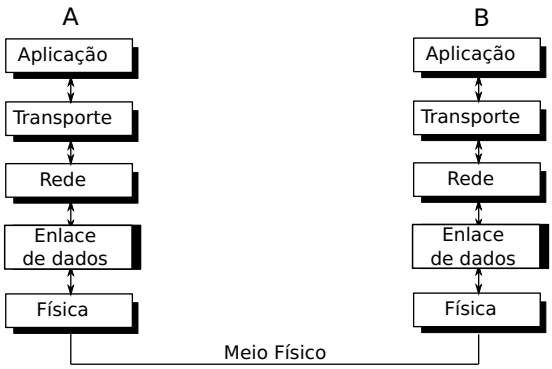
INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA RIO GRANDE DO NORTE

Professor: Macêdo Firmino
Disciplina: Arquitetura de Rede
Aula 15: Camada de Transporte e Protocolo TCP

Na aula de hoje iremos compreender o funcionamento do protocolo TCP e suas principais características, diferenciar o TCP do UDP em termos de confiabilidade e controle de conexão, identificar as etapas do three-way handshake e do encerramento de conexão e utilizar ferramentas de rede para observar conexões TCP em funcionamento. Vamos lá?? Preparados??

Camada de Transporte

A camada de transporte é responsável pela comunicação entre processos finais (programas) entre diferentes equipamentos. Embora a camada de rede gerencie a entrega de pacotes individuais da origem até seu destino, ela trata cada pacote de forma independente, como se cada um deles pertencessem a uma mensagem distinta. Desta forma, a camada de transporte poderá garantir a integridade e a ordem de entrega dos pacotes, controlar erros na transmissão, bem como o fluxo de dados.



As principais funções desta camada são:

- Endereçamento por Portas: permite que várias aplicações utilizem a rede simultaneamente, associando programas a diferentes portas de origem e destino
- Controle do Enlace (opcional): permite estabelecer uma conectividade (conversa) fim-a-fim entre aplicações.
- Controle de Erros (opcional): realiza um controle de erro fim a fim. Assegura que toda a mensagem chegue ao destino final livre de erros. A correção de erros normalmente se faz através de um pedido de retransmissão.
- Segmentação e Reagrupamento (opcional): permite dividir uma mensagem em vários segmentos de tamanhos variáveis, onde cada segmento contém um número deidentificação. Com este número é possivel o receptor remontar, identificar e/ou substituir pacotes extraviados;
- Controle de Fluxo (opcional): realiza um controle de fluxo fim a fim (entre programas). Ela garante que o remetente não envie dados mais rápido do que o receptor pode processar, evitando sobrecarga e perda de pacotes.;

Mais detalhes sobre a camada de transporte reveja a aula 14.

Protocolo TCP

O TCP (Transmission Control Protocol) é um dos pilares fundamentais da pilha de protocolos TCP/IP, que constitui a base da Internet moderna. Ele é o protocolo de transporte que implementa todas as funcionalidades. Ele permite a comunicação entre processos finais (endereçamento por portas), cria uma conexão virtual entre dois processos (controle de conexão), realiza segmentação, e implementa mecanismos de controle de fluxo e de erros.

Sua função é garantir uma comunicação confiável, ordenada e livre de erros entre aplicações que trocam dados pela rede. Ele é amplamente utilizado em serviços que exigem entrega garantida de mensagens, tais como navegação web (HTTP/HTTPS), transferência de arquivos (FTP), envio de e-mails (SMTP/IMAP/POP3) e conexões remotas (SSH, Telnet).

Embora o TCP e o UDP operem na mesma camada (transporte), suas filosofias de funcionamento são distintas. Enquanto o TCP prioriza a confiabilidade, o UDP privilegia a velocidade e a simplicidade. Observe algumas diferenças:

Com relação a aplicações, o TCP é usado quando a confiabilidade é fundamental, como em transações financeiras, e-mails ou transferência de arquivos. Enquanto o UDP é usado quando a velocidade é mais importante, como em transmissões de vídeo em tempo real, chamadas VoIP ou jogos online, onde pequenas perdas são toleráveis.

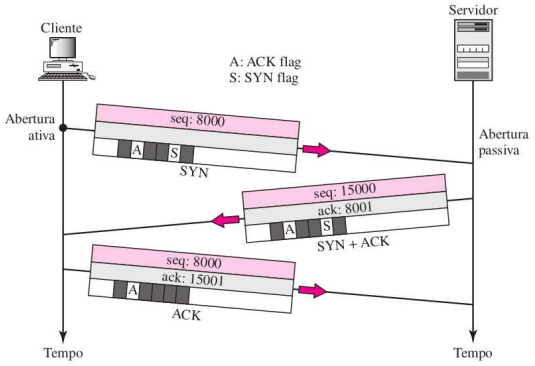
Aspecto	TCP	UDP
Orientado à Conexão	Sim	Não
Confiabilidade	Alta	Baixa
Controle de Fluxo	Sim	Não
Controle de Congestionamento	Sim	Não
Reordenação de Pacotes	Sim	Não
Velocidade	Lento	Rápido
Retransmissão de Pacotes Perdidos	Sim	Não

Controle de Enlace

O TCP é um protocolo orientado à conexão (connection-oriented). Isso significa que, antes que qualquer dado seja transmitido, é necessário estabelecer uma conexão lógica entre os dois dispositivos - geralmente chamada de cliente e servidor. Essa conexão garante que ambos estejam prontos para trocar dados e que a comunicação ocorra de maneira segura e controlada.

Estabelecimento de Conexão

O processo de inicialização de uma conexão TCP é conhecido como "three-way handshake" (aperto de mão em três vias), e ocorre em três etapas fundamentais:



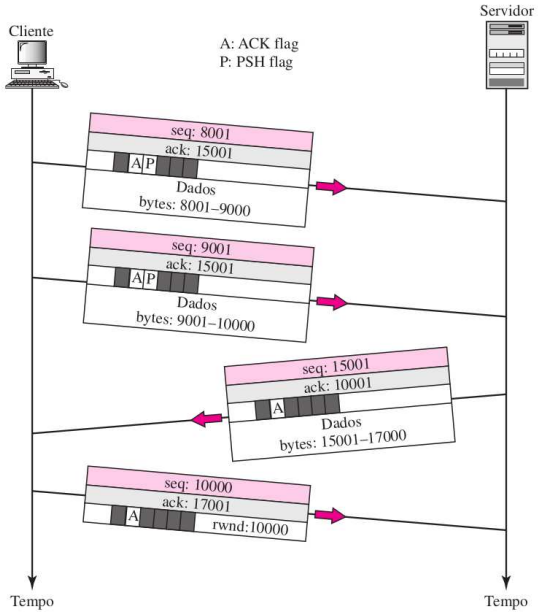
01. SYN (Synchronize): o cliente envia um segmento TCP com o bit SYN ativado, solicitando a abertura de uma conexão e informando também seu o número inicial de sequência.

02. SYN-ACK (Synchronize + Acknowledge): O servidor responde confirmando o recebimento (ACK) e envia também o seu número de sequência inicial.

03. ACK (Acknowledge): o cliente confirma o recebimento da resposta do servidor, estabelecendo assim o canal de comunicação bidirecional.

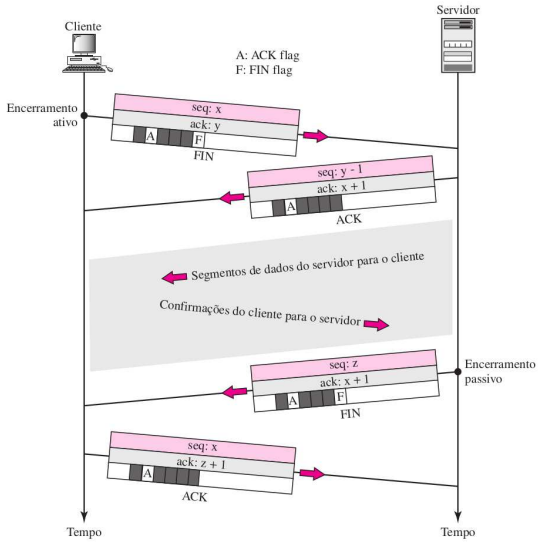
Transmissão dos Dados

Após essas três etapas, ambos os lados sabem que a comunicação está ativa e pronta para o envio de dados. Por exemplo, mostrado na figura abaixo, o cliente transmite 2.000 *bytes* de dados em dois segmentos TCP, enquanto que o servidor envia 2.000 *bytes* em um único segmento. Os segmentos com o *flag* PSH setado informa ao TCP que ele deve entregar os dados para o processo servidor imediatamente.



Encerramento de Conexão

O encerramento da sessão TCP também é controlado e seguro. Para isso,ele utiliza quatro etapas, conhecidas como four-way handshake:



01. O lado que deseja encerrar a comunicação envia um segmento com a flag FIN (Finish).

02. O receptor responde com um ACK, confirmando o recebimento. Quando FIN é confirmado, esse sentido é desativado para novos dados.

03. Quando o receptor também deseja encerrar, envia um FIN.

04. O primeiro lado responde com outro ACK, completando o encerramento.

Se uma resposta para um FIN não chegar no período equivalente a duas vezes o tempo máximo de duração de um pacote, o transmissor do FIN encerrará a conexão.

Controle de Erro

O controle de erro no TCP tem como objetivo detectar, corrigir e evitar a perda de dados durante a transmissão. Para isso, o TCP utiliza três mecanismos principais:

Checksum (Soma de Verificação)

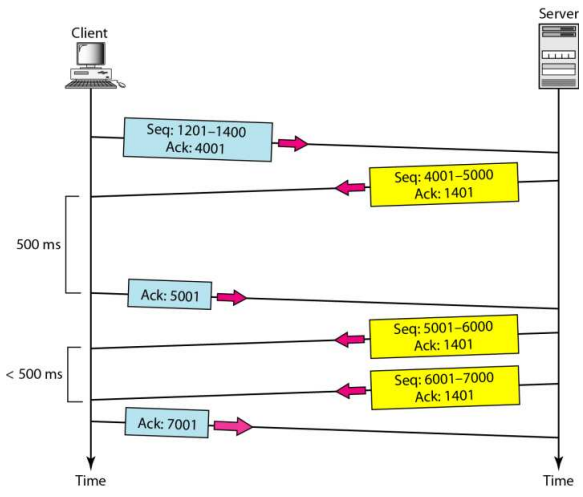
Cada segmento TCP inclui um campo chamado checksum, que é uma soma de verificação calculada a partir do conteúdo do cabeçalho e dos dados do segmento. Quando o emissor envia o segmento, ele calcula o valor do checksum e o insere no campo apropriado do cabeçalho. Ao receber o segmento, o receptor realiza o mesmo cálculo.

Se o resultado for igual ao valor enviado, o segmento é considerado íntegro e é aceito. Se for diferente, significa que houve erro durante a transmissão, e o segmento é descartado. O checksum já foi visto na aula de camada de enlace.

Numeração de Confirmação (ACK)

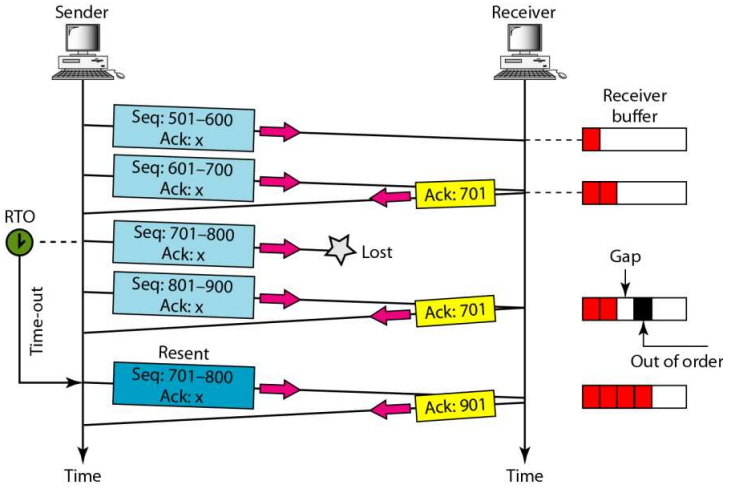
O TCP divide os dados em segmentos e atribui um número de sequência a cada byte transmitido. O dispositivo receptor utiliza mensagens de confirmação (ACKs) para informar ao emissor quais bytes foram recebidos corretamente.

Por exemplo, se o receptor envia um ACK com o número 501, significa que todos os bytes até o número 500 foram recebidos com sucesso e o próximo esperado é o 501. Caso o emissor não receba o ACK dentro de um determinado tempo, ele presume que o segmento foi perdido ou danificado e realiza uma retransmissão.



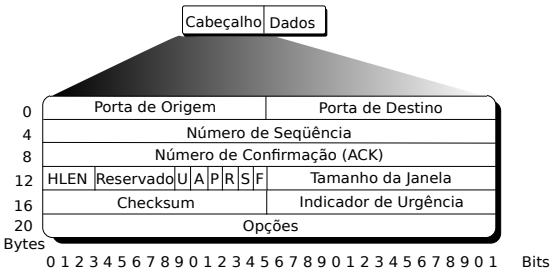
Retransmissão e Controle de Tempo (Timeout)

O TCP utiliza temporizadores (timers) para acompanhar o tempo entre o envio de um segmento e o recebimento do respectivo ACK. Se o ACK não for recebido dentro do período de tempo esperado (chamado de timeout), o TCP retransmite o segmento.



Cabeçalho

Um segmento TCP é formado por um cabeçalho de 20 a 60 bytes, seguido pelos dados do programa aplicativo. O cabeçalho tem 20 bytes quando não existem informações opcionais sendo transmitidas (até 60 bytes). O formato de um segmento TCP é:



Os campos são:

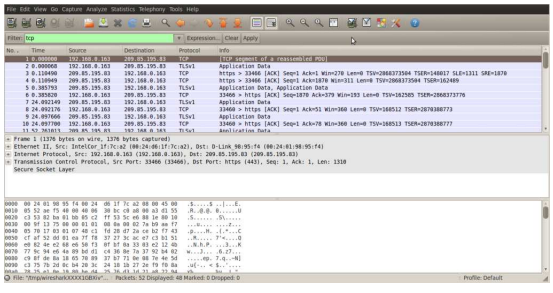
- Endereço da porta de origem: identifica o número da porta do programa de aplicação no host que está enviando o segmento (origem).
- Endereço da porta de destino: identifica o número da porta do programa de aplicação no host que está recebendo o segmento (destino).
- Número de sequência: identifica o número atribuído ao primeiro byte de dados de um segmento. Para garantir confiabilidade na entrega, cada byte a ser transmitido é numerado. O número de sequência informa ao destino como poderá reagrupar os bytes recebidos.
- Número de confirmação: identifica o número de bytes que o receptor espera receber da outra parte.

- Comprimento do cabeçalho: identifica a quantidade de palavras de 4 bytes de um cabeçalho TCP.
- URG: o valor do campo de Urgent Point é válido.
- ACK: o valor do campo de confirmação é válido.
- PSH: empurra os dados, passar os dados, caso estejam em ordem, para o programa sem armazenar em memória.
- RST: reinicia a conexão.
- SYN: sincroniza números de sequência durante o estabelecimento de uma conexão.
- FIN: encerra a conexão.
- Tamanho da janela: de?ne o tamanho de uma janela TCP, em bytes, que a outra parte deve manter em memória.
- Checksum: calculo para detecção de erro. No TCP é obrigatória.
- Urgent Pointer: é válido apenas se o flag URG (Urgente) es tiver ativo, é usado quando um segmento contém dados urgentes e que precisa de prioridade.
- Opções: conter um total de até 40 bytes de informações opcionais que serão inclusas ao cabeçalho TCP.

Wireshark

O Wireshark é um programa (conhecido como *sniffer*) que verifica os pacotes transmitidos pelo dispositivo de comunicação (placa de rede, placa de fax modem, etc.) do computador. O programa analisa o tráfego de entrada e saída e organiza-os por protocolo. Já utilizamos em outras aulas desta disciplina.

Para selecionar os pacotes baseados no protocolo, basta digitar o nome do protocolo no qual você está interessado no campo *Filter* (Filtro) e pressione o botão "Apply" (aplicar) para iniciar o filtro. A Figura abaixo mostra um exemplo de filtragem sobre o protocolo TCP.



Atividade

Agora iremos realizar uma Análise do protocolo TCP. Em dupla, realize as atividades abaixo e depois coloque as respostas no Google Sala de Aula.

01. Começe a captura de tráfego na rede;

02. Utilize a Internet para acessar o site `www.ifrn.edu.br`);

03. No Wireshark, use o filtro tcp para isolar apenas os pacotes TCP.

```
tcp
```

04. Quais os campos existentes no cabeçalho de um segmento TCP (*Transmission Control Protocol*)?

05. Apresente alguns valores de segmentos TCP que você capturou: Source Port, Destination Port, Sequence Number, Acknowledgment Number e Windows. Explique quais as funções destes parâmetros.

06. Identifique os pacotes com flags SYN, SYN-ACK e ACK. Quais portas TCP foram utilizadas na comunicação observada? Dê um print e coloque no arquivo de texto.

07. Qual a importância do handshake de três vias (three-way handshake) para a confiabilidade do TCP?

Atividade Extra (Ponto Extra)

Vocês irão trabalhar em duplas para implementar uma comunicação utilizando o protocolo TCP em Python. Em seguida, deverão capturar e analisar os pacotes trocados durante essa comunicação usando o Wireshark, observando o estabelecimento da conexão (handshake), o envio das mensagens e o encerramento da sessão.

Formem duplas: um aluno atuará como Servidor e o outro como Cliente. O programa será um chat simples, da seguinte forma:

- O Servidor deverá criar uma conexão TCP utilizando a biblioteca socket, abrir uma porta e aguardar a conexão do Cliente.
- O Cliente será responsável por iniciar a conexão com o Servidor e, após conectado, enviará mensagens digitadas no terminal.
- O Servidor exibirá as mensagens recebidas e também poderá enviar mensagens de volta para o Cliente, permitindo uma comunicação interativa entre os dois lados.

Faça um vídeo demonstrando o funcionamento do programa, destacando as mensagens TCP de estabelecimento da conexão, a troca de dados entre Cliente e Servidor, e o encerramento da conexão. Após finalizar, envie o vídeo no Google Sala de Aula.