CENTRO UNIVERSITÁRIO DE BRASÍLIA -UniCEUB
CURSO DE ENGENHARIA DE COMPUTAÇÃO

Rodrigo Otávio Gonçalves de Oliveira

Utilização de QoS para priorização de tráfego de voz em links WAN

Orientador: Prof. Marco Antônio Araujo

Brasília
Junho, 2010

Rodrigo Otávio Gonçalves de Oliveira

Utilização de QoS para priorização de tráfego de voz em links WAN

Trabalho apresentado ao Centro Universitário de Brasília (UniCEUB) como pré-requisito para a obtenção de Certificado de Conclusão de Curso de Engenharia de Computação.
Orientador: Prof. Marco Antônio Araújo.

Brasília
Junho, 2010

Utilização de QoS para priorização de tráfego de voz em links WAN

Trabalho apresentado ao Centro Universitário de Brasília (UniCEUB) como pré-requisito para a obtenção de Certificado de Conclusão de Curso de Engenharia de Computação.
Orientador: Prof. Marco Antônio Araújo.

Este Trabalho foi julgado adequado para a obtenção do Título de Engenheiro de Computação, e aprovado em sua forma final pela Faculdade de Tecnologia e Ciências Sociais Aplicadas - FATECS.

_____

Prof. Abiezer Amarilia Fernandez
Coordenador do Curso

Banca Examinadora:

_____

Prof. Marco Antônio Araujo.
Orientador

_____

Prof. Luis Cláudio
UniCEUB

_____

Prof. Paldês
UniCEUB

_____

Prof. nome, titulação.
Instituição

DEDICATÓRIA

A minha esposa, Tatiane Lopes Ferreira de Oliveira, e filho, que entenderam que esse momento era especial, importante e único e me apoiaram incondicionalmente nessa tarefa.

Ao meu pai, Joel Ferreira de Oliveira, figura na qual pude sempre contar e que foi minha base para que eu hoje atinja esse objetivo.

Ao meu irmão, Marco Aurélio Gonçalves de Oliveira, fonte constante de inspiração e modelo de honestidade e caráter.

A minha mãe, Ieda Gonçalves de Oliveira, que mesmo não estando mais presente, sempre foi peça fundamental em minhas escolhas de sucesso.

AGRADECIMENTOS

Ao meu orientador, Prof. Marco Antônio, pelo constante apoio e incentivo, essenciais para o desenvolvimento desse trabalho.

Ao Prof. Javier Dobaldia, do curso de Engenharia da Computação, verdadeiro co-orientador deste trabalho, que sem dúvidas, contibuiu de forma significativa no desenvolver das minhas atividades.

Ao Prof. Abiezer Amarilia Fernandez, do curso e Engenharia da Computação, que sempre me apoio nas escolhas e dificuldades encontradas nessa longa jornada.

A Huawei Technologies e meus colegas de trabalho, pelo incentivo e apoio, contribuindo com equipamentos e conhecimento vitais para a conclusão dessa pesquisa.

Aos companheiros de UniCEUB, que juntos enfrentaram todas as dificuldades pertinentes à carreira de Engenheiro da Computação.

A todos, meus sinceros agradecimentos.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

RESUMO

O trabalho aqui apresentado, pretende apresentar modelos de customização e aplicação de QoS (Quality of Service), utilizando políticas escolhidas e de acordo com suas especificações e aplicabilidade nos cenários simulados, permitindo a implementação mais eficiente para os diversos tráfegos tempo real, como voz, e sua interação com o tráfego de dados. Aplicando solução visando a priorização de tráfico de pacotes de voz RTP (Real Time Protocol) em links WAN. (Wide Area Networks).

Para isso serão analisados três cenários que simulam condições semelhantes às encontradas em grandes redes, efetuando comparações entre as tecnologias utilizadas visando escolher a mais indicada para utilização.

ABSTRACT

This work here presented, has as objective to present a definition and applicability of the QoS, using chosen politics according to specification and applicability into the simulated scenarios, permiting an efficient implementation for the several kinds of real time traffic, as voice, and the interaction with data traffic. Appling the solution on the priorization of the voice real time traffic RTP (Real-time Transport Protocol) in WAN (Wide Area Networks).

CAPÍTULO 1 – INTRODUÇÃO:

1.1 – Motivação

Os ambientes corporativos utilizam o VoIP (Voice over Internet Protocol) para facilitar a comunicação entre funcionários e pessoas, além de ser uma solução de baixo custo e que utiliza estrutura de dados existente, não há necessidade de se implantar uma rede independente para esse serviço, podendo utilizar a estrutura de dados existente. Porém, não só serviços de voz utilizam as redes de dados (redes IP). O crescimento do número de aplicações e usuários de rede e, consequentemente, a demanda de aumento da largura de banda que suporte a necessidade.

Tamanha demanda na disponibilidade dos recursos de rede causam constantes congestionamentos no tráfego dos dados, principalmente nos links que interligam LANs (Local Área Network), links chamados de WAN (Wide Área Network), onde o "gargalo" dos dados acontece.

Para serviços de voz, congestionamentos são vistos como os vilões da qualidade de comunicação, uma vez que os pacotes de voz são extremamente sensíveis aos atrasos, as perdas e a retransmissão de pacotes. Qualquer um desses problemas causa impacto na qualidade da voz e até indisponibilidade do serviço.

Para solucionar esse problema, torna-se necessário aplicar mecanismos que visam atuar nesses "gargalos", identificando os pacotes de voz e os encaminhando com a devida prioridade que garantirá a qualidade da comunicação fim-a-fim.

1.2 – Objetivo geral do trabalho

O objetivo deste trabalho de conclusão de curso é avaliar a utilização de técnicas de classificação e tratamento de tráfego IP em links WAN. O intuito é priorizar determinado tipo de tráfego, no caso, o tráfego de pacotes de voz, coletar e comparar dados que julguem a eficácia dos métodos aplicados e ofereçam embasamento para escolha de melhor método para os cenários apresentados.

1.3 – Objetivos específicos

Apresentar técnicas de classificação de tráfego e métodos de priorização de tráfego IP, aplicados em experimento prático, simulando cenários reais de congestionamento em links WAN, onde existam tráfegos de voz e outros tráfegos comumente encontrados em redes corporativas. Contribuir para a qualidade da voz mesmo em links extremamente afetados.

O uso do VoIP é cada vez mais comum. Os links de comunicação WAN que apresentam congestionamento necessitam de mecanismos que garantam a qualidade de determinados serviços, como a voz.

Não é viável a implantação de serviços de voz sobre redes IP que não apresentem níveis mínimos de qualidade e que a comunicação fim-a-fim não seja garantida. O princípio básico da comunicação é que ambas as partes possam deixar clara a mensagem a ser transmitida, de modo a entender e ser entendido.

Nesse momento, o uso do QoS é fundamental. Esse é o mecanismo que, em momentos de degradação da disponibilidade dos serviços de rede, atuará como mecanisno de priorização aos interlocutores para que se comuniquem e que os pacotes de voz sejam tratados de forma diferenciada.

1.4 – Escopo do trabalho

Apresentar três cenários que simulem situações reais. Esses cenários serão apresentados em detalhe ao longo da dissertação, onde serão detalhados a topologia física e lógica, o endereçamento, as rotas, as configurações, as interfaces utilizadas e a metodologia de testes.

Cenário 1: Cenário base (simulação do problema apresentado):

Demonstrar o impacto no serviço de voz quando o link de comunicação (link WAN – SERIAL) apresentar congestionamento (o congestionamento será gerado), registrar o impacto da qualidade dos serviços de voz.

Cenário 2: PQ (Priority Queue)

Aplicar no cenário base técnica 1, classificação e tratamento de tráfego, QoS (Quality of Service) configurada e customizada, visando priorizar tráfego de voz, coletar dados para efeito de comparação com cenário base.

Cenário 3: WFQ (Weighted Fair Queue)

Aplicar no cenário base técnica 2, classificação e tratamento de tráfego QoS (Quality of Service) configurada e customizada, visando priorizar tráfego de voz, coletar dados para efeito de comparação com cenário base e cenário 1.

Apresentar análise comparativa dos cenários e técnicas utilizadas e eleição de melhor técnica para solução do problema apresentado.

1.5 – Resultados esperados

Espera-se que seja identificada a melhor forma de priorização do tráfego de voz em redes IP que apresentem links de comunicação com níveis altos de congestionamento e degradação da disponibilidade de recursos, além de análise de dados que comprovem a eficácia do método eleito como ideal dentre os testados para essa função.

1.6 – Estrutura do trabalho

O trabalho é basicamente dividido em quatro partes:

Apresentação do problema: nesta primeira parte, o problema será apresentado, bem como as suas causas e principalmente as suas conseqüências e os cenários mais frequentes onde o problema é encontrado.

Embasamento teórico: nesta parte serão apresentados os fundamentos técnicos e teóricos necessários para a implantação e configuração do experimento prático.

Experimento prático: simulação de cenário em bancada, configuração de equipamentos utilizados baseados nas técnicas apresentadas nos fundamentos técnicos.

Coleta e análise de dados: os dados extraídos do experimento técnico serão avaliados e analisados, culminando com a definição do melhor método dentre os testados para a solução do problema proposto.

CAPÍTULO 2 – APRESENTAÇÃO DO PROBLEMA

2.1– Considerações iniciais

Na introdução, são apresentados, brevemente, os cenários que serão simulados no experimento prático visando trazer para o ambiente de laboratório uma reprodução em menor escala do problema a ser tratado.

Para um correto entendimento das causas e consequências do congestionamento, o cenário padrão é apresentado sob diversos pontos de vista, possibilitando uma visão em detalhes.

Seguem as topologias física e lógica onde o problema de congestionamento se apresenta com frequência.

2.2 – Topologias – Cenários típicos onde o problema é encontrado (WAN)

Na figura 1 um exemplo de topologia onde o problema poderá ocorrer. O cenário é básico, uma empresa com mais de um escritório geograficamente distantes, porém interligados por redes IP.

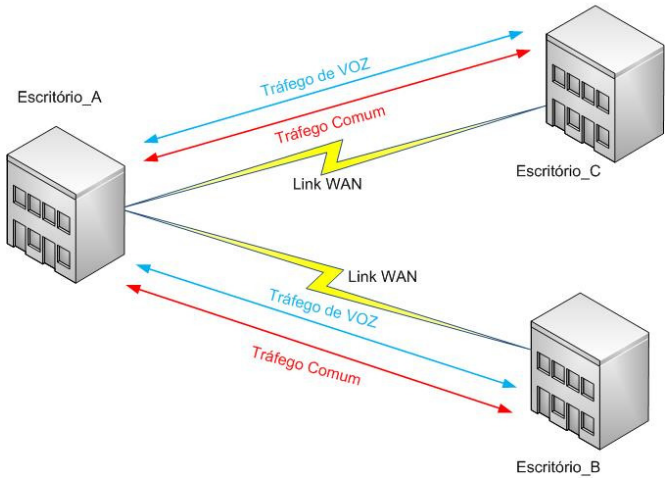


Figura 1 – Topologia Física – Visão Geral 1.

Nota-se que todo o tráfego de comunicação entre os escritórios A, B e C passam pelos links WAN, tráfegos esses que podem ser dados, voz, vídeo entre outros.

Dentro desses escritórios encontram-se computadores, notebooks, impressoras, servidores, telefones IP (utilizam VoIP), e, quando qualquer um desses dispositivos de rede se comunica com outro escritório, utiliza o link WAN.

É possível enumerar diversas situações em que a comunicação entre escritórios ocorre, como por exemplo:

- Troca de dados entre os escritórios A, B e C.
- O servidor de backup da companhia se encontra no escritório A, portanto, quando os escritórios B e C acessam o servidor haverá troca de dados.
- O servidor WEB da intranet e de e-mails da empresa se encontra em um dos três escritórios, e para acessá-los, os demais utilizam do link WAN.
- Chamadas VoIP entre os escritórios.

Observa-se, na Figura 2, o detalhamento dos escritórios e os diversos elementos de rede que compartilham o uso do link WAN na comunicação entre os pontos A,B e C.

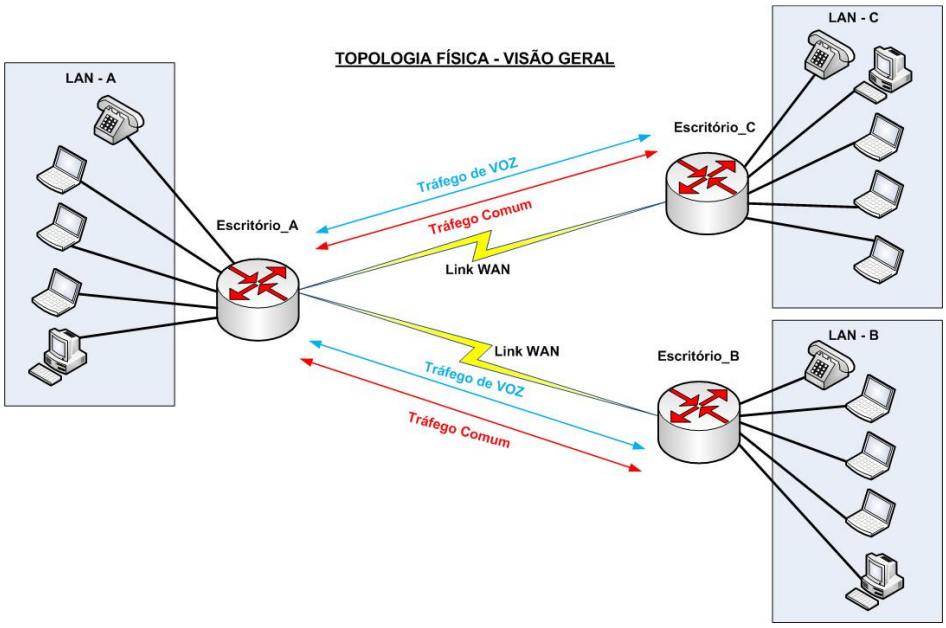


Figura 2 – Topologia Física – Visão Geral 2.

Detalhando ainda mais a visão de como o tráfego nesse cenário acontece, toma-se o roteador do escritório A, como exemplo. Por se encontrar na fronteira entre a rede A e as demais, ele é o responsável por receber todo o tráfego e encaminhá-lo para as demais redes.

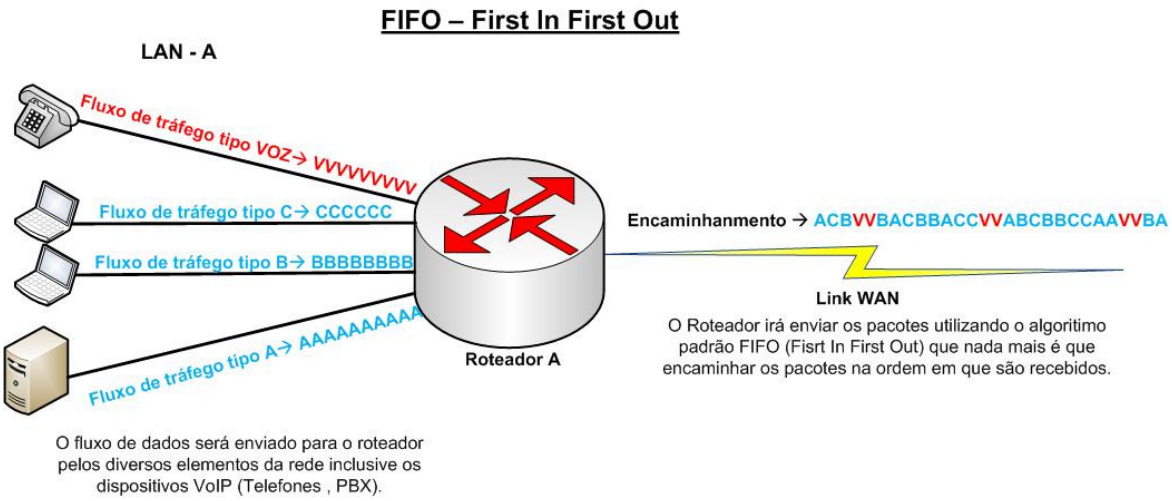


Figura 3 – FIFO – First In First Out.

Observa-se na Figura 3, que todos os elementos da rede local A enviam pacotes para o roteador A ao mesmo tempo. Por sua vez, o roteador A os encaminha adotando a política do FIFO (Fist In First Out), isto é, encaminha os pacotes na ordem em que são recebidos.

Dependendo do hardware e da disponibilidade da rede local, alguns elementos têm maior capacidade de envio de pacotes e, por sua vez, serão encaminhados de acordo com a sua chegada.

É possível observar também o tráfego de voz, que compete com os demais tráfegos para serem encaminhados. Nessa situação o roteador encara todos os tipos de tráfico como se fossem os mesmos.

Inevitavelmente, em um dado momento esse link de comunicação irá apresentar dificuldades para encaminhar todo o tráfego recebido das redes locais. Dificuldades que acontecem devido aos seguintes fatores:

- Os links da rede local têm maior velocidade que o link WAN.

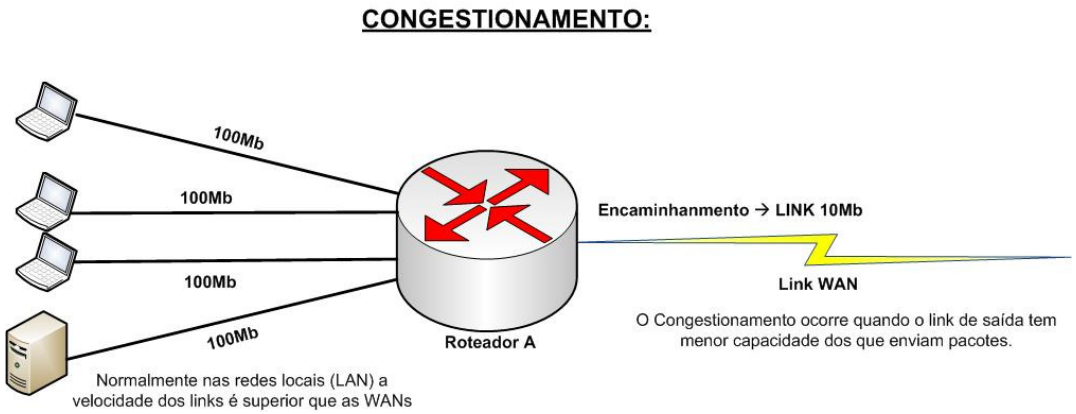


Figura 4 - Congestionamento

- Os links da rede local e da rede WAN possuem a mesma capacidade, porém o número de dispositivos e de usuários da rede local esgota a capacidade do link de envio, como mostrado na Figura 4.

Quando situações como essas acontecem, a rede pode apresentar alguns desses problemas como:

- Atrasos (Delay): os pacotes que em condições normais são encaminhados com uma determinada velocidade, começam a apresentar atrasos para serem encaminhados.

- Retransmissão: devido ao longo tempo para serem encaminhados, acontece um timeout e torna-se necessária a retransmissão.

- Descarte: quando o link de envio está muito congestionado, o roteador não consegue armazenar em fila todos os pacotes recebidos, assim, com o link congestionado e a fila cheia, não há alternativa a não ser descartar os pacotes recebidos por último.

2.3 – Tráfego de voz – Problema

Quando se trata de tráfego de voz, alguns itens não podem ser ignorados. Por se tratar de uma aplicação em tempo real, atrasos, perdas e retransmissões dos pacotes causam um grande impacto no serviço.

Portanto, a transmissão dos pacotes de voz por links WAN congestionados e que utiliza a política FIFO, inviabiliza a utilização de serviços de voz.

As consequências são:

- Picotamento da voz durante a chamada;
- Má qualidade durante a comunicação;
- Queda de chamada;
- Indisponibilidade do serviço.

Esse problema será simulado no cenário 1.

2.4 – Soluções atuais

Para sanar os problemas de congestionamento, podem-se adotar algumas soluções:

- Aumento da largura de banda dos links WAN: uma das soluções que poderá ajudar a solucionar o problema dos congestionamentos é o aumento da capacidade dos links WAN, que tornará possível aumentar a quantidade de pacotes a serem encaminhados.

- Definição de horários específicos para determinadas atividades na rede: outra possível solução é definir horários específicos para certas atividades que consomem largura de banda. Exemplo: fazer os backups dos servidores durante a madrugada para evitar o congestionamento do link WAN no horário comercial.

2.5 – Desvantagens das soluções atuais

Ambas as soluções propostas acima têm deficiências. O fato de aumentarmos a capacidade do link WAN, além de gerar custo, é uma solução paliativa. Inevitavelmente, dentro de um período de tempo os usuários e serviços da rede irão consumir e necessitar de mais disponibilidade de banda.

Já a definição de horário para determinadas atividades ajuda a desafogar o link, porém, atualmente, mesmo com essas medidas, apenas o tráfego do dia a dia no horário comercial já é suficiente para congestionar os links.

2.6 - Apresentação da solução demonstrada (QoS)

A solução adotada nos testes é o QoS (Quality of Service).

Resumidamente o QoS irá:

- Classificar e encaminhar o tráfego de acordo com especificação pré-definida.
- Identificar o que está chegando ao roteador.

- Encaminhar o tráfego baseado em regras definidas.

- Mandar pacotes de dados com a prioridade definida para dados e enviar os pacotes de voz com a prioridade definida para voz.

Figura 5 – QoS – Quality of Service 1

Em um primeiro momento o roteador irá classificar o tráfego.

Após a classificação e dependendo do método de enfileiramento e priorização, o tráfego é encaminhado de acordo com especificação pré-definida, Figura 5.

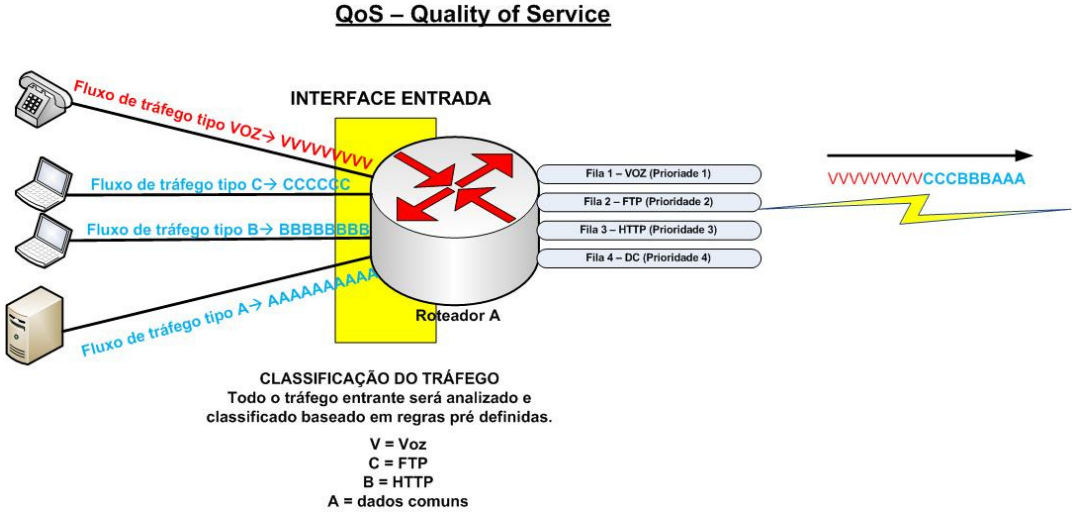


Figura 6 – QoS – Quality of Service 2

Pode-se verificar, na figura 6, que a prioridade do envio do tráfego de voz é a maior, garantindo, assim, que todos os pacotes de voz sejam encaminhados sem atrasos, mantendo a qualidade das chamadas.

Os benefícios são:

- Não há custo, se o roteador de borda tiver suporte à tecnologia, será necessário apenas configurar o equipamento.
- Pode-se manter a capacidade do link WAN existente.
- Customização: é adaptável a diversas necessidades, se a prioridade for passar os pacotes de voz, então se customiza a configuração para tal. Se em outra rede a prioridade é o tráfego do BACKUP, então este será priorizado.

Para melhor compreensão da situação proposta, faz-se alusão a uma situação hipotética. Um grave problema em uma determinada cidade, demanda a evacuação em massa de toda a população. Porém, a única saída disponível é uma pequena ponte (caso das LANs tentando enviar todo seu tráfego pelas WANs).

Obviamente, não será possível que todas as pessoas saiam pela ponte ao mesmo tempo. Forma-se, então, uma fila na entrada da ponte (congestionamento) e as pessoas a acessam conforme chegam na sua entrada (FIFO). Porém, além de mulheres e crianças, existem especialistas que precisam chegar do outro lado com prioridade. Faz-se uma força tarefa e verifica-se, na população que vai chegando à ponte, quem são crianças, quem são mulheres e quem são os especialistas (classificação). Eles são colocados em filas especiais e terão prioridade ao atravessar a ponte. Isso é o que o QoS faz com os pacotes nas redes IP.

CAPÍTULO 3 – REFERÊNCIAL TEÓRICO

3.1 - VoIP – Voice over Internet Protocol

3.1.1 - Descrição geral

O VoIP (Voice over IP) possui diversas aplicações e uma delas é comumente conhecida como telefone IP. A aplicação do VoIP em roteadores tornou possível para uma rede IP transportar serviços de voz, como, por exemplo, o serviço simples de telefonia. O VoIP é implementado através da transmissão dos pacotes de voz (Site: Huawei, 2009).

Redes IP por sua vez são baseadas em troca de pacotes. Os dados de voz devem ser encapsulados em quadros após o processamento no DSP (Digital Signal Processor) e armazenados em pacotes, depois disso, encaminhados via uma rede IP. Em essência, o VoIP disponibiliza serviço de voz através de software. Todavia, sua implementação requer suporte a módulos de voz nos roteadores (HUAWEI, 2009. **Operatio Manual (V3.53_01)).**

3.1.2 – Estrutura básica

No serviço simples de telefonia, todas as funções desde a parte que está originando a chamada até a parte que recebe a chamada são implementadas pelas PSTN (Public Switched Telephone Network). O VoIP, por sua vez, funciona de maneira diferente.

Quando um roteador disponibiliza funções de um gateway de voz, ele irá transformar a voz de um tradicional circuito PSTN em pacotes IP. Durante essa transformação o roteador mapeia números telefônicos tradicionais em endereços IP, que são facilmente reconhecidos em redes de comutação de pacotes, basicamente mapeando números de telefones em endereços IP (HUAWEI, 2009. **Command Manual (V3.53_01)).**

Figura 7 – Estrutura Básica

Os GW(Gateways) VoIP disponibilizam a interface entre a rede IP e a PSTN/ISDN (Public Switched Telephone Network/Integrated Services Digital Network). Dessa forma o usuário pode estar conectado ao gateway VoIP via PSTN, onde os sinais analógicos serão convertidos em sinais digitais e comprimidos em pacotes de voz que podem ser transmitidos através de uma rede IP. Esses pacotes serão encaminhados para o gateway VoIP do lado de destino da chamada, onde os pacotes de voz serão regenerados em sinais analógicos reconhecíveis e transmitidos ao telefone chamado via PSTN. Esse é o processo de comunicação completo telefone – telefone. Nas redes VoIP atuais os Gate Keepers (GK) são necessários para um melhor roteamento e controle de acesso.

3.1.3 - Fluxo básico de chamadas VoIP

A sequência abaixo descreve basicamente o fluxo de chamadas no VoIP.

O usuário retira o fone do gancho, e a placa de voz detecta a retirada de gancho (off-hook) em tempo real.

A placa de voz envia um sinal de off-hook para o processamento de sinais de VoIP no roteador.

O processador de sinais VoIP gera uma tom de discagem.

O usuário escuta o tom de discagem enviado pela sessão de aplicação do VoIP e inicia a digitação. Essa operação deve ser feita antes do fim do prazo do tom de discagem.

A sessão de aplicação do VoIP coleta o número discado pelo usuário.

Durante o processo de coleta do número, a sessão de aplicação compara esse número com os configurados como possíveis destinos em tempo real.

Após a identificação do número com um dos modos de destino, o número do telefone será mapeado no gateway de voz. O gateway de voz é referenciado com o gateway chamado e está diretamente conectado ao telefone de destino ou PBX de destino.

O gateway que origina a chamada de voz inicia uma nova chamada para o gateway de destino, baseado em protocolos H323 através da rede IP, e define um canal lógico para o recebimento e transmissão de dados de voz (HUAWEI, 2009. **Command Manual (V3.53_01);** TANENBAUM, 2003; COLCHER *et.al.,* 2005)

O gateway de destino da chamada recebe uma chamada H323 do lado IP, em seguida pesquisa pelo telefone de destino de acordo com identificação prévia. Após isso a chamada será encaminhada diretamente para o PBX ou PSNT.

Durante a conexão da chamada, ambos os lados da chamada negociam o método de codificação e de decodificação para a chamada, os dados de voz são transferidos utilizando RTP.

Canais de voz RTP são usados para transmitir sinais de controle durante a conexão da chamada, além de outros sinais que poderão ser transmitidos na banda através da rede IP.

No momento que qualquer um dos lados da chamada finaliza a ligação, a aplicação de sessão do VoIP finaliza a sessão e aguarda por uma nova chamada.

3.1.4 – VoIP QoS

Como sugestão para os experimentos apresentados nesse trabalho e partindo do princípio que o serviço de voz é extremamente sensível a variações na rede, alguns métodos de QoS, como, por exemplo, PQ, CQ e WFQ, poderão ser adotados para garantir largura de banda adequada para a transmissão de voz, além do mecanismo de CAR que pode ser adotado para classificação e monitoramento.

3.1.5 - Interfaces de voz

Aplicações de VoIP em roteadores torna possível que serviços de voz sejam transportados em redes IP, como,por exemplo, serviços simples de telefonia. O DSP (Digital Signal Processor) encapsula os sinais de voz em quadros e os armazena em pacotes para transmiti-los em redes IP. Nas redes VoIP, interfaces de voz são necessárias para conexão com dispositivos PSTN, PBX e aparelhos de telefone.

As interfaces de voz em roteadores lidam com funções de camada física entre o roteador e a PSTN, incluindo disponibilizar energia para os telefones analógicos, detecção de retirada de gancho, recebimento e envio de chamadas de voz analógicas e digitais, e recebimento e transmissão de números digitados para o roteamento de chamadas. Abaixo as placas utilizadas no experimento apresentado neste trabalho (HUAWEI, 2009. **Command Manual (V3.53_01);** TANENBAUM, 2003; COLCHER *et.al.,* 2005).

FXS (Foreign eXchange Station): Uma linha de voz analógica, normalmente conhecida como porta simples de serviço telefônico, usualmente conectada a terminais telefônicos ou aparelhos telefônicos analógicos simples,provendo corrente de chamada, voltagem e tom de discagem.

A placa FXS interage com a sinalização de telefones comuns, aparelhos de fax ou PBX através do cabo de telefone e conector RJ11.

3.1.6 – RTP (Real-time Transport Protocol)

O RTP (Real-time Transport Protocol) é um protocolo que oferece funções de transporte de rede fim a fim direcionadas para aplicações que transmitem fluxos de dados em tempo real, como áudio, vídeo e dados de simulações, através de serviços de rede unicast e multicast. Esse protocolo não trata da reserva de recursos e não garante qualidade de serviço (QoS) para serviços de tempo real , mas atribui esses requisitos para serviços tais como os oferecidos pelas arquiteturas IntServ e DiffServ.

O transporte de dados é complementado por um protocolo de controle (RTCP- Real-time Transport Control Protocol) para permitir o monitoramento da entrega de dados de forma escalável em redes multicast e oferecer funcionalidades mínimas de controle e identificação. Esse protocolo é baseado na transmissão periódica de pacotes de controle para todos os participantes de uma sessão RTP – um conjunto de transmissores e receptores que trocam entre si fluxos de dados relacionados a uma aplicação ou serviço específico. O RTCP utiliza o mesmo mecanismo de distribuição definido para os pacotes que compõem os fluxos de dados das aplicações. Ambos os protocolos, RTP e RTCP, constituem-se elementos centrais da maioria (senão todas) das arquiteturas e serviços de VoIP.

Tanto o RTP quanto o RTCP foram elaborados para serem independentes da camada de transporte subjacente e das camadas de rede. O protocolo subjacente deve prover multiplexação adequada para os pacotes de dados e controle. Em redes IP, é tipicamente utilizada a multiplexação fornecida pelas portas UDP.

Em Friedman (2003), foi definida a incorporação do pacote Extended Report (XR) criando o RTCP XR (RTP Control Protocol Extenden Reports), para melhorar o controle de informações estatísticas que era oferecido originalmente pelo RTCP (HUAWEI, 2009. **Command Manual (V3.53_01);** TANENBAUM, 2003; COLCHER *et.al.,* 2005).

Os pacotes XR são compostos por blocos de informação, e sete tipos de blocos são definidos inicialmente no RTCP XR. O propósito é complementar as seis estatísticas contidas nos blocos informativos usados pelos pacotes do Sender Report e Receiver Report do RTCP.

Contudo, as aplicações como VoIP podem requerer outras informações estatísticas não tratadas em Friedman (2003). Assim outros tipos de blocos informativos podem ser definidos, de acordo, com o arcabouço para pacotes XR definido no RTCP XR.

Embora o RTP seja um protocolo de transmissão otimizado para aplicações que transmitem fluxos de dados em tempo real, após o encapsulamento do RTP no UDP/IP, o cabeçalho total corresponde a 40 octetos (12 para o RTP, 8 para o UDP e 20 para o IP).

Para minimizar o problema do overhead de cabeçalho do RTP em linhas de transmissão de baixa velocidade, Casner (1999) descreveu um método: comprimir os cabeçalhos do IP/UDP/RTP, dando origem ao CRTP - Compressed RTP. Com esse método, os três cabeçalhos podem ser comprimidos em 2-4 octetos.

3.1.7 - RTP Data Transfer Protocol

No formato do pacote RTP, os 12 primeiros octetos estão presentes em todos os pacotes de um fluxo de dados de uma sessão RTP, enquanto a lista de identificadores CSRC (Contributing Source) só se encontra presente quando inserida por mixers. Cada campo é descrito a seguir:

- V (Version): versão RTP (atualmente, o valor é 2).
- P (Padding): indica a presença ou não de preenchimento das posições finais do pacote com um ou mais bytes que não fazem parte da carga (Payload).
- X (Extension): indica presença ou não de extensão de cabeçalho (Header Extensions).
- CC (CSRC Counter): contador do número de identificadores CSRC após o cabeçalho fixo.
- M (Marker Bit): delimita um conjunto de dados relacionados (por exemplo, o início de uma rajada de áudio ou o fim de um quadro de vídeo).
- PT (Payload Type): indica o formato da carga do RTP e determina sua interpretação pela aplicação.
- Sequence Number: incrementado em um para cada pacote RTP, e pode ser utilizado pelo receptor para detectar perda de pacote ou restaurar a própria sequência.

- Timestamp: utilizado pelo receptor para sincronização e cálculo do jitter.

- SSRC (Synchronization Source) Identifier: valor utilizado para identificar um fluxo específico em uma sessão RTP. Esse campo é necessário para o receptor poder agrupar pacotes com o mesmo SSRC para a reprodução.

- CSRC (Contributing Source) Identifiers: lista de identificadores CSRC inseridos por mixers (HUAWEI, 2009. **Command Manual (V3.53_01);** TANENBAUM, 2003; COLCHER *et.al.,* 2005).

Quando pacotes RTP relativos a um fluxo chegam ao seu destino, o número de sequência de cada pacote é examinado para determinar a sequência correta dos dados e também para registrar a fração de dados (por exemplo, amostrar de áudio ou quadros de vídeo) perdidos. O valor do timestamp do pacote é utilizado para determinar o espaçamento (gap) entre pacotes do fluxo. O valor do timestamp é marcado pelo transmissor do fluxo durante a codificação e envio dos dados. Conforme os pacotes do fluxo chegam ao receptor, a variação no espaçamento entre pacotes pode ser examinada, e, durante a reprodução, essa informação pode ser utilizada para regeneração do conteúdo na mesma taxa na qual este foi codificado. A partir da utlização de buffers no receptor, o transmissor pode gerar seu tráfego de saída de modo independente do jitter introduzido pela rede de pacotes.

3.1.8 - RTP Control Protocol

O RTCP é baseado na transmissão periódica de pacotes de controle para todos os participantes em uma sessão RTP, utilizando o mesmo mecanismo de distribuição definido para os pacotes dos fluxos.

O RTPC desempenha quatro funções principais:

- Prover informação a respeito da qualidade da distribuição dos dados de um fluxo. Essa é uma parte essencial do papel do RTP como um protocolo de transporte, e está relacionada às funções de controle de fluxo e congestionamento. A informação de retorno pode ser especialmente útil para o controle de codificações adaptativas, embora experimentos com IP multicasting tenham mostrado que é fundamental obter as informações de retorno dos receptores para diagnosticar falhas na distribuição.

- Transportar um identificador de nível de transporte persistente para um transmissor em uma sessão RTP, chamado nome canônico (CNAME). O CNAME é importante, já que um

identificador SSRC de um fluxo específico em uma sessão RTP pode mudar em função de uma interrupção de seu transmissor, o que significa que, se o transmissor for restabelecido, ele pode voltar a transmitir o mesmo fluxo, como um identificador de SSRC diferente. Assim, os receptores usam o CNAME para se manterem informados sobre cada participante. O CNAME também pode ser solicitado pelos receptores para associar múltiplos fluxos de um participante em um conjunto de sessões RTP relacionadas, como, por exemplo, nos casos em que é necessária a sincornização de áudio e vídeo.

- As funções anteriores requerem que todos os participantes enviem pacotes periodicamente; assim, a taxa de envio dos mesmos deve ser controlada para que o RTP possa ser escalável para um grande número de participantes.

- Transportar informações mínimas de controle de sessão, como, por exemplo, a identificação do participante que deve ser apresentada na interface com o usuário. Embora o RTP sirva como um canal conveniente para alcançar todos os participantes, outros requisitos para controle de comunicação de aplicações devem ser oferecidos por um protocolo de controle de sessão de mais alto nível (HUAWEI, 2009. **Command Manual (V3.53_01);** TANENBAUM, 2003; COLCHER *et.al.,* 2005).

3.2 - QoS – Quality of Service:

3.2.1 - Descrição geral

O QoS (Quality of Service) é uma das tecnologias mais utilizadas nos serviços de telecomunicações. Ele avalia o desempenho da transmissão dos pacotes na rede e o tráfego frente aos diversos tipos de serviços disponíveis na rede. Normalmente o QoS analisa o desempenho baseado nos principais requerimentos para o bom funcionamento de uma rede durante a transmissão de pacotes, tais como, Delay (atraso), Jitter e taxas de perda de pacotes (COLCHER *et.al.,* 2005).

Em redes IP, os roteadores tratam os pacotes de forma idêntica e lidam com o seu encaminhamento seguindo a política do First In, First Out (FIFO) - primeiro pacote a chegar será o primeiro pacote a ser encaminhado, reservando os recursos de acordo com a ordem da chegada dos pacotes.

Todos os pacotes compartilham os recursos da rede, a quantidade de recursos que serão utilizados dependerá diretamente da ordem de chegada dos pacotes. Essa política de serviço é chamada Best-effort, que entrega os pacotes em seu destino da melhor forma possível e

dependendo diretamente da disponibilidade de recursos, sem nenhuma garantia na entrega, delay, Jitter e taxa de perda dos pacotes.

A política de serviço Best-effort se aplica somente a ambientes de rede nos quais os serviços não são suscetíveis a atrasos na entrega dos pacotes, disponibilidade de banda, como por exemplo WWW, transferência de arquivos e e-mail.

Com a quantidade crescente de redes acessando a internet, e o surgimento de novas tecnológias e serviços além dos tradicionais WWW, FTP e E-MAIL, tais como, VoIP e VoD (Vídeo on Demand) e o uso desses novos serviços no meio corporativo uniram escritórios localizados em áreas distintas.

O que essas novas aplicações têm em comum é a necessidade de garantia dos recursos de redes, como largura de banda, baixo delay e jitter. Essas aplicações necessitam de um tratamento diferenciado na análise e entrega de seus pacotes além do simples encaminhamento dos mesmos, como por exemplo, garantia de banda por usuário, redução das taxas de perda de pacotes, mecanismos que evitem congestionamentos, reguladores de tráfego na rede, definição de prioridade no envio dos pacotes (HUAWEI, 2009. **Command Manual (V3.53_01);** TANENBAUM, 2003; COLCHER *et.al.,* 2005).

3.2.2 – Fundamentos do QoS

- Prevenir e gerenciar congestionamento em redes IP.
- Reduzir as taxas de perda de pacotes.
- Controlar o tráfego IP na rede.
- Disponibilizar largura de banda para um serviço ou usuário específico.
- Prover suporte para serviços de tempo real em redes IP (TANENBAUM, 2003; COLCHER *et.al.,* 2005; Site: Huawei, 2009).

3.2.3 - Modelos de QoS IP

- Modelo Best-Effort: o modelo Best-effort não garante a entrega dos pacotes.

- Modelo IntServ: serviço que trabalha enviando um sinal para o equipamento de gerencia de rede solicitando um determinado QoS, o mesmo reserva o recurso necessário de acordo com os parâmetros do tráfego em questão para satisfazer a solicitação.

- Modelo DiffServ: em casos de congestionamento na rede, aplicar o controle do tráfego e encaminhamento diferenciado baseados de políticas previamente adotadas, este foi o modelo escolhido para os cenários adotados nesse trabalho.

BEST-EFFORT:

Modelo dedicado a serviços não suscetíveis a delay, perda de pacotes e jitter. Usando esse modelo é possível transmitir qualquer número de pacotes em qualquer momento dentro de uma rede sem necessidade de análise prévia. Não há garantia de entrega de pacotes nem de taxas de perda de pacotes e atrasos, por fim, utiliza a tecnologia FIFO (First In, First Out) (HUAWEI, 2009. **Command Manual (V3.53_01);** TANENBAUM, 2003; COLCHER *et.al.,* 2005).

INTSERV:

O termo serviços integrados é utilizado para designar um modelo de serviços que acrescenta ao serviço de melhor esforço, historicamente oferecido na internet, categorias de serviços com diferentes graus de comprometimento de recursos (banda passante e buffers, em particular) e, consequentemente, com diferentes níveis de QoS para fluxos de transporte distintos da rede IP. Um fluxo é tipicamente identificado na arquitetura IntServ pela dupla (Endereço IP de origem, Porta UDP/TCP de origem, endereço IP de destino, Porta UDP/TCP de destino), o que permite uma alocação individual de recursos por fluxo de transporte, embora, em tese, possa haver níveis maiores ou menores de granularidade de alocação (por exemplo, alocando recursos de forma agregada para todos os fluxos entre duas estações finais específicas) (TANENBAUM, 2003; COLCHER *et.al.,* 2005; Site: Huawei, 2009).

Em Braden (1994) também é apresentado um modelo de referência para implementação desses serviços em termos de componentes presentes em roteadores e estações finais, bem como de protocolos que permitem a sinalização para alocação de recursos de banda passante nos enlaces entre esses equipamentos . Os principais componentes que fazem parte desse modelo de referência são:

- Controle de admissão: determina se um novo fluxo pode ser aceito sem interferir na QoS de fluxos admitidos anteriormente. Esse controle é feito a partir dos parâmetros de tráfego e de QoS especificados pelas aplicações para cada um de seus fluxos, bem como em função dos recursos (tipicamente, banda passante e buffers) disponíveis na rede.

- Escalonador de pacotes: gerencia os buffers das filas de saída dos roteadores e estações, usando alguma política de atendimento às mesmas.

- Classificador: reconhece os fluxos segundo suas identificações, mapeia os pacotes desses fluxos nas diferentes categorias de serviço, notifica a função de policiamento e, caso os pacotes estejam em conformidade com o controle imposto pelo policiamento, os coloca nos buffers das filas de saída apropriadas.

- Policiamento: determina se os pacotes notificados pelo classificador estão em conformidade com os parâmetros de tráfego e QoS negociados para o fluxo – parâmetros esses informados durante o controle de admissão (TANENBAUM, 2003; COLCHER *et.al.,* 2005; Site: Huawei, 2009).

Figura 8 - INTSERV:

Desvantagens:

- Necessidade que todos os equipamentos fim a fim suportem o protocolo utilizado;
- Não é aplicável a redes de grande porte;
- O Overhead das mensagens trocadas com a vizinhança é alto, e essas mensagens devem ser trocadas periodicamente informando o status (TANENBAUM, 2003; COLCHER *et.al.,* 2005; Site: Huawei, 2009).

DIFFSERV:

Com o objetivo de contornar o problema da escalabilidade da arquitetura InterServ, foi proposta no IETF a arquitetura DiffServ , que define um modelo de serviços diferenciados. Através da agregação de fluxos individuais em classes de serviço, essa arquitetura possibilita uma redução no número de estados que devem ser mantidos nos roteadores da rede (TANENBAUM, 2003; COLCHER *et.al.,* 2005; Site: Huawei, 2009).

Deve-se enfatizar que a agregação de fluxos proposta na arquitetura DiffServ corresponde a uma solução de compromisso objetivando viabilizar a aplicação de um modelo de serviços em redes de maior porte (como redes de backbone). Em muitos casos, o mecanismo de agregação pode implicar a associação de um fluxo a uma classe que forneça um serviço cuja qualidade é melhor do que a requerida, o que reduz, obviamente, a eficiência da reserva de recursos no provedor (TANENBAUM, 2003; COLCHER *et.al.,* 2005).

A agregação de fluxos é realizada através da identificação da classe de serviço em um ou mais campos específicos do cabeçalho dos pacotes IP. Há dois tipos de classificação principais. A classificação MF (Multi-Field) usa na identificação dos fluxos um esquema similar ao da arquitetura IntServ. Na classificação BA (Behaviour Aggregate), é usado um único campo especial. No IPv4, esse campo é o originalmente chamado de TOS, tenso sido renomeado para DS (Differentiated Services). No IPv6, é usado um campo chamado Traffic Class. A marcação de classe de serviço no campo DS é normalmente realizada pelo roteador de borda (Edge Router – ER) de um domínio DiffServ – um conjunto de sub-redes e roteadores que seguem o mesmo conjunto de regras administrativas para estabelecimento de SLAs (HUAWEI, 2009. **Command Manual (V3.53_01);** TANENBAUM, 2003; COLCHER *et.al.,* 2005).

Porém, quando a leitura de alguns dos campos do cabeçalho dos pacotes não pode ser realizada pelos roteadores de borda, o que ocorre quando são utilizados mecanismos de criptografia, por exemplo, cabe à própria aplicação efetuar a marcação na origem dos pacotes (host marking). Outras situações também podem requerer que a marcação seja realizada pela aplicação, como quando são utilizadas portas de transporte ou endereços IP voláteis (aprendidos via DHCP ou mascarados via NAT, por exemplo). Em ambos os casos, o valor da classe de serviço a ser marcado no campo DS deve ser informado pela administração do domínio DiffServ, uma vez que a aplicação desconhece essa informação, a princípio. A exceção ocorre quando são utilizados valores universais para o referido campo, os chamados GWSIDs (Global Well-Known Service IDs), em geral definidos no próprio IETF, por

exemplo. Uma vez identificados pela marcação da classe de serviço no campo DS, os pacotes receberão um tratamento diferenciado nos roteadores do domínio DiffServ (TANENBAUM, 2003; COLCHER *et.al.,* 2005).

A divisão de funcionalidade entre os roteadores de um domínio DiffServ é uma outra característica particular dessa arquitetura. Cabe aos ERs as tarefas mais complexas de classificação, marcação e policiamento do tráfego entrante no domínio. Opcionalmente, funções de moldagem do tráfego de saída do domínio, responsáveis por formatar um fluxo de acordo com um perfil de tráfego especificado, também podem ser acrescentadas ao ER. A inclusão dessas funções de processamento mais intenso nos ERs não introduz problemas de escalabilidade ao modelo, uma vez que é esperado, nessa região, um menor volume de tráfego. Por outro lado, no núcleo do domínio DiffServ, onde a banda disponível e a quantidade de fluxos são proporcionalmente maiores, o processamento realizado pelos roteadores internos (Transient Routers – TRs) é simplificado ao máximo, sendo eliminada a função de policiamento e moldagem dos fluxos, além de reduzida a complexidade da função de classificação, que identificará as classes de serviço pela leitura direta do campo DS previamente marcado (HUAWEI, 2009. **Command Manual (V3.53_01);** TANENBAUM, 2003; COLCHER *et.al.,* 2005).

Solicitação de serviços DiffServ

SLAs na arquitetura DiffServ, quando comparados com aqueles da arquitetura IntServ, são mais abrangentes e, normalmente, mais estáveis, apresentando um tempo de vida maior. Um SLA IntServ limita-se aos aspectos técnicos da caracterização de um serviço, como o nível de garantia desejado, a descrição do tráfego correspondente aos fluxos e os parâmetros de QoS. Um SLA DiffServ, no entanto, considera relevante outras características de um serviço, como a forma de tarifação, as penalidades contratuais nos casos de interrupção etc. A porção técnica do SLA DiffServ é denominada SLS (Service Level Specification). Um SLS pode especificar, entre outros requisitos:

- A disponibilidade do serviço, descrevendo ações a serem realizadas em situações de descontinuidade do mesmo.

- Os mecanismos de autenticação e criptografia empregados.

- As restrições de rotas a serem utilizadas no encaminhamento do tráfego agregado.

- As formas de monitorização e auditoria da QoS fornecida.

Além desses parâmetros, a SLS contém ainda o TCS (Traffic Conditioning Specification), responsável por descrever:

- Os parâmetros de caracterização do tráfego.

- As ações de policiamento aplicadas aos pacotes não conformes com o perfil de tráfego prometido.

- Os parâmetros específicos de QoS, como vazão esperada, probabilidade de descarte, retardo máximo etc.

- O escopo do serviço, através de restrições de entrada e saída por onde os fluxos serão transportadas (HUAWEI, 2009. **Command Manual (V3.53_01);** TANENBAUM, 2003; COLCHER *et.al.,* 2005).

A arquitetura IntServ emprega categorias de serviço padronizadas, estando especificadas, atualmente, as categorias garantida e carga controlada. O uso de categorias padronizadas limita os serviços que podem ser fornecidos, dando pouco espaço à criatividade e competição entre provedores de serviço.

A arquitetura DiffServ não padroniza serviços, especificando apenas comportamentos de encaminhamento ou PHBs (Per Hop Behaviors). Um PHB descreve como será realizado o encaminhamento dos pacotes pertencentes a uma mesma classe em cada roteador, sendo as unidades básicas utilizadas na definição dos serviços oferecidos pela rede. A combinação de PHBs, juntamente com os demais parâmetros de caracterização da SLS, é que define o serviço a ser efetivamente fornecido em um domínio DiffServ. Devido à ausência de padronização de serviços, a arquitetura DiffServ permite que diferentes provedores concorram entre si no fornecimento de funcionalidades que agreguem vantagens aos usuários, como disponibilidade, facilidades de restauração automática na presença de falhas, entre outras. Cabe ao usuário escolher o provedor que forneça o conjunto de serviços e funcionalidades que melhor atenda às necessidades específicas das suas aplicações.

Dois PHBs principais encontram-se padronizados pelo IETF: o encaminhamento expresso (Expedited Forwarding – EF) e o encaminhamento assegurado (Assured Forwarding – AF) (HUAWEI, 2009. **Operation Manual (V3.53_01);** TANENBAUM, 2003; COLCHER *et.al.,* 2005).

O PHB EF é recomendável ao fornecimento de serviços com baixo retardo e variação, taxa de erros controlada e largura de banda assegurada. Serviços com essas características são

normalmente referenciados, no mercado, como serviços de linha virtual alugada (Virtual Leased Line – VLL) ou serviços premium. A implementação desse PHB procurará eliminar as filas nos roteadores, o que é obtido através do uso de uma taxa de saída mínima maior ou igual do que a taxa de entrada agregada máxima. A disciplina de escalonamento usada deve garantir a proteção de fluxo, ou seja, o tráfego de uma classe de serviço não deve interferir no de outra. VoIP é o principal exemplo de aplicação que pode fazer uso desse serviço.

O PHB AF define, na realidade, um grupo formado por várias modalidades independentes de encaminhamento. O grupo AF permite a implementação de serviços com diferentes níveis de garantia de QoS, como serviços com garantia probabilística, por exemplo. No âmbito de cada classe AF, pacotes são marcados adicionalmente por um valor de precedência descarte. A probabilidade de entrega de um pacote servido por um PHB AF depende, portanto, de três fatores:

- Classe AF ao qual pertence: quanto maior o valor, maior a probabilidade.

- Tráfego agregado atual da referida classe.

- Valor de precedência de descarte do pacote: quanto menor a precedência, maior a probabilidade.

Cada pacote IP é identificado, no campo DS, por um valor AFij, onde "i" define a classe AF e "j" o valor de precedência de descarte do pacote dentro da referida classe. Quatro classes AF principais encontram-se definidas, cada qual podendo ter até três níveis de precedência. Os parâmetros de qualidade a serem aplicados a cada uma dessas classes são definidos de forma independente por cada domínio DiffServ. A única restrição refere-se à preservação da ordem de garantia e de descarte no encaminhamento dos pacotes pertencentes ao grupo de serviços AF fornecidos pelo domínio DiffServ (HUAWEI, 2009. **Command Manual (V3.53_01);** TANENBAUM, 2003; COLCHER *et.al.,* 2005).

Uma grande variedade de serviços diferenciados podem ser fornecidos usando o grupo de encaminhamento assegurado, como, por exemplo, os chamados serviços olímpicos, com as classes ouro, prata e bronze. Obviamente, de forma a garantir que o serviço da classe ouro apresente um desempenho de encaminhamento melhor do que o oferecido pelos serviços das demais classes, um dimensionamento apropriado dos recursos deve ser feito baseado nas demandas de tráfego esperadas para cada uma das classes que se deseja suportar no domínio DiffServ.

3.2.5 - Congestionamento

O congestionamento é um dos fatores que causam a degradação da qualidade das redes atualmente. O congestionamento nada mais é do que a limitação dos serviços devido a limitação dos recursos disponíveis - normalmente causando grande atraso na entrega dos pacotes (Site: Huawei, 2009).

Figura 9 - Causas de Congestionamento

CAUSAS:

1 – Pacotes chegam ao roteador via um link de alta velocidade (LAN)e são enviados utilizando um link de velocidade inferior (WAN);

2 – Pacotes chegam ao roteador via diversos links de uma determinada velocidade e são enviados por um único link com a mesma capacidade (Site: Huawei, 2009).

O congestionamento é normalmente causado por esse gargalo nos links disponíveis para o encaminhamento dos pacotes (Site: Huawei, 2009).

Além do congestionamento causado pelo gargalo da capacidade dos links, existem outros fatores que também causam congestionamento, entre eles a limitação dos próprios recursos envolvidos no encaminhamento dos pacotes, tais como, limitação de processamento do equipamento, buffer e memória (Site: Huawei, 2009).

Além dos citados acima, o congestionamento poderá ocorrer caso o gerenciamento no recebimento do tráfego não funcione corretamente e os recursos disponíveis para o encaminhamento sejam inadequados. Dessa maneira, o congestionamento pode causar os seguintes impactos em uma rede:

- Aumento do atraso e do Jitter na transmissão dos pacotes.
- Retransmissão de pacotes devido aos altos tempos de atraso.

- Degradação da eficiência da largura de banda disponível na rede causando desperdício dos recursos de rede.

- Congestionamentos muito intensos podem causar uma grande ocupação dos recursos de rede, e a reserva descontrolada desses recursos pode levar ao bloqueio dos mesmos ou até uma queda total do sistema (HUAWEI, 2009. **Command Manual (V3.53_01);** TANENBAUM, 2003; COLCHER *et.al.,* 2005).

Fica claro que o congestionamento irá causar impactos no ambiente de rede, tanto ocupando indevidamente os recursos disponíveis quando degradando a qualidade dos serviços utilizados.

O congestionamento acontece freqüentemente em ambientes de rede complexos, o que é muito comum nos dias de hoje, onde a SWITICHING dos pacotes e as aplicações para multi-usuários coexistem.

Portanto, para casos de congestionamento é preciso tomar as devidas precauções. Uma das primeiras maneiras é o aumento da largura de banda disponível, em alguns casos essa medida pode solucionar o problema em um primeiro momento, mas com o aparecimento de novas aplicações e a quantidade de usuários, essa medida se tornará ineficiente (Site: Huawei, 2009).

Uma maneira mais eficiente de tratar o problema da garantia da qualidade de serviço é a utilização de melhorias no controle do tráfego e alocação de recursos na camada de rede (Camada 03), além de prover serviços diferenciados para aplicações com necessidades específicas, e com isso alocar os recursos de forma mais acertada (Site: Huawei, 2009).

Durante o processo de controle de tráfego e alocação de recursos, devemos usar o mecanismo de Best-Effort (FIFO) para controlar os fatores diretos e indiretos que possam causar congestionamento diminuindo a probabilidade desses evetos ocorrerem na rede. Caso o congestionamento ocorra, o controle de tráfego e alocação de recursos deverá entrar em ação levando em conta as necessidades e especificações das aplicações envolvidas (HUAWEI, 2009. **Command Manual (V3.53_01);** TANENBAUM, 2003; COLCHER *et.al.,* 2005).

3.2.6 - Tecnologias de controle de tráfego

O Traffic classification, traffic policing, traffic shaping, congestion management, congestion avoidance, and physical-interface LR são os intens. fundamentais para que uma

rede possa prover serviços diferenciados, fundamentalmente eles desempenham as seguintes funções:

- Traffic Classification (Classificação de Tráfego): É um pré-requisito para a diferenciação de serviços, basicamente é a identificação de determinados objetos - normalmente de interesse da aplicação em questão, essa identificação é feita baseada em uma serie de regras.

- Traffic Policing (Monitoramento de Tráfego): São políticas de especificam um tipo particular de tráfego entrando no roteador. Quando o tráfego foge as especificações então regras de restrição ou punição entram em ação garantindo a qualidade do serviço e prevenindo que os recursos da rede sejam exauridos ou fiquem indisponíveis. As políticas de tráfego são aplicadas na camada de rede.

- Congestion management (Gerenciamento de Congestionamento): Gerencia a disponibilidade dos recursos durante um congestionamento na rede, normalmente, armazena os pacotes em filas, e despacha os pacotes utilizando um algoritmo que garanta o correto encaminhamento dos pacotes.

- Congestion Avoidance (Medidas anti-congestionamento): Congestionamentos excessivos consomem em demasia os recursos da rede, medidas anti-congestionamento monitoram esses recursos e conforme o congestionamento se agrave, utilizam políticas de eliminação dos pacotes (DROP), evitando que o congestionamento continue.

- Traffic Shapping (Modelagem de Tráfego): Uma medida de controle de tráfego que ajusta a velocidade de encaminhamento dos pacotes, normalmente ajusta esse encaminhamento baseado nas menores velocidades disponíveis nos roteadores da rede, prevenindo assim desnecessária eliminação de pacotes (DROP) e o congestionamento. Traffic Policing e Traffic Shapping são aplicados na camada de rede.

- Physical-interface LR: Diferentemente das políticas de tráfego, que trata dos pacotes na camada IP, LR restringe os pacotes na camada física e isso é melhor aplicado em situações onde se pretende limitar todos os pacotes.

Dentre todas as tecnologias de gerenciamento de tráfego, a classificação do tráfego é a básica, pois é o pré-requisito para diferenciar um determinado serviço, identificando, por exemplo, um certo pacote baseado em regras de identificação (HUAWEI, 2009. **Command Manual (V3.53_01);** TANENBAUM, 2003; COLCHER *et.al.,* 2005).

Já as demais, implementam o gerenciamento do tráfego de rede e a alocação de recursos diferenciadamente e baseados nas necessidades das aplicações.

3.2.7 - Classificação de tráfego

A classificação de tráfego é um dos pré-requisitos e fundamentos da diferenciação de serviços, para tal, certas regras para a identificação de características nos pacotes são utilizadas (Site: Huawei, 2009).

Para a identificação de tráfego é possível a utilização de regras de classificação baseadas no campo ToS (Type of Service),encontrados nos cabeçalhos dos pacotes IP, ou utilizar o CLP (Cell Loss Priority) do cabeçalho das células ATM.

Também é possível fazer a classificação utilizando políticas de classificação de tráfego, integrando informações como por exemplo:

- Endereço IP.
- Endereço IP de destino.
- Endereço MAC.
- IP.
- Porta das aplicações (Ex: VoIP).

Normalmente podemos usar uma combinação quíntupla de itens, endereço de origem IP, endereço de porta de origem, endereço de porta de destino e tipo de protocolo transporte ou podemos classificar todos os pacotes de um determinado segmento de rede (HUAWEI, 2009. **Command Manual (V3.53_01);** TANENBAUM, 2003; COLCHER *et.al.,* 2005).

Enquanto os pacotes são classificados na borda da rede, os bits de precedência encontrados no byte ToS do cabeçalho IP são definidos de tal forma que a precedência IP pode ser usada como um padrão direto de classificação dentro de uma rede.

Nas tecnologias de enfileiramento, como por exemplo, o WFQ, que utiliza a preferência para lidar com os pacotes de rede, redes encontradas dentro de outras redes, podem receber os resultados seletivos de uma classificação prévia ou reclassificar os pacotes de acordo com seu próprio padrão.

A classificação de tráfego é usada para prover diferenciação nos serviços e para isso deverá ser associada a certos tipos de políticas de tráfego ou mecanismos de alocação de recursos. Para definir o tipo de ação e de política de tratamento a ser utilizada deve-se ter em mãos o estado atual e da carga na rede e fazer enfileiramento de tráfego em eventos de congestionamento e para empregar métodos de prevenção de congestionamento em momentos que o congestionamento atinge níveis altos (HUAWEI, 2009. **Command Manual (V3.53_01);** TANENBAUM, 2003; COLCHER *et.al.,* 2005).

3.2.8 - monitoramento e modelagem de tráfego

Caso não exista nenhuma restrição imposta ao tráfego na rede criada por usuários, o envio descontrolado de dados pelos usuários continuamente irá fazer com que a rede congestione. Assim para que seja mantido a eficiência das funções da rede e os níveis de serviço, os tráfegos dos usuários devem ser restringidos, por exemplo, restringir um tráfego em um determinado momento alocado a um recurso com intenção de prevenir o congestionamento.

Monitoramento de tráfego e modelagem de dados são políticas de monitoramento para ajustar o tráfico e recursos através da comparação com as especificações de tráfego, saber se o tráfico excede a especificação ou não é um pré-requisito para o monitoramento e a modelagem de trafego. Então baseado no resultado de avaliação pode-se implementar uma política regulatória. Normalmente Token Bucket é usado para avaliar a especificação de tráfico (HUAWEI, 2009. **Command Manual (V3.53_01);** TANENBAUM, 2003; COLCHER *et.al.,* 2005).

3.2.9 - Avaliação de tráfego e token bucket

CARACTERÍSTICAS TOKEN BUCKET

Token bucket pode ser descrito como um container para armazenar os Tokens, que possui uma capacidade limitada, o sistema irá colocar tokens no token bucket a uma taxa definida, caso o token bucket esteja cheio, os tokens extras irão transbordar e nenhum token adicional irá ser armazenado. (HUAWEI, 2009. **Command Manual (V3.53_01))**

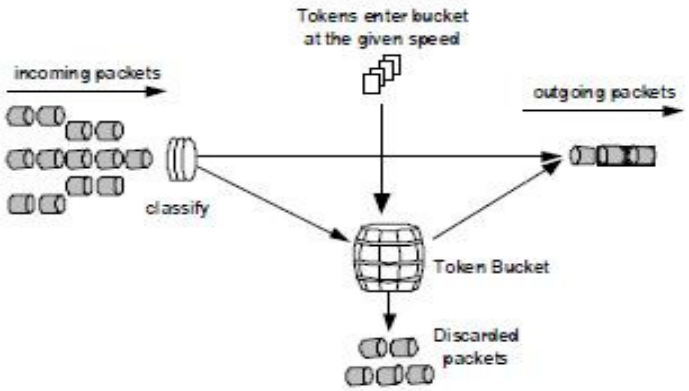


Figura 10 - Token Bucket

MEDINDO O TRÁFEGO COM O TOKEN BUCKET

O fato da capacidade de armazenar Tokens do Token Bucket atender ou não a demanda do encaminhamento de pacotes é a premissa que Token bucket utiliza para a avaliação da especificação de tráfego, caso existam uma quantidade suficiente de Tokens que suporte a taxa de encaminhamento de pacotes, o tráfego é considerado de acordo com as especificações (Geralmente um Token é associado à habilidade de encaminhar um bit) caso contrário, não está de acordo com a especificação e considera-se excesso.

Ao medir o tráfico com Token Bucket, estes parâmetros são considerados:

- Mean rate: A taxa que informa a colocação de Tokens no Token Bucket (Taxa média de tráfego permitido, normalmente configurada como CIR (Commited Information Rate).

- Burst Size: A capacidade do Token Bucket (Tráfego máximo de cada rajada, geralmente é configurado como CBS (Commited Burst Size), o Burst Size deverá ser maior que o tamanho máximo dos pacotes.

Uma nova avaliação será feita quando um pacote novo chegar. Se houver bastante tokens no repositório para cada avaliação, mostra que o tráfego está dentro dos limites, e neste momento a quantidade de tokens se adéqua corretamente a quantidade de pacotes que são encaminhados, caso contrário, demonstra-se que muitos Tokens foram usados e as especificações de tráfego excedem (HUAWEI, 2009. **Command Manual (V3.53_01);** TANENBAUM, 2003; COLCHER *et.al.,* 2005)

AVALIAÇÃO COMPLEXA:

Dois Token Buckets podem ser configurados para avaliar as condições que são consideradas mais complexas e para implementar políticas de regulação mais flexíveis. Por exemplo: Traffic Policing (TP), possui 3 parâmetros, como segue abaixo:

CIR – Commited Information Rate

CBS – Commited Burst Size

EBS – Excess Burst Size

Ele utiliza dois token Buckets com a taxa de colocação de Tokens de cada um dos repositórios configuradas como CIR iguais, mas com capacidades de armazenamento diferentes: CBS e EBS (CBS < EBS, chamados C Bucket e E Bucket). O que representa a permissão de diferentes classes de rajadas. Em cada avaliação é possível utilizar diferentes políticas de controle de tráfego para diferentes situações, como por exemplo:

Repositório C possui Tokens suficientes

Repositório C não possui tokens suficientes, porém repositório E possui Tokens suficientes.

Repositórios C e E não possuem capacidade suficiente de Tokens. E n Tokens suficientes (HUAWEI, 2009. **Command Manual (V3.53_01);** TANENBAUM, 2003; COLCHER *et.al.,* 2005).

3.2.10 - monitoramento de tráfego

Basicamente podemos descrever o monitoramento de tráfego como sendo a monitoração de um tráfego específico entrante na rede e mantê-lo dentro de limites razoáveis, caso contrário serão aplicadas penalidades no tráfego excedente visando proteger os recursos da rede e benefícios da portadora.

Exemplo, é possível restringir o tráfego HTTP a ocupar até 50% da largura de banda disponível, no momento que se identificar que o tráfego excedeu esse limite, pode-se descartar os pacotes ou renovar a precedência dos pacotes.

O Monitoramento de tráfego permite a definição de regras de comparação baseadas na precedência IP ou no DiffServ Code Point (DSCP), o que é largamente utilizado hoje pelos ISP (Internet Service Provider) para monitorar o tráfego em suas redes, o TP também incluí um serviço de classificação de tráfego para os tráfegos já monitorados, e dependendo dos

diferentes resultados encontrados, o TP irá aplicar ações de monitoração pré-estabelecidas, descritas abaixo:

- Forward: Continuar encaminhando os pacotes avaliados com conformidade.
- Drop: Descartar os pacotes avaliados como sem conformidade.
- Alterar Precedência e Forward: Pacotes avaliados como conformidade parcial, trocar a precedência e encaminhar
- Avaliar em monitoramento de nível diferente: o monitoramento de tráfego pode ser empilhado gradualmente no qual cada nível trata de um tipo de monitoração de objetos especifico (HUAWEI, 2009. **Command Manual (V3.53_01);** TANENBAUM, 2003; COLCHER *et.al.,* 2005).

3.2.11 – Modelagem de tráfego

A modelagem de tráfego é um modo ativo de ajustar as taxas de saída de tráfego. Um dos usos típicos da modelagem de tráfego é a aplicação para controle do tráfego de saída de rede baseado na lista de capacidades de Downstream dos nós na rede (HUAWEI, 2009. **Command Manual (V3.53_01);** TANENBAUM, 2003; COLCHER *et.al.,* 2005).

A diferença entre o Monitoramento de tráfego e a Modelagem de Tráfego é:

Os pacotes que deverão ser descartados devido ao monitoramento de tráfego são armazenados em filas (queues) durante a modelagem de tráfego, a partir do momento que existam Tokens suficientes no repositório, os pacotes armazenados serão enviados, o tamanho do tráfego altera com valores perto das taxas de CIR e dentro dos limites especificados pelo CBS.

Outra diferença é que a modelagem de dados pode intensificar o DELAY (Atraso), já o monitoramento do tráfego raramente faz isso.

Para reduzir o descarte de pacotes, GTS (Generic Traffic Shaping) pode ser usado para os pacotes na interface de saída do roteador A. Os pacotes a parte das características do GTS serão armazenados no roteador A. Enquanto envia o próximo grupo de pacotes, GTS irá enviar os pacotes do buffer ou Fila e encaminha-los. Assim todos os pacotes serão encaminhados para o roteador B de acordo com as regulamentações do roteador B.

3.2.12 - Gerenciamento de congestionamento

Em qualquer dispositivo de rede, o congestionamento irá ocorrer na interface onde a chegada de pacotes é mais rápida que a capacidade de encaminhamento, caso não exista buffer suficiente para armazenar esses pacotes excedentes, eles irão ser descartados, o que irá gerar a necessidade de retransmissão dos pacotes pelos hosts ou roteadores da rede devido ao timeout (HUAWEI, 2009. **Command Manual (V3.53_01);** TANENBAUM, 2003; COLCHER *et.al.,* 2005).

A principal missão do gerenciamento de congestionamento é como definir uma política de despacho para os recursos decidirem a ordem de encaminhamento dos pacotes no momento que o congestionamento ocorrer.

3.2.13 - Políticas de gerenciamento de congestionamento

Geralmente, o gerenciamento de congestionamento adota a tecnologia de enfileiramento (queuing), dessa forma o sistema classifica o trafego utilizando um tipo de algoritimo de enfileiramento, e depois envia os pacotes baseados com um algorítimo de preferência específico, cada algoritimo de enfileiramento é usado para lidar com um determinado problema de trafego e causa impactos sensíveis na reserva dos recursos e largura de banda, delay e Jitter.

Abaixo, seguem os mais comuns mecanismos de enfileiramento:

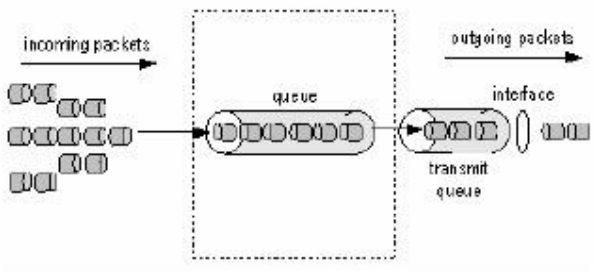


Figura 11 – Exemplo FIFO

Como podemos acompanhar pela imagem acima, o FIFO determina a ordem do encaminhamento dos pacotes baseado diretamente na ordem de chegada dos mesmos, em um roteador, os recursos responsáveis pelo tráfego de dados se baseiam no horário de chegada

dos pacotes e a ocupação momentânea da rede (HUAWEI, 2009. **Command Manual (V3.53_01);** TANENBAUM, 2003; COLCHER *et.al.,* 2005).

Se existir apenas enfileiramentos entrada/saída baseados em FIFO nas interfaces do roteador, determinadas aplicações poderão ocupar totalmente os recursos de rede e afetar seriamente a transmissão de dados críticos.

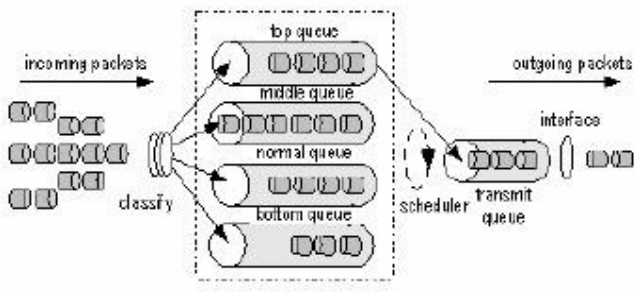


Figura 12 - PQ – (Priority Queuing):

A utilização do PQ (Priority Queuing) é destinada a aplicações críticas, essas aplicações apresentam características comuns - em momentos de congestionamento as aplicações requisitarão recursos visando diminuir o atraso nas respostas, o PQ pode designar flexivelmente seqüência de prioridades de acordo com:

- Protocolo de rede utilizado (Ex: IP ou IPX).
- Interface de recebimento dos pacotes.
- Tamanho dos pacotes.
- Endereço de IP – destino ou origem, entre outras.

O PQ classifica os pacotes em 4 categorias: TOP, MIDDLE, NORMAL e BOTTOM, em ordem descendente de prioridade. Por padrão os pacotes que chegam na interface são classificados como NORMAL.

Durante o despacho das filas, PQ obedece fielmente a seqüência de prioridade da mais alta a mais baixa, e irá encaminhar os pacotes da prioridade mais alta primeiro, quando essa fila estiver vazia, PQ iniciará a enviar os pacotes das prioridades mais baixas.

O sistema sempre colocará os pacotes das aplicações críticas na fila de alta prioridade (TOP) e os pacotes de aplicações normais em filas de prioridades inferiores, com isso o PQ garante que os pacotes das aplicações críticas serão sempre enviados primeiramente e os

pacotes das aplicações de prioridades inferiores serão enviados nos intervalos em que a prioridade mais alta não é exigida.

A desvantagem do PQ é que os pacotes das prioridades inferiores podem ficar longos períodos negligenciados caso não existam intervalos no encaminhamento dos pacotes das aplicações críticas (HUAWEI, 2009. **Command Manual (V3.53_01);** TANENBAUM, 2003; COLCHER *et.al.,* 2005).

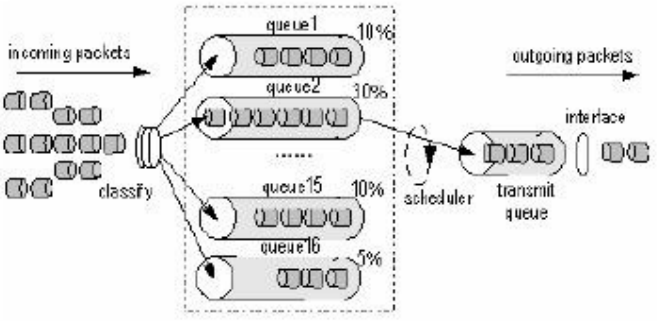


Figura 13 - CQ (Custom Queuing):

CQ classifica os pacotes em 17 classes de acordo com determinadas regras (correspondentes às 17 filas). Dentro de suas filas, os pacotes irão entrar em suas filas utilizando a política FIFO.

Das 17 filas disponíveis, a fila 0 é destinada ao sistema - não é possível visualizá-la ou configurá-la - e as fila de 1 a 16 são as filas de usuários como ilustrado na imagem acima. Com o CQ é possível configurar as regras de classificação de tráfego e definir as proporções de ocupação da banda disponível para as 16 filas disponíveis. Durante o despacho dos pacotes, os pacotes da fila 0 (sistema) serão encaminhados preferencialmente até que a fila esteja vazia, após isso, utilizando o método POLLING, um certo número de pacotes tirados das filas 1 – 16 e com a configuração de proporção de banda previamente configurada, serão enviados.

Dessa forma pacotes de diferentes aplicações poderão ser encaminhados com diferentes larguras de banda, o que não só garantirá que a aplicação crítica utilize a maior parte da banda disponível como também evitará que aplicações normais fiquem sem nenhuma banda disponível, por padrão, todo o tráfego entrante é classificado como fila 1.

Outra vantagem do CQ é que a largura de banda pode ser definida de acordo com o negócio da aplicação, que se encaixa perfeitamente para as aplicações que exigem largura de banda mínima garantida. Utilizando o POLLING para encaminhamento das 16 filas, o tempo

de serviço para cada fila não é fixo, portanto em momentos em que não há pacotes em determinadas classes, o mecanismos de encaminhamento do CQ pode aumentar a banda ocupada pelos pacotes das classes existentes (HUAWEI, 2009. **Command Manual (V3.53_01);** TANENBAUM, 2003; COLCHER *et.al.,* 2005).

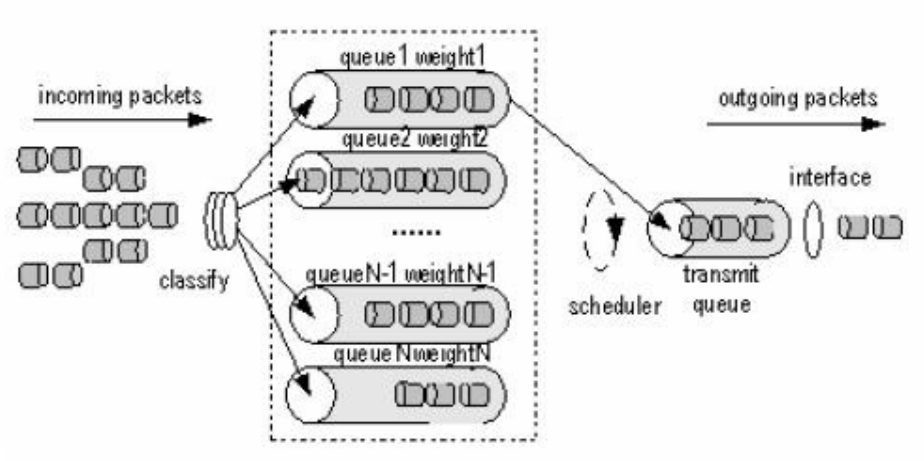


Figura 14 - WFQ (Weighted Fair Queuing):

Antes de falar em WFQ (Weighted Fair Queuing), é necessário introduzir informações sobre o FQ (Fair Queuing), o FQ é designado para situações onde é necessário o compartilhamento justo dos recursos da rede, o FQ irá tentar reduzir o delay e o Jitter de todos os tráfegos para os seus níveis otimizados, ele utiliza todos os aspectos em consideração e possui as seguintes características:

- Diferentes filas possuem uma oportunidade justa de despachar os pacotes para equilibrar o delay de cada fluxo dentre a demanda total.

- Pacotes pequenos e pacotes grandes são tratados com equilíbrio ao sair da fila: Caso existam pacotes grandes em uma fila e pacotes pequenos em outra esperando simultaneamente para serem encaminhados, os pacotes pequenos também deverão ser levados em consideração - estatisticamente os pacotes pequenos deverão ser tratados preferencialmente - Assim o jitter entre os pacotes de todos os tráfegos serão reduzidos no seu total.

Comparado com o FQ o WFQ leva em consideração a prioridade ao calcular a seqüência de pacotes que deverão ser encaminhados. Com o WFQ os tráfegos de alta prioridade tem prioridade sobre pacotes de baixa prioridade ao serem encaminhados. O WFQ pode automaticamente classificar o trafego de acordo com as informações da SESSÃO (Tipo de protocolo, Origem/Destino da porta TCP/UDP, Origem/Destino de endereço IP, Bits de

preferência do ToS...) e assim tentar disponibilizar mais filas para que cada tráfego seja colocado igualitariamente equilibrando assim o delay de todos os tráfegos.

Enquanto encaminha os pacotes o WFQ pode definir uma largura de banda na interface de saída para cada fluxo baseado na precedência IP ou DSCP. Quanto maior o valor numérico da precedência, mais largura de banda poderá ser reservada e por fim, enquanto envia os pacotes o WFQ varre cada fila e seleciona os pacotes baseado na razão da banda.

Exemplo: suponha que existam 5 tráfegos na interface, e seus níveis de precedência são 0,1,2,3,4 respectivamente, então a cota total de largura de banda será a soma de todas as precedências +1 (1+2+3+4+5 = 15).

A proporção de ocupação de banda para cada tráfego é (Prioridade + 1)/ Cota total da banda, dessa maneira chegamos a: 1/15, 2/15, 3/15, 4/15 e 5/15.

O fato de que o WFQ balancear o delay e o Jitter de todos os tráfegos em momentos de congestionamento é eficientemente aplicado em campos específicos, por exemplo, em serviços garantidos que utilizam o RSVP (Resource Reservation Protocol, geralmente o WFQ é usado como política de despacho, também no GTS, o WFQ é utilizado para despachar os pacotes em buffer (HUAWEI, 2009. **Command Manual (V3.53_01);** TANENBAUM, 2003; COLCHER *et.al.,* 2005).

CBQ (Class-Based Queuing):

CBQ é uma extensão do WFQ, porém que comporta classes definidas pelo usuário. O CBQ reserva uma fila independente FIFO para cada umas das classes definidas pelo usuário para armazenar (Buffer) dados da mesma classe. Em casos de congestionamento da rede o CBQ identifica os pacotes de saída de acordo com as regras de classe definidas pelo usuário e as habilita a entrarem em suas filas correspondentes. É necessário checar os mecanismos de prevenção de congestionamento (Tail drop ou WRED) e restrições de banda antes que os pacotes sejam enfileirados, após isso, o WFQ é aplicados aos pacotes nas filas correspondentes as classes no momento em que deixam suas filas.

Se o CBQ tratasse as filas de todas as classes no modo Weighted Fair, fluxos de dados sensíveis a delay como pacotes de voz poderiam não ser atendidos a tempo. Essa é a razão pela qual o PQ é introduzido. Esse recurso disponibiliza uma fila de emergência, que adota o modo FIFO de agendamento e não restringe banda além de ser capaz de prover serviços de prioridade máxima (Strict Priority) SP como serviços de voz. CBQ disponibilizando uma fila de emergência é chamado de LLQ (Low Latency Queue).

LLQ combina mecanismos SP com o CBQ, onde o usuário pode configurar uma classe para usar o serviço SP enquanto define as classes, uma classe dessas é chamada de classe prioritária (Priority Class) e todos os pacotes dessa classe entrarão na mesma fila prioritária. É necessário a verificação de restrição de banda para esses pacotes antes que entrem na fila. Quando os pacotes saem das filas, os pacotes da fila prioritária são encaminhandos primeiramente e depois os pacotes das outras filas são encaminhados no modo WF.

Para evitar que os pacotes nas outras filas não sejam atrasados em demasia, a largura máxima de banda pode ser disponibilizada para cada classe de prioridade quando se usa o LLQ. O valor de largura de banda é utilizado para monitorar o tráfego em casos de congestionamento. Caso nenhum congestionamento ocorra, permite-se a classe de prioridade a usar largura de banda que excede o limite especificado, caso ocorra congestionamento, os pacotes da classe de prioridade que excedam o limite de banda determinado serão descartados.

O sistema sempre irá definir a classe prioritária primeiramente e depois as outras classes no momento de aplicas as regras de identificação dos pacotes. Múltiplas classes prioritárias são identificadas na seqüência de configuração (HUAWEI, 2009. **Command Manual (V3.53_01);** TANENBAUM, 2003; COLCHER *et.al.,* 2005).

RTP (Real-time Transport Protocol):

Fila de prioridade RTP é utilizada para solucionar os problemas de QoS para serviços de tempo real (Incluindo serviços de AUDIO e VÍDEO). Seu princípio é a colocação dos pacotes RTP que carregam informações de áudio ou vídeo em uma fila de alta prioridade e enviá-los primeiro que os demais, assim diminuindo o delay e o Jitter e garantindo a qualidade do áudio ou vídeo que são serviços que possuem extrema sensibilidade ao atraso (delay).

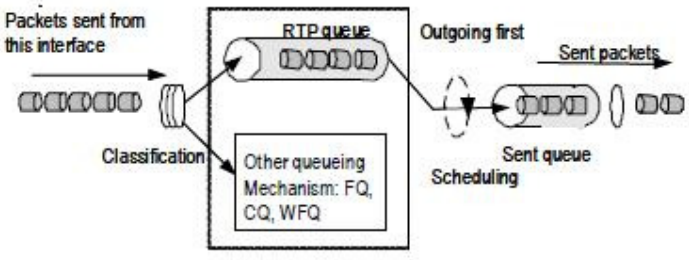


Figura 15 - RTP

Como mostrado na figura acima, um pacote RTP é enviado para uma fila de alta prioridade. O pacote RTP é um pacote UDP no qual o número da porta é uniforme. A faixa desse número de porta é configurável. Filas de prioridade RTP podem ser usadas com

qualquer mecanismo de enfileiramento (FIFO, PQ, CQ, WFQ e CBQ) enquanto possuir a prioridade mais alta. Desde que o LLQ no CBQ também pode ser usado pára tratar problemas com serviços de tempo real, é recomendável não usar o RTP juntamente com o CBQ (HUAWEI, 2009. **Operation Manual (V3.53_01);** TANENBAUM, 2003; COLCHER *et.al.,* 2005).

3.3 – SIP

3.3.1 – Visão geral

O SIP (Session Initiation Protocol) é um protocolo de camada de aplicação utilizado para estabelecer, modificar e finalizar chamadas ou sessões multimídias (TANENBAUM, 2003; COLCHER *et.al.,* 2005).

O SIP suporta cinco características que o auxiliam a estabelecer e finalizar comunicação multimedia.

- Localização de usuário: determinação do sistema final (ponta) que deverá ser usado na comunicação.
- Capacidades do usuário – determina a media e dos parametros de media que deverão ser usados.
- Disponibilidade do usuário – determina a disponibilidade da parte chamada para aceitar e inciar a comunicação.
- Organização da chamada – Estabelecimento dos parametros da chamada em ambos os lados o que origina e o que recebe a chamada.
- Controle da chamada: Estabelecimento, modificação e finalização de sessões multimídias ou chamadas (TANENBAUM, 2003; COLCHER *et.al.,* 2005).

3.3.2 – Conceitos básicos

- Cliente: Um cliente é um elemento de rede que envia solicitações SIP e recebe respostas SIP, os clientes podem ou não interagir com o usuário final.
- Servidor: Um servidor é o elemento de rede que recebe as solicitações SIP, as processa e envia respostas às solicitações recebidas.

- REQUEST (Solicitações): Uma mensagem SIP enviado de um cliente para um servidor, com o propósito de invocar uma operação particular.

- RESPONSE (Respostas): Uma mensagem SIP enviada de um servidor a um cliente, indicando o status da solicitação.

- TRANSAÇÃO: Uma transação SIP ocorre entre um cliente e um servidor, e corresponde a todas as mensagens trocadas, desde a primeira solicitação enviada pelo cliente até a última resposta do servidor.

- DIÁLOGO: Um diálogo é uma relação entre dois usuários (UA) que persiste durante um determinado intervalo de tempo, um diálogo é composto por mensagens SIP (TANENBAUM, 2003; COLCHER *et.al.,* 2005).

3.3.3 – Tipos de mensagens

REQUEST (SOLICITAÇÃO)

- INVITE: inicia a sessão.

- ACK: resposta ao INVITE

- CANCEL: Cancela a sessão

- BYE: termina a sessão

- REGISTER: regista em um servidor.

- OPTIONS: Solicita aos servidores informações sobre sua capacidade (TANENBAUM, 2003; COLCHER *et.al.,* 2005).

CAPÍTULO Nº 4 – METODOLOGIA E EXPERIMENTO PRÁTICO

4.1 – Descrição do ambiente de testes

O ambiente de testes é uma reprodução de duas redes interligadas por um link SERIAL (WAN).

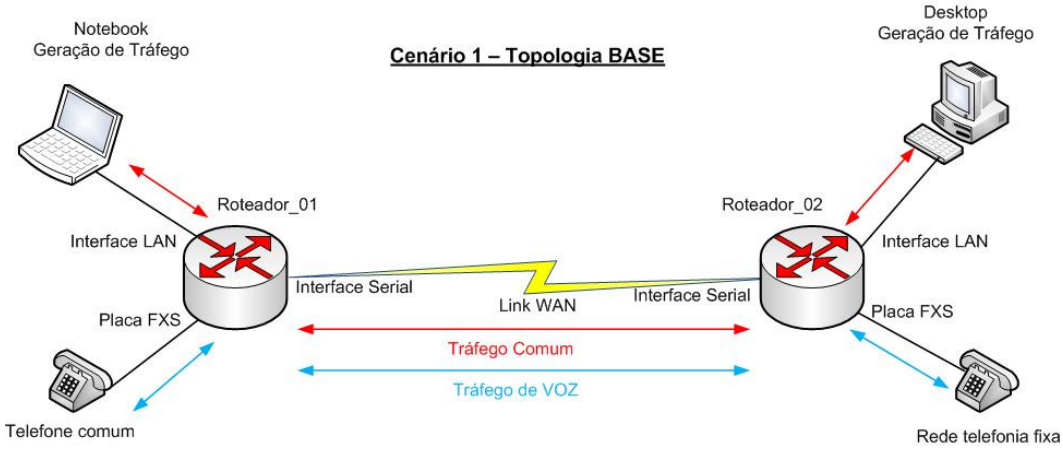


Figura 16 - Cenário 01

Abaixo seguem as configurações efetuadas nos roteadores visando estabelecer conectividade e roteamento básico entre as redes usadas no experimento.

CONECTIVIDADE:

ROTEADOR_01:

```
interface Ethernet0/0
 ip address 192.168.1.1 255.255.255.0
interface Serial0/0
 baudrate 2048000
 link-protocol ppp
 ip address 10.10.10.1 255.255.255.0
ip route-static 0.0.0.0 0.0.0.0 10.10.10.0 preference 60
```

ROTEADOR_02:

```
interface Ethernet0/0
 ip address 192.168.2.1 255.255.255.0
interface Serial0/0
 clock DTECLK1
 link-protocol ppp
 ip address 10.10.10.2 255.255.255.0
ip route-static 0.0.0.0 0.0.0.0 10.10.10.0 preference 60
```

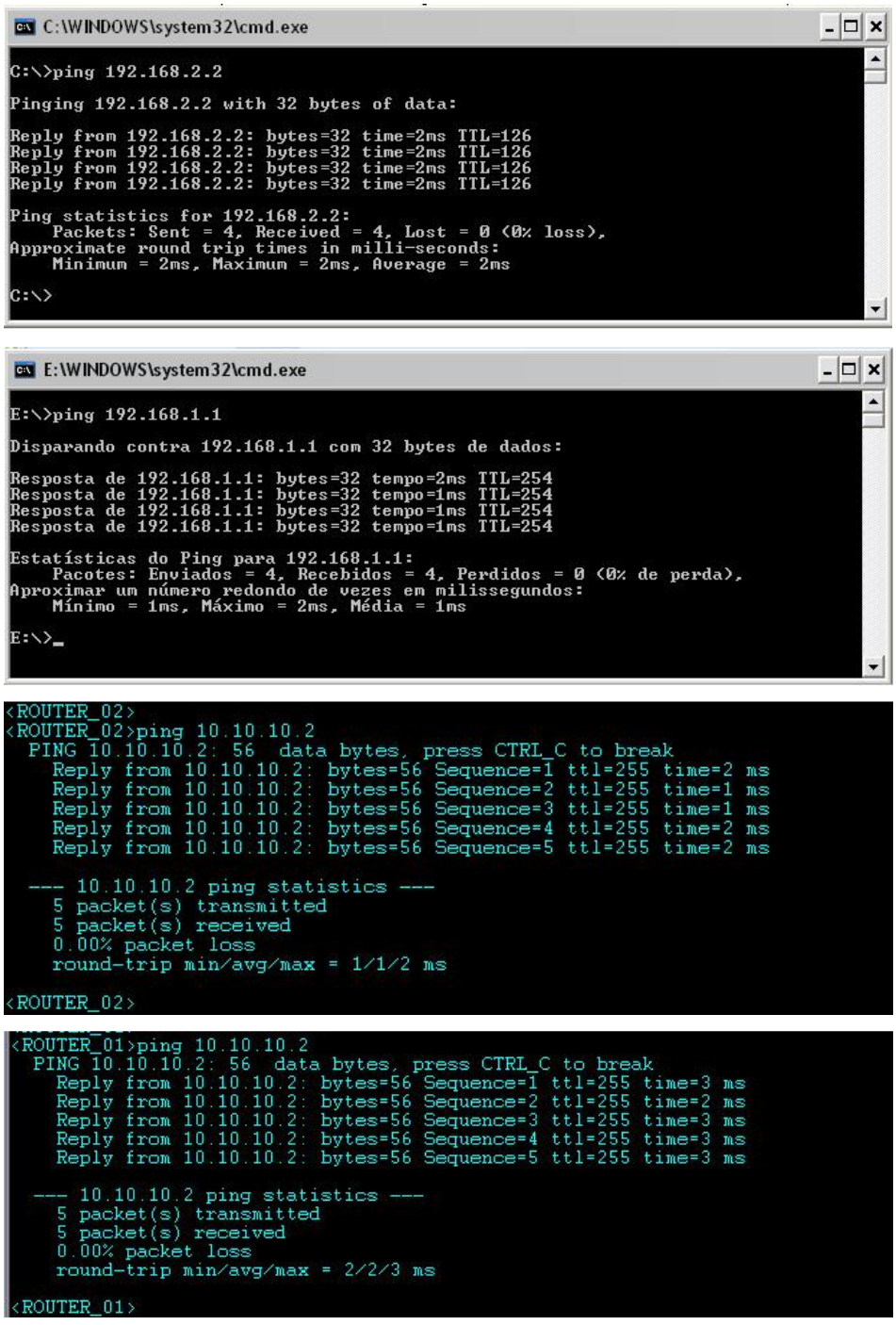


Figura 17 - Testes de Conectividade

4.2 - Componentes do ambiente de teste

4.2.1 - Roteadores

Modelo: AR2809

Fabricante: Huawei Technologies

Interfaces utilizadas:

- Fast Ethernet (integrada)
- Serial (integrada)
- FXS (Modular)

Papel dentro do experimento:

- Atuar como roteador de borda, provendo conectividade entre as redes utilizadas, suporte a voz, classificação e aplicação de políticas de QoS nos pacotes.

4.2.2 -Placa FXS (Foreign eXchange Station):

Modelo: 4FXS e 2FXS

Fabricante: Huawei Technologies

Papel dentro do experimento:

- Oferecer suporte a serviços de voz.

4.2.3- Switch

Modelo: S5600

Fabricante: Huawei Technologies

Papel dentro do experimento:

- Prover conectividade entre as redes utilizadas.

4.2.4 - Computadores

Serão utilizados dois computadores:

Desktop – Especificações: Intel(R) Core(TM)2 Duo CPU E7500 @ 2.93GHz / 3,50 GB de RAM

Papel dentro do experimento:

- Gerar diferentes tipos de tráfegos, com o intuito de congestionar o link WAN.

Notebook – Especificações:

Processador: Intel Core 2 Duo T5670 1.80Ghz / 3Gb RAM

Sistema Operacional: Windows XP – Service Pack 3

Papel dentro do experimento:

- Gerar diferentes tipos de tráfegos, com o intuito de congestionar o link WAN.

4.3 – Cenário 01 – Tráfego sem QoS

4.3.1 – Objetivos

Simular em laboratório problemas típicos apresentados em chamadas de voz.

Para isso será gerado tráfego de dados em ambos os sentidos entre o notebook e o desktop, com o intuito de gerar congestionamento esperado no link WAN.

Checar a qualidade da voz nos dois momentos distintos:

Tabela 1 – Cenário 1/Momento 1

CENÁRIO 01 - CHAMADAS REALIZADAS SEM USO DO QoS:			
Nr. Chamada:	**Resultado:**	**Duração:**	**Observações:**
1		1 Minuto	
2		1 Minuto	
3		1 Minuto	
4		1 Minuto	
5		1 Minuto	

Tabela 2 - Cenário 1/Momento 2

CENÁRIO 01 - CHAMADAS REALIZADAS DURANTE CONGESTIONAMENTO SEM QoS:			
Nr. Chamada:	**Resultado:**	**Duração:**	**Observações:**
1		1 Minuto	
2		1 Minuto	
3		1 Minuto	
4		1 Minuto	
5		1 Minuto	

4.3.2 – Topologia adotada

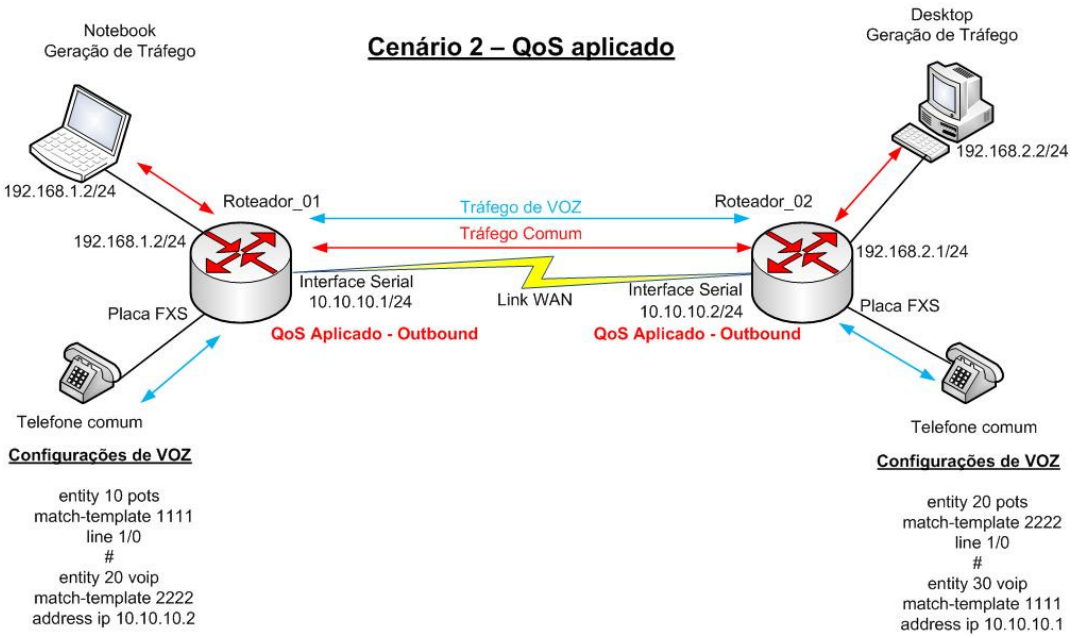


Figura 18 - Cenário 1 - QoS aplicado

4.3.3 – Condições prévias:

Equipamentos conectados e configurados de acordo com teste proposto.

Tabela 3 - Lista de testes prévios realizados

TESTES - CONECTIVIDADE			
Item de controle:	**Descrição**	**Resultado:**	**Observações:**
1	PING interfaces SERIAIS	**OK**	Ping realizado com sucesso sem perda de pacotes
2	PING interfaces ETH	**OK**	Ping realizado com sucesso sem perda de pacotes
3	PING entre Desktop e Notebook	**OK**	Ping realizado com sucesso sem perda de pacotes
4	Chamadas R1 para R2 (3 chamadas)	**OK**	Chamadas realizadas com sucesso - Qualidade OK
5	Chamadas R2 para R1 (3 chamadas)	**OK**	Chamadas realizadas com sucesso - Qualidade OK

4.3.4 – Procedimentos de teste:

- Confirmar conectividade e teste de chamada;
- Iniciar chamadas de voz entre roteadores;
- Verificar qualidade dessa chamada;
- Iniciar transferência de dados entre computadores (Congestionamento do link);
- Iniciar chamadas entre os roteadores;
- Verificar qualidade da chamada

- Checar e anotar resultados.

4.3.5 – Resultados Esperados/Obtidos

Tabela 4 – Cenário1/Momento 1: Resultados

CENÁRIO 01 - CHAMADAS REALIZADAS SEM USO DO QoS:		
Nr. Chamada:	**Resultado:**	**Observações:**
1	**OK**	Chamada realizada com sucesso - Qualidade OK
2	**OK**	Chamada realizada com sucesso - Qualidade OK
3	**OK**	Chamada realizada com sucesso - Qualidade OK
4	**OK**	Chamada realizada com sucesso - Qualidade OK
5	**OK**	Chamada realizada com sucesso - Qualidade OK

➔ Os resultados esperados para o momento 01 foram atingidos com sucesso.

Tabela 5 - Cenário1Momento 2: Resultados

CENÁRIO 01 - CHAMADAS REALIZADAS DURANTE CONGESTIONAMENTO SEM QoS:			
Nr. Chamada:	**Resultado:**	**Duração:**	**Observações:**
1	**NOK**	1 Minuto	Chamada estabelecida, porém conversação e qualidade seriamente comprometidos
2	**NOK**	1 Minuto	Chamada estabelecida, porém conversação e qualidade seriamente comprometidos
3	**NOK**	1 Minuto	Chamada estabelecida, porém conversação e qualidade seriamente comprometidos
4	**NOK**	1 Minuto	Chamada estabelecida, porém conversação e qualidade seriamente comprometidos
5	**NOK**	1 Minuto	Chamada estabelecida, porém conversação e qualidade seriamente comprometidos

➔ Os resultados esperados para o momento 02 foram atingidos com sucesso.

Anotações:

- Ao iniciar a transferência de dados a chamada apresentou:
- Falhas de voz (som da voz falhando constantemente), culminando com o não entendimento da mensagem;
- Picotamento da voz;
- Dificuldade de entendimento durante a conversação;
- Problemas descritos acima ocorreram durante toda a duração das chamadas.

4.3.6 – Configurações utilizadas

ROTEADOR_01:

```
interface Ethernet0/0
ip address 192.168.1.1 255.255.255.0
#
interface Serial0/0
baudrate 2048000
link-protocol ppp
ip address 10.10.10.1 255.255.255.0
```

```
voice-setup
#
aaa-client
#
dial-program
#
entity 10 pots
match-template 1111
line 1/0
#
entity 20 voip
match-template 2222
address ip 10.10.10.2
#
gk-client
#
sip
#
subscriber-line 1/0
#
subscriber-line 1/1
#
subscriber-line 1/2
#
subscriber-line 1/3
ip route-static 0.0.0.0 0.0.0.0 10.10.10.0 preference 60

ROTEADOR_02:
interface Ethernet0/0
ip address 192.168.2.1 255.255.255.0
#
interface Serial0/0
clock DTECLK1
link-protocol ppp
ip address 10.10.10.2 255.255.255.0
voice-setup
#
aaa-client
#
dial-program
#
entity 20 pots
match-template 2222
line 1/0
#
entity 30 voip
match-template 1111
address ip 10.10.10.1
#
gk-client
```

```
#
sip
#
subscriber-line 1/0
#
subscriber-line 1/1
ip route-static 0.0.0.0 0.0.0.0 10.10.10.0 preference 60
```

4.4 – Cenário 02 – Tráfego com QoS (PQ – Priority Queue)

4.4.1 – Objetivos

Configurar e aplicar QoS – PQ (Priority Queue) nas interfaces de saída dos pacotes.

Gerar tráfego de dados entre o notebook e o desktop, com o intuito de gerar congestionamento no link WAN.

Simular, em laboratório, chamadas de voz entre os roteadores;

Checar a qualidade da voz em três momentos distintos.

Momento 1: Checar a qualidade da chamada sem congestionamento do link WAN.

- Tabela similar a inserida no Cenário 01

Momento 2: Checar a qualidade da chamada durante congestionamento do link WAN.

- Tabela similar a inserida no Cenário 01

Tabela 6 – Cenário2/Momento 3

CENÁRIO 02 - CHAMADAS REALIZADAS DURANTE CONGESTIONAMENTO UTILIZANDO QoS - PQ			
Nr. Chamada:	**Resultado:**	**Duração:**	**Observações:**
1		1 Minuto	
2		1 Minuto	
3		1 Minuto	
4		1 Minuto	
5		1 Minuto	

4.4.2 – Topologia adotada:

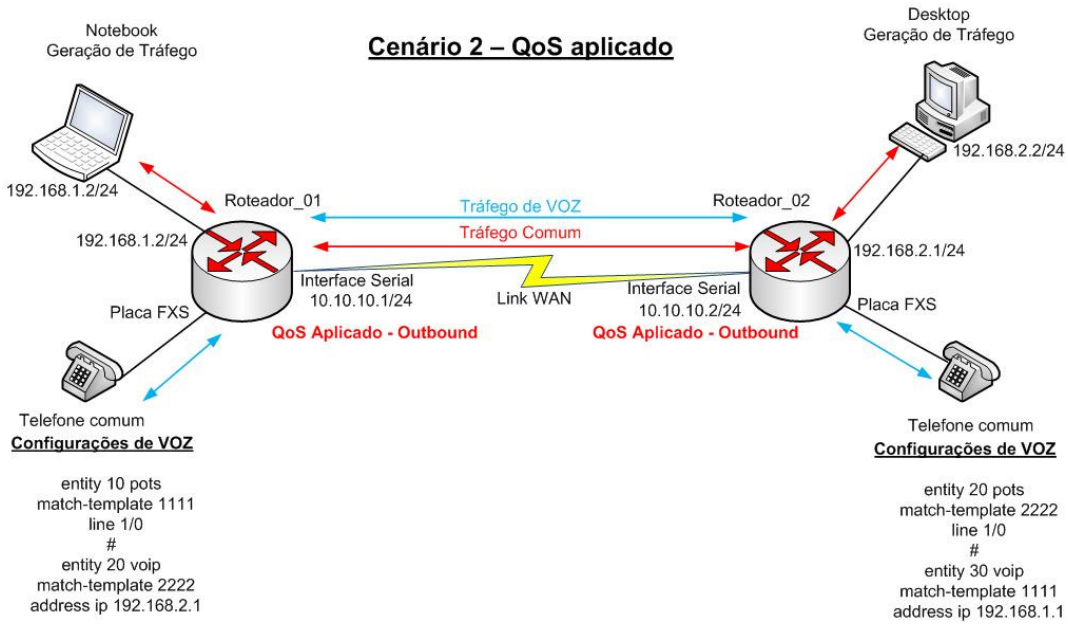


Figura 19 - Cenário 2 - QoS aplicado

4.4.3 – Condições prévias:

Equipamentos conectados e configurados de acordo com teste proposto.

Tabela 7 - Lista de testes prévios realizados

TESTES - CONECTIVIDADE			
Item de controle:	**Descrição**	**Resultado:**	**Observações:**
1	PING interfaces SERIAIS	**OK**	Ping realizado com sucesso sem perda de pacotes
2	PING interfaces ETH	**OK**	Ping realizado com sucesso sem perda de pacotes
3	PING entre Desktop e Notebook	**OK**	Ping realizado com sucesso sem perda de pacotes
4	Chamadas R1 para R2 (3 chamadas)	**OK**	Chamadas realizadas com sucesso - Qualidade OK
5	Chamadas R2 para R1 (3 chamadas)	**OK**	Chamadas realizadas com sucesso - Qualidade OK

4.4.4– Procedimento de teste:

- Confirmar conectividade e teste de chamada;
- Iniciar chamadas de voz entre roteadores;
- Verificar qualidade dessa chamada;
- Aplicar política de QoS definida (PQ – Priority Queue);
- Iniciar transferência de dados entre computadores (Congestionamento do link);
- Iniciar chamadas entre os roteadores;

- Verificar qualidade da chamada
- Checar e anotar resultados.

4.4.5 – Resultados Esperados/Obtidos:

Tabela 8 - Cenário1/Momento 1

CENÁRIO 02 - CHAMADAS REALIZADAS SEM USO DO QoS:		
Nr. Chamada:	**Resultado:**	**Observações:**
1	**OK**	Chamada realizada com sucesso - Qualidade OK
2	**OK**	Chamada realizada com sucesso - Qualidade OK
3	**OK**	Chamada realizada com sucesso - Qualidade OK
4	**OK**	Chamada realizada com sucesso - Qualidade OK
5	**OK**	Chamada realizada com sucesso - Qualidade OK

➔ Os resultados esperados para o momento 01 foram atingidos com sucesso.

Tabela 9 - Cenário1/Momento 2

CENÁRIO 02 - CHAMADAS REALIZADAS DURANTE CONGESTIONAMENTO SEM QoS:			
Nr. Chamada:	**Resultado:**	**Duração:**	**Observações:**
1	**NOK**	1 Minuto	Chamada estabelecida porém conversação e qualidade seriamente comprometidos
2	**NOK**	1 Minuto	Chamada estabelecida porém conversação e qualidade seriamente comprometidos
3	**NOK**	1 Minuto	Chamada estabelecida porém conversação e qualidade seriamente comprometidos
4	**NOK**	1 Minuto	Chamada estabelecida porém conversação e qualidade seriamente comprometidos
5	**NOK**	1 Minuto	Chamada estabelecida porém conversação e qualidade seriamente comprometidos

➔ Os resultados esperados para o momento 02 foram atingidos com sucesso.

Tabela 10 - Cenário1/Momento 3

CENÁRIO 02 - CHAMADAS REALIZADAS DURANTE CONGESTIONAMENTO UTILIZANDO QoS - PQ			
Nr. Chamada:	**Resultado:**	**Duração:**	**Observações:**
1	**OK**	1 Minuto	Mesmo durante congestionamento as chamadas apresentaram qualidade - OK
2	**OK**	1 Minuto	Mesmo durante congestionamento as chamadas apresentaram qualidade - OK
3	**OK**	1 Minuto	Mesmo durante congestionamento as chamadas apresentaram qualidade - OK
4	**OK**	1 Minuto	Mesmo durante congestionamento as chamadas apresentaram qualidade - OK
5	**OK**	1 Minuto	Mesmo durante congestionamento as chamadas apresentaram qualidade - OK

➔ Os resultados esperados para o momento 03 foram atingidos com sucesso.

Anotações:

- Ao iniciar a transferência de dados a chamada apresentou o mesmo comportamento relatado no cenário 01 (Problemas de voz);
- Após a aplicação das políticas de QoS nas interfaces:
- Não houve picotamento na voz;
- A mensagem estava clara e de perfeito entendimento para ambos os lados.

4.4.6 – Configurações utilizadas

ROTEADOR_01:

```
[ROUTER_01]dis cur
#
 sysname ROUTER_01
#
 super password level 3 simple huawei
#
 qos pql 1 queue top queue-length 60
 qos pql 1 queue bottom queue-length 100
 qos pql 1 protocol ip acl 3300 queue top
 qos pql 1 protocol ip acl 3301 queue bottom
#
 flow-interval qos 1
#
radius scheme system
#
domain system
#
local-user admin
 password cipher .]@USE=B,53Q=^Q`MAF4<1!!
 service-type telnet terminal
 level 3
 service-type ftp
#
acl number 3300
 rule 0 permit udp source-port range 16384 32768 destination-port range 16384 32768
acl number 3301
 rule 0 permit ip
#
interface Aux0
 async mode flow
#
interface Ethernet0/0
 ip address 192.168.1.1 255.255.255.0
#
interface Serial0/0
 baudrate 512000
 link-protocol ppp
 ip address 10.10.10.1 255.255.255.0
 qos pq pql 1
#
interface NULL0
#
#
voice-setup
 #
 aaa-client
 #
```

```
 dial-program
  #
  entity 10 pots
    match-template 1111
    line 1/0
  #
  entity 20 voip
    match-template 2222
    address ip 10.10.10.2
 #
 gk-client
 #
 sip
 #
 subscriber-line 1/0
 #
 subscriber-line 1/1
 #
 subscriber-line 1/2
 #
 subscriber-line 1/3
#
 FTP server enable
#
 ip route-static 0.0.0.0 0.0.0.0 10.10.10.0 preference 60
#
 snmp-agent
 snmp-agent local-engineid 000007DB7F00000100006527
 snmp-agent community read public
 snmp-agent sys-info version all
#
user-interface con 0
user-interface aux 0
user-interface vty 0 4
 authentication-mode scheme
#
```

ROTEADOR_02:

```
<ROUTER_02>dis cur
#
 sysname ROUTER_02
#
 cpu-usage cycle 1min
#
 qos pql 1 queue top queue-length 60
 qos pql 1 queue bottom queue-length 100
 qos pql 1 protocol ip acl 3300 queue top
 qos pql 1 protocol ip acl 3301 queue bottom
#
radius scheme system
#
```

```
domain system
#
local-user admin
 password cipher .]@USE=B,53Q=^Q`MAF4<1!!
 service-type telnet terminal
 level 3
 service-type ftp
#
acl number 3300
 rule 0 permit udp source-port range 16384 32768 destination-port range 16384 32768
acl number 3301
 rule 0 permit ip
#
interface Aux0
 async mode flow
#
interface Ethernet0/0
 ip address 192.168.2.1 255.255.255.0
#
interface Serial0/0
 clock DTECLK1
 link-protocol ppp
 virtualbaudrate 512000
 ip address 10.10.10.2 255.255.255.0
 qos pq pql 1
#
interface NULL0
#
#
voice-setup
 #
 aaa-client
 #
 dial-program
  #
  entity 20 pots
   match-template 2222
   line 1/0
  #
  entity 30 voip
   match-template 1111
   address ip 10.10.10.1
 #
 gk-client
 #
 sip
 #
 subscriber-line 1/0
 #
 subscriber-line 1/1
```

```
#
 FTP server enable
#
 ip route-static 0.0.0.0 0.0.0.0 10.10.10.0 preference 60
#
user-interface con 0
user-interface aux 0
user-interface vty 0 4
 authentication-mode scheme
#
return
<ROUTER_02>
```

4.5 – Cenário 03 – Tráfego com QoS (WFQ – Weighted Fair Queuing):

4.5.1 – Objetivos:

Configurar e aplicar QoS (WFQ - Weighted Fair Queuing) nas interfaces de saída dos pacotes.

Gerar tráfego de dados entre o notebook e o desktop, com o intuito de gerar congestionamento no link WAN.

Simular em laboratório, chamadas de voz entre os roteadores;

Checar a qualidade da voz em três momentos distintos.

Momento 1: Checar a qualidade da chamada sem congestionamento do link WAN.

- Tabela similar a inserida no Cenário 01

Momento 2: Checar a qualidade da chamada durante congestionamento do link WAN.

- Tabela similar a inserida no Cenário 01

Momento 3: Checar a qualidade da chamada durante congestionamento utilizando QoS:

- Tabela similar a inserida no Cenário 01

4.5.2 – Topologia adotada:

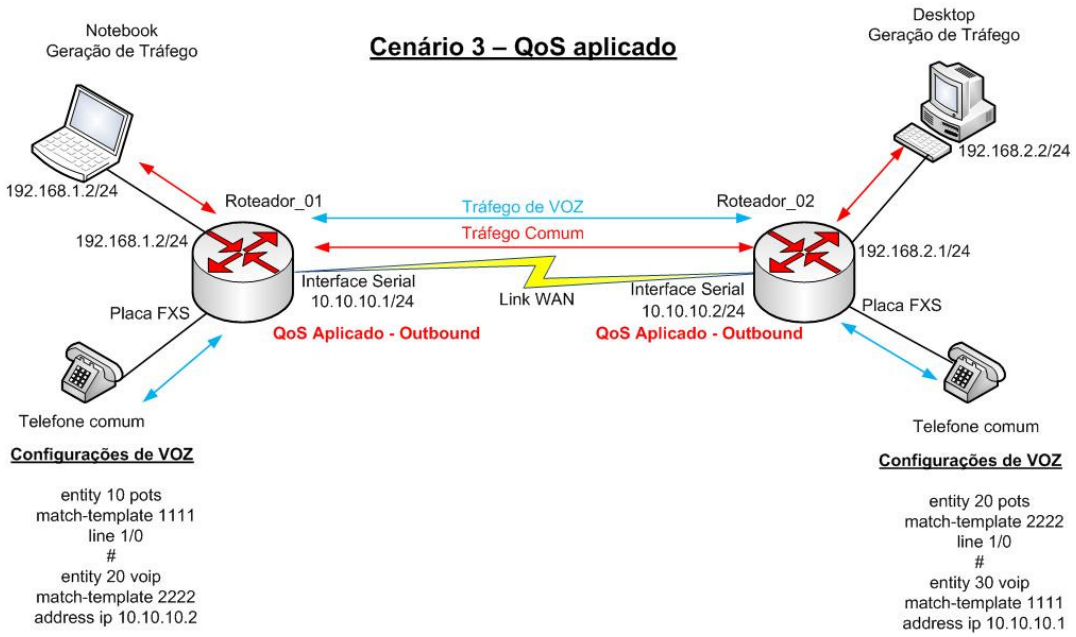


Figura 20 - Cenário 3 - QoS aplicado

4.5.3 – Condições prévias:

Equipamentos conectados e configurados de acordo com teste proposto.

Tabela 11 - Lista de testes prévios realizados

TESTES - CONECTIVIDADE			
Item de controle:	**Descrição**	**Resultado:**	**Observações:**
1	PING interfaces SERIAIS	**OK**	Ping realizado com sucesso sem perda de pacotes
2	PING interfaces ETH	**OK**	Ping realizado com sucesso sem perda de pacotes
3	PING entre Desktop e Notebook	**OK**	Ping realizado com sucesso sem perda de pacotes
4	Chamadas R1 para R2 (3 chamadas)	**OK**	Chamadas realizadas com sucesso - Qualidade OK
5	Chamadas R2 para R1 (3 chamadas)	**OK**	Chamadas realizadas com sucesso - Qualidade OK

4.5.4 – Procedimento de teste:

- Confirmar conectividade e teste de chamada;
- Iniciar chamadas de voz entre roteadores;
- Verificar qualidade dessa chamada;
- Aplicar política de QoS definida (PQ – Priority Queue);
- Iniciar transferência de dados entre computadores (Congestionamento do link);

- Iniciar chamadas entre os roteadores;
- Verificar qualidade da chamada
- Checar e anotar resultados.

4.5.5 – Resultados Esperados/Obtidos:

Tabela 12 - Cenário 03/Momento 1

CENÁRIO 03 - CHAMADAS REALIZADAS SEM USO DO QoS:		
Nr. Chamada:	**Resultado:**	**Observações:**
1	**OK**	Chamada realizada com sucesso - Qualidade OK
2	**OK**	Chamada realizada com sucesso - Qualidade OK
3	**OK**	Chamada realizada com sucesso - Qualidade OK
4	**OK**	Chamada realizada com sucesso - Qualidade OK
5	**OK**	Chamada realizada com sucesso - Qualidade OK

➔ Os resultados esperados para o momento 01 foram atingidos com sucesso.

Tabela 13 - Cenário 03/Momento 2 Checar a qualidade da chamada durante congestionamento do link WAN.

CENÁRIO 03 - CHAMADAS REALIZADAS DURANTE CONGESTIONAMENTO SEM QoS:			
Nr. Chamada:	**Resultado:**	**Duração:**	**Observações:**
1	**NOK**	1 Minuto	Chamada estabelecida porém conversação e qualidade seriamente comprometidos
2	**NOK**	1 Minuto	Chamada estabelecida porém conversação e qualidade seriamente comprometidos
3	**NOK**	1 Minuto	Chamada estabelecida porém conversação e qualidade seriamente comprometidos
4	**NOK**	1 Minuto	Chamada estabelecida porém conversação e qualidade seriamente comprometidos
5	**NOK**	1 Minuto	Chamada estabelecida porém conversação e qualidade seriamente comprometidos

➔ Os resultados esperados para o momento 02 foram atingidos com sucesso.

Tabela 14 - Cenário 03/Momento 3

CENÁRIO 03 - CHAMADAS REALIZADAS DURANTE CONGESTIONAMENTO UTILIZANDO QoS - WFQ			
Nr. Chamada:	**Resultado:**	**Duração:**	**Observações:**
1	**OK**	1 Minuto	Mesmo durante congestionamento as chamadas apresentaram qualidade - OK
2	**OK**	1 Minuto	Mesmo durante congestionamento as chamadas apresentaram qualidade - OK
3	**OK**	1 Minuto	Mesmo durante congestionamento as chamadas apresentaram qualidade - OK
4	**OK**	1 Minuto	Mesmo durante congestionamento as chamadas apresentaram qualidade - OK
5	**OK**	1 Minuto	Mesmo durante congestionamento as chamadas apresentaram qualidade - OK

➔ Os resultados esperados para o momento 03 foram atingidos com sucesso.

Anotações:

- Ao iniciar a transferência de dados a chamada apresentou o mesmo comportamento relatado no cenário 01 (Problemas de voz);
- Após a aplicação das políticas de QoS nas interfaces:
- Não houve picotamento na voz;

- A mensagem estava clara e de perfeito entendimento para ambos os lados.

4.5.6 – Configurações utilizadas

ROTEADOR 01:

```
<ROUTER_01>dis cur
#
 sysname ROUTER_01
#
 super password level 3 simple huawei
#
 flow-interval qos 1
#
radius scheme system
#
domain system
#
local-user admin
 password cipher .]@USE=B,53Q=^Q`MAF4<1!!
 service-type telnet terminal
 level 3
 service-type ftp
#
traffic classifier VOZ operator and
 if-match rtp start-port 2000 end-port 65535
#
traffic behavior VOZ
  remark dscp ef
  queue ef bandwidth 64 cbs 1500
#
qos policy TESTE
 classifier VOZ behavior VOZ
#
interface Aux0
 async mode flow
#
interface Ethernet0/0
 ip address 192.168.1.1 255.255.255.0
#
interface Serial0/0
 baudrate 512000
 link-protocol ppp
 qos max-bandwidth 512000
 ip address 10.10.10.1 255.255.255.0
 qos apply policy TESTE outbound
#
interface NULL0
#
```

```
#
voice-setup
 #
 aaa-client
 #
 dial-program
  #
  entity 10 pots
   match-template 1111
   line 1/0
  #
  entity 20 voip
   match-template 2222
   address ip 10.10.10.2
 #
 gk-client
 #
 sip
 #
 subscriber-line 1/0
 #
 subscriber-line 1/1
 #
 subscriber-line 1/2
 #
 subscriber-line 1/3
#
 FTP server enable
#
 ip route-static 0.0.0.0 0.0.0.0 10.10.10.0 preference 60
#
 snmp-agent
 snmp-agent local-engineid 000007DB7F00000100006527
 snmp-agent community read public
 snmp-agent sys-info version all
#
user-interface con 0
```

ROTEADOR 02:

```
<ROUTER_02>dis cur
#
 sysname ROUTER_02
#
 cpu-usage cycle 1min
#
radius scheme system
#
domain system
#
local-user admin
 password cipher .]@USE=B,53Q=^Q`MAF4<1!!
```

```
 service-type telnet terminal
 level 3
 service-type ftp
#
traffic classifier VOZ operator and
 if-match rtp start-port 2000 end-port 65535
#
traffic behavior VOZ
 remark dscp ef
 queue ef bandwidth 64 cbs 1500
#
qos policy TESTE
 classifier VOZ behavior VOZ
#
interface Aux0
 async mode flow
#
interface Ethernet0/0
 ip address 192.168.2.1 255.255.255.0
#
interface Serial0/0
 clock DTECLK1
 link-protocol ppp
 virtualbaudrate 512000
 ip address 10.10.10.2 255.255.255.0
 qos apply policy TESTE outbound
#
interface NULL0
#
#
voice-setup
 #
 aaa-client
 #
 dial-program
  #
  entity 20 pots
   match-template 2222
   line 1/0
  #
  entity 30 voip
   match-template 1111
   address ip 10.10.10.1
 #
 gk-client
 #
 sip
 #
 subscriber-line 1/0
```

CAPÍTULO 5 – RESULTADOS

5.1 – Cenário 01 – Tráfego sem QoS

5.1.1 – Topologia adotada

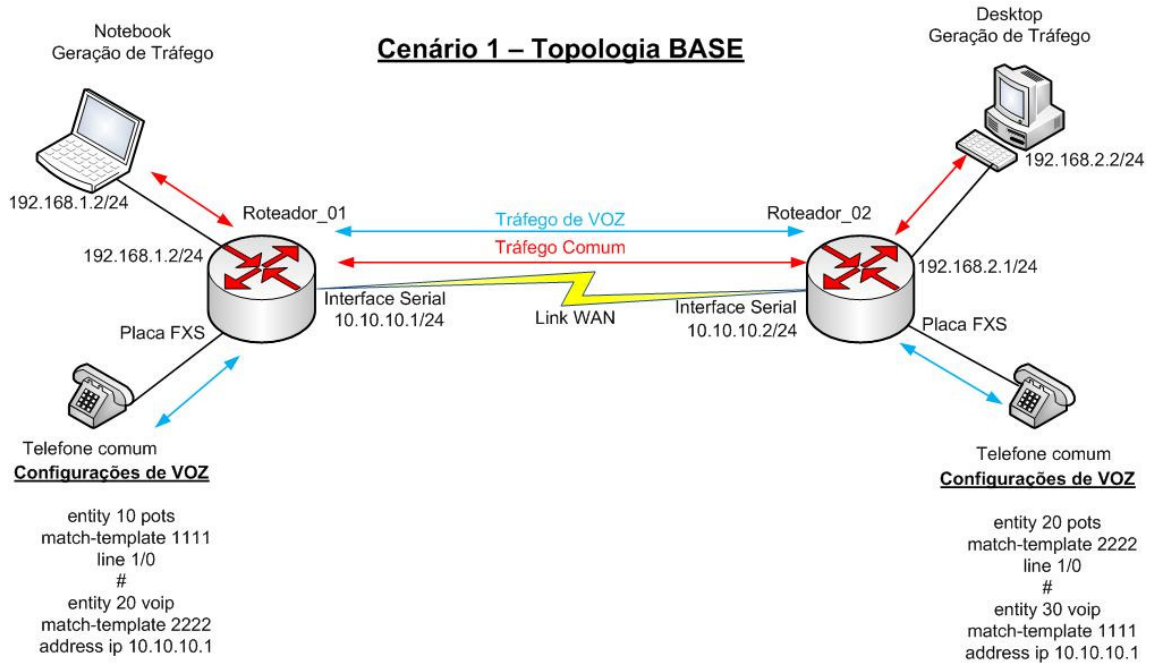


Figura 21 - Cenário 1

5.1.2 – Resultados Esperados/Obtidos

Tabela 15 – Cenário 1 - Momento 1: Resultados obtidos

CENÁRIO 01 - CHAMADAS REALIZADAS SEM USO DO QoS:		
Nr. Chamada:	**Resultado:**	**Observações:**
1	**OK**	Chamada realizada com sucesso - Qualidade OK
2	**OK**	Chamada realizada com sucesso - Qualidade OK
3	**OK**	Chamada realizada com sucesso - Qualidade OK
4	**OK**	Chamada realizada com sucesso - Qualidade OK
5	**OK**	Chamada realizada com sucesso - Qualidade OK

➔ Os resultados esperados para o momento 01 foram atingidos com sucesso.

Tabela 16 – Cenário 01 - Momento 2: Resultados obtidos

CENÁRIO 01 - CHAMADAS REALIZADAS DURANTE CONGESTIONAMENTO SEM QoS:			
Nr. Chamada:	**Resultado:**	**Duração:**	**Observações:**
1	**NOK**	1 Minuto	Chamada estabelecida, porém conversação e qualidade seriamente comprometidos
2	**NOK**	1 Minuto	Chamada estabelecida, porém conversação e qualidade seriamente comprometidos
3	**NOK**	1 Minuto	Chamada estabelecida, porém conversação e qualidade seriamente comprometidos
4	**NOK**	1 Minuto	Chamada estabelecida, porém conversação e qualidade seriamente comprometidos
5	**NOK**	1 Minuto	Chamada estabelecida, porém conversação e qualidade seriamente comprometidos

➔ Os resultados esperados para o momento 02 foram atingidos com sucesso.

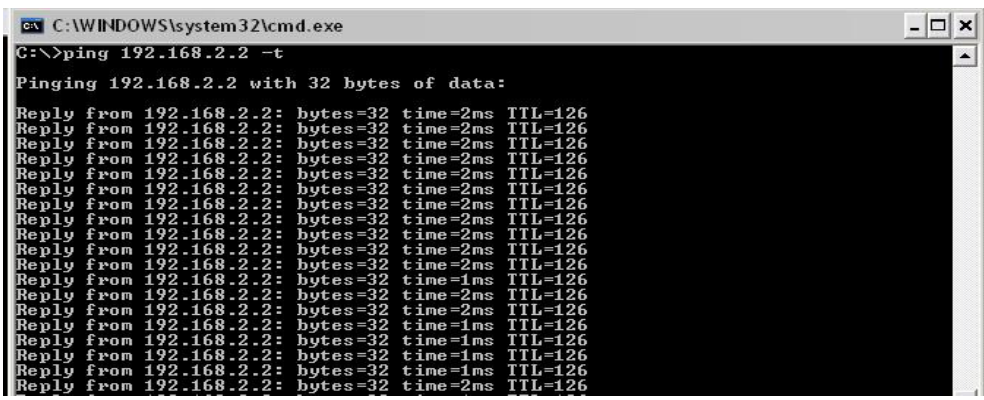
Anotações:

- Ao iniciar a transferência de dados a chamada apresentou:
- Falhas de voz (som da voz falhando constantemente), culminando com o não entendimento da mensagem;
- Picotamento da voz;
- Dificuldade de entendimento durante a conversação;
- Problemas descritos acima ocorreram durante toda a duração das chamadas.

5.1.3 Dados coletados

5.1.3.1 PING momento 01

Foram efetuados PINGs entre os terminais (Notebook e Desktop) para verificação de conectividade e condições da rede sem congestionamento.



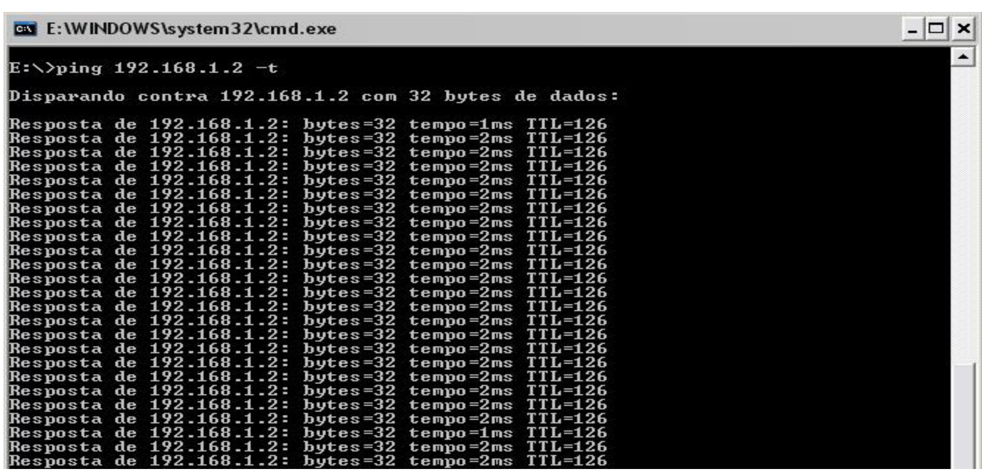


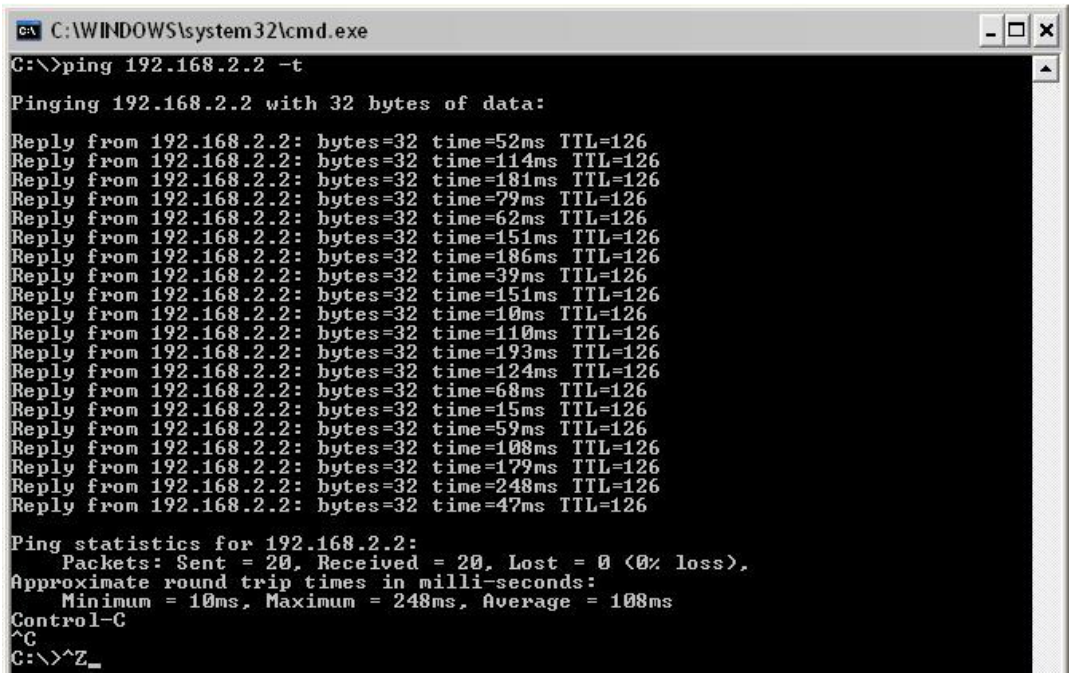
Figura 22 - PING Cenário 1 - Sem congestionamento

Nota-se que o tempo de resposta está vairando entre 1ms e 2ms, o que é considerável excelente para qualquer serviço de rede.

5.1.3.2 PING – Momento 02:

Iniciado a transferência de arquivos – Link WAN congestionado.

Os mesmos testes de PING foram realizados, seguem resultados:



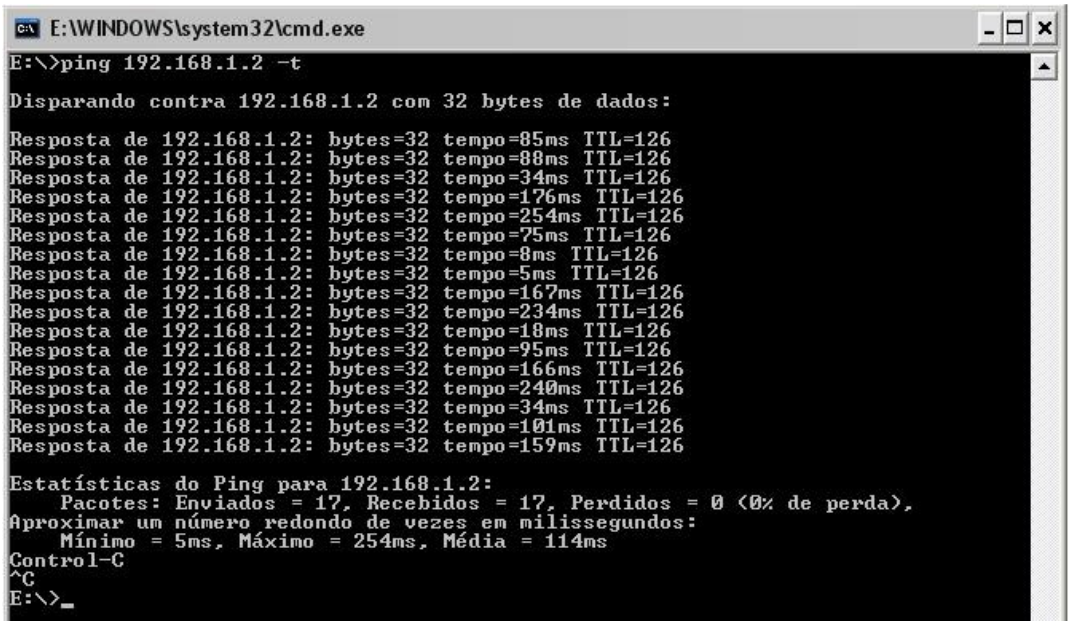


Figura 23 - Cenário 01 - PING com congestionamento

Nota-se claramente que o tempo de resposta subiu muito atingindo picos de 240ms, o que é não é aceitável para serviços de voz (RTP) em uma rede.

5.1.4 - Historico de ligações – Momento 01

Seguem informações retiradas dos roteadores referentes às chamadas realizadas durante os testes.

```
<ROUTER_02>dis voice call-history-record last 5
 #
 CallRecord :
   CallerNum            = 1111
   CalledNum            = 2222
   EncodeType           = 729r8
   PeerAddress          = 10.10.10.1
   DisconnectCause      = 12
   DisconnectText       = Caller hook on
   TalkingTimes         = 00h 01m 52s
   VoiceTimes           = 00h 01m 52s
   FaxTimes             = 00h 00m 00s
   ImgPages             = 0
   CallDirection        = 2
   SetupTime(voip)      = Jun 19, 2010 22:00:33
   ConnectTime(voip)    = Jun 19, 2010 22:00:34
   DisconectTime(voip)  = Jun 19, 2010 22:02:26
   Transmit(voip)       = 3729 (packages): 156618 (bytes)
   Received(voip)       = 3729 (packages): 156618 (bytes)
   SetupTime(pstn)      = Jun 19, 2010 22:00:33
   ConnectTime(pstn)    = Jun 19, 2010 22:00:33
   DisconectTime(pstn)  = Jun 19, 2010 22:02:26
   Transmit(pstn)       = 3729 (packages): 111870 (bytes)
   Received(pstn)       = 3729 (packages): 111870 (bytes)

 #
 CallRecord :
   CallerNum            = 1111
   CalledNum            = 2222
   EncodeType           = 729r8
   PeerAddress          = 10.10.10.1
   DisconnectCause      = 12
   DisconnectText       = Caller hook on
   TalkingTimes         = 00h 01m 22s
   VoiceTimes           = 00h 01m 22s
   FaxTimes             = 00h 00m 00s
   ImgPages             = 0
   CallDirection        = 2
   SetupTime(voip)      = Jun 19, 2010 21:59:07
   ConnectTime(voip)    = Jun 19, 2010 21:59:08
```

```
  DisconectTime(voip)     = Jun 19, 2010 22:00:30
  Transmit(voip)          = 2722 (packages): 114324 (bytes)
  Received(voip)          = 2721 (packages): 114282 (bytes)
  SetupTime(pstn)         = Jun 19, 2010 21:59:07
  ConnectTime(pstn)       = Jun 19, 2010 21:59:07
  DisconectTime(pstn)     = Jun 19, 2010 22:00:30
  Transmit(pstn)          = 2721 (packages): 81630 (bytes)
  Received(pstn)          = 2722 (packages): 81660 (bytes)

#
CallRecord :
  CallerNum               = 1111
  CalledNum               = 2222
  EncodeType              = 729r8
  PeerAddress             = 10.10.10.1
  DisconnectCause         = 12
  DisconnectText          = Caller hook on
  TalkingTimes            = 00h 00m 56s
  VoiceTimes              = 00h 00m 56s
  FaxTimes                = 00h 00m 00s
  ImgPages                = 0
  CallDirection           = 2
  SetupTime(voip)         = Jun 19, 2010 21:57:58
  ConnectTime(voip)       = Jun 19, 2010 21:57:59
  DisconectTime(voip)     = Jun 19, 2010 21:58:55
  Transmit(voip)          = 1848 (packages): 77616 (bytes)
  Received(voip)          = 1848 (packages): 77616 (bytes)
  SetupTime(pstn)         = Jun 19, 2010 21:57:58
  ConnectTime(pstn)       = Jun 19, 2010 21:57:58
  DisconectTime(pstn)     = Jun 19, 2010 21:58:55
  Transmit(pstn)          = 1848 (packages): 55440 (bytes)
  Received(pstn)          = 1848 (packages): 55440 (bytes)

#
CallRecord :
  CallerNum               = 1111
  CalledNum               = 2222
  EncodeType              = 729r8
  PeerAddress             = 10.10.10.1
  DisconnectCause         = 12
  DisconnectText          = Caller hook on
  TalkingTimes            = 00h 01m 26s
  VoiceTimes              = 00h 01m 26s
  FaxTimes                = 00h 00m 00s
  ImgPages                = 0
  CallDirection           = 2
  SetupTime(voip)         = Jun 19, 2010 21:56:25
  ConnectTime(voip)       = Jun 19, 2010 21:56:26
  DisconectTime(voip)     = Jun 19, 2010 21:57:52
  Transmit(voip)          = 2851 (packages): 119742 (bytes)
```

```
  Received(voip)          = 2850 (packages): 119700 (bytes)
  SetupTime(pstn)         = Jun 19, 2010 21:56:25
  ConnectTime(pstn)       = Jun 19, 2010 21:56:25
  DisconectTime(pstn)     = Jun 19, 2010 21:57:52
  Transmit(pstn)          = 2850 (packages): 85500 (bytes)
  Received(pstn)          = 2851 (packages): 85530 (bytes)

 #
 CallRecord :
  CallerNum               = 1111
  CalledNum               = 2222
  EncodeType              = 729r8
  PeerAddress             = 10.10.10.1
  DisconnectCause         = 13
  DisconnectText          = Called hook on
  TalkingTimes            = 00h 00m 19s
  VoiceTimes              = 00h 00m 19s
  FaxTimes                = 00h 00m 00s
  ImgPages                = 0
  CallDirection           = 2
  SetupTime(voip)         = Jun 19, 2010 21:42:31
  ConnectTime(voip)       = Jun 19, 2010 21:42:32
  DisconectTime(voip)     = Jun 19, 2010 21:42:51
  Transmit(voip)          = 565 (packages): 23730 (bytes)
  Received(voip)          = 561 (packages): 23562 (bytes)
  SetupTime(pstn)         = Jun 19, 2010 21:42:31
  ConnectTime(pstn)       = Jun 19, 2010 21:42:31
  DisconectTime(pstn)     = Jun 19, 2010 21:42:51
  Transmit(pstn)          = 561 (packages): 16830 (bytes)
  Received(pstn)          = 565 (packages): 16950 (bytes)

<ROUTER_01>dis voice call-history-record last 5
 #
 CallRecord [ 17]:
  CallerNum               = 1111
  CalledNum               = 2222
  EncodeType              = 729r8
  PeerAddress             = 10.10.10.2
  DisconnectCause         = 12
  DisconnectText          = Caller hook on
  TalkingTimes            = 00h 01m 52s
  VoiceTimes              = 00h 01m 52s
  FaxTimes                = 00h 00m 00s
  ImgPages                = 0
  CallDirection           = 1
  SetupTime(voip)         = Jun 19, 2010 22:00:01
  ConnectTime(voip)       = Jun 19, 2010 22:00:02
  DisconectTime(voip)     = Jun 19, 2010 22:01:54
  Transmit(voip)          = 3729 (packages): 156618 (bytes)
  Received(voip)          = 3728 (packages): 156576 (bytes)
```

```
  SetupTime(pstn)        = Jun 19, 2010 21:59:59
  ConnectTime(pstn)      = Jun 19, 2010 21:59:59
  DisconectTime(pstn)    = Jun 19, 2010 22:01:54
  Transmit(pstn)         = 3728 (packages): 111840 (bytes)
  Received(pstn)         = 3729 (packages): 111870 (bytes)

#
CallRecord [ 16]:
  CallerNum              = 1111
  CalledNum              = 2222
  EncodeType             = 729r8
  PeerAddress            = 10.10.10.2
  DisconnectCause        = 12
  DisconnectText         = Caller hook on
  TalkingTimes           = 00h 01m 21s
  VoiceTimes             = 00h 01m 21s
  FaxTimes               = 00h 00m 00s
  ImgPages               = 0
  CallDirection          = 1
  SetupTime(voip)        = Jun 19, 2010 21:58:35
  ConnectTime(voip)      = Jun 19, 2010 21:58:36
  DisconectTime(voip)    = Jun 19, 2010 21:59:57
  Transmit(voip)         = 2721 (packages): 114282 (bytes)
  Received(voip)         = 2721 (packages): 114282 (bytes)
  SetupTime(pstn)        = Jun 19, 2010 21:58:32
  ConnectTime(pstn)      = Jun 19, 2010 21:58:32
  DisconectTime(pstn)    = Jun 19, 2010 21:59:57
  Transmit(pstn)         = 2721 (packages): 81630 (bytes)
  Received(pstn)         = 2721 (packages): 81630 (bytes)

#
CallRecord [ 15]:
  CallerNum              = 1111
  CalledNum              = 2222
  EncodeType             = 729r8
  PeerAddress            = 10.10.10.2
  DisconnectCause        = 12
  DisconnectText         = Caller hook on
  TalkingTimes           = 00h 00m 55s
  VoiceTimes             = 00h 00m 55s
  FaxTimes               = 00h 00m 00s
  ImgPages               = 0
  CallDirection          = 1
  SetupTime(voip)        = Jun 19, 2010 21:57:26
  ConnectTime(voip)      = Jun 19, 2010 21:57:27
  DisconectTime(voip)    = Jun 19, 2010 21:58:22
  Transmit(voip)         = 1848 (packages): 77616 (bytes)
  Received(voip)         = 1847 (packages): 77574 (bytes)
  SetupTime(pstn)        = Jun 19, 2010 21:57:24
  ConnectTime(pstn)      = Jun 19, 2010 21:57:24
```

```
  DisconectTime(pstn)     = Jun 19, 2010 21:58:22
  Transmit(pstn)          = 1847 (packages): 55410 (bytes)
  Received(pstn)          = 1848 (packages): 55440 (bytes)

#
CallRecord [ 14]:
  CallerNum               = 1111
  CalledNum               = 2222
  EncodeType              = 729r8
  PeerAddress             = 10.10.10.2
  DisconnectCause         = 12
  DisconnectText          = Caller hook on
  TalkingTimes            = 00h 01m 25s
  VoiceTimes              = 00h 01m 25s
  FaxTimes                = 00h 00m 00s
  ImgPages                = 0
  CallDirection           = 1
  SetupTime(voip)         = Jun 19, 2010 21:55:53
  ConnectTime(voip)       = Jun 19, 2010 21:55:54
  DisconectTime(voip)     = Jun 19, 2010 21:57:19
  Transmit(voip)          = 2850 (packages): 119700 (bytes)
  Received(voip)          = 2850 (packages): 119700 (bytes)
  SetupTime(pstn)         = Jun 19, 2010 21:55:50
  ConnectTime(pstn)       = Jun 19, 2010 21:55:50
  DisconectTime(pstn)     = Jun 19, 2010 21:57:19
  Transmit(pstn)          = 2850 (packages): 85500 (bytes)
  Received(pstn)          = 2850 (packages): 85500 (bytes)
```

Nota-se que não existem diferenças entre os pacotes recebidos e transmitidos, sinalizando que os resultados esperados (sem problemas nas chamadas) foram atingidos.

5.1.5 - Interfaces seriais – Momento – 01

Seguem as estatísticas das interfaces seriais dos roteadores.

ANTES:

```
<ROUTER_02>dis int ser 0/0
Serial0/0 current state :UP
Line protocol current state :UP
Description : Serial0/0 Interface
Virtualbaudrate is 512000 bps
The Maximum Transmit Unit is 1500, Hold timer is 10(sec)
Internet Address is 10.10.10.2/24
Link layer protocol is PPP
LCP opened, IPCP opened, OSICP opened
```

Output queue : (Urgent queuing : Size/Length/Discards) 0/50/0
Output queue : (Protocol queuing : Size/Length/Discards) 0/500/0
Output queue : (FIFO queuing : Size/Length/Discards) 0/75/0
Physical layer is synchronous,
Interface is DTE, Cable type is V35
Last clearing of counters: 21:55:58 UTC Sat 06/19/2010
Last 300 seconds input rate 0.00 bytes/sec, 0 bits/sec, 0.00 packets/sec
Last 300 seconds output rate 0.00 bytes/sec, 0 bits/sec, 0.00 packets/sec
Input: 2014 packets, 104780 bytes
0 broadcasts, 0 multicasts
0 errors, 0 runts, 0 giants
0 CRC, 0 align errors, 0 overruns
0 dribbles, 0 aborts, 0 no buffers
0 frame errors
Output:2013 packets, 104908 bytes
0 errors, 0 underruns, 0 collisions
0 deferred
DCD=UP DTR=UP DSR=UP RTS=UP CTS=UP

<ROUTER_01>dis int ser 0/0
Serial0/0 current state :UP
Line protocol current state :UP
Description : Serial0/0 Interface
The Maximum Transmit Unit is 1500, Hold timer is 10(sec)
Internet Address is 10.10.10.1/24
Link layer protocol is PPP
LCP opened, IPCP opened, OSICP opened
Output queue : (Urgent queuing : Size/Length/Discards) 0/50/0
Output queue : (Protocol queuing : Size/Length/Discards) 0/500/0
Output queue : (FIFO queuing : Size/Length/Discards) 0/75/0
Physical layer is synchronous,Baudrate is 512000 bps
Interface is DCE, Cable type is V35
Last clearing of counters: 21:55:42 UTC Sat 06/19/2010
Last 300 seconds input rate 0.00 bytes/sec, 0 bits/sec, 0.00 packets/sec
Last 300 seconds output rate 0.00 bytes/sec, 0 bits/sec, 0.00 packets/sec
Input: 538 packets, 27896 bytes
0 broadcasts, 0 multicasts
0 errors, 0 runts, 0 giants
0 CRC, 0 align errors, 0 overruns
0 dribbles, 0 aborts, 0 no buffers
0 frame errors
Output:537 packets, 27908 bytes
0 errors, 0 underruns, 0 collisions
0 deferred
DCD=UP DTR=UP DSR=UP RTS=UP CTS=UP

DEPOIS:

<ROUTER_01>dis int ser 0/0

```
Serial0/0 current state :UP
Line protocol current state :UP
Description : Serial0/0 Interface
The Maximum Transmit Unit is 1500, Hold timer is 10(sec)
Internet Address is 10.10.10.1/24
Link layer protocol is PPP
LCP opened, IPCP opened, OSICP opened
Output queue : (Urgent queuing : Size/Length/Discards)  0/50/0
Output queue : (Protocol queuing : Size/Length/Discards) 0/500/0
Output queue : (FIFO queuing : Size/Length/Discards)  0/75/0
Physical layer is synchronous,Baudrate is 512000 bps
Interface is DCE, Cable type is V35
Last clearing of counters: 21:55:42 UTC Sat 06/19/2010
  Last 300 seconds input rate 8463.98 bytes/sec, 67711 bits/sec, 146.88 packets/sec
  Last 300 seconds output rate 8479.50 bytes/sec, 67836 bits/sec, 146.93 packets/sec
  Input: 57618 packets, 3296427 bytes
      0 broadcasts, 0 multicasts
      0 errors, 0 runts, 0 giants
      0 CRC, 0 align errors, 0 overruns
      0 dribbles, 0 aborts, 0 no buffers
      0 frame errors
  Output:57636 packets, 3302513 bytes
      0 errors, 0 underruns, 0 collisions
      0 deferred
  DCD=UP  DTR=UP  DSR=UP  RTS=UP  CTS=UP
```

```
<ROUTER_01>
<ROUTER_02>dis int ser 0/0
Serial0/0 current state :UP
Line protocol current state :UP
Description : Serial0/0 Interface
Virtualbaudrate is 512000 bps
The Maximum Transmit Unit is 1500, Hold timer is 10(sec)
Internet Address is 10.10.10.2/24
Link layer protocol is PPP
LCP opened, IPCP opened, OSICP opened
Output queue : (Urgent queuing : Size/Length/Discards)  0/50/0
Output queue : (Protocol queuing : Size/Length/Discards) 0/500/0
Output queue : (FIFO queuing : Size/Length/Discards)  0/75/0
Physical layer is synchronous,
Interface is DTE, Cable type is V35
Last clearing of counters: 21:55:58 UTC Sat 06/19/2010
  Last 300 seconds input rate 7010.20 bytes/sec, 56081 bits/sec, 121.95 packets/sec
  Last 300 seconds output rate 7003.08 bytes/sec, 56024 bits/sec, 121.90 packets/sec
  Input: 63780 packets, 3621862 bytes
      0 broadcasts, 0 multicasts
      0 errors, 0 runts, 0 giants
      0 CRC, 0 align errors, 0 overruns
      0 dribbles, 0 aborts, 0 no buffers
      0 frame errors
```

```
  Output:63747 packets, 3616579 bytes
       0 errors, 0 underruns, 0 collisions
       0 deferred
  DCD=UP  DTR=UP  DSR=UP  RTS=UP  CTS=UP
```

5.1.6 - Historico de ligações – Momento – 02

```
<ROUTER_01>dis voice call-history-record last 10
 #
 CallRecord [ 22]:
   CallerNum            = 1111
   CalledNum            = 2222
   EncodeType           = 729r8
   PeerAddress          = 10.10.10.2
   DisconnectCause      = 12
   DisconnectText       = Caller hook on
   TalkingTimes         = 00h 01m 33s
   VoiceTimes           = 00h 01m 33s
   FaxTimes             = 00h 00m 00s
   ImgPages             = 0
   CallDirection        = 1
   SetupTime(voip)      = Jun 19, 2010 22:35:05
   ConnectTime(voip)    = Jun 19, 2010 22:35:06
   DisconectTime(voip)  = Jun 19, 2010 22:36:39
   Transmit(voip)       = 3082 (packages): 129444 (bytes)
   Received(voip)       = 3068 (packages): 128856 (bytes)
   SetupTime(pstn)      = Jun 19, 2010 22:35:03
   ConnectTime(pstn)    = Jun 19, 2010 22:35:03
   DisconectTime(pstn)  = Jun 19, 2010 22:36:39
   Transmit(pstn)       = 3068 (packages): 92040 (bytes)
   Received(pstn)       = 3082 (packages): 92460 (bytes)

 #
 CallRecord [ 21]:
   CallerNum            = 1111
   CalledNum            = 2222
   EncodeType           = 729r8
   PeerAddress          = 10.10.10.2
   DisconnectCause      = 12
   DisconnectText       = Caller hook on
   TalkingTimes         = 00h 01m 28s
   VoiceTimes           = 00h 01m 28s
   FaxTimes             = 00h 00m 00s
   ImgPages             = 0
   CallDirection        = 1
   SetupTime(voip)      = Jun 19, 2010 22:33:32
   ConnectTime(voip)    = Jun 19, 2010 22:33:33
   DisconectTime(voip)  = Jun 19, 2010 22:35:01
   Transmit(voip)       = 2910 (packages): 122220 (bytes)
   Received(voip)       = 2909 (packages): 122178 (bytes)
```

```
  SetupTime(pstn)        = Jun 19, 2010 22:33:31
  ConnectTime(pstn)      = Jun 19, 2010 22:33:31
  DisconectTime(pstn)    = Jun 19, 2010 22:35:01
  Transmit(pstn)         = 2909 (packages): 87270 (bytes)
  Received(pstn)         = 2910 (packages): 87300 (bytes)

#
CallRecord [ 20]:
  CallerNum              = 1111
  CalledNum              = 2222
  EncodeType             = 729r8
  PeerAddress            = 10.10.10.2
  DisconnectCause        = 12
  DisconnectText         = Caller hook on
  TalkingTimes           = 00h 01m 10s
  VoiceTimes             = 00h 01m 10s
  FaxTimes               = 00h 00m 00s
  ImgPages               = 0
  CallDirection          = 1
  SetupTime(voip)        = Jun 19, 2010 22:32:18
  ConnectTime(voip)      = Jun 19, 2010 22:32:19
  DisconectTime(voip)    = Jun 19, 2010 22:33:29
  Transmit(voip)         = 2289 (packages): 96138 (bytes)
  Received(voip)         = 2283 (packages): 95886 (bytes)
  SetupTime(pstn)        = Jun 19, 2010 22:32:14
  ConnectTime(pstn)      = Jun 19, 2010 22:32:14
  DisconectTime(pstn)    = Jun 19, 2010 22:33:29
  Transmit(pstn)         = 2283 (packages): 68490 (bytes)
  Received(pstn)         = 2289 (packages): 68670 (bytes)

#
CallRecord [ 19]:
  CallerNum              = 1111
  CalledNum              = 2222
  EncodeType             = 729r8
  PeerAddress            = 10.10.10.2
  DisconnectCause        = 12
  DisconnectText         = Caller hook on
  TalkingTimes           = 00h 01m 01s
  VoiceTimes             = 00h 01m 01s
  FaxTimes               = 00h 00m 00s
  ImgPages               = 0
  CallDirection          = 1
  SetupTime(voip)        = Jun 19, 2010 22:31:11
  ConnectTime(voip)      = Jun 19, 2010 22:31:12
  DisconectTime(voip)    = Jun 19, 2010 22:32:13
  Transmit(voip)         = 1960 (packages): 82320 (bytes)
  Received(voip)         = 1951 (packages): 81942 (bytes)
  SetupTime(pstn)        = Jun 19, 2010 22:31:05
  ConnectTime(pstn)      = Jun 19, 2010 22:31:05
```

```
  DisconectTime(pstn)     = Jun 19, 2010 22:32:13
  Transmit(pstn)          = 1951 (packages): 58530 (bytes)
  Received(pstn)          = 1960 (packages): 58800 (bytes)

#
CallRecord [ 18]:
  CallerNum               = 1111
  CalledNum               = 2222
  EncodeType              = 729r8
  PeerAddress             = 10.10.10.2
  DisconnectCause         = 12
  DisconnectText          = Caller hook on
  TalkingTimes            = 00h 02m 29s
  VoiceTimes              = 00h 02m 29s
  FaxTimes                = 00h 00m 00s
  ImgPages                = 0
  CallDirection           = 1
  SetupTime(voip)         = Jun 19, 2010 22:28:34
  ConnectTime(voip)       = Jun 19, 2010 22:28:36
  DisconectTime(voip)     = Jun 19, 2010 22:31:05
  Transmit(voip)          = 4929 (packages): 207018 (bytes)
  Received(voip)          = 4895 (packages): 205590 (bytes)
  SetupTime(pstn)         = Jun 19, 2010 22:28:32
  ConnectTime(pstn)       = Jun 19, 2010 22:28:32
  DisconectTime(pstn)     = Jun 19, 2010 22:31:05
  Transmit(pstn)          = 4895 (packages): 146850 (bytes)
  Received(pstn)          = 4929 (packages): 147870 (bytes)


<ROUTER_02>dis voice call-his las 10
#
CallRecord :
  CallerNum               = 1111
  CalledNum               = 2222
  EncodeType              = 729r8
  PeerAddress             = 10.10.10.1
  DisconnectCause         = 12
  DisconnectText          = Caller hook on
  TalkingTimes            = 00h 01m 33s
  VoiceTimes              = 00h 01m 33s
  FaxTimes                = 00h 00m 00s
  ImgPages                = 0
  CallDirection           = 2
  SetupTime(voip)         = Jun 19, 2010 22:35:37
  ConnectTime(voip)       = Jun 19, 2010 22:35:38
  DisconectTime(voip)     = Jun 19, 2010 22:37:11
  Transmit(voip)          = 3085 (packages): 129570 (bytes)
  Received(voip)          = 3082 (packages): 129444 (bytes)
  SetupTime(pstn)         = Jun 19, 2010 22:35:37
  ConnectTime(pstn)       = Jun 19, 2010 22:35:37
```

```
  DisconectTime(pstn)     = Jun 19, 2010 22:37:11
  Transmit(pstn)          = 3082 (packages): 92460 (bytes)
  Received(pstn)          = 3085 (packages): 92550 (bytes)

#
CallRecord :
  CallerNum               = 1111
  CalledNum               = 2222
  EncodeType              = 729r8
  PeerAddress             = 10.10.10.1
  DisconnectCause         = 12
  DisconnectText          = Caller hook on
  TalkingTimes            = 00h 01m 28s
  VoiceTimes              = 00h 01m 28s
  FaxTimes                = 00h 00m 00s
  ImgPages                = 0
  CallDirection           = 2
  SetupTime(voip)         = Jun 19, 2010 22:34:05
  ConnectTime(voip)       = Jun 19, 2010 22:34:05
  DisconectTime(voip)     = Jun 19, 2010 22:35:33
  Transmit(voip)          = 2915 (packages): 122430 (bytes)
  Received(voip)          = 2910 (packages): 122220 (bytes)
  SetupTime(pstn)         = Jun 19, 2010 22:34:05
  ConnectTime(pstn)       = Jun 19, 2010 22:34:05
  DisconectTime(pstn)     = Jun 19, 2010 22:35:33
  Transmit(pstn)          = 2910 (packages): 87300 (bytes)
  Received(pstn)          = 2915 (packages): 87450 (bytes)

#
CallRecord :
  CallerNum               = 1111
  CalledNum               = 2222
  EncodeType              = 729r8
  PeerAddress             = 10.10.10.1
  DisconnectCause         = 12
  DisconnectText          = Caller hook on
  TalkingTimes            = 00h 01m 10s
  VoiceTimes              = 00h 01m 10s
  FaxTimes                = 00h 00m 00s
  ImgPages                = 0
  CallDirection           = 2
  SetupTime(voip)         = Jun 19, 2010 22:32:51
  ConnectTime(voip)       = Jun 19, 2010 22:32:52
  DisconectTime(voip)     = Jun 19, 2010 22:34:02
  Transmit(voip)          = 2290 (packages): 96180 (bytes)
  Received(voip)          = 2287 (packages): 96054 (bytes)
  SetupTime(pstn)         = Jun 19, 2010 22:32:51
  ConnectTime(pstn)       = Jun 19, 2010 22:32:51
  DisconectTime(pstn)     = Jun 19, 2010 22:34:02
  Transmit(pstn)          = 2287 (packages): 68610 (bytes)
```

```
  Received(pstn)        = 2290 (packages): 68700 (bytes)

#
CallRecord :
  CallerNum             = 1111
  CalledNum             = 2222
  EncodeType            = 729r8
  PeerAddress           = 10.10.10.1
  DisconnectCause       = 12
  DisconnectText        = Caller hook on
  TalkingTimes          = 00h 01m 00s
  VoiceTimes            = 00h 01m 00s
  FaxTimes              = 00h 00m 00s
  ImgPages              = 0
  CallDirection         = 2
  SetupTime(voip)       = Jun 19, 2010 22:31:44
  ConnectTime(voip)     = Jun 19, 2010 22:31:45
  DisconectTime(voip)   = Jun 19, 2010 22:32:45
  Transmit(voip)        = 1964 (packages): 82488 (bytes)
  Received(voip)        = 1956 (packages): 82152 (bytes)
  SetupTime(pstn)       = Jun 19, 2010 22:31:44
  ConnectTime(pstn)     = Jun 19, 2010 22:31:44
  DisconectTime(pstn)   = Jun 19, 2010 22:32:45
  Transmit(pstn)        = 1956 (packages): 58680 (bytes)
  Received(pstn)        = 1964 (packages): 58920 (bytes)

#
CallRecord :
  CallerNum             = 1111
  CalledNum             = 2222
  EncodeType            = 729r8
  PeerAddress           = 10.10.10.1
  DisconnectCause       = 12
  DisconnectText        = Caller hook on
  TalkingTimes          = 00h 02m 29s
  VoiceTimes            = 00h 02m 29s
  FaxTimes              = 00h 00m 00s
  ImgPages              = 0
  CallDirection         = 2
  SetupTime(voip)       = Jun 19, 2010 22:29:07
  ConnectTime(voip)     = Jun 19, 2010 22:29:08
  DisconectTime(voip)   = Jun 19, 2010 22:31:37
  Transmit(voip)        = 4929 (packages): 207018 (bytes)
  Received(voip)        = 4918 (packages): 206556 (bytes)
  SetupTime(pstn)       = Jun 19, 2010 22:29:07
  ConnectTime(pstn)     = Jun 19, 2010 22:29:07
  DisconectTime(pstn)   = Jun 19, 2010 22:31:37
  Transmit(pstn)        = 4918 (packages): 147540 (bytes)
  Received(pstn)        = 4929 (packages): 147870 (bytes)
```

```
<ROUTER_02>
```

Nesse caso já se nota a diferença de pacotes e bytes transmitidos e recebidos, evidenciando os problemas de voz apresentados durante os testes.

5.1.7 - Interfaces seriais – Momento 02:

ANTES:

```
<ROUTER_02>dis int ser 0/0
Serial0/0 current state :UP
Line protocol current state :UP
Description : Serial0/0 Interface
Virtualbaudrate is 512000 bps
The Maximum Transmit Unit is 1500, Hold timer is 10(sec)
Internet Address is 10.10.10.2/24
Link layer protocol is PPP
LCP opened, IPCP opened, OSICP opened
Output queue : (Urgent queuing : Size/Length/Discards)  0/50/0
Output queue : (Protocol queuing : Size/Length/Discards) 0/500/0
Output queue : (FIFO queuing : Size/Length/Discards)  0/75/0
Physical layer is synchronous,
Interface is DTE, Cable type is V35
Last clearing of counters: 22:26:40 UTC Sat 06/19/2010
  Last 300 seconds input rate 0.00 bytes/sec, 0 bits/sec, 0.00 packets/sec
  Last 300 seconds output rate 0.00 bytes/sec, 0 bits/sec, 0.00 packets/sec
  Input: 1338 packets, 69529 bytes
      0 broadcasts, 0 multicasts
      0 errors, 0 runts, 0 giants
      0 CRC, 0 align errors, 0 overruns
      0 dribbles, 0 aborts, 0 no buffers
      0 frame errors
  Output:1327 packets, 68896 bytes
      0 errors, 0 underruns, 0 collisions
      0 deferred
  DCD=UP  DTR=UP  DSR=UP  RTS=UP  CTS=UP

<ROUTER_01>dis int ser 0/0
Serial0/0 current state :UP
Line protocol current state :UP
Description : Serial0/0 Interface
The Maximum Transmit Unit is 1500, Hold timer is 10(sec)
Internet Address is 10.10.10.1/24
Link layer protocol is PPP
LCP opened, IPCP opened, OSICP opened
Output queue : (Urgent queuing : Size/Length/Discards)  0/50/0
Output queue : (Protocol queuing : Size/Length/Discards) 0/500/0
Output queue : (FIFO queuing : Size/Length/Discards)  0/75/0
```

Physical layer is synchronous,Baudrate is 512000 bps
Interface is DCE, Cable type is V35
Last clearing of counters: 22:26:21 UTC Sat 06/19/2010
Last 300 seconds input rate 0.00 bytes/sec, 0 bits/sec, 0.00 packets/sec
Last 300 seconds output rate 0.00 bytes/sec, 0 bits/sec, 0.00 packets/sec
Input: 133 packets, 6836 bytes
0 broadcasts, 0 multicasts
0 errors, 0 runts, 0 giants
0 CRC, 0 align errors, 0 overruns
0 dribbles, 0 aborts, 0 no buffers
0 frame errors
Output:133 packets, 6852 bytes
0 errors, 0 underruns, 0 collisions
0 deferred
DCD=UP DTR=UP DSR=UP RTS=UP CTS=UP

DEPOIS:

<ROUTER_01>dis int ser 0/0
Serial0/0 current state :UP
Line protocol current state :UP
Description : Serial0/0 Interface
The Maximum Transmit Unit is 1500, Hold timer is 10(sec)
Internet Address is 10.10.10.1/24
Link layer protocol is PPP
LCP opened, IPCP opened, OSICP opened
Output queue : (Urgent queuing : Size/Length/Discards) 0/50/0
Output queue : (Protocol queuing : Size/Length/Discards) 0/500/0
Output queue : (FIFO queuing : Size/Length/Discards) 0/75/0
Physical layer is synchronous,Baudrate is 512000 bps
Interface is DCE, Cable type is V35
Last clearing of counters: 21:55:42 UTC Sat 06/19/2010
Last 300 seconds input rate 8463.98 bytes/sec, 67711 bits/sec, 146.88 packets/sec
Last 300 seconds output rate 8479.50 bytes/sec, 67836 bits/sec, 146.93 packets/sec
Input: 57618 packets, 3296427 bytes
0 broadcasts, 0 multicasts
0 errors, 0 runts, 0 giants
0 CRC, 0 align errors, 0 overruns
0 dribbles, 0 aborts, 0 no buffers
0 frame errors
Output:57636 packets, 3302513 bytes
0 errors, 0 underruns, 0 collisions
0 deferred
DCD=UP DTR=UP DSR=UP RTS=UP CTS=UP

<ROUTER_02>dis int ser 0/0
Serial0/0 current state :UP
Line protocol current state :UP
Description : Serial0/0 Interface
Virtualbaudrate is 512000 bps

```
The Maximum Transmit Unit is 1500, Hold timer is 10(sec)
Internet Address is 10.10.10.2/24
Link layer protocol is PPP
LCP opened, IPCP opened, OSICP opened
Output queue : (Urgent queuing : Size/Length/Discards)  0/50/0
Output queue : (Protocol queuing : Size/Length/Discards) 0/500/0
Output queue : (FIFO queuing : Size/Length/Discards)  0/75/0
Physical layer is synchronous,
Interface is DTE, Cable type is V35
Last clearing of counters: 21:55:58 UTC Sat 06/19/2010
   Last 300 seconds input rate 7010.20 bytes/sec, 56081 bits/sec, 121.95 packets/sec
   Last 300 seconds output rate 7003.08 bytes/sec, 56024 bits/sec, 121.90 packets/sec
   Input: 63780 packets, 3621862 bytes
        0 broadcasts, 0 multicasts
        0 errors, 0 runts, 0 giants
        0 CRC, 0 align errors, 0 overruns
        0 dribbles, 0 aborts, 0 no buffers
        0 frame errors
   Output:63747 packets, 3616579 bytes
        0 errors, 0 underruns, 0 collisions
        0 deferred
   DCD=UP  DTR=UP  DSR=UP  RTS=UP  CTS=UP
```

5.1.8 – Análise dados – Cenário 01:

Segue análise para chamada 01:

Tabela 17 – Analise dados Cenário 01

Diferença pacotes - Ligação 01				
Roteador_01		Roteador_02		Diferença:
Transmit(voip):	3082	Transmit(voip):	3085	3
Received(voip):	3068	Received(voip):	3082	14

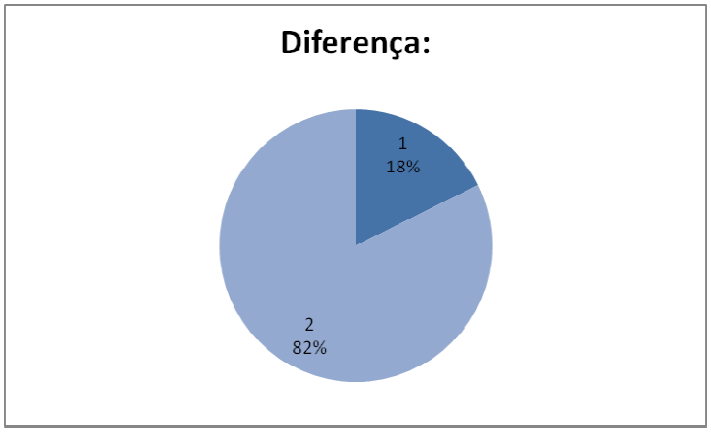


Figura 24 - Analise dados - Cenário 01

Nota-se que a diferença de pacotes confirma a existência do problema de voz relatado durante o experimento.

O problema se repete durante todas as outras chamadas efetuadas.

5.2 – Cenário 02 – Tráfego com QoS PQ (Priority Queue):

5.2.1 – Topologia adotada:

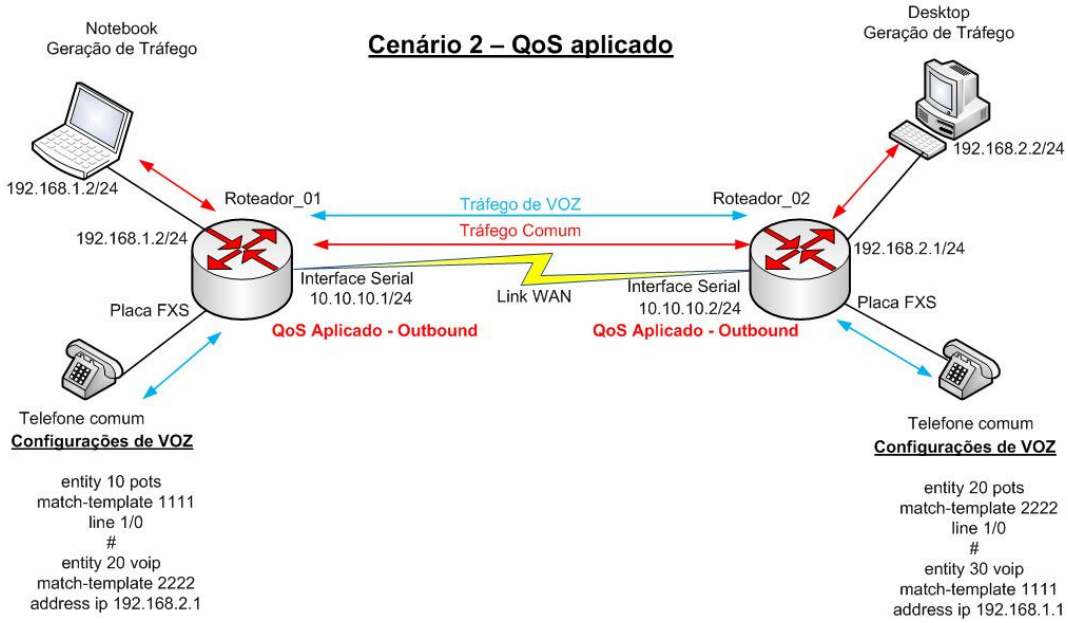


Figura 25 - Topologia – Cenário 02

5.2.2 – Resultados Esperados/Obtidos:

Tabela 18 – Cenário 02/Momento 1: Resultados obtidos

CENÁRIO 02 - CHAMADAS REALIZADAS SEM USO DO QoS:		
Nr. Chamada:	**Resultado:**	**Observações:**
1	**OK**	Chamada realizada com sucesso - Qualidade OK
2	**OK**	Chamada realizada com sucesso - Qualidade OK
3	**OK**	Chamada realizada com sucesso - Qualidade OK
4	**OK**	Chamada realizada com sucesso - Qualidade OK
5	**OK**	Chamada realizada com sucesso - Qualidade OK

➔ Os resultados esperados para o momento 01 foram atingidos com sucesso.

Tabela 19 - Cenário 02/Momento 2: Resultados obtidos

CENÁRIO 02 - CHAMADAS REALIZADAS DURANTE CONGESTIONAMENTO SEM QoS:			
Nr. Chamada:	**Resultado:**	**Duração:**	**Observações:**
1	**NOK**	1 Minuto	Chamada estabelecida porém conversação e qualidade seriamente comprometidos
2	**NOK**	1 Minuto	Chamada estabelecida porém conversação e qualidade seriamente comprometidos
3	**NOK**	1 Minuto	Chamada estabelecida porém conversação e qualidade seriamente comprometidos
4	**NOK**	1 Minuto	Chamada estabelecida porém conversação e qualidade seriamente comprometidos
5	**NOK**	1 Minuto	Chamada estabelecida porém conversação e qualidade seriamente comprometidos

➔ Os resultados esperados para o momento 02 foram atingidos com sucesso.

Tabela 20 - Cenário 02/Momento 3: Resultados obtidos

CENÁRIO 02 - CHAMADAS REALIZADAS DURANTE CONGESTIONAMENTO UTILIZANDO QoS - PQ			
Nr. Chamada:	**Resultado:**	**Duração:**	**Observações:**
1	OK	1 Minuto	Mesmo durante congestionamento as chamadas apresentaram qualidade - OK
2	OK	1 Minuto	Mesmo durante congestionamento as chamadas apresentaram qualidade - OK
3	OK	1 Minuto	Mesmo durante congestionamento as chamadas apresentaram qualidade - OK
4	OK	1 Minuto	Mesmo durante congestionamento as chamadas apresentaram qualidade - OK
5	OK	1 Minuto	Mesmo durante congestionamento as chamadas apresentaram qualidade - OK

➔ Os resultados esperados para o momento 03 foram atingidos com sucesso.

Anotações:

- Ao iniciar a transferência de dados a chamada apresentou o mesmo comportamento relatado no cenário 01 (Problemas de voz);
- Após a aplicação das políticas de QoS nas interfaces:
- Não houve picotamento na voz;
- A mensagem estava clara e de perfeito entendimento para ambos os lados

5.2.3 - Dados coletados

Os dados referentes aos momentos 01 e 02 coletados durante o experimento se assemelham com os coletados no cenário 01.

Ficando adotados como padrão de comparação para esse experimento.

5.2.3.1 – PING – Momento 03:

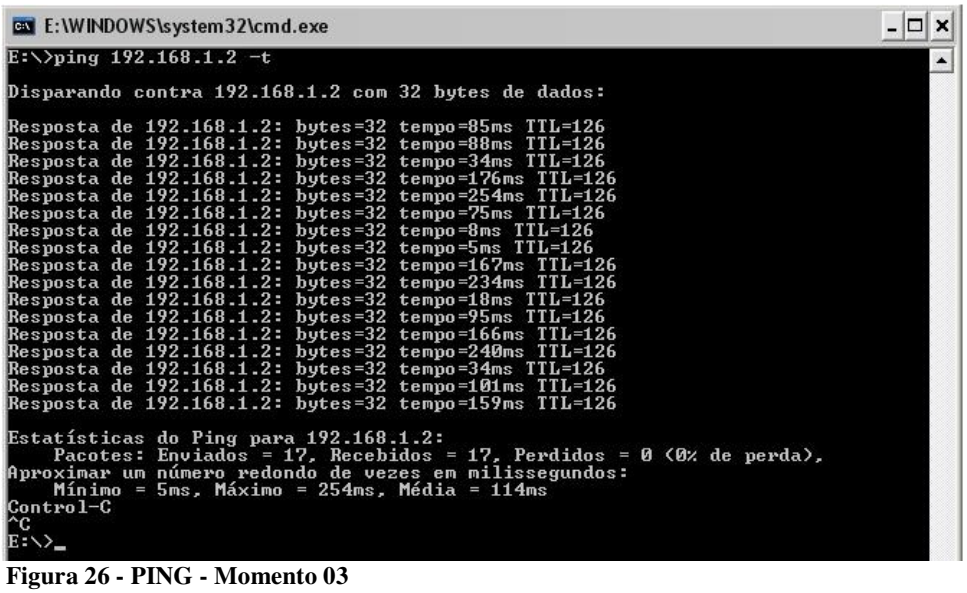


Figura 26 - PING - Momento 03

5.2.4 - Historico de ligações – Momento 03:

```
<ROUTER_01>dis voice call-history-record last 10
 #
 CallRecord [ 30]:
   CallerNum            = 2222
   CalledNum            = 1111
   EncodeType           = 729r8
   PeerAddress          = 10.10.10.2
   DisconnectCause      = 13
   DisconnectText       = Called hook on
   TalkingTimes         = 00h 01m 09s
   VoiceTimes           = 00h 01m 09s
   FaxTimes             = 00h 00m 00s
   ImgPages             = 0
   CallDirection        = 2
   SetupTime(voip)      = Jun 20, 2010 00:21:57
   ConnectTime(voip)    = Jun 20, 2010 00:21:58
   DisconectTime(voip)  = Jun 20, 2010 00:23:07
   Transmit(voip)       = 2253 (packages): 94626 (bytes)
   Received(voip)       = 2250 (packages): 94500 (bytes)
   SetupTime(pstn)      = Jun 20, 2010 00:21:57
   ConnectTime(pstn)    = Jun 20, 2010 00:21:57
   DisconectTime(pstn)  = Jun 20, 2010 00:23:07
   Transmit(pstn)       = 2250 (packages): 67500 (bytes)
   Received(pstn)       = 2253 (packages): 67590 (bytes)

 #
 CallRecord [ 29]:
   CallerNum            = 1111
   CalledNum            = 2222
   EncodeType           = 729r8
   PeerAddress          = 10.10.10.2
   DisconnectCause      = 12
   DisconnectText       = Caller hook on
   TalkingTimes         = 00h 02m 08s
   VoiceTimes           = 00h 02m 08s
   FaxTimes             = 00h 00m 00s
   ImgPages             = 0
   CallDirection        = 1
   SetupTime(voip)      = Jun 20, 2010 00:19:42
   ConnectTime(voip)    = Jun 20, 2010 00:19:43
   DisconectTime(voip)  = Jun 20, 2010 00:21:51
   Transmit(voip)       = 4269 (packages): 179298 (bytes)
   Received(voip)       = 4268 (packages): 179256 (bytes)
   SetupTime(pstn)      = Jun 20, 2010 00:19:40
   ConnectTime(pstn)    = Jun 20, 2010 00:19:40
   DisconectTime(pstn)  = Jun 20, 2010 00:21:51
   Transmit(pstn)       = 4268 (packages): 128040 (bytes)
   Received(pstn)       = 4269 (packages): 128070 (bytes)
```

```
#
CallRecord [ 28]:
  CallerNum          = 1111
  CalledNum          = 2222
  EncodeType         = 729r8
  PeerAddress        = 10.10.10.2
  DisconnectCause      = 12
  DisconnectText       = Caller hook on
  TalkingTimes       = 00h 01m 41s
  VoiceTimes         = 00h 01m 41s
  FaxTimes           = 00h 00m 00s
  ImgPages           = 0
  CallDirection      = 1
  SetupTime(voip)      = Jun 20, 2010 00:17:52
  ConnectTime(voip)    = Jun 20, 2010 00:17:54
  DisconectTime(voip)    = Jun 20, 2010 00:19:35
  Transmit(voip)       = 3314 (packages): 139188 (bytes)
  Received(voip)       = 3308 (packages): 138936 (bytes)
  SetupTime(pstn)      = Jun 20, 2010 00:17:50
  ConnectTime(pstn)    = Jun 20, 2010 00:17:50
  DisconectTime(pstn)    = Jun 20, 2010 00:19:35
  Transmit(pstn)       = 3308 (packages): 99240 (bytes)
  Received(pstn)       = 3314 (packages): 99420 (bytes)

#
CallRecord [ 27]:
  CallerNum          = 1111
  CalledNum          = 2222
  EncodeType         = 729r8
  PeerAddress        = 10.10.10.2
  DisconnectCause      = 12
  DisconnectText       = Caller hook on
  TalkingTimes       = 00h 01m 31s
  VoiceTimes         = 00h 01m 31s
  FaxTimes           = 00h 00m 00s
  ImgPages           = 0
  CallDirection      = 1
  SetupTime(voip)      = Jun 20, 2010 00:16:13
  ConnectTime(voip)    = Jun 20, 2010 00:16:15
  DisconectTime(voip)    = Jun 20, 2010 00:17:46
  Transmit(voip)       = 3031 (packages): 127302 (bytes)
  Received(voip)       = 3030 (packages): 127260 (bytes)
  SetupTime(pstn)      = Jun 20, 2010 00:16:08
  ConnectTime(pstn)    = Jun 20, 2010 00:16:08
  DisconectTime(pstn)    = Jun 20, 2010 00:17:46
  Transmit(pstn)       = 3030 (packages): 90900 (bytes)
  Received(pstn)       = 3031 (packages): 90930 (bytes)

#
CallRecord [ 26]:
```

```
  CallerNum            = 1111
  CalledNum            = 2222
  EncodeType           = 711u
  PeerAddress          = 10.10.10.2
  DisconnectCause      = 13
  DisconnectText       = Called hook on
  TalkingTimes         = 00h 00m 00s
  VoiceTimes           = 00h 00m 00s
  FaxTimes             = 00h 00m 00s
  ImgPages             = 0
  CallDirection        = 1
  SetupTime(voip)      = Jun 20, 2010 00:16:04
  ConnectTime(voip)    = None
  DisconectTime(voip)  = Jun 20, 2010 00:16:05
  Transmit(voip)       = 0 (packages): 0 (bytes)
  Received(voip)       = 0 (packages): 0 (bytes)
  SetupTime(pstn)      = Jun 20, 2010 00:16:00
  ConnectTime(pstn)    = Jun 20, 2010 00:16:00
  DisconectTime(pstn)  = Jun 20, 2010 00:16:05
  Transmit(pstn)       = 0 (packages): 0 (bytes)
  Received(pstn)       = 0 (packages): 0 (bytes)

#
CallRecord [ 25]:
  CallerNum            = 1111
  CalledNum            = 2222
  EncodeType           = 729r8
  PeerAddress          = 10.10.10.2
  DisconnectCause      = 12
  DisconnectText       = Caller hook on
  TalkingTimes         = 00h 01m 16s
  VoiceTimes           = 00h 01m 16s
  FaxTimes             = 00h 00m 00s
  ImgPages             = 0
  CallDirection        = 1
  SetupTime(voip)      = Jun 20, 2010 00:14:26
  ConnectTime(voip)    = Jun 20, 2010 00:14:28
  DisconectTime(voip)  = Jun 20, 2010 00:15:44
  Transmit(voip)       = 2524 (packages): 106008 (bytes)
  Received(voip)       = 2522 (packages): 105924 (bytes)
  SetupTime(pstn)      = Jun 20, 2010 00:14:25
  ConnectTime(pstn)    = Jun 20, 2010 00:14:25
  DisconectTime(pstn)  = Jun 20, 2010 00:15:44
  Transmit(pstn)       = 2522 (packages): 75660 (bytes)
  Received(pstn)       = 2524 (packages): 75720 (bytes)

-------------------------------------------------------------------------------
<ROUTER_02>dis voice call-history-record last 10
 #
 CallRecord :
```

```
  CallerNum          = 2222
  CalledNum          = 1111
  EncodeType         = 729r8
  PeerAddress        = 10.10.10.1
  DisconnectCause    = 13
  DisconnectText     = Called hook on
  TalkingTimes       = 00h 01m 08s
  VoiceTimes         = 00h 01m 08s
  FaxTimes           = 00h 00m 00s
  ImgPages           = 0
  CallDirection      = 1
  SetupTime(voip)    = Jun 20, 2010 00:22:30
  ConnectTime(voip)  = Jun 20, 2010 00:22:31
  DisconectTime(voip)= Jun 20, 2010 00:23:39
  Transmit(voip)     = 2254 (packages): 94668 (bytes)
  Received(voip)     = 2253 (packages): 94626 (bytes)
  SetupTime(pstn)    = Jun 20, 2010 00:22:28
  ConnectTime(pstn)  = Jun 20, 2010 00:22:28
  DisconectTime(pstn)= Jun 20, 2010 00:23:39
  Transmit(pstn)     = 2253 (packages): 67590 (bytes)
  Received(pstn)     = 2254 (packages): 67620 (bytes)

#
CallRecord :
  CallerNum          = 1111
  CalledNum          = 2222
  EncodeType         = 729r8
  PeerAddress        = 10.10.10.1
  DisconnectCause    = 12
  DisconnectText     = Caller hook on
  TalkingTimes       = 00h 02m 09s
  VoiceTimes         = 00h 02m 09s
  FaxTimes           = 00h 00m 00s
  ImgPages           = 0
  CallDirection      = 2
  SetupTime(voip)    = Jun 20, 2010 00:20:14
  ConnectTime(voip)  = Jun 20, 2010 00:20:15
  DisconectTime(voip)= Jun 20, 2010 00:22:24
  Transmit(voip)     = 4270 (packages): 179340 (bytes)
  Received(voip)     = 4269 (packages): 179298 (bytes)
  SetupTime(pstn)    = Jun 20, 2010 00:20:14
  ConnectTime(pstn)  = Jun 20, 2010 00:20:14
  DisconectTime(pstn)= Jun 20, 2010 00:22:24
  Transmit(pstn)     = 4269 (packages): 128070 (bytes)
  Received(pstn)     = 4270 (packages): 128100 (bytes)

#
CallRecord :
  CallerNum          = 1111
  CalledNum          = 2222
```

```
    EncodeType          = 729r8
    PeerAddress         = 10.10.10.1
    DisconnectCause     = 12
    DisconnectText      = Caller hook on
    TalkingTimes        = 00h 01m 42s
    VoiceTimes          = 00h 01m 42s
    FaxTimes            = 00h 00m 00s
    ImgPages            = 0
    CallDirection       = 2
    SetupTime(voip)     = Jun 20, 2010 00:18:26
    ConnectTime(voip)   = Jun 20, 2010 00:18:26
    DisconectTime(voip) = Jun 20, 2010 00:20:08
    Transmit(voip)      = 3313 (packages): 139146 (bytes)
    Received(voip)      = 3314 (packages): 139188 (bytes)
    SetupTime(pstn)     = Jun 20, 2010 00:18:26
    ConnectTime(pstn)   = Jun 20, 2010 00:18:26
    DisconectTime(pstn) = Jun 20, 2010 00:20:08
    Transmit(pstn)      = 3314 (packages): 99420 (bytes)
    Received(pstn)      = 3313 (packages): 99390 (bytes)

  #
  CallRecord :
    CallerNum           = 1111
    CalledNum           = 2222
    EncodeType          = 729r8
    PeerAddress         = 10.10.10.1
    DisconnectCause     = 12
    DisconnectText      = Caller hook on
    TalkingTimes        = 00h 01m 32s
    VoiceTimes          = 00h 01m 32s
    FaxTimes            = 00h 00m 00s
    ImgPages            = 0
    CallDirection       = 2
    SetupTime(voip)     = Jun 20, 2010 00:16:46
    ConnectTime(voip)   = Jun 20, 2010 00:16:47
    DisconectTime(voip) = Jun 20, 2010 00:18:19
    Transmit(voip)      = 3034 (packages): 127428 (bytes)
    Received(voip)      = 3031 (packages): 127302 (bytes)
    SetupTime(pstn)     = Jun 20, 2010 00:16:46
    ConnectTime(pstn)   = Jun 20, 2010 00:16:46
    DisconectTime(pstn) = Jun 20, 2010 00:18:19
    Transmit(pstn)      = 3031 (packages): 90930 (bytes)
    Received(pstn)      = 3034 (packages): 91020 (bytes)

   #
  CallRecord :
    CallerNum           = 1111
    CalledNum           = 2222
    EncodeType          = 729r8
    PeerAddress         = 10.10.10.1
```

```
DisconnectCause        = 12
DisconnectText         = Caller hook on
TalkingTimes           = 00h 01m 17s
VoiceTimes             = 00h 01m 17s
FaxTimes               = 00h 00m 00s
ImgPages               = 0
CallDirection          = 2
SetupTime(voip)        = Jun 20, 2010 00:14:59
ConnectTime(voip)      = Jun 20, 2010 00:14:59
DisconectTime(voip)    = Jun 20, 2010 00:16:16
Transmit(voip)         = 2527 (packages): 106134 (bytes)
Received(voip)         = 2524 (packages): 106008 (bytes)
SetupTime(pstn)        = Jun 20, 2010 00:14:59
ConnectTime(pstn)      = Jun 20, 2010 00:14:59
DisconectTime(pstn)    = Jun 20, 2010 00:16:16
Transmit(pstn)         = 2524 (packages): 75720 (bytes)
Received(pstn)         = 2527 (packages): 75810 (bytes)
```

Devido a utilização do QoS não se nota diferença entre pacotes recebidos e transmitidos.

Qualidade da voz mantida.

5.2.5 - Interfaces seriais – Momento 03:

ANTES:

```
<ROUTER_02>reset count int
<ROUTER_02>dis int ser 0/0
Serial0/0 current state :UP
Line protocol current state :UP
Description : Serial0/0 Interface
Virtualbaudrate is 512000 bps
The Maximum Transmit Unit is 1500, Hold timer is 10(sec)
Internet Address is 10.10.10.2/24
Link layer protocol is PPP
LCP opened, IPCP opened, OSICP opened
Output queue : (Urgent queuing : Size/Length/Discards)  0/50/0
Output queue : (Protocol queuing : Size/Length/Discards) 0/500/0
Output queue : (Priority queuing : PQL 1 Size/Length/Discards)
Top:  0/60/0    Middle:  0/40/0    Normal:  0/60/0    Bottom:  0/100/0
Physical layer is synchronous,
Interface is DTE, Cable type is V35
Last clearing of counters: 00:08:02 UTC Sun 06/20/2010
  Last 300 seconds input rate 0.00 bytes/sec, 0 bits/sec, 0.00 packets/sec
  Last 300 seconds output rate 0.00 bytes/sec, 0 bits/sec, 0.00 packets/sec
  Input: 282 packets, 14505 bytes
      0 broadcasts, 0 multicasts
      0 errors, 0 runts, 0 giants
```

0 CRC, 0 align errors, 0 overruns
0 dribbles, 0 aborts, 0 no buffers
0 frame errors
Output:273 packets, 14088 bytes
0 errors, 0 underruns, 0 collisions
0 deferred
DCD=UP DTR=UP DSR=UP RTS=UP CTS=UP

<ROUTER_01>dis int ser 0/0
Serial0/0 current state :UP
Line protocol current state :UP
Description : Serial0/0 Interface
The Maximum Transmit Unit is 1500, Hold timer is 10(sec)
Internet Address is 10.10.10.1/24
Link layer protocol is PPP
LCP opened, IPCP opened, OSICP opened
Output queue : (Urgent queuing : Size/Length/Discards) 0/50/0
Output queue : (Protocol queuing : Size/Length/Discards) 0/500/0
Output queue : (Priority queuing : PQL 1 Size/Length/Discards)
Top: 0/60/0 Middle: 0/40/0 Normal: 0/60/0 Bottom: 0/100/0
Physical layer is synchronous,Baudrate is 512000 bps
Interface is DCE, Cable type is V35
Last clearing of counters: 00:07:42 UTC Sun 06/20/2010
Last 300 seconds input rate 0.00 bytes/sec, 0 bits/sec, 0.00 packets/sec
Last 300 seconds output rate 0.00 bytes/sec, 0 bits/sec, 0.00 packets/sec
Input: 642 packets, 33304 bytes
0 broadcasts, 0 multicasts
0 errors, 0 runts, 0 giants
0 CRC, 0 align errors, 0 overruns
0 dribbles, 0 aborts, 0 no buffers
0 frame errors
Output:642 packets, 33384 bytes
0 errors, 0 underruns, 0 collisions
0 deferred
DCD=UP DTR=UP DSR=UP RTS=UP CTS=UP

<ROUTER_01>

DEPOIS:

ROUTER_01>dis int ser 0/0
Serial0/0 current state :UP
Line protocol current state :UP
Description : Serial0/0 Interface
The Maximum Transmit Unit is 1500, Hold timer is 10(sec)
Internet Address is 10.10.10.1/24
Link layer protocol is PPP
LCP opened, IPCP opened, OSICP opened
Output queue : (Urgent queuing : Size/Length/Discards) 0/50/0
Output queue : (Protocol queuing : Size/Length/Discards) 0/500/0

```
Output queue : (Priority queuing : PQL 1 Size/Length/Discards)
Top:  0/60/0    Middle:  0/40/0    Normal:  0/60/0    Bottom:  0/100/10
Physical layer is synchronous,Baudrate is 512000 bps
Interface is DCE, Cable type is V35
Last clearing of counters: 00:07:42 UTC Sun 06/20/2010
  Last 300 seconds input rate 34918.85 bytes/sec, 279350 bits/sec, 240.51 packets/sec
  Last 300 seconds output rate 42689.78 bytes/sec, 341518 bits/sec, 257.27 packets/sec
  Input: 174396 packets, 22221721 bytes
      0 broadcasts, 0 multicasts
      0 errors, 0 runts, 0 giants
      0 CRC, 0 align errors, 0 overruns
      0 dribbles, 0 aborts, 0 no buffers
      0 frame errors
  Output:189173 packets, 32442583 bytes
      0 errors, 0 underruns, 0 collisions
      0 deferred
  DCD=UP  DTR=UP  DSR=UP  RTS=UP  CTS=UP


------------------------------------------------------------------------------
<ROUTER_02>dis int ser 0/0
Serial0/0 current state :UP
Line protocol current state :UP
Description : Serial0/0 Interface
Virtualbaudrate is 512000 bps
The Maximum Transmit Unit is 1500, Hold timer is 10(sec)
Internet Address is 10.10.10.2/24
Link layer protocol is PPP
LCP opened, IPCP opened, OSICP opened
Output queue : (Urgent queuing : Size/Length/Discards)  0/50/0
Output queue : (Protocol queuing : Size/Length/Discards) 0/500/0
Output queue : (Priority queuing : PQL 1 Size/Length/Discards)
Top:  0/60/0    Middle:  0/40/0    Normal:  0/60/0    Bottom:  0/100/0
Physical layer is synchronous,
Interface is DTE, Cable type is V35
Last clearing of counters: 00:08:02 UTC Sun 06/20/2010
  Last 300 seconds input rate 29666.75 bytes/sec, 237334 bits/sec, 172.53 packets/sec
  Last 300 seconds output rate 24604.34 bytes/sec, 196834 bits/sec, 160.91 packets/sec
  Input: 210690 packets, 33577786 bytes
      0 broadcasts, 0 multicasts
      0 errors, 0 runts, 0 giants
      0 CRC, 0 align errors, 0 overruns
      0 dribbles, 0 aborts, 0 no buffers
      0 frame errors
  Output:195896 packets, 23355313 bytes
      0 errors, 0 underruns, 0 collisions
      0 deferred
  DCD=UP  DTR=UP  DSR=UP  RTS=UP  CTS=UP
```

5.2.6 - ACL – Momento 03

ANTES:

ROUTER_02>dis acl all
Total ACL Number: 2

Advanced ACL 3300, 1 rule
Acl's step is 1
rule 0 permit udp source-port range 16384 32768 destination-port range 16384 32768 (0 times matched)

Advanced ACL 3301, 1 rule
Acl's step is 1
rule 0 permit ip (242228 times matched)

[ROUTER_01]dis acl all
Total ACL Number: 2

Advanced ACL 3300, 1 rule
Acl's step is 1
rule 0 permit udp source-port range 16384 32768 destination-port range 16384 32768 (0 times matched)

Advanced ACL 3301, 1 rule
Acl's step is 1
rule 0 permit ip (14205 times matched)

DEPOIS:

<ROUTER_01>dis acl all
Total ACL Number: 2

Advanced ACL 3300, 1 rule
Acl's step is 1
rule 0 permit udp source-port range 16384 32768 destination-port range 16384 32768 (9680 times matched)

Advanced ACL 3301, 1 rule
Acl's step is 1
rule 0 permit ip (144077 times matched)

<ROUTER_02>dis acl all
Total ACL Number: 2

Advanced ACL 3300, 1 rule
Acl's step is 1
rule 0 permit udp source-port range 16384 32768 destination-port range 16384 32768 (6221 times matched)

Advanced ACL 3301, 1 rule
Acl's step is 1
rule 0 permit ip (263517 times matched)

5.2.7 – Análise de dados – Cenário 02

Verifica-se que após a aplicação do QoS, a ACL que identifica os pacotes de voz é incrementada.

Fato que confirma o funcionamento da política adotada.

Tabela 21 - Análise cenário 02

ACL - Match - Regra			
Roteador_01		Roteador_02	
Antes	0	Antes	0
Depois	9680	Depois	6221

5.3 - Cenário - 03 – Tráfego com QoS WFQ (Weighted Fair Queuing)

5.3.1 – Topologia adotada

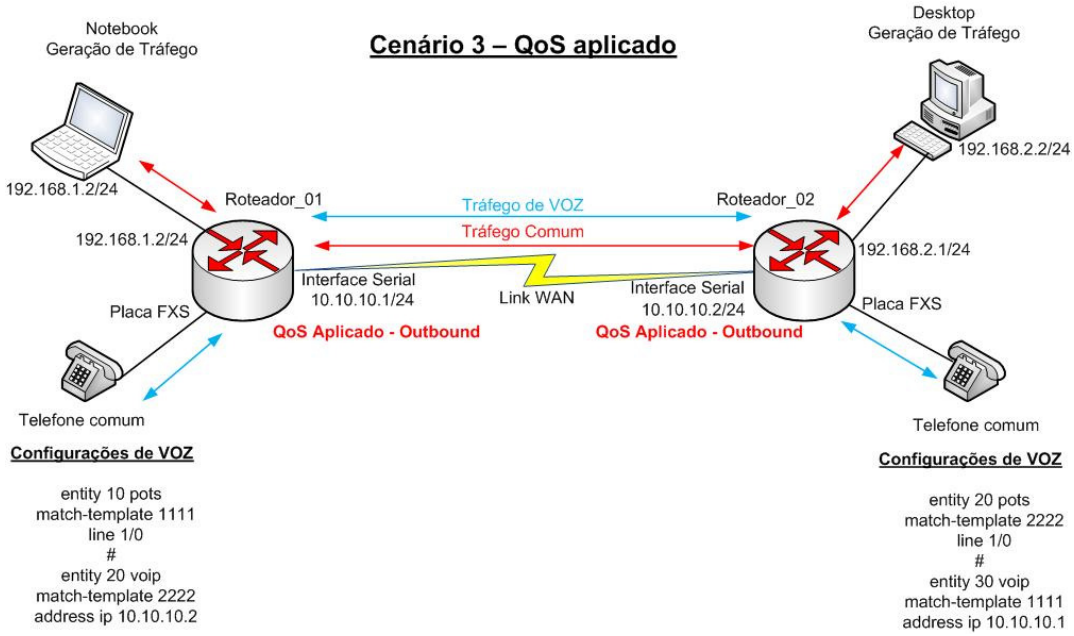


Figura 27 - Topologia Cenário 03

5.3.2 – Resultados Esperados / Obtidos

Tabela 22 – Cenário 3/Momento: Resultados obtidos

CENÁRIO 03 - CHAMADAS REALIZADAS SEM USO DO QoS:		
Nr. Chamada:	**Resultado:**	**Observações:**
1	**OK**	Chamada realizada com sucesso - Qualidade OK
2	**OK**	Chamada realizada com sucesso - Qualidade OK
3	**OK**	Chamada realizada com sucesso - Qualidade OK
4	**OK**	Chamada realizada com sucesso - Qualidade OK
5	**OK**	Chamada realizada com sucesso - Qualidade OK

➔ Os resultados esperados para o momento 01 foram atingidos com sucesso.

Tabela 23 - Cenário 3/Momento 2: Resultados obtidos

CENÁRIO 03 - CHAMADAS REALIZADAS DURANTE CONGESTIONAMENTO SEM QoS:			
Nr. Chamada:	**Resultado:**	**Duração:**	**Observações:**
1	**NOK**	1 Minuto	Chamada estabelecida porém conversação e qualidade seriamente comprometidos
2	**NOK**	1 Minuto	Chamada estabelecida porém conversação e qualidade seriamente comprometidos
3	**NOK**	1 Minuto	Chamada estabelecida porém conversação e qualidade seriamente comprometidos
4	**NOK**	1 Minuto	Chamada estabelecida porém conversação e qualidade seriamente comprometidos
5	**NOK**	1 Minuto	Chamada estabelecida porém conversação e qualidade seriamente comprometidos

➔ Os resultados esperados para o momento 02 foram atingidos com sucesso.

Tabela 24 - Cenário 3/Momento 3: Resultados obtidos

CENÁRIO 03 - CHAMADAS REALIZADAS DURANTE CONGESTIONAMENTO UTILIZANDO QoS - WFQ			
Nr. Chamada:	**Resultado:**	**Duração:**	**Observações:**
1	**OK**	1 Minuto	Mesmo durante congestionamento as chamadas apresentaram qualidade - OK
2	**OK**	1 Minuto	Mesmo durante congestionamento as chamadas apresentaram qualidade - OK
3	**OK**	1 Minuto	Mesmo durante congestionamento as chamadas apresentaram qualidade - OK
4	**OK**	1 Minuto	Mesmo durante congestionamento as chamadas apresentaram qualidade - OK
5	**OK**	1 Minuto	Mesmo durante congestionamento as chamadas apresentaram qualidade - OK

➔ Os resultados esperados para o momento 03 foram atingidos com sucesso.

Anotações:

- Ao iniciar a transferência de dados a chamada apresentou o mesmo comportamento relatado no cenário 01 (Problemas de voz);
- Após a aplicação das políticas de QoS nas interfaces:
- Não houve picotamento na voz;
- A mensagem estava clara e de perfeito entendimento para ambos os lados.

5.3.3 - Dados coletados

Os dados referentes aos momentos 01 e 02 coletados durante o experimento se assemelham com os coletados no cenário 01.

Ficando adotados como padrão de comparação para esse experimento.

5.3.4 - Historico de ligações – Momento – 03:

```
<ROUTER_01>dis voice call-his last 10
 #
 CallRecord [ 36]:
   CallerNum            = 1111
   CalledNum            = 2222
   EncodeType           = 729r8
   PeerAddress          = 10.10.10.2
   DisconnectCause      = 12
   DisconnectText       = Caller hook on
   TalkingTimes         = 00h 01m 09s
   VoiceTimes           = 00h 01m 09s
   FaxTimes             = 00h 00m 00s
   ImgPages             = 0
   CallDirection        = 1
   SetupTime(voip)      = Jun 20, 2010 01:37:27
   ConnectTime(voip)    = Jun 20, 2010 01:37:28
   DisconectTime(voip)  = Jun 20, 2010 01:38:37
   Transmit(voip)       = 2282 (packages): 95844 (bytes)
   Received(voip)       = 2280 (packages): 95760 (bytes)
   SetupTime(pstn)      = Jun 20, 2010 01:37:24
   ConnectTime(pstn)    = Jun 20, 2010 01:37:24
   DisconectTime(pstn)  = Jun 20, 2010 01:38:37
   Transmit(pstn)       = 2280 (packages): 68400 (bytes)
   Received(pstn)       = 2282 (packages): 68460 (bytes)

 #
 CallRecord [ 35]:
   CallerNum            = 1111
   CalledNum            = 2222
   EncodeType           = 729r8
   PeerAddress          = 10.10.10.2
   DisconnectCause      = 12
   DisconnectText       = Caller hook on
   TalkingTimes         = 00h 01m 08s
   VoiceTimes           = 00h 01m 08s
   FaxTimes             = 00h 00m 00s
   ImgPages             = 0
   CallDirection        = 1
   SetupTime(voip)      = Jun 20, 2010 01:36:13
   ConnectTime(voip)    = Jun 20, 2010 01:36:14
   DisconectTime(voip)  = Jun 20, 2010 01:37:22
   Transmit(voip)       = 2267 (packages): 95214 (bytes)
   Received(voip)       = 2267 (packages): 95214 (bytes)
   SetupTime(pstn)      = Jun 20, 2010 01:36:11
   ConnectTime(pstn)    = Jun 20, 2010 01:36:11
   DisconectTime(pstn)  = Jun 20, 2010 01:37:22
   Transmit(pstn)       = 2267 (packages): 68010 (bytes)
   Received(pstn)       = 2267 (packages): 68010 (bytes)
```

```
#
CallRecord [ 34]:
  CallerNum          = 1111
  CalledNum          = 2222
  EncodeType         = 729r8
  PeerAddress        = 10.10.10.2
  DisconnectCause    = 12
  DisconnectText     = Caller hook on
  TalkingTimes       = 00h 01m 39s
  VoiceTimes         = 00h 01m 39s
  FaxTimes           = 00h 00m 00s
  ImgPages           = 0
  CallDirection      = 1
  SetupTime(voip)    = Jun 20, 2010 01:34:30
  ConnectTime(voip)  = Jun 20, 2010 01:34:30
  DisconectTime(voip)  = Jun 20, 2010 01:36:09
  Transmit(voip)     = 3268 (packages): 137256 (bytes)
  Received(voip)     = 3269 (packages): 137298 (bytes)
  SetupTime(pstn)    = Jun 20, 2010 01:34:27
  ConnectTime(pstn)  = Jun 20, 2010 01:34:27
  DisconectTime(pstn)  = Jun 20, 2010 01:36:09
  Transmit(pstn)     = 3269 (packages): 98070 (bytes)
  Received(pstn)     = 3268 (packages): 98040 (bytes)

#
CallRecord [ 33]:
  CallerNum          = 1111
  CalledNum          = 2222
  EncodeType         = 729r8
  PeerAddress        = 10.10.10.2
  DisconnectCause    = 12
  DisconnectText     = Caller hook on
  TalkingTimes       = 00h 01m 18s
  VoiceTimes         = 00h 01m 18s
  FaxTimes           = 00h 00m 00s
  ImgPages           = 0
  CallDirection      = 1
  SetupTime(voip)    = Jun 20, 2010 01:33:06
  ConnectTime(voip)  = Jun 20, 2010 01:33:07
  DisconectTime(voip)  = Jun 20, 2010 01:34:25
  Transmit(voip)     = 2585 (packages): 108570 (bytes)
  Received(voip)     = 2579 (packages): 108318 (bytes)
  SetupTime(pstn)    = Jun 20, 2010 01:33:04
  ConnectTime(pstn)  = Jun 20, 2010 01:33:04
  DisconectTime(pstn)  = Jun 20, 2010 01:34:25
  Transmit(pstn)     = 2579 (packages): 77370 (bytes)
  Received(pstn)     = 2585 (packages): 77550 (bytes)

#
CallRecord [ 32]:
```

```
    CallerNum            = 1111
    CalledNum            = 2222
    EncodeType           = 729r8
    PeerAddress          = 10.10.10.2
    DisconnectCause      = 12
    DisconnectText       = Caller hook on
    TalkingTimes         = 00h 01m 43s
    VoiceTimes           = 00h 01m 43s
    FaxTimes             = 00h 00m 00s
    ImgPages             = 0
    CallDirection        = 1
    SetupTime(voip)      = Jun 20, 2010 01:31:18
    ConnectTime(voip)    = Jun 20, 2010 01:31:19
    DisconectTime(voip)  = Jun 20, 2010 01:33:02
    Transmit(voip)       = 3416 (packages): 143472 (bytes)
    Received(voip)       = 3414 (packages): 143388 (bytes)
    SetupTime(pstn)      = Jun 20, 2010 01:31:16
    ConnectTime(pstn)    = Jun 20, 2010 01:31:16
    DisconectTime(pstn)  = Jun 20, 2010 01:33:02
    Transmit(pstn)       = 3414 (packages): 102420 (bytes)
    Received(pstn)       = 3416 (packages): 102480 (bytes)

<ROUTER_02>dis voice call-his last 10
  #
  CallRecord :
    CallerNum            = 1111
    CalledNum            = 2222
    EncodeType           = 729r8
    PeerAddress          = 10.10.10.1
    DisconnectCause      = 12
    DisconnectText       = Caller hook on
    TalkingTimes         = 00h 01m 09s
    VoiceTimes           = 00h 01m 09s
    FaxTimes             = 00h 00m 00s
    ImgPages             = 0
    CallDirection        = 2
    SetupTime(voip)      = Jun 20, 2010 01:38:00
    ConnectTime(voip)    = Jun 20, 2010 01:38:00
    DisconectTime(voip)  = Jun 20, 2010 01:39:09
    Transmit(voip)       = 2287 (packages): 96054 (bytes)
    Received(voip)       = 2282 (packages): 95844 (bytes)
    SetupTime(pstn)      = Jun 20, 2010 01:38:00
    ConnectTime(pstn)    = Jun 20, 2010 01:38:00
    DisconectTime(pstn)  = Jun 20, 2010 01:39:09
    Transmit(pstn)       = 2282 (packages): 68460 (bytes)
    Received(pstn)       = 2287 (packages): 68610 (bytes)

  #
  CallRecord :
    CallerNum            = 1111
```

```
    CalledNum            = 2222
    EncodeType           = 729r8
    PeerAddress          = 10.10.10.1
    DisconnectCause      = 12
    DisconnectText       = Caller hook on
    TalkingTimes         = 00h 01m 09s
    VoiceTimes           = 00h 01m 09s
    FaxTimes             = 00h 00m 00s
    ImgPages             = 0
    CallDirection        = 2
    SetupTime(voip)      = Jun 20, 2010 01:36:46
    ConnectTime(voip)    = Jun 20, 2010 01:36:46
    DisconectTime(voip)  = Jun 20, 2010 01:37:55
    Transmit(voip)       = 2271 (packages): 95382 (bytes)
    Received(voip)       = 2267 (packages): 95214 (bytes)
    SetupTime(pstn)      = Jun 20, 2010 01:36:46
    ConnectTime(pstn)    = Jun 20, 2010 01:36:46
    DisconectTime(pstn)  = Jun 20, 2010 01:37:55
    Transmit(pstn)       = 2267 (packages): 68010 (bytes)
    Received(pstn)       = 2271 (packages): 68130 (bytes)

  #
  CallRecord :
    CallerNum            = 1111
    CalledNum            = 2222
    EncodeType           = 729r8
    PeerAddress          = 10.10.10.1
    DisconnectCause      = 12
    DisconnectText       = Caller hook on
    TalkingTimes         = 00h 01m 39s
    VoiceTimes           = 00h 01m 39s
    FaxTimes             = 00h 00m 00s
    ImgPages             = 0
    CallDirection        = 2
    SetupTime(voip)      = Jun 20, 2010 01:35:02
    ConnectTime(voip)    = Jun 20, 2010 01:35:03
    DisconectTime(voip)  = Jun 20, 2010 01:36:42
    Transmit(voip)       = 3274 (packages): 137508 (bytes)
    Received(voip)       = 3268 (packages): 137256 (bytes)
    SetupTime(pstn)      = Jun 20, 2010 01:35:02
    ConnectTime(pstn)    = Jun 20, 2010 01:35:02
    DisconectTime(pstn)  = Jun 20, 2010 01:36:42
    Transmit(pstn)       = 3268 (packages): 98040 (bytes)
    Received(pstn)       = 3274 (packages): 98220 (bytes)

  #
  CallRecord :
    CallerNum            = 1111
    CalledNum            = 2222
    EncodeType           = 729r8
```

```
  PeerAddress          = 10.10.10.1
  DisconnectCause      = 12
  DisconnectText       = Caller hook on
  TalkingTimes         = 00h 01m 18s
  VoiceTimes           = 00h 01m 18s
  FaxTimes             = 00h 00m 00s
  ImgPages             = 0
  CallDirection        = 2
  SetupTime(voip)      = Jun 20, 2010 01:33:38
  ConnectTime(voip)    = Jun 20, 2010 01:33:39
  DisconectTime(voip)  = Jun 20, 2010 01:34:57
  Transmit(voip)       = 2586 (packages): 108612 (bytes)
  Received(voip)       = 2585 (packages): 108570 (bytes)
  SetupTime(pstn)      = Jun 20, 2010 01:33:38
  ConnectTime(pstn)    = Jun 20, 2010 01:33:38
  DisconectTime(pstn)  = Jun 20, 2010 01:34:57
  Transmit(pstn)       = 2585 (packages): 77550 (bytes)
  Received(pstn)       = 2586 (packages): 77580 (bytes)

#
CallRecord :
  CallerNum            = 1111
  CalledNum            = 2222
  EncodeType           = 729r8
  PeerAddress          = 10.10.10.1
  DisconnectCause      = 12
  DisconnectText       = Caller hook on
  TalkingTimes         = 00h 01m 43s
  VoiceTimes           = 00h 01m 43s
  FaxTimes             = 00h 00m 00s
  ImgPages             = 0
  CallDirection        = 2
  SetupTime(voip)      = Jun 20, 2010 01:31:50
  ConnectTime(voip)    = Jun 20, 2010 01:31:51
  DisconectTime(voip)  = Jun 20, 2010 01:33:34
  Transmit(voip)       = 3417 (packages): 143514 (bytes)
  Received(voip)       = 3416 (packages): 143472 (bytes)
  SetupTime(pstn)      = Jun 20, 2010 01:31:50
  ConnectTime(pstn)    = Jun 20, 2010 01:31:50
  DisconectTime(pstn)  = Jun 20, 2010 01:33:34
  Transmit(pstn)       = 3416 (packages): 102480 (bytes)
  Received(pstn)       = 3417 (packages): 102510 (bytes)
```

5.3.5 - Interfaces seriais – Momento_03:

ANTES:

```
<ROUTER_02>dis int ser 0/0
Serial0/0 current state :UP
Line protocol current state :UP
```

```
Description : Serial0/0 Interface
Virtualbaudrate is 512000 bps
The Maximum Transmit Unit is 1500, Hold timer is 10(sec)
Internet Address is 10.10.10.2/24
Link layer protocol is PPP
LCP opened, IPCP opened, OSICP opened
Output queue : (Urgent queuing : Size/Length/Discards)  0/50/0
Output queue : (Protocol queuing : Size/Length/Discards) 0/500/0
Output queue : (Class Based queuing : Size/Discards)  0/0
   Queue Size: 0/0/0 (EF/AF/BE)
   BE queues:  0/0/256 (Active/Max active/Total)
   AF queues:  0 (Allocated)
   Bandwidth(Kbps): 345/409 (Available/Max reserve)

Physical layer is synchronous,
Interface is DTE, Cable type is V35
Last clearing of counters: 01:26:09 UTC Sun 06/20/2010
   Last 300 seconds input rate 0.00 bytes/sec, 0 bits/sec, 0.00 packets/sec
   Last 300 seconds output rate 0.00 bytes/sec, 0 bits/sec, 0.00 packets/sec
   Input: 540 packets, 27937 bytes
        0 broadcasts, 0 multicasts
        0 errors, 0 runts, 0 giants
        0 CRC, 0 align errors, 0 overruns
        0 dribbles, 0 aborts, 0 no buffers
        0 frame errors
   Output:529 packets, 27400 bytes
        0 errors, 0 underruns, 0 collisions
        0 deferred
   DCD=UP  DTR=UP  DSR=UP  RTS=UP  CTS=UP

<ROUTER_01>dis int ser 0/0
Serial0/0 current state :UP
Line protocol current state :UP
Description : Serial0/0 Interface
The Maximum Transmit Unit is 1500, Hold timer is 10(sec)
Internet Address is 10.10.10.1/24
Link layer protocol is PPP
LCP opened, IPCP opened, OSICP opened
Output queue : (Urgent queuing : Size/Length/Discards)  0/50/0
Output queue : (Protocol queuing : Size/Length/Discards) 0/500/0
Output queue : (Class Based queuing : Size/Discards)  0/0
   Queue Size: 0/0/0 (EF/AF/BE)
   BE queues:  0/0/256 (Active/Max active/Total)
   AF queues:  0 (Allocated)
   Bandwidth(Kbps): 409536/409600 (Available/Max reserve)

Physical layer is synchronous,Baudrate is 512000 bps
Interface is DCE, Cable type is V35
Last clearing of counters: 01:26:04 UTC Sun 06/20/2010
   Last 300 seconds input rate 0.00 bytes/sec, 0 bits/sec, 0.00 packets/sec
```

Last 300 seconds output rate 0.00 bytes/sec, 0 bits/sec, 0.00 packets/sec
Input: 322 packets, 16664 bytes
0 broadcasts, 0 multicasts
0 errors, 0 runts, 0 giants
0 CRC, 0 align errors, 0 overruns
0 dribbles, 0 aborts, 0 no buffers
0 frame errors
Output:322 packets, 16704 bytes
0 errors, 0 underruns, 0 collisions
0 deferred
DCD=UP DTR=UP DSR=UP RTS=UP CTS=UP

<ROUTER_01>

DEPOIS:

<ROUTER_02>dis int ser 0/0
Serial0/0 current state :UP
Line protocol current state :UP
Description : Serial0/0 Interface
Virtualbaudrate is 512000 bps
The Maximum Transmit Unit is 1500, Hold timer is 10(sec)
Internet Address is 10.10.10.2/24
Link layer protocol is PPP
LCP opened, IPCP opened, OSICP opened
Output queue : (Urgent queuing : Size/Length/Discards) 0/50/0
Output queue : (Protocol queuing : Size/Length/Discards) 0/500/0
Output queue : (Class Based queuing : Size/Discards) 0/0
Queue Size: 0/0/0 (EF/AF/BE)
BE queues: 0/15/256 (Active/Max active/Total)
AF queues: 0 (Allocated)
Bandwidth(Kbps): 345/409 (Available/Max reserve)

Physical layer is synchronous,
Interface is DTE, Cable type is V35
Last clearing of counters: 01:26:09 UTC Sun 06/20/2010
Last 300 seconds input rate 47171.14 bytes/sec, 377369 bits/sec, 201.83 packets/sec
Last 300 seconds output rate 31366.73 bytes/sec, 250933 bits/sec, 197.48 packets/sec
Input: 156203 packets, 29087518 bytes
0 broadcasts, 0 multicasts
0 errors, 0 runts, 0 giants
0 CRC, 0 align errors, 0 overruns
0 dribbles, 0 aborts, 0 no buffers
0 frame errors
Output:153592 packets, 19698461 bytes
0 errors, 0 underruns, 0 collisions
0 deferred
DCD=UP DTR=UP DSR=UP RTS=UP CTS=UP

```
<ROUTER_02>

<ROUTER_01>dis int ser 0/0
Serial0/0 current state :UP
Line protocol current state :UP
Description : Serial0/0 Interface
The Maximum Transmit Unit is 1500, Hold timer is 10(sec)
Internet Address is 10.10.10.1/24
Link layer protocol is PPP
LCP opened, IPCP opened, OSICP opened
Output queue : (Urgent queuing : Size/Length/Discards)  0/50/0
Output queue : (Protocol queuing : Size/Length/Discards) 0/500/0
Output queue : (Class Based queuing : Size/Discards)  0/7
   Queue Size: 0/0/0 (EF/AF/BE)
   BE queues:  0/15/256 (Active/Max active/Total)
   AF queues:  0 (Allocated)
   Bandwidth(Kbps): 409536/409600 (Available/Max reserve)

Physical layer is synchronous,Baudrate is 512000 bps
Interface is DCE, Cable type is V35
Last clearing of counters: 01:26:04 UTC Sun 06/20/2010
   Last 300 seconds input rate 33594.89 bytes/sec, 268759 bits/sec, 172.23 packets/sec
   Last 300 seconds output rate 42660.60 bytes/sec, 341284 bits/sec, 174.00 packets/sec
   Input: 149758 packets, 19499992 bytes
        0 broadcasts, 0 multicasts
        0 errors, 0 runts, 0 giants
        0 CRC, 0 align errors, 0 overruns
        0 dribbles, 0 aborts, 0 no buffers
        0 frame errors
   Output:152358 packets, 28888030 bytes
        0 errors, 0 underruns, 0 collisions
        0 deferred
   DCD=UP  DTR=UP  DSR=UP  RTS=UP  CTS=UP

<ROUTER_01>
```

5.3.6 - QoS – Cenário 03/Momento 03:

ANTES:

```
<ROUTER_01>dis qos pol int

 Interface: Serial0/0

 Direction: Outbound

 Policy: TESTE

  Classifier: default-class
```

```
    Matched : 3842/184965 (Packets/Bytes)
    1 Min Rate: 14678 bps
    Rule(s) : if-match any
    Behavior: be
    Default Queue:
     Flow Based Weighted Fair Queuing
      Max number of hashed queues: 256
      Matched  : 0/0 (Packets/Bytes)
      Enqueued : 0/0 (Packets/Bytes)
      Discarded: 0/0 (Packets/Bytes)
      Discard Method: Tail

  Classifier: VOZ
    Matched : 0/0 (Packets/Bytes)
    1 Min Rate: 0 bps
    Operator: AND
    Rule(s) : if-match rtp start-port 2000 end-port 65535
    Behavior: VOZ
     Marking:
      Remark DSCP ef
      Remarked: 0 (Packets)
     Expedited Forwarding:
      Bandwidth 64 (Kbps), CBS 1500 (Bytes)
      Matched  : 0/0 (Packets/Bytes)
      Enqueued : 0/0 (Packets/Bytes)
      Discarded: 0/0 (Packets/Bytes)
<ROUTER_01>

<ROUTER_02>dis qos pol int

 Interface: Serial0/0

 Direction: Outbound

 Policy: TESTE

  Classifier: default-class
    Matched : 11110/534844 (Packets/Bytes)
    5 Min Rate: 12326 bps
    Rule(s) : if-match any
    Behavior: be
    Default Queue:
     Flow Based Weighted Fair Queuing
      Max number of hashed queues: 256
      Matched  : 0/0 (Packets/Bytes)
      Enqueued : 0/0 (Packets/Bytes)
      Discarded: 0/0 (Packets/Bytes)

  Classifier: VOZ
    Matched : 0/0 (Packets/Bytes)
```

```
    5 Min Rate: 0 bps
    Operator: AND
    Rule(s) : if-match rtp start-port 2000 end-port 65535
    Behavior: VOZ
     Marking:
      Remark DSCP ef
      Remarked: 0 (Packets)
     Expedited Forwarding:
      Bandwidth 64 (Kbps), CBS 1500 (Bytes)
      Matched  : 0/0 (Packets/Bytes)
      Enqueued : 0/0 (Packets/Bytes)
      Discarded: 0/0 (Packets/Bytes)
<ROUTER_02>
<ROUTER_02>
-------------------------------------------------------------------------------
<ROUTER_01>dis qos pol int

 Interface: Serial0/0

 Direction: Outbound

 Policy: TESTE

  Classifier: default-class
   Matched : 140053/27474572 (Packets/Bytes)
   1 Min Rate: 162633 bps
   Rule(s) : if-match any
   Behavior: be
   Default Queue:
    Flow Based Weighted Fair Queuing
     Max number of hashed queues: 256
     Matched  : 66307/18765967 (Packets/Bytes)
     Enqueued : 66300/18760551 (Packets/Bytes)
     Discarded: 7/5416 (Packets/Bytes)
     Discard Method: Tail

  Classifier: VOZ
   Matched : 13881/970410 (Packets/Bytes)
   1 Min Rate: 2239 bps
   Operator: AND
   Rule(s) : if-match rtp start-port 2000 end-port 65535
   Behavior: VOZ
    Marking:
     Remark DSCP ef
     Remarked: 13881 (Packets)
    Expedited Forwarding:
     Bandwidth 64 (Kbps), CBS 1500 (Bytes)
     Matched  : 7185/531230 (Packets/Bytes)
     Enqueued : 7185/531230 (Packets/Bytes)
     Discarded: 0/0 (Packets/Bytes)
```

```
<ROUTER_01>

<ROUTER_02>dis qos pol int

 Interface: Serial0/0

 Direction: Outbound

 Policy: TESTE

  Classifier: default-class
    Matched : 152546/18738413 (Packets/Bytes)
    5 Min Rate: 187793 bps
    Rule(s) : if-match any
    Behavior: be
    Default Queue:
     Flow Based Weighted Fair Queuing
       Max number of hashed queues: 256
       Matched  : 31913/10655460 (Packets/Bytes)
       Enqueued : 31913/10655460 (Packets/Bytes)
       Discarded: 0/0 (Packets/Bytes)

  Classifier: VOZ
    Matched : 13899/971650 (Packets/Bytes)
    5 Min Rate: 11184 bps
    Operator: AND
    Rule(s) : if-match rtp start-port 2000 end-port 65535
    Behavior: VOZ
     Marking:
       Remark DSCP ef
       Remarked: 13899 (Packets)
     Expedited Forwarding:
       Bandwidth 64 (Kbps), CBS 1500 (Bytes)
       Matched  : 6293/465142 (Packets/Bytes)
       Enqueued : 6293/465142 (Packets/Bytes)
       Discarded: 0/0 (Packets/Bytes)
<ROUTER_02>
```

5.3.7 - Analise de dados – Cenário 03

Importante destacar que após aplicação de política de QoS o classificador de VOZ, classificou e enfileirou os pacotes.

Evidênciando que não houve perda de pacotes.

Confirmando resoltados da experimentação (Sem problemas de VOZ).

5.4 – Analise final:

Apresentados os cenários 01, que nos foi útil para apresentação do problema, e os cenários 02 e 03, que efetivamente solucinaram o problema apresentado, fica baseado nos seguites fatores a escolha da política de QoS WFQ como sendo a melhor dentre as testadas para priorizar o tráfego escolhido.

Fatores:

- Atendeu a todas as expectativas;
- Apresenta configuração com nível de dificuldade mediano;
- Altamente customizável;
- Apresentou sucesso em todos os testes realizados em bancada.

5.4.1 – Comparativo de Tecnologias Utilizadas

Tabela 25 - Comparativo de tecnologias de Gerenciamento de congestionamento

Tipo	Qtd. Filas	Vantagens	Desvantagens
FIFO	1	Não Necessita de configuração;	Todos os pacotes são tratados da mesma forma
		Fácil operação	Sem garantia de entrega para pacotes RTP (Ex: Voz)
PQ	4	Prove serviço de preferência absoluta para determinado tráfego	Necessidade de configuração
		Garante baixo atraso para aplicações Real-time	Gera alto overhead
		Garante largura de banda determinada para a aplicação	Os pacotes com baixa prioridade podem ficar longos períodos sem ser enviados
CQ	17	Possível configurar proporção de banda por aplicação	Necessidade de configuração
		Realocação de banda de filas não utlizadas	Gera overhead
WFQ	Customizável	Capacidade de garantia de banda por IP de orígem	Overhead é maior que o FIFO
		Capacidade de reduzir JITTER	Overhead é menos que o PQ e CQ
		Capaz de reduzir o atraso - Delay	
		Capacidade de alocar banda para tráfegos com diferentes prioridades	
		Frente a eventual diminuiçao do tráfego aumenta-se a banda dos existentes	

CAPÍTULO 6 – CONCLUSÃO

6.1 – Conclusões

As conclusões aqui descritas mostram o resultado final a que se propôs o trabalho. O resultado final desse trabalho é um conjunto de recomendações de como as confiugurações devem ser escolhidas, utilizadas e aplicadas em redes WAN que estão em funcionamento ou que virão a ser instaladas.

Sendo assim foram efetuadas diversos testes nos cenários descritos nos capítulos anteriores, onde pudemos verificar o comportamento do tráfego de voz e qualidade de serviço, levando em consideração itens de controle pré-determinados. A comparação desses itens ofereceu informações para a decisão da melhor política de QoS a ser utilizada dentre as testadas, o WFQ – cenário 03.

A escolha pelo cenário 03 não se deve apenas pelo fato de se priorizar o tráfego de voz com sucesso, feito realizado também no cenário 02, mas também pela versatilidade e capacidade de customização oferecida pelo WFQ, fatores que foram decisivos para a eleição.

Como explanado no início deste trabalho, as redes WAN são utilizadas em larga escala, desde pequenos escritórios a multinacionais, se apresentando com diversas capacidades de link, com isso se torna necessário atenção ao tipo de tráfego que atravessa esses links bem como a tratativa que deve ser utilizada para serviços críticos como voz e vídeo.

Por fim, e pelos resultados obtidos e apresentados, tem-se aqui um referencial para aplicação no dia a dia das empresas de esquemas de QoS priorizando voz, e com potencial para aplicação em qualquer tipo de tráfago crítico em uma rede WAN.

6.2 – Sugestões para trabalhos futuros

Como sugestão para continuidade e trabalhos futuros, aconselho o trabalho com QoS visando priorizar não apenas o tráfego de voz, mas sim divesos tráfegos críticos simultaneamente.

Segue configuração topologia e configurações sugeridas:

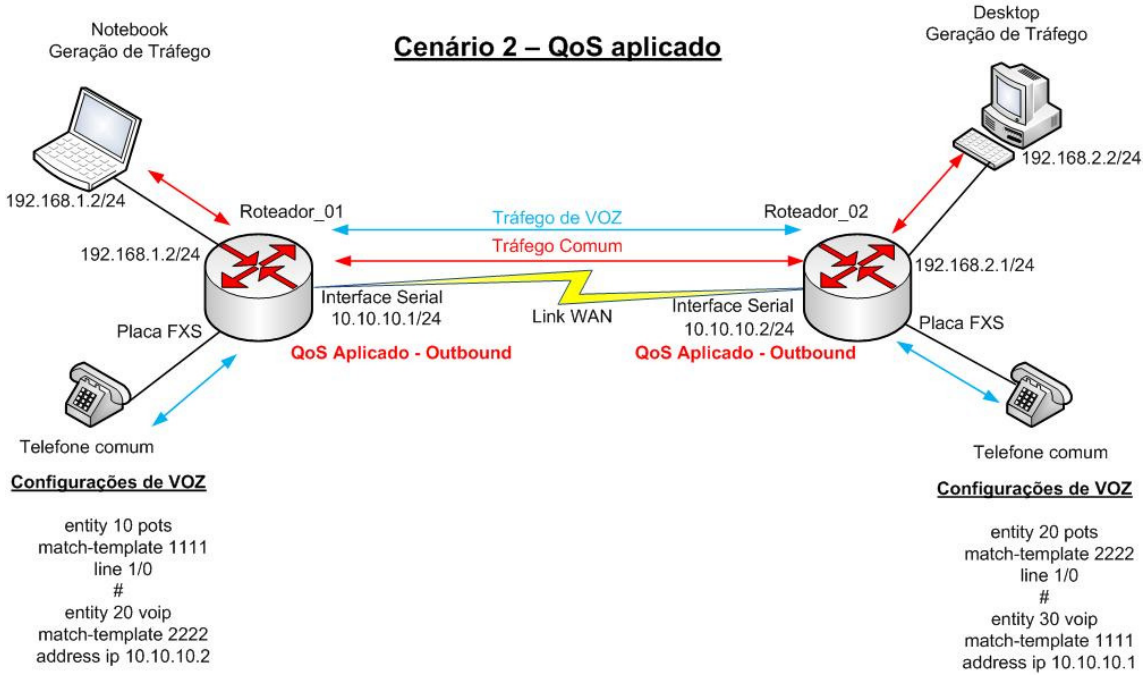


Figura 28 – Topologia sugerida

Configuração sugerida:

```
[ROUTER_01]
[ROUTER_01]dis cur
#
 sysname ROUTER_01
#
 super password level 3 simple huawei
#
 flow-interval qos 1
#
radius scheme system
#
domain system
#
local-user admin
 password cipher .]@USE=B,53Q=^Q`MAF4<1!!
 service-type telnet terminal
 level 3
 service-type ftp
#
traffic classifier VOZ operator and
 if-match rtp start-port 2000 end-port 65535
traffic classifier SINALIZACAO operator and
 if-match acl 3002
traffic classifier MULTIMIDIA operator and
 if-match acl 3003
traffic classifier DADOS_EXPRESSO operator and
 if-match acl 3004
```

```
#
traffic behavior VOZ
 remark dscp ef
 queue ef bandwidth 64 cbs 1500
traffic behavior SINALIZACAO
 remark dscp af31
 queue af bandwidth 64
traffic behavior MULTIMIDIA
 remark dscp af41
 queue af bandwidth 64
traffic behavior DADOS_EXPRESSO
 remark dscp af21
 queue af bandwidth 64


qos policy TESTE
 classifier VOZ behavior VOZ
#
acl number 3001
 description VOZ
 rule 0 permit udp destination-port eq 5000
acl number 3002
 description SINALIZACAO
 rule 0 permit udp destination-port eq 5001
acl number 3003
 description MULTIMIDIA
 rule 0 permit udp destination-port eq 5002
acl number 3004
 description DADOS_EXPRESSO
 rule 0 permit udp destination-port eq 5003
acl number 3300
 rule 0 permit udp source-port range 16384 32768 destination-port range 16384 32768
acl number 3301
 rule 0 permit ip
#
interface Aux0
 async mode flow
#
interface Ethernet0/0
 ip address 192.168.1.1 255.255.255.0
#
interface Serial0/0
 baudrate 512000
 link-protocol ppp
 ip address 10.10.10.1 255.255.255.0
 qos pq pql 1
#
interface NULL0
#
#
```

```
voice-setup
 #
 aaa-client
 #
 dial-program
  #
  entity 10 pots
    match-template 1111
```

REFERÊNCIAS

BAUGHER, M. et al. **Ther secure real-time transport protocol (SRTP).** RFC 3711, março de 2004.

CASNER, S.; JACOBSON, V. **Compressing IP/UDPRTP Headers for Low-Speed Serial Links.** RFC 2508, fevereiro de 1999.

COLCHER, S.; et al. **VoIP: Voz sobre IP**. Rio de Janeiro: Elsevier, 2005. 288 p. 5º Reimpressão. (Telecomunicações). ISBN 85-352-1787-8

FRIEDMAN, T. CACERES, R.; CLARK, A. **"RTP Control Protocol Extended Reports (RTCP XR)".** RFC 3611, novembro de 2003.

HUAWEI, 2009. **Command Manual (V3.53_01).** Disponível em: www.huawei.com - 3119A023-VRP3.4. Acesso em: 05 out. 2009

TANENBAUM, A.S. 1944 – **Redes de Computadores**; trad.: Vandenberg D. de Souza. – Rio de Janeiro: Elsevier, 2003 – 15° Reimpressão. 945p. ISBN 85-3521185-3