ESR

+ ***Multicast*** *- Introdução a comunicações de grupo em redes TCP/IP*

Objetivo: Enviar múltiplas cópias de dados a um grupo de hosts, que são identificados por um endereço multicast.

Hosts – ***Multicast Group Membership Discovery Protocols***

Nodos da Rede – ***Multicast Routing Protocols***

- ***Multicast Group Membership Discovery Protocols***

  Usado por hosts para informar a rede sobre a sua vontade de receber dados.

  Ex.: IGMP*, IPv4 e* ***Multicast Listener Discovery****, IPv6*

  Modos de Transmissão Multicast

  - ★ Any Source Multicast (***ASM***) – Host recebe dados de um determinado grupo e de qualquer fonte (Multicast Group);
  - ★ Source-specific Multicast (***SSM***) – Host recebe dados de um determinado grupo e de uma determinada fonte (Multicast Group + Source).

- ***Multicast Routing Protocols***

Permite comunicação entre nodos da rede que poderão não pertencer ao mesmo grupo multicast, através da construção de uma árvore de distribuição.

Técnicas de construção de uma árvore de distribuição

- ★ **Opt-in Protocols** – A construção da árvore é feita através dos pedidos de junção dos hosts;
- ★ **Opt-out Protocols** *(flood/prune)* – Assume que todos os nodos da rede querem receber os dados e, depois, caso não haja interessados, retira-se da árvore.

Tipo de árvore

★ **Source-based Trees**

a. Cada fonte cria a sua árvore de distribuição separada das restantes;
b. Cada árvore tem base no nodo adjacente à fonte;
c. Caso um router se quiser juntar ao grupo, tem de especificar a fonte e o grupo em questão (Multicast Group, Source).

Nota.:

- **SSM** requer **SBT**

Vantagens:

- ✓ Caminhos eficientes dos dados multicast;

Desvantagens:

- × Pouca escalabilidade (> número de fontes).

★ **Shared Trees**

a. É construída uma única árvore de distribuição entre todas as fontes para um determinado grupo;
b. Caso um router se quiser juntar ao grupo, tem de especificar apenas o grupo em questão (Multicast Group, *);
c. A árvore tem base num RP que poderá ser pré configurado ou eleito;
d. A entrega dos dados é **unidirecional**: cada pacote encapsulado é enviado para a raiz da árvore (através de unicast) e depois é descapsulado (através do protocolo PIM-SM) para ser enviado aos destinatários.

Vantagens:

- ✓ Boa escalabilidade (< número de fontes);

Desvantagens:

- × Caminhos ineficientes de dados;
- × Requer um mecanismo de seleção para o RP.

Determinar o Upstream Router

Algoritmo RPF (+ **MRIB** – Quando é utilizado o protocolo PIM)

Nota.:

- A interface do upstream router é utilizada como entrada e saída de pacotes, porém se estes chegarem por uma interface diferente da identificada pelo RPF, são descartados ou ignorados.

- ***Protocol Independent Multicast (PIM)***

Protocol	Opt-in / Opt-out	Supports SSM	Tree Type	Upstream Router Info Via
PIM-SM	Opt-in	Yes	Shared or source-based	MRIB
PIM-DM	Opt-out	Yes	Source-based	MRIB

★ **PIM-SM**

a. Protocolo *opt-in*, que utiliza, por default, *shared trees* mas também suporta *source based trees* para evitar encapsulamento, otimizar os caminhos e suportar SSM;
b. Protocolo *soft-state*, visto que o estado tem tempo de vida e mensagens têm de ser retransmitidas para manter o estado.

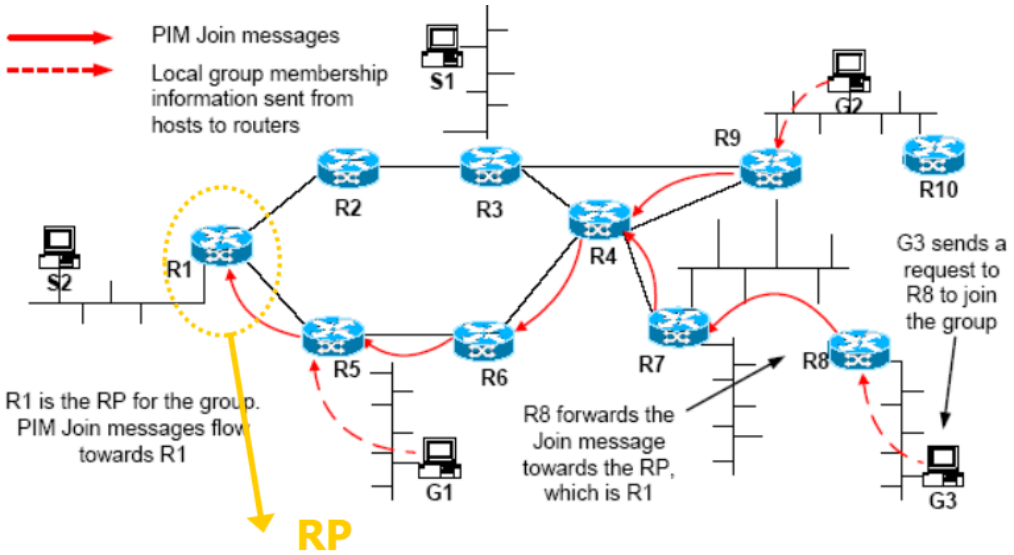
Vantagens:

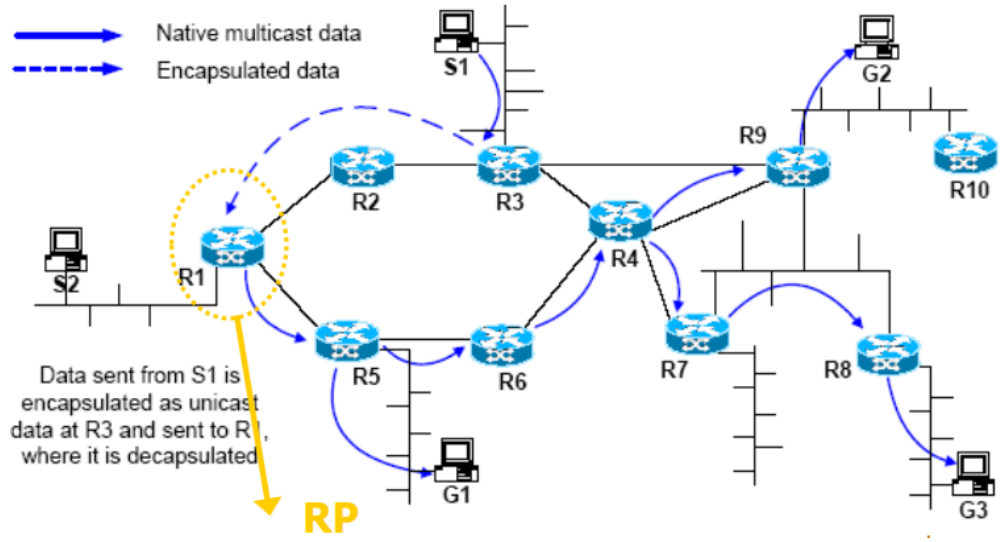
- ✓ Independência do protocolo unicast que opera na rede;
- ✓ Boa escalabilidade;
- ✓ Suporta SSM e ASM;
- ✓ Suporta ST (sem necessidade de reter o estado) e SBT (caminhos mais eficientes).

Desvantagens:

- × Em ST é necessário encapsulamento e/ou decapsulamento entre a fonte e o RP;
- × Necessário um mecanismo para encaminhar tráfego entre a fonte e o RP.

Fluxo PIM-SM





★ **PIM-DM**

a. É um protocolo *opt-out* e suporta *source based trees*;
b. Assume que maioria dos domínios na rede querem receber dados;
c. Má escalabilidade (usado para domínios pequenos);

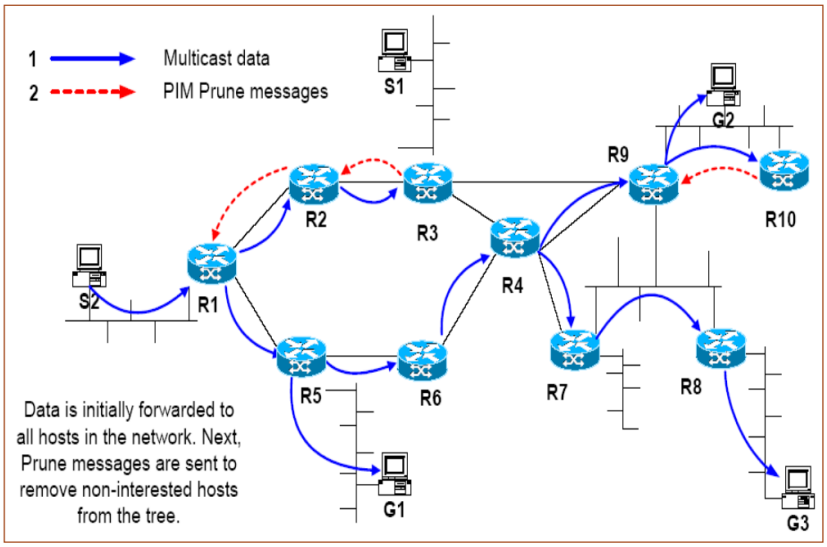
Vantagens:

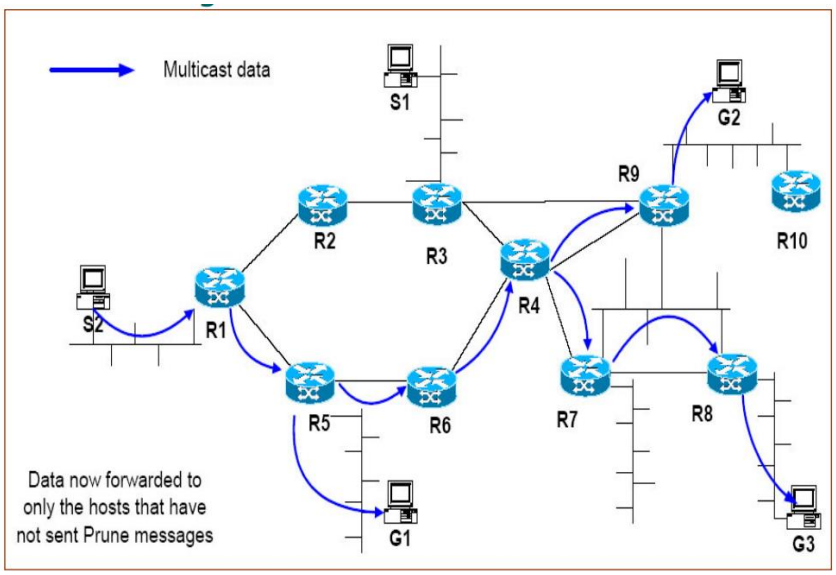
- ✓ É eficiente se o número de hosts que quer receber dados for elevado;
- ✓ Evita a complexidade da configuração de um RP;

Desvantagens:

- Todos os routers precisam de manter um estado para cada fonte;
- Não escala bem quando há poucos hosts que querem receber dados;

Fluxo PIM-DM





Nota.:

- O estado das mensagens *Prune* nos routers têm time-out, ou seja, é novamente enviada uma mensagem *Prune* (recusar dados) ou *Graft* (aceitar dados) após time-out;
- Caso um host se queira juntar ao grupo numa área podada, o router local envia uma mensagem Graft.

+ *Multimedia Networking*

Tipos de multimédia:

- **Áudio**: *Sinal Análogo*
  1. CD
  2. MP3
  3. *Internet telephony*

- **Vídeo**: *Sequência de imagens exibidas a um ritmo constante*

  Tipo de codificação

Utiliza redundância entre imagens sequencias para reduzir o número de bits na codificação da imagem em questão;

- ★ **Espacial** (mesma imagem) – Em vez de repetir a mesma mensagem *N* vezes, envia o "objeto" e o número de vezes que este se repete;
- ★ **Temporal** (entre imagens) – Envia apenas as frames que diferem para a imagem seguinte.

Bit rate

Quantidade de bits que são transmitidos por unidade de tempo;

- ★ **CBR** – Taxa constante;
- ★ **VBR** – Taxa varia mediante a variação do tipo de codificação.

Tipos de aplicações de multimédia:

- **Streaming**

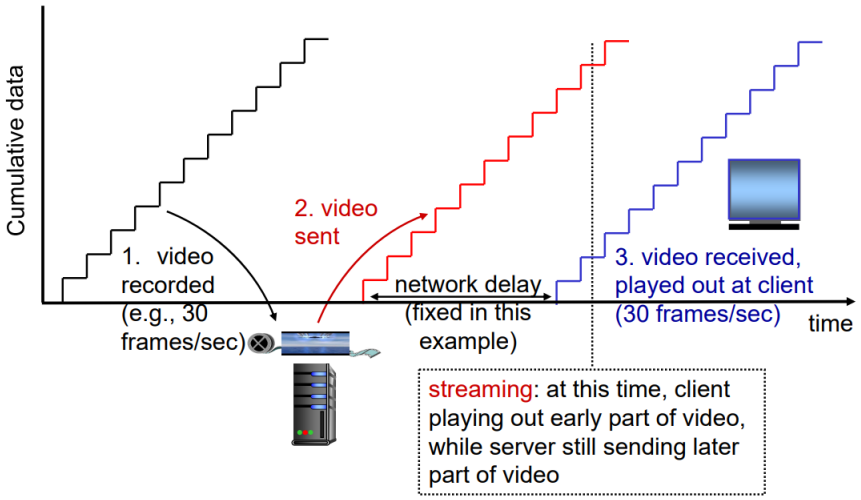
  Ex.: Youtube, Netflix, Hulu

  Tipos de Streaming

- ★ **Com** Armazenação – Pode transmitir antes de descarregar o conteúdo completo;
- ★ **Sem** Armazenação – O conteúdo pode estar armazenado no servidor permitindo uma transmissão mais rápida;

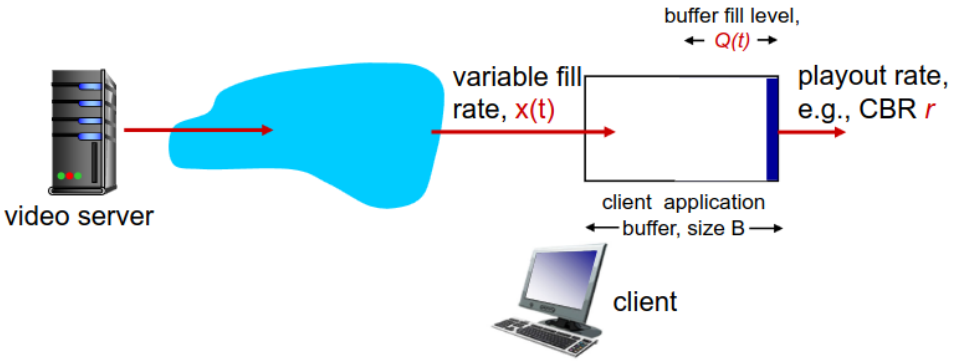
Desafios

1. *Restrição de playout contínuo* – Quando iniciada a reprodução pelo cliente, esta deve coincidir com o cronograma original, exigindo a utilização de um buffer no lado do cliente devido aos atrasos variáveis na rede.



2. *Interatividade do cliente* – Terá de ser permitido o cliente pausar, avançar e recuar no vídeo;
3. *Recuperação e retransmissão* – Pacotes perdidos terão de ser retransmitidos.

Buffering no Cliente



Após ser feito o preenchimento inicial do buffer do cliente, é inicializada a stream.

No decorrer da transmissão, o nível de preenchimento do buffer do cliente (Q(t)) varia mediante a variação da taxa de preenchimento (x(t)) e a taxa de reprodução do vídeo (r) permanece constante.

→ x < r – O buffer ficará vazio temporariamente, o que causa interrupções na stream até o voltar a ser preenchido;

→ x > r – O buffer não esvaziará, desde que haja um atraso inicial suficientemente grande para reduzir a chance de interrupções, mas também significa que o usuário espera mais para iniciar a reprodução.

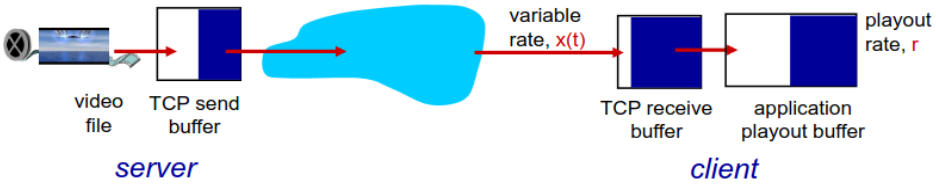
Tipos de Protocolo

★ **UDP**

a. O servidor envia dados a uma taxa adequada para o cliente, frequentemente com uma taxa constante, permitindo que a taxa de transmissão seja independente dos níveis de congestionamento na rede;
b. Apresenta um curto atraso de reprodução (1-2 segundos) para eliminar a variação na rede;
c. A recuperação de erros é tratada a nível aplicacional e requer protocolos como RTP/RTCP para gerir tipos de carga de multimídia e RTSP para controlar a sessão de streaming (pausar, retomar, reposicionar);
d. Difícil implementação em larga escala;
e. No entanto, este protocolo apresenta <u>três desvantagens</u>:
    i. Em caso de largura de banda insuficiente comparativamente à taxa de codificação é possível interrupções de stream, perda pacotes e má experiência do utilizador;
    ii. Necessidade de controlo adicional através de uma ligação RTSP separada para tratar das interações do lado do cliente;
    iii. Possibilidade de ser bloqueado por firewalls, o que não permite a receção do vídeo pelos utilizadores.

★ **HTTP**

a. A receção de ficheiros multimédia é feita através de pedidos HTTP GET (normalmente utilizados para obter páginas web) e são enviados ao máximo de velocidade possível através da ligação TCP.

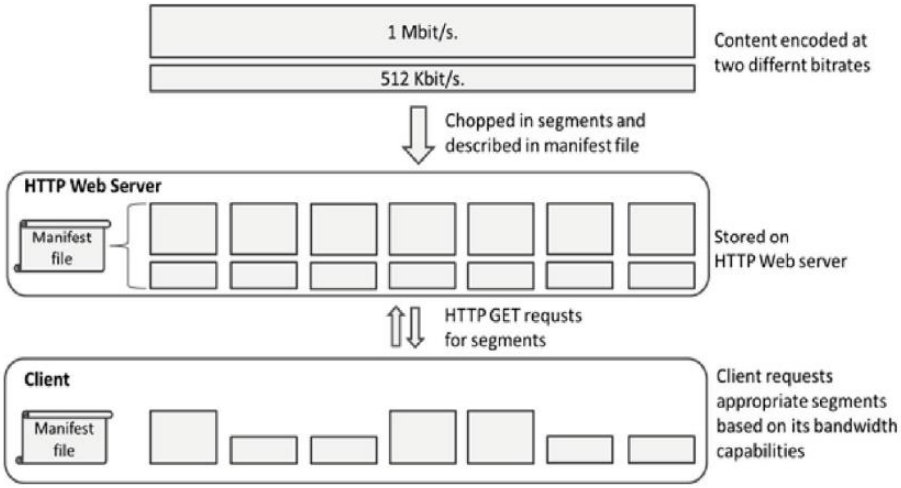


b. A velocidade de preenchimento (x(t)) varia devido ao controlo de congestão do TCP e às retransmissões de dados para garantir que os dados chegam ao destino na ordem correta;
   - → x > r – O buffer do cliente enche mais rápido, o que resulta numa produção mais suave e o servidor poderá ser pressionado para ajustar a taxa de envio;
   - → x < r – O buffer do cliente esgotará e a reprodução do vídeo poderá parar temporariamente.

   A taxa de envio do servidor não pode ser superior a r, o que limita a taxa de envio do conteúdo.

c. Não necessita de um servidor de controlo separado;
d. A reposição de vídeo é tratada através de novos pedidos HTTP GET com cabeçalhos de byte-range para especificar a nova posição. (...)
e. O HTTP/TCP é mais propenso a passar por firewalls;
f. Financeiramente vantajoso quando implementado em larga escala, comparativamente com UDP.

★ **DASH** over http



Para o servidor,

1. O vídeo é dividido em chunks;
2. Cada chunks possuí a sua taxa de codificação (qualidade do vídeo poderá então variar);
3. As taxas de codificação são armazenadas em diferentes ficheiros, que são replicados pelos diferentes nodos CDN;
4. *Manifest File* – Fornece URLs para os diferentes chunks;

Para o cliente,

1. Mede, periodicamente, a largura de banda da ligação servidor-cliente;
2. Consulta o manifesto para ter acesso aos chunks, um de cada vez. Além disso, escolhe a taxa máxima de codificação mediante a largura de banda disponível o que provoca variações na escolha da taxa com o passar do tempo;
3. É *inteligente*:
    a. O cliente determina o momento ideal para solicitar o próximo chunk de vídeo, garantindo que haja sempre dados suficientes no buffer para evitar interrupções na reprodução (starvation), mas sem acumular dados em excesso que possam causar transbordamento do buffer (overflow);
    b. O cliente escolhe a taxa de codificação (qualidade) do chunk a ser solicitado com base na largura de banda disponível. Se

houver mais largura de banda, o cliente pode solicitar chunks de maior qualidade (maior taxa de bits);

c. O cliente pode escolher de qual servidor CDN solicitar o chunk. A decisão pode ser baseada em fatores como proximidade geográfica ou largura de banda disponível no servidor;

- ***Conversação de Voz/Vídeo em Tempo Real***

  Ex.: Skype, Zoom, Viber, Whatsapp

  Objetivo: Manter a "conversa"

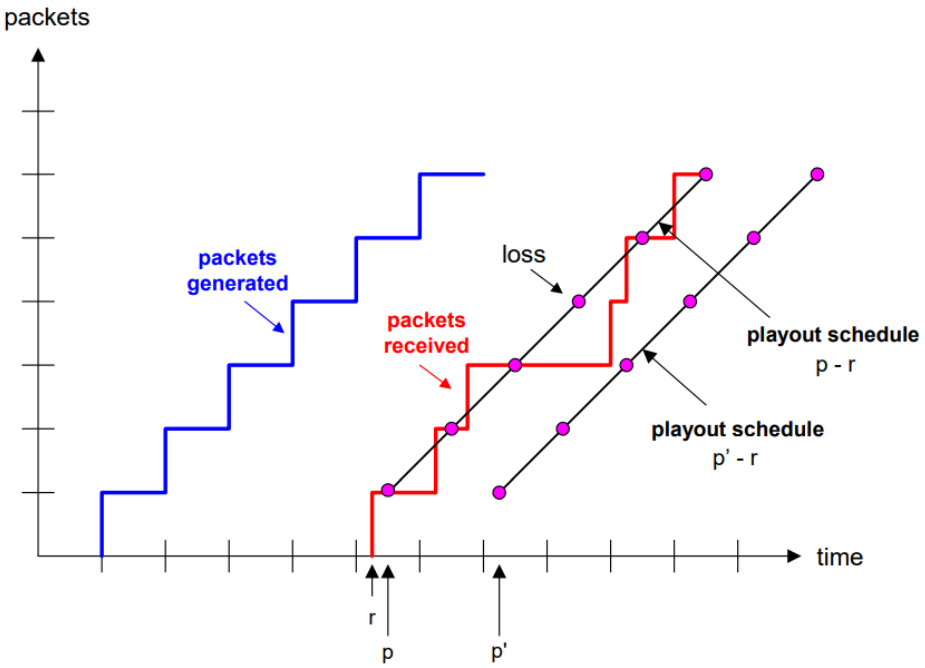
  - ✓ < 150ms
  - × > 300ms

  VoIP características

  - ★ Os chunks são apenas gerados durante os períodos de fala (talk spurts), a uma taxa de transmissão de 64kbps;
  - ★ A cada chunk é adicionado um cabeçalho para ser adicionado a um segmento UDP (ou TCP), sendo este enviado através de um socket a cada 20ms durante um período de fala.
  - ★ Apresenta perdas de pacotes ao nível da rede e atrasos de origens variadas, porém após 300ms cada pacote é dado como perdido. Também apresenta tolerância à perda entre 1% a 10%.

  Fluxo VoIP

  - ★ Emissor gera chunks a cada 20msec durante períodos de fala;
  - ★ Recetor tenta produzir cada chunk após q msec de ser gerado. Considerando que cada chunk tem um timestamp t, será produzido em t + q, porém caso o chunk ultrapasse esse intervalo é dado como perdido.
    - → + q provoca uma menor perda de pacotes;
    - → - q provoca uma melhor experiência interativa.



Ajuste do Atraso de Playout

Objetivo: Manter o atraso de playout baixo enquanto também se mantém baixa a taxa de perda de pacotes.

Abordagem:

1. Ajustar o playout de cada chunk com base no atraso da rede;
2. Durante os períodos de silêncio, o tempo de playout é comprimido ou alongado;
3. Os chunks continuam a ser gerados a cada 20msec.

a. Atraso do Pacote *i*

$$d_i = (1-\alpha)d_{i-1} + \alpha\,(r_i - t_i)$$

delay estimate after ith packet — *small constant, e.g. 0.1* — *time received - time sent (timestamp)* = *measured delay of ith packet*

b. Variação do Atraso do Pacote *i*

$$v_i = (1-\beta)v_{i-1} + \beta\,|r_i - t_i - d_i|$$

c. Playout do 1.º pacote gerado durante o 1.º período de fala

$$playout\text{-}time_i = t_i + d_i + Kv_i$$

Nota.: Como é que um recetor descobre o 1.º pacote de cada período de fala?

- Caso não haja perdas de pacotes, o recetor apenas verifica se o tempo dos sucessivos envios é <20msec, neste caso, começou um novo período de fala;
- Quando existe a possibilidade de perda de pacotes, para além do tempo de envio, o recetor verifica os números de sequência dos pacotes.

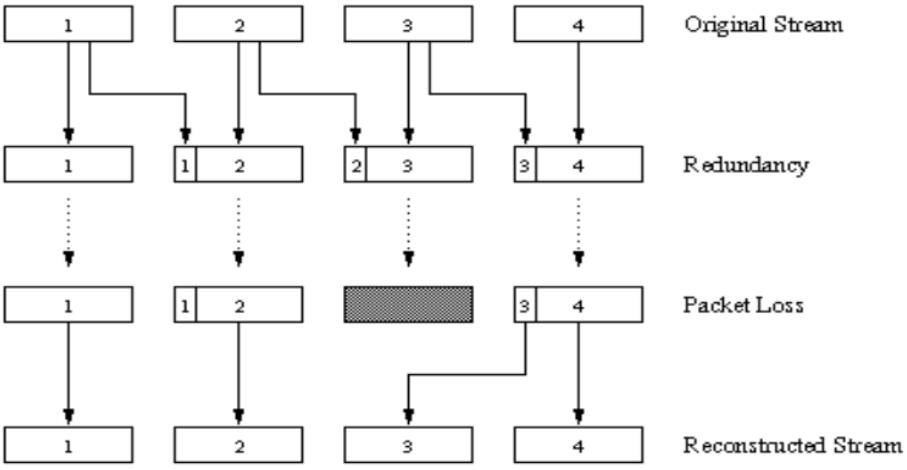
Recuperação depois da Perda de um Pacote

Objetivo: Recuperar o pacote considerando um atraso tolerável entre a transmissão original e o playout.

Abordagem:

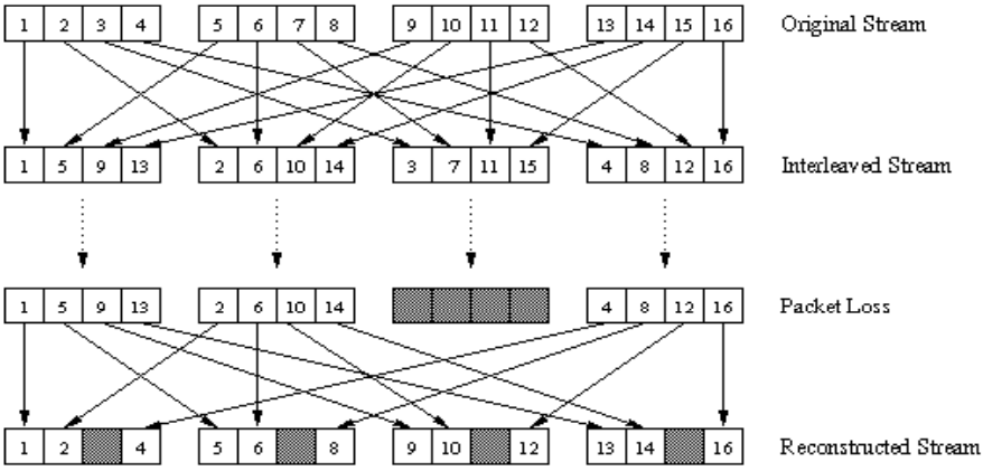
- **ACK/NAK** – Cada confirmação (ACK) ou não-confirmação (NAK) leva aproximadamente um Round-Trip Time (RTT) para ser processada. Devido ao atraso envolvido, esta não é sempre a opção mais viável para a recuperação rápida de pacotes perdidos.

- **FEC**
    a. *FEC básico* – Para cada grupo de *n* chunks, cria-se um chunk redundante através do XOR (exclusive OR) dos *n* chunks originais. Assim, enviam-se n + 1 segmentos, aumentando a largura de banda em um fator de 1 / n. Isso permite reconstruir os n segmentos originais se, no máximo, um chunk for perdido entre n + 1 chunks;
    b. *"piggyback lower quality stream"* – Neste caso, uma stream de áudio de resolução mais baixa é enviada como informação redundante, ajudando a ocultar perdas não consecutivas.

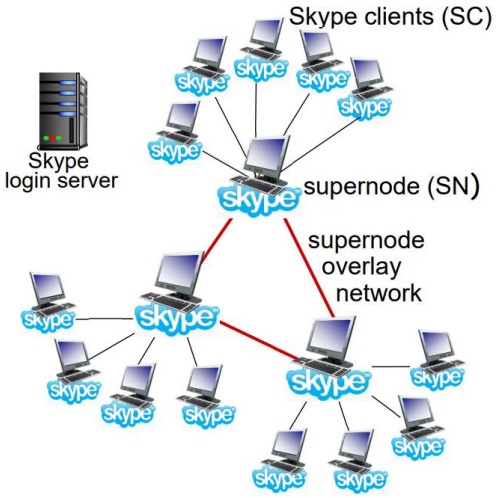


Ex.: A stream principal pode ser PCM a 64 kbps e a stream redundante GSM a 13 kbps.

    c. *"interleaving"* – Chunks de áudio de 20 ms são divididos em unidades menores de 5 ms, onde cada um contém várias dessas pequenas unidades de diferentes chunks. Assim, se um pacote for perdido, a maioria do chunk original é preservado, minimizando a perda de informação, mas pode levar a um aumento na latência e no atraso de playout.



Skype – ***P2P***



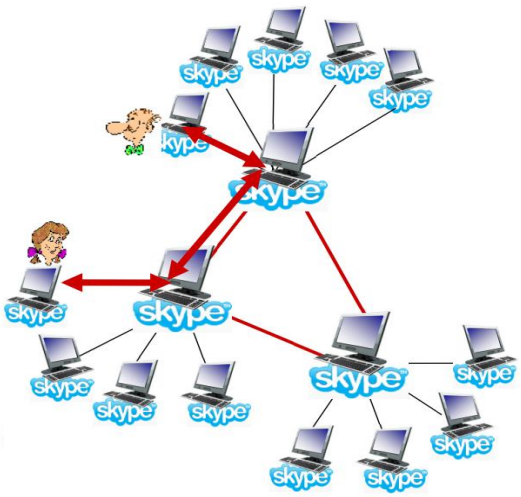
Nota.:

- SN é um cliente Skype com funções especiais;

Problema:

- O problema surge quando Alice e Bob estão atrás de "NATs" e querem comunicar entre si. Por sua vez, a NAT proibe que partes externas se conectem com uma parte interna, mas o inverso é possível.
- Para superar essa limitação, eles mantêm conexões abertas com os SNs mais próximos para estes serem usados como intermediários.

Ex.: Quando a Alice quer comunicar com Bob, ela sinaliza o seu SN para se conectar ao SN do Bob. Em seguida, os SNs da Alice e do Bob estabelecem uma conexão entre si, permitindo que a comunicação ocorra por meio da conexão inicial que Bob estabeleceu com seu próprio SN.



- ***Streaming Live***

  Ex.: Jogo de Futebol

  Tipos de Protocolos:

  - ★ **RTP** *com RTCP*
  - ★ **SIP**

+ *Signaling Protocols*

Ex.: SIP (Session Initiation Protocol)

O que é?

- ★ É um protocolo de controlo que opera na camada de aplicação para criar, modificar e terminar as sessões com um ou mais participantes;
- ★ É independente, mas pode ser usado com outros protocolos (RTP, RTCP, SDP) para construir uma arquitetura de multimédia completa;

Características

- Utiliza o formato URI que é semelhante ao endereço de um email;

sip|sips:user:password@host:port;uri-parameters?headers

sip:bob@domain.com

- Baseia-se modelo de transação de pedido/resposta semelhante ao HTTP, com campos de cabeçalho e códigos de status idênticos;
- Utiliza pesquisas DNS recursivas e iterativas para localizar participantes;
- Incorpora o protocolo SDP (Session Description Protocol) para descrever as características da sessão em questão;
- Normalmente, utiliza UDP para melhorias no desempenho, mas pode usar TCP ou TLS para uma ligação segura;

Funcionalidades

- ★ Localizar o destinatário;
- ★ Determinar a disponibilidade do destinatário;
- ★ Determinar as capacidades do destinatário;
- ★ Capaz de configurar uma sessão;
- ★ Capaz de gerir uma sessão;

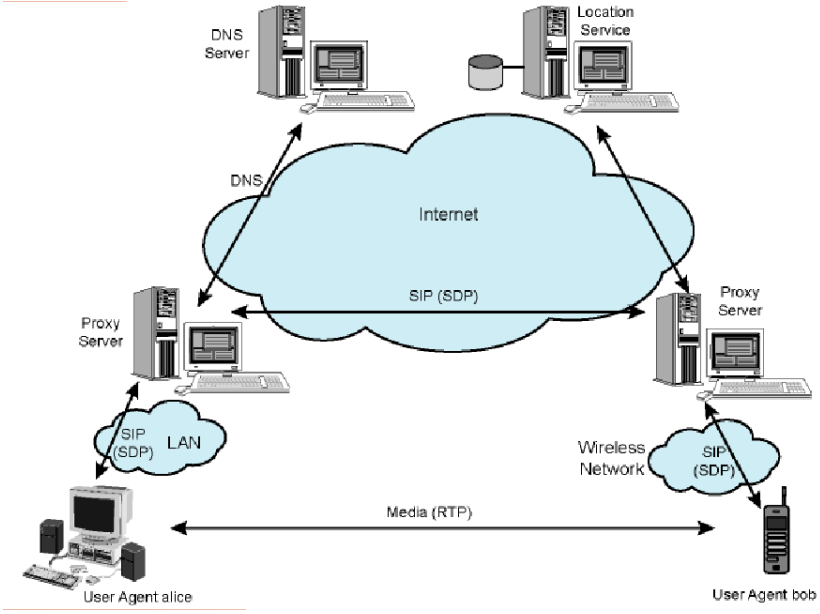
Componentes e Primitivas de Serviço

a. **User Agent** – Em cada terminal SIP, existe um *user agent client (UAC)* que emite solicitações e um *user agent server* (UAS) que recebe e responde às solicitações;
b. **Redirect Server** – Redireciona o cliente para um conjunto alternativo de URIs e utiliza procuras DNS iterativas;

c. **Proxy Server** – É tanto cliente como servidor. É responsável pelo *routing* entre clientes e utiliza procuras DNS recursivas;
d. **Registrar Server** – É um servidor que aceita pedidos *REGISTER* e coloca a informação contida nesses pedidos nas solicitações feitas no localization service;
e. **Localization Service** – Mantem uma base de dados de mapeamentos entre endereços SIP e endereços IP para ser possível fornecer informações sobre as possíveis localizações de um destinatário (callee) para redirect and proxy servers.

Nota.:

- Os *user agents* podem comunicar através de uma infraestrutura de hosts, designados por *proxy servers.*



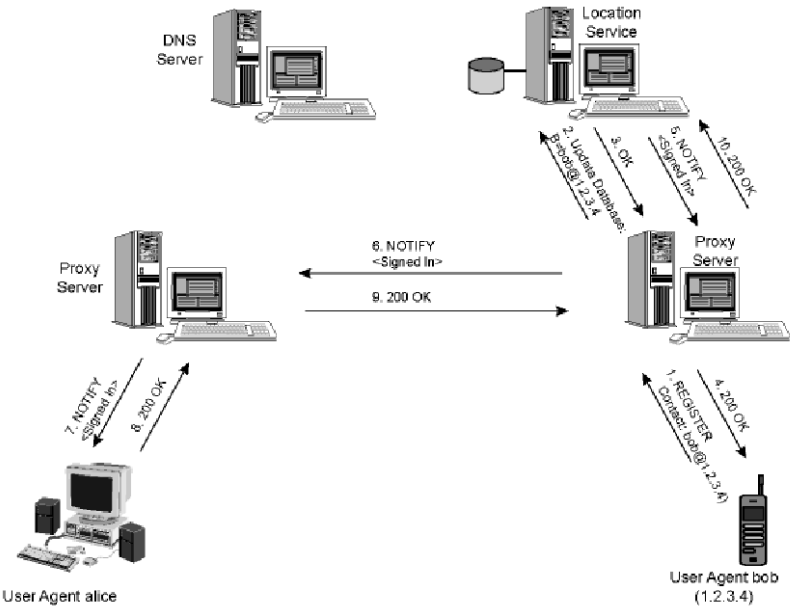
Tipos de solicitações:

- *REGISTER* – Notifica a rede SIP sobre os endereços IP e URIs para os quais deseja receber chamadas. É utilizado principalmente para permitir que um servidor SIP saiba onde encontrar um determinado dispositivo SIP na rede;
- *INVITE* – Estabelece uma sessão entre os *user agents*, permitindo a comunicação entre eles;
- *ACK*
- *CANCEL*
- *BYE* – Termina a sessão;
- *OPTIONS* – Solicita informação sobre as capacidades do destinatário;

- ★ *SUBSCRIBE* – Acompanha a disponibilidade de outros destinatários;
- ★ *NOTIFY* – Resposta ao SUBSCRIBE e é crucial para informar o *subscriber* sobre eventos relevantes, como mudanças na presença de um participante.

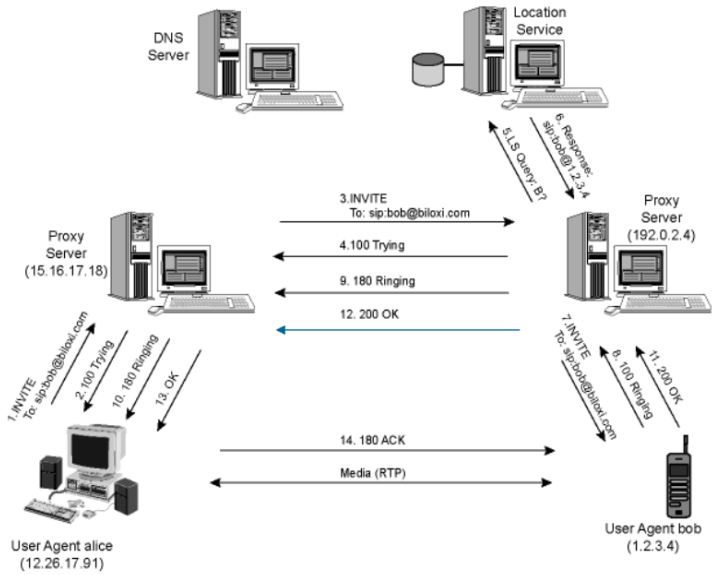
SIP – Notification Example

Escola de
Departamento de

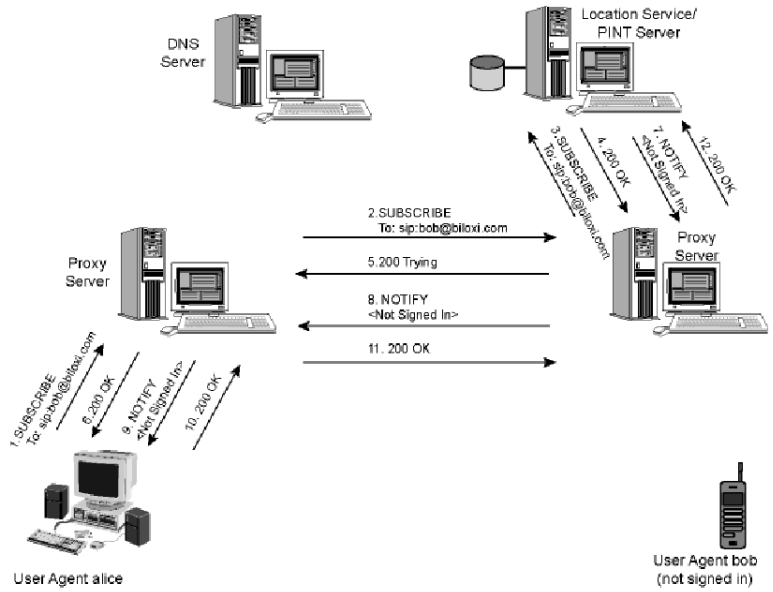


SIP – Successful Call Setup

Escola de
Departamento de I



SIP – Presence Example



SDP

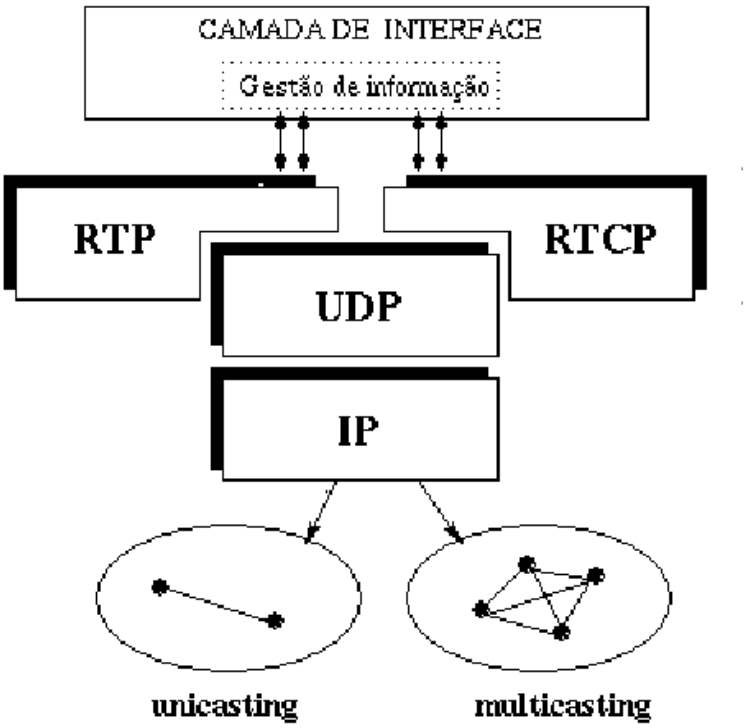
- O Protocolo de Descrição de Sessão (SDP) é frequentemente usado em conjunto com o Protocolo de Iniciação de Sessão (SIP) para configurar e descrever sessões de comunicação em redes IP. O SIP convida os participantes para uma sessão, e o corpo da mensagem SIP, codificado em SDP, contém informações sobre como os diferentes participantes usarão e codificarão os diferentes tipos de média, como voz e vídeo. Após o convite SIP e a troca de informações SDP, a transmissão de dados começa utilizando o protocolo de transporte apropriado, como o Protocolo em Tempo Real (RTP);
- Este protocolo contem muitas informações, nomeadamente:
    1. Streams – Uma sessão SDP pode incluir vários fluxos de média com conteúdos diferentes, como áudio, vídeo, dados, controle e aplicativos, onde cada um é especificado na descrição da sessão;
    2. Endereços – Endereços de destino para cada strea, que podem ser unicast (um para um) ou multicast (um para muitos), dependendo da forma como os dados são distribuídos;
    3. Portas – Para cada stream, são atribuídas portas específicas para a transmissão e receção dos dados;
    4. Tipos de Payload – Para cada tipo de stream (como áudio ou vídeo), são especificados os tipos de payload;
    5. Tempo inicial e final
    6. Originador – Em broadcast, é especificada a informação sobre o originador da sessão;

+ ***RTP***

Objetivo: Suportar o transporte de dados em tempo real, incluindo áudio e vídeo, através de mecanismos de entrega ponta a ponta.

Características RTP

- Usa um canal RTP para transporte de dados, designado por data channel, e um canal RTCP para reportar o estado do canal RTP, designado por control channel. Além disso, é encapsulado em UDP e suporta sessões unicast e multicast;



- Sessão RTP
  a. Sessão = RTP + RTCP
  b. Endereço da Sessão = Endereço da Rede (unicast, multicast) + Par de portas
  c. Canal RTCP = Porta RTP + 1

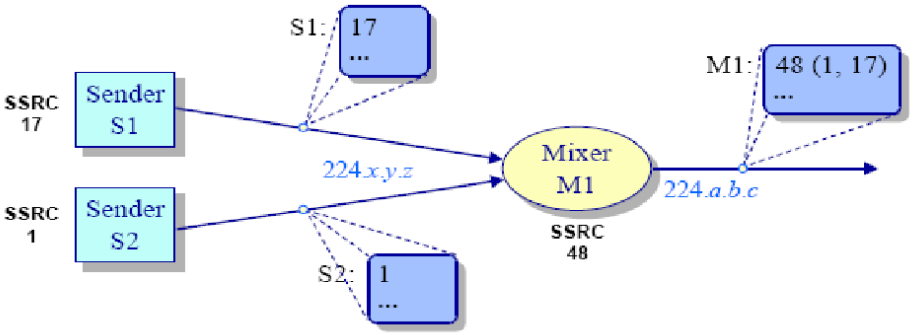
- Cabeçalho RTP

payload type	sequence number	time stamp	Synchronization Source ID	Miscellaneous fields

a. *Tipo de Payload* (7 bits) – Indica o tipo de codificação a ser utilizado;
b. *Número de Sequência* (16 bits) – É incrementado +1 por cada pacote RTP **enviado** e é responsável por detetar perda de pacotes;
c. *Time Stamp* (32 bits) – Representa o instante de envio do primeiro byte do pacote em questão. Além disso, pode ser usado para colocar os pacotes enviados por ordem ou calcular a variância da rede;
d. *SSRC* (32 bits) – Identifica a fonte de cada stream RTP.

Nota.: SSRCs vs. CSRCs

- Poderá existir um sistema intermediário que receba e combina diferentes PDUs de uma ou mais sessões RTP e devolva um novo PDU. Então Synchronization Sources(SSRC) transformasse em Contributing Sources (CSRC);



Vantagens:

- ✓ Permite restaurar a temporização precisa dos dados (timestamp);
- ✓ Deteta perdas de pacotes (número de sequência);
- ✓ Identifica conteúdo

Desvantagens:

- × Não garante reserva da largura de banda;
- × Não garante qualidade de serviço (QoS);
- × Não garante entrega de dados;

Nota.:

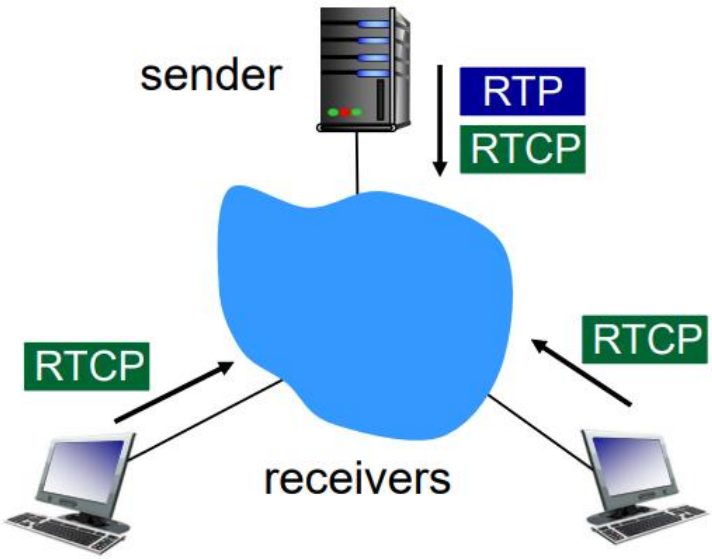
- A recuperação dos dados perdidos deve ser feita ao nível da aplicação e não ao nível do transporte;

*Protocolo **RTCP***

- Trabalha em conjunto com o RTP e cada participante da ligação RTP envia pacotes de controlo RTCP. Cada um destes pacotes contém os relatórios de envio e recebimento, informando sobre o número de pacotes enviado, número de pacotes perdido e, por fim, a variação no intervalo entre a chegada de pacotes. Com base nestes relatórios, o emissor pode ajustar a sua transmissão.

  Ex.: Se muitos pacotes estão perdidos ou a variação da rede está alta, o remetente pode ajustar sua taxa de envio para melhorar a qualidade da transmissão.

- Em cada sessão RTP, é comum o uso de um único endereço multicast para a transmissão de dados, abrangendo tanto os pacotes RTP quanto os pacotes RTCP relacionados, sendo utilizadas portas distintas para identificar cada um. Conforme o número de participantes em uma sessão aumenta, o número de pacotes RTCP também aumenta, visto que são diretamente proporcionais, o que obriga aos participantes ajustarem a quantidade de pacotes RTCP gerados, evitando congestionamento de rede. Desta forma, facilmente se conclui que este protocolo tem problemas de escabilidade de largura de banda, tentando manter a sua utilização a 5% por sessão.



- Tipos de pacotes
  a. **Sender Reports** (SR) – Contêm informações como o identificador SSRC da sessão RTP, o horário atual, o número de pacotes enviados e o número de octetos enviados. Eles permitem que o recetor estime a taxa média de dados, o tamanho médio de pacotes e distinguir entre pausas na transmissão e problemas na rede;

b. **Receiver Reports** (RR) – Contêm informações sobre a fração de pacotes perdidos, o número de sequência do último pacote recebido e a média da variação no intervalo entre a chegada de pacotes.
c. **Source Description** (SD) – Contêm informações sobre o remetente, como o endereço de e-mail, o nome do remetente e o identificador SSRC associado ao fluxo RTP. Eles são usados para mapear o identificador SSRC aos nomes de usuários ou hosts, facilitando a identificação dos participantes em uma sessão;

Nota.:

- Os RRs e SRs indicam estatísticas de receção de cada recetor e para cada remetente numa sessão RTP, incluindo:
    1. A taxa de perda de pacotes desde o envio do último SR ou RR;
    2. O número total de pacotes perdidos desde o início da sessão;
    3. O número de sequência mais alto recebido até o momento, o que permite estimar a quantidade de dados em trânsito (quando usado em conjunto com os carimbos de tempo dos SRs e RRs);
    4. A média da variação da rede no intervalo entre a chegada de pacotes de dados, fornecendo ao remetente uma visão indireta do tamanho de qualquer buffer adaptativo de reprodução usado no recetor. Isso pode ser útil em sessões de Voz sobre IP (VoIP) para ajustar a reprodução de áudio de forma adequada.