Redes de Computadores

Camada de Transporte

Agenda

Introdução

Princípios de Transferência Confiável de Dados

1

Introdução

Serviços, Elementos, Relação entre as camadas, Multiplexação e Demultiplexação

Camada de Transporte

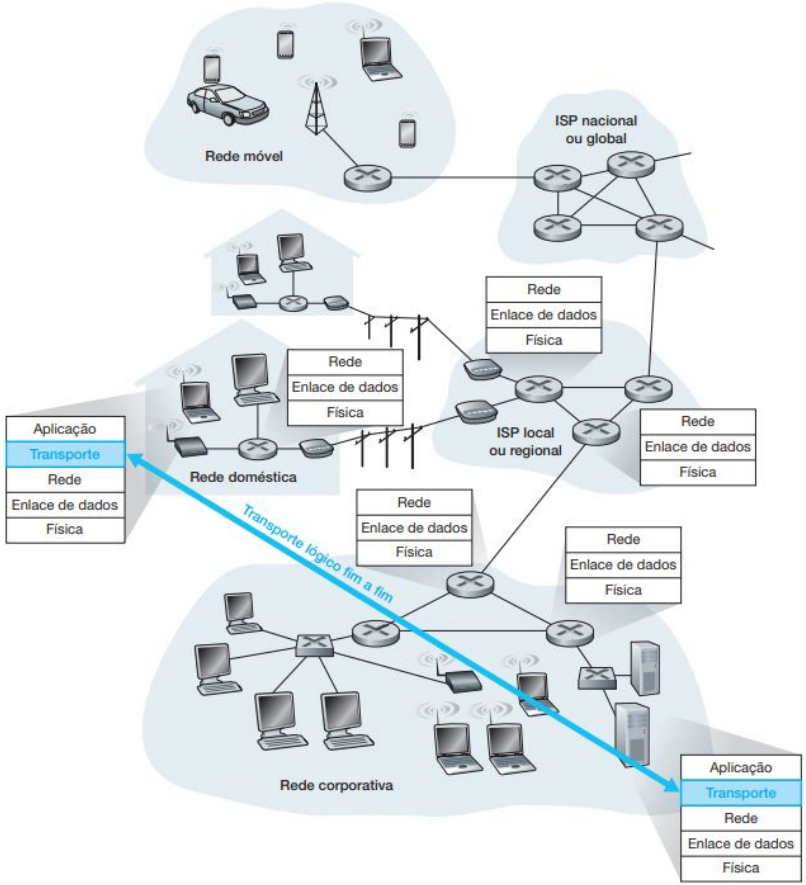
- Desempenha o papel fundamental de fornecer serviços de **comunicação lógica entre processos** de aplicação que rodam em hospedeiros diferentes
  - Como se os hospedeiros que rodam os processos estivessem conectados diretamente
  - Livres da preocupação dos detalhes da infraestrutura física

Camada de Transporte

- É responsável por dividir os dados enviados da camada de aplicação em mensagens que serão transmitidas pela rede
- Depois de receber os dados da camada de aplicação, essa camada adiciona informações para o seu controle, como a porta de origem, a porta de destino, entre outras informações.
    - Diferenciam qual aplicação fez o requerimento

Camada de Transporte

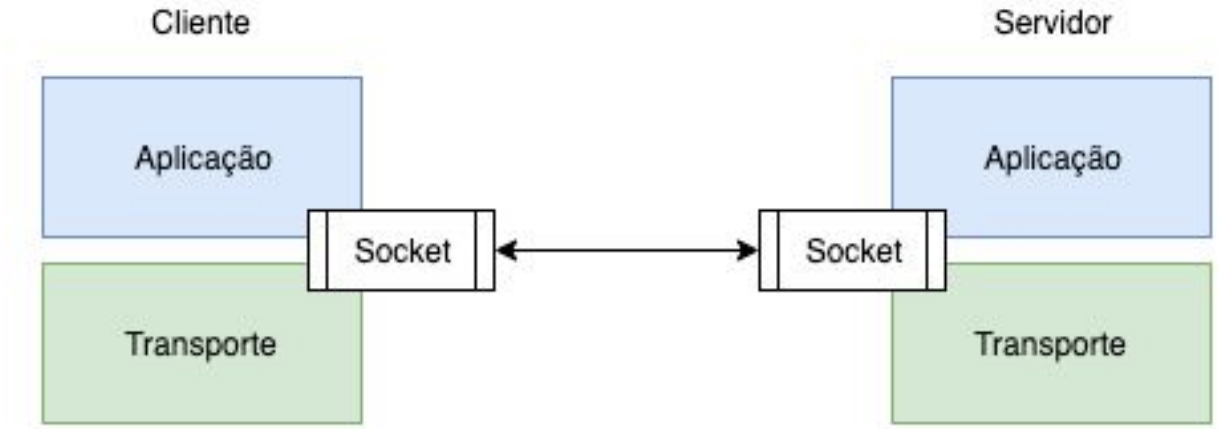
- Os protocolos de transporte são executados nos sistemas finais
- Lado emissor: quebra as mensagens da aplicação em segmentos e envia para a camada de rede
- Lado receptor: remonta os segmentos em mensagens e passa para a camada de aplicação

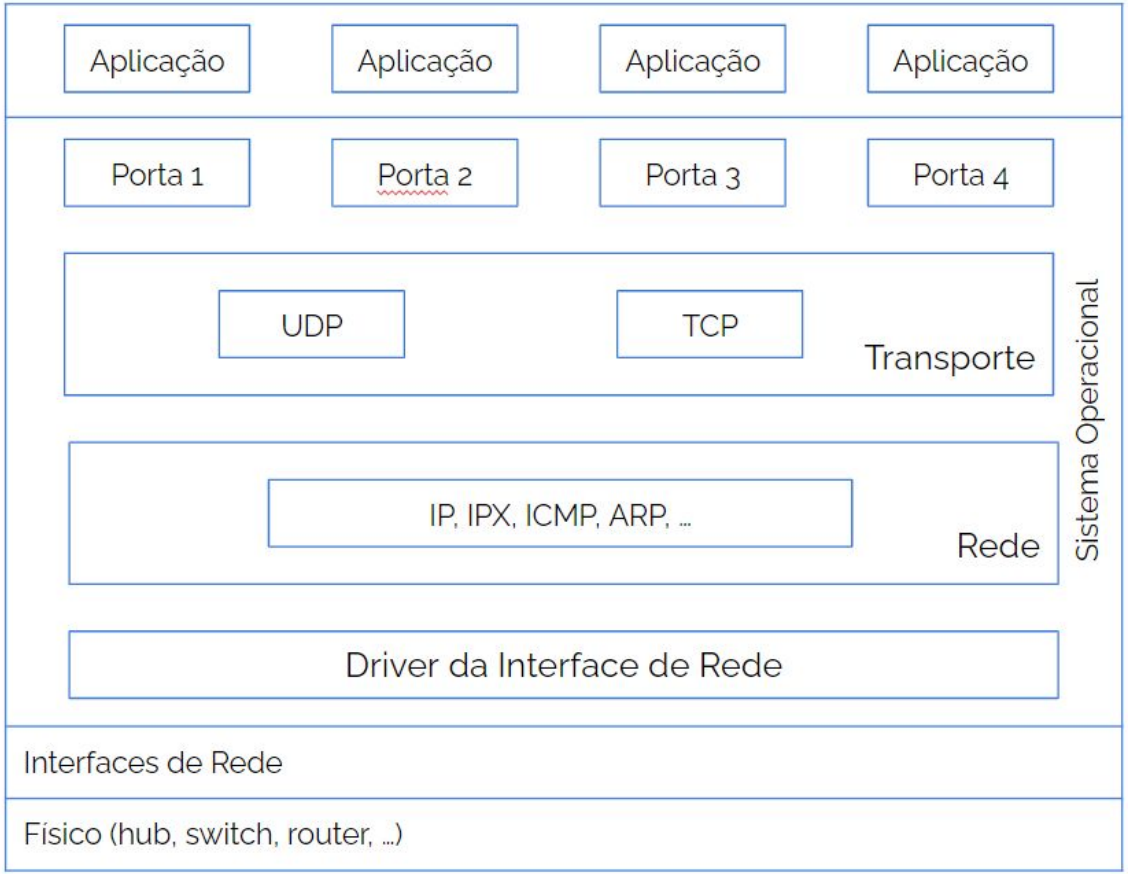

Rede móvel
ISP nacional
ou global
Rede
Enlace de dados
Física
Rede doméstica
ISP local
ou regional
Aplicação
Transporte
Rede
Enlace de dados
Física
Transporte lógico fim a fim
Rede corporativa

Relação entre as camadas

- Informações
  - Mensagem - Camada de Aplicação
  - Segmentos - Camada de Transporte
- Comunicação
  - Socket provê a comunicação entre fonte e destino
  - Na rede, a representação de um socket se dá por ip:porta, por exemplo: 127.0.0.1:4477

Relação entre as camadas




Aplicação
Aplicação
Aplicação
Aplicação
Porta 1
Porta 2
Porta 3
Porta 4
UDP
TCP
Transporte
IP, IPX, ICMP, ARP, ...
Rede
Driver da Interface de Rede
Sistema Operacional
Interfaces de Rede
Físico (hub, switch, router, ...)

Relação entre as camadas

- Funcionamento semelhante a camada de Rede
  - Transporte - Hospedeiro
  - Rede - Roteadores
- Camadas superiores são usuários do Transporte
- Camadas inferiores são provedores de Transporte

Aplicação	Protocolo (Camada de Aplicação)	Protocolo (Camada de Transporte)
Correio eletrônico	SMTP	TCP
Acesso a terminal remoto	Telnet	TCP
Web	HTTP	TCP
Transferência de Arquivo	FTP	TCP
Servidor de Arquivo Remoto	NFS	Tipicamente UDP
Recepção de Multimídia	Tipicamente Proprietário	UDP ou TCP
Telefonia por Internet	Tipicamente Proprietário	UDP ou TCP
Gerenciamento de Rede	SNMP	Tipicamente UDP
Protocolo de Roteamento	RIP	Tipicamente UDP
Tradução de Nomes	DNS	Tipicamente UDP

Relação entre as camadas

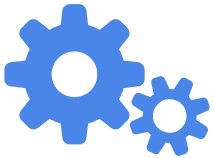
- O que acontece se a camada de rede oferecer um serviço inadequado?
- E se pacotes forem perdidos com frequência?
- O que acontece se os roteadores acabarem travando de vez em quando?

Serviços de Transporte

- Transmissão de dados **eficiente**, **confiável** e **econômica**
- Uso de software e/ou hardware
  - Kernel do SO
  - Biblioteca de Aplicativos de Rede
  - Processo de Usuário
  - Interface de Rede

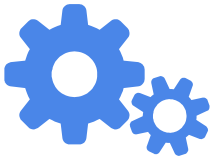
Serviços de Transporte

- Multiplexação e Demultiplexação
- Controle de Fluxo
- Correção de Erros
- Controle de Sequência



Multiplexação

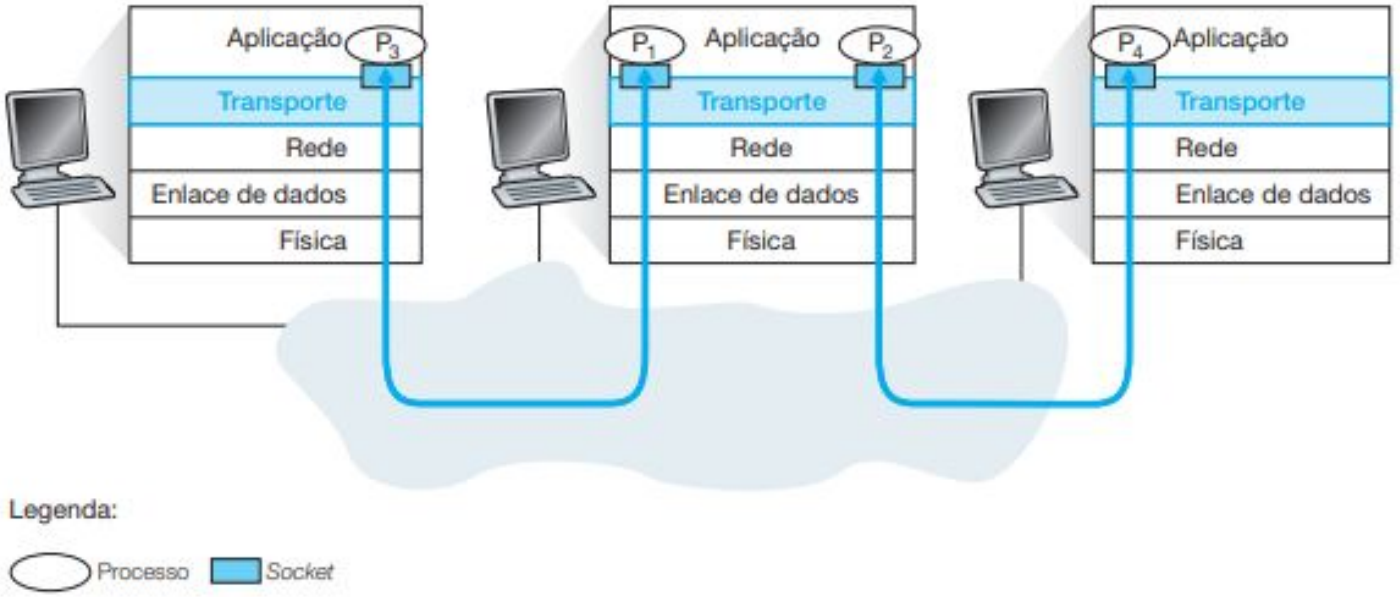
- Processo para reunir dados provenientes de diferentes sockets (diferenciados pelo número de portas)
- Encapsula cada parte dos dados (payload) com informações de cabeçalho para criar segmentos
- Passa os segmentos para a camada de rede
  - Vários-para-um



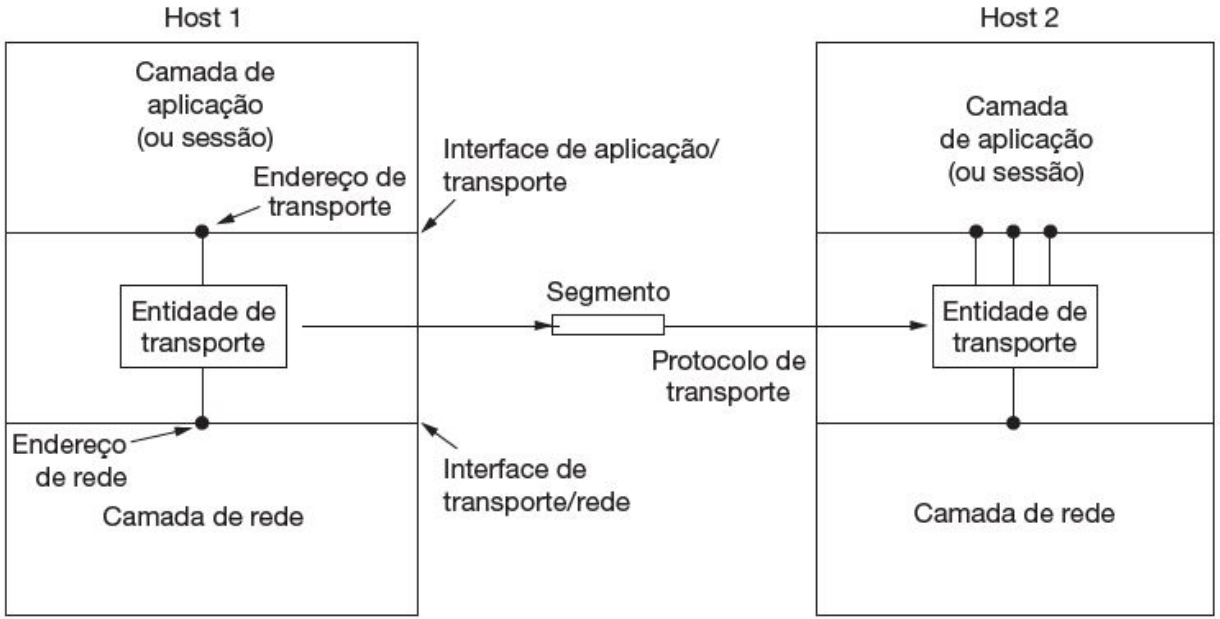
Demultiplexação

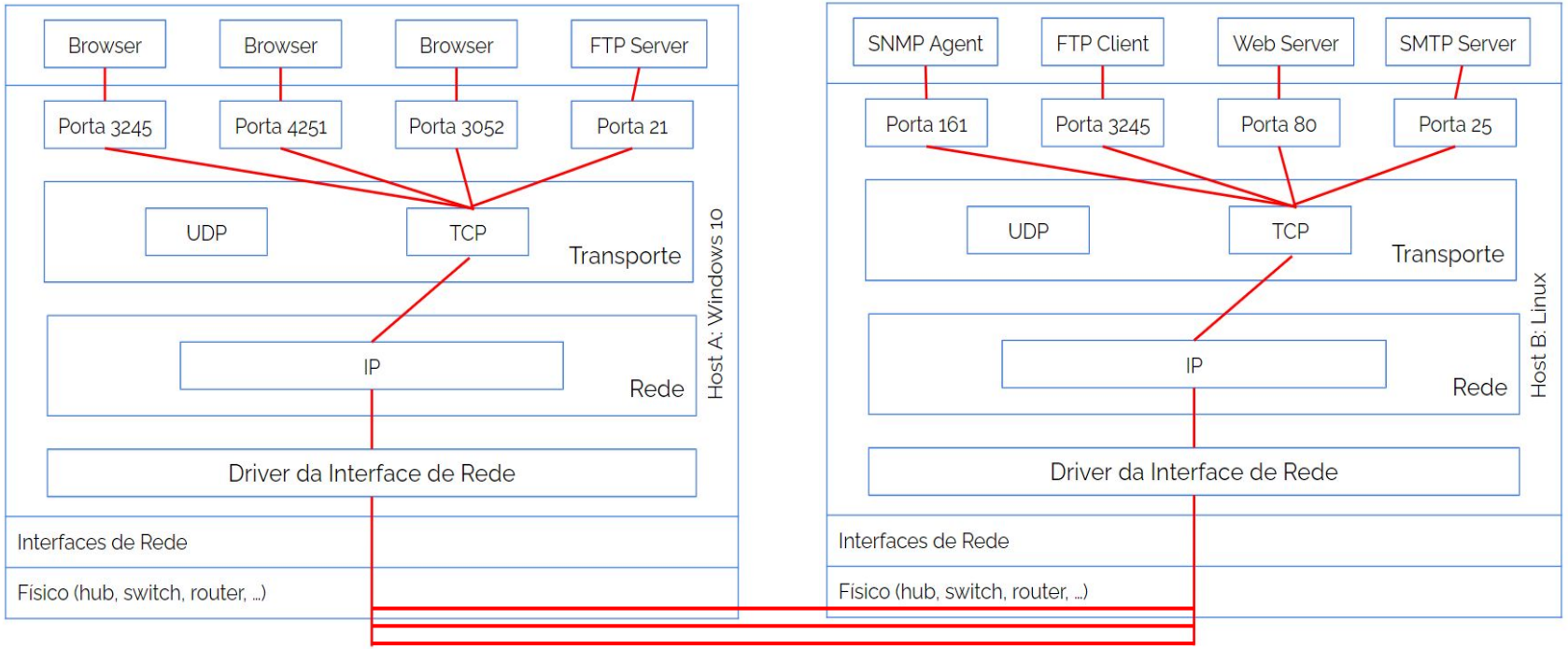
- Processo no receptor que recebe e distribui os dados provenientes dos datagramas da camada de rede
  - Lê o número da porta do pacote
  - Entrega os dados para o socket
  - Encaminha para o processo que pertence ao socket
- Um-para-vários

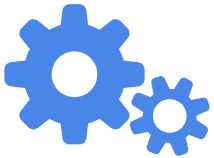
Multiplexação e Demultiplexação



Multiplexação e Demultiplexação

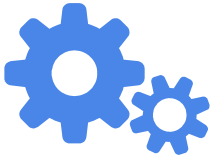



Browser
Browser
Browser
FTP Server
Porta 3245
Porta 4251
Porta 3052
Porta 21
UDP
TCP
Transporte
IP
Rede
Driver da Interface de Rede
Interfaces de Rede
Físico (hub, switch, router, ...)
Host A: Windows 10
SNMP Agent
FTP Client
Web Server
SMTP Server
Porta 161
Porta 3245
Porta 80
Porta 25
UDP
TCP
Transporte
IP
Rede
Driver da Interface de Rede
Interfaces de Rede
Físico (hub, switch, router, ...)
Host B: Linux



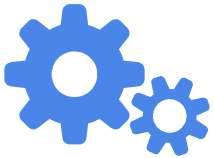
Multiplexação e Demultiplexação

- Entrega os segmentos recebidos ao socket correto
- Coleta dados de múltiplos sockets, envelopa os dados com cabeçalho (usado depois para demultiplexação)



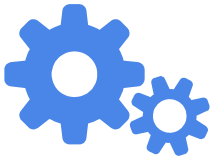
Multiplexação e Demultiplexação

- Computador recebe datagramas IP
- Cada datagrama possui endereço IP de origem e IP de destino
- Cada datagrama carrega 1 segmento da camada de transporte
- Cada segmento possui números de porta de origem e destino (números de porta bem conhecidos para aplicações específicas)
- O hospedeiro usa endereços IP e números de porta para direcionar o segmento ao socket apropriado



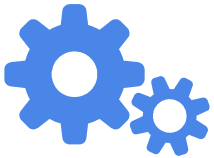
Multiplexação e Demultiplexação

- Como cada máquina é identificada unicamente na Internet ?
  - Número IP
- Como a entidade de rede (IP) identifica qual o protocolo de transporte está sendo utilizado ?
  - Tipo de protocolo está indicado no cabeçalho IP



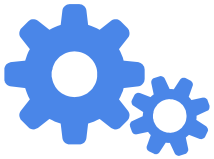
Multiplexação e Demultiplexação

- Dentro do host, como a entidade de transporte identifica qual aplicação está sendo utilizada ?
  - Cada aplicação tem uma **Porta** única no host,
  - É identificada no pacote IP
- Como uma aplicação cliente sabe qual a porta de uma aplicação servidora para poder enviar pacotes?
  - Alguns serviços têm números de portas já convencionadas (portas “bem conhecidas”)



Multiplexação e Demultiplexação

- 1-255: reservadas para serviços padrão (portas “bem conhecidas”)
- 256-1023: reservado para serviços Unix
- 1-1023: Somente podem ser usadas por usuários privilegiados (super-usuários)
- 1024-4999: Usadas por processos de sistema e de usuário
- 5000-... :Usadas somente por processos de usuário



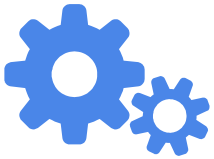
Multiplexação e Demultiplexação

- 21 FTP
- 22 SSH
- 23 Telnet
- 25 SMTP
- 53 DNS
- 69 TFTP
- 79 Finger
- 80 HTTP
- 88 KERBEROS
- 110 POP3
- 135-139 NetBIOS Services
- 161-162 SNMP
- 443 HTTPS (HTTP+SSL)
- 995 POP3S (POP3+SSL)
- 1433 MS-SQL Server
- 2049 NFS
- 3006 MySQL
- 6000 X Windows

2

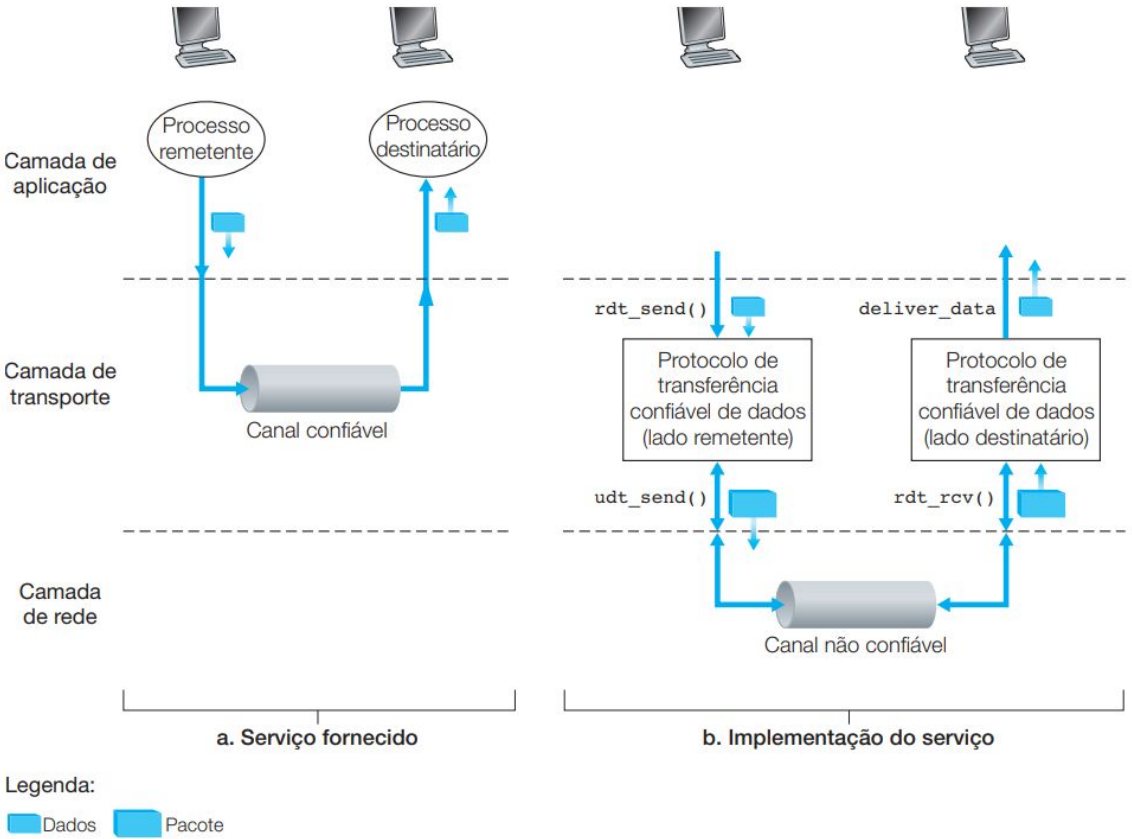
Princípios de Transferência Confiável de Dados

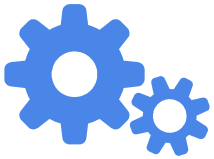
Protocolos rdt, Reconhecimentos



Princípios de Transferência Confiável de Dados

- Desenvolver incrementalmente o transmissor e o receptor de um protocolo confiável de transferência de dados (rdt - Realiable Data Transfer)
- Considerar apenas transferências de dados unidirecionais
  - A informação de controle deve fluir em ambas as direções!
- Usar máquinas de estados finitos (FSM) para especificar o protocolo transmissor e o receptor


Processo remetente
Processo destinatário
Camada de aplicação
Camada de transporte
Camada de rede
Canal confiável
rdt_send()
deliver_data
Protocolo de transferência confiável de dados (lado remetente)
Protocolo de transferência confiável de dados (lado destinatário)
udt_send()
rdt_rcv()
Canal não confiável
a. Serviço fornecido
b. Implementação do serviço
Legenda:
Dados
Pacote

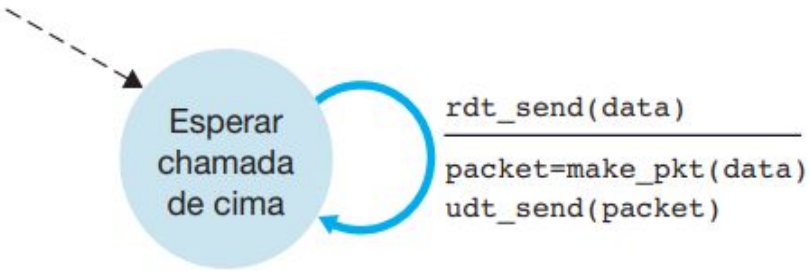


Princípios de Transferência Confiável de Dados

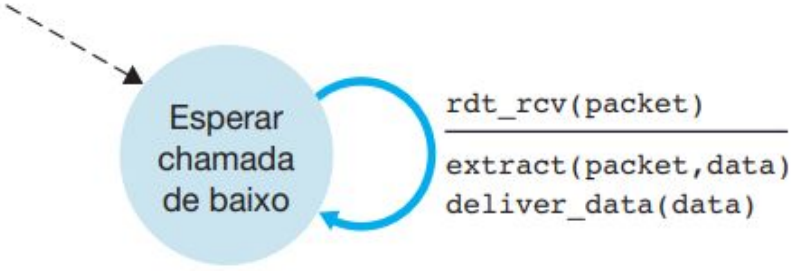
- Transferência confiável de dados sobre um canal perfeitamente confiável: rdt1.0
  - canal de transmissão perfeitamente confiável
    - não há erros de bits
    - não há perdas de pacotes
  - FSMs separadas para transmissor e receptor
    - transmissor envia dados para o canal subjacente
    - receptor lê os dados do canal subjacente

Princípios de Transferência Confiável de Dados

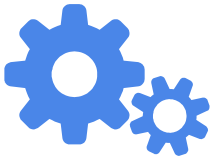
- Transferência confiável de dados sobre um canal perfeitamente confiável: rdt1.0



a. rdt1.0: lado remetente

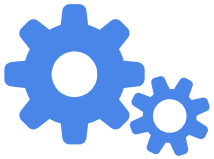


b. rdt1.0: lado destinatário



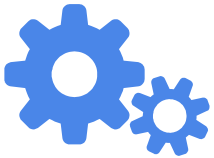
Princípios de Transferência Confiável de Dados

- Transferência confiável de dados usando um canal com erro de bits
  - Canal subjacente pode trocar valores dos bits num pacote
    - Checksum para detectar erros de bits



Princípios de Transferência Confiável de Dados

- Como recuperar esses erros?
  - Reconhecimentos (ACKs): receptor avisa explicitamente ao transmissor que o pacote foi recebido corretamente
  - Reconhecimentos negativos (NAKs): receptor avisa explicitamente ao transmissor que o pacote tem erros
    - Transmissor reenvia o pacote quando da recepção de um NAK

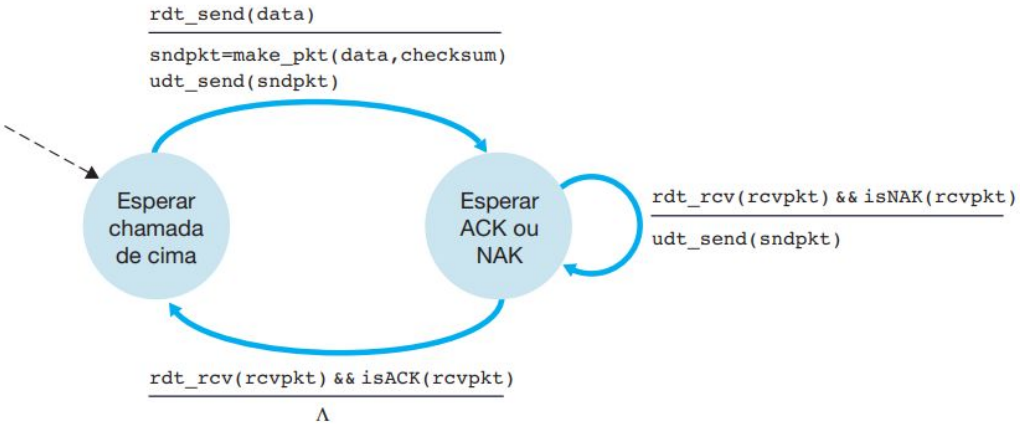


Princípios de Transferência Confiável de Dados

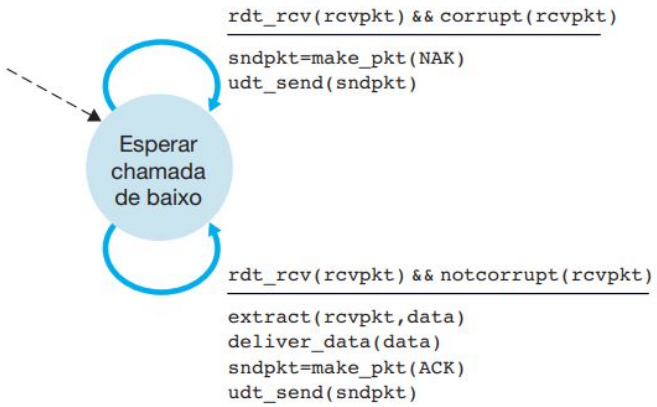
- Mecanismos necessários (rdt2.0):
  - Detecção de erros
  - Retorno do receptor: mensagens de controle (ACK, NAK) receiver->sender

Princípios de Transferência Confiável de Dados

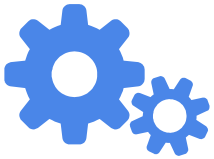
- Transferência confiável de dados usando um canal com erro de bits



a. rdt2.0: lado remetente



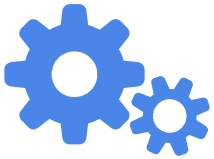
b. rdt2.0: lado destinatário



Princípios de Transferência Confiável de Dados

O que acontece se o ACK/NAK é corrompido ou perdido?

- O Transmissor não sabe o que aconteceu no receptor!
- O Transmissor deve esperar durante um tempo razoável pelo ACK e se não recebê-lo deve retransmitir a informação
    - Não pode apenas retransmitir: possível duplicata

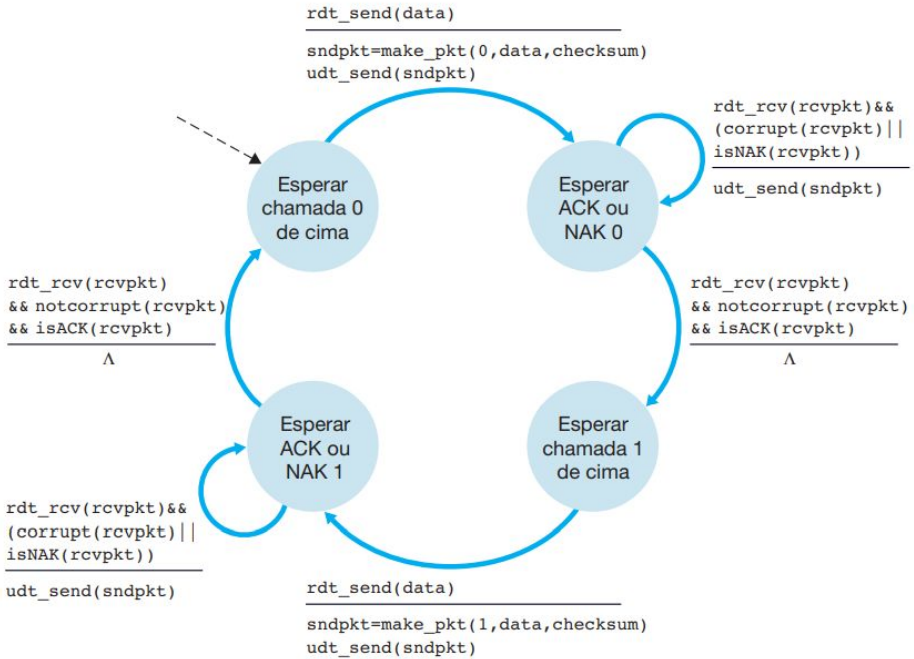


Princípios de Transferência Confiável de Dados

- Tratando duplicatas:
  - Transmissor acrescenta número de seqüência em cada pacote
  - Transmissor reenvia o último pacote se ACK/NAK for perdido
  - Receptor descarta (não passa para a aplicação) pacotes duplicados

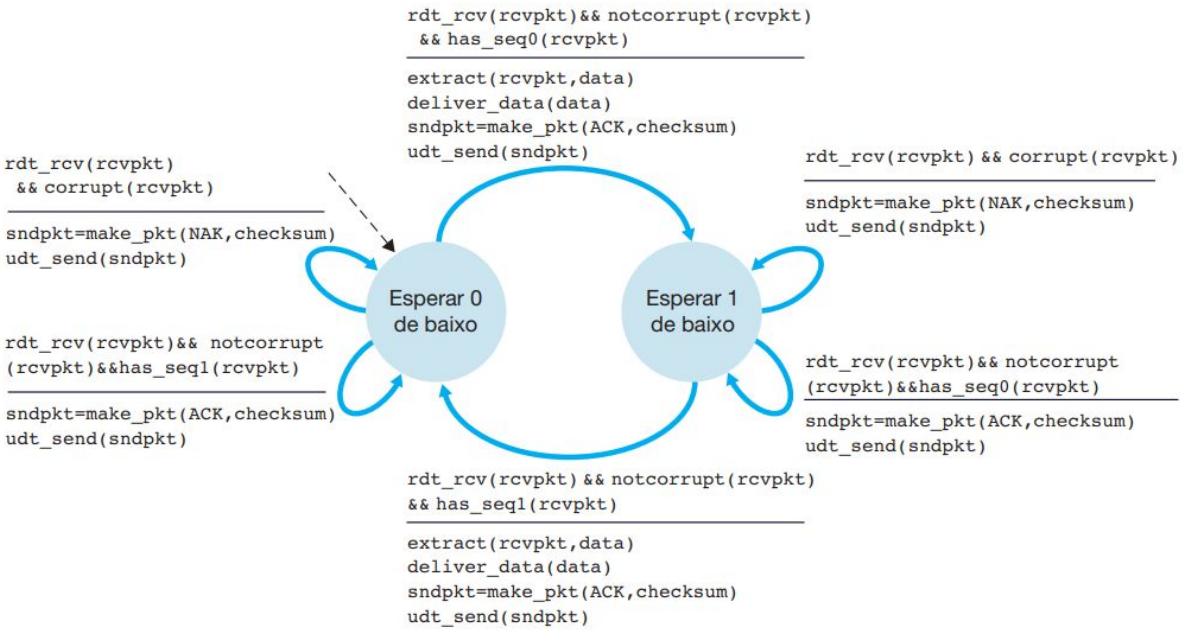
Princípios de Transferência Confiável de Dados

- Rdt 2.1: Transmissor



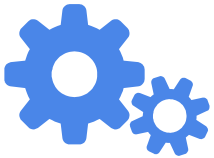
Princípios de Transferência Confiável de Dados

- Rdt 2.1: Receptor



Princípios de Transferência Confiável de Dados

- Transmissor:
  - adiciona número de seqüência ao pacote: 0/1
  - Deve verificar se os ACK/NAK recebidos estão corrompidos
  - Duas vezes o número de estados
  - O estado deve “lembrar” se o pacote “corrente” tem número de seqüência 0 ou 1
- Receptor:
  - deve verificar se o pacote recebido é duplicado
  - estado indica se o pacote 0 ou 1 é esperado
  - O receptor pode não saber se seu último ACK/NAK foi recebido pelo transmissor

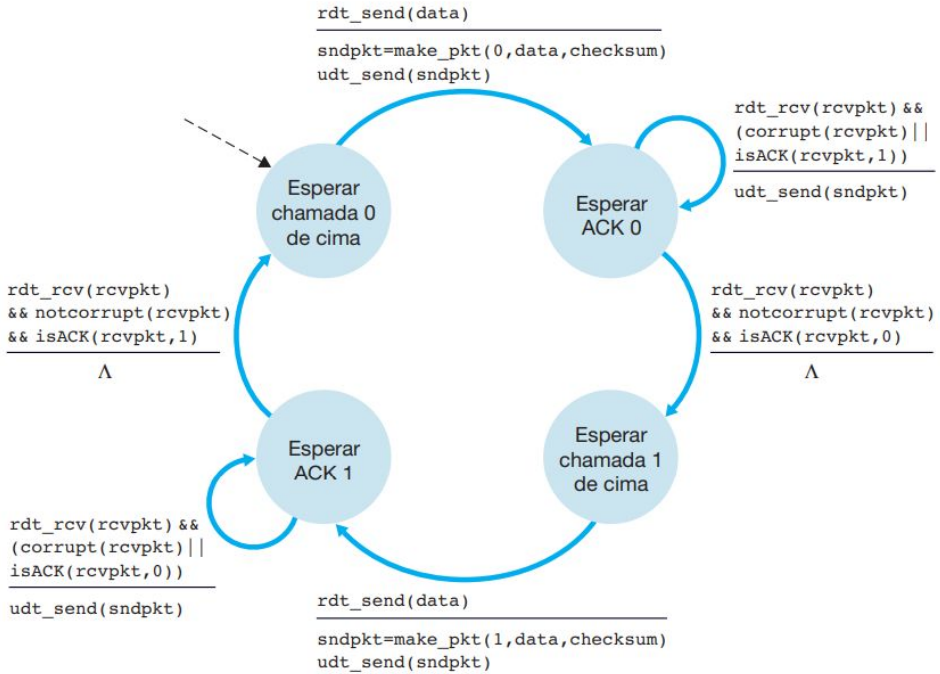


Princípios de Transferência Confiável de Dados

- rdt2.2: um protocolo sem NAK
- Mesma funcionalidade do rdt2.1, usando somente ACKs
- Ao invés de enviar NAK, o receptor envia ACK para o último pacote recebido sem erro
- O receptor deve incluir explicitamente o número de seqüência do pacote sendo reconhecido
- ACKs duplicados no transmissor resultam na mesma ação do NAK: retransmissão do pacote corrente

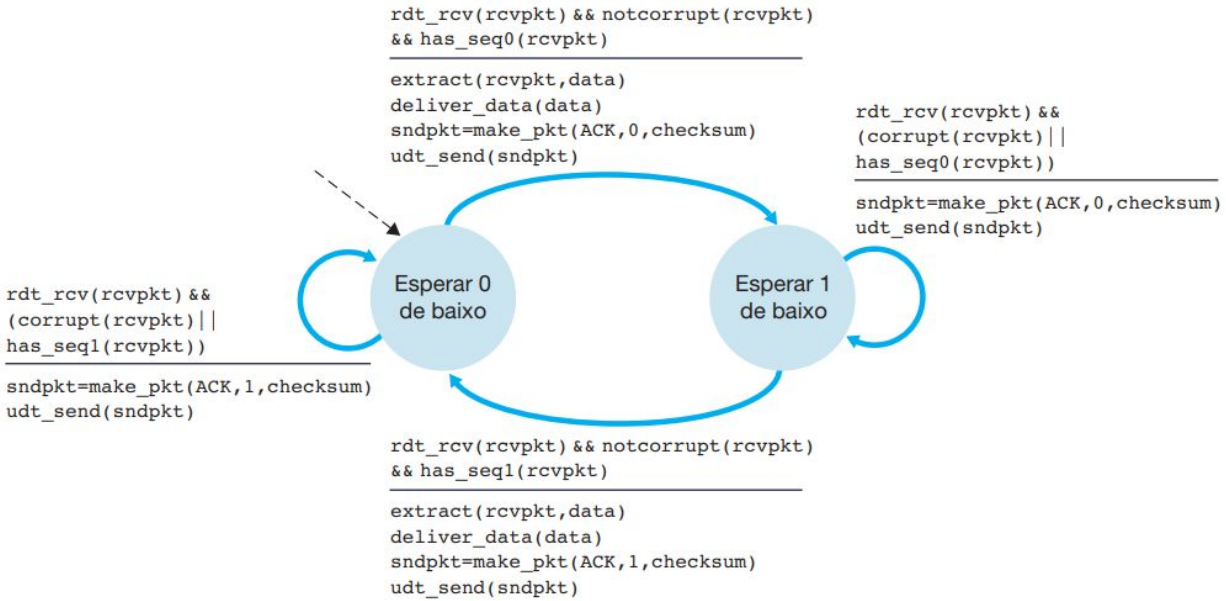
Princípios de Transferência Confiável de Dados

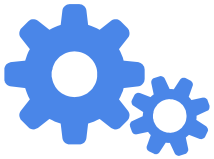
- Rdt 2.2: Transmissor



Princípios de Transferência Confiável de Dados

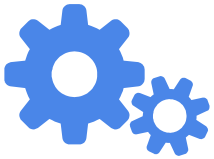
- Rdt 2.2: Receptor





Princípios de Transferência Confiável de Dados

- rdt3.0: canais com erros e perdas
- Nova Hipótese: canal de transmissão pode também perder pacotes (dados ou ACKs)
- Checksum, números de seqüência, ACKs, retransmissões serão de ajuda, mas não o bastante
- Como tratar as perdas?
    - Transmissor espera até que certos dados ou ACKs sejam perdidos, então retransmite

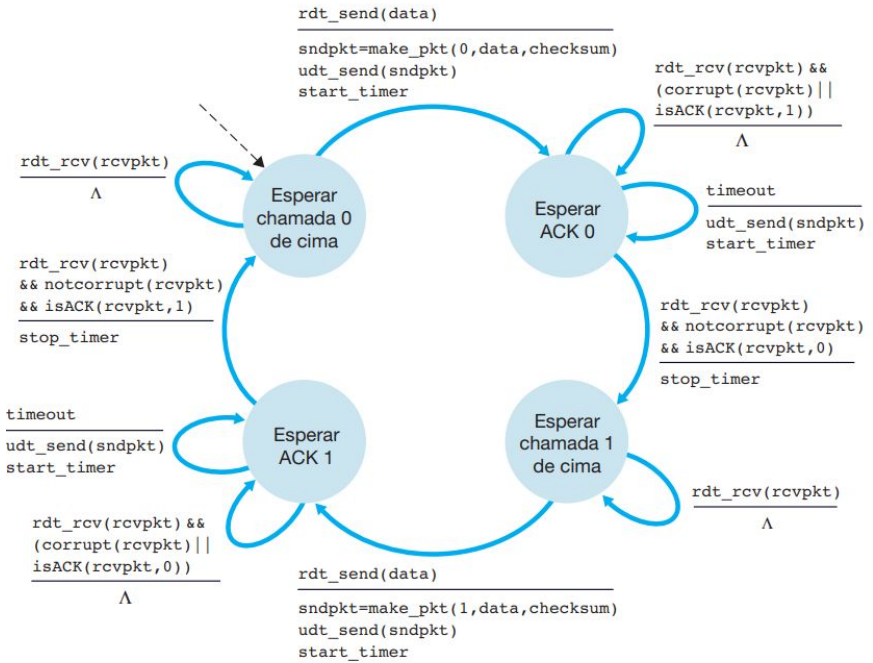


Princípios de Transferência Confiável de Dados

- Abordagem: transmissor espera um tempo “razoável” pelo ACK
  - Retransmite se nenhum ACK for recebido neste tempo
  - Se o pacote (ou ACK) estiver apenas atrasado (não perdido):
    - A retransmissão será duplicata, mas os números de seqüência já tratam com isso
    - O receptor deve especificar o número de seqüência do pacote sendo reconhecido
  - É exigido um temporizador decrescente

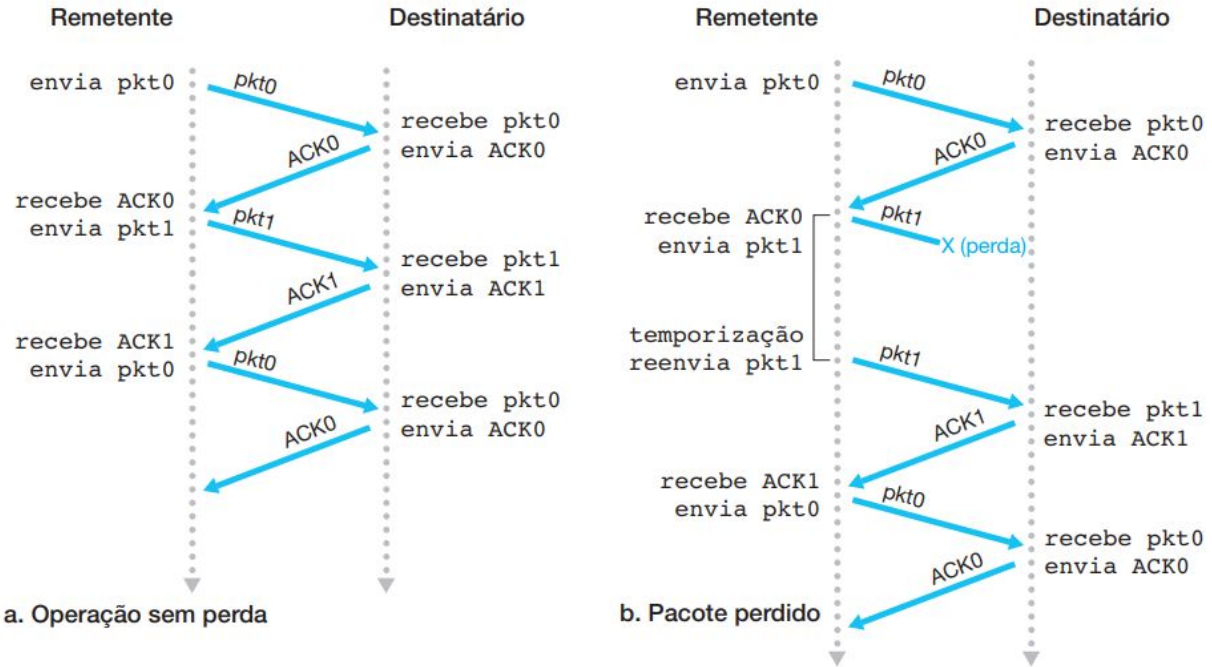
Princípios de Transferência Confiável de Dados

- Rdt 3.0: Transmissor



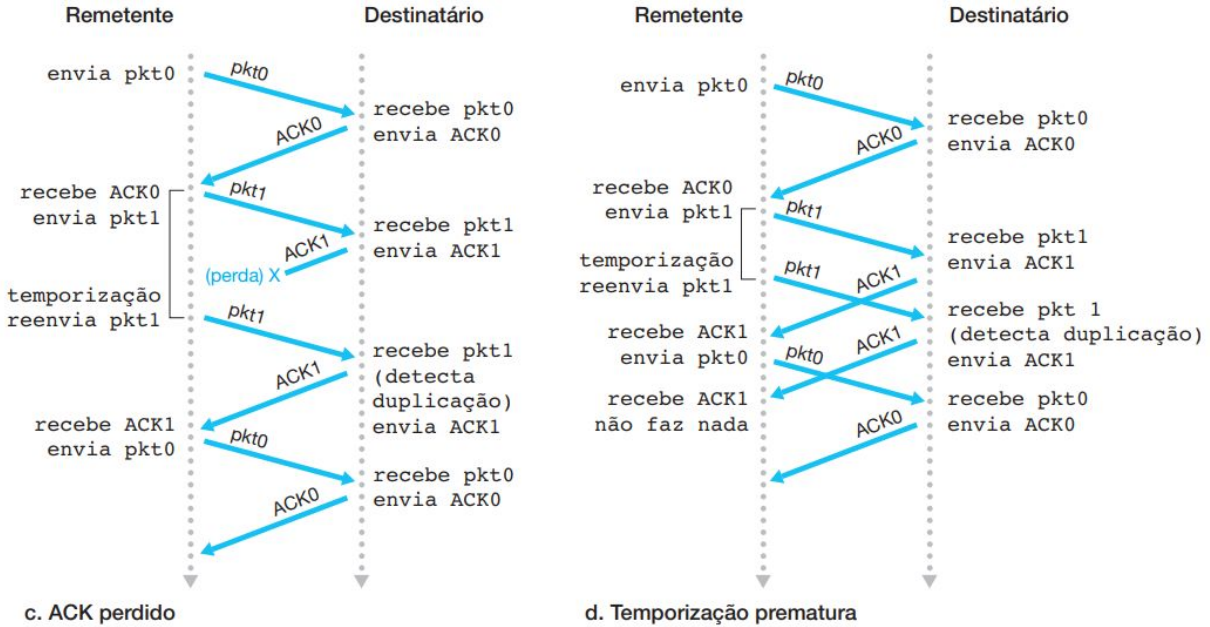
Princípios de Transferência Confiável de Dados

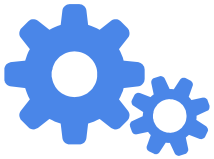
- Rdt 3.0: Operação



Princípios de Transferência Confiável de Dados

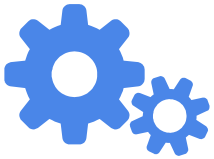
- Rdt 3.0: Operação





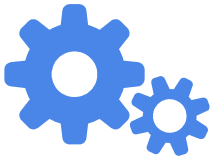
Princípios de Transferência Confiável de Dados - Resumo

- Soma de verificação
  - Usada para detectar erros de bits
- Temporizador
  - Controlar a temporização/retransmissão de um pacote
  - Tratar pacotes perdidos dentro de um canal



Princípios de Transferência Confiável de Dados - Resumo

- Número de sequência
  - Usado para numeração sequencial de pacotes
  - Permitir o paralelismo no envio de pacotes
  - Suportar o envio sem esperar o ACK de cada pacote



Princípios de Transferência Confiável de Dados - Resumo

- Reconhecimento Positivo
  - Avisa o remetente que um ou conjunto de pacotes chegaram
  - ACK
  - Acumulativo ou seletivo
- Reconhecimento Negativo
  - Avisa que um pacote não foi recebido corretamente
  - NAK

Obrigado!

Dúvidas?