UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
DEPARTAMENTO ACADÊMICO DE INFORMÁTICA
ESPECIALIZAÇÃO EM REDES DE COMPUTADORES

FABIO LEANDRO JANISZEVSKI

ROHC: ESTUDO COMPARATIVO DA APLICAÇÃO DE COMPRESSÃO DE CABEÇALHOS IP A VISUALIZAÇÃO DE VÍDEOS DO YOUTUBE

TRABALHO DE CONCLUSÃO DE CURSO

PATO BRANCO

2015

FABIO LEANDRO JANISZEVSKI

ROHC: ESTUDO COMPARATIVO DA APLICAÇÃO DE COMPRESSÃO DE CABEÇALHOS IP A VISUALIZAÇÃO DE VÍDEOS DO YOUTUBE

Trabalho de Conclusão de Curso, apresentado ao II Curso de Especialização em Redes de Computadores – Configuração e Gerenciamento de Servidores e Equipamentos de Redes, da Universidade Tecnológica Federal do Paraná, câmpus Pato Branco, como requisito parcial para obtenção do título de Especialista.

Orientador: Prof. Dr. Fabio Favarim

PATO BRANCO

2015

TERMO DE APROVAÇÃO

ROHC: Estudo Comparativo da Aplicação de Compressão de Cabeçalhos IP a Visualização de Vídeos do Youtube

por

Fabio Leandro Janiszevski

Esta monografia foi apresentada às 09h00min do dia 27 de outubro de 2015, como requisito parcial para obtenção do título de ESPECIALISTA, no II Curso de Especialização em Redes de Computadores – Configuração e Gerenciamento de Servidores e Equipamentos de Redes, da Universidade Tecnológica Federal do Paraná, Câmpus Pato Branco. O acadêmico foi arguido pela Banca Examinadora composta pelos professores abaixo assinados. Após deliberação, a Banca Examinadora considerou o trabalho **aprovado**.

Banca Examinadora

Prof. Dr. Fábio Favarim
Orientador / UTFPR-PB

Prof. Dr. Marco Antonio de Castro Barbosa
UTFPR-PB

Prof. Dr. Éden Ricardo Dosciatti
UTFPR-PB

Prof. Dr. Fábio Favarim
Coordenador do II Curso de Especialização
em Redes de Computadores

Dedico este trabalho a Deus, meus pais, e a todas as pessoas que não posso mais compartilhar da alegria de cada conquista, especialmente a minha "baba" Helena Nalevaiko(*in memorian*), e ao meu tio Roberto Janiszewski(*in memorian*), por terem nos deixado nesse ano de 2015.

"Não existe partida para aqueles que permanecerão eternamente em nossos corações."

AGRADECIMENTOS

Agradeço a todos os envolvidos, mesmo que indiretamente, na produção e conclusão deste trabalho e apoio ao processo da especialização.

Agradeço a Deus por sempre me mostrar onde devemos procurar forças para cada dia continuar em frente.

Agradeço especialmente aos meus pais Mario Janiszevski, e Natalia Marta Nalevaico Janiszevski que sempre me orientaram como pessoa, ajudaram na contrução do meu caráter e me apoiaram e desafiaram a procurar sempre uma nova conquista.

Agradeço ao professor Dr. Fábio Favarim, por me orientar e principalmente me apoiar no desenvolvimento deste trabalho, e ao tornar esse curso possível por coordená-lo. E a Secretaria do curso por dar todo o apoio necessário.

Agradeço a Michele Pysklevitz por sempre manter todos os envolvidos do curso atualizados sobre suas obrigações.

Agradeço ao professor Msc. Hermano Pereira, que me inspirou a seguir a área de redes de computadores, e me ajudou na confecção deste trabalho.

I'm doing a (free) operating system (just a hobby, won't be big and professional like gnu) for 386(486) AT clones.

Linus Torvalds

RESUMO

JANISZEVSKI, Fabio Leandro. ROHC: ESTUDO COMPARATIVO DA APLICAÇÃO DE COMPRESSÃO DE CABEÇALHOS IP A VISUALIZAÇÃO DE VÍDEOS DO YOUTUBE. 2015. 35 f. Monografia de Trabalho de Conclusão de Curso (II Curso de Especialização em Redes de Computadores), Departamento Acadêmico de Informática, Universidade Tecnológica Federal do Paraná. Pato Branco, 2015.

Este trabalho apresenta um estudo da utilização de um compressor de cabeçalho IP em consumo a vídeos na Internet, com enfoque no sistema YouTube, analisando o impacto na largura de banda necessária. Um cenário proposto foi simulado utilizando virtualização de máquinas, e implementado o compressor ROHC na rede com o auxílio das ferramentas rohc-lib e iprohc. Durante os experimentos foi realizada a coleta de informações com a visualização de um vídeo do YouTube nas qualidades automática, 1080p e 144p, em que resultaram em diferentes ganhos conforme a quantidade de *bytes* necessários na comunicação.

Palavras-chave: Compressão de cabeçalhos IP, ROHC, Linux

ABSTRACT

JANISZEVSKI, Fabio Leandro. ROHC: COMPARATIVE STUDY OF IP HEADER COMPRESSION APPLICATION YOUTUBE VIDEOS DISPLAY. 2015. 35 f. Monografia de Trabalho de Conclusão de Curso (II Curso de Especialização em Redes de Computadores), Departamento Acadêmico de Informática, Universidade Tecnológica Federal do Paraná. Pato Branco, 2015.

This paper presents a study of the use of an IP header compressor in consumer videos on the Internet, focusing on the YouTube system, analyzing the impact on the bandwidth required. A proposed scenario was simulated using virtualization machines, and implemented the ROHC compressor in the network with the help of ROHC-lib and iprohc tools. The experiments were gathering information to display a YouTube video in automatic qualities, 1080P and 144P, which resulted in different gains as the amount of bytes required in communication.

Keywords: IP Header Compression, ROHC, Linux

LISTA DE SIGLAS

IP	*Internet Protocol*
ROHC	*RObust Header Compression*
CTCP	*IP Header Compression*
CRTP	*Compressing IP/UDP/RTP Headers for Low-Speed Serial Link*
RTP	*Real-time Transport Protocol*
TAROC	*TCP-Aware RObust Header Compression*
EPIC	*Efficient Protocol Independent Compression*
ROHC+	*Robust Header Compression+*
BER	*Bit error rate*
RTT	*Round Trip Time*
CID	*Context Identifier*
IR	*Initialization and Refresh*
FO	*First Order*
SO	*Second Order*
U-Mode	*Unidirectional mode*
O-Mode	*Bidirectional Optimistic mode*
R-Mode	*Bidirectional Reliable mode*
ACK	*Acknowledgement*
NACK	*Negative* ACK
NC	*No Context*
SC	*Static Context*
FC	*Full Context*
CRC	*Cyclic Redundancy Check*
PC	*Personal Computer*
VPN	*Virtual Private Network*

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

1.1 CONSIDERAÇÕES INICIAIS

Segundo dados estatísticos obtidos pelo eCommerceOrg (ECOMMERCEORG, 2015), que se baseiam em dados de diversos institutos, dentre eles o *Internet World Stats*, em 2000, no Brasil existiam aproximadamente 5 milhões de internautas. Já em pesquisa mais recente, de junho de 2012, o número foi acima de 88 milhões de internautas, demonstrando um crescimento de mais de 83 milhões de internautas, considerando a população nacional.

Na proporção mundial, segundo a pesquisa da *Internet Society* (REPORT, 2015a), em maio de 2015, existiam aproximadamente 3 bilhões de internautas em todo o mundo. Muitos destes internautas utilizam de aplicações multimídias, destacando especialmente o uso de vídeos, dentre estes vídeos curtos, *live stream* ou mesmo vídeo sob demanda. Desta forma, conforme a estimativa da Cisco (CISCO, 2015), até o fim de 2015, o consumo de vídeo *stream* na *Internet* representará 72.45% (27466 petabytes) do tráfego mundial, e em 2019 serão 89319 petabytes. Já na América Latina, o consumo de vídeo *stream* estará representado em 64.15%. Considerando somente o consumo de tráfego do sistema de vídeo *stream* na pesquisa da Sandvine (REPORT, 2015b), o YouTube em 2014 representou 28.94% de consumo de *downstream* e 14.90% de *upstream* na América Latina, versus 13.19% e 5.52% na América do Norte, respectivamente.

Neste contexto, a proposta do presente trabalho é realizar um teste de desempenho de um protocolo de compressão de dados IP (*Internet Protocol*) sobre o uso do sistema YouTube, a fim de garantir uma economia no uso de banda.

1.2 PROBLEMAS E PREMISSAS

A atual infra-estrutura de rede disponível para utilização no âmbito nacional é escassa, visto que, segundo a pesquisa da Anatel (2014), em setembro de 2014, a conexão com velocidade média mais alta disponível na banda larga fixa, é de apenas 20.22 Mbps,

considerando todo o Brasil. Porém, se for considerado somente o sul do país, o resultado é 17.55 Mbps. Considerando a avaliação do Google (GOOGLE, 2015), a classificação para reproduzir apenas um vídeo do YouTube em definição padrão, necessita uma conexão entre 0,7 Mpbs e 2,5 Mbps, e mais de 2,5 Mbps de conexão para reproduzir em alta definição sem atrasos. Na melhor das combinações, uma conexão de banda larga fixa no Brasil, não comportaria mais que 8 vídeos de alta definição simultaneamente sem ocorrer atrasos na reprodução.

Desta forma, implementar uma solução que possibilite a economia de consumo da banda, pode viabilizar o aproveitamento por parte dos internautas, sem ocorrência de atrasos na reprodução de vídeo *stream*, especificamente na utilização do sistema YouTube.

1.3 OBJETIVOS

1.3.1 OBJETIVO GERAL

Testar o ROHC (*RObust Header Compression*) em dois roteadores virtualizados, a fim de medir o consumo de banda entre eles, com a rede consumindo vídeos do YouTube.

1.3.2 OBJETIVOS ESPECÍFICOS

- Compilar o ROHC no *kernel* de dois roteadores com o sistema operacional Debian Wheezy.
- Comparar o consumo de banda com a utilização do ROHC e sem a sua utilização.
- Tentar reduzir o consumo de banda da *Internet* para o tráfego de vídeo;

1.4 ORGANIZAÇÃO DO DOCUMENTO

O Capítulo 2 descreve uma introdução de compressores de cabeçalhos, e o funcionamento do ROHC.

O Capítulo 3 descreve os métodos utilizados.

O Capítulo 4 descreve os resultados.

O Capítulo 5 descreve as conclusões e os trabalhos futuros.

2 REFERENCIAL TEÓRICO

2.1 COMPRESSÃO DE CABEÇALHOS

Os problemas de desempenho do protocolo IP em conexões de baixa velocidade são estudados desde 1984, com o protocolo *Thinwire* especificado por (FARBER et al., 1984). Case (1990) propôs um mecanismo baseado na informação redundante dos cabeçalhos, o qual comprime os 40 *bytes* do cabeçalho TCP/IPv4 para algo entre 3 e 6 *bytes*. Existem também outras especificações para diferentes protocolos baseados nas informações redundantes dos cabeçalhos, como o CTCP (*IP Header Compression*) proposto por Degermark et al. (1999), o qual gerencia uma variedade alta de *stream* e também da atenção especial ao IPv6. O CRTP (*Compressing IP/UDP/RTP Headers for Low-Speed Serial Link*) desenvolvido por Casner e Jacobson (1999) é uma especificação detalhada para compressão do protocolo RTP (*Real-time Transport Protocol*).

Nos anos 2000, surgiram novas contribuições para compressão de cabeçalhos TCP/IP, sendo elas TAROC (*TCP-Aware RObust Header Compression*)(LIAO et al., 2000), EPIC (*Efficient Protocol Independent Compression*)(PRICE et al., 2001) e ROHC+ (*Robust Header Compression+*)(BOGGIA et al., 2002). O objetivo de TAROC é limitar a propagação de erros na *congestion window tracking* quando utilizado o protocolo TCP. EPIC utiliza uma variante da codificação de Huffman para produzir um conjunto de formatos de cabeçalhos comprimidos. ROHC+ propõe um novo perfil e um algoritmo para comprimir TCP *stream* com o ROHC *framework*. Essas contribuições são a base para o padrão de compressão de cabeçalhos IP ROHC definidos em (BORMANN et al., 2001).

2.2 O ROHC

O algoritmo ROHC foi desenvolvido com o intuito de reduzir o tamanho dos cabeçalhos dos pacotes IP com enfoque em enlaces que se caracterizam pelo alto BER (*Bit error rate*), longo RTT (*Round Trip Time*) e erros contínuos. Um exemplo desse tipo de enlace

é o sem fio. A Figura 1 demonstra o modelo de trabalho que o ROHC faz para comprimir os pacotes, no caso utilizando o perfil IP/UDP/RDP. Segundo Chen (2005), para a compressão o ROHC faz uma classificação dos campos de cada cabeçalho. Essa análise é baseada em como os valores de cada campo dos cabeçalhos se alteram durante o *stream*. Esses campos são separados e classificados como estáticos e dinâmicos, na cadeia dos pacotes de cabeçalhos comprimidos. A classificação estática se refere a informações que possam permanecer constantes, e a dinâmica se refere a informações que se alteram conforme o *stream*, porém cujo padrão de mudança possa ser conhecido.

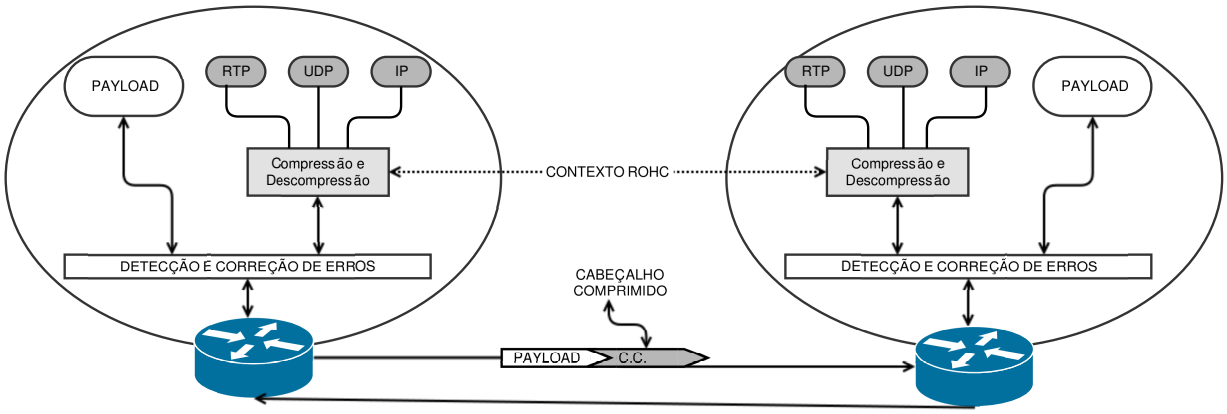

Figura 1: Modelo de trabalho do ROHC

Fonte: Acticom ()

O ROHC utiliza um contexto mantido entre o compressor e o descompressor para armazenar a informação sobre o *stream* dos cabeçalhos. Este contexto contém a última atualização correta do cabeçalho original e a informação redundante do *stream*, e a cada valor alterado dentro do contexto, o mesmo é atualizado. Se o contexto é perdido devido aos erros de transmissão, então não há a sincronização entre o compressor e o descompressor, e então o descompressor pode requisitar para atualizar o contexto com a utilização de ACK. Cada *stream* em um canal possui seu contexto, o qual é identificado pelo CID (*Context Identifier*), o qual basicamente é um número que diferenciam o *stream* dentro de um canal, e o contexto entre o compressor e o descompressor.

2.2.1 ESTADOS E NÍVEIS DE COMPRESSÃO

Existem 3 estados para compressão e outros 3 estados em que o ROHC pode ficar para descompressão. IR (*Initialization and Refresh*), FO (*First Order*) e SO (*Second Order*) são estados de compressão. Os estados correspondentes mantidos no descompressor são:

- *No Context State*: Não existe um contexto nesse estado;
- *Static Context State*: Estado que indica o contexto para descomprimir conteúdos estáticos;
- *Full Context State*: Estado que indica o contexto para descomprimir conteúdos estáticos e dinâmicos.

A Figura 2 apresenta os estados de compressão do ROHC. No estado IR, o compressor inicializa ou re-inicializa o contexto do descompressor quando necessário, enviando as informações de um cabeçalho completo com todos os campos não comprimidos e informações adicionais do ROHC. Quando o compressor entra no estado FO, partes estáticas do cabeçalho não serão mais enviadas, somente campos dinâmicos serão enviados para a correlata informação dinâmica do cabeçalho, armazenados no contexto do descompressor. E no estado SO, serão encaminhados pacotes com informações do último cabeçalho que não continha nenhuma informação estática e dinâmica. Sendo então o estado IR o nível mais baixo de compressão, porque os cabeçalhos não serão comprimidos, e o estado SO sendo o nível mais alto de compressão, por encaminhar o menor número possível de *bits* de informação no cabeçalho para a rede. Segundo Couvreur et al. (2006), no estado IR, o tamanho do cabeçalho fica entre 48 e 130 *bytes*, no estado FO o cabeçalho varia entre 3 e 84 *bytes*, e por fim, no estado SO, o tamanho fica em somente 1 *byte*. Estes diferentes tamanhos em cada estado depende da informação no cabeçalho que o descompressor requer.

2.2.2 MODOS DE OPERAÇÃO

Tanto no compressor como no descompressor, há três possíveis modos de operação, sendo U-Mode (*Unidirectional mode*), O-Mode (*Bidirectional Optimistic mode*) e R-Mode (*Bidirectional Reliable mode*). Estes três modos de operação definem como trabalhará os estados de compressão dentro do contexto do ROHC, como explanados nas figuras 3, 4 e 5. Inicialmente, o compressor opera em U-mode, e envia periodicamente atualizações para o descompressor. Quando começa a transmissão nesse modo, o compressor envia o cabeçalho com todas as informações (estáticas e dinâmicas) para o descompressor, e repete-se até que o compressor esteja confiante que o descompressor tenha recebido pelo menos um pacote com todas as informações. Segundo Bormann et al. (2001), esse número inicial de pacotes não é fixo. O principal objetivo deste modo é manter o compressor e o descompressor sincronizados. Após a primeira compressão, o ROHC pode convergir para um dos outros dois modos, e assim o descompressor enviará um *feedback*, fazendo com que o ROHC pule entre os modos de operação.

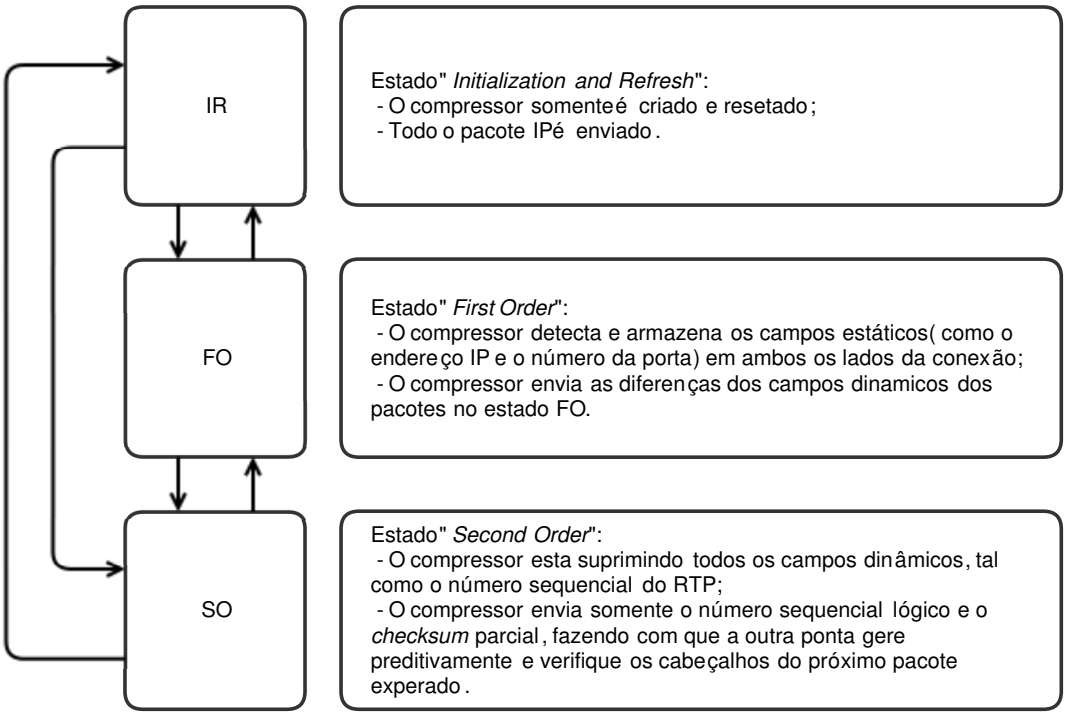

Figura 2: Estados de compressão do ROHC

Fonte: ShareTechNote ()

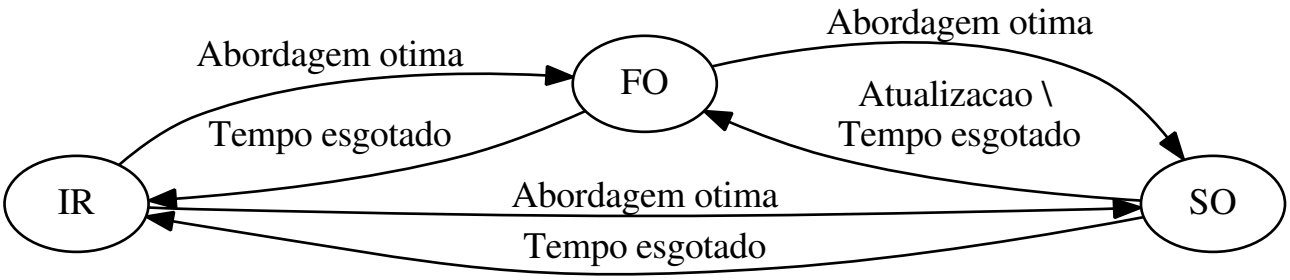

Figura 3: Estados de compressão em U-mode

Fonte: Iqbal (2013)

O *feedback* além de indicar o modo que o compressor vai trabalhar, ele também retorna uma mensagem de confirmação sobre o estado de descompressão, sendo possível uma das 3 mensagens:

- ACK (*Acknowledgement*): Reconhece e indica uma descompressão com sucesso;
- NACK (*Negative* ACK): Indica que o contexto dinâmico ficou inválido no lado do descompressor;

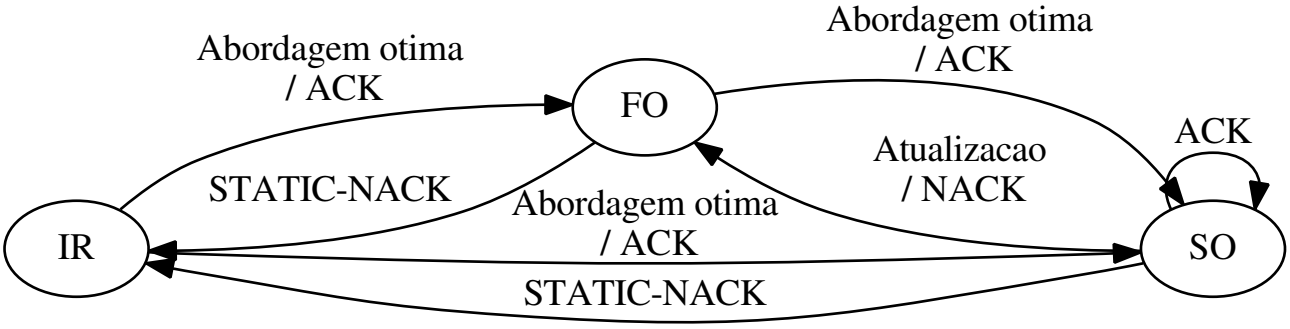

Figura 4: Estados de compressão em O-mode

Fonte: Iqbal (2013)

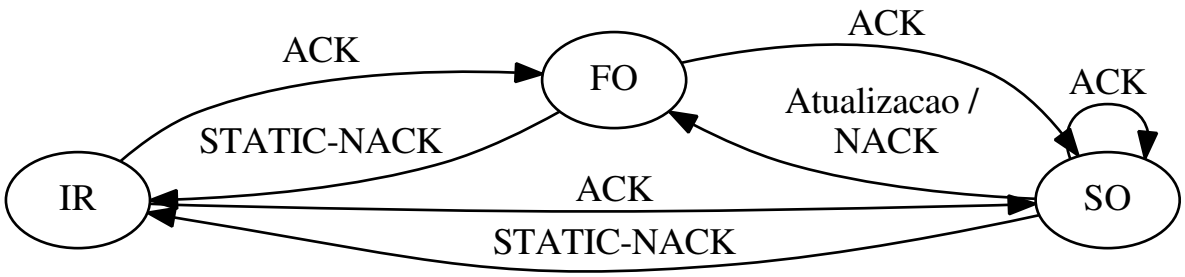

Figura 5: Estados de compressão em R-Mode

Fonte: Iqbal (2013)

- STATIC-NACK: Indica que todo o contexto estático estava danificado na hora de descomprimir.

No O-mode, não são enviadas as atualizações periódicas para o descompressor, com isso tendo um ganho de compressão em relação ao U-mode. Neste modo, o compressor encaminha pacotes com cabeçalhos completos, esperando que o descompressor encaminhe um ACK, indicando que o contexto foi criado, e está pronto para comprimir os cabeçalhos dos pacotes. Diferentemente do U-mode, o O-mode possui um canal de *feedback* ativo, fazendo com que seja possível diminuir a perda de pacotes. O R-Mode é um modo mais robusto que o O-mode, pois fornece um nível de confiança, fazendo com que o compressor encaminhe atualizações periodicamente até que receba um ACK do descompressor. Ele também possui um canal de *feedback* para recuperação de erros na requisição. Ao invés de utilizar uma "abordagem ótima" como no O-mode, o R-Mode atualiza o nível de compressão baseado no ACK de cada pacote que atualiza o contexto, mas nem todo o pacote atualiza o contexto no R-Mode (IQBAL, 2013).

2.2.3 PERFIS DE COMPRESSÃO

Os perfis são utilizados para definir os diferentes tipos *streams* de cabeçalhos. O descompressor pode definir o tipo de *stream* analisando o perfil de compressão. A RFC3095 (BORMANN et al., 2001) propõem 4 perfis de compressão e descompressão, IP/UDP/RTP, IP/UDP, IP/ESP e "Sem compressão", e essa mesma RFC considera as implementações para IPv4 e IPv6. A RFC4019 (PELLETIER, 2005), propõe os perfis IP/UDP-Lite e IP/UDP-Lite/RTP, e a RFC4996 (PELLETIER et al., 2007) propõe o perfil IP/TCP. O primeiro perfil é o "Sem compressão", e quando está em uso somente o identificador do contexto ROHC é adicionado em cada pacote, para o descompressor saber que este *stream* não é comprimido. Cada perfil abaixo relaciona quais cabeçalhos de protocolos serão comprimidos, sendo eles:

- Sem compressão;
- Somente IP;
- IP/UDP/RTP;
- IP/UDP;
- IP/UDP-Lite;
- IP/UDP-Lite/RTP;
- IP/ESP;
- IP/TCP.

2.2.4 DESCOMPRESSÃO

Como dito na Seção 2.2.1, existem estados de descompressão que correspondem a compressão. Esses estados são responsáveis por determinar que a compressão e descompressão resultaram em um pacote consistente, para que se evite a perda de informações. Todos os estados são baseados no contexto do ROHC, e esses estados como já explicados são NC (*No Context*), SC (*Static Context*) e FC (*Full Context*). Couvreur (2006) explica que no primeiro estado, o NC, ele permanece enquanto nenhum contexto entre o compressor e o descompressor é criado, e retorna para esse estado quando o contexto é perdido. Enquanto o descompressor permanece nesse estado, somente os pacotes com estados IR do compressor são descomprimidos, qualquer outro pacote com um estado diferente vindo do compressor é descartado. O descompressor altera para o estado FC assim que o contexto é criado, ou

quando um pacote é descomprimido com sucesso, o qual pode ser verificado pelo CRC (*Cyclic Redundancy Check*). O estado SC somente é atingido quando uma parte do campo dinâmico é perdida. O descompressor então aguarda um tempo para exceder no modo U-Mode, e em outros modos, é enviado um NACK para o compressor, a fim de atingir o nível FO de compressão. O descompressor utiliza uma regra de falha denominada "k out of n", onde k é o número de pacotes recebidos com um erro nos últimos n pacotes transmitidos. Esta regra é utilizada para determinar se o contexto está corrompido, e retroceder entre os estados após enviar um NACK para o compressor, quando está trabalhando no modo O-Mode ou R-Mode. Essa alternância entre os estados pode ser visualizada na Figura 6.

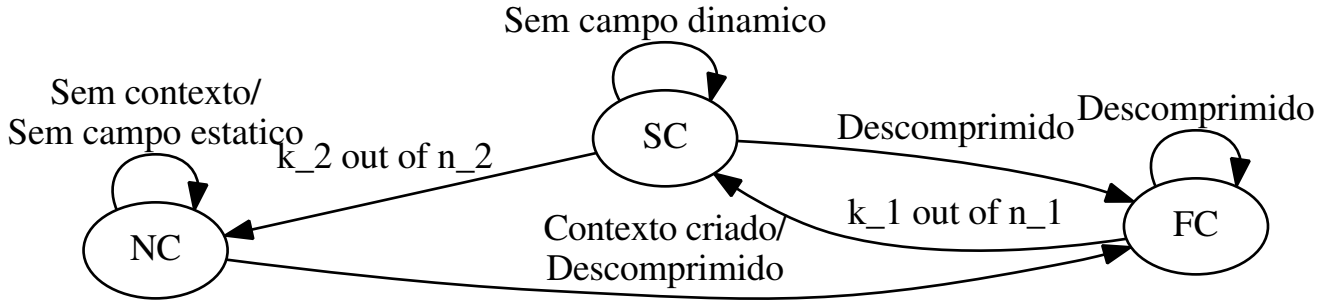

Figura 6: Estados de descompressão

Fonte: Iqbal (2013)

3 MATERIAIS E MÉTODOS

Neste capítulo é descrita a metodologia utilizada para o desenvolvimento deste trabalho. São descritas as etapas do projeto e os principais fundamentos e tecnologias a serem empregados.

3.1 CENÁRIO PROPOSTO

Para os experimentos, foi simulada uma topologia de rede em máquinas virtuais. Esse cenário possui dois roteadores e um *host*. O primeiro roteador é conectado a Internet, compartilhando-a com o segundo roteador, e o qual compartilha a Internet com o *host*. No enlace entre o roteador 1 e 2 foi ativado o uso do ROHC, e também coletado as amostras dos experimentos. Essa topologia do cenário pode ser visualizada na Figura 7.

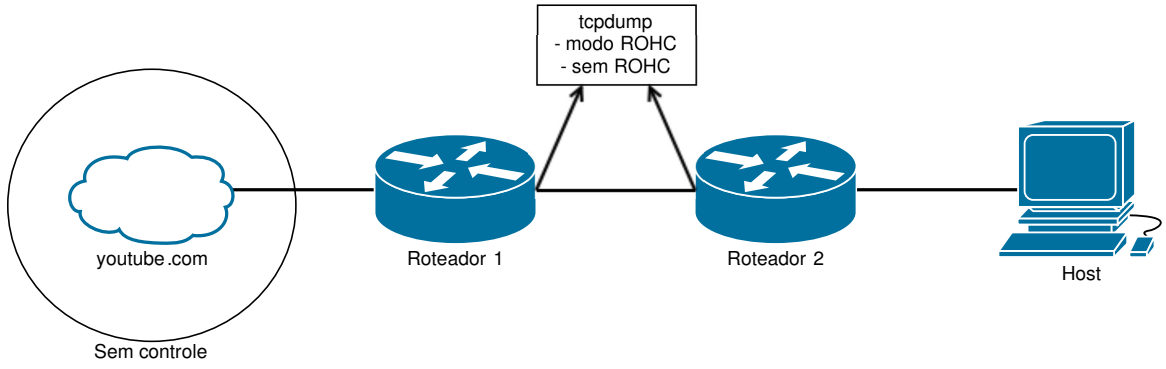

Figura 7: Diagrama de rede utilizado

Fonte: Elaborado pelo autor

3.2 MATERIAIS

Os materiais necessários se dividiram entre o hardware e software. O hardware, um PC (*Personal Computer*) possui um processador AMD PhenomII X6 com 8GB de memória RAM,

e acesso a Internet com largura de banda de 10Mbit/s.

Os Softwares necessários foram:

- VirtualBox versão 5.0;
- Sistema Operacional Debian/GNU *whezzy*;
- rohc-lib versão 1.7.1;
- iprohc versão 0.7.1;
- tcpdump versão 4.5.1 para captura de pacotes;
- Wireshark versão 1.10.6 para análise dos pacotes.

3.3 MÉTODO

Foram criadas três máquinas virtuais utilizando o VirtualBox, em que duas delas simularam roteadores com estados de ROHC ativo e inativo, e uma terceira máquina virtual simulou o *host* que fez acesso ao sítio YouTube. Nos roteadores, foi instalado o sistema operacional Debian, e implementado o roch-lib e o iprohc (CNES et al., 2015). Para cada vídeo que era carregado, o tcpdump era ativado nas "bordas" de conexão entre os dois roteadores.

Foram analisados os resultados do consumo de banda, com o arquivo *.pcap* extraído com o tcpdump entre os dois roteadores virtualizados, na ferramenta Wireshark. Desta maneira, pode ser feita comparações do impacto da utilização do protocolo ROHC, no consumo de acesso ao sítio YouTube.

As capturas se deram em versões 1, 2 e 3, as quais definem diferentes tipos de qualidade de vídeo. A versão 1 foi a versão com qualidade automática, a versão 2 em qualidade 1080p, e a versão 3 em qualidade de 144p. A fim de evitar comparações com diferentes dados de vídeo, e obter um resultado mais preciso na compressão, para cada versão, foi fixada a utilização de um mesmo vídeo, o qual nos experimentos é intitulado "AMIGUINHO" (FUNDOS, 2015).

4 RESULTADOS

Foram capturados e analisados três modos (Vide descrição no Capítulo 3) de um vídeo no YouTube, cada modo em uma qualidade diferente. Essas capturas foram realizadas em dois roteadores diferentes, e com o ROHC ativo e inativo, totalizando 12 amostras de captura com o tcdump, no formato *.pcap*. As amostras foram classificadas em 3 conjuntos, cada qual define a qualidade de vídeo exibida, sendo o primeiro, V1, em modo automático, o segundo, V2, em qualidade 1080p, e o terceiro conjunto de amostras, denominado como V3, em 144p. Todas essas amostras foram extraídas no mesmo formato, ativando o tcpdump, carregando o vídeo no YouTube, assistí-lo por completo, e parando a captura em ambos os roteadores. Apesar da captura não ter sido iniciada e parada ao mesmo instante em ambos os roteadores, foi diferenciada a captura total de *bytes* do vídeo. Durante todo o experimento e coleta das 12 amostras, o vídeo foi reproduzido conforme um internauta teria acesso, sem nenhum tipo de atraso no vídeo ou áudio durante a reprodução. Pelo fato de os experimentos terem como objetivo a compressão de vídeo do YouTube para o *host*, na configuração do iproch a opção "unidirecional" foi ativada (Detalhes podem ser encontrados no arquivo de configuração do iprohc, vide Anexo B), logo o único contexto que o ROHC trabalhou é o U-mode, e então a compressão teve o comportamento visualizado na Figura 3, na Seção 2.2.2.

4.1 MÉTRICAS DE DESEMPENHO

Como o objetivo principal do trabalho é a economia de largura de banda, as métricas definidas para analisar o desempenho do uso ou não do ROHC foram:

- Quantidade de *bytes* trafegados: Quantia de *bytes* capturados no momento em que o vídeo foi visualizado.
- Largura de banda: Determina a medida de vazão de dados pela rede.

4.2 QUALIDADE AUTOMÁTICA - V1

A Figura 8 apresenta o resultado obtido com a captura do vídeo em qualidade automática, executado em 2 etapas. A primeira etapa foi com os roteadores *Router 01* e *Router 02* sem o módulo ROHC, e a segunda etapa com o módulo ROHC ativo nestes mesmos roteadores, denominado no gráfico com o sufixo "ROHC" nos rótulos. A qualidade automática é uma denominação do YouTube, a qual determina uma qualidade de vídeo conforme a largura de banda do internauta, não podendo determinar ao certo qual foi a qualidade estabelecida durante o experimento.

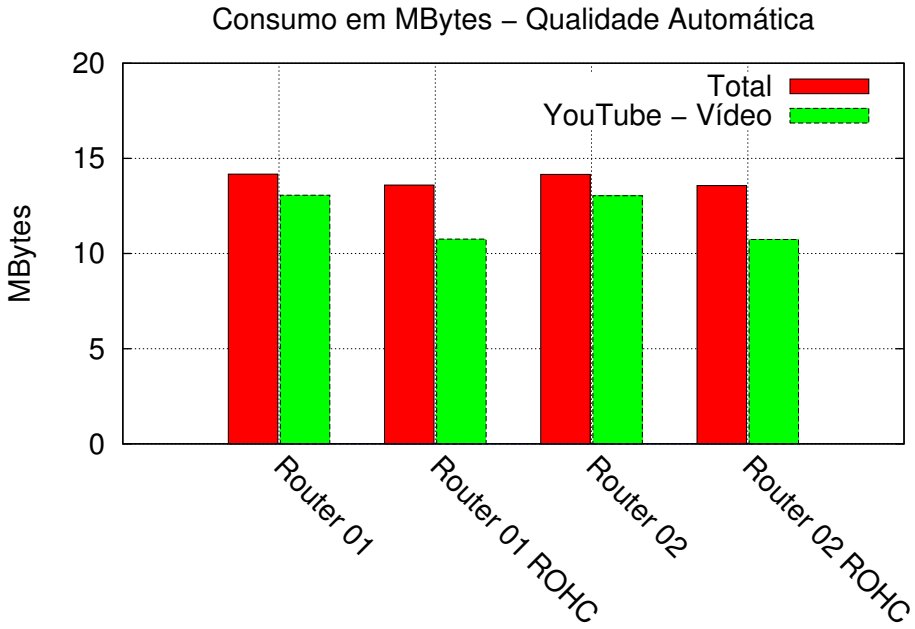

Figura 8: Compressão de vídeo em qualidade automática

Fonte: Elaborado pelo autor

Os resultados indicam a quantidade de *Megabytes* gerados durante a captura ao visualizar o vídeo na qualidade automática. A diferença entre o "total" e o "vídeo" se deve ao fato de que, quando é carregada a página do YouTube, muitos elementos surgiram na tela, não somente o vídeo em si, a exemplo disso foram miniaturas de outros vídeos relacionados, propagandas, comentários sobre o vídeo, e afins.

Uma melhor análise pode ser verificada na Tabela 1, em que é possível visualizar que houve aproximadamente 4% de diferença entre a utilização do ROHC ou sem o seu uso, na quantidade de dados gerados. Porém, saltando para mais de 17% essa diferença se somente comparado o vídeo. Essa diferença de porcentagem, um ganho menor no total, pode ser

explicado pelo motivo em que, no momento da captura sem o ROHC, houve um menor número de elementos extras como já citado.

Tabela 1: Quantidade de *Megabytes* gerados com vídeo na qualidade automática

	ROHC Inativo	ROHC Ativo	Diferença
Router 01 - Total	14,18MB	13,60MB	4,09 %
Router 02 - Total	14,16MB	13,57MB	4,16 %
Router 01 - Vídeo	13,06MB	10,76MB	17,61 %
Router 02 - Vídeo	13,04MB	10,74MB	17,64 %

Fonte: Elaborado pelo autor

É possível deduzir pela análise da Tabela 1 que pode ter havido uma economia de 4% de largura de banda, pelo fato de diminuir proporcionalmente nessa mesma quantidade os *MegaBytes* gerados.

4.3 QUALIDADE 1080P - V2

Conforme já explicado na Seção 4.2, a coleta de amostras para a V2, qualidade 1080p, foram executadas nas mesmas etapas descritas. Analisando a Figura 9, é possível notar uma maior margem de diferença entre as amostras com e sem ROHC, em ambos os roteadores, essa margem pelo gráfico nota-se que é acima de 10MB.

Pode-se visualizar uma diferença de 24,49MB para vídeo na Tabela 2 no *Router 01*, totalizando 28,91% de economia no tráfego de dados em relação ao vídeo, e 26,41% no tráfego total. O objetivo deste experimento era demonstrar se o ROHC se comportaria diferente para uma taxa constante de qualidade de vídeo, e no máximo de capacidade de qualidade, o qual teve um resultado significativo, aumentando em 10% a diferença em uso do tráfego de dados.

Tabela 2: Quantia de *Megabytes* gerados com vídeo na qualidade 1080p

	ROHC Inativo	ROHC Ativo	Diferença
Router 01 - Total	86,00MB	63,29MB	26,41 %
Router 02 - Total	84,01MB	63,14MB	24,84 %
Router 01 - Vídeo	84,72MB	60,23MB	28,91 %
Router 02 - Vídeo	82,72MB	60,09MB	17,64 %

Fonte: Elaborado pelo autor

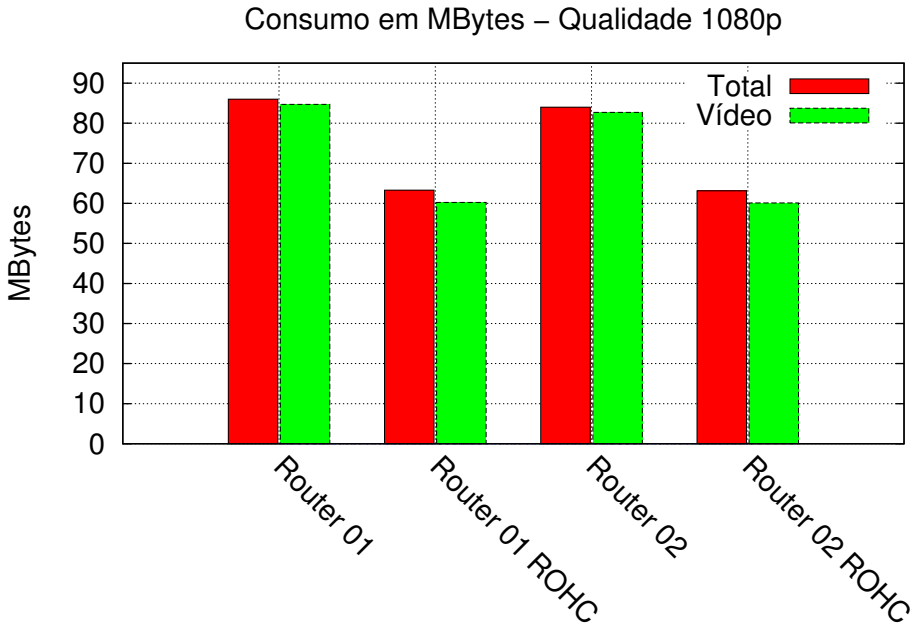

Figura 9: Compressão de vídeo em qualidade 1080p

Fonte: Elaborado pelo autor

4.4 QUALIDADE 144P - V3

Seguindo as explicações de coletas anteriores, as amostras com 144p demonstraram ter um ganho na economia no consumo de banda ao utilizar o ROHC. Pode-se visualizar na Figura 10 que a margem de compressão do vídeo foi menor que a compressão total de *MegaBytes*, mas também como observado anteriormente, a margem pode ser devido a elementos extras ao acessar o vídeo.

Analisando a Tabela 3, a porcentagem de compressão baixou em relação as outras amostras, mas também é a menor quantidade de dados trafegados. Nota-se que a diferença entre o uso do ROHC é mais eficiente conforme a qualidade de vídeo selecionada.

Tabela 3: Quantia de *Megabytes* gerados com vídeo na qualidade 144p

	ROHC Inativo	ROHC Ativo	Diferença
Router 01 - Total	9,04MB	6,80MB	24,78 %
Router 02 - Total	8,27MB	6,78MB	18,02 %
Router 01 - Vídeo	7,42MB	6,30MB	15,09 %
Router 02 - Vídeo	6,90MB	6,29MB	8,84 %

Fonte: Elaborado pelo autor

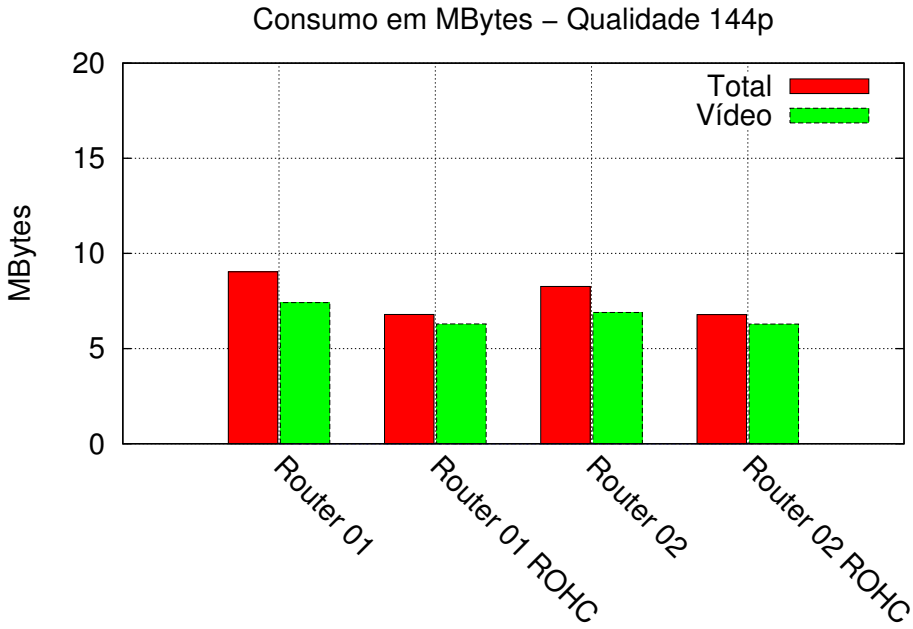

Figura 10: Compressão de vídeo em qualidade 144p

Fonte: Elaborado pelo autor

4.5 UTILIZAÇÃO DE LARGURA DE BANDA

Nesta seção é analisado o acumulado da velocidade de banda utilizada com os experimentos, e o resultado comparativo do uso do ROHC. Conforme a Figura 11, nota-se uma queda de quase 1Mbit/s de consumo da banda para a visualização do vídeo nas três diferentes qualidades coletadas. A maior diferença encontra-se na coleta da amostra “V2”.

Na Tabela 4 é possível notar os ganhos de consumo para “V1” e “V3”. Em ambos os casos, o ganho é abaixo de 0,050 Mbit/s, mas ao acumular, a economia fica em 0,843Mbit/s no *Router 01*, que trabalhou como compressor durante a comunicação, e 0,799Mbit/s no *Router* 02, o qual trabalhou como descompressor.

Tabela 4: Uso da largura de banda

Router	V1	V2	V3	Acumulado
Router 01	0,460 Mbit/s	2,807 Mbit/s	0,292 Mbit/s	3,559 Mbit/s
Router 01 - ROHC	0,414 Mbit/s	2,070 Mbit/s	0,232 Mbit/s	2,716 Mbit/s
Router 01 - Diferença	0,046 Mbit/s	0,737 Mbit/s	0,060 Mbit/s	0,843 Mbit/s
Router 02	0,455 Mbit/s	2,804 Mbit/s	0,269 Mbit/s	3,528 Mbit/s
Router 02 - ROHC	0,433 Mbit/s	2,065 Mbit/s	0,231 Mbit/s	2,729 Mbit/s
Router 02 - Diferença	0,022 Mbit/s	0,739 Mbit/s	0,038 Mbit/s	0,799 Mbit/s

Fonte: Elaborado pelo autor

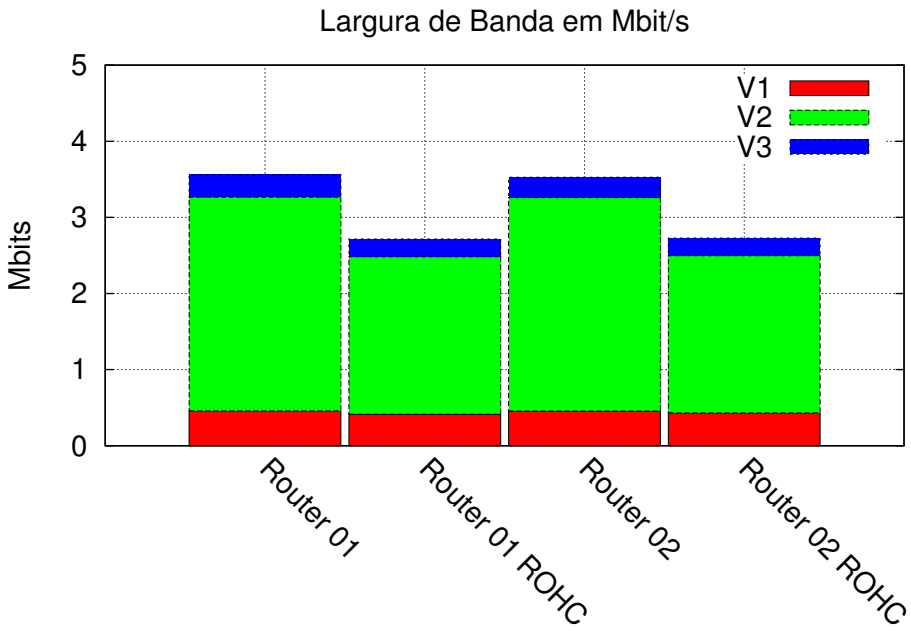

Figura 11: Largura de banda

Fonte: Elaborado pelo autor

5 CONCLUSÕES E TRABALHOS FUTUROS

As análises das amostras coletadas neste trabalho confirmam o impacto positivo na economia de banda ao utilizar o ROHC. A quantidade de dados, e o tipo de dados, possuem relevância para a compressão de cabeçalhos. Uma forma "ideal" de compressão seria a compressão e ganhos de economia constante para qualquer caso de uso. A economia total variando entre 4,09% e 28,91%, no pior e melhor caso respectivamente, demonstram o impacto do simples uso do ROHC para as redes que consomem o sítio YouTube. Entre as amostras analisadas, pode-se observar que conforme a quantia de dados aumenta, o ROHC desempenha melhor sua função para a economia no consumo de banda.

Para futuras pesquisas, sugere-se que sejam utilizadas abordagens de compressão híbrida, ou seja, em que o *payload* também seja comprimido. Também é sugerido criar túneis de conexão para que a informação comprimida seja enviada através de vários nós na rede, e a análise de desempenho do ROHC sobre um enlace VPN (*Virtual Private Network*). Aos que desejam reproduzir a implementação, o Anexo A possui um guia passo-a-passo de como foi compilado e instalado o rohc-lib utilizado nos experimentos.

REFERÊNCIAS

ACTICOM. **Robust Header Compression (RoHC + RoHCv2)**. Disponível em: <http://www.acticom.de/robust-header-compression-2/>. Acesso em: 1 out. 2015.

ANATEL. **Qualidade da Banda Larga**. [S.l.], dez 2014.

BOGGIA, G.; CAMARDA, P.; SQUEO, V. Rohc+: A new header compression scheme for tcp streams in 3g wireless systems. **ICC 2002**, v. 5, p. 3271–3278, 2002.

BORMANN, C. et al. **RObust Header Compression (ROHC): Framework and four profiles: RTP, UDP, ESP, and uncompressed**. [S.l.], jul 2001. RFC3095.

CASE, J. **Compressing TCP/IP Headers for Low-Speed Serial Links**. [S.l.], jan 1990. RFC1144.

CASNER, S.; JACOBSON, V. **Compressing IP/UDP/RTP Headers for Low-Speed Serial Links**. [S.l.], fev 1999. RFC2508.

CHEN, K.-H. **Energy Conservation in 3GPP/WLAN Integrated Networks**. Dissertação (Mestrado) — NATIONAL TSING HUA UNIVERSITY, mai 2005.

CISCO. **Cisco Visual Networking Index: Forecast and Methodology, 2014 – 2019**. mai 2015.

CNES; SPACE, T. A.; TECHNOLOGIES, V. **An efficient, free, and opensource header compression library**. ago 2015. Disponível em: <https://rohc-lib.org/>. Acesso em: 1 out. 2015.

COUVREUR, A. Performance analysis of a header compression protocol: The rohc unidirectional mode. **Telecommun Syst**, n. 31, p. 85–98, 2006.

COUVREUR, A. et al. Performance analysis of a header compression protocol: The rohc unidirectional mode. **Telecommun. Syst.**, Kluwer Academic Publishers, Norwell, MA, USA, v. 31, n. 1, p. 85–98, jan. 2006. ISSN 1018-4864. Disponível em: <http://dx.doi.org/10.1007/s11235-006-5524-z>. Acesso em: 1 out. 2015.

DEGERMARK, M.; NORDGREN, B.; PINK, S. **IP Header Compression**. [S.l.], fev 1999. RFC2507.

ECOMMERCEORG. **Evolução da Internet e do e-commerce**. ago 2015. Disponível em: <http://www.e-commerce.org.br/stats.php>. Acesso em: 1 out. 2015.

FARBER, D. J.; DELP, G. S.; CONTE, T. M. **A Thinwire Protocol for connecting personal computers to the INTERNET**. [S.l.], set 1984. RFC914.

FUNDOS, P. dos. **AMIGUINHO**. out 2015. Disponível em: <https://www.youtube.com/watch?v=NxzUU-cZD1o>. Acesso em: 1 out. 2015.

GOOGLE. **Relatório de qualidade de vídeo**. ago 2015. Disponível em: <https://www.google.com/get/videoqualityreport/>. Acesso em: 1 out. 2015.

IQBAL, F. **Performance Evaluation of Robust Header Compression Protocol for Low Data Rate Networks**. Dissertação (Mestrado) — UNIVERSITY OF AGDER, jun 2013.

LIAO, H. et al. **TCP-Aware RObust Header Compression (TAROC)**. [S.l.], 2000. Internet Draft.

PELLETIER, G. **RObust Header Compression (ROHC): Profiles for User Datagram Protocol (UDP) Lite**. [S.l.], abr 2005. RFC4019.

PELLETIER, G. et al. **RObust Header Compression (ROHC): A Profile for TCP/IP (ROHC-TCP)**. [S.l.], jul 2007. RFC4996.

PRICE, R. et al. **Framework for EPIC-LITE**. [S.l.], 2001. Internet Draft.

REPORT, I. S. G. I. **MOBILE EVOLUTION AND DEVELOPMENT OF THE INTERNET**. [S.l.], 2015.

REPORT, I. S. G. I. **MOBILE EVOLUTION AND DEVELOPMENT OF THE INTERNET**. 1h. ed. [S.l.], mai 2015.

SHARETECHNOTE. **LTE Quick Reference**. Disponível em: <http://www.sharetechnote.com/html/Handbook_LTE_ROHC.html>. Acesso em: 1 out. 2015.

ANEXO A – PASSOS PARA INSTALAÇÃO DO ROHC EM UM DEBIAN VERSÃO WHEZZY

```
# Instalacao de dependencias
apt-get update
apt-get install build-essential automake autoconf libtool git
    libpcap0.8 libpcap-dev sed gawk coreutils gnuplot doxygen
    cmake make gcc clang libyaml-dev libgnutls-dev iproute-dev
    openssl ssl-cert

# Compilacao e instalacao do rohc-lib
cd /usr/src
wget rohc-lib.org/download/rohc-1.7.x/1.7.0/rohc-1.7.0.tag.xz
wget rohc-lib.org/download/rohc-1.7.x/1.7.0/rohc-1.7.0.tag.xz.
    sha256
wget rohc-lib.org/download/rohc-1.7.x/1.7.0/rohc-1.7.0.tag.xz.
    asc
gpg --recv-keys 008E8DAD 1B2BB9C1
sha256sum -c rohc-1.7.0.tar.xz.sha256
gpg --verify rohc-1.7.0.tar.xz
gpg --verify rohc-1.7.0.tar.xz.asc
tar xvJf rohc-1.7.0.tar.xz
cd rohc-1.7.0/
./configure --prefix=/usr --enable-app-performance --enable-app
    -sniffer --enable-app-stats --enable-linux-kernel-module
make all
make check
make install

# Compilacao e instalacao do iprohc
```

```
cd /usr/src
wget https://launchpad.net/rohc/iprohc-0.7.x/iprohc-0.7.1/
    download/iprohc-0.7.1.tar.gz
tar -zxvf iprohc-0.7.1.tar.gz
cmake CMakeLists.txt -DCMAKE_INSTALL_PREFIX=/usr
make all
make install
ldconfig

# Gerar certificados e autoridades para criar o tunnel ROHC -
    Servidor (Encoder) - Router01
cd /usr/src
/usr/lib/ssl/misc/CA.ps -newca
/usr/lib/ssl/misc/CA.ps -newreq
/usr/lib/ssl/misc/CA.ps -sign
openssl pkcs12 -in newcert.pem -inkey newkey.pem -certfile
    demoCA/cacert.pem -out newcert.pl2 -export
mkdir demoCA/certs/IpRohcServer
mv new* demoCA/certs/IpRohcServer

# Gerar certificados e autoridades para criar o tunnel ROHC -
    Client02 (decoder) - Router02
cd /usr/src
/usr/lib/ssl/misc/CA.ps -newreq
/usr/lib/ssl/misc/CA.ps -sign
openssl pkcs12 -in newcert.pem -inkey newkey.pem -certfile
    demoCA/cacert.pem -out newcert.pl2 -export
mkdir demoCA/certs/IpRohcClient
mv new* demoCA/certs/IpRohcClient
# Agora enviar essa chave para o cliente que sera o decoder

# Ativando o ROHC Server (Router01)
cp /usr/src/demoCA/certs/IpRohcSever/newcert.p12 /etc/ssl/
    server.p12
cp /usr/src/iprohc-0.7.1/server/iprohc_server.conf /etc/
```

```
cp /usr/src/iprohc-0.7.1/debian/default/iprohc_server /etc/
    default/
cp /usr/src/iprohc-0.7.1/debian/init.d/iprohc_server /etc/init
    .d/
/etc/init.d/iprohc_server start

# Conectando ao ROHC server(Router02)
# Considerando que voce copiou toda a pasta IpRohcClient para /
    usr/src
cp /usr/src/IpRohcClient/newcert.p12 /etc/ssl/iprohc_client.p12
iprohc --remote 192.168.3.1 --port 3126 --dev iprohc --p12 /etc
    /ssl/iprohc_client.p12 --basedev eth0
# Onde 192.168.3.1 e o servidor e eth0 e a interface de saida
    para o servidor

# Capturando os pacotes (Ambos)
tcpdump -i eth1 -w arquivo_saida.pcap
```

ANEXO B – ARQUIVO DE CONFIGURAÇÃO DO IPROHC

```
# Config file for iprohc server

general:
    max_clients: 50                    # Maximum number of
       simultaneous clients
    port: 3126                         # TCP port to bind to
    pidfile: /var/run/iprohc_server.pid # Optional pid file
    p12file: /etc/ssl/server.p12       # Required pcks12 file

tunnel:
    ipaddr: 192.168.1.3/24 # Local IP address, client will be
       in the assoiciated /24 range
    packing: 5             # Packing value
    maxcid:  15            # Maximum allowed CID in ROHC
       compressor (must be <=16)
    unidirectional: 1      # Can be 0 or 1, describe the ROHC
       mode (1=unidirection, 0=bi)
    keepalive: 60          # Maximum time to receive keepalive
       before dying.
                           # The keepalives are sent every
                              third of this value.
# vim:ft=yaml
```