-se levar em consideração diversos fatores como a caracterização do tamanho dos *buffers* de armazenamento temporário, a análise da saturação de enlaces e o congestionamento da rede fim a fim.

3. Tempo de resposta (latência ou atraso da rede): estima o tempo que um pacote demora para sair de sua origem e chegar ao seu destino. Em geral, a latência da rede é medida como o atraso de ida e volta de um pacote na rede, também conhecido como *Round Trip Time* (RTT).

4. *Jitter:* é obtido como o desvio padrão do atraso de pacotes enviados em sequência. A medição da variação da latência impacta no uso da rede para transferência de grandes volumes de dados, em especial para aplicações que necessitem de alto desempenho em tempo real.

Como parte integrante de um projeto coordenado pela Coordenação de Atividades Técnicas (CAT) do CBPF, cujo objetivo principal é garantir e otimizar o acesso do CBPF aos grandes experimentos internacionais e a troca de grandes volumes de dados sob a Internet em alta velocidade e alta disponibilidade, é que desenvolvemos o presente documento. Com este projeto pretendemos pesquisar e desenvolver as ferramentas para avaliação de desempenho inter-redes, com o objetivo de melhorar a taxa de utilização dos recursos computacionais disponíveis fim a fim dos projetos desenvolvidos no CBPF. Dessa forma, este documento surge como uma necessidade de se compreender o funcionamento da ferramenta de medição ativa de rede IPERF, como uma solução de código aberto, para realizar medições de vazão dos enlaces da RedeRio/FAPERJ e do projeto REDECOMEP-Rio.

Apesar de haverem disponíveis na Internet diversos tutorias sobre a ferramenta IPERF, além de páginas Web que demonstrem o seu funcionamento, há a carência de artigos científicos que abordem de uma maneira clara e objetiva o seu funcionamento e utilização. Ademais, não há na literatura científica textos que demonstrem em conjunto explicações sobre a ferramenta e suas métricas observáveis, além de casos práticos da vida profissional de administradores e engenheiros de rede em que essa aplicação seja bem adequada e simples de ser utilizada. Desse modo, neste trabalho, estamos interessados em demonstrar o funcionamento da ferramenta IPERF e alguns casos específicos de sua utilização, que sejam úteis à vida prática de engenheiros e administradores de redes, inclusive para os responsáveis pelo CEO da RedeRio/FAPERJ.

Nas próximas seções, são dadas uma visão geral das ferramentas de medição ativas de rede e da ferramenta IPERF. Em sequência, são abordados alguns tópicos essenciais para a compreensão do funcionamento da ferramenta em relação aos protocolos da camada de transporte da pilha de protocolos TCP/IP, o UDP e o TCP. Em seguida, são apresentados os procedimentos de instalação e de utilização da ferramenta IPERF e também são expostos alguns casos específicos de políticas de controle de banda, testes de balanceamento de carga e geração de fluxos UDP de altas taxas de transferência. Por fim, são apresentadas as conclusões referentes à análise do funcionamento da ferramenta IPERF e dos estudos de casos.

2. VISÃO GERAL DAS FERRAMENTAS DE MEDIÇÃO ATIVAS DE REDE

As ferramentas de medição ativa de rede proveem a abordagem mais simples e mais flexível para a estimação da vazão da rede [11]. Essas ferramentas são fundamentais, pois permitem analisar o tráfego de dados e os pontos críticos de uma rede, avaliando a sua topologia a fim de verificar se a qualidade dos serviços oferecidos pela rede está sendo atendida, além de auxiliar no planejamento futuro da rede.

Esse tipo de ferramenta de medição tem como objetivo injetar pacotes de teste na rede, para a partir de então medir o desempenho da rede avaliando como esse tráfego de teste se comporta na rede.

É nesse contexto que se inserem os geradores de tráfego como ferramentas de medição ativas de redes de computadores, assim como a ferramenta IPERF. Os geradores de tráfego permitem que sejam gerados fluxos de dados com características específicas para simular o acesso a uma aplicação, como o tráfego de voz ou vídeo, por exemplo. Com

[3] O método de medição passivo analisa o desempenho de redes através do uso de dispositivos passivos, que não interferem no tráfego da rede quando realizam suas medições, i.e., esses dispositivos apenas observam o tráfego corrente que passa pelo ponto de observação. Exemplos de ferramentas de medição passiva são as que utilizam o protocolo Simple Network Management Protocol (SNMP), como CACTI [9] e MRTG [10].

isso, é possível testar a eficiência das configurações de *Quality of Service*(QoS), *Access Control List (*ACLs), *traffic-shapping*[4], *Virtual Private Networks* (VPNs), dentre muitas outras.

Diversas são as ferramentas de medição ativa disponíveis na Internet, tanto de código aberto como Clink [12], IPERF [13], Netperf [14], Pathrate [15], NUTTCP [16], quanto pagas como PathView [17]. Cada uma dessas ferramentas utiliza técnicas diferentes para determinar a capacidade de tráfego, porém não é objetivo desse documento realizar a análise nem comparação de todas essas ferramentas. Portanto, é apresentado, a partir da seção 3, uma análise do funcionamento da ferramenta IPERF.

3. VISÃO GERAL DA FERRAMENTA IPERF

O IPERF é uma ferramenta que reúne em uma única aplicação o relatório da análise de várias métricas, como a capacidade máxima fim-a-fim a nível de transporte, o *jitter* e a perda de pacotes. A sua utilização simplifica a análise de problemas de rede por parte dos administradores de redes. Desse modo, essa foi uma das motivações de sua escolha como ferramenta para análise nesse documento.

Em se tratando de ferramentas de medição ativa de redes, o IPERF é uma ferramenta amplamente utilizada para medir a vazão e a qualidade de um *enlace* de rede. Ele permite a análise da qualidade de um *enlace* segundo algumas das métricas apontadas na seção 1, como a seguir:

1. *Jitter* (variação do atraso entre os pacotes de dados sucessivos): pode ser medida por meio do envio de fluxos de pacotes do *User Datagram Protocol (UDP)* com a ferramenta IPERF.

2. Perda de datagramas: pode ser medida, também, com testes UDP.

3. Vazão: medidas através de testes TCP (*Transfer Control Protocol*) e UDP por meio do IPERF.

O IPERF é um *software* livre desenvolvido pelo NLANDR/DAST (*National Laboratory for Applied Network Research/Distributed Applications Support Team)* no início da década de 2000, como uma alternativa para medição do rendimento da banda de redes de computadores através de TCP e UDP [18]. A ferramenta não possui interface gráfica por padrão, sendo sua operação relativamente simples através da CLI (*Command Line Interface*). Porém, existe também um *front-end* gráfico desenvolvido em linguagem Java conhecido como Jperf, que permite a visualização dos resultados graficamente. No entanto, a utilização desse *front-end* não será abordada nesse documento.

O IPERF funciona, basicamente, em um modelo cliente/servidor, onde o servidor atende às solicitações de testes e o cliente inicia as sessões de testes. Ele está disponível como *open source* compilável ou binário executável para diversas plataformas incluindo Windows, Linux, Solaris, Mac OS, OpenBSD e FreeBSD.

Os testes de performance realizados pelo IPERF podem ser utilizados para validar uma rede, tanto em segmentos cabeados quanto sem fio. Os testes podem ser utilizados, por exemplo, para identificar o mau desempenho de uma rede ou até mesmo para desqualificar a porta de um *switch* ou roteador defeituoso.

Por padrão, o protocolo utilizado pelos testes com IPERF é o TCP, porém o protocolo UDP pode ser também utilizado. Dentre as principais características da ferramenta em relação a cada um dos protocolos citados podem ser destacadas:

1. TCP
   (a) Medida da largura de banda
   (b) Reporta o tamanho do MSS/MTU (*Maximum Segment Size*/*Maximum Transmission Unit*)
   (c) Suporta a modificação do tamanho de janelas TCP
   (d) Conexões simultâneas tanto para o cliente quanto para o servidor

2. UDP
   (a) Cliente pode criar fluxos UDP de largura de banda variada
   (b) Mede a perda de pacotes
   (c) Mede o *jitter*
   (d) Cliente e servidor podem ter múltiplas conexões simultâneas (indisponível no Windows)

Entretanto, vale-se ainda ressaltar que muitos fatores podem limitar o rendimento de conexões TCP, como perdas, congestionamento da rede e entregas fora de ordem. Desse modo, testes com UDP proveem maior transparência, onde podemos medir diretamente as perdas, o *jitter* e as entregas fora de ordem.

4. PROTOCOLOS TCP E UDP

O modelo de protocolos TCP/IP que a Internet implementa tem dois protocolos de transporte principais: o UDP e o TCP.

O *User Datagram Protocol* (UDP), definido na RFC 768 [19], utiliza o protocolo IP para transportar uma mensagem de uma máquina para outra, e provê o mesmo mecanismo de entrega de datagramas sem conexão e inseguro que o IP. Ele não utiliza confirmações para garantir que as mensagens cheguem ao destino, além de não ordenar as mensagens entrantes e não provê um *feedback* para controlar a taxa com que a informação trafega entre as máquinas [20]. Dessa forma, uma aplicação que faz uso do protocolo UDP para entrega de mensagens entre *hosts* aceita a total responsabilidade de lidar com o problema de confiabilidade, incluindo a perda de mensagens, duplicação, atraso, entrega fora de ordem e perda de conectividade.

[4] O termo traffic shapping (em português, modelagem de tráfego) trata da limitação da vazão de tráfego de redes de dados, em geral por meio de técnicas de priorização de tráfego.

O UDP utiliza o protocolo IP para transportar mensagens, mas adiciona a capacidade de distinguir entre múltiplos destinos dentro de um determinado *host*, através de mecanismos conhecidos como portas. As portas, em conjunto com o endereço IP, identificam a fonte e o destino de cada mensagem. Em suma, ele nada mais é do que uma interface para o protocolo IP, cuja função básica é servir como multiplexador e demultiplexador de processos para o tráfego de informações do protocolo IP.

Assim sendo, os programas aplicativos que dependem de UDP funcionam muito bem em um ambiente local, mas falham de modo dramático quando são utilizados na Internet. Desse modo, é onde o IPERF se faz presente, permitindo aos profissionais de Tecnologia da Informação (TI) testarem a conectividade UDP.

O *Transport Control Protocol* (TCP), definido na RFC 793 [21] é o protocolo da camada de transporte orientado à conexão e que oferece um serviço de entrega confiável, ao contrário do que ocorre com o UDP. Os protocolos de transporte confiáveis usam uma única técnica fundamental conhecida como confirmação positiva com retransmissão (também conhecida como ACK). A técnica requer que um receptor se comunique com a fonte, enviando de volta uma mensagem de confirmação (ACK) quando recebe os dados. O remetente mantém um registro de cada pacote que envia e espera por uma confirmação antes de enviar o próximo pacote. A Figura 4. 1 mostra como o mais simples protocolo de transferência com confirmação positiva transfere os dados.

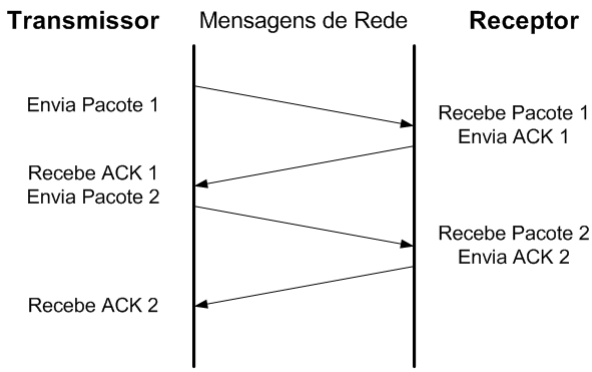

Figura 4. 1: Exemplo simples de confirmação positiva com retransmissão. Os eventos ocorridos no transmissor e no receptor estão apresentados à esquerda e à direita. Cada linha diagonal cruzando o meio mostra a transferência de uma mensagem através da rede.

Nessa técnica de retransmissão, o transmissor inicia um *timer* depois de transmitir um pacote. Quando o *timer* expira, o transmissor assume que o pacote foi perdido e retransmite-o.

O protocolo TCP implementa uma forma mais complexa de confirmação positiva com retransmissão, conhecida como técnica de janela deslizante. Essa técnica permite uma melhor utilização da largura de banda disponível, pois permite que o transmissor envie múltiplos pacotes antes de receber uma confirmação, e permite que se impeça que um transmissor rápido sobrecarregue um receptor lento. Essa técnica de controle é conhecida como controle de fluxo do TCP.

O funcionamento do mecanismo de janela deslizante pode ser visto como uma sequência de pacotes a serem transmitidos, como pode ser visualizado na Figura 4.2. O protocolo implementa uma pequena janela de tamanho fixo na sequência de dados a serem transmitidos e transmite todos os pacotes que se encontram nessa janela.

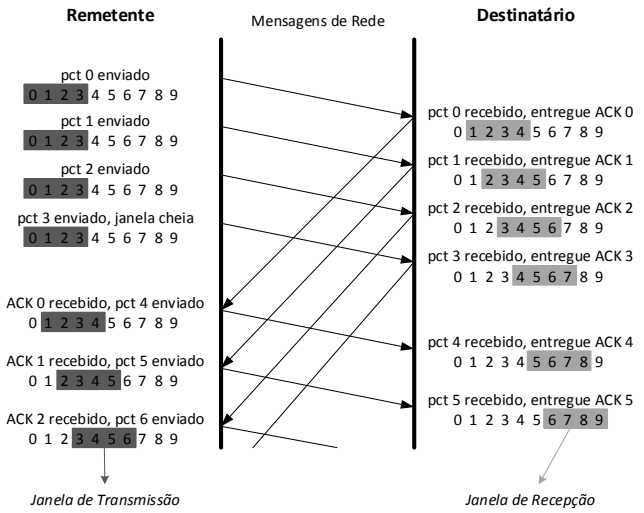

Figura 4. 2: Protocolo de janela deslizante com quatro pacotes na janela. A janela desliza de modo que o pacote em seguida à ela pode ser enviado quando uma confirmação for recebida para o primeiro pacote da mesma.

Ao passo que as confirmações vão sendo recebidas a janela vai deslizando. Desse modo, o número de pacotes que podem ser enviados enquanto não são reconhecidos em qualquer dado momento é limitado pelo tamanho da janela.

O desempenho dos protocolos de janelas deslizantes depende do tamanho da janela e da velocidade de transmissão de pacotes na rede. Aumentando-se o tamanho da janela deslizante pode-se eliminar quase que completamente o tempo ocioso da rede [20].

O protocolo TCP implementa um mecanismo de janela deslizante especializado para resolver dois problemas primordiais: a transmissão eficiente e o controle de fluxo. O mecanismo de janela do TCP torna possível enviar vários segmentos antes de uma confirmação chegar ao receptor, mantendo a rede ocupada e aumentando o rendimento. Esse mecanismo permite que o receptor restrinja a transmissão até que haja espaço suficiente em seu *buffer* para acomodar mais dados, resolvendo o problema de controle de fluxo fim-a-fim. Um ponto importante também a ser observado é que o mecanismo de janela deslizante do TCP opera a nível de octetos, e não no nível de segmentos ou pacotes, numerando sequencialmente os octetos do fluxo de dados.

Resumidamente, podemos dizer que o TCP permite o envio de x segmentos de tamanho máximo conhecido como MSS (*Maximum Segment Size*) antes da recepção de um ACK, que é o tamanho da janela TCP. Logo, a taxa de transmissão máxima de uma sessão TCP pode ser controlada pelo tamanho da janela TCP. Portanto, a vazão para uma determinada conexão TCP transmitindo a cada RTT segundos pode ser dada por [22]:

$$\text{Vazão} = \frac{x\ \text{[segmentos]} \cdot \text{MSS [bytes]}}{\text{RTT [segundos]}} = \frac{\text{Tamanho da Janela TCP [bytes]}}{\text{RTT [segundos]}} \quad (1)$$

Além do mecanismo de janela deslizante o TCP implementa também um mecanismo de controle de congestionamento que visa adaptar a taxa de envio de dados à carga da rede. Esse mecanismo é, basicamente, composto por dois algoritmos: o algoritmo de partida lenta e o de prevenção de congestionamento [22].

O algoritmo de partida lenta é utilizado no estabelecimento de uma conexão TCP, com o objetivo de fazer com que a taxa de transmissão do transmissor convirja para a capacidade de transmissão disponível na rede. Em seguida, é iniciada a fase de prevenção de congestionamento, utilizada para controlar o tamanho da janela de transmissão dinamicamente [23].

Com o auxílio dos algoritmos de partida lenta e controle de congestionamento, do lado do transmissor uma janela de congestionamento (TCP CWND) é calculada baseada na taxa de perda de pacotes, e do lado do receptor é utilizada uma janela de recepção (TCP RWND) que informa ao transmissor quantos bytes ele (receptor) é capaz de aceitar em um dado momento. Por fim, o menor valor entre a TCP CWND e a TCP RWND anunciada determina quantos bytes podem ser realmente transmitidos pelo transmissor em um dado momento.

4.1. Ajustando a Janela TCP

Segundo a RFC 6349 [24], para que se evite a limitação da performance TCP ambas as janelas TCP (RWND e CWND) devem ser maiores do que o produto do atraso de largura de banda ou *Bandwidth-Delay Product* (BDP):

$$\text{BDP [Bytes .s]} = \frac{\text{RTT [segundos]} \cdot \text{menor largura de banda do link [bits]}}{8} \quad (2)$$

A Figura 4.3 auxilia na compreensão de como a configuração incorreta da janela TCP pode influenciar na performance TCP.

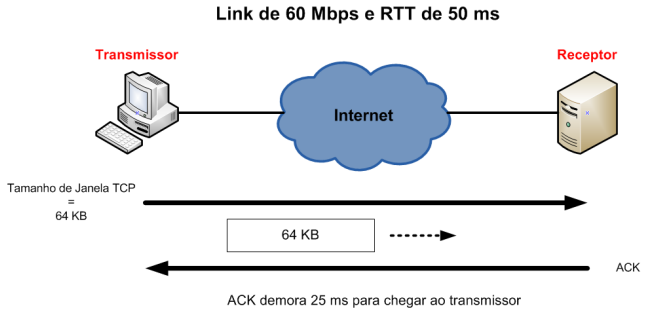

Figura 4. 3: Exemplo para o cálculo do produto da largura de banda pelo atraso.

Supondo um *enlace Wide Area Network* (WAN) de 60 Mbps com atraso de ida e volta de 50 ms ($RTT = 50ms$) e tamanho de janela do transmissor de 64 KB, o BDP seria de 384 KB. Esse BDP representa seis vezes o tamanho da janela do transmissor. Dessa forma, o transmissor alcançaria uma vazão de, aproximadamente, 10 Mbps.

Porém, quando o tamanho da janela TCP excede o BDP, o limite máximo de FPS (*Frames Per Second*) do *enlace* é atingido, e então a fórmula para o cálculo da Vazão máxima TCP alcançável é [24]:

$$\text{Vazão TCP}_{\text{máx}}[bps] = FPS^{5}_{\text{máx}} \times (\text{MTU[Bytes]-40}) \times 8 \quad (3)$$

onde FPS_{mx} é o número máximo de quadros por segundo para o *enlace* em questão e o MTU é o tamanho máximo do quadro. Além disso, esse cálculo é baseado no protocolo IP versão 4 com cabeçalhos TCP/IP de 20 bytes cada (20 para TCP e 20 para IP) dentro do MTU, onde cada *byte* representa 8 *bits*.

Por exemplo, levando em consideração o FPS máximo de uma conexão *Fast Ethernet* de 100 Mbps valendo 8127 quadros por segundo e o MTU *Ethernet* de 1500 Bytes, logo a Vazão TCP máxima é de 94,9 Mbps.

Enfim, para se calcular a Vazão TCP em função do tamanho da janela TCP, a seguinte fórmula pode ser utilizada:

$$\text{Vazão TCP [bps]} = \frac{\text{tamanho da janela TCP [Bytes]} \cdot 8}{\text{RTT[segundos]}} \quad (4)$$

Vale ainda ressaltar que pacotes perdidos podem degradar seriamente uma conexão TCP. Por exemplo, em uma LAN *Gigabit Ethernet* conectada a uma WAN a 10 Mbps pode resultar em casos em que a WAN operará de modo indevido, com rajadas de gigabits, gerando perdas de pacotes e retransmissões TCP.

[5] O presente documento não visa demonstrar os métodos para se obter o FPS de cada tipo de enlace, porém a RFC 6349 [24] demonstra os principais casos práticos de sua obtenção.

5. INSTALAÇÃO DA FERRAMENTA IPERF

Nesta seção, serão abordados os procedimentos de instalação da ferramenta IPERF em plataformas UNIX/Linux e Windows. Em ambos os sistemas a ferramenta pode ser facilmente instalada.

Como o IPERF funciona segundo o modelo cliente/servidor, então, em geral, após sua instalação é necessário que em um *host* a ferramenta seja executada como cliente, enquanto que em outro a mesma seja executada como servidor.

A Figura 5. 1 apresenta um esquema onde o IPERF é instalado em máquinas Linux e Windows, servindo como servidor e cliente, respectivamente.

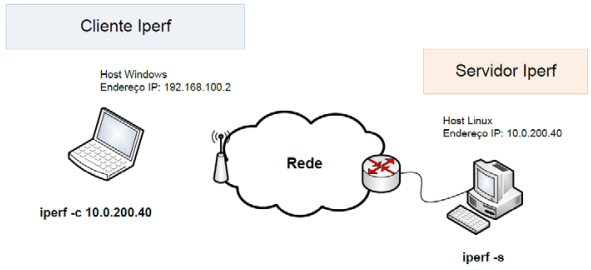

Figura 5. 1: Exemplo de funcionamento na ferramenta IPERF em arquitetura cliente/servidor.

A aplicação é disponibilizada amplamente em diversos locais espalhados pela web, seja como versão compilada ou código fonte. Neste documento, será feito referência aos códigos disponibilizados pelo Fórum Francês do IPERF, encontrado em: *http://iperf.fr*.

A versão utilizada em todos os testes realizados nesse documento foi a versão 2 mais recente, o *release* 2.0.5.

Atualmente, existe também a versão 3 da ferramenta, conhecida como iperf3. Ela é uma nova implementação disponibilizada por uma comunidade global de desenvolvedores, cujo intuito é simplificar e diminuir o código base da aplicação, e prover um conjunto de bibliotecas com funcionalidades que possam ser utilizadas em outros programas. Além disso, essa versão implementa um conjunto de novas funções, como o informe do número de pacotes TCP retransmitidos e da taxa de utilização média da CPU. Porém, como essa mais nova versão ainda está em fase de desenvolvimento inicial e apresenta uma série de *bugs* ainda não corrigidos optamos por utilizar a versão 2.x nesse documento.

5.1. GNU/Linux

O aplicativo pode ser baixado e utilizado como um arquivo executável, sem a necessidade de se realizar sua instalação. Para utilizá-lo basta seguir os procedimentos descritos abaixo:

```
root@servidor:~# wget http://iperf.fr/download/iperf_2.0.5/iperf/iperf_2.0.5-2_i386
root@servidor:~# chmod +x iperf_2.0.5-2_i386
root@servidor:~# mv iperf_2.0.5-2_i386 /usr/bin/iperf
```

5.2. Windows

Assim como para o GNU/Linux, no Windows, o IPERF também pode ser encontrado como um arquivo executável. Para utilizá-lo basta realizar o download da ferramenta em: *http://iperf.fr/download/iperf_2.0.5/iperf-2.0.5-2-win32.zip*, extrair o arquivo e executá-lo.

6. PARÂMETROS DISPONÍVEIS

Nesta seção, serão apresentados todos os parâmetros disponibilizados pela ferramenta e a descrição de cada um deles. Como o objetivo deste documento não é esgotar todas as opções disponíveis pela aplicação, serão apresentadas somente as consideradas como mais importantes e as mais utilizadas. São elas: *configuração padrão; formatação da saída; medida da largura de banda bidirecional e bidirecional simultânea; alteração do tamanho da janela TCP; alteração da porta TCP, do intervalo de relatórios e do tempo de transmissão de dados.*

Após a seção 6 será apresentada a seção 7 que servirá como um complemento, onde serão demonstrados alguns casos práticos em que a ferramenta IPERF pode ser utilizada por administradores e engenheiros de rede para realizar testes de desempenho de rede.

6.1. Descrição dos Parâmetros Disponíveis

Segundo a seção de ajuda do IPERF podemos verificar todas as possíveis opções disponíveis pela aplicação:

Cliente/Servidor:	
Opções	**Descrição**
-f, –format	Formato do relatório: Kbits,Mbits, Kbytes, MBytes
-i, –interval	Segundos entre os relatórios periódicos de vazão
-l, –len	Comprimento do *buffer* para se ler ou escrever (padrão de 8 KB)
-m, –print_mss	Imprime o tamanho máximo do segmento TCP (MTU – cabeçalho TCP/IP)
-o, –output <arquivo>	Emite o relatório ou a mensagem de erro para o arquivo especificado
-p, –port	Porta do servidor a escutar/ se conectar
-u, –udp	Utiliza UDP em vez de TCP
-w, –window	Tamanho da janela TCP (tamanho do *buffer* de *socket*[6])
-B, –bind <host>	Se liga a um <host>, uma interface ou endereço multicast
-C, –compatibility	Para utilizar com versões antigas, não envia mensagens extras
-M, –mss	Estabelece o tamanho máximo do segmento TCP (MTU – 40 bytes)
-N, –nodelay	Estabelece nenhum atraso TCP, desabilitando o algoritmo de Nagle[7]
-V, –IPv6Version	Estabelece o domínio para IPv6
Servidor:	
Opções	**Descrição**
-s, –server	Roda em modo servidor
-U, –single_udp	Roda em mode UDP *single thread*
-D, –daemon	Roda o servidor como um daemon
Cliente:	
Opções	**Descrição**
-b, –bandwidth	Para UDP, largura de banda para enviar em bits/s (padrão de 1 Mbit/s, implica a opção –u)
-c, –client <host>	Roda em modo cliente, conectando-se ao <host>
-d, –dualtest	Realiza um teste bidirecional simultaneamente
-n, –num	Número de bytes a serem transmitidos (em vez de –t)
-r, –tradeoff	Realiza um teste bidirecional individualmente
-t, –time	Tempo em segundos para transmitir dados (padrão de 10 segundos)
-F, –fileinput <name>	Entra com os dados a serem transmitidos de um arquivo
-I, –stdin	Entra com os dados a serem transmitidos do stdin
-L, –listenport	Porta para receber testes bidirecionais de volta
-P, –parallel	Número de *threads* de clientes em paralelo para serem executadas
-T, –ttl	*Time-to-Live*, para multicast (padrão 1)
-Z, –linux-congestion <algo>	Estabelece o algoritmo de controle de congestionamento (somente para Linux)
Diversos:	
Opções	**Descrição**
-x, –reportexclude [CDMSV]	Exclui os relatórios C (conexão) D (dados) M (multicast) S (definições) V (servidor)
-y, –reportstyle C	Relatório com valores separados por vírgulas
-h, –help	Imprime a mensagem de ajuda e sai
-v, –version	Imprime as informações de versão e sai

Tabela 6. 1: Opções do IPERF.

6.2. Configuração de Parâmetros

Para cada um dos casos apresentados a seguir consideraremos o IP do cliente IPERF como 192.168.0.3 e o do servidor como 192.168.0.4. A Figura 6. 1 demonstra essa configuração.

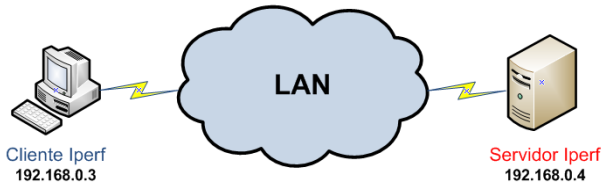

Figura 6. 1: Configuração de cliente/servidor.

6.2.1. *Configurações Padrão*

Por padrão, o cliente IPERF se conecta ao servidor IPERF na porta TCP 5001.

[6] Um socket de redes TCP estabelece um elo de comunicação entre duas aplicações que estão conectadas em uma rede. São definidos como a combinação de um endereço IP e o número de uma porta do protocolo TCP [25].

[7] O algoritmo TCP/IP de Nagle foi projetado para evitar problemas com pacotes pequenos, chamados de tinygrams, em redes lentas. O algoritmo diz que uma conexão TCP/IP pode ter somente um segmento pequeno pendente que ainda não fora confirmado. A definição de "pequeno"varia, mas geralmente é definido como "inferior ao tamanho do segmento"que para redes *ethernet* é de cerca de 1500 bytes.

Servidor	Cliente
#iperf -s	#iperf -c 192.168.0.3
----------	----------
Server listening on TCP port 5001 TCP window size: 85.3 KByte (default)	Client connecting to 192.168.0.3,TCP port 5001 TCP window size: 16.0 KByte (default)
----------	----------
[4] local 192.168.0.4 port 5001 connected with 192.168.0.3 port 38110 [ID] Interval Transfer Bandwidth [4] 0.0-10.0 sec 112 MBytes 93.9 Mbits/sec	[3] local 192.168.0.4 port 38110 connected with 192.168.0.3 port 5001 [ID] Interval Transfer Bandwidth [3] 0.0-10.0 sec 112 MBytes 94.2 Mbits/sec

6.2.2. *Formatação da Saída*

A opção –f permite exibir a saída nos seguintes formatos: (b) bits, (B) bytes, (k) kilobits, (K) kilobytes, (m) megabits, (M) megabytes, (g) gigabits e (G) gigabytes.

Servidor	Cliente
#iperf -s	#iperf -c 192.168.0.3 -f M
----------	----------
Server listening on TCP port 5001 TCP window size: 85.3 KByte (default)	Client connecting to 192.168.0.3, TCP port 5001 TCP window size: 0.02 MByte (default)
----------	----------
[4] local 192.168.0.3 port 5001 connected with 192.168.0.4 port 38111 [ID] Interval Transfer Bandwidth [4] 0.0-10.0 sec 112 MBytes 93.9 Mbits/sec	[3] local 192.168.0.4 port 38111 connected with 192.168.0.3 port 5001 [ID] Interval Transfer Bandwidth [3] 0.0-10.0 sec 112 MBytes 11.2 MBytes/sec

6.2.3. *Medida de Largura de Banda Bidirecional*

O servidor IPERF se conecta de volta ao cliente permitindo a medição da largura de banda de modo bidirecional individualmente. Por padrão, somente a largura de banda do cliente ao servidor é medida.

Servidor	Cliente
#iperf -s	#iperf -c 192.168.0.3–r
----------	----------
Server listening on TCP port 5001 TCP window size: 85.3 KByte (default)	Server listening on TCP port 5001 TCP window size: 85.3 KByte (default)
----------	----------
[4] local 192.168.0.3 port 5001 connected with 192.168.0.4 port 56648 [ID] Interval Transfer Bandwidth [4] 0.0-10.1 sec 113 MBytes 93.9 Mbits/sec	----------
----------	Client connecting to 192.168.0.3, TCP port 5001 TCP window size: 66.8 KByte (default)
Client connecting to 192.168.0.4, TCP port 5001 TCP window size: 65.2 KByte (default)	----------
----------	[5] local 192.168.0.4 port 56648 connected with 192.168.0.3 port 5001 [ID] Interval Transfer Bandwidth
[4] local 200.20.94.60 port 38335 connected with 200.20.94.64 port 5001 [4] 0.0-10.0 sec 112 MBytes 93.6 Mbits/sec	[5] 0.0-10.0 sec 113 MBytes 94.9 Mbits/sec [4] local 192.168.0.4 port 5001 connected with 192.168.0.3 port 38335 [4] 0.0-10.0 sec 112 MBytes 93.4 Mbits/sec

6.2.4. *Medição da Largura de Banda Bidirecional Simultânea*

Por padrão, somente a largura de banda do cliente para o servidor é mensurada. Para realizar medições bidirecionais simultâneas, utiliza-se o argumento –d.

Servidor	Cliente
#iperf -s	#iperf -c 192.168.0.3–d
----------	----------
Server listening on TCP port 5001 TCP window size: 85.3 KByte (default)	Server listening on TCP port 5001 TCP window size: 85.3 KByte (default)
----------	----------
[4] local 192.168.0.3 port 5001 connected with 192.168.0.4 port 38118	----------
----------	Client connecting to 192.168.0.3, TCP port 5001 TCP window size: 72.9 KByte (default)
Client connecting to 192.168.0.4, TCP port 5001 TCP window size: 42.2 KByte (default)	----------
----------	[3] local 192.168.0.4 port 38118 connected with 192.168.0.3 port 5001
[6] local 192.168.0.3 port 35503 connected with 192.168.0.4 port 5001 [ID] Interval Transfer Bandwidth	[5] local 192.168.0.4 port 5001 connected with 192.168.0.3 port 35503 [ID] Interval Transfer Bandwidth
[4] 0.0-10.0 sec 111 MBytes 92.9 Mbits/sec [6] 0.0-10.0 sec 31.4 MBytes 26.2 Mbits/sec	[3] 0.0-10.0 sec 111 MBytes 93.1 Mbits/sec [5] 0.0-10.1 sec 31.4 MBytes 26.2 Mbits/sec

6.2.5. *Alteração do Tamanho da Janela TCP*

O tamanho da janela TCP pode variar entre 2 e 65.535 bytes, de acordo com o cabeçalho TCP. Em especial, em sistemas Linux quando especificado o tamanho do *buffer* TCP com argumento –w, o *kernel* aloca o dobro do tamanho especificado.

Servidor	Cliente
#iperf –s –w 13KB	#iperf -c 192.168.0.3–w 13KB
----------	----------
Server listening on TCP port 5001 TCP window size: 13.0 KByte	Client connecting to 192.168.0.3,TCP port 5001 TCP window size: 13.0 Kbyte
----------	----------
[324] local 192.168.0.3 port 5001 connected with 192.168.0.4 port 38120 [ID] Interval Transfer Bandwidth [324] 0.0-10.0 sec 104 MBytes 87.1 Mbits/sec	[136] local 192.168.0.4 port 38120 connected with 192.168.0.3 port 5001 [ID] Interval Transfer Bandwidth [136] 0.0-10.0 sec 104 MBytes 87.1 Mbits/sec

6.2.6. *Alteração da Porta TCP, Intervalo de Relatórios e Tempo de Transmissão de Dados*

O argumento –p pode ser utilizado para alterar a porta de comunicação TCP do servidor IPERF. Entretanto, ela deve ser alterada em ambos cliente e servidor, onde o padrão é a porta TCP 5001. O argumento –t especifica o tempo de duração dos testes (duração padrão de 10 segundos) e o argumento –i especifica o intervalo entre os relatórios de largura de banda medida em segundos.

Servidor	Cliente
#iperf –s –p 10000	#iperf -c 192.168.0.3–p 10000 –t 20 –i 5
----------	----------
Server listening on TCP port 10000 TCP window size: 85.3 KByte (default)	Client connecting to 192.168.0.3, TCP port 10000 TCPwindow size: 8.00 KByte (default)
----------	----------
[4] local 192.168.0.3 port 10000 connected with 192.168.0.4 port 57008 [ID] Interval Transfer Bandwidth [4] 0.0-20.3 sec 1.14 MBytes 471 Kbits/sec	[136] local 192.168.0.4 port 57008 connected with 192.168.0.3 port 10000 [ID] Interval Transfer Bandwidth [136] 0.0-5.0 sec 296 KBytes 485 Kbits/sec [136] 5.0-10.0 sec 288 Kbytes 472 Kbits/sec [136] 10.0-15.0 sec 288 KBytes 472 Kbits/sec [136] 15.0-20.0 sec 288KBytes 472 Kbits/sec [136] 0.0-20.4 sec 1.14 Mbytes470 Kbits/sec

6.2.7. *Testes UDP*

A opção de testes UDP fornece informações sobre o *jitter* e a perda de pacotes. O *jitter* é a variação (desvio-padrão) dos tempos de chegada de pacotes, ou seja, o *jitter* pode ser considerado como a variação da latência [26]. Para aplicações como transmissão de vídeo e áudio (VoIP por exemplo), não importa se pacotes demoram 10ms, 20ms ou 100ms para chegarem ao receptor contanto que o tempo de transmissão seja constante. Para o caso de chamadas VoIP, altas taxas de *jitter* podem interromper uma ligação. O problema de *jitter* é, em geral, atenuado com o armazenamento dos fluxos em *buffers* do lado do receptor, o que não afeta a largura de banda, mas aumenta o retardo suavizando a flutuação.

Vale ressaltar que os resultados reportados pelo servidor são ligeiramente mais precisos, uma vez que o transmissor pode calcular a taxa de transmissão logo após realizar

Servidor	Cliente
#iperf –s –u	#iperf -c 192.168.0.3–u –b 50m
----------	----------
Server listening on UDP port 5001 Receiving 1470 byte datagrams UDP buffer size: 110 KByte (default) ---------- [3] local 192.168.0.3 port 5001 connected with 192.168.0.4 port 54729 [ID] Interval Transfer Bandwidth Jitter Lost/Total Datagrams [3] 0.0-10.0 sec 59.6 MBytes 50.0 Mbits/sec 0.126 ms 0/42531 (0%) [3] 0.0-10.0 sec 1 datagrams received out-of-order	Client connecting to 192.168.0.3, UDP port 5001 Sending 1470 byte datagrams UDP buffer size: 110 KByte (default) ---------- [3] local 192.168.0.4 port 54729 connected with 192.168.0.3 port 5001 [ID] Interval Transfer Bandwidth [3] 0.0-10.0 sec 59.6 MBytes 50.0 Mbits/sec [3] Sent 42532 datagrams [3] Server Report: [3] 0.0-10.0 sec 59.6 MBytes 50.0 Mbits/sec 0.125 ms 0/42531 (0%) [3] 0.0-10.0 sec 1 datagrams received out-of-order

a última escrita em seu *buffer* de transmissão, isto é, antes de os dados terem efetivamente percorrido o enlace.

6.2.8. *MSS – Maximum Segment Size*

O MSS (*Maximum Segment Size*) ou Tamanho Máximo do Segmento é a maior quantidade de dados, em bytes, que pode ser transportada em um *frame* sem ser fragmentado. Como o tamanho dos cabeçalhos TCP e IP valem 40 bytes sem levar em consideração as opções do cabeçalho IPv4 e o MTU é o tamanho máximo em *bytes* em um *frame*, então o MSS pode ser calculado da seguinte forma:

$$\begin{aligned} MSS &= MTU - (cabeçalho\ TCP + cabeçalho\ IP) \\ &= MTU - 40\ bytes - opções\ \ do\ cabeçalho\ IPv4 \end{aligned}$$

Servidor	Cliente
#iperf –s	#iperf -c 192.168.0.3–m
----------	----------
Server listening on TCP port 5001 TCP window size: 85.3 KByte (default) ----------	Client connecting to 192.168.0.3,TCP port 5001 TCP window size: 16.0 KByte (default) ---------- [3] local 192.168.0.4 port 41053 connected with 192.168.0.3 port 5001 [ID] Interval Transfer Bandwidth [3] 0.0-10.0 sec 2967912595171227 bits 0.00 (null)s/sec [3] MSS size 1448 bytes (MTU 1500 bytes, ethernet)

6.2.9. *Testes em Paralelo*

A opção de realizar ou não conexões TCP múltiplas em paralelo depende, diretamente, do tamanho do BDP em relação ao tamanho da janela TCP configurada. Por exemplo, para uma largura de banda de 100 Mbps e RTT de 1 ms, então o BDP vale aproximadamente 13 KB. Nesse caso, para uma janela TCP de 8 KB o número de conexões TCP para ocupar a largura de banda disponível é 2.

Servidor	Cliente
#iperf –s	#iperf -c 192.168.0.3–P 2
----------	----------
Server listening on TCP port 5001 TCP window size: 8.00 KByte (default) ---------- [280] local 192.168.0.3 port 5001 connected with 192.168.0.4 port 50936 [308] local 192.168.0.3 port 5001 connected with 192.168.0.4 port 50937 [ID] Interval Transfer Bandwidth [308] 0.0-10.0 sec 54.0 MBytes 45.3 Mbits/sec [280] 0.0-10.0 sec 54.0 MBytes 45.3 Mbits/sec [SUM] 0.0-10.0 sec 108 MBytes 90.5 Mbits/sec	Client connecting to 192.168.0.3, TCP port 5001 TCP window size: 8.00 KByte (default) ---------- [148] local 192.168.0.4 port 50937 connected with 192.168.0.3 port 5001 [136] local 192.168.0.4 port 50936 connected with 192.168.0.3 port 5001 [ID] Interval Transfer Bandwidth [148] 0.0-10.0 sec 54.0 MBytes 45.3 Mbits/sec [136] 0.0-10.0 sec 54.0 MBytes 45.3 Mbits/sec [SUM] 0.0-10.0 sec 108 MBytes 90.5 Mbits/sec

7. ESTUDOS DE CASO

Nesta seção, são apresentados alguns casos reais em que a utilização do IPERF auxilia na homologação de redes que fazem uso de técnicas de políticas de controle de banda e de técnicas de balanceamento de carga, e auxilia também na geração de fluxos UDP de altas taxas de transmissão para medição de largura de banda em enlaces onde não é possível utilizar um servidor IPERF.

7.1. Políticas de Controle de Banda

As políticas de controle de banda são amplamente utilizadas por *Internet Service Providers* (ISPs) de modo a limitar a banda estabelecida em contrato.

Esse documento não visa demonstrar as técnicas de implementação desse tipo de política. Ele visa apenas demonstrar como a ferramenta IPERF pode ser utilizada para verificar e testar a aplicação dessas políticas.

A Figura 7.1 demonstra o *layout* da rede utilizada nos testes. Temos ambos os servidor e cliente IPERF conectados a um roteador. O cliente 192.168.0.2 está conectado à interface f0/2 do roteador e o servidor está conectado à interface f0/1 do roteador. Nesse caso, está sendo aplicada uma política de limitação de tráfego de 15 Mbps entrante na interface f0/2.

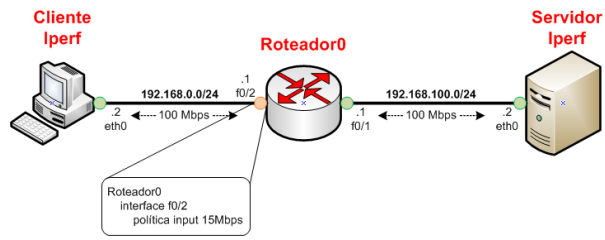

Figura 7. 1: Layout da rede para testes de política de controle de banda.

Na Figura 7.2, podemos verificar o comando aplicado no cliente e a sua respectiva saída antes de serem aplicadas as políticas de controle. É possível observar que foram gerados 94,5 Mbps de tráfego UDP pelo cliente e foram reportados 94,4 Mbps e nenhuma perda pelo servidor.

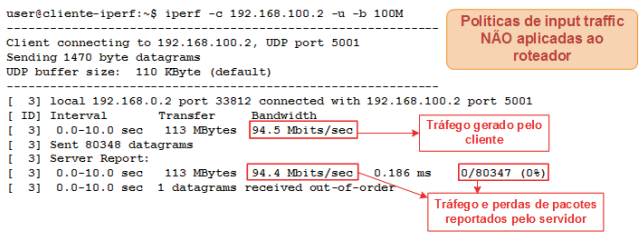

Figura 7. 2: Comando IPERF aplicado ao cliente antes da aplicação das políticas ao roteador.

Na Figura 7.3, podemos verificar o comando após serem aplicadas as políticas ao roteador. Podemos notar que apesar de serem gerados 94,5 Mbps pelo cliente, assim como no exemplo anterior sem as políticas aplicadas, após as mesmas serem aplicadas o tráfego reportado pelo servidor se limitou a 14,2 Mbps, apresentando 85% de perdas. Isso de-

monstra as políticas de limitação fazendo com que os pacotes entrantes na interface f0/2 sejam perdidos.

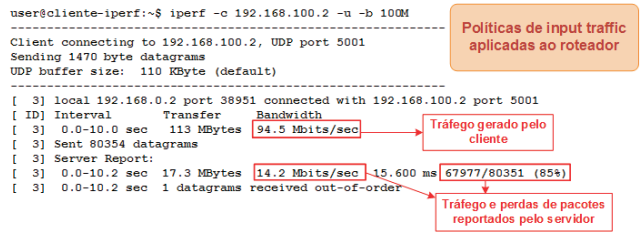

Figura 7. 3: Comando IPERF aplicado ao cliente após a aplicação das políticas ao roteador.

7.2. Testes de Balanceamento de Carga

As técnicas de balanceamento de carga ou *load balancing* são amplamente utilizadas por provedores de serviços de Internet de modo a tornar mais eficiente o uso da largura de banda disponível. Essas técnicas remetem a uma funcionalidade dos roteadores de distribuir pacotes através de múltiplos enlaces baseado em informações de roteamento.

Um balanceamento de carga efetivo tenta fazer o uso mais eficiente da banda disponível e distribui o tráfego entre diversas rotas, baseando-se em protocolos de roteamento, e/ou interfaces distintas, baseando-se em protocolos de agregação de *enlaces* como *Link Aggregation Control Protocol* (LACP), por exemplo [27]. O algoritmo responsável por realizar a distribuição de tráfego depende de cada implementação individualmente e, com isso, sua eficiência pode variar. Especificamente, tratando-se de equipamentos Cisco, equipamentos esses que são utilizados no *backbone* da RedeRio/FAPERJ e REDECOMEP Rio de Janeiro, são suportados dois modos de balanceamento de carga. São eles: o balanceamento baseado em pacotes e o baseado em fluxos (ou destinos).

Cada um dos modos citados acima possui uma aplicabilidade diferente. No modo por destino todos os pacotes destinados a um dado IP são entregues através de um mesmo caminho, preservando assim a ordem dos pacotes, mas com um uso desigual dos enlaces. No modo por pacotes, cada pacote é entregue através de enlaces diferentes, garantindo um uso igual de cada um dos enlaces. Porém, nesse caso os pacotes podem chegar fora de ordem ao destino, devido a atrasos diferentes em cada um dos enlaces da rede.

O modo de balanceamento padrão apresentado pelos roteadores Cisco através dos mecanismos de roteamento dinâmico como os *Open Shortest Path First* (OSPF), *Interior Gateway Routing Protocol* (IGRP) e *Intermediate System to Intermediate System* (IS-IS) é o modo por destino. Como o *backbone* da RedeRio/FAPERJ e REDECOMEP opera através de uma dessas configurações, vamos exemplificar um método simples para verificar o funcionamento do balanceamento utilizando a ferramenta IPERF.

Como em modo UDP, o IPERF não possui controle de fluxo nem controle de congestionamento, então o cliente tenta enviar dados ao servidor somente a taxa especificada pelo parâmetro **–b**. Assim, como em modo UDP não ocorre a transmissão confiável de dados, não existindo a necessidade de que o servidor envie confirmações ao cliente ao receber os dados, não há a necessidade de um servidor escutando na porta UDP especificada pelo cliente. Portanto, podemos utilizar somente um cliente IPERF gerando tráfego em modo UDP e utilizar alguma outra ferramenta passiva de monitoramento de redes para checar as taxas de transmissão atingidas em cada enlace.

No teste demonstrado na Figura 7.4 foram utilizados 6 *switches*/roteadores Cisco ME 3400E e um cliente IPERF gerando tráfego UDP. Nesse caso, 4 roteadores (R1, R2, R3 e R4) foram utilizados para simular um *backbone* com *Interior Gateway Protocol* (IGP), especificamente o OSPF, com balanceamento de carga por fluxos e 2 deles (R5 e R6) foram utilizados para simular dois *enlaces* com operadoras distintas. Dois fluxos distintos de 50 Mbps foram gerados no mesmo cliente com destino para R5 e R6, durante 12 horas (43200 segundos). Cada um dos comandos utilizados para gerar cada um dos fluxos pode ser encontrado a seguir:

Fluxo 1:$iperf –c 10.0.40.2 –u –b 50m –t 43200 –i 2
Fluxo 2:$iperf –c 10.0.40.2 –u –b 50m –t 43200 –i 2

Desse modo, podemos verificar as técnicas de balanceamento de tráfego operando através da utilização da ferramenta IPERF, por meio da geração de tráfego UDP, sem a necessidade de um servidor em operação. Casos similares a esse podem ser de grande valia quando for necessário testar enlaces, mas não for possível o acesso a um servidor IPERF para responder às solicitações.

Esse documento não visa demonstrar as técnicas de balanceamento utilizadas nem como implementá-las, porém as mesmas podem ser encontradas em [28]. Para obtermos as imagens referentes às taxas de transmissão em cada uma das interfaces de R1 foi utilizada a ferramenta RRDTool [29]. Essa ferramenta é simples e fácil de utilizar, sendo amplamente utilizada para monitoramento de redes de computadores.

7.3. Geração de Tráfego UDP de Altas Taxas

Para caminhos em que sejam necessários um espaço de *buffers* grande, como é o caso de caminhos em que o RTT é alto por exemplo, é necessário que algumas opções de alta performance, discutidas a seguir, sejam ativadas.

A maioria dos sistemas operacionais suportam limites de *buffer* de transmissão e recepção por conexão separados, os quais podem ser configurados pelo usuário, pela aplicação ou outro mecanismo, contanto que estejam dentro dos limites máximos de memória.

Os tamanhos dos *socket buffers* padrões podem ser alterados por controles globais do sistema operacional. O ajuste manual desses *buffers* é a maneira mais simples de se aumentar o desempenho de aplicações quaisquer, como o caso do IPERF.

Nessa seção, são exemplificados como os parâmetros de *buffers* em um sistema Linux devem ser alterados para aumentarmos a taxa de geração de tráfego UDP com o IPERF. O ajuste desses parâmetros em outros tipos de sistemas e uma explicação mais detalhada sobre transferência de alto desempenho podem ser encontrados em [30].

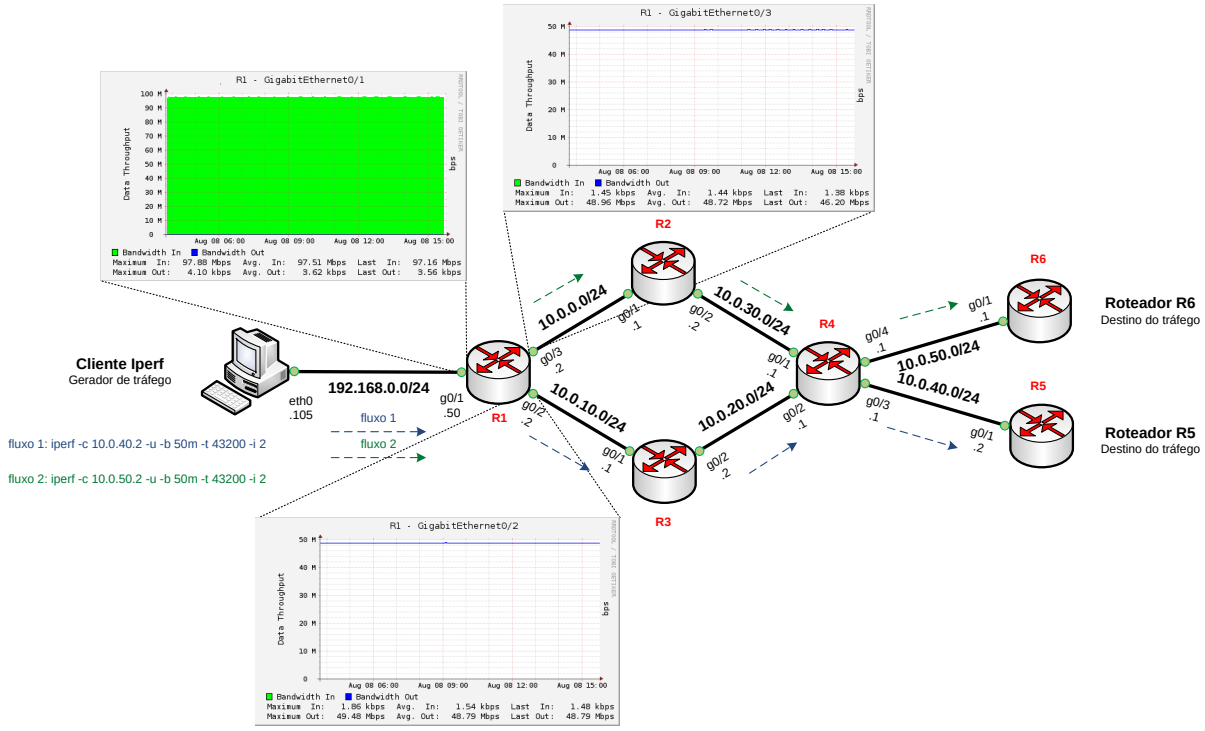

Figura 7. 4: Testes de balanceamento de carga com a ferramenta IPERF.

Em um sistema Linux, os parâmetros do sistema podem ser lidos ou alterados pela ferramenta *sysctl*, como em:

Leitura de todos os parâmetros: *#sysctl –a*

Ajuste de parâmetro: *#sysctl –w [parâmetro]=[valor]*

Os parâmetros responsáveis por alterar o tamanho dos *buffers* padrão para *sockets* UDP para transmissão e recepção são, respectivamente: *net.core.wmem_default* e *net.core.rmem_default*[30]. Ao ajustarmos esses parâmetros conseguimos aumentar ou diminuir o tamanho do *buffer* UDP, de acordo com as necessidades. Por exemplo, podemos ajustar ambos os *buffers* para 1 MB da seguinte forma:

root@iperf_client:~# sysctl –w net.core.wmem_default =1048576

root@iperf_server:~# sysctl –w net.core.rmem_default =1048576

Como valor inicial para ajuste do tamanho dos *buffers* o valor do BDP pode ser utilizado acrescentado de um valor adicional referente a um *overhead* específico do sistema operacional.

A Figura 7.5 exemplifica um caso testado em laboratório, em que dois roteadores modelos Cisco ME3400 E são conectados diretamente, sendo que o roteador R1 é conectado ao cliente IPERF e o R2 possui uma rota para o endereço 192.168.250.1/32 apontando para *Null0*. Nesse caso, o cliente gera tráfego UDP para o endereço 192.168.250.1 sem receber nenhuma resposta, servindo apenas para testarmos o enlace.

Assim sendo, para esse caso de um enlace a 1 Gbps e RTT de 1,691 ms, e considerando o *overhead* citado anteriormente, temos um tamanho de *buffer* de cerca de:

$$\text{Buffer UDP=BDP}\left(\frac{1.000.000.000\ bits}{1\ segundo}\times\frac{1\ byte}{8\ bits}\times\right.$$
$$\left.\times 0,001691\ \text{segundo} = 211375\ bytes\right)$$
$$+\text{overhead} = 307200\ bytes$$

	***Buffer Default*(110 KB)**	***Buffer* Alterado (300 KB)**
Cliente IPERF	**# iperf -c 192.168.250.1 -u -b 1000m -t 20 -i 5**	**# iperf -c 192.168.250.1 -u -b 1000m -t 20 -i 5**
Geração de Tráfego	**------------------------------------ Client connecting to 192.168.250.1, UDP port 5001 Sending 1470 byte datagrams UDP buffer size: 110 KByte (default) ------------------------------------ [3] local 192.168.200.2 port 39549 connected with 192.168.250.1 port 5001 [ID] Interval Transfer Bandwidth [3] 0.0- 5.0 sec 218 MBytes 365 Mbits/sec [3] 5.0-10.0 sec 220 MBytes 370 Mbits/sec [3] 10.0-15.0 sec 220 MBytes 369 Mbits/sec [3] 15.0-20.0 sec 198 MBytes 332 Mbits/sec [3] 0.0-20.0 sec 855 MBytes 359 Mbits/sec [3] Sent 610125 datagrams [3] WARNING: did not receive ack of last datagram after 10 tries.**	**------------------------------------ Client connecting to 192.168.250.1, UDP port 5001 Sending 1470 byte datagrams UDP buffer size: 300 KByte (default) ------------------------------------ [3] local 192.168.200.2 port 50355 connected with 192.168.250.1 port 5001 [ID] Interval Transfer Bandwidth [3] 0.0- 5.0 sec 337 MBytes 565 Mbits/sec [3] 5.0-10.0 sec 352 MBytes 591 Mbits/sec [3] 10.0-15.0 sec 358 MBytes 600 Mbits/sec [3] 0.0-20.0 sec 1.37 GBytes 587 Mbits/sec [3] Sent 998107 datagrams [3] WARNING: did not receive ack of last datagram after 10 tries.**

Tabela 7. 1: Alteração de buffer UDP para maior desempenho de transmissão.

Então, conforme a Tabela 7.1 demonstra, um aumento de aproximadamente 3 vezes o tamanho do *buffer* UDP *default* implicou em um aumento de 63,50% na taxa de transmissão UDP. Dessa maneira, podemos alterar o tamanho do *buffer* até encontrarmos a taxa de transmissão má-

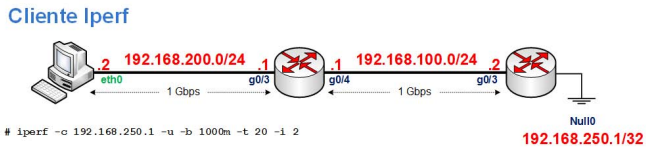

Figura 7. 5: Layout da rede de teste para geração de tráfego UDP de altas taxas.

xima UDP que conseguimos obter para um determinado enlace.

8. CONCLUSÃO

O presente trabalho visou demonstrar a utilização da ferramenta de geração e medição de tráfego IPERF. Primeiramente, foram apresentadas uma visão geral das ferramentas de medição ativa de redes e da ferramenta IPERF. Em seguida, foi apresentada uma breve introdução sobre os principais conceitos referentes aos protocolos UDP e TCP, primordiais para compreender o funcionamento da ferramenta. Em sequência, foram apresentados os procedimentos de instalação da ferramenta, os parâmetros principais de utilização e, por fim, alguns casos práticos em que pode ser utilizada.

Em suma, dentre os protocolos de transporte o TCP fragmenta as mensagens da aplicação em pacotes IP, podendo realizar a detecção e correção de erros, o controle de fluxo fim-a-fim (da origem ao destino), além de ser um protocolo orientado a conexões. O UDP é não orientado a conexão, não faz controle de fluxo nem de congestionamento e não garante que os pacotes sejam entregues ao destino. Basicamente, ele serve para transações que envolvem apenas mensagens curtas, rápidas e do tipo pergunta/resposta. Ou seja, é útil para situações em que a rapidez é mais importante do que a precisão. Apesar de servir, essencialmente, para transmissão de mensagens curtas, o que implica em baixa utilização de banda, muito se tem estudado para otimização da pilha de protocolos UDP/IP sobre redes de altas velocidades [31].

Em se tratando da ferramenta IPERF, os procedimentos de instalação apresentados na seção 5 demonstraram ser bastante simples e rápidos, tanto para plataformas Windows quanto Linux. O mesmo se aplica a outras plataformas como MAC e Solaris, não tendo sido apresentados nesse documento.

O IPERF se mostrou uma ferramenta simples, porém muito útil. Ela pode ser utilizada para testar o desempenho de conexões TCP e sessões UDP, além da perda de pacotes e *jitter* através do modo UDP. Provê informações básicas necessárias para a solução de problemas relacionados aos protocolos UDP e TCP.

Como a banda disponível pode ser influenciada por diversos fatores, como os citados a seguir, o IPERF demonstra ser mais útil para testes em ambientes fora de produção:

1. os horários do dia e dias da semana em que as medições são realizadas;

2. carga muito alta de utilização de CPU, o que pode impedir que a aplicação e a pilha de protocolos de rede obtenham tempo suficiente de CPU para processar os pacotes;

3. rotas em uso no momento das medições. Em alguns momentos, diferentes caminhos na rede podem ser tomados, implicando em atrasos diferentes, podendo gerar entregas fora de ordem e, consequentemente, provocando retransmissões no caso do TCP

Devido a esses e outros fatores, tais como: rotas assimétricas do cliente para o servidor e do servidor para o cliente; e a compressão de dados em função dos protocolos utilizados pela aplicação, os testes de largura de banda devem ser realizados de forma bidirecional. Isso permite verificar se a banda disponível é comparável em ambos os sentidos dependendo de que nó esteja sendo usado como servidor.

Além disso, vale ainda ressaltar que os testes UDP realizados utilizando somente o cliente IPERF é útil para casos em que somente se deseja saturar enlaces e se tem conhecimento da banda de cada enlace em todo o caminho. Para casos em que se deseja medir a banda útil, reportando a perda de pacotes e o *jitter,* é necessário a utilização do servidor IPERF, sendo possível assim notar congestionamentos na rede.

Contudo, o IPERF ainda apresentou como principais características:

1. Simplicidade de uso;

2. Eficácia no que se propõe, que é gerar e medir o tráfego;

3. Permissão de utilizar dados reais para simular a transferência confiável de dados, simulando o envio através de um protocolo de transferência confiável de dados como o FTP;

4. Auxílio na compreensão de como a alteração dos parâmetros do TCP, como a janela TCP e a utilização do algoritmo de Nagle, podem influenciar na utilização da largura de banda disponível;

5. Auxílio a administradores de rede a dimensionar as aplicações que fazem uso da rede, como por exemplo a vazão de servidores.

6. Possibilidade de gerar tráfego UDP de altas taxas, sem a necessidade de um servidor para receber o tráfego, simplificando os testes na rede.

Entretanto, apesar das características positivas que a ferramenta apresenta, ela ainda carece de algumas melhorias para o seu aprimoramento. Dentre elas, podemos citar as que foram consideradas como mais pertinentes:

1. Medição do atraso de ida e volta dos pacotes diretamente pela aplicação IPERF, para auxiliar no cálculo do BDP.

2. Método de detecção da largura de banda de cada um dos enlaces no caminho dos pacotes na rede, a fim de detectar gargalos fim a fim, para auxiliar na detecção de congestionamentos na rede.

3. Método de detecção do modo Duplex (*Half* ou *Full*) de cada um dos enlaces fim a fim.

Cada uma dessas considerações tornam-se relevantes para melhorar a análise e desempenho de redes através da ferramenta IPERF.

Em síntese, pode-se afirmar que o IPERF torna simples medir o desempenho de aplicações baseadas em fluxos TCP e UDP, apesar de não ser possível simular e testar o desempenho de todo tipo de aplicação, como é o caso de aplicações Web interativas.

---

[1] KISSEL, E. et al. Efficient wide area data transfer protocols for 100 Gbps networks and beyond. Proceedings of the Third International Workshop on Network-Aware Data Management. Denver, Colorado: ACM: 1-10 p. 2013.2

[2] GARZOGLIO, G. et al. Big Data Over a 100G Network at Fermilab. Journal of Physics: Conference Series, v. 513, n. 6, p. 7, 2014. ISSN 1742-6596. Disponível em: <http://stacks.iop.org/1742-6596/513/i=6/a=062017>.

[3] PINTO, J. D. O. et al. DWDM em Redes Metropolitanas. Nota Técnica do CBPF – NT001/02. Rio de Janeiro, Brasil 2002.

[4] ESTEVES, A. M. B. Sistema de monitoramento de redes baseado nos protocolos SNMP e SpanningTree 2013. (Dissertação de Mestrado em Física - Instrumentação Científica). Centro Brasileiro de Pesquisas Físicas, Rio de Janeiro, RJ, Brasil.

[5] MIRANDA, E. F. Desenvolvimento de Sitema para Monitoramento de Redes de Computadores e Servidor Looking Glass 2013. (Dissertação de Mestrado em Física - Instrumentação Científica). Centro Brasileiro de Pesquisas Físicas, Rio de Janeiro, RJ, Brasil.

[6] MIRANDA, E. F.; ALVES JR., N.; SOUZA, M. G. M. Desenvolvimento de um Repositório RIB-BGP. Notas Técnicas do CBPF, Rio de Janeiro, RJ, Brasil, v. 3, n. 2, p. 6, 2013. ISSN 0101-9201. Disponível em: <http://cbpfindex.cbpf.br/publication_pdfs/nt00413apub.2013_08_02_09_04_39.pdf>.

[7] ALVES JR., N. et al. Topologia e Modelagem Relacional da Internet Brasileira. Nota Técnica do CBPF - NT-004/4. Rio de Janeiro, RJ, Brasil 2004.

[8] ALVES JR., N. Caracterização de redes complexas aplicação à modelagem relacional entre Sistemas Autônomos da Internet 2007. (Tese de Doutorado em Modelagem Computacional). IPRJ, Universidade do Estado do Rio de Janeiro, Nova Friburgo, RJ, Brasil.

[9] CACTI GROUP INC. Cacti - The complete RRDTool-based graphing solution. 2014. Disponível em: <http://www.cacti.net/>. Acesso em: 27 de julho de 2014.

[10] OETIKER, T. MRTG - Tobi Oetiker's MRTG - The Multi Router Traffic Grapher. 2011. Disponível em: <http://oss.oetiker.ch/mrtg/>. Acesso em: 28 de julho de 2014.

[11] LABIT, Y.; OWEZARSKI, P.; LARRIEU, N. Evaluation of Active Measurement Tools for Bandwidth Estimation in Real Environment. Third IEEE/IFIP Workshop on End-to-End Monitoring Techniques and Services, E2EMON. Nice, França: IEEE: 71 - 85 p. 2005.

[12] DOWNEY, A. B. Clink: a tool for estimating Internet link characteristics. 1999. Disponível em: <http://allendowney.com/research/clink/>. Acesso em: 28 de julho de 2014.

[13] FRENCH FORUM FOR IPERF. Iperf. 2014. Disponível em: < https://iperf.fr/ >. Acesso em: 28 de julho de 2014.

[14] JONES, R. Netperf Homepage. 2014. Disponível em: <http://www.netperf.org/netperf/NetperfPage.html>. Acesso em: 28 de julho de 2014.

[15] DOVROLIS, C. Pathrate - A measurement tool for the capacity of network paths. 2006. Disponível em: <http://www.cc.gatech.edu/~dovrolis/bw-est/pathrate.html>. Acesso em: 28 de julho de 2014.

[16] NUTTCP DEVELOPMENT TEAM. NUTTCP Welcome Page. 2014. Disponível em: <http://www.nuttcp.net/nuttcp/Welcome%20Page.html>. Acesso em: 28 de julho de 2014.

[17] APPNETA. PathView - Network health monitoring over any network. 2014. Disponível em: <http://www.appneta.com/products/pathview/>. Acesso em: 28 de julho de 2014.

[18] NCSA NEWS REPORT TEAM. NLANR DAST Team Releases New Software. NCSA NEWS, 2001. Disponível em: <http://access.ncsa.illinois.edu/Briefs/01Briefs/010508.NLANR.html>. Acesso em: 12 de maio de 2014.

[19] POSTEL, J. RFC 768 - User Datagram Protocol. Internet Engineering Task Force (IETF), p.3. 1980.

[20] COMMER, D. E. Redes de Computadores e Internet. 4ª edição. ed. Porto Alegre: Artmed/Bookman, 2007.

[21] POSTEL, J. RFC 793 - Transmission Control Protocol. Internet Engineering Task Force (IETF), p.85. 1981.

[22] REZENDE, J. F. D.; COSTA, L. H. M. K.; RUBINSTEIN, M. G. Avaliação Experimental e Simulação do Protocolo TCP em Redes de Alta Velocidade XXII Simpósio Brasileiro de Telecomunicações - SBrT'05 Campinas, SP, Brasil: 6 p. 2005.

[23] ALLMAN, M.; PAXSON, V.; BLANTON, E. RFC 5681 - TCP Congestion Control. Internet Engineering Task Force (IETF), p.18. 2009.

[24] CONSTANTINE, B. et al. RFC 6349 - Framework for TCP Throughput Testing. Internet Engineering Task Force (IETF), p.27. 2011.

[25] KUROSE, J. F.; ROSS, K. W. Redes de computadores e a Internet: uma abordagem top-down. 5ª edição. São Paulo: Addison Wesley, 2010.

[26] TANEMBAUM, A. S. Redes de Computadores. 3ª edição. São Paulo: Elsevier, 2003.

[27] CISCO SYSTEMS INC. Configuring EtherChannels. In: (Ed.). Cisco ME 3400 Switch Software Configuration Guide, Rel. 12.2(25)EX. San Jose, California, Estados Unidos, 2005. cap. 31, p.602 - 623.

[28] CISCO SYSTEMS INC. Configuring a Load-Balancing Scheme. In: (Ed.). IP Switching Cisco Express Forwarding Configuration Guide. San Jose, California, Estados Unidos, 2012. cap. 5, p.47 - 58.

[29] OETIKER, T. About RRDtool - Logging and graphing 2013. Disponível em: <http://oss.oetiker.ch/rrdtool/>. Acesso em: 12 de maio de 2014.

[30] PITTSBURGH SUPERCOMPUTING CENTER. Enabling High Performance Data Transfers. 2014. Disponível em: <http://www.psc.edu/index.php/networking/641-tcp-tune#detailed>. Acesso em: 12 de maio de 2014.

[31] JIN, H.-W.; YOO, C. Impact of protocol overheads on network throughput over high-speed interconnects: measurement, analysis, and improvement. J. Supercomput., v. 41, n. 1, p. 17-40, 2007. ISSN 0920-8542.