


Resumo 3

↗ Matéria	 <u>Redes de Computadores</u>
Σ Progresso	0
☰ Projeto	Resumo
⚙ Status	Em andamento

Camada Transporte

| Principais da Internet: UDP e TCP

Acima da Camada de Rede

| A Rede é a rota das mensagens. A camada Transporte cuida da comunicação lógica entre os processos.

Mensagens enviadas pelo Transporte são chamadas de Segmentação. O receptor faz a remontagem dos Segmentos.

▼ Funções da camada Transporte

▼ Endereçamento: do emissor e do receptor

É utilizado o mesmo endereço da camada de rede

▼ Estabelecer e terminar conexões

Permite o envio confiável e em ordem. Coloca o usuário e o servidor no mesmo canal.

A finalização de conexões confirma o recebimento e processamento da mensagem, e libera para finalizar a conexão.

Finalização de Conexão

- Pedir o fim da conexão
- Resposta ao pedido
- Confirmação (ACK)

| 3 Mensagens para finalizar a conexão de forma correta

O timer garante a finalização das conexões quando há problema no meio (não tem o ACK). O timer que sabe o momento do Timeout.

▼ Controle de fluxo: segurar a origem para não gerar um fluxo muito grande

Utiliza de buffering também, além do controle da quantidade de dados enviados por unidade de tempo.

▼ Envio e recepção confiável e em ordem: os segmentos podem chegar bagunçados e fora de ordem

Utiliza de números de sequência. Evita envio de números repetidos em caso de erros.

▼ Buffering: armazenamento.

Quando o receptor não conseguir processar tudo, ele armazena novos pacotes que chegam (com um limite de espaço, se alcançado, novos pacotes são descartados)

▼ Controle de Congestionamento

Ajuste da taxa de acordo com o gargalo do meio

▼ Multiplexação: identifica os processos em um computador

Evita que um programa receba um dado que não é direcionado à ele

- Recuperação de erros

▼ Transmissão confiável e em ordem

Primeiro Protocolo

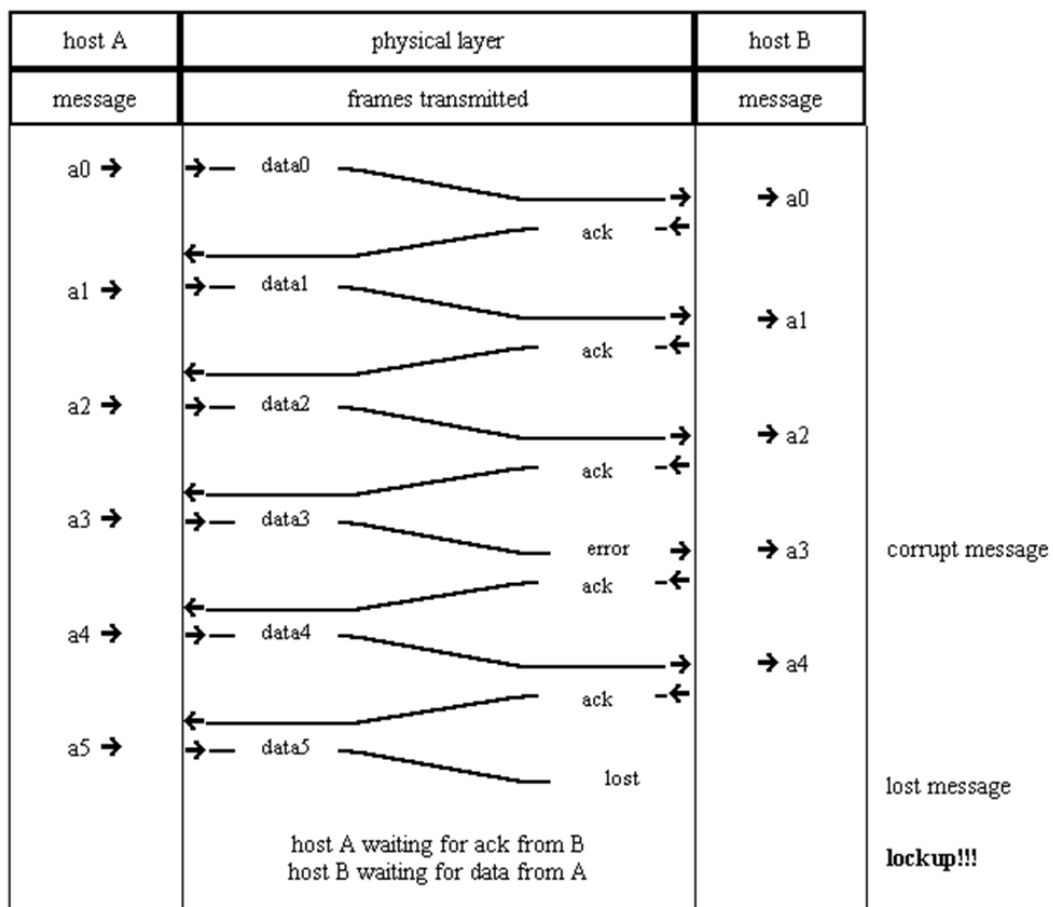
- Comunicação Simplex
- Sem erros no canal (sem problema de congestionamento)
- Pacotes recebidos e analisados instantaneamente (sem problemas de fluxo)
- Mundo ideal

Protocolo com tratamento de tempo de processamento

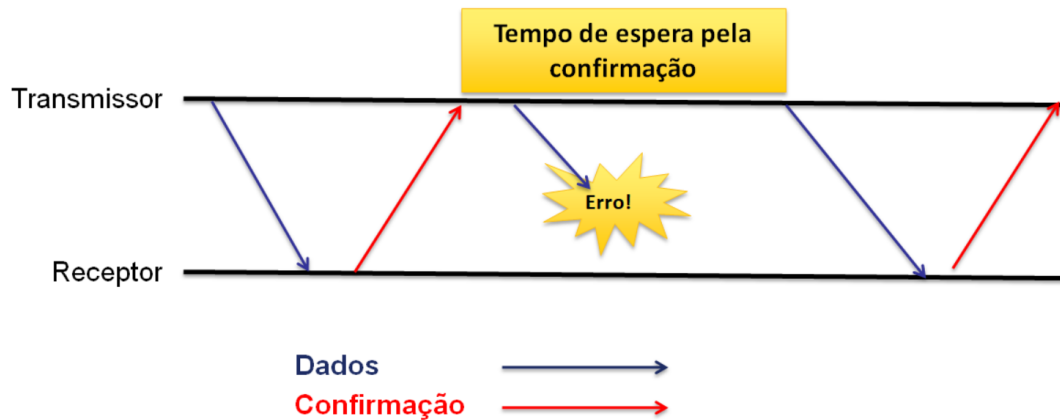
▼ Stop and Wait

Mensagem de indicação de recebimento.

- Só enviar mais dados quando o receptor terminar o processamento
- O receptor precisa notificar que terminou o processamento
- Um timer ajuda no problema do meio (caso exista problema no meio, o emissor envia a mensagem novamente após o timeout)



Parte negativa do Stop and Wait: Sem um temporizador, quando existe um erro no canal, o transmissor fica bloqueado. Entretanto, o temporizador de confirmação impõe um limite de tempo na rede (Ficar esperando uma resposta, o meio é mal utilizado, fica grande parte do tempo ocioso aguardando confirmação).



O temporizador também tem um limite de alcance da rede, o cálculo do timeout depende do tempo de propagação do sinal.

$$Tempo = T_{ida} + T_{volta} + T_{processamento}$$

$$T_{ida} + T_{volta} = round\ trip\ time\ (RTT)$$

Definição de um alcance máximo:

$$t = \frac{(2d)}{v}$$

▼ Quadros em Rajada

Mantêm o meio melhor utilizado, evita a quantidade de tempo que nada é enviado

Para o envio de várias informações, é preciso controlar a ordem de envio. É preciso adicionar sequência. Em comparação, no Stop and Wait, não necessariamente precisa do número de sequência.

Aqui é necessário buffers maiores

▼ Janela Deslizante: Buffer em sequência

O tamanho da janela é muito importante. Se o tamanho for 1, é um Stop and Wait. Se for infinito, é uma memória muito grande.

Tamanho ideal:

$$W = (RTT + \xi) \cdot \frac{(Largura\ de\ Banda)}{Tamanho}$$

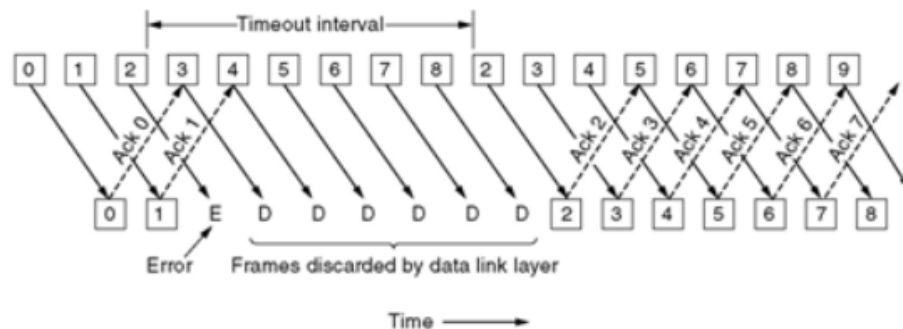
$$\frac{(Largura\ de\ Banda)}{Tamanho} = \text{Quantos pacotes chegam por unidade de tempo}$$

$$\xi = \text{Tempo de erro do meio}$$

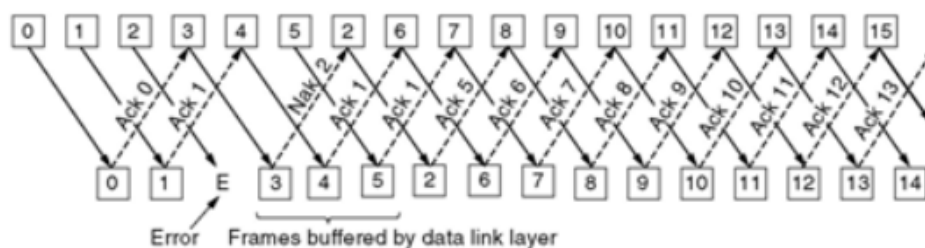
▼ Problema de retransmissão

Quando na sequência existe um erro no envio de um segmento, existe duas formas de arrumar.

- Volta N
 - Volta a partir do pacote que não foi recebido e reenvia toda a sequência depois de novo.



- Janela pode ter tamanho menor. O segmentos depois do erro são descartados.
- Retransmissão seletiva
 - Envia individualmente o pacote que teve o erro.



- Janela precisa ser maior para armazenar a sequência entre o erro e o reenvio

▼ Camada de Transporte da Internet

TCP

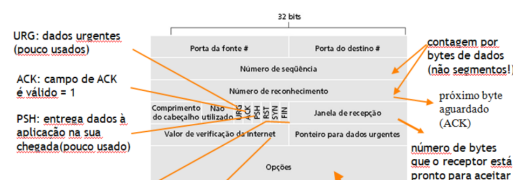
Confiável, garante a ordem de entrega

Utilizado para aplicações que requerem garantia de entrega de todo o arquivo. Pode demorar mais.

Possui:

- Controle de Congestionamento
- Garantia de recepção
- Controle de Fluxo
- Orientado à Conexão
- Full-duplex

20 Bytes de cabeçalho (



UDP

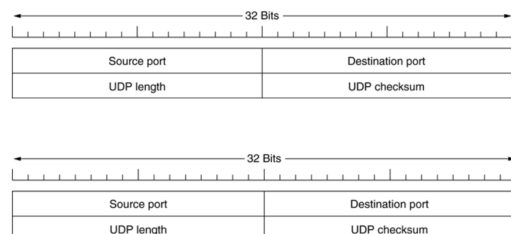
Não confiável, sem ordem de entrega

Não possui nenhum dos controles da TCP (fluxo, entrega, ordenação, conexão). Serviços rápidos que algumas perdas são toleradas. Segmentos UDP são tratados independente do outro.

Vantagens:

- Não tem atrasos iniciais de conexão
- Não é necessário ordenamento
- Transmissão à uma taxa constante

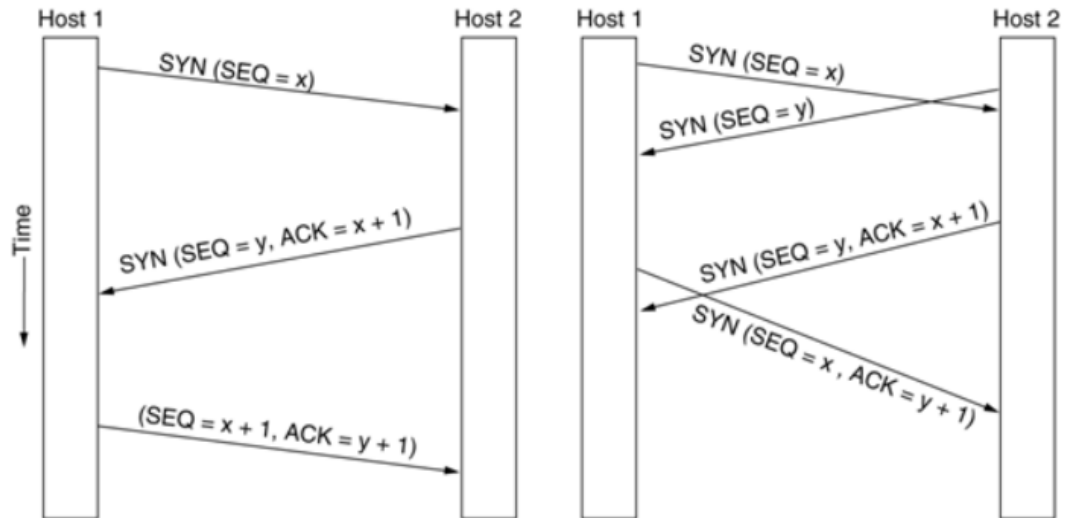
8 Bytes de cabeçalho



▼ Protocolo TCP

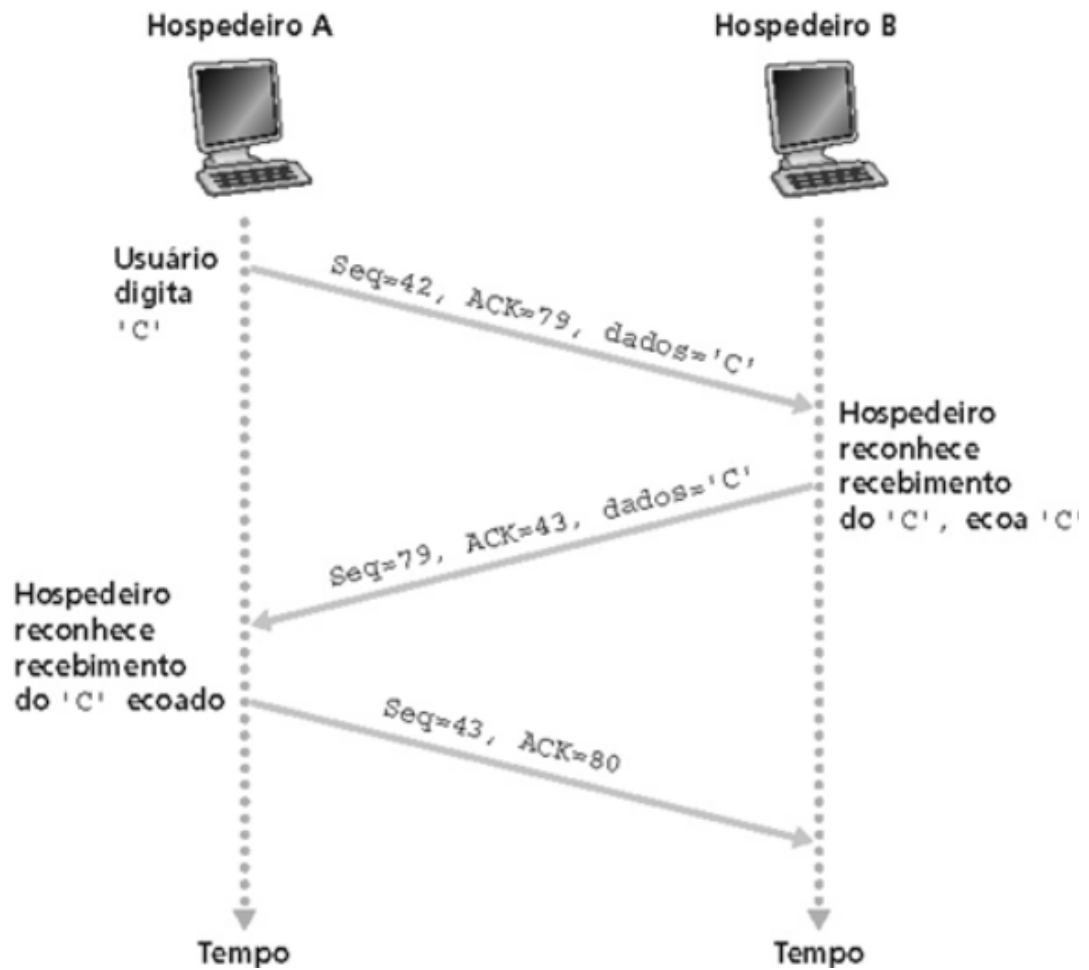
Abertura de Conexão

Utiliza 3 mensagens para a abertura da conexão. O pedido pode vir de um Host ou de dois simultaneamente, mas independente, abre apenas uma conexão.

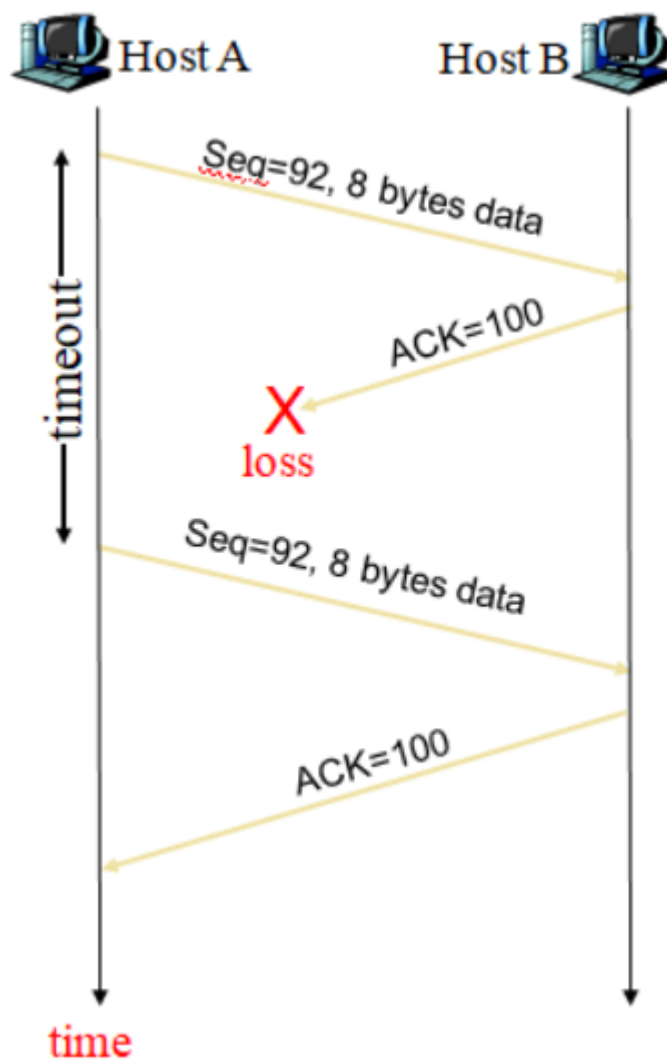


Confirmação de Recepção

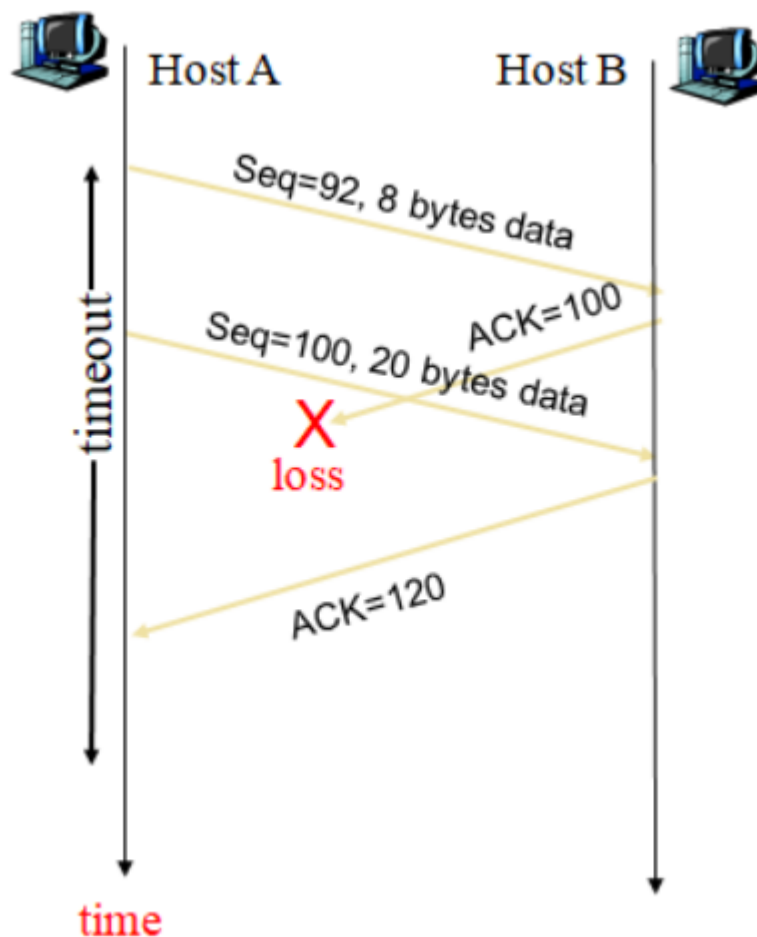
Possui o número de sequência que está sendo enviado e o número do próximo número que é esperado como resposta



A confirmação do próximo envolve a quantidade de dados que está sendo enviada, e é acumulativa. Quando existe um erro no meio do caminho, é possível uma confirmação conjunta.



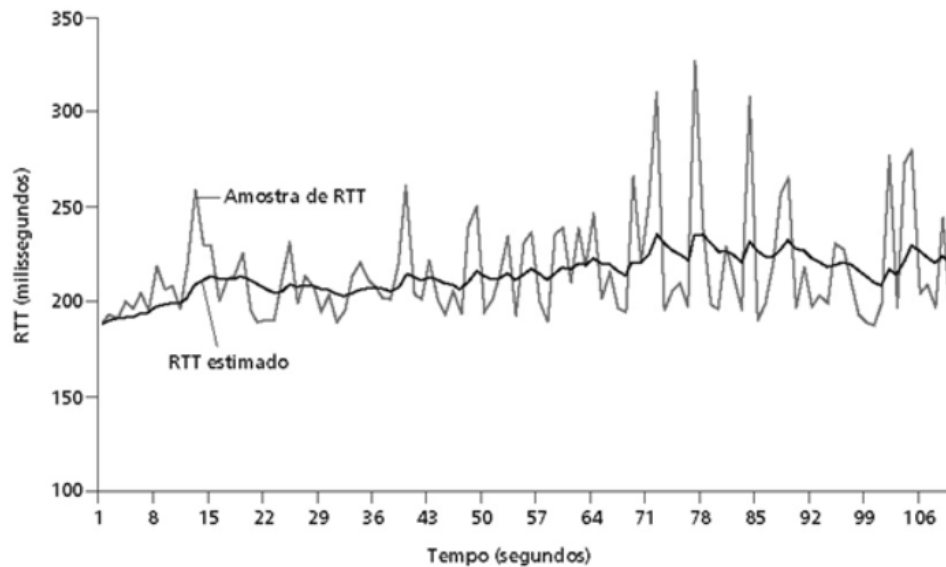
O transmissor aguarda a confirmação antes de enviar o próximo item da sequência (no exemplo, o último item enviado é o 99, e o próximo esperado é 100). Quando não recebe após o timeout, ele reenvia a informação.



O ACK 120 confirma a confirmação da sequência até o 99 e até o 119, e agora o próximo bloco esperado é o 120 (não é preciso reenviar o ACK 100).

O tempo de timeout é importante para não acontecer reenvio desnecessários. Ele precisa ser maior que o RTT para evitar a retransmissão desnecessária, mas não pode ser muito longo porque isso torna a reação à perda de segmento lenta, e desperdiça o meio.

Para estimar o RTT, é utilizado um controlador PD.



Controle de Fluxo

Receptor informa a área de janela disponível no momento que envia o ACK da mensagem. Esta informação vai no cabeçalho da mensagem, e o transmissor controla a quantidade de segmentos que é enviada. Assim, é evitado o overflow no buffer do receptor.

Tamanho livre da janela:

$$RcvWindow = TamBuffer - TamEmUso$$

$$TamEmUso = \acute{U}ltimoByteRecebido - \acute{U}ltimoByteLido$$

Controle de Congestionamento

O congestionamento ocorre quando no meio, existe uma parte do trajeto que possui um limite de transmissão. Para solucionar o problema, é feito um cálculo de quantos segmentos podem ser enviados juntos pelo meio (CongWin), e a taxa de envio é calculada a partir disso.

O CongWin é calculado de forma dinâmica e é medido em segmentos. Quando o transmissor identifica um congestionamento (Por meio de timeout ou 3 ACK's duplicados) o transmissor diminui a taxa, diminui o CongWin.

O controle de congestionamento é um evento de apostas. O transmissor começa com uma partida lenta, identificando uma banda máxima (quantos segmentos de uma vez é possível de enviar). Depois do primeiro envio, a taxa aumenta exponencialmente até a ocorrência do primeiro evento de perda.

Depois, a taxa cai de volta para o mínimo, e cresce exponencialmente até a metade da banda máxima identificada. Após a marca, cresce linearmente até um evento de perda.

