**Camada de Transporte**

* **Transporte de Dados**

Os aplicativos e programas da camada de aplicação tratam os dados que são trafegados entre os hosts de origem e de destino. A camada de transporte realiza a comunicação lógica entre aplicativos executados em hosts diferentes. Isso inclui serviços, tais como o estabelecimento da sessão temporária entre dois hosts e a transmissão confiável de informações para um aplicativo. A camada de transporte não conhece o tipo de host de destino, o tipo de meio pelo qual os dados viajam, o caminho percorrido pelos dados, o congestionamento em um enlace ou o tamanho da rede.

*Rastreio de Conversas Individuais:*

Na camada de transporte um conjunto de dados que flui entre um aplicativo na origem e um aplicativo no destino é conhecido como uma conversação e é rastreado separadamente. A camada de transporte é responsável por manter e rastrear essas diversas conversações. Em um host pode haver vários aplicativos que se comunicam simultaneamente pela rede. Grande parte das redes tem uma limitação na quantidade de dados que são transportados em um único pacote. Por isso os dados são divididos em partes gerenciáveis.

*Segmentação de Dados:*

Uma das responsabilidades da camada de transporte é dividir os dados da aplicação em blocos de tamanhos apropriados. Depende do protocolo utilizado na camada de transporte como tais blocos são chamados: segmentos (TCP) ou datagramas (UDP). A camada de transporte divide os dados em blocos menores (segmentos ou datagramas) que facilitam o gerenciamento e o transporte.

*Adição de Informação no cabeçalho:*

O protocolo da camada de transporte também adiciona cabeçalho que contém informações organizadas em vários campos para cada bloco de dados. São os valores desses campos que possibilitam que vários protocolos da camada de transporte executem diferentes funções no gerenciamento da comunicação. A camada de transporte garante que vários aplicativos em execução em um dispositivo recebam os dados corretos. Os segmentos enviados têm cabeçalhos de camada de transporte com informações de controle.

*Identificação de Aplicações:*

A camada de transporte consegue separar e gerenciar várias comunicações com necessidades diferentes de requisitos de transporte. Para passar os fluxos de dados para os aplicativos adequados, a camada de transporte identifica o aplicativo de destino ao usar um identificador denominado número de porta. Cada processo de software que necessita acessar a rede recebe um número de porta exclusivo para o host.

*Multiplexação:*

O envio de certos tipos de dados, por exemplo, um streaming de vídeo através de uma rede como um fluxo de comunicação completo pode consumir toda a banda disponível. Tal fato impediria que outras comunicações ocorressem ao mesmo tempo. A camada de transporte usa segmentação e multiplexação que permitem que diferentes conversações se intercalem na mesma rede. A verificação de erros pode ser feita nos dados do segmento para saber se o segmento foi alterado durante a transmissão.

* **TCP**

O protocolo TCP da camada de transporte é confiável e completo que garante que os dados cheguem ao destino.  O TCP possui campos que garantem a entrega dos dados do aplicativo. Tais campos implicam em processamento adicional pelos hosts de envio e de recepção. O TCP provê confiabilidade e controle de fluxo. O TCP primeiro estabelece uma conexão entre o remetente (origem) e o destinatário para manter o estado de uma conversação e rastrear as informações. Desta forma, o TCP é um protocolo orientado a conexão.

*Funcionalidades:*

O TCP é orientado a conexão, pois negocia e estabelece uma conexão permanente (ou sessão) entre os dispositivos de origem e de destino antes de encaminhar qualquer tráfego. Através do estabelecimento da sessão os dispositivos negociam a quantidade de tráfego que pode ser encaminhada em um certo momento e os dados de comunicação entre os dois são gerenciados.

*Garantia de Entrega Confiável:*

Por várias razões é possível que um segmento seja corrompido ou perdido completamente ao ser transmitido pela rede. O TCP garante que cada segmento enviado chegue ao destino.

*Fornecimento de Entrega Ordenada:*

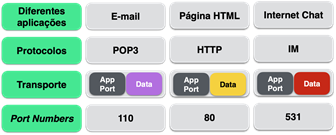
Como as redes podem fornecer várias rotas com taxas de transmissão diferentes os dados podem chegar fora de ordem. Ao numerar e sequenciar os segmentos o TCP garante que os segmentos sejam remontados na ordem correta.

*Suporte a controle de fluxo:*

Os hosts têm recursos limitados (por exemplo, memória e capacidade de processamento). Quando o TCP percebe que esses recursos estão sobrecarregados pode solicitar que o aplicativo reduza a taxa de fluxo de dados de envio. Isso é feito pelo TCP que regula a quantidade de dados que a fonte transmite. Desta forma o controle de fluxo evita a necessidade de retransmissão de dados quando os recursos do host receptor estão sobrecarregados.

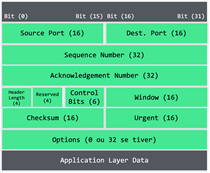
*Portas:*

A Figura a seguir mostra exemplos de número das portas associadas aos protocolos da camada de aplicação.

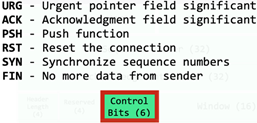


*Header (Cabeçalho):*

A Figura a seguir ilustra o cabeçalho do TCP.



A Figura a seguir mostra os bits de controle.



* **UDP**

O UDP é outro protocolo de camada de transporte. Os datagramas UDP são processados ​​mais rapidamente. O UDP é sem conexão e não provê confiabilidade ou controle de fluxo.  Não requer uma conexão pré-estabelecida. Como o UDP não rastreia as informações enviadas (ou recebidas) entre o cliente e o servidor é um protocolo sem estado (stateless). O UDP também é um protocolo de entrega de melhor esforço, pois não há confirmação de que os dados foram recebidos no destino.

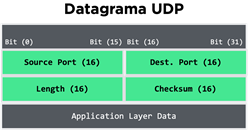
*Funcionalidades:*

O UDP é um protocolo de melhor esforço. É um protocolo leve que oferece a mesma segmentação e remontagem de dados que o TCP, mas sem confiabilidade e controle de fluxo do TCP. O UDP é tão simples que é descrito pelo que não faz em comparação ao TCP. Os recursos UDP incluem:

* Na ordem em que os dados são recebidos são reconstruídos.
* Não reenvia quaisquer segmentos perdidos.
* Não há estabelecimento de sessão.
* A transmissão (envio) não é informada sobre a disponibilidade do recurso.

*Header*(Cabeçalho):

A Figura a seguir ilustra o cabeçalho UDP.



*Aplicações:*

Há aplicativos adequados para UDP:

* Vídeo e aplicativos de multimídia: toleram alguma perda de dados, mas exigem pouco atraso. Exemplos,  VoIP e streaming de vídeo (ao vivo, tempo real).
* Aplicativos de solicitação e resposta simples: transações simples em que um host envia uma solicitação e pode (ou não) receber uma resposta. Exemplos, DNS e DHCP.
* Aplicativos que lidam com confiabilidade por si só: comunicação unidirecional em que o controle de fluxo, detecção de erros, confirmações e recuperação de erros não são necessários ou podem ser gerenciados pelo aplicativo. Exemplos, SNMP e TFTP.

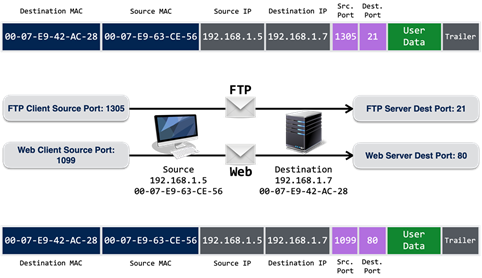
* **Processo de comunicação TCP e UDP**

*Socket:*

As portas de origem e destino são colocadas no segmento. Os segmentos são então encapsulados dentro de um pacote IP que, por sua vez, contém os endereços IP (origem e destino). A combinação IP de origem e do número da porta de origem, ou o IP de destino e o número da porta de destino é conhecido como um socket. Exemplo: 192.168.1.7:80

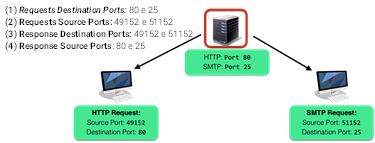
*Socket Pairs:*

A Figura a seguir mostra  exemplos de pares de soquetes.



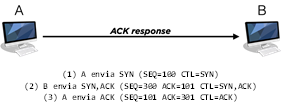
*TCP Port Allocation:*

A Figura a seguir ilustra exemplos de requisições e respostas ao servidor TCP.



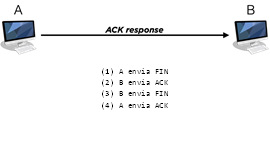
*TCP: Estabelecendo:*

A Figura a seguir mostra as mensagens de estabelecimento da conexão TCP.



*TCP: Terminando:*

A Figura a seguir mostra  as mensagens de encerramento da conexão TCP.



* **Controle de Fluxo e Confiabilidade**

O TCP reenvia os pacotes perdidos e numera os pacotes. O TCP também mantém o fluxo de pacotes para que os dispositivos não sejam sobrecarregados. Pode ocorrer que os segmentos TCP não cheguem ao seu destino. Às vezes, os segmentos TCP chegam fora de ordem. Para que a mensagem original seja compreendida pelo destinatário todos os segmentos devem ser recebidos e remontados corretamente. Para isso os números de sequência são atribuídos no cabeçalho de cada pacote. Na configuração da sessão um número de sequência inicial (ISN – Initial Sequence Number) é definido e é o valor inicial dos bytes que são transmitidos ao aplicativo de recepção. Durante a sessão, conforme os dados são transmitidos, o número da sequência é incrementado pelo número de bytes que foram transmitidos. Tal rastreamento de bytes de dados possibilita que cada segmento seja identificado e reconhecido de forma única. Os segmentos ausentes são identificados. O processo de recebimento TCP coloca os dados de um segmento em um buffer de recepção. Assim, os segmentos são colocados na ordem de sequência correta e, quando remontados, passados para a camada de aplicativo. Os segmentos que chegam com números de sequência fora de ordem são retidos para posterior processamento.

*Perda de Dados:*

Pode ocorrer perda de dados em uma rede. O TCP fornece métodos de gerenciamento dessas perdas de segmento. Por exemplo, um mecanismo de retransmissão dos segmentos não confirmados. O número de sequência (SEQ) e o número de confirmação (ACK) são usados ​​juntos para confirmar o recebimento dos bytes de dados contidos nos segmentos transmitidos. O número SEQ identifica o primeiro byte de dados no segmento transmitido. O TCP usa o número ACK enviado de volta à fonte transmissora para indicar o próximo byte que o receptor espera receber.

*Window Size (Tamanho da janela):*

O TCP provê também mecanismos para controle de fluxo. É a quantidade de dados que o destino pode receber e processar de forma confiável. Para tal, o cabeçalho TCP possui um campo de 16 bits chamado window size (tamanho da janela) que define o número de bytes que podem ser enviados antes de esperar uma confirmação. A confirmação corresponde ao número do próximo byte esperado. O tamanho da janela inicial é definido quando a sessão TCP é estabelecida durante o handshake (troca de mensagens no estabelecimento da conexão). O dispositivo de origem limita o número de bytes enviados ao dispositivo de destino baseado no tamanho da janela de destino. Depois que o dispositivo de origem receber uma confirmação de que os bytes foram recebidos é que pode continuar o envio de mais dados para a sessão.

*MSS (Maximum Segment Size)  e Controle de Congestionamento:*

O MSS é o tamanho máximo do segmento que o dispositivo de destino pode receber. Faz parte do campo de opções no cabeçalho TCP que determina a maior quantidade de dados que um dispositivo pode receber em um único segmento TCP. Na ocorrência de congestionamento em uma rede os pacotes são descartados pelo roteador (devido à sobrecarga). Quando os pacotes (que contém segmentos TCP) não alcançam seu destino não são reconhecidos. Ao determinar a taxa na qual os segmentos TCP são enviados, mas não confirmados, a fonte (transmissão) pode assumir um certo nível de congestionamento da rede. Se a fonte determinar que os segmentos TCP não são confirmados ou não são confirmados em tempo hábil pode reduzir o número de bytes que envia antes de receber uma confirmação.