Parte 3

Protocolos Internet TCP/IP

Os protocolos da Internet TCP/IP foram primeiramente apresentados a mais de 15 anos, muito tempo considerando a era da informação; todavia, muitos de seus princípios fundamentais continuam atuais, e mais, com a grande difusão da Internet, estes protocolos formam hoje a tecnologia hegemônica das redes de computadores.

Arquitetura da Internet TCP/IP

O conjunto de **protocolos TCP/IP** (*Transmission Control Protocol/Internet Protocol*) é um padrão industrial de protocolos destinados a redes geograficamente distribuídas, ou WANs (*wide area networks*), sendo as principais peças da **arquitetura Internet**.

A **arquitetura Internet** objetiva a interligação de **computadores**, não importando em qual tipo de rede os mesmos estejam conectados, a qualquer outro computador da rede mundial de computadores. Para interligar redes distintas a arquitetura Internet usa uma máquina como ponto de ligação entre as redes, sendo está máquina conhecida como **roteador** (ou *gateway)*. Os roteadores são os responsáveis pelo roteamento das mensagens na malha que forma a Internet. (Figura 3.1)

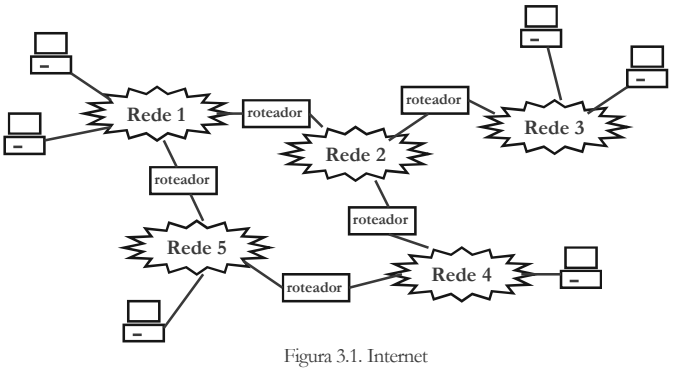


Figura 3.1. Internet

Os protocolos da arquitetura Internet TCP/IP estão organizados em quatro camadas: a **camada de aplicação**, a **camada de transporte**, a **camada de rede,** interligando as inter-redes, e a **camada enlace/física**, inferior, representando os protocolos de enlace e a rede física (Figura 3.2).

Na **camada de transporte** a arquitetura baseia-se principalmente em um serviço de transporte orientado a conexão, fornecido pelo protocolo **TCP** (*Transmission Control Protocol*). Todavia, um serviço de datagrama, não orientado a conexão, também é disponível com o protocolo **UDP** (*User Datagram Protocol*). Na **camada de rede**, temos um serviço não-orientado a conexão, fornecido pelo protocolo **IP** (*Internet Protocol*).

Figura 3.2. Pilha de protocolos da Internet

Camada de Transporte

Situada entre a camada de aplicação e a camada de rede, a **camada de transporte** tem a função de prover um **canal de comunicação lógico fim-a-fim** entre os processos de aplicação rodando em diferentes computadores, sem se preocupar com os detalhes da infra-estrutura física usada para carregar as mensagens entre eles.

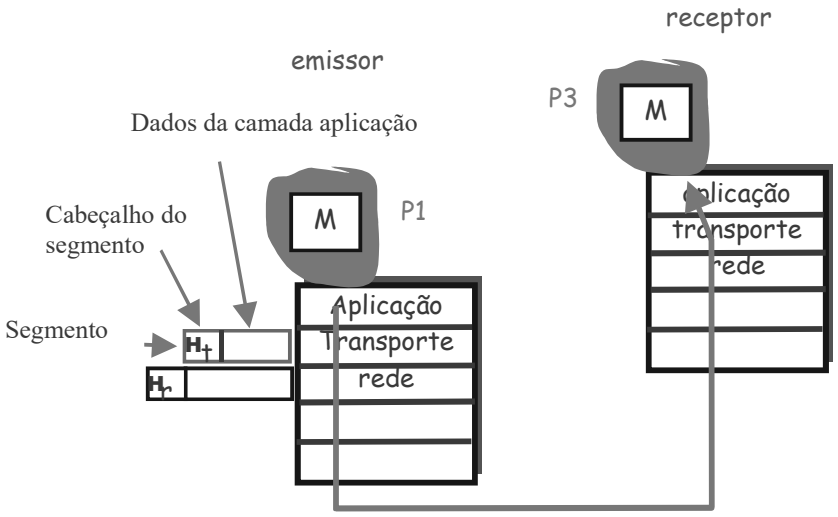
Os protocolos de transporte são implementados nos sistemas terminais, já que oferecem um canal lógico fim-a-fim às aplicações, não necessitando, no entando, serem implementados nos roteadores da rede, os quais atuam somente até a camada rede.

No lado do emissor, as **mensagens** redebidas das aplicações são fragmentadas e encapsuladas em **unidades de dados de protocolos,** ou **PDUs** (*protocol data unit*), chamadas **segmentos,** aos quais adiciona-se um cabeçalho (Figura 3.3). Cada segmento é então repassado a camada rede que por sua vez encapsula em **unidades de dados de protocolos** da camada de rede, ou **datagramas**.

Relação entre a camada de transporte e a camada de rede

Na Internet o protocolo da **camada rede** é chamado **IP** e fornece um serviço de comunicação de **computador-a-computador** na inter-rede. O modelo de serviço do **protocolo IP** é do tipo **"melhor esforço"** (*best effort*), isto é, ele faz o melhor esforço para o envio de um datagrama entre computadores, mas não dá nenhuma garantia. Em particular, não garante a entrega do datagrama, não garante que sejam entregues em ordem e nem garante a integridade dos dados. É por isso chamado de **serviço não garantido**.

Os **protocolos de transporte TCP e UDP** estendem a entrega computador-a-computador, fornecida pelo IP, pela entrega **processo-a-processo**, que é chamada de **multiplexação/demultiplexação de aplicações**. O TCP e o UDP oferecem também checagem da integridade dos dados, incluindo campos para **detecção de erros** no seu cabeçalho. No caso do **UDP**, a **multiplexação/demultiplexação de aplicações** e a **checagem de erros** nos dados, são os dois únicos serviços oferecidos, sendo portanto um **serviço não garantido.** O **TCP**, além destes dois, oferece ainda a **transferência garantida**, usando controle de fluxo, números de sequência, reconhecimentos e temporizadores.



O serviço de multiplexação e demultiplexação de aplicações

O protocolo IP entrega dados entre dois sistemas terminais (*hosts*), cada qual identificado por seu **endereço IP**. A responsabilidade dos protocolos de transporte é entregar estes dados (segmentos) a aplicação apropriada rodando em cada *host*.

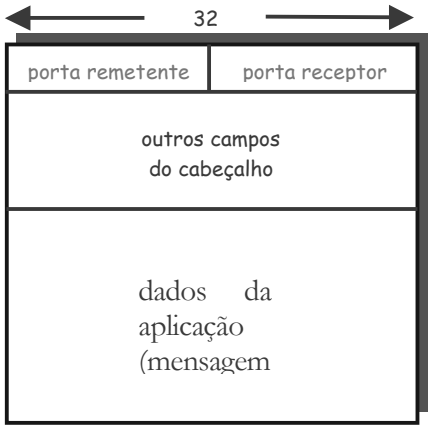


Figura 3.4. Segmento da camada transporte

Cada um dos **segmentos** da camada transporte tem em seu cabeçalho um campo que indica a qual processo o mesmo deve ser entregue. Estes campos são conhecidos como **números de porta**. O cabeçalho inclui um campo com o número de **porta do emissor** e o número de **porta do receptor**. (Figura 3.4)

Os números de porta variam de 0 a 65535, sendo que até a porta 1023 são números reservados para aplicações específicas. Por exemplo:

HTTP – porta 80

SMNP – porta 25

TELNET – porta 23

SSH – porta 22

FTP – porta 21.

Considere como exemplo a aplicação **Telnet**, que utiliza a **porta 23**. Quando um **cliente Telnet** inicia uma seção, ele envia ao **servidor Telnet** um segmento TCP destinado à porta 23 (porta reservada para a aplicação Telnet) e coloca como número de porta da fonte uma porta que não esteja sendo utilizada por nenhum processo no *host* cliente, por exemplo, porta **X**. Quando o servidor recebe o segmento, ele verifica que o mesmo é endereçado a porta 23 e então sabe que se

trata da aplicação Telnet e a porta da fonte **X** vai identificar um processo Telnet específico (já que

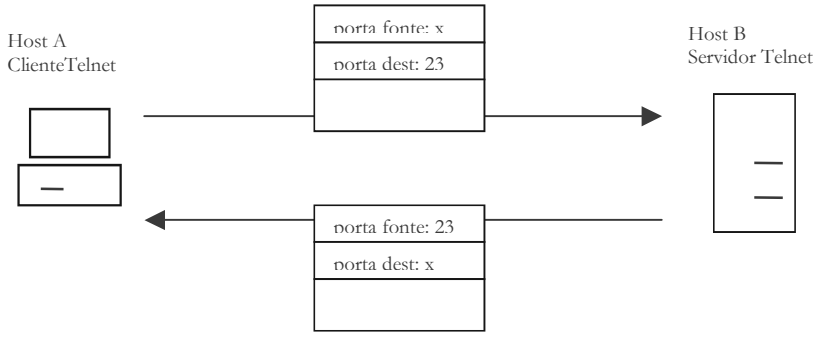


Figura 3.5. Uso de portas para acessar servidor de aplicação Telnet

pode haver outras solicitações). (Figura 3.5).

UDP (*User Datagram Protocol*)

O **protocolo UDP** adiciona ao IP a **multiplexação e demultiplexação** de aplicações e o mecanismo de **detecção de erros**.

No UDP não há processo de abertura de conexão para o envio de dados, por isto é chamado de **protocolo sem conexão** (*connectionless*).

Características:

- Sem estabelecimento de conexão, não introduzindo, portanto, atrasos para esta tarefa.
- Não mantém estado da conexão, que implicaria em *buffers* (memórias) de envio e recepção, números de seqüência e reconhecimento.
- Tem pequeno *overhead* (informações de controle) no cabeçalho.
- Taxa de envio sem regulação ou controle de fluxo.

Por estas características é apropriado para aplicações tempo real, como telefonia e transferência de áudio e vídeo sobre a Internet.

O formato do "**segmento" UDP** (alguns autores chamam de *datagrama UDP,* pois pouco acrescenta ao *datagrama IP*) é bastante simples (Figura 3.6), além dos campos reservados para as **portas de origem e destino**, há um campo que indica o **comprimento do segmento** e o ***checksum***, utilizado para o reconhecimento de erros no segmento. O campo de **dados da aplicação** é preenchido com os dados da aplicação, por exemplo, para a aplicação DNS os dados podem ser mensagens de consulta e resposta, para aplicações de aúdio tempo real, o campo é preenchido com amostras de aúdio.

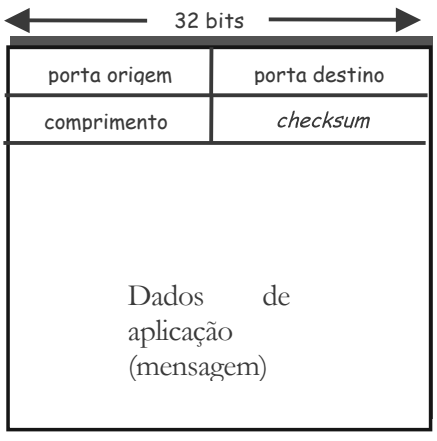


Figura 3.6. Formato do segmento UDP

Checksum

O ***checksum*** do UDP permite a **detecção de erros** nos dados transmitidos. Para isto, o **emissor UDP** faz o **complemento 1** da **soma** de todos as palavras de 16 bits do segmento e coloca o resultado no campo ***cheksum***. Por exemplo, suponha que temos três palavras de 16 bits:

0110011001100110
0101010101010101
0000111100001111

A soma será

0110011001100110
0101010101010101
\+ _____
1011101110111011

Adicionando a terceira palavra a esta soma

1011101110111011
0000111100001111
\+ _____
1100101011001010

O **complemento 1** é obtido invertendo cada bit 1 por 0 e vice-versa. Desta forma o complemento da soma será 0011010100110101, o qual será o *cheksum*. No lado do **receptor UDP**, todas as palavras de 16 bits recebidas são adicionadas, incluindo o *cheksum*.Se não houve erros na transmissão, a soma será 1111111111111111. Se um dos bits for 0, então é sabido que houve erros.

TCP (*Transmission Control Protocol*)

O **protocolo TCP,** como o UDP, também oferece a **multiplexação/demultiplexação** de aplicações e o mecanismo de **detecção de erros**. A grande diferença é que o TCP é um **protocolo orientado a conexão** e com **transferência garantida,** onde os dois processos devem acordar entre eles uma abertura de conexão para que os dados possam ser transferidos. Além destas características, o TCP integra ainda um serviço de **controle de fluxo,** que assegura que nenhum dos lados da comunicação envie pacotes rápido demais, pois uma aplicação em um lado pode não conseguir processar a informação na velocidade que está recebendo, e um serviço de **controle de congestão** ajuda a prevenir congestionamentos na rede.

Uma conexão TCP é uma conexão *full-duplex* (isto é, em ambos os sentidos e simultânea) e é sempre **fim-a-fim**, entre o *host* emissor e o *host* receptor. Uma vez estabelecida à conexão os dois processos podem trocar informações. O processo cliente, no lado **emissor,** passa o bloco de dados através da **porta** apropriada. O TCP então manipula estes dados, dirigindo para o ***buffer* de envio**. Os dados são então fragmentados e encapsulados na forma de **segmentos**. Os segmentos, por sua vez, são passados a camada rede onde eles são separadamente encapsulados em **datagramas IP**, que são enviados através da rede. Quando o TCP do **receptor** recebe os dados, os mesmos são recebidos no ***buffer* de recepção**. A aplicação no lado do receptor então lê os dados a partir deste *buffer*.

Transferência garantida: analogia com a compra de uma enciclopédia

Voltando a analogia com o sistema postal, pode-se dizer que o serviço de entrega de correspondências entre usuários é um serviço tipo **melhor esforço** (*best effort*), isto é, ele faz o melhor esforço para o envio de uma carta entre usuários, mas não dá nenhuma garantia. Em particular, não garante a entrega da carta, pois a mesma pode se perder e não há formas de avisar o emissor sobre o ocorrido. Da mesma forma, não há um serviço de confirmação de recebimento do receptor ao emissor. É por isso que pode ser chamado de **serviço não garantido**.

Este serviço é análogo ao serviço oferecido pelo protocolo da camada rede da Internet, o IP. Na Internet o **serviço garantido** é implementado pelo TCP, e roda sobre o serviço não garantido fornecido pelo IP, utilizando números de sequência, reconhecimentos e temporizadores.

Vamos comparar o serviço garantido fornecido pelo TCP utilizando uma analogia com o que acontece na compra de uma enciclopédia em fascículos.

Suponha que você resolva adquirir uma enciclopédia, cujos volumes são vendidos em fascículos que são entregues pelo correio. Imagine que a coleção completa tenha 100 fascículos sendo eles enviados um a cada semana.

Quando você resolve fazer a compra, você envia uma carta a editora responsável pela venda da enciclopédia com seu pedido. A editora então faz a **abertura de um cadastro** de cliente para você e na semana seguinte lhe envia a confirmação do seu cadastro, juntamente com o primeiro fascículo e os procedimentos para confirmação de recebimento e pagamento.

Suponha que a cada cinco fascículos recebidos, você deve enviar uma correspondência de **confirmação de recebimento**, juntamente com a parcela de pagamento correspondente.

Como a entrega dos fascículos usa o **serviço não garantido** dos correios, os mesmos podem ser perdidos ou mesmo danificados no transporte. Caso isto ocorra, você deverá enviar a editora uma carta de aviso informando o número fascículo não chegou ou que chegou danificado. A editora então fará o **reenvio** do fascículo com problemas.

As trocas de mensagens entre o comprador e a editora continuam até que o total de 100 fascículos sejam entregues e a última confirmação e o respectivo pagamento seja efetuado. Neste momento, a editora **encerrará o cadastro** do cliente e você poderá usufruir da enciclopédia completa.

Voltando aos protocolos da Internet, para implementar o serviço garantido no TCP, ocorrem procedimentos similares aos efetuados entre o cliente e o vendedor da enciclopédia. No caso do TCP, primeiro há uma fase chamada de **abertura de conexão**, onde se estabelece os parâmetros para a comunicação, como inicalização de variáveis e *buffers*. Em seguida, inicia-se a **troca de dados**, onde cada pacote de informação trocado entre o emissor e o receptor tem um **número de sequência**, o qual vai ser tomado como base para o receptor **reconhecer** o recebimento. Caso o reconhecimento não seja confirmado dentro de um tempo limite, o emissor **retransmite** o pacote.

Protocolo com Transmissão Garantida

Para garantir uma entrega de dados livre de erros, os protocolos com transmissão garantida, como o TCP, utilizam uma técnica conhecida como **confirmação positiva com retransmissão**. A técnica exige que um receptor comunique-se com a origem, retornando uma mensagem de **reconhecimento** (*acknoledge*), a medida que recebe os dados. O transmissor, por sua vez, inicia um temporizador para cada pacote que envia, e **retransmite** o pacote se este temporizador se complete antes que chegue uma confirmação de recebimento.

A figura 3.7 mostra um exemplo de confirmação positiva.

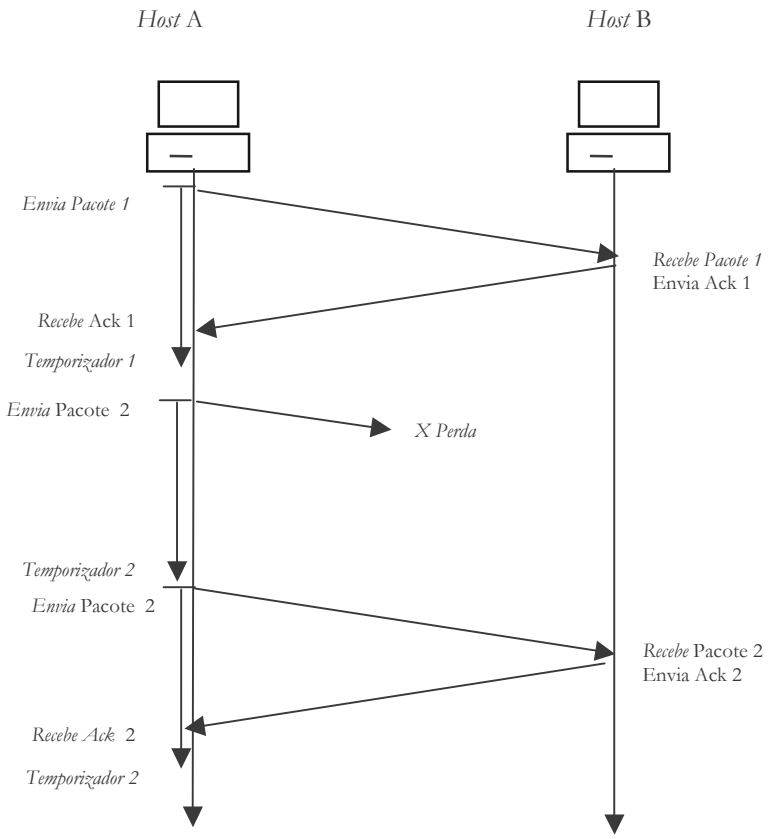


Figura 3.7. Protocolo de confirmação positiva.

O problema de um protocolo como o da figura 3.7 é que o emissor deve esperar o reconhecimento de cada pacote antes que um novo pacote possa ser enviado, o que torna a transmissão bastante ineficiente. Protocolos mais elaborados, como o TCP, permitem que o emissor transmita múltiplos pacotes antes de esperar uma confirmação. No TCP isto é implementado através de um mecanismo conhecido como **janelas deslizantes**.

No mecanismo de janelas deslizantes, mostrado na figura 3.8, o emissor pode enviar uma sequência de pacotes, contidos dentro de uma "janela" de tamanho fixo, antes de esperar uma confirmação.

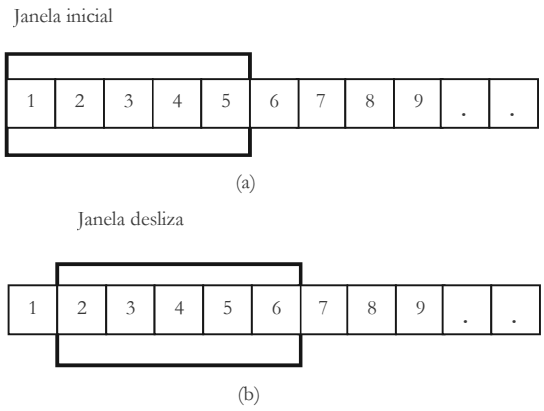


Figura 3.8 Mecanismo de janelas deslizantes

Na figura 3.8 (a) , os pacotes contidos dentro da janela (numerados de 1 a 5) podem ser enviados em sequência. Quando o transmissor redece a confirmação do primeiro pacote da janela, a janela "desliza", figura 3.8 (b), permitindo que um novo pacote seja enviado.

A figura 3.9 mostra uma sequência de três pacores sendo transmitida com o mecanismo de janela deslizante.

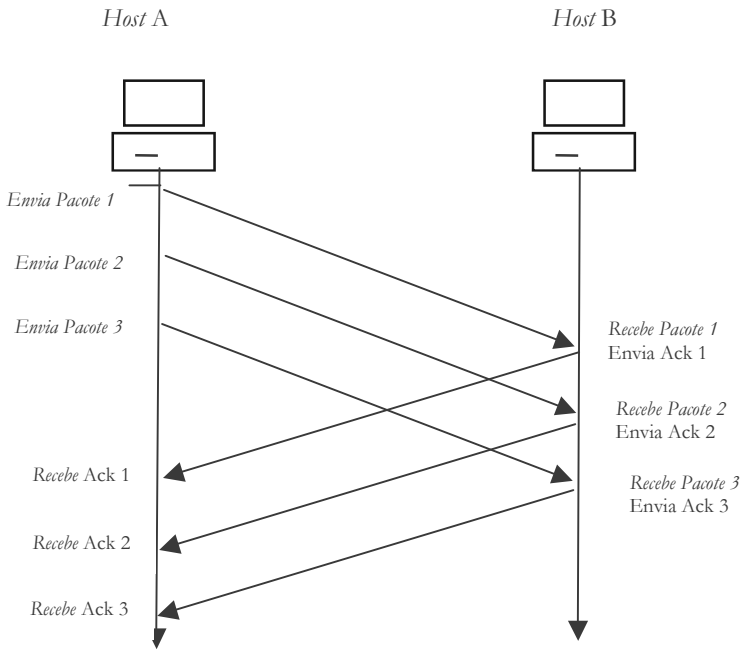


Figura 3.9. Sequência de pacotes transmitidos com janelas deslizantes.

Segmento TCP

A figura 3.7 mostra a estrutura do **segmento TCP**. No cabeçalho, além dos **números de porta** e ***checksum*** que também existem no UDP, há outros campos com informações necessárias a implementação do serviço de transferência garantida, controle de fluxo e controle de congestionamento.

O campo de **dados da aplicação** do segmento TCP (Figura 3.10), contém um fragmento ou pedaço dos dados da aplicação, cujo tamanho máximo, chamado de **MSS** (*maximum segment size*), depende da implementação do TCP. Os valores típicos são 1.500 bytes, 536 bytes e 512 bytes, não incluindo o cabeçalho. (Em geral o valor de MSS é escolhido para evitar a fragmentação do datagrama IP na camada inferior, conforme veremos a frente. Este valor em algumas implementações pode ser configurado manualmente ou estabelecido automaticamente pelo protocolo).

Outros campos fundamentais do segmento TCP são os seguintes:

- **Número de seqüência** e **reconhecimento**, utilizado para o emissor e receptor implementarem o serviço de transferência garantida.
- Tamanho da **janela do receptor**, usado para o controle de fluxo, e indica o número de bytes que o receptor é capaz de receber.
- **Tamanho do cabeçalho**, especifica o tamanho da cabeçalho, que pode variar em funções do campo de opções, todavia, tipicamente, o tamanho do cabeçalho é de 20 bytes.
- O campo de **opções** é usado quando o emissor e receptor precisam negociar o tamanho máximo de segmento (MSS).
- Os ***flags*** (bandeiras) contém 6 bits. O **Ack** é usado para indicar que o campo de reconhecimento é válido, O **Rst**, **Syn** e **Fin** são usados para abertura e encerramento de conexão, o **Psh** indica que o receptor deve passar imeidatamente o dado a camada superior e o **Urg** indica um dado urgente.

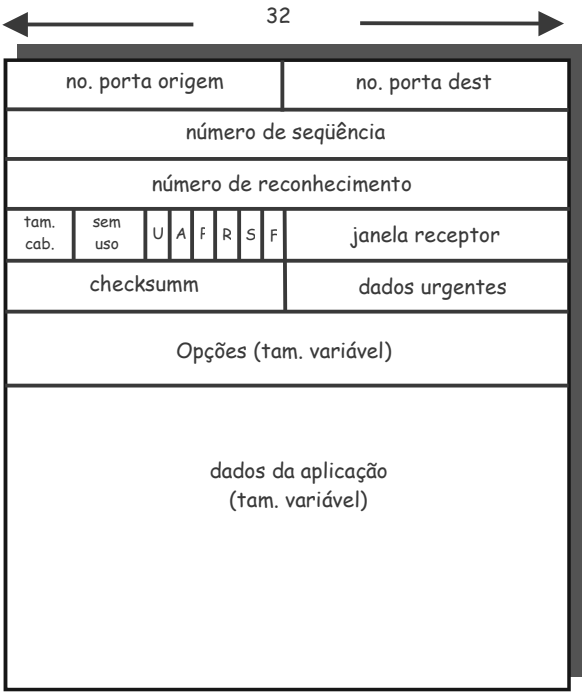


Figura 3.10. Formato do segmento TCP

Números de seqüência e reconhecimento no TCP

Dois campos importantes do segmento TCP são os **números de seqüência** e **reconhecimento**, os quais fazem a parte crítica do trabalho de **transferência de dados confiável**.

Como vimos, os **dados das aplicações** são transportados pelos segmentos TCP. Caso as mensagens forem maior que o valor de **MSS, tamanho máximo do segmento**, as mesmas são fragmentadas para poderem ser acomodadas na parte de dados do segmento. Por exemplo, um arquivo GIF de 500K bytes trocado pelo HTTP será fragmentado em vários pedaços para ser transmitido pelo TCP. Os **números de seqüência** servem, portanto, para que o lado receptor TCP possa reordenar corretamente os dados recebidos.

Os **números de seqüência** não correspondem a uma série de segmentos transmitidos, mas refletem a quantidade de bytes que o TCP está transmitindo. Por exemplo, suponha que o bloco total de dados que será transmitido tenha 500.000 bytes, que o valor de MSS é de 1.000 bytes, e que o primeiro byte dos dados é numerado como zero. Para transmitir esta quantidade de bytes o TCP formará 500 segmentos. Ao primeiro segmento atribui-se o número de seqüência zero, ao segundo 1000, ao terceiro 2000 e assim por diante. (Figura 3.11).

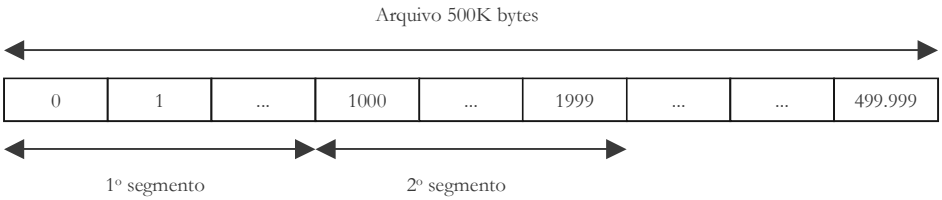


Figura 3.11. Divisão de um arquivo de dados em

Os **reconhecimentos** servem para o receptor informar o emissor quais blocos que foram recebidos corretamente. Todavia, lembre-se que uma comunicação TCP é sempre *full-duplex*, o que significa que o *host* A pode estar recebendo dados do *host* B ao mesmo tempo em que está enviando dados ao *host* B (como parte da mesma conexão TCP). Desta forma, haverá números de reconhecimentos para dados seguindo de A para B e outros para dados seguindo de B para A.

> O **número de reconhecimento** que o *host* A coloca no seu segmento é o **número de seqüência** do próximo byte que o *host* A espera receber do *host* B.

Por exemplo, suponha que o *host* A recebeu todos os bytes numerados de 0 a 535 de B e que está prestes a enviar um segmento a B. Neste caso, o *host* A coloca como número de reconhecimento 536, o que vai indicar a B que o mesmo recebeu todos os bytes até este número.

Em outro exemplo, suponha que o *host* A recebeu todos os bytes numerados de 0 a 535 de B e em seguida recebeu de B um segmento contendo bytes de 900 a 1000. Note que A não recebeu os bytes que vão de 536 a 899. Como A ainda está esperando bytes a partir de 536, ele reenvia a B um segmento com número de reconhecimento 536. Continuando este exemplo, suponha agora que A receba o segmento que faltava, com os bytes que vão de 536 a 899. Neste caso, como ele já recebeu inclusive os dados contendo os bytes de 900 a 1000, ele envia um reconhecimento com número 1001. Isto é chamado de **reconhecimento cumulativo**, que indica que recebeu todos os bytes até este número

Telnet: Caso de estudo para números de seqüência e reconhecimento

O **Telnet** é uma aplicação interativa usada para acesso remoto a sistemas e roda sobre o protocolo de transporte **TCP**.

O Telnet permite que um usuário utilize uma máquina A e estabeleça uma seção interativa em uma máquina B, como se estivesse utilizando um terminal. Quem solicita o Telnet assume o papel de cliente. Cada caractere digitado pelo usuário cliente será enviado ao computador remoto; o computador remoto então enviará uma cópia de cada caractere para ser mostrado na tela do cliente. Desta forma, cada caractere atravessa a rede duas vezes entre o tempo em que o usuário digita uma tecla e a visualização da mesma na tela.

Vamos examinar os segmentos TCP trocados durante uma seção Telnet (Figura 3.12). Suponha que o usuário tecla a letra "C". Suponha ainda que os **números de seqüência** iniciais usados pelo cliente e pelo servidor sejam 42 e 79, respectivamente. Isto indica que o primeiro byte a ser enviado pelo cliente ao servidor terá o número de seqüência 42 e o primeiro byte a ser enviado pelo servidor ao cliente terá o número de seqüência 79. Lembre também que o **número de reconhecimento** indica o número de seqüência do próximo byte esperado. Desta forma, depois de estabelecia a

conexão TCP, e antes do envio de quaisquer dados, o cliente esta esperando pelo byte 79 e o servidor está esperando pelo byte 42.

A figura 3.12 mostra três segmentos trocados entre o cliente e o servidor. O primeiro segmento é enviado pelo cliente, contendo um caractere ASCII com a letra "C" (número de seqüência 42). O segundo segmento é enviado pelo servidor ao cliente e serve para dois propósitos: provê um reconhecimento do caractere recebido (número de reconhecimento 43) e envia o caractere "C" de volta para ser apresentado na tela do cliente (número de seqüência 79). No terceiro segmento trocado, o cliente reconhece o caractere recebido (número de reconhecimento 80).

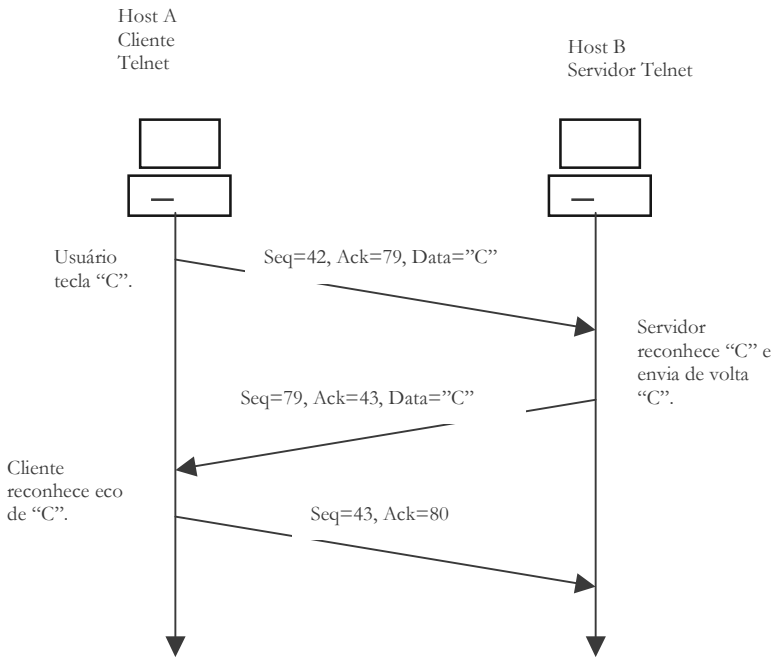


Figura 3.12. Números de seqüência e reconhecimento para a aplicação Telnet

O serviço transferência de dados garantida no TCP

Para criar o **serviço de transferência de dados garantida** o TCP manipula três grandes eventos relacionados à transmissão/retransmissão de dados.

1. Quando recebe dados da camada aplicação o TCP cria segmentos com **números de seqüência**, correspondentes aos próximos número de seqüência a serem transmitidos, e inicia um **temporizador** para cada segmento criado.
2. Caso o temporizador de um segmento enviado **estoure o tempo** (*time-out*), o TCP retransmite este segmento.
3. Caso o TCP receba um **reconhecimento** um segmento enviado (ou de um conjunto de segmentos), ele cancela os temporizadores remanescentes a estes segmentos; ou ainda, caso receba reconhecimentos de segmentos que já haviam sido reconhecidos (reconhecimentos cumulativos), ele retransmite os segmentos cujos números de seqüência são superiores ao reconhecimento cumulativo.

Vamos explicar como estes eventos que são tratados pelo TCP analisando alguns **cenários**. No primeiro cenário (Figura 3.13), o *host* A envia 8 bytes de dados ao *host* B (com número de seqüência

92). O *host* B envia reconhecimento dos 8 bytes recebidos (reconhecimento 100), o qual é perdido. Depois do estouro do temporizador do segmento 92, o mesmo é reenviado pelo *host* A. Quando o *host* B recebe o segmento duplicado, ele reenvia o reconhecimento.

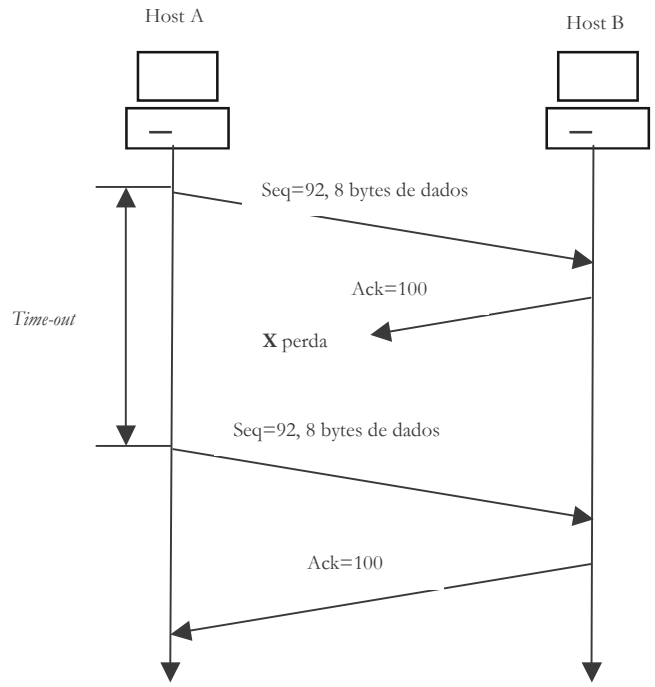


Figura 3.13. Retransmissão devido a reconhecimentos perdidos

No segundo cenário (Figura 3.14), o *host* A transmitiu ao *host* B um segmento com 8 bytes (número de seqüência 92) e em seguida mais um segmento com 20 bytes (número de seqüência 100). O *host* B recebeu estes segmentos e enviou números de reconhecimento (100 e 120 respectivamente). Todavia, o reconhecimento do segmento 92 (reconhecimento 100) chegou depois do *time-out*. Logo o *host* A retransmitiu o segmento com número de seqüência 92. Como o *host* B já havia recebido este segmento e também o seguinte (com número de seqüência 100), ele reenviou o reconhecimento cumulativo deste último segmento (reconhecimento 120).

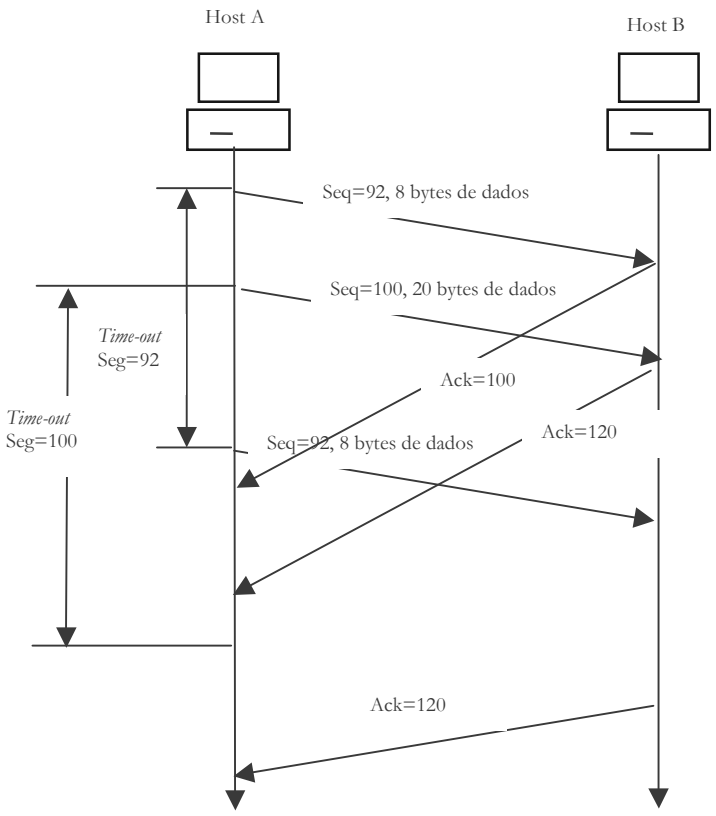


Figura 3.14. Segmento retransmitido porque seu reconhecimento chegou depois do time-out

No próximo cenário (Figura 3.15), como no caso anterior, o *host* A transmitiu ao *host* B um segmento com 8 bytes (número de seqüência 92) e em seguida mais um segmento com 20 bytes (número de seqüência 100). O *host* B recebeu estes segmentos e enviou números de reconhecimento (100 e 120 respectivamente). Todavia, o reconhecimento do segmento 92 (número de reconhecimento 100) se perdeu, o que não aconteceu com o segmento 100 (número de reconhecimento 120). Como o *host* A recebeu este último reconhecimento, ele sabe que o *host* B recebeu todos os segmentos por ele enviados, e está esperando agora segmentos com número de seqüência 120.

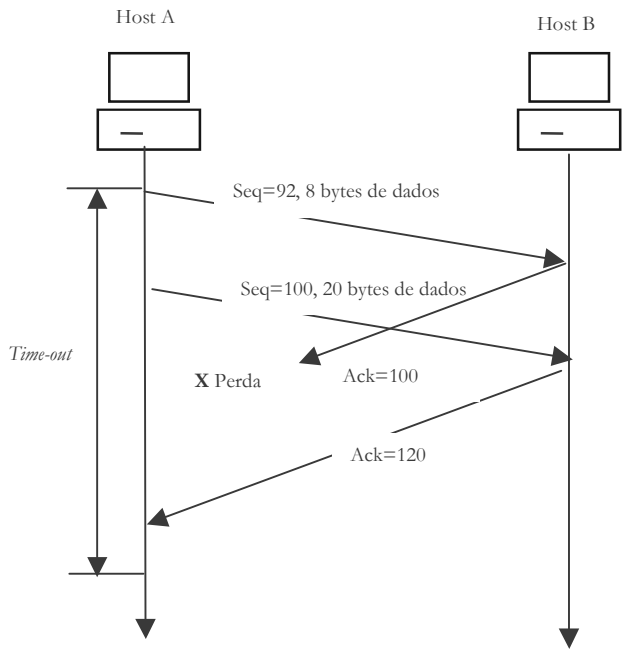


Figura 3.15. Reconhecimento cumulativo evita retransmissão do primeiro segmento

Gerenciamento de conexões no TCP

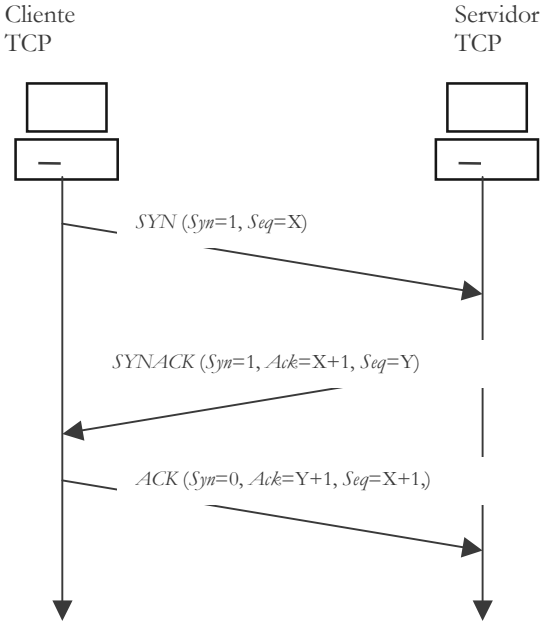
Para trocarem segmentos de dados utilizando o TCP o emissor e o receptor estabelecer uma "conexão" TCP através da troca de pacotes de controle entre si. Isto é chamado de **procedimento de estabelecimento de conexão** (*handshaking)*, onde se estabelecem os parâmetros para a comunicação. Uma vez concluído o *handshaking* a conexão é dita estabelecida e os dois sistemas terminais podem trocar dados.

ABERTURA DE CONEXÃO

Na fase de estabelecimento de conexão, são inicializadas as variáveis do protocolo TCP, como os números de seqüência e o tamanho de *buffers*. O processo cliente é o que inicia o estabelecimento da conexão sendo o servidor contatado pelo cliente.

O estabelecimento da conexão se dá em três passos (Figura 3.16):

1. O lado cliente do TCP envia um segmento de sincronização, chamado SYN (com o *flag* Syn setado em 1), ao lado servidor do TCP, especificando um número inicial de seqüência.
2. O servidor recebe o SYN, aloca *buffers* e inicializa variáveis, e envia



3.16. Três passos da abertura de conexão TCP

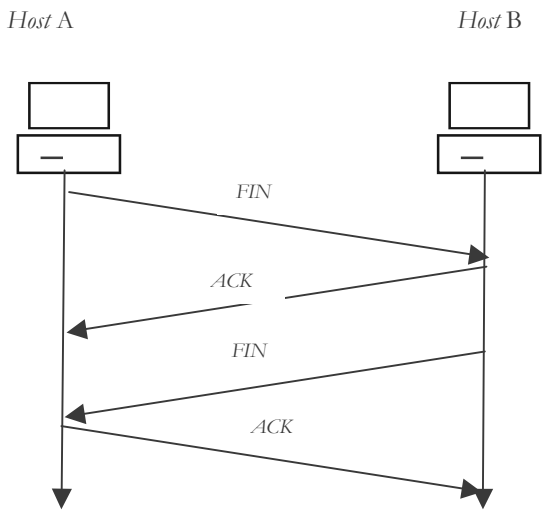
uma mensagem de aceite da conexão, chamada SYNACK (com o *flag* Syn e *flag* Ack setados em 1), onde reconhece o pedido de conexão e especifica seu número inicial de seqüência.

3. Uma vez recebido o aceite da conexão pelo servidor, o cliente confirma o recebimento com um segmento chamado ACK (*flag* Syn agora em 0 e *flag* Ack setado em 1 indicando um reconhecimento válido) e também aloca *buffers* e inicializa variáveis da conexão.

Uma vez que os três passos do estabelecimento da conexão forem completados, os *hosts* cliente e servidor podem trocar segmentos contendo dados entre eles.

ENCERRAMENTO DE CONEXÃO

Para o enceramento da conexão quatro segmentos são trocados (Figura 3.17). Quem inicia a desconexão envia de um segmento especial, chamado FIN (com *flag* Fin setado em 1). Quem recebe o segmento solicitando o fim da conexão, primeiro reconhece o segmento recebido e depois envia ele também um segmento FIN. O encerramento definitivo da conexão se dá quando o que iniciou a desconexão recebe e reconhece o segundo segmento FIN.



3.17. Encerramento de conexão TCP

Camada Rede

A **camada de transporte** provê um canal lógico **processo-a-processo** para as aplicações rodando em diferentes *hosts*. Para prover este serviço, a camada de transporte usa a **camada rede**, a qual provê um serviço de comunicação de **computador-a-computador** na inter-rede.

Papéis da camada rede:

- **Determinação da rota** que tomarão os datagramas desde o computador origem até o destino, a partir de **algorítmos de roteamento**.
- **Chaveamento de datagramas** chegando nos enlaces de entrada de cada roteador para a saída apropriada.

Protocolo IP (*Internet protocol*)

Na Internet a **camada rede** é implementada pelo **protocolo IP**, o qual oferece um **serviço de datagramas,** onde cada datagrama é tratado como uma unidade independente e não recebe nenhum tratamento de erros ou reconhecimento fim a fim. O datagrama permanece inalterado enquanto passa da origem ao destino.

Quando a camada de rede do lado de um emissor recebe um **segmento** da camada de transporte ela o encapsula em um **datagrama IP,** escreve o **endereço** do destino e outros campos do cabeçalho e envia ao primeiro roteador em direção ao *host* destino. Para que o datagrama atinga o destino, a camada rede envolve cada ***host*** e cada **roteador** no caminho entre a origem e o destino dos segmentos.

Os três principais componentes da camada rede da Internet são:

- **Protocolo IP**, que provê uma forma de **endereçamento**, **formato do datagrama** e convenções de empacotamento.
- **Protocolos de roteamento,** que permitem a determinação de rotas e elaboração de **tabelas de roteamento**. Os protocolos de roteamento mais conhecidos são o **RIP, OSPF** e **BGP**.
- **Protocolo ICMP,** utilizado para reportagem de erros e sinalização entre os roteadores.

Datagrama IP

Um **datagrama IP** é a unidade básica de transferência na Internet. O formato do datagrama apresenta um cabeçalho, que contém os **endereços IP da fonte** e do **destino**, além de outros campos, e uma área de **dados**. (Figura 3.18)

- O campo **versão** indica a versão do protocolo.
- O **comprimento do cabeçalho**, indica o comprimento do cabeçalho, em função dos

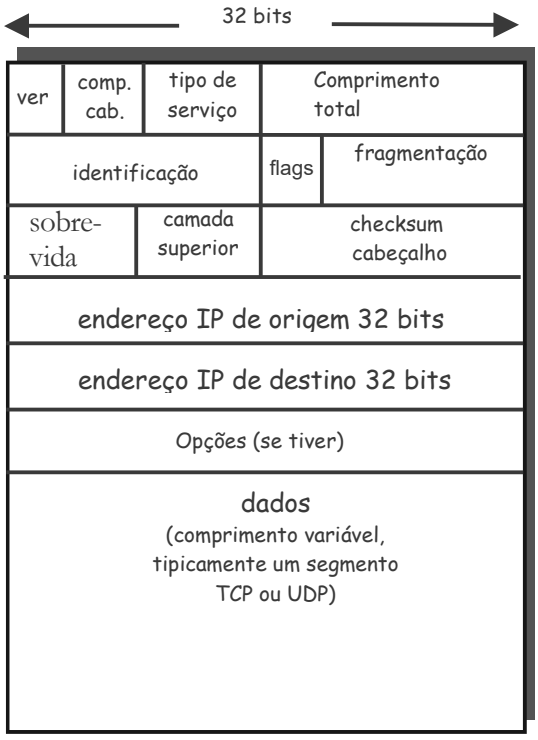


Figura 3.18. Formato do datagrama IP

campos opcionais, tipicamente o datagrama tem 20 bytes.

- O **tipo de serviço** permite diferenciar diferentes datagramas, como mensagens de controle (como ICMP) e dados normais (como mensagens HTTP), datagramas tempo-real (como aplicações de telefonia), etc.
- Os ***flags*** e **fragmentação** são usados em caso de fragmentação do datagrama IP.
- O **tempo de sobrevida**, **TTL** (*time-to-live*), indica o tempo de vida do datagrama, após o qual o mesmo é descartado.
- O **protocolo da camada superior** utilizado, como por exemplo TCP ou UDP.
- O ***checksum***, utilizado para detecção de erros no cabeçalho.
- O campo de **opções** é raramente usado.
- O campo de **dados**, que é a razão de ser do datagrama, e tipicamente carrega segmentos TCP ou UDP.

O **comprimento total** do datagrama, teoricamente poderia ser de 64K bytes (em função dos 16 bits do campo), todavia, na prática, nunca é maior que 1.500 bytes e freqüentemente é limitado em 576 bytes. Isto é feito para evitar a fragmentação do datagrama na rede física, já que o mesmo é encapsulado em um quadro da camada enlace e nem todas tem quadros de mesmo tamanho. No caso das redes locais Ethernet o tamanho do quadro é de 1.500 bytes e em outros enlaces é de 576 bytes. O tamanho máximo dos pacotes que podem ser transportados pela camada enlace é chamado de **MTU** (*maximum transfer unit*) (Figura 3.19).

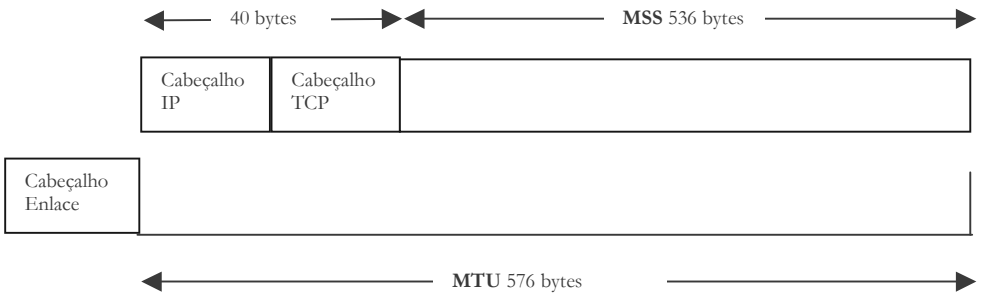


Figura 3.19. Valores práticos de MSS e MTU

Endereçamento IP

Endereço IP é um endereço lógico de **32 bits**, escrito em quatro octetos representados em decimal, cada um variando de O a 255. Os números são separados por pontos. Por exemplo, 193.32.216.9 seria um endereço válido, e sua notação em binário seria:

11000001 00100000 11011000 00001001.

Cada computador que esteja rodando o TCP/IP exige um endereço IP exclusivo. A exclusividade de endereço deve ser sempre mantida, mesmo ao se conectar a Internet.

Cada endereço IP engloba duas partes: o **identificador da rede** e o **identificador do *host***. O identificador da rede identifica a rede onde se encontram todos os *hosts* da mesma rede local. O identificador do *host* identifica um dispositivo em uma rede local, como um computador ou roteador. Por exemplo, a figura 3.20 ilustra três redes locais interconectadas por um roteador com

três interfaces. Olhando para os endereços IP atribuídos a cada computador e a cada interface do roteador, podemos notar, por exemplo, que os dispositivos conectados a rede local da esquerda e acima tem os endereços IP da forma 200.1.2.X. Isto é, compartilham os 24 bits mais à esquerda do endereço IP. No jargão IP, esta parte do endereço forma o identificador da rede. Os 8 bits restantes permitem identificar cada *host* da rede local. O endereço da rede local seria 200.1.2.0/24, onde a notação "/24" é também conhecida como **máscara de rede**, e indica que os 24 bits mais à esquerda dos 32 bits do IP identificam a rede.

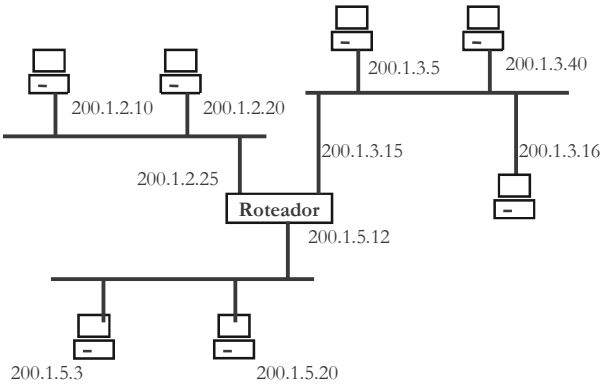


Figura 3.20. Endereçamento IP

Classes de endereçamento de IP

Para garantir endereços exclusivos em âmbito mundial, os endereços IP são licenciados a partir de uma localização central[1]. Quando foi criado, havia quatro classes básicas de licenças para endereços IP, cada uma especificando uma gama de endereços que podem ser atribuídos à licença (Figura 3.21). Na **classe A** os primeiros 8 bits identificavam a rede e os últimos 24 bits poderiam ser atribuídos aos *hosts* nesta rede, o que permitiria 2^{24} endereços. Na **classe B** o espaço de endereçamento para *hosts* seria de 2^{16} endereços. Já na **classe C,** a menor delas, deixaria 8 bits para serem atribuídos a *hosts*, ou 2^8 endereços. A **classe D** é reservada para endereços de *multicast*.

Estas classes de endereçamento não são mais utilizadas como parte formal dos da arquitetura de endereçamento IP, pois, com o crescimento do número de organizações de pequeno e médio porte o espaço de endereçamento ficou limitado. Por exemplo, uma rede classe C (/24) pode acomodar

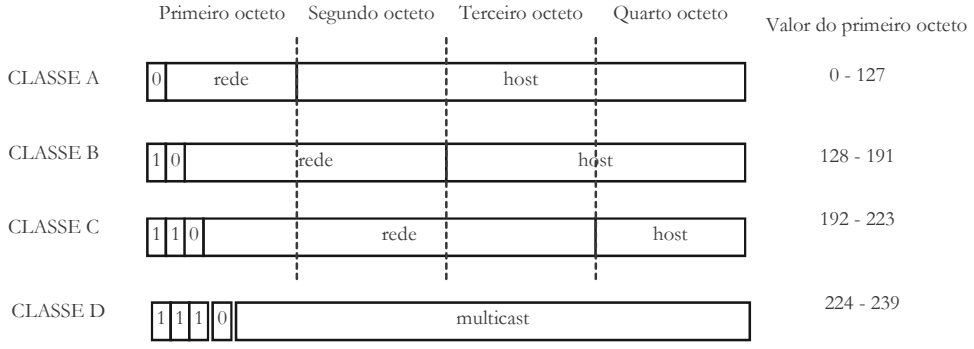


Figura 3.21. Classes de endereços IP

até 2^8 endereços, ou seja 256 *hosts*, o que pode ser muito pouco para muitas organizações. Já uma classe B (/16), poderia acomodar 2^{16} endereços, ou 64.634 endereços, o que seria demais para uma organização com, por exemplo, 2000 computadores.

Isto foi resolvido pelo IETF com a definição do padrão chamado **CIDR** (*classes interdomain routing*), que permite as organizações obterem um identificador de rede com qualquer tamanho. A notação utilizada pelo CIDR é **a.b.c.d/x**, onde o **x** é a **máscara de rede** que indica o número de bits reservados para a identificação da rede. Por exemplo, uma organização com 2000 computadores poderia solicitar um bloco de 2048 endereços, cuja notação seria a.b.c.d/21, e indica que os primeiros 21 bits identificam a rede e os 11 bits restantes (2^{11} =2048) caracterizam o espaço de endereçamento. No caso da nossa rede no CEFET em São José, licença é 200.135.233.0/24, a qual nos permite atribuir internamente até 256 endereços.

Alguns endereços IP têm utilização especial. Por convenção, um **endereço de rede** tem o campo identificador de *host* com todos os bits iguais a **0**. Podemos também nos referir a todos os *hosts* de uma rede através de um **endereço de difusão**, onde todos os bits são iguais a **1**. Um endereço com todos os 32 bits iguais a 1 é considerado um endereço de difusão para a rede do *host* origem do datagrama. O endereço 127.0.0.0 é reservado para teste (*loopback*) e comunicação entre processos da mesma máquina. Os endereços com o primeiro octeto entre 240 e 255 são reservados para uso futuro.

Roteamento

O **roteamento** inter-redes é a principal função do protocolo IP. O protocolo assume que um *host* é capaz de enviar datagramas a qualquer outro *host* conectado á mesma rede local. Caso o destinatário não esteja na mesma rede, parte da função de roteamento é transferida para os **roteadores** (*gateways*).

Os roteadores podem ser equipamentos específicos ou computadores normais que possuem mais de uma interface de rede. O roteamento no IP baseia-se exclusivamente no **identificador de rede** do endereço destino. Cada roteador possui uma tabela, chamada **tabela de roteamento**, cujas entradas são pares: **endereço de rede/endereço de roteador**. Por exemplo, quando um *host* deseja enviar um datagrama, inicialmente ele verifica se o destinatário está conectado a rede local. Se for o caso, ele entrega o datagrama a interface de rede que se encarrega de mapear o IP no endereço físico do *host* destino, encapsular o datagrama IP em um quadro da rede e transmiti-lo. Caso o *host* destino não se encontre na rede local, ele envia o datagrama ao **roteador padrão** (*gateway default*) da rede local. O roteador procura na sua **tabela de roteamento** o endereço do roteador que deve ser usado para alcançar a rede onde está conectado o destinatário do datagrama. O roteador encontrado pode não fazer parte da rede destino, mas, deve fazer parte do caminho a ser percorrido para alcançá-la.

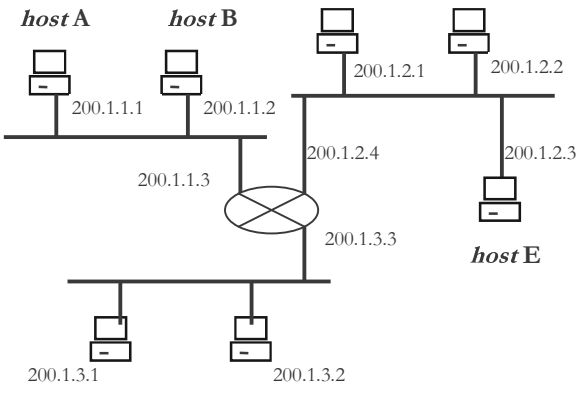


Figura 3.22. Redes e roteamento

Veja um exemplo de como funcionam as

[1] No Brasil o fornecimento de endereços IP é realizado pela FAPESP em São Paulo (www.fapesp.br).

tabelas de roteamento, considerando o contexto da rede apresentada na figura 3.22.

Suponha que o *host* A tenha a tabela de roteamento dada na tabela a seguir e deseja enviar um datagrama IP ao *host* B. Neste caso, o *host* A consulta sua tabela de roteamento e descobre que a rede 200.1.1.0/24 casa com o identificador da rede do *host* B. A tabela indica que o número de *hops* (número de enlaces a percorrer) é 1, o que quer dizer que está na mesma rede local. Então o *host* A passa o datagrama diretamente a camada enlace para proceder à entrega ao *host* B.

Tabela de roteamento do *host* A		
Rede destino	Próximo roteador	N. hops
200.1.1.0/24	-	1
200.1.2.0/24	200.1.1.3	2
200.1.3.0/24	200.1.1.3	2

Suponha agora o caso em que o *host* A queira enviar um datagrama ao *host* E, situado em outra rede, no caso a rede 200.1.2.0/24. Consultando sua tabela de roteamento ele verifica que o número de *hops* é 2, logo não está na mesma rede local, e que o acesso ao *host* E deve se dar através do roteador 200.1.1.3. Então ele passa o datagrama ao roteador para dar prosseguimento a entrega.

O roteador então consulta sua tabela de roteamento (veja tabela abaixo) e verifica que a rede 200.1.2.0/24 é acessível diretamente através da sua interface endereçada por 200.1.2.4. Sendo assim, ele entrega o datagrama a camada de enlace da rede 200.1.2.0/24 para fazer a entrega ao *host* E.

Tabela de roteamento do roteador			
Rede destino	Próximo roteador	N. hops	Interface
200.1.1.0/24	-	1	200.1.1.3
200.1.2.0/24	-	1	200.1.2.4
200.1.3.0/24	-	1	200.1.3.3

Protocolo de roteamento RIP

Na Internet, um **algoritmo de roteamento** ainda bastante utilizado é o **RIP** (*routing information protocol*) e apresenta **tabelas de roteamento** bastante parecidas com as do exemplo anterior.

As tabelas de roteamento RIP são construídas dinamicamente, baseadas em um algoritmo de roteamento que calcula as rotas tendo como base o número de enlaces a percorrer, escolhendo a rota que percorre o menor número de enlaces.

A partir do comando Unix `netstat -rn` pode-se visualizar as tabelas de roteamento RIP de um roteador Unix.

Parâmetros básicos para configuração do TCP/IP

Qualquer computador utilizando o TCP/IP possui três parâmetros básicos de configuração: **endereço IP**, **máscara de rede** e **roteador padrão**.

Endereço de IP

Endereço lógico exclusivo de 32 bits, escrito em quatro octetos representados em decimal.

Máscara de Rede

A máscara de rede é utilizada para "mascarar" uma parte do endereço IP para que se possa distinguir o identificador da rede do identificador do *host*. Quando dois *hosts* desejam se comunicar, a máscara da rede é utilizada para determinar se um *host* está localizado na rede local ou em uma rede remota.

Exemplos de máscara de rede:

Classe	N. de hosts	Bits usados para a máscara	Notação em decimal
/20	2^{12} = 4096	11111111 11111111 11110000 00000000	255.255.240.0
/21	2^{11} = 2048	11111111 11111111 11111000 00000000	255.255.248.0
/24	2^{8} = 256	11111111 11111111 11111111 00000000	255.255.255.0

Para se extrair o **identificador da rede** a partir do endereço IP completo, uma operação lógica **AND** é realizada com a máscara de rede.

Por exemplo, para descobrir o identificador de rede do *host* Joplin cujo endereço IP é 200.135.233.4 e cuja máscara de rede é 255.255.255.0, devemos fazer uma operação AND desdes dois valores:

```
        11001000 10000111 11101001 00000100
        11111111 11111111 11111111 00000000
AND     ___________________________________
        11001000 10000111 1110100100000000
```

o qual será igual a 200.135.233.0 (rede do CEFET em São José).

Roteador Padrão

Para comunicação com um *host* de uma outra rede, deve-se configurar um endereço IP para o **roteador padrão** (*defaut gateway*). O roteador padrão é o local para onde o TCP/IP envia pacotes destinados a redes remotas. Se um roteador padrão não for especificado, as comunicações se limitarão à rede local.

Exercício

A partir do Painel de Controle do Microsoft Windows, verifique a configuração do TCP/IP de seu computador.

Mapeamento do IP em um endereço físico da rede local

Protocolo ARP

Quando um *host* deseja enviar um datagrama a um destinatário conectado à sua rede local, ele entrega o datagrama a interface de rede para ela mapear o **endereço IP** no **endereço físico** (endereço de placa[2]) do *host* destino.

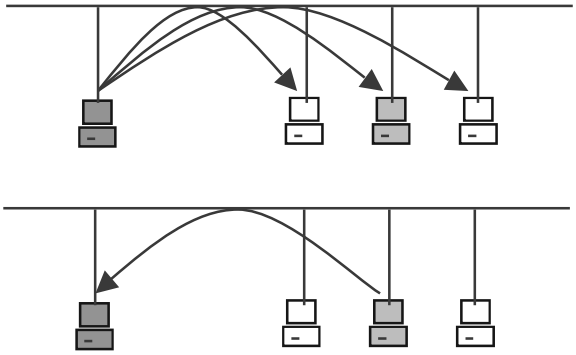


Figura 3.23. Protocolo ARP

O **protocolo ARP** permite encontrar o endereço físico a partir do endereço IP da máquina alvo. Para tal, o protocolo usa um mecanismo de difusão (*broadcast)*, enviando uma solicitação a todas as máquinas da rede, sendo que a máquina alvo responde indicando o par **endereço IP/endereço físico** (Figura 3.23).

Para melhorar a performance do protocolo, cada máquina possui uma memória (*cache)* com as últimas consultas realizadas, evitando múltiplos *broadcasts.* Ainda como refinamento, junto com o *broadcast,* a estação solicitante envia seu par endereço IP/endereço físico, permitindo que todas as máquinas da rede incluam este par em suas *caches* locais.

Quando um hardware é trocado, a máquina que sofreu a mudança se anuncia na rede com o novo par endereço IP/endereço físico, logo após sua entrada em operação.

Protocolo RARP

O **protocolo RARP** realiza a operação inversa do ARP, isto é, a partir de um **endereço físico** permite encontrar o **endereço IP** da máquina.

É geralmente utilizado por máquinas sem disco rígido (*disk-less)* para obter um endereço IP de um servidor. Para tal, um *host* RARP envia um *broadcast* com o seu endereço físico solicitando um endereço IP. A máquina autorizada a responder o pedido RARP envia a resposta.

Alternativas mais modernas ao protocolo RARP são o **BOOTP** e o **DHCP**, ambos construídos sobre protocolos de mais alto nível, como o IP e o UDP.

Protocolo BOOTP

O RARP é um protocolo de baixo nível, que exige um acesso direto ao *hardware* de rede para obter um IP.

Pelo fato de usar o UDP e o IP, o **BOOTP** pode ser implementado como um programa de aplicação. Além disto, é mais eficiente que o RARP, especificando vários itens necessários para a inicialização além do endereço de IP, como o endereço de um roteador ou de um servidor.

O BOOTP usa o UDP para carregar uma mensagem que é encapsulada em um datagrama IP. Para realizar o broadcast deste datagrama com a solicitação de um endereço IP, é utilizado o *broadcast* limitado na rede local (endereço IP: 255.255.255.255), mesmo antes de se saber qual o endereço IP da rede local ou do *host.*

---

[2] Por exemplo, as redes Ethernet possuem um endereço físico de 48 bits, gravados em memória Eprom pelo fabricante da placa.

Alocação dinâmica de IP

O **protocolo DHCP** (*dynamic host configuration protocol*) é uma extensão do protocolo BOOTP e permite a alocação dinâmica de endereços IP (o BOOTP é baseado em tabelas estáticas). Com o DHCP, um **servidor DHCP** recebe uma solicitação de um **cliente** e aloca dinamicamente um **endereço IP** em resposta ao pedido do cliente. Com o DHCP um computador cliente pode adquirir toda a configuração necessária em uma única mensagem (por exemplo, o endereço IP, máscara de rede, roteador padrão, servidor DNS, etc).

O **servidor DHCP** deve ser configurado com a **faixa de endereços IP** disponíveis para oferecer. Quando um computador se conecta na rede, ele solicita um endereço IP se apresentando com seu endereço físico. O servidor então escolhe um endereço IP dentro da faixa disponível e aloca ao solicitante.

Protocolo ICMP

Conforme já mencionado, o **protocolo IP** fornece um serviço de datagramas não confiável e não orientado a conexão, onde um datagrama segue de roteador em roteador até alcançar seu destino final. Se um roteador não consegue encontrar uma rota ou entregar um datagrama, ou se uma condição anormal é detectada, o roteador precisa informar a fonte original dos dados para que esta tome alguma ação ou corrija o problema. O **protocolo ICMP** (*Internet Control amd Message Protocol*) permite que os roteadores enviem **mensagens de erro e controle** a outros roteadores ou *hosts*; oferecendo uma comunicação entre a camada rede de uma máquina e a camada rede de outra máquina.

Tecnicamente o ICMP é um mecanismo de reportagem de erros. Ou seja, quando um datagrama causa um erro, o ICMP pode reportar a condição de erro de volta a fonte original do datagrama; a fonte então relata o erro para a aplicação ou realiza uma ação com vistas a corrigir o erro. Por exemplo, quando rodando uma aplicação Telnet ou HTTP, podemos encontrar mensagens como "rede destino não encontrada" (*destination network unreachable*), que tem origem no protocolo ICMP.

O ICMP é normalmente considerado como parte do IP, todavia está situado logo acima. As mensagens ICMP são carregadas na porção de dados de um datagrama IP, que as identifica como tipo ICMP. Os datagramas contendo as mensagens ICMP seguem de volta, seguindo exatamente o caminho que tomaram os dados do usuário, podendo elas também serem perdidas ou corrompidas.

Formato das Mensagens ICMP

Cada mensagem ICMP tem um campo de **tipo** e um campo de **código**, e também contém os primeiros 8 bytes do datagrama que causou o erro (com isto o emissor pode determinar o pacote que causou o erro).

Algumas mensagens ICMP:

ICMP Tipo	Código	Descrição
0	0	*echo reply (*para o Ping*)*
3	0	*destination network unreachchable*
3	1	*destination host unreachchable*
3	6	*destination network unknow*
3	7	*destination host unknow*
8	0	*echo request*
11	0	*TTL* (*time to live*) *expire*

Nem todas as mensagens ICMP são de reportagem de erros. A aplicação Ping, por exemplo, utiliza mensagens ICMP *Echo Request* e *Echo Reply* para verificar se um host está disponível e sua respectiva resposta.

O Traceroute, que é capaz de traçar a rota que liga um *host* a outro *host*, também usa mensagens ICMP. Para determinar o nome e o endereço dos roteadores entre a fonte e o destino, o Traceroute na fonte envia uma série de datagrama IP ordinários ao destino. O primeiro datagrama tem o TTL igual a 1, o segundo 2, o terceiro 3, e assim por diante, e inicia temporizadores para cada datagrama. Quando o enésimo datagrama chega ao enésimo roteador, este verifica que o tempo de sobrevida do datagrama acaba de terminar. Pelas regras do IP, o datagrama é então descartado e uma mensagem ICMP de advertência é enviada a fonte (tipo 11 código 0), com o nome do roteador e seu endereço IP. Quando a resposta chega de volta a fonte, a mesma calcula o tempo de viagem em função dos temporizadores.

Sistema de Nomes de Domínio

Um nome de domínio é um nome hierárquico implementado com a utilização de um **Sistema de Nomes de Domínio** (**DNS** *domain name system*).

O **DNS** proporciona um banco de dados *on-line* e distribuído para resolver nomes de domínios a seus endereços IP correspondentes. Isto facilita na medida em que não precisamos mais memorizar endereços IP, mas sim **nomes de domínio**, muito mais fáceis de serem lembrados e ao mesmo tempo identificados com o proprietário do domínio.

Se uma organização deseja participar da Internet, deve registrar o seu nome de domínio no Centro de Informações de Rede.

Principais nomes de domínio Internet

Nome de Domínio	Significado
edu	Instituição educacional
com	Organização comercial
gov	Instituição governamental
org	Organização não governamental
<código de país>	Cada país (esquema geográfico)

Exemplos:

ufsc.br cefetesc.edu.br mec.gov.br matrix.com.br
mit.edu national.com (nos USA não há sigla de país)

Além da sintaxe para os nomes, o esquema DNS inclui um sistema distribuído eficiente, seguro e de propósito geral para se mapear nomes em endereços.

O DNS consiste da união de sistemas cooperativos independentes, chamados **servidores de nomes**, que fazem a translação do **nome de domínio** em **endereço IP**. O software cliente, chamado **resolvedor de nomes**, usa um ou mais servidores de nomes para traduzir um nome.

Resolução de Nomes

A resolução de nomes esta baseada em uma árvore hierárquica de nomes. Conceitualmente a resolução inicia de cima para baixo (*top-down)*, começando no servidor raiz e seguindo para os servidores localizados nos ramos da árvore.

Há dois modos possíveis para um servidor resolver um nome: **resolução interativa** (passo-a-passo) ou **resolução recursiva**.

Em ambos os casos, o servidor consultado verifica se o nome solicitado pertence a um sub-domínio seu. Se for o caso, traduz o nome ao endereço de acordo com sua base de dados. Se não puder resolver o nome completamente, verifica o tipo de solicitação feita pelo cliente. Se o cliente solicitou busca recursiva o servidor contata um DNS que possa resolver o nome e devolve a resposta ao cliente. Caso a solicitação foi do tipo interativa, ele fornece o nome de um DNS ao cliente e não a resposta da resolução completa do nome.

Para iniciar a busca, um cliente precisa saber como conectar pelo menos um servidor de nomes raiz. Em adição, um servidor de nomes deve saber o endereço de um servidor de nomes de domínio imediatamente superior (servidor pai).

Como a maioria das consultas é de âmbito local, a eficiência do sistema é aumentada iniciando-se a busca em um servidor de nomes local. Além disto, os servidores de nomes da Internet usam memória *cache* para otimizar os custos da busca de nomes não locais. Todos os nomes recentemente usados são armazenados na sua memória *cache*, bem como a informação de como foram obtidos. Como a informação em memória pode estar desatualizada, o servidor de nomes marca como não autoritativa (*non authoritative)*, podendo o cliente contatar a autoridade para ver se o nome ainda é válido.

Questões

1. Qual o papel dos protocolos da **camada de transporte** da Internet?
2. Explique a relação entre os protocolos da **camada transporte** e da **camada rede** da Internet.
3. Em que consiste o serviço de **multiplexação** de aplicações oferecido pelos protocolos de transporte TCP e UDP.
4. Qual informação é utilizada por um processo que está executando em um computador, para identificar um processo que está executando em outro computador remoto.
5. Pesquise na Internet a lista completa das **portas** TCP e UDP reservadas para aplicações específicas. Ache um endereço URL com esta informação.
6. Suponha que você está desenvolvendo uma aplicação para a Internet. Que tipo de protocolo de transporte você utilizaria, **TCP** ou **UDP**? Explique, tendo como base à aplicação que será desenvolvida
7. Quais são os princípios utilizados pelos **protocolos de transporte confiável**, como o TCP, para garantir que os dados transmitidos são livres de erros?
8. Diferencie os objetivos dos serviços de **controle de fluxo** e de **controle de congestionamento**, presentes no protocolo de transporte TCP.
9. Para que serve e como funciona o mecanismo de ***chesksum*** utilizado pelo TCP, UDP e IP? Cite um exemplo prático.
10. Para que servem os **números de seqüência** e **reconhecimento** presentes no cabeçalho do segmento TCP? Explique o processo utilizado para numerar os segmentos.