UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
CURSO DE ENGENHARIA DE COMPUTAÇÃO

ARTHUR BÖCKMANN GROSSI

Explorando Características de Qualidade de Serviço de Videoconferência na Internet

Monografia apresentada como requisito parcial para a obtenção do grau de Bacharel em Engenharia da Computação

Orientador: Prof. Dr. Luciano Paschoal Gaspary
Co-orientador: Prof. Dr. Roberto Irajá Tavares da Costa Filho

Porto Alegre
2023

Dedico este trabalho aos meus pais, familiares e amigos pelo amor, incentivo e apoio recebido durante toda a vida acadêmica.

AGRADECIMENTOS

Gostaria de expressar meus sinceros agradecimentos a todos que contribuíram para a realização deste Trabalho de Conclusão de Curso.

Primeiramente, quero agradecer à minha família pelo apoio e incentivo durante todo o meu percurso acadêmico. Sem o amor e o suporte de vocês, eu não teria chegado até aqui. Obrigado pela paciência nos momentos em que precisei desligar a internet de casa, em virtude dos experimentos deste trabalho. =)

Também gostaria de agradecer à minha namorada, pela paciência, compreensão, carinho nos momentos difíceis, e pelo incentivo em todos os momentos da minha jornada.

Não posso deixar de mencionar o meu orientador, Luciano Paschoal Gaspary, e meu co-orientador, Roberto Irajá Tavares da Costa Filho, por toda a orientação, dedicação, paciência e sabedoria compartilhada ao longo deste trabalho.

Agradeço também à Universidade Federal do Rio Grande do Sul, pela oportunidade de cursar uma graduação de qualidade e excelência, e pelos amigos que conheci. A esses amigos, dedico um agradecimento especial pelo incentivo, pelos momentos de descontração, pela companhia e pelas horas de estudo compartilhadas. Podem ter certeza, vocês foram essenciais em minha jornada acadêmica.

Expresso meus sinceros agradecimento à empresa Mconf Tecnologia, onde trabalho, por todo suporte fornecido durante o desenvolvimento deste trabalho. Agradeço especialmente ao meu colega de equipe, Paulo Renato Lanzarin, pela disposição para sanar dúvidas, pelo conhecimento compartilhado e pelo apoio ao longo deste projeto.

Por fim, agradeço a todos aqueles que, de alguma forma, contribuíram para esse trabalho, que me ensinaram algo novo e que fizeram parte dessa caminhada.

P.S.: Agradeço especialmente ao meu colega João Carlos Almeida da Silva por ter investido seu tempo em um feriado para me auxiliar no encerramento deste trabalho.

RESUMO

Nos últimos tempos, o uso de plataformas de videoconferência tem aumentado consideravelmente, principalmente devido à pandemia COVID-19, o que voltou as atenções para a qualidade da experiência do usuário. WebRTC é uma tecnologia que permite a troca de dados em tempo real e fornece uma ampla gama de indicadores, sendo amplamente utilizada pelas plataformas de videoconferência. A troca de dados de áudio e vídeo entre os participantes de uma videoconferência depende de uma série de fatores que possuem comportamento dinâmico, como a largura de banda dos participantes, a capacidade de processamento dos dispositivos utilizados e a localização geográfica. Nesse contexto, é crucial que as plataformas de videoconferência identifiquem e se adaptem a esses fatores para proporcionar uma boa qualidade de experiência aos usuários.

Dessa forma, a escolha cuidadosa de um conjunto de indicadores que melhor descrevam o estado da conexão e a saúde dos sistemas subjacentes, bem como o monitoramento constante desses indicadores, é essencial. Uma plataforma de videoconferência que monitora um conjunto adequado de indicadores consegue aferir a estimativa da qualidade da experiência do usuário de forma mais confiável e, assim, tomar medidas adaptativas para maximizá-la.

Para contribuir com esse cenário, este trabalho propõe uma abordagem de monitoramento de serviço de videoconferência. Para tal, projetou-se e desenvolveu-se um arcabouço para caracterizar o comportamento da plataforma de videoconferência Elos sob diferentes cenários. A metodologia adotada inclui a seleção de métricas de qualidade de serviço representativas, a montagem de dois ambientes experimentais e a execução automatizada de experimentos. A automatização dos experimentos permitiu a realização de um conjunto extenso e repetitivo de experimentos assim como a coleta das métricas selecionadas e a posterior geração de gráficos para análise dos resultados. Os resultados obtidos mostram que a plataforma de videoconferência Elos é eficiente na adaptação à cenários com limitações permanentes ou temporárias em sua capacidade de rede.

Palavras-chave: Vídeoconferência. WebRTC. QoS.

Exploring Quality of Service Characteristics of Videoconferencing over the Internet

ABSTRACT

In recent times, the use of videoconferencing platforms has increased considerably, mainly due to the COVID-19 pandemic, which has shifted attention towards the quality of the user experience. WebRTC is a technology that allows for real-time data exchange and provides a wide range of indicators, which are widely used by videoconferencing platforms. The exchange of audio and video data between videoconference participants depends on a series of factors that have dynamic behavior, such as the bandwidth of the participants, the processing capacity of the devices used, and the geographic location. In this context, it is crucial that videoconferencing platforms identify and adapt to these factors to provide a good quality of experience for users.

Thus, the careful selection of a set of indicators that best describe the state of the connection and the health of the underlying systems, as well as the constant monitoring of these indicators, is essential. A videoconferencing platform that monitors an adequate set of indicators is able to reliably estimate the quality of the user experience and take adaptive measures to maximize it.

To contribute to this scenario, this work proposes an approach to monitor videoconferencing service. To this end, a framework was designed and developed to characterize the behavior of the Elos videoconferencing platform under different scenarios. The adopted methodology includes the selection of representative quality of service metrics, the setup of two experimental environments, and the automated execution of experiments. The automation of experiments allowed for an extensive and repetitive set of experiments to be carried out, as well as the collection of the selected metrics and the subsequent generation of graphs for result analysis. The obtained results show that the Elos videoconferencing platform is efficient in adapting to scenarios with permanent or temporary limitations in its network capacity.

Keywords: Video conference, WebRTC, QoE.

LISTA DE ABREVIATURAS E SIGLAS

API	Application Programming Interface
DTLS	Datagram Transport Layer Security
GCC	Google Congestion Control
ICE	Interactive Connectivity Establishment
IP	Internet Protocol
MCU	Multi-point Conferencing Unit
NAT	Network Address Translation
QoE	Quality of Experience
QoS	Quality of Service
RTCP	Real-time Transport Control Protocol
RTP	Real-time Transport Protocol
SCTP	Stream Control Transmission Protocol
SDP	Session Description Protocol
SFU	Selective Forwarding Unit
SRTP	Secured Real-time Transport Protocol
STUN	Session Traversal Utilities for NAT
TCP	Transport Control Protocol
TURN	Traversal Using Relays around NAT
UDP	User Datagram Protocol
WebRTC	Web Real-Time Communication

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

A pandemia de COVID-19 que acometeu o mundo no ano de 2020 provocou uma mudança de comportamento geral na sociedade, sendo o *distanciamento social* uma das mais significativas. Uma grande parcela da população passou a ficar a maior parte do tempo isolada dentro de casa, desempenhando suas funções de forma remota, em regime de *home office*. A migração para o mundo digital, que já vinha ocorrendo de forma consistente no período pré-pandemia, foi acelerada. Naturalmente, tanto as conversas descontraídas quanto as reuniões de trabalho, que antes aconteciam de forma presencial, migraram para o mundo digital fazendo uso de plataformas de videoconferência. Nesse contexto, algo que não possui um paralelo direto no cenário presencial tornou-se objeto de grande atenção: a *qualidade da experiência de videoconferência*. Câmeras travando, vozes picotadas e participantes sem responder são indícios de problemas, muitas vezes oriundos de conexões de rede precárias.

Plataformas de videoconferência são um tipo de aplicação de tempo real que exige estabilidade de conexão bem como dos sistemas e serviços subjacentes. Até a menor das oscilações em algum fator pode causar uma degradação significativa na experiência do usuário e impactar a forma como é percebido e interage com os outros participantes. Exemplos desses fatores incluem a localização geográfica do usuário, a taxa de *download* e *upload* de pacotes, a latência da rede e a capacidade de processamento do dispositivo.

Diante do contexto mencionado, é *fundamental* que um provedor de serviço de videoconferência disponha de *caracterizações detalhadas, precisas e atualizadas sobre diferentes indicadores*, como os recém mencionados, que possam auxiliar na compreensão sobre o por quê determinados usuários estão tendo uma experiência satisfatória (ou não). Tal tem potencial, por exemplo, para subsidiar e justificar algum tipo de ação adaptativa dessas plataformas, até mesmo de maneira antecipada. Um exemplo é a troca para um *codec* de vídeo cujo *bitrate* médio seja mais adequado à largura de banda disponível.

Buscando contribuir na direção de soluções para o problema e a motivação apresentados, o objetivo deste trabalho é criar um arcabouço que permita a caracterização de plataformas de videoconferência, com foco na plataforma Elos(ELOS.VC,). O ponto de partida consiste em identificar um conjunto de indicadores representativos que permitam a avaliação da qualidade da conexão. A abordagem desenvolvida inclui o projeto e a implantação de dois cenários experimentais, assim como um mecanismo de coleta e armazenamento periódico desses indicadores. Esses são utilizados para obter uma com-

preensão sobre o estado das sessões e entender seu comportamento.

O restante deste documento está organizado da seguinte forma. No Capítulo 2 aborda-se as tecnologias e os fundamentos no âmbito de comunicação WebRTC, qualidade de serviço e qualidade de experiência. No Capítulo 3 descreve-se a arquitetura montada para o processo de medição, assim como o processo de seleção das métricas e o processo de implementação do aparato tecnológico. No Capítulo 4 apresenta-se e analisa-se os resultados obtidos durante os experimentos. Por fim, no Capítulo 5 tece-se uma conclusão sobre o trabalho desenvolvido, contendo considerações finais e aspectos interessantes a serem abordados em trabalhos futuros.

2 FUNDAMENTOS E ESTADO DA ARTE

Neste capítulo, aborda-se os tópicos relevantes e elementares sobre os quais este trabalho de conclusão de curso é fundamentado.

2.1 Real-time Transport Protocol e Real-time Transport Control Protocol

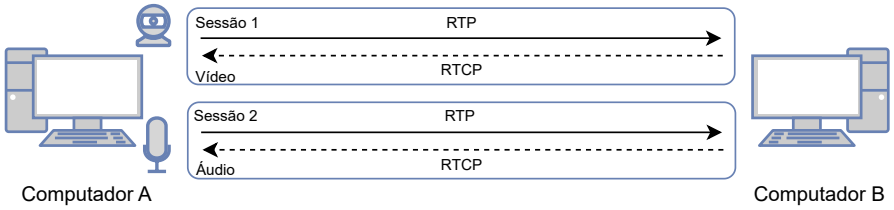
Os protocolos da camada de transporte permitem a conexão fim-a-fim e fornecem a comunicação lógica entre processos, além de oferecerem serviços ao nível de aplicação. O protocolo TCP (Transport Control Protocol) é um dos protocolos que atuam na camada de transporte, sendo tradicionalmente utilizado por oferecer serviços muito convenientes ao nível de aplicação. Alguns dos principais são transmissão confiável de *streams* de bytes, controle de fluxo e controle de congestionamento. Essas garantias são ótimas do ponto de vista de aplicações convencionais, onde a precisão na entrega de dados é indispensável. Contudo, para aplicações de tempo real, acabam sendo prejudiciais. Isso ocorre porque todas essas garantias oferecidas pelo protocolo TCP acabam acrescentando um atraso adicional devido à sobrecarga de sinalizações necessárias entre transmissor e receptor, mecanismo de retransmissões de pacotes perdidos e reordenamento de pacotes no receptor.

Aplicações que envolvem transmissão de mídia em tempo real, e que, por conseguinte, possuem uma tolerância a perdas maior que a tolerância a atrasos, tendem fortemente a utilizar o protocolo UDP. Esse é mais leve. Não oferece garantias de entrega de pacotes, mas também não adiciona atrasos, deixando a cargo da aplicação implementar esses serviços, caso, ainda assim, sejam necessários.

Neste contexto, surge o protocolo RTP, um protocolo desenvolvido pelo Audio/Video Transport Working Group da IETF, que atua na camada de aplicação e é específico para transporte de mídia. É tipicamente implementado sobre o protocolo UDP na camada de transporte e oferece ferramentas convenientes para cenários de transmissão de mídias, como: compensação de *jitter*, detecção de perdas de pacotes e detecção de entrega fora de ordem. Apesar de oferecer essas ferramentas, o protocolo RTP não estipula limites para latência ou confiabilidade. Sendo assim, fica a cargo da aplicação utilizá-las ou não para corrigir problemas intercorrentes. É um protocolo flexível, projetado para suportar diversos formatos de mídia, inclusive permitindo utilizar novos formatos sem que seja necessário revisitar sua especificação.

O RTP é um protocolo que tem seu funcionamento baseado em sessões. Uma sessão consiste em um grupo de participantes que estão se comunicando utilizando RTP e é projetada para transportar um único tipo de mídia. Em uma aplicação de videoconferência, por exemplo, devem ser criadas duas sessões separadas, uma para o fluxo de vídeo e outra para o fluxo de áudio, conforme ilustrado na Figura 2.1. Para que uma sessão seja iniciada, é necessário que as partes entrem em acordo sobre alguns parâmetros, entre eles o formato da mídia a ser transferida. Para isso, o protocolo possui perfis, os quais definem os *codecs* utilizados para codificar a mídia e, assim, indicam ao receptor, por meio de uma tabela estática ou de forma dinâmica, como interpretá-la. Um perfil muito utilizado para a transferência segura de mídia é o Secured Real-time Transport Protocol (SRTP), definido na RFC 3711(BAUGHER et al., 2004) Trata-se de uma extensão do perfil padrão para conferências de vídeo e áudio, definido na RFC 3551(SCHULZRINNE; CASNER, 2003), que adiciona uma camada de criptografia sobre a mídia transportada. O protocolo RTP também prevê o uso de protocolos externos para sinalização e negociação dos parâmetros de uma sessão. Um protocolo comumente utilizado para a negociação de sessões RTP é o protocolo Session Description Protocol (SDP), que define uma sintaxe padrão para a descrição desses parâmetros em formato texto. Nele estão descritos todos os formatos aceitos pelo iniciador da sessão para áudio e vídeo, além da indicação do título da sessão, endereço, porta, outras informações de contato e de temporização.

Figura 2.1: Exemplo de uma comunicação RTP

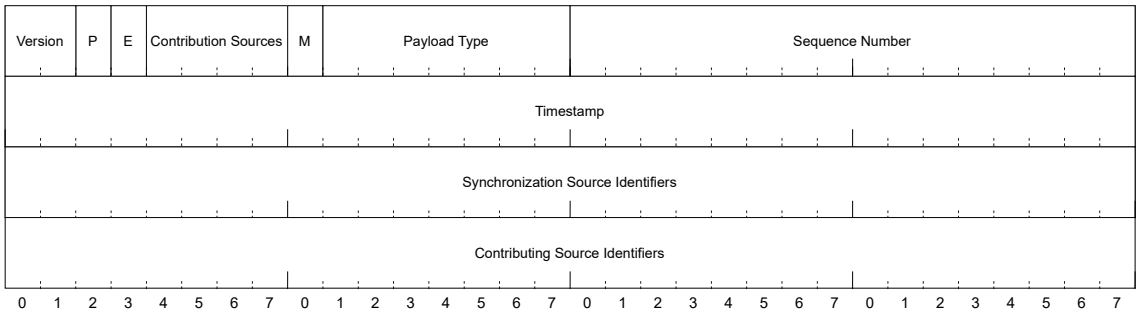


Fonte: O Autor

Uma vez que uma sessão RTP é iniciada, os pacotes transportando a mídia começam a ser enviados da fonte para os outros participantes. Cada pacote RTP possui um cabeçalho com informações essenciais que permitem aos receptores a identificação dos participantes e o processamento correto da mídia. Na Figura 2.2 é mostrada a estrutura de um cabeçalho RTP. Os primeiros campos do cabeçalho são o número da versão do protocolo RTP sendo utilizado, seguido de um bit indicador de *padding*. Logo após, existe um bit que indica a presença de extensão de cabeçalho, que é utilizado para suportar funcionalidades adicionais, como por exemplo, a inclusão de informações de sincronização

ou *timestamps* mais precisos. Em situações normais, os dados transmitidos usando RTP são gerados por uma única fonte, mas quando vários fluxos RTP passam por um *mixer* ou *translator*, o pacote resultante pode conter dados provindos de mais de uma fonte. A informação do número de fontes contribuintes aparece logo após o indicador de extensão de cabeçalho. Então, vem um bit de marcador, que é utilizado para marcar eventos de interesse dentro do fluxo de mídia. O tipo exato do evento indicado depende do campo seguinte, que identifica o formato do conteúdo sendo transmitido. Aqui consta o formato da mídia que foi negociado no início da sessão RTP. Logo após, vem o número de sequência do pacote, que tem como principal função indicar ao receptor se os pacotes estão sendo perdidos ou entregues fora de ordem. Em seguida, existe o campo *timestamp*, que indica o instante em que o primeiro octeto de dados do pacote foi amostrado. Os dois últimos campos são Sinchronization Source Identifier, que identifica o fluxo RTP, e o campo Contributing Source Identifiers, que contém uma lista de identificadores das fontes contribuintes para o conteúdo do pacote.

Figura 2.2: Estrutura de um cabeçalho RTP



Fonte: O Autor

Até o momento, falou-se somente do protocolo RTP para transporte de mídia, porém, existe um outro protocolo que é utilizado em conjunto chamado Real Time Control Protocol (RTCP). O protocolo RTCP fornece estatísticas e informações de controle sobre uma sessão RTP de forma periódica. Todos os participantes de uma sessão RTP enviam relatórios RTCP para o transmissor de mídia e para os outros participantes. Com as informações do relatório, a fonte toma conhecimento de como está a recepção da mídia para os participantes da sessão e, caso precise de ajustes, pode aplicar codificação adaptativa de mídia e detectar falhas de transmissão. Dado que todos participantes enviam relatórios, o tráfego de rede aumenta proporcionalmente ao número de participantes na sessão. Sendo assim, a fim de evitar congestionamento de rede, o protocolo possui um sistema de controle de banda da sessão. Além disso, a frequência da transmissão dos relatórios RTCP é randomizada para evitar que relatórios sejam transmitidos sincronizadamente, de forma

não intencional. Em geral, o uso de banda não excede 5% da largura de banda consumida pelas sessões ativas.

2.2 WebRTC

WebRTC, abreviatura de Web Real-Time Communication, é tanto uma API quanto um protocolo que fornece às aplicações suporte à comunicação em tempo real. Sendo assim, a tecnologia possui uma ampla gama de fins, como aplicações de videoconferência, jogos *online*, acesso remoto, compartilhamento de arquivos, entre outros. O protocolo WebRTC define um conjunto de regras para que dois agentes estabeleçam uma conexão e se comuniquem em tempo real, sendo mantido pelo RTCWeb Working Group do IETF. Já a API WebRTC é responsável por definir chamadas padronizadas de código para que desenvolvedores utilizem o protocolo WebRTC, sendo especificada somente para a linguagem JavaScript pelo World Wide Web Consortium (W3C). A tecnologia WebRTC fornece comunicação em tempo real para aplicações por meio de um modelo de conexão ponto-a-ponto. Para tal, ela agrupa e orquestra um conjunto de protocolos e mecanismos, pré-existentes ao seu surgimento.

Para que ocorra uma comunicação entre agentes WebRTC, são necessárias quatro etapas, sendo que em cada uma delas é utilizado um conjunto de protocolos. Importante notar que é necessário que as etapas ocorram de forma sequencial e que uma etapa somente seja executada depois que a etapa anterior for concluída. A primeira etapa é a *sinalização*, onde os pares se identificam e trocam informações para iniciar uma conexão. Neste ponto é utilizado o protocolo SDP (Session Description Protocol), mencionado na Seção anterior, onde os pares trocam informações sobre endereços IP e portas acessíveis, número e formato de fluxos de mídia, *codecs* suportados e *hashes* de segurança. Vale ressaltar que a sinalização acontece de forma isolada, ou seja, a aplicação não utiliza o WebRTC em si para trocar essas mensagens, sendo geralmente utilizada alguma outra forma de comunicação já existente na aplicação. Uma vez que os pares possuem as informações necessárias para iniciar uma conexão, avança-se para a segunda etapa.

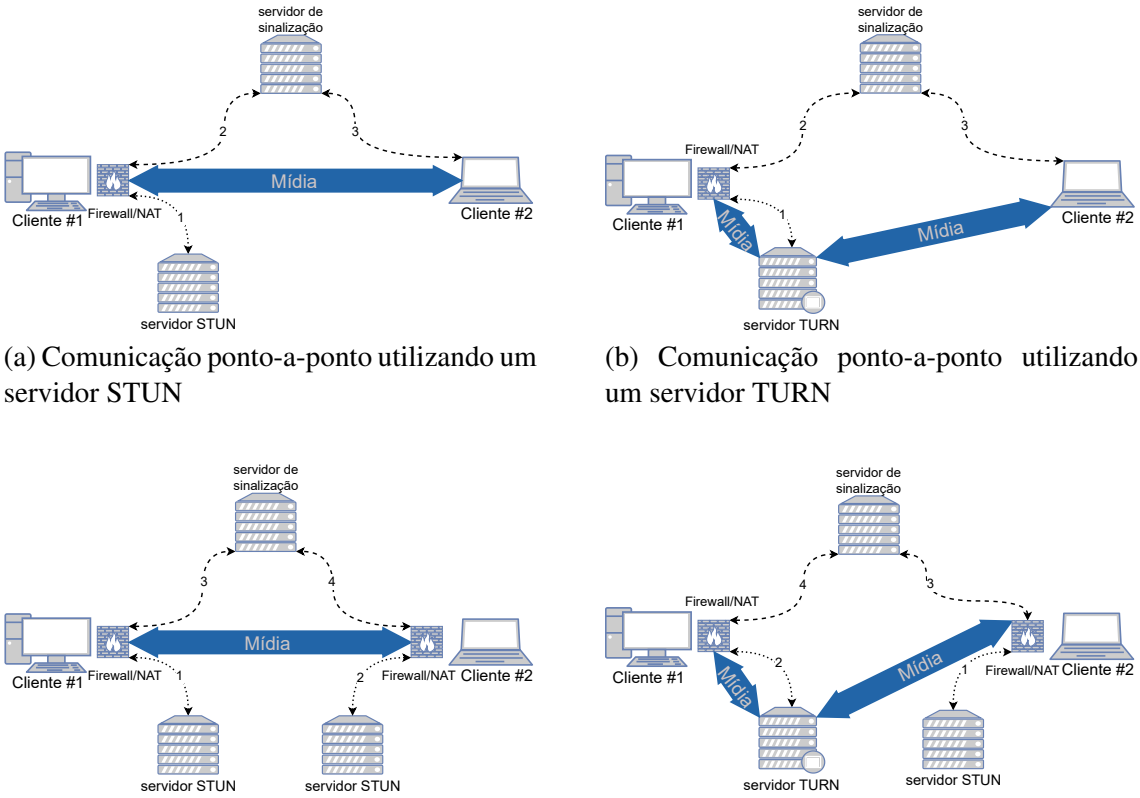
A *conexão* é a etapa mais complexa, consequência do modelo de comunicação ponto-a-ponto adotado pelo protocolo. Seria intuitivo que, neste momento, os pares usassem as informações de endereço IP e porta do outro agente adquiridas na etapa anterior para iniciar uma conexão, porém, na internet, as coisas não são tão simples. A chance de existirem obstáculos impedindo a conexão direta entre os dois pontos é grande. Mecanis-

mos de NAT(Network Address Translation) e de *firewall* são os exemplos mais comuns desses obstáculos. Por esse motivo, entra em ação um outro protocolo chamado Interactive Conectivity Establishment (ICE), o qual conduz os pares por uma série de testes para permitir a conexão entre os agentes, mesmo que eles estejam atrás desses mecanismos de segurança. Para isso, o protocolo ICE utiliza dois outros protocolos: o Session Traversal Utilities for NAT (STUN) e Traversal Using Relays around NAT (TURN). O primeiro consiste de um servidor externo ao qual os agentes se conectam para trocar dados. Tal procedimento é realizado para permitir a criação de um mapeamento inverso do NAT, que traduza o endereço público e porta para o endereço correspondente da rede local desse agente. Isso é possível quando o NAT em questão é do tipo full cone NAT, restricted cone NAT ou port restricted cone NAT. NATs desses tipos utilizam um mecanismo de mapeamento que independe do endereço IP do destinatário e, por conta disso, se mantém inalterados quando conexões de um mesmo IP interno são estabelecidas com diferentes IPs externos. Exemplos desses cenários são ilustrados nas Figuras 2.3a e 2.3c. Servidores do protocolo STUN são suficientes para permitir a conexão entre dois agentes em 70%[1] dos casos. Nos outros 30% restantes, não é possível estabelecer uma conexão direta entre os agentes. Dessa forma, o tráfego precisa ser intermediado por um servidor TURN, conforme apresentado nas Figuras 2.3b e 2.3d. Assim, ao final dessa etapa, os agentes possuem os seus endereços públicos ou equivalentes para que a conexão seja iniciada.

Uma vez que existe uma comunicação bidirecional por meio de uma conexão direta ou indireta entre os pontos, é necessário adicionar *proteção* a ela. Para criptografar o canal de dados, é utilizado o protocolo Datagram Transport Layer Security (DTLS), que consiste de uma junção do protocolo criptográfico TLS com o protocolo UDP, e permite a negociação de uma sessão para a troca de dados de forma segura. Durante a negociação, os pares trocam informações de identificação e autenticidade como certificados e *hashes*, por meio da conexão adquirida usando o protocolo ICE. São trocados certificados e uma sessão DTLS é iniciada caso esses certificados correspondam ao *hash* adquirido durante o processo de sinalização. Para concluir essa etapa, é necessário iniciar uma sessão RTP para a transmissão da mídia. Utiliza-se então o perfil SRTP do protocolo RTP, para que as mídias fluam por um canal de transferência criptografado. Essa criptografia tem como base as chaves trocadas na sessão DTLS estabelecida anteriormente. No momento em que há uma conexão onde os fluxos de mídia podem ser transferidos de forma segura, a comunicação WebRTC pode ser iniciada. Sendo assim, os pacotes de mídia

[1] https://www.callstats.io/blog/2017/10/26/webrtc-product-turn-server

Figura 2.3: Cenários de uma comunicação utilizando servidores STUN e TURN



(a) Comunicação ponto-a-ponto utilizando um servidor STUN

(b) Comunicação ponto-a-ponto utilizando um servidor TURN

(c) Comunicação ponto-a-ponto utilizando dois servidores STUN

(d) Comunicação ponto-a-ponto utilizando um servidor STUN e um servidor TURN

Fonte: (GARCíA et al., 2019)

transferidos seguem um fluxo onde são, primeiramente, criptografados utilizando o perfil SRTP e enviados por meio da sessão RTP. Por sua vez, o protocolo SCTP é utilizado para enviar informações arbitrárias pelo canal de dados criado na sessão DTLS. Todo o funcionamento do protocolo WebRTC descrito até aqui ocorre através do uso de chamadas da API WebRTC. Nela, são definidos objetos e abstrações que simplificam e padronizam o desenvolvimento de uma aplicação que utiliza o protocolo WebRTC. Dessa forma, o protocolo WebRTC torna-se interoperável e independente de navegadores e aplicações, o que facilita a adoção da tecnologia. Os principais elementos dessa API são:

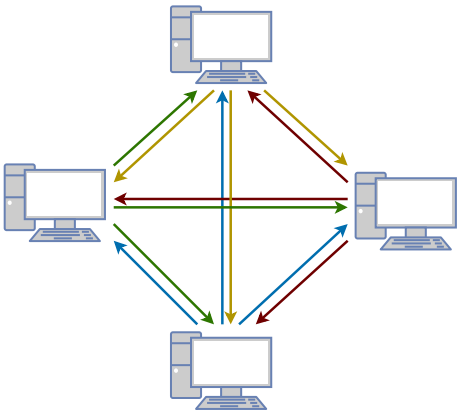
- RTCPeerConnection: Objeto que representa uma conexão WebRTC entre o computador local e um par remoto, sendo a entidade central de toda API. Possui todos os métodos relativos à comunicação WebRTC.
- createDataChannel: Cria um novo canal de dados, que corresponde a um novo fluxo SCTP. Canais de dados não são criados por padrão, porém basta que um dos pares solicite para que seja criado.
- createOffer e createAnswer: São os dois métodos utilizados para a negociação de

uma sessão. No momento que o método createOffer é chamado, uma descrição da sessão local é gerada para que seja trocada com o par remoto. Utilizando a chamada createAnswer o par remoto envia a sua descrição de sessão local. A conexão só é estabelecida caso haja parâmetros em comum nas sessões dos dois agentes.

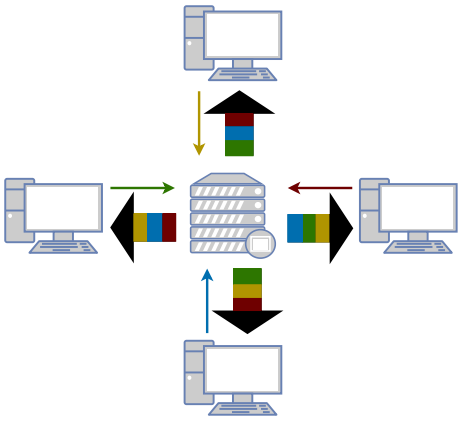
- MediaStream: Objeto que representa um fluxo de conteúdo de mídia. Um fluxo de conteúdo de mídia pode conter várias faixas de mídia, sendo elas de áudio ou vídeo.
- getUserMedia: Chamada de API que prepara a captura de um dispositivo de áudio ou vídeo. Retorna um objeto MediaStream.
- getStats: Chamada de API que retorna um objeto contendo várias informações sobre a conexão. Utiliza os relatórios gerados pelo protocolo RTCP.

Não é raro que haja mais de dois participantes em uma aplicação WebRTC. Como exemplo, temos as videoconferências com diversos participantes. O que possibilita essa agregação de vários integrantes em uma mesma aplicação, mesmo que o protocolo seja ponto-a-ponto, são as diferentes topologias que podem ser utilizadas, as quais estão ilustradas na Figura 2.4. A primeira topologia, ilustrada na Figura 2.4a, é a mais intuitiva, denominada Full Mesh. Nela, cada participante conecta-se diretamente aos outros participantes. Esse arranjo é o que introduz a menor latência na comunicação, pois não necessita de servidores de retransmissão. Contudo, devido à grande quantidade de conexões que cada agente precisa realizar (n-1 conexões), essa topologia utiliza uma maior largura de banda e também concentra nos clientes toda a carga de processamento. Dessa forma, não é uma topologia que escala bem para uma grande quantidade de participantes. A topologia ilustrada na Figura 2.4b chama-se Multi-point Conferencing Unit (MCU). Nesse cenário, cada agente WebRTC se conecta ao servidor de mídia central, o qual recebe as mídias de cada participante, as re-codifica, agregando-as em um único fluxo para, em seguida, redistribuí-las. Essa topologia acrescenta latência à comunicação por ter um servidor central, porém alivia os clientes da carga pesada de processamento e do uso excessivo de banda. A terceira topologia, que corresponde à Figura 2.4c, chama-se Selective Forwarding Unit. Sua diferença para a anterior é que o servidor de mídia central não agrega todos os fluxos de mídia dos participantes em um único fluxo, mas os redistribui como fluxos separados. Isso permite uma certa adaptação do servidor de mídia para cenários internos da aplicação de videoconferência, como disposição de *layouts*, levando a uma economia de recursos. Outro ponto importante dessa topologia é que alivia o servidor da carga de processamento decorrente da re-codificação e mixagem dos fluxos de mídia advindos dos participantes, quando comparado ao MCU.

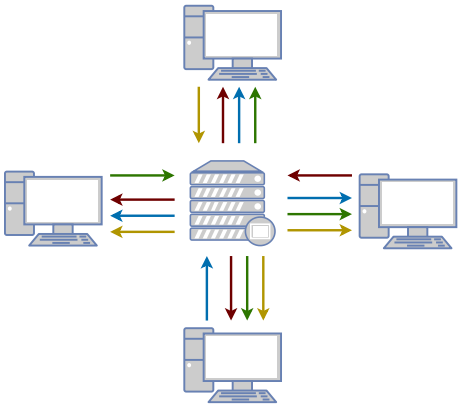
Figura 2.4: Topologias WebRTC



(a) Full Mesh



(b) Multipoint Control Unit



(c) Selective Forwarding Units

2.3 QoS e QoE em Aplicações de Videoconferência

Em uma videoconferência, quando um participante gera pacotes de mídia, ao falar no microfone ou ao enviar o video de sua câmera, por exemplo, esses pacotes de mídia são submetidos a uma série de processos até chegar aos outros usuários. Assim que esse conteúdo é gerado, o mesmo precisa ser gravado, codificado e comprimido, para logo após ser encapsulado em pacotes RTP e enviado pela rede. Uma vez na rede, ele passa por uma série de nodos intermediários, onde é roteado até chegar aos outros participantes. Nesse processo de entrega dos pacotes de mídia, existem fatores que adicionam degradações na comunicação, como consequência do funcionamento da Internet, dos mecanismos do protocolo e da topologia adotada pela aplicação. Como explicado na Seção 2.2, é comum que os pontos de uma comunicação WebRTC possuam NAT ou firewall configurados para a proteção da rede. Sendo assim, em muitos casos é necessário que se utilize um servidor de *relay* entre os pares, TURN, para permitir que esses pontos se

comuniquem. Dessa forma, todos os pacotes de comunicação entre esses pontos passam pelo servidor de TURN, aumentando a latência da comunicação. Além disso, é comum que aplicações de videoconferência utilizem servidores de mídia, que também adicionam latência à conexão, uma vez que toda mídia da comunicação é retransmitida nesse componente central. Dado esse grande número de processos intermediários entre a geração da mídia por um participante até a sua exibição para os outros participantes, é importante que se mantenha um controle/monitoramento do desempenho desses elementos, a fim de garantir o funcionamento eficiente e correto da aplicação. É nesse contexto que aparecem dois conceitos importantes: Quality of Service (QoS) e Quality of Experience (QoE).

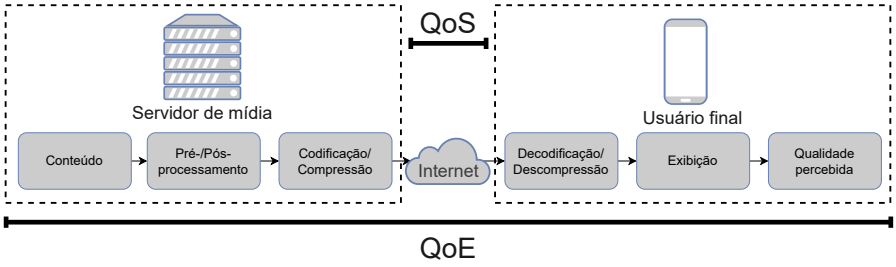
Segundo a ITU, QoS é "a totalidade das características de um serviço de telecomunicações que se relacionam com sua capacidade de satisfazer as necessidades declaradas e implícitas do usuário do serviço". Dessa forma, QoS mede o desempenho da arquitetura do serviço, sendo particularmente pertinente em aplicações que dependem de condições mínimas de rede para operar satisfatoriamente. A medição da qualidade de serviço é realizada por meio de parâmetros quantitativos técnicos, entre eles: latência, *jitter*, vazão e taxa de erros. Os principais desafios para a qualidade de serviço são restrições de largura de banda, canais de dados não confiáveis e tecnologias de acesso heterogêneas.

Existem mecanismos que podem ser aplicados a fim de maximizar a qualidade de serviço. Esses mecanismos podem ser divididos em duas categorias: de rede e de aplicação. Em uma situação onde a rede está recebendo uma quantidade excessiva de pacotes, os nodos intermediários, que realizam o roteamento desse pacotes, começam a ficar sobrecarregados. Isso leva a um aumento no tempo que os pacotes aguardam em filas antes de serem encaminhados (contribuindo para latência e *jitter*). Caso a frequência de envio de pacotes não seja controlada, as filas podem, eventualmente, ficar cheias e, assim, ocorrer perdas de pacotes. Algoritmos que adaptam a frequência do envio de pacotes às condições de rede, ou seja, algoritmos de controle de congestionamento, assim como algoritmos de controle de erros, são exemplos desses mecanismos de rede que ajudam a implementar qualidade de serviço. No cenário de aplicações de videoconferência utilizando WebRTC, o funcionamento desses algoritmos é baseado nas informações advindas dos relatórios do protocolo RTCP, que são requisitadas utilizando a chamada de alto nível *getStats* da API WebRTC. No lado da aplicação também existem medidas a serem tomadas a fim de melhorar a eficiência da transferência de dados. As principais são: esquemas avançados de codificação de vídeo, cancelamento de erros e protocolos de *streaming* adaptativo.

Mesmo que a rede esteja fornecendo uma ótima qualidade de serviço, existem di-

versos outros fatores que determinam a experiencia do usuário ao utilizar uma plataforma de videoconferência. A análise desse conjunto de fatores a fim de quantificar a experiencia do usuário é chamada de QoE. Segundo a ITU, QoE é definida como "a aceitabilidade geral de um aplicativo ou serviço, conforme percebido subjetivamente pelo usuário final". Ou seja, envolve uma série de indicadores subjetivos como o contexto e as expectativas do usuário para com a aplicação e, por esse motivo, é difícil de ser medido. A relação entre QoE e QoS está ilustrada na Figura 2.5.

Figura 2.5: Diferença entre QoS e QoE



Fonte: (TRESTIAN; COMSA; TUYSUZ, 2018)

Os métodos para medir QoE podem ser classificados como subjetivos e objetivos. Nos testes subjetivos, os usuários são expostos a questionários e avaliações sobre a experiencia com a plataforma. Os testes subjetivos são os mais válidos para se medir QoE, porém possuem algumas desvantagens. Entre elas, destaca-se o fato de que esses testes são, tipicamente, realizados em ambientes controlados, sob condições limitadas, custosos e demandam demasiado tempo do usuário. Tendo em vista esses aspectos, surgiram, então, os testes objetivos. Esses se propõem a estimar a experiência do usuário por meio de um modelo objetivo de qualidade, que utiliza como base os parâmetros de QoS agregados a indicadores externos. Esses modelos se diferenciam conforme o seu método de classificação, sendo eles ou baseados em padrões de distorção de vídeo, ou baseados em referências do vídeo original ou baseados em grupos de dados de entrada.

2.4 A Plataforma Elos

O Elos é uma plataforma de videoconferência voltada para ensino *online*, com diversas ferramentas, além do compartilhamento de áudio e vídeo. Tem como base o projeto de código aberto BigBlueButton (BIGBLUEBUTTON.ORG, 2019), que teve seu início no Instituto de Empreendedorismo e Comercialização de Tecnologia da Universidade de Carleton, em Ottawa no Canadá. Atualmente, o Elos é utilizado por mais de 700

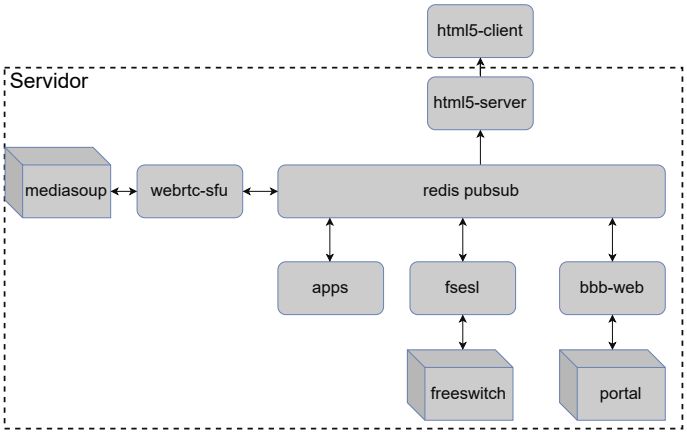
mil usuários.

A Figura 2.6 ilustra a arquitetura do Elos, onde são dispostos os componentes e sua comunicação.

- html5-client: módulo responsável pela visualização da conferência. É a aplicação *web* responsiva, onde o usuário interage com a interface para executar ações na conferência.
- html5-server: módulo implementado em Node.js e que serve a aplicação *web* html5-client.
- bbb-web: aplicação implementada em Java, responsável por tratar chamadas de API ao servidor. É a porta de comunicação para que outras aplicações que implementam um *front-end* se comuniquem com o servidor.
- portal: aplicação *front-end* que se comunica com a API bbb-web, além de permitir integrações com outras ferramentas externas: Moodle, Wordpress, Canvas, entre outras.
- redis pubsub: canal de comunicação dos módulos, o qual implementa o paradigma de troca de mensagens publish-subscribe.
- apps-akka: aplicação escrita em Scala que concentra toda a lógica de negócio da plataforma e orquestra o funcionamento da conferência.
- fsesl: componente adaptador que faz a ligação do sistema com a aplicação que trata o áudio da conferência. Dá maior flexibilidade à arquitetura.
- FreeSWITCH: servidor de áudio da plataforma, realiza a mixagem e a gravação do áudio da conferência.
- webrtc-sfu: módulo que atua como controlador de mídia, tratando negociações de mídia e gerenciando os fluxos.
- mediasoup: servidor de mídia da plataforma que implementa a topologia SFU e é responsável pelos fluxos de áudio do modo ouvinte, do vídeo de compartilhamento de tela e do vídeo das câmeras.

Quando um usuário entra em uma conferência, ele escolhe conectar-se no modo ouvinte ou no modo microfone. Caso escolha o modo ouvinte, cria-se somente um fluxo RTP para recebimento do áudio dos outros participantes. Já no modo microfone, são necessários 2 fluxos RTP, um para recebimento do áudio dos outros participantes e outro para envio do áudio do microfone local. O *codec* padrão utilizado para áudio é o Opus. Tratando-se de vídeo, o fluxo é um pouco diferente. O componente webrtc-sfu é

Figura 2.6: Arquitetura de módulos e componentes do Elos



Fonte: O Autor

responsável pela parte da sinalização, negociação dos parâmetros da conexão WebRTC e gerenciamento dos fluxos de mídia. O *codec* padrão utilizado para vídeo é o VP8. O servidor de vídeo utilizado no Elos é o mediasoup, o qual implementa a topologia SFU, explicada na Seção 2.2. Sendo assim, uma vez que a negociação termina, são criados fluxos para recebimento do vídeo de cada participante que está compartilhando sua câmera e é criado um fluxo para envio do vídeo da câmera local.

2.5 Trabalhos Relacionados

O crescente uso de plataformas de videoconferência para a realização de atividades gerou também uma maior preocupação com a qualidade de experiência. Nesse contexto, no passado recente foram publicados artigos que buscam estudar diferentes aspectos relacionados a aplicações de videoconferência e qualidade. A seguir, descreve-se três investigações identificadas como importantes para o desenvolvimento deste trabalho.

Iniciou-se pelo trabalho do autor Doreid Ammar (AMMAR et al., 2016), publicado em 2016, onde foram exploradas as estatísticas de uma sessão WebRTC a fim de identificar a potencial relevância dessas informações em indicar problemas na qualidade da experiência dos usuários de uma videoconferência. Para isso, o autor realizou uma série de testes em um ambiente de videoconferência com dois usuários, utilizando a plataforma *appear.in* (hoje conhecida como *whereby.com*). Durante os testes, foram simuladas diferentes condições de qualidade de conexão em ambos os participantes, e, em seguida, analisou-se como essas condições se refletiam nas estatísticas geradas pelo protocolo WebRTC. Neste trabalho ficou claro que existem diversas métricas oferecidas pelo protocolo

WebRTC que permitem estimar a qualidade da experiência do usuário.

Na sequência, em 2018, foi publicado um trabalho do autor Bart Jansen (JANSEN et al., 2018), onde realizou-se um estudo sobre os efeitos de diferentes condições de rede no funcionamento do protocolo WebRTC. O trabalho focou-se no comportamento dos algoritmos de controle de congestionamento e o desempenho dos novos *codecs* de vídeo VP9 e H.264. Neste trabalho, também foi feita uma comparação de desempenho do protocolo WebRTC em computadores e dispositivos móveis. Para tal, o autor implantou dois ambientes de experimentos, um deles configurado para emular condições degradadas de rede de forma sintética, utilizando Dummynet, e outro onde foram utilizadas redes reais, focado, principalmente, em dispositivos móveis conectados a redes sem fio. O trabalho demonstrou que a inclusão de um mecanismo de limiar adaptativo no protocolo de controle de congestionamento, GCC, contribuiu para a equidade no uso da rede pelas aplicações WebRTC, na presença de outros fluxos TCP. Da mesma forma, ficou claro que os *codecs* testados não desempenharam como esperado na presença de situações de congestionamento de rede ou perda de pacotes. Por fim, identificou que o desempenho de aplicações WebRTC em dispositivos móveis é prejudicada devido a perdas de pacotes em rajadas e retransmissões as quais são frequentes em redes sem fio.

No ano de 2021, o autor Kyle MacMillan publicou o trabalho (MACMILLAN et al., 2021), que tinha como objetivo entender a utilização de rede das aplicações de videoconferência. O estudo analisou a variação das métricas de desempenho sob diferentes capacidades de vazão de rede, a resposta das aplicações de videoconferência a interrupções na comunicação de rede e diferentes modalidades de uso, bem como o comportamento dessas aplicações na presença de outros fluxos de rede. Para isso, os autores implantaram duas configurações experimentais. A primeira delas composta de dois computadores com especificações suficientes e idênticas para executar as aplicações de videoconferência, e foram submetidos a restrições aplicadas de forma automatizada utilizando a ferramenta Traffic Control[2]. No segundo experimento, foram conectadas duas máquinas a um *switch*. Uma delas acessava a plataforma de videoconferência e a outra executava uma aplicação, sendo elas Netflix, iperf ou outra aplicação de videoconferência, para gerar tráfego de competição no *link* da rede. O trabalho concluiu que as três aplicações de videoconferência estudadas variam significativamente nos experimentos aplicados devido aos seus diferentes mecanismos de controle de congestionamento e de transporte, assim como pelas estratégias de codificação e infraestrutura utilizadas. Outro ponto demonstrado é que

[2]https://tldp.org/HOWTO/Traffic-Control-HOWTO/intro.html

a utilização de rede por essas aplicações pode variar significativamente, a depender de configurações de *layout* aplicadas, incluindo casos onde conferências com menos participantes consumem mais banda que conferências com um maior número de participantes para a mesma plataforma.

Como se observa na literatura, é comum a utilização de indicadores quantitativos para caracterizar o funcionamento de aplicações de videoconferência. A monitoração destes indicadores permite uma análise ampla do comportamento dessas aplicações e sua qualidade em relação a diferentes aspectos, desde restrições em maior ou menor escala até o uso de diferentes *codecs*. A partir desses estudos, é possível obter informações relevantes sobre o contexto da videoconferência e, assim, entender melhor o comportamento dessas aplicações.

Inspirado nessas pesquisas, o presente trabalho propõe uma análise focada na plataforma de videoconferência Elos, utilizada por mais de 700 mil usuários[3], mas que ainda carece de estudos específicos. O objetivo é realizar uma análise mais detalhada do tráfego de uma aplicação de videoconferência real. Busca-se compreender seu perfil de uso e entender como essa aplicação se comporta em situações de restrições de rede, ao simular cenários que estão mais próximos da realidade de países, como o Brasil, onde a capacidade de internet disponível à população tende a ser limitada. Essa investigação se aplica não apenas a essa plataforma específica, mas também a todos os projetos baseados em BigBlueButton, permitindo uma compreensão mais clara do desempenho dessas plataformas em diferentes contextos e trazendo insumos relevantes para desenvolvimento de mecanismos que promovam uma melhor qualidade de experiência.

[3]https://blog.elos.vc/como-a-pearson-adaptou-se-aos-desafios-da-pandemia-atraves-do-ensino-a-distancia/

3 MEDIÇÕES DE QOS E QOE NA PLATAFORMA ELOS

Conforme análise comparativa da Seção 2.5, este trabalho se propõe a observar, caracterizar e compreender o efeito do uso de infraestruturas de rede com capacidade limitada em relação a seus indicadores de funcionamento, em aplicações de videoconferência, de modo especial na plataforma Elos. Ressalta-se que, embora QoE seja relevante para entender o sentimento do usuário, este trabalho foca-se na medição objetiva de QoS, o qual permite estimar a contribuição dos componentes de rede para composição de QoE. A fim de viabilizar esse estudo, foi necessário o desenvolvimento de três pontos: (*i*) a elaboração de uma arquitetura de medições que permita a realização de medidas de maneira controlada, sem que sofram interferências de fontes externas; (*ii*) o levantamento de indicadores adequados, os quais permitam atestar o efeito de degradações de rede na qualidade de experiência do usuário de aplicações de videoconferência; e (*iii*) a implementação de ferramentas de software auxiliares que permitam a automatização do processo de experimentação. Na sequência deste capítulo, apresenta-se cada um desses pontos em detalhe.

3.1 Arquiteturas de Medição

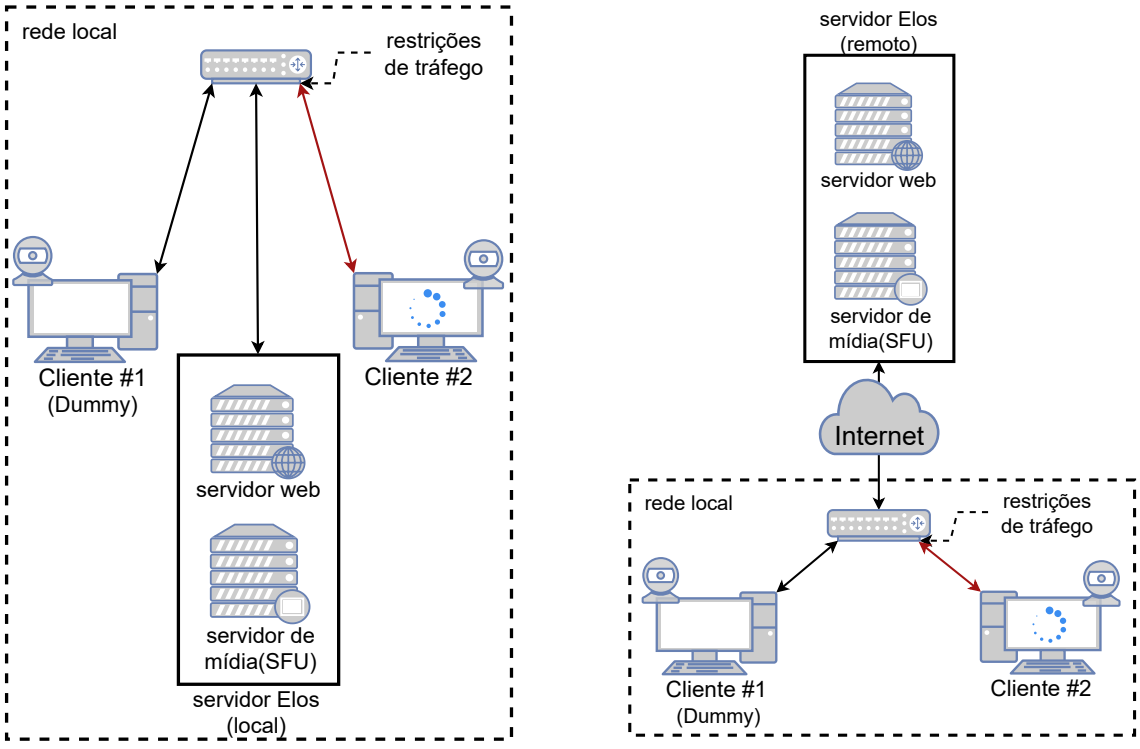
Para poder realizar um conjunto extensivo de experimentos que envolvam aplicações de videoconferência, sob diferentes níveis de degradação de rede, foram considerados dois ambientes experimentais. O primeiro, ilustrado na Figura 3.1a, foi concebido com a intenção de simular, ao máximo, um cenário ideal, onde gargalos provenientes da rede podem ser desprezados, assim como as degradações provenientes de mecanismos de conexão do protocolo WebRTC, como TURN. Esse ambiente é composto por dois clientes conectados por cabo ao roteador da rede local e que interagem com uma instância de um servidor Elos. Esse servidor foi implantado localmente em uma máquina com especificações suficientes para sua execução e encapsulado em um *container docker* em modo *bridge* com a rede local. O roteador da rede local possui capacidade de 1 Gbps.

O segundo ambiente experimental é composto por dois clientes interagindo entre si em uma videoconferência. Esses clientes são conectados por cabo à rede de Internet, e, por sua vez, trocam pacotes com um servidor remoto que hospeda a aplicação Elos, conforme mostrado na Figura 3.1b. A ideia desse ambiente é simular um cenário real, onde oscilações de rede e mecanismos de retransmissão decorrentes da conexão WebRTC

podem estar presentes.

O fluxo geral nos dois experimentos é semelhante. Os clientes se comunicam com um servidor *web* da plataforma Elos, o qual corresponde ao componente html5-server da arquitetura, mostrada anteriormente na Figura 2.6. Lembra-se que esse componente é responsável por fornecer comunicação à interface de interação dos clientes dentro da videoconferência. Enquanto isso, a transferência das mídias é intermediada por um servidor de mídia, em uma topologia SFU, onde os clientes enviam o seu fluxo e recebem os fluxos dos outros participantes, separadamente. Para tal, os clientes e o servidor utilizam a pilha de protocolos explicada nas seções anteriores, ocorrendo a transferência de mídia, em tempo real, por meio do protocolo WebRTC. Sendo assim, cada cliente fará parte de uma sessão RTP correspondente à transferência de uma mídia. Os clientes e o servidor enviam *feedbacks* sobre a recepção dessas mídias por meio do protocolo RTCP, a fim de informar a "saúde" da comunicação.

Figura 3.1: Ambientes experimentais



(a) Ambiente ideal

(b) Ambiente real

Fonte: O Autor

Sobre essa infraestrutura, que está em sintonia com a utilizada em trabalhos anteriores e de forma especial no trabalho (MACMILLAN et al., 2021), a ideia é criar um mecanismo que permita, de forma controlada, gerar degradação em parâmetros de funcionamento do canal de comunicação e, assim, observar os efeitos na experiência de uso

da plataforma. Sobre os mecanismos para aplicar degradações no canal de comunicação, explorou-se três ferramentas diferentes: tc, wondershaper e controle de banda no roteador. Adotou-se essa última, por ter se mostrado mais robusta, *i.e.*, com a aplicação mais consistente das restrições. Vale notar que ao degradar a rede no roteador, é necessário configurar um endereçamento IP à parte para isolar as outras máquinas da que sofre degradações.

3.2 Métricas de Interesse

Tendo em vista a vasta quantidade de indicadores sobre aspectos de qualidade da comunicação que a API WebRTC oferece, assim como a facilidade na sua captura, acabou-se utilizando esses indicadores como a fonte mais importante de informação sobre uma videoconferência. Como essa API dá acesso a mais de 200 métricas, sendo elas muito diversas, optou-se por fazer uma triagem sobre esse conjunto, a fim de identificar as mais relevantes para este trabalho.

Em uma primeira iteração fez-se um processo de triagem que se sucedeu da seguinte forma. Primeiramente, foram selecionadas métricas com base em sua semântica e que, à primeira vista, possuíam relação direta com a experiência de uso. Logo após, submeteu-se as métricas selecionadas a um critério de compatibilidade, onde permaneceram as métricas suportadas pelas três plataformas de *browser* mais comuns, *i.e.*, Mozilla Firefox, Google Chrome e Safari. Por último, atribuiu-se uma maior importância a indicadores que foram utilizados em trabalhos anteriores relacionados a este. Sendo assim, ao final, chegou-se a três categorias de indicadores: *claramente importantes*, *claramente não importantes* e *não definidos*. Indicadores que acabaram integrando o último grupo são indicadores que podem, ou não, ser relevantes para trabalhos relacionados, mas que acabaram não sendo selecionados para este trabalho. Os indicadores classificados ao final desta iteração como *claramente importantes* são mostrados na Tabela 3.1.

A primeira iteração sobre o conjunto amplo de métricas mostrou que a análise desses indicadores é complexa por si só. Como realiza-se um estudo sobre ambiente com transmissões de *bitrate* variável, muitos aspectos da infraestrutura e até o conteúdo da transmissão podem causar oscilações nessas métricas. Sendo assim, em uma segunda iteração, decidiu-se restringir o conjunto das métricas a um subconjunto do grupo das métricas *claramente importantes*, que são mostradas na 3.2. Essa redução tem como objetivo permitir a realização de um exercício de qualidade em compreender como elas se relaci-

onam. Outro ponto importante a ser citado é que, durante a realização dos experimentos, deparou-se com métricas que não estavam na definição da API de estatísticas WebRTC. Acabou-se descobrindo que são métricas derivadas, calculadas pelos *browsers* com base nas métricas originais da API, e que essas métricas têm seu nome sempre envolto por colchetes (*[ ]*). A lista das métricas derivadas utilizadas e suas respectivas fórmulas são apresentados na Tabela 3.3.

Tabela 3.1: Tabela com as métricas selecionadas na primeira iteração do TG

Grupo	Métrica	Descrição
InboundRTP	bytesReceived	Número de bytes recebidos de um SSRC
InboundRTP	nackCount	Número total de NACKs enviados por este receptor
InboundRTP	packetsReceived	Número total de pacotes recebidos de umSSRC
InboundRTP	packetsLost	Número total de pacotes perdidos de um SSRC
InboundRTP	jitter	Jitter de pacote medido em segundos
InboundRTP	packetsDiscarded	Número total de pacotes RTP descartados pelo jitter buffer
InboundRTP	framesPerSecond	Número de frames de um SSRC, medidos no ultimo segundo
InboundRTP	firCount	Numero de vezes que esse receptor do fluxo enviou pacotes FLI
InboundRTP	pliCount	Numero de vezes que esse receptor do fluxo enviou pacotes PLI
OutboundRTP	retransmittedPacketsSent	Número total de pacotes que foram retransmitidos
OutboundRTP	retransmittedBytesSent	Número total de bytes de payload que foram retransmitidos
OutboundRTP	nackCount	Número total de NACKs recebidos por esse transmissor
OutboundRTP	packetsSent	Número total de pacotes enviados para esse SSRC
OutboundRTP	targetBitrate	Taxa de bits alvo do codificador
OutboundRTP	bytesSent	Número total de bytes enviados para esse SSRC
OutboundRTP	pliCount	Número de vezes que esse transmissor do fluxo recebeu pacotes PLI
OutboundRTP	frameWidth	Largura do ultimo frame codificado
OutboundRTP	frameHeight	Altura do ultimo frame codificado
OutboundRTP	framesPerSecond	Número de quadros codificados no último segundo
OutboundRTP	qpSum	Soma dos valores QP dos frames codificados
OutboundRTP	framesEncoded	Número de quadros codificados

Fonte: O Autor

Tabela 3.2: Tabela com as métricas selecionadas na segunda iteração do TG

Grupo	Métrica	Descrição
OutboundRTP	targetBitrate	Taxa de bits alvo do codificador
OutboundRTP	bytesSent	Número total de bytes enviados para esse SSRC
OutboundRTP	frameWidth	Largura do ultimo frame codificado
OutboundRTP	frameHeight	Altura do ultimo frame codificado
OutboundRTP	framesPerSecond	Número de quadros codificados no último segundo
OutboundRTP	qpSum	Soma dos valores QP dos quadros codificados
OutboundRTP	framesEncoded	Número de quadros codificados

Fonte: O Autor

Tabela 3.3: Tabela com as métricas derivadas que foram utilizadas no TG

Grupo	Métrica	Fórmula
OutboundRTP	[bytesSent_in_bits/s]	$8 \cdot \frac{currReport.BytesSent - prevReport.BytesSent}{currReport.Time - prevReport.Time}$
OutboundRTP	[qpSum/framesEncoded]	$\frac{qpSum}{framesEncoded}$

Fonte: O Autor

3.3 Implementação

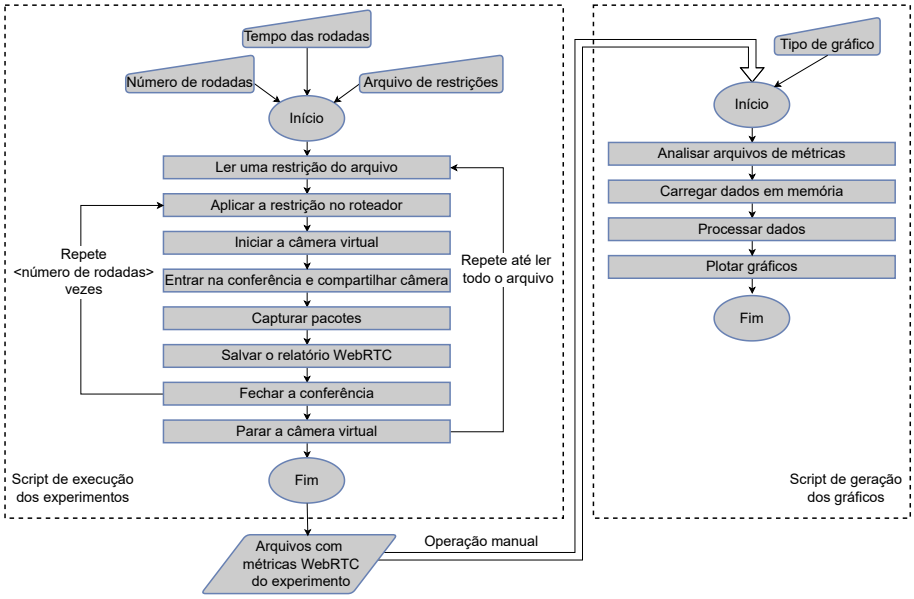
Dada a arquitetura de medições introduzida na Seção 3.1, e as métricas de interesse apresentadas na Seção 3.2, parte-se para a implementação e a instanciação de um ambiente de experimentos que permita a realização de um conjunto de observações. Foram desenvolvidos dois grupos de *scripts*. O primeiro grupo é responsável pela execução dos experimentos, onde toda a interação com a interface da plataforma Elos foi automatizada, assim como a aplicação das restrições na rede, o que possibilita a repetição dos experimentos, sem demandar demasiado esforço manual. O segundo *script* é responsável pela interpretação e geração dos gráficos com base nos dados coletados durante a execução do primeiro conjunto de *scripts*.

Para a automatização dos experimentos, foi utilizado como base o *script* Python[1] desenvolvido no trabalho (MACMILLAN et al., 2021), que já estava adaptado às plataformas Google Meet, Zoom e Microsoft Teams, mas não para o Elos. Então, foram mapeadas e implementadas[2] todas as interações necessárias para o ingresso em uma videoconferência, assim como compartilhamento de câmera e saída de uma videoconferência no Elos. Para tal, o *script* utiliza duas bibliotecas que trabalham em conjunto, PyAutoGui, que executa cliques e controla o teclado, e a GuiBot, que identifica elementos na tela por meio de visão computacional e inteligencia artificial, permitindo que elementos sejam identificados somente pela captura de tela. O fluxo dos *scripts* é mostrado na Figura 3.2 e descrito a seguir.

No início de cada experimento, o *script* faz a leitura de um arquivo texto, onde estão definidas as restrições de rede a serem aplicadas. Para cada restrição de rede, por padrão, são repetidas cinco rodadas do experimento, visando a uma maior base estatística, sendo que esse número deve ser ajustado em função da variabilidade do desempenho do ambiente em questão. Uma vez que uma rodada se inicia, um conjunto de restrições é aplicado por meio de uma conexão SSH ao roteador, e, com o auxilio da biblioteca ffmpeg e o módulo v4l2loopback, é emulado um dispositivo de câmera reproduzindo em ciclo um vídeo predeterminado do tipo *talking head*. Então, depois de entrar na conferência e habilitar a câmera, inicia-se uma captura de pacotes que dura um tempo pré-configurado. Sendo assim, ao final desse tempo, salva-se o relatório fornecido pela página interna[3] do navegador Google Chrome, onde consta uma série de informações sobre a sessão We-

[1] https://github.com/kyle-macmillan/vca-imc-21
[2] https://github.com/arthurk12/vca-imc-21
[3] chrome://webrtc-internals

Figura 3.2: Fluxo completo de um experimento



Fonte: O Autor

bRTC.

Após a realização de um experimento, os dados coletados são copiados para um armazenamento em nuvem para que sejam acessados por um segundo *script* Python que executa no ambiente Google Colab. Nele, foi implementado um analisador de arquivos de *dump* WebRTC, que extrai as informações e as armazena em uma estrutura de dados em memória, para que, com auxílio da biblioteca matplotlib, os gráficos sejam plotados. Esse *script* foi pensado para que seja possível alterar, de forma simples, os arquivos a serem analisados, assim como as métricas que irão compor os gráficos.

4 RESULTADOS

Neste capítulo, são apresentados os resultados obtidos no desenvolvimento deste trabalho. Com o intuito de enriquecer a análise, os resultados foram divididos em três seções, as quais versam sobre diferentes situações e cenários experimentais.

4.1 Entendendo o Comportamento de um Vídeo em Condições Ideais

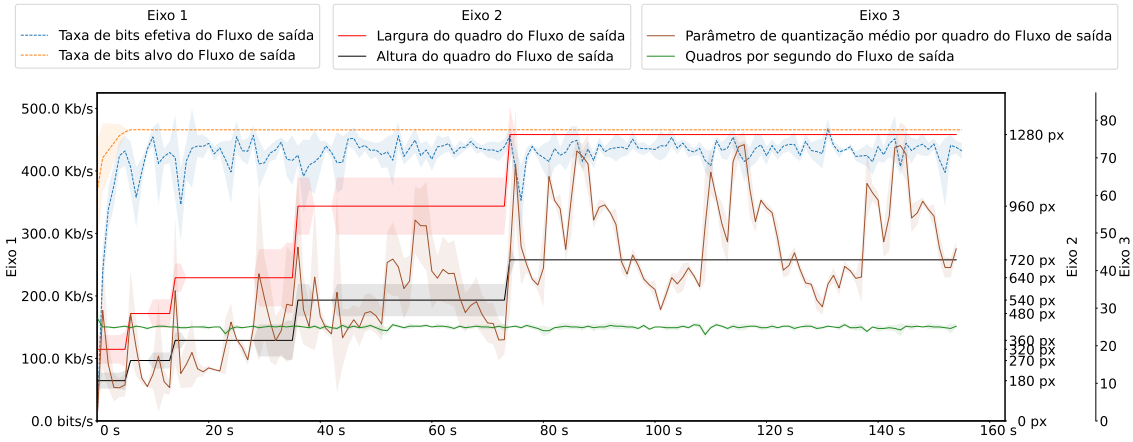
Para entender o comportamento de uma plataforma de videoconferência quando seus usuários estão sob condições de rede degradadas, é necessário, primeiro, entender como a plataforma se comporta em cenários ideais. Nesta seção, são apresentados e analisados os resultados obtidos ao executar os experimentos nessas circunstâncias. O objetivo desses experimentos é entender como a plataforma se comporta e avaliar o desempenho dos seus componentes. Os dados obtidos fornecem informações valiosas para a compreensão das capacidades da plataforma em contextos ideais, bem como para a identificação de possíveis limitações inerentes da plataforma.

A fim de entender o comportamento de uma videoconferência, excluindo-se, ao máximo, a influência da rede para com a qualidade da experiência do usuário, foi realizado um experimento usando-se o ambiente ilustrado na Figura 3.1a. O experimento simula um cenário ideal, onde o servidor da plataforma de videoconferência e os clientes estão na mesma rede. Esse experimento, ao qual atribuímos o nome de “Canônico”, foi conduzido sem a aplicação de restrições na rede e executado com cinco rodadas, cada uma com duração de 150 segundos. No gráfico da Figura 4.1, são mostradas a média da taxa alvo de bits por segundo, da taxa efetiva de bits por segundo, dos quadros por segundo e do parâmetro de quantização médio por quadro. Por outro lado, as curvas da largura e altura dos quadros do vídeo enviado são representativas da moda, uma vez que essas medidas são discretas. Adicionalmente, os sombreamentos presentes no entorno das curvas indicam o cálculo do desvio padrão para as amostras em questão.

Ao analisar o gráfico, é possível perceber que a taxa de bits por segundo alvo calculada pelo algoritmo de congestionamento, atinge um patamar de cerca de 450 Kbits/s, que é, aproximadamente, o máximo de banda consumida pelo compartilhamento de câmera com a qualidade mais alta[1], e se mantém assim até o fim do experimento. Esse fato, atrelado à uma analise complementar que constatou que não houveram bytes re-

[1] https://ajuda.elos.vc/kb/article/140985/requisitos-tecnicos-para-usar-o-elos

Figura 4.1: Desempenho do Elos em ambiente ideal e sem restrições de banda



Fonte: O Autor

transmitidos durante este experimento, indica, de fato, que a rede não sofreu degradações indesejadas e reforça a tese de que esse cenário se aproxima muito de um cenário ideal.

Pode-se observar, também, um comportamento contraintuitivo na relação entre a taxa de bits efetivamente enviada e a resolução. Constatou-se que a taxa de bits efetivamente enviada permanece em um patamar estático, com pequenas oscilações, independentemente da variação na resolução. Esse fenômeno não pode ser explicado pela variação na quantidade de quadros por segundo enviados, pois, conforme observado no gráfico, a curva deste indicador mantém-se em torno de 30 quadros por segundo, apresentando poucas oscilações durante o experimento.

Com base nesses resultados, acredita-se que a estabilidade da taxa efetiva de bits por segundo possa ser explicada, em parte, pela curva do parâmetro de quantização médio por quadro. Conforme a resolução é elevada, há uma tendência de aumento desse parâmetro, o que significa uma redução na qualidade do vídeo[2]. No entanto, essa redução na qualidade é compensada pela elevação na resolução, resultando em uma taxa efetiva de bits estável. Essa hipótese é reforçada pela presença de picos na curva do parâmetro de quantização médio, que coincidem com as trocas de resolução, a fim de evitar saltos abruptos na taxa de bits efetiva.

É importante destacar que, embora a taxa alvo de bits por segundo tenha permanecido estável na maior parte do experimento, ocorreram oscilações na taxa de bits enviados por segundo. Essas oscilações são influenciadas pelo conteúdo do vídeo compartilhado, o que pode resultar em picos no gráfico, quando há transições abruptas de muitos *pixels*, e vales, quando o vídeo permanece estático. Além disso, à medida que o experimento

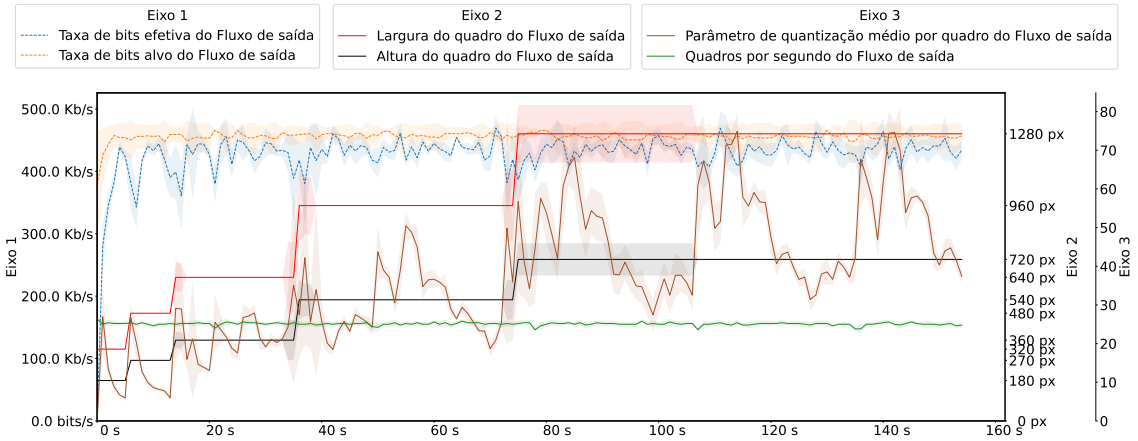
[2]https://developer.mozilla.org/en-US/docs/Web/API/RTCRtpStreamStats/qpSum

progride, a resolução do vídeo aumenta para aproveitar ao máximo a largura de banda disponível e proporcionar uma melhor qualidade de experiência. Essas mudanças na resolução também contribuem para as oscilações na taxa de bits por segundo enviados, pois alteram a quantidade de *pixels* que precisa ser codificada e enviada.

Ainda sobre a resolução, observa-se que ela sofre quatro alterações durante o experimento, passando por cinco resoluções diferentes até atingir a resolução máxima de 1280x720 *pixels*. Também é importante notar que a resolução inicial é a mesma, 320x180 *pixels*, para as cinco rodadas do experimento e que em todas elas, a resolução máxima é atingida antes dos 80 segundos do experimento.

Após a caracterização do comportamento das métricas e da plataforma de videoconferência em um cenário ideal, que servirá como referência para os demais experimentos, novos testes foram executados. A fim de compreender o impacto da comunicação com a plataforma Elos em um servidor remoto, executou-se um experimento similar ao anterior, porém sobre a arquitetura experimental da Figura 3.1b, que possibilita a conexão da rede local com um servidor remoto do Elos através da Internet. O gráfico representado na Figura 4.2 mostra o comportamento da plataforma Elos em um experimento com 5 rodadas, cada uma com duração de 150 segundos e sem quaisquer restrições de banda aplicadas.

Figura 4.2: Desempenho do Elos em ambiente real e sem restrições de banda



Fonte: O Autor

Analisando o gráfico em questão, é fácil perceber um comportamento muito semelhante ao do experimento anterior. O vídeo enviado também sofre alterações na sua resolução, passando pelos mesmos cinco patamares. A curva do parâmetro de quantização médio por quadro também apresenta um comportamento semelhante ao anterior, aumentando junto com a resolução, com picos nas trocas. Contudo, neste experimento,

nem todas as rodadas alcançam a resolução máxima de 1280x720 *pixels* antes dos 80s. Esse fato é indicado pelo sombreamento (desvio padrão) em torno das curvas que representam a moda por amostra da largura e altura do quadro de vídeo enviado. Em análise complementar, foi constatado que houve perdas e, por consequência, retransmissões durante esse experimento, o que explica a maior demora em alcançar a resolução máxima. Além disso, ainda analisando os desvios padrão, pode-se observar que neste experimento houve uma maior dispersão das resoluções sobre o último patamar (1280x720 *pixels*), em relação ao experimento "Canônico".

4.2 Entendendo o Comportamento de um Vídeo sob Restrições de Banda

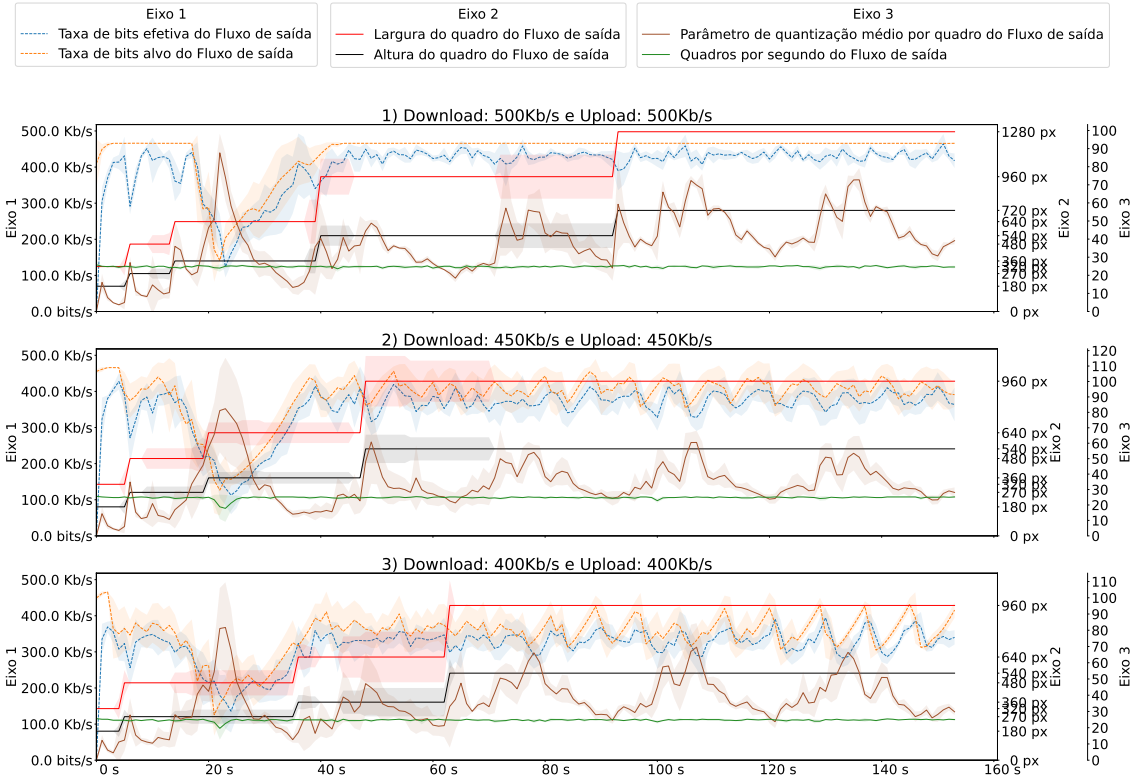
A plataforma Elos é amplamente utilizada por uma grande variedade de usuários em diferentes locais, com diferentes operadoras de Internet, bem como distintos planos e capacidades. Diante desse cenário, compreender o comportamento de um vídeo sob vários níveis de capacidade de rede é extremamente importante para tomar conhecimento do impacto desse tipo de limitação de rede na qualidade da experiência dos usuários. Tendo em vista esse aspecto, foi realizado um experimento na plataforma Elos, com seis níveis diferentes de capacidade de rede (150 Kbps, 200 Kbps, 300 Kbps, 400 Kbps, 450 Kbps e 500 Kbps, todas simétricas, *download* e *upload*), escolhidos levando em conta a especificação dos requisitos da plataforma, e foram realizadas cinco rodadas em cada nível. O experimento consiste de dois usuários em uma videoconferência, um deles compartilhando câmera na qualidade máxima e o outro participando como espectador. As Figuras 4.3 e 4.4 ilustram o comportamento da plataforma frente a esses diferentes níveis. Cada gráfico segue o modelo dos experimentos anteriores, mostrando as curvas da média da taxa efetiva de bits por segundo, da taxa alvo de bits por segundo, do parâmetro de quantização médio e dos quadros por segundo. As curvas da largura e da altura do quadro denotam a representação da moda dessas medidas. O sombreamento representa o desvio padrão em todas as curvas.

Durante a realização dos experimentos, foram identificados fenômenos não esperados nas curvas da taxa efetiva de bits por segundo e na taxa alvo de bits por segundo. Analisando os gráficos das Figuras 4.3 e 4.4, percebe-se que em todos experimentos, aproximadamente aos 20 segundos ocorre uma queda na taxa alvo de bits por segundo e, consequentemente, na taxa efetiva de bits por segundo. Esse comportamento se reflete, também, no outros indicadores. Ressalta-se que tais fenômenos não foram previstos ou

esperados, e sua natureza ainda não foi completamente compreendida. Cabe salientar que, por conta do tempo limitado disponível para a realização do trabalho, não foi possível realizar uma análise extensiva e profunda a cerca da causa desse comportamento. Porém, o que se constatou é que esse vale na curva da taxa alvo de bits por segundo acontece como consequência de RTP NACKs que são recebidos, requisitando retransmissões. Sendo assim, o algoritmo de congestionamento do transmissor recebe esses NACKs e reajusta a taxa alvo de bits por segundo para um valor inferior, dando origem a esses vales. O motivo específico do porquê esses RTP NACKs são enviados pelo servidor não foram investigados neste trabalho, ficando como sugestão de trabalhos futuros.

Embora tenham sido identificados fenômenos contraintuitivos nos indicadores dos experimentos, é importante ressaltar que os resultados obtidos são válidos. Isto se deve ao fato de que tais fenômenos ocorrem no início dos experimentos e a duração desses permite que a plataforma se recupere e alcance um estado estável.

Figura 4.3: Desempenho do Elos frente a restrições permanentes na capacidade da rede



Fonte: O Autor

Ao analisar o gráfico (1) da Figura 4.3, referente à limitação de banda de 500 Kbits/s, percebe-se um comportamento muito semelhante ao experimento de “Campo”, apresentado anteriormente. Isso ocorre porque o perfil de câmera com a maior qualidade utiliza, de fato, em torno de 450 Kbits/s, logo a plataforma “não percebe” a capacidade

limitada da rede. Sendo assim, a resolução do vídeo consegue alcançar consistentemente o máximo disponível de 1280x720 *pixels* em todas as rodadas, o que acontece, aproximadamente, até os 90 segundos do experimento.

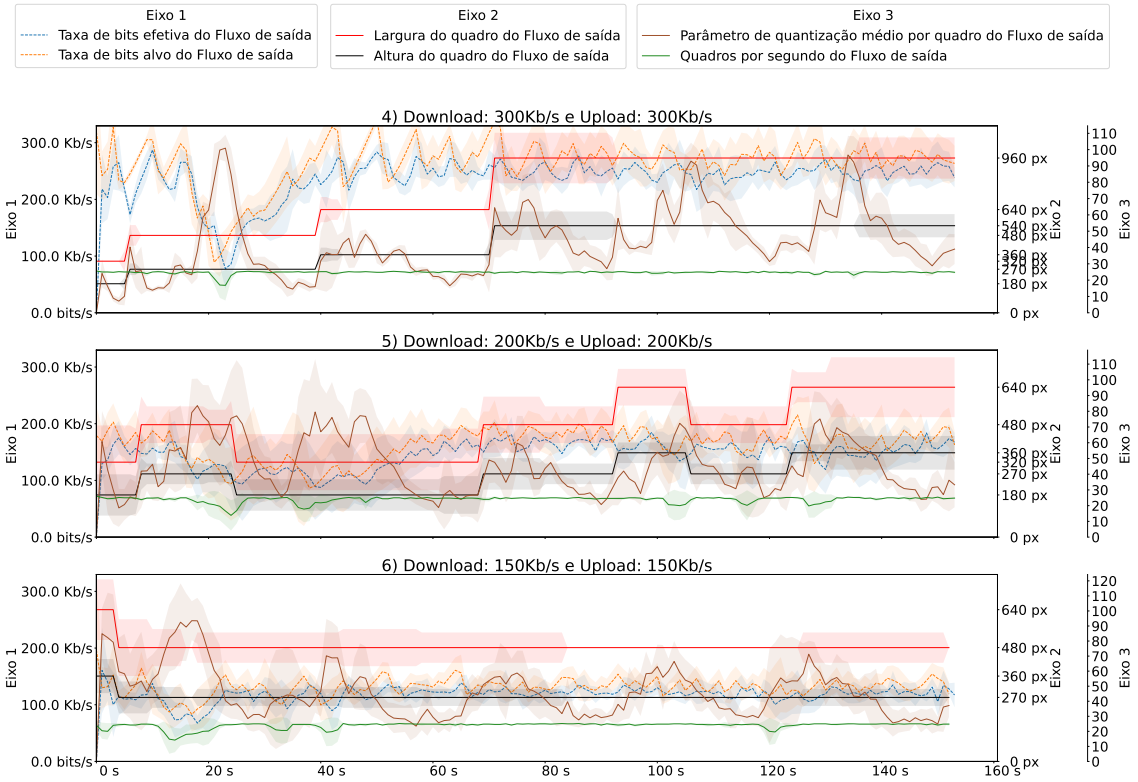
No gráfico (2), referente à limitação de banda de 450 Kbits/s, nota-se um comportamento diferente do visto até o momento para a taxa de bits alvo, a qual apresenta formato de dentes. Isso ocorre pois o vídeo "requer" mais banda do que a limitação imposta. Logo, a estimativa de banda do algoritmo de congestionamento vai sendo elevada até o momento em que perdas ou atrasos são reportados, quando então essa estimativa é ajustada. A repetição desse ciclo leva à formação de dentes, os quais aparecem também nos gráficos subsequentes. A resolução do vídeo também inicia em 320x180 *pixels*, porém atinge somente a resolução de 960x540 *pixels*, que não é a resolução máxima disponível, consequência direta da limitação de rede. Também, percebe-se que a curva do parâmetro de quantização médio assume valores mais elevados durante o período em que ambos experimentos (1) e (2) atingem a resolução mais elevada, quando em comparação com o gráfico da limitação de 500 Kbits/s, mesmo estando em resoluções diferentes.

Analisando o gráfico (3), referente à limitação de 400 Kbits/s, percebe-se fenômenos interessantes quando em comparação com o anterior. Primeiramente, neste gráfico pode-se observar que também há dentes na curva da taxa alvo de bits por segundo que, por serem mais acentuadas, se refletem em um padrão semelhante na taxa efetiva de bits por segundo. Além disso, percebe-se que, apesar do aumento na restrição, a resolução máxima atingida é a mesma do gráfico anterior, de 960x540 *pixels*. Contudo, a curva do parâmetro de quantização médio assume valores mais altos, o que significa uma menor qualidade, a fim de se acomodar à limitação de rede imposta.

O gráfico (4) da Figura 4.4 apresenta o comportamento da plataforma Elos sob a restrição de 300 Kbits/s. Ao analisá-lo, percebe-se um comportamento semelhante ao gráfico da restrição anterior, em que a taxa alvo de bits por segundo apresenta um formato de dentes, com topos próximos da restrição imposta. Além disso, a resolução também inicia em 320x180 *pixels* e termina o experimento em 960x540 *pixels*, também não atingindo o máximo disponível. Quando comparado ao gráfico anterior, percebe-se que os valores do parâmetro de quantização são mais elevados durante o período em que o vídeo alcança a sua maior resolução. Isso explica o fato de que mesmo com a rede mais restringida, ainda foi possível alcançar a mesma resolução que os dois experimentos anteriores.

Com uma restrição de 200 Kbits/s, o gráfico (5) mostra que a curva da resolução não apresenta um comportamento consistente de elevação, diferente dos gráficas anali-

Figura 4.4: Desempenho do Elos frente a restrições permanentes na capacidade da rede



Fonte: O Autor

sados até aqui. Em dois momentos, a resolução do vídeo sofre uma diminuição para se ajustar à taxa de bits alvo. Além disso, neste cenário, observa-se que a quantidade de quadros por segundo é decrementada de forma mais acentuada, a fim de aproximar a taxa efetiva de bits por segundo da curva da taxa de bits alvo. Um ponto importante a ser notado é que, no gráfico analisado, os desvios padrão são maiores em todas as curvas e se mantém durante toda a duração do experimento, indicando que em ambientes com capacidades de rede mais reduzidas, o comportamento da plataforma é mais dinâmico. Nesses casos, uma pequena variação em um dos aspectos da capacidade da rede representa uma grande perda sobre o todo.

Por último, no gráfico (6), que mostra o comportamento da plataforma sob a restrição de 150 Kbits/s, acontece algo curioso. A resolução do vídeo inicia em um patamar elevado, contrariando o que foi visto nos outros experimentos. O vídeo inicia com a resolução de 640x360 *pixels* e poucos segundos após reduz para 480x270 *pixels*, o que demonstra um comportamento atípico, visto que a rede está em seu estado mais restringido. Durante o restante do experimento, a resolução do vídeo tende a se manter no patamar de 480x270 *pixels* com um parâmetro de quantização médio elevado. Essa tendência pode indicar uma preferência por resoluções mais altas, que são equilibradas, em termos de

taxa efetiva de bits por segundo, por parâmetros de quantização mais altos, em relação a resoluções inferiores com parâmetro de quantização também mais baixo.

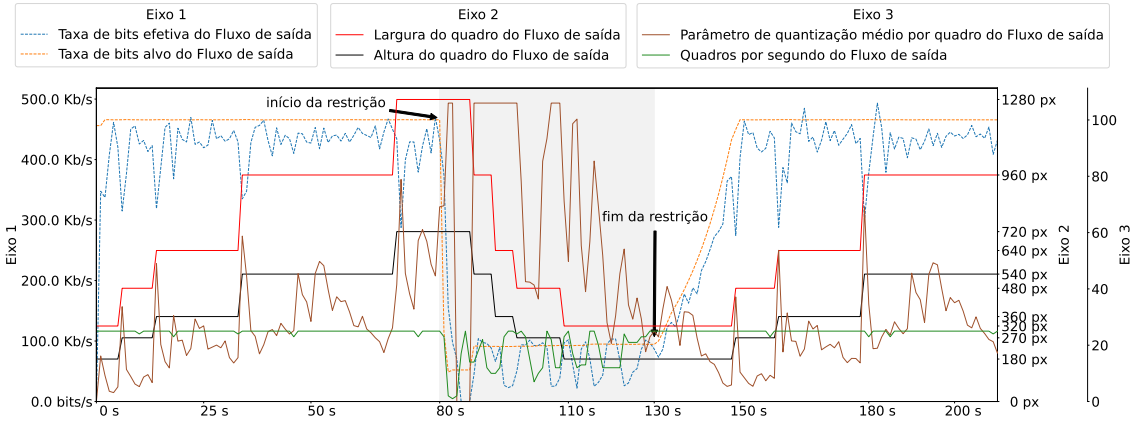
4.3 Desempenho do Elos em Oscilações de Rede

Após a análise do comportamento do Elos em condições ideais, assim como sob diferentes níveis de restrição de rede, parte-se para um experimento dinâmico. Este experimento tem como objetivo simular e entender o comportamento da plataforma de videoconferência Elos quando submetida a oscilações temporárias na capacidade da rede. Esse experimento é particularmente interessante porque se assemelha ao comportamento real de redes, onde oscilações temporárias na qualidade da conexão são mais comuns do que restrições fixas permanentes. Isso significa que, ao simular esse tipo de situação, pode-se obter informações-chave sobre como a plataforma se comporta em condições reais, o que é fundamental para o desenvolvimento de soluções mais eficazes para problemas relacionados ao desempenho da plataforma em ambientes de rede variáveis e imprevisíveis.

O gráfico da Figura 4.5 ilustra o comportamento de uma videoconferência quando submetida a restrições temporárias de rede. No experimento realizado sob o ambiente experimental 3.1b (“Real”), iniciou-se uma videoconferência sem restrições de rede, na qual observou-se um comportamento regular da plataforma, com a resolução do vídeo enviado escalando até o máximo disponível. Em seguida, introduziu-se restrições de rede temporárias, reduzindo sua capacidade a 200 Kbits/s entre os segundos 80 e 130 do experimento. Durante esse período de restrição, observa-se que a plataforma de videoconferência rapidamente reconhece a queda na banda disponível e reduz de forma acelerada a resolução do vídeo enviado com o objetivo de se adaptar à nova limitação. A plataforma leva cerca de 30 segundos para transicionar o vídeo de sua resolução máxima até a resolução mínima. Além disso, percebe-se que o parâmetro de quantização médio atinge o valor máximo possível em três momentos, a fim de acelerar a queda na taxa de bits por segundo. Da mesma forma, a taxa de quadros por segundo oscila bastante, chegando quase ao patamar zero assim que a restrição é aplicada. Essa adaptação permite que a conexão se mantenha estável e que a transmissão de vídeo continue, ainda que em uma qualidade reduzida. Vale ressaltar que a taxa efetiva de bits por segundo mantém-se abaixo da restrição imposta, o que pode indicar um comportamento cauteloso da plataforma na presença de quedas abruptas na capacidade da rede. A plataforma demonstrou, portanto, a capacidade de reagir rapidamente em resposta a restrições de rede temporárias, o que é um

aspecto importante para garantir a qualidade da experiência do usuário em ambientes de rede variáveis e imprevisíveis.

Figura 4.5: Desempenho do Elos frente à oscilação na capacidade da rede



Fonte: O Autor

Após o período de restrição de rede, observa-se que a plataforma aumenta gradualmente a estimativa de banda disponível, levando cerca de 20 segundos para reestabelecer a mesma taxa de transmissão que havia antes da aplicação da restrição. No entanto, a resolução do vídeo transmitido permanece reduzida por um período mais prolongado e não alcança a resolução anterior até o final do experimento. Essa observação sugere que a plataforma é mais conservadora ao aumentar a resolução do vídeo após um período de restrição, possivelmente para evitar novas oscilações ou interrupções na conexão. Essa abordagem pode garantir uma experiência mais estável para o usuário, ainda que com resolução reduzida em alguns momentos, em detrimento de um aumento mais rápido na resolução, que poderia levar a uma instabilidade na conexão.

4.4 Desafios na Realização do Trabalho

Durante o desenvolvimento deste trabalho, enfrentou-se uma série de desafios que afetaram o seu andamento. Um dos mais significativos foi a ocorrência e a investigação dos fenômenos não esperados que aparecem nos experimentos da Seção 4.2. As capturas de pacotes realizadas com a ferramenta tshark mostraram-se úteis somente para a análise de cabeçalhos de pacotes RTP, mas não para destrinchar conteúdo dos pacotes, como relatórios RTCP. Isso acontece porque o protocolo WebRTC prevê, como explicado na Seção 2.2, a utilização de um perfil RTP criptografado. Sendo assim, a correta investigação da causa dos vales mencionados anteriormente depende de acesso aos servidores da plata-

forma Elos, para captura de informações internas. Tal investigação não foi desenvolvida neste trabalho, por limitação de tempo, e fica a cargo de trabalhos futuros abordá-la.

Além disso, durante o andamento do trabalho, ocorreu uma alteração do formato do arquivo de métricas WebRTC da página interna do Google Chrome, o que demandou a reescrita quase completa do *script* de geração dos gráficos com base nos dados coletados. Embora essa tarefa tenha exigido muito tempo e esforço, foi essencial para manter a integridade e a confiabilidade dos dados coletados, e garantir que as conclusões do estudo fossem precisas e confiáveis.

Por último, outro desafio encontrado foi a grande variedade de implementações para os mecanismos e ferramentas previstos no protocolo WebRTC, as quais, não raramente, fogem às especificações. Ademais, muitas dessas ferramentas não possuem documentação, o que obriga o interessado a ler o código-fonte (quando disponível) para um entendimento mais detalhado do seu funcionamento.

5 CONCLUSÃO

A forma como as pessoas se comunicam, trabalham e estudam mudou significativamente nos últimos anos. Com a crescente adoção do trabalho remoto e a necessidade de distanciamento social, plataformas de videoconferência se tornaram uma opção popular para a comunicação em todo o mundo. Como resultado, o foco sobre a qualidade da experiência do usuário em videoconferências tem aumentado consideravelmente. É essencial entender como essas plataformas desempenham quando seus usuários estão sob condições degradadas nos parâmetros de rede, a fim de desenvolver e refinar esses mecanismos de adaptação.

Neste trabalho, criou-se um arcabouço a fim de caracterizar o comportamento da plataforma de videoconferência Elos sob diferentes condições de rede. Ao simular ambientes com degradações nos parâmetros de *download* e *upload* da rede, constatou-se que a plataforma de videoconferência Elos é capaz de se adaptar a condições severas de rede, tanto permanentes quanto temporárias, de forma rápida. Esses resultados indicam que a interação eficiente entre os múltiplos mecanismos, implementações e protocolos, no contexto de videoconferência utilizando o protocolo WebRTC, é bem-sucedida. No entanto, é importante destacar que a interface de múltiplas implementações acrescenta uma complexidade elevada ao funcionamento deste protocolo WebRTC e que, não raramente, as mesmas extrapolam os padrões, resultando em possíveis comportamentos peculiares e dificuldades operacionais.

Por fim, o trabalho desenvolvido contribui para o desenvolvimento de outros estudos que possam estimar QoE com base em um entendimento mais refinado dos efeitos da internet em QoS. Além disso, apresenta encaminhamentos valiosos para questões sobre o comportamento de plataformas de videoconferência sob diferentes cenários de rede. No entanto, também gera muitas outras questões e abre espaço para trabalhos futuros. Ainda há muito a ser explorado em relação aos impactos de diferentes parâmetros de rede, como *jitter* e perda de pacotes, na qualidade da experiência do usuário em videoconferências. Uma vez tendo o entendimento do impacto dessas degradações nos parâmetros de rede, abre-se espaço para o desenvolvimento de um sistema de monitoramento em tempo real da Qualidade de Serviço e mensuração da Qualidade de Experiência.

REFERÊNCIAS

AMMAR, D. et al. Video qoe killer and performance statistics in webrtc-based video communication. In: **2016 IEEE Sixth International Conference on Communications and Electronics (ICCE)**. [S.l.: s.n.], 2016. p. 429–436.

BAUGHER, M. et al. The secure real-time transport protocol (srtp). **www.rfc-editor.org**, 03 2004. Available from Internet: <https://www.rfc-editor.org/rfc/rfc3711>.

BIGBLUEBUTTON.ORG. **BigBlueButton - Web Conferencing System Designed For Online Learning**. 2019. Available from Internet: <https://bigbluebutton.org/>.

ELOS.VC. **Elos • Além da Videoconferência, incentivando a colaboração.** Available from Internet: <https://elos.vc>.

GARCíA, B. et al. Understanding and estimating quality of experience in webrtc applications. **Computing**, v. 101, 11 2019.

JANSEN, B. et al. Performance evaluation of webrtc-based video conferencing. **SIGMETRICS Perform. Eval. Rev.**, Association for Computing Machinery, New York, NY, USA, v. 45, n. 3, p. 56–68, mar 2018. ISSN 0163-5999. Available from Internet: <https://doi.org/10.1145/3199524.3199534>.

MACMILLAN, K. et al. Measuring the performance and network utilization of popular video conferencing applications. In: **Proceedings of the 21st ACM Internet Measurement Conference**. New York, NY, USA: Association for Computing Machinery, 2021. (IMC '21), p. 229–244. ISBN 9781450391290. Available from Internet: <https://doi.org/10.1145/3487552.3487842>.

SCHULZRINNE, H.; CASNER, S. Rtp profile for audio and video conferences with minimal control. **www.rfc-editor.org**, 07 2003. Available from Internet: <https://www.rfc-editor.org/rfc/rfc3551.html>.

TRESTIAN, R.; COMSA, I. S.; TUYSUZ, M. Seamless multimedia delivery within a heterogeneous wireless networks environment: Are we there yet? **IEEE Communications Surveys Tutorials**, PP, p. 1–1, 01 2018.