UDP e TCP

Conceitos Gerais

1. Introdução

- **Os protocolos TCP** e **UDP** atuam na **camada de transporte** (Camada 4 do modelo OSI). Eles ficam “acima” do IP e são responsáveis por **como os dados são entregues entre aplicações**.

- Enquanto o TCP foca em garantir segurança e organização na entrega dos dados, o UDP foca na rapidez, mas sem garantia de entrega.

- Vamos analisar as características destes e protocolos, e definir o tipo de situação em que cada um deve ser utilizado.

2. UDP

- O protocolo UDP foi criado com o intuito de dar mais celeridade na troca de informação na rede sem muita burocracia. Desta forma, podemos apontar as seguintes características chaves:

1 - **Não confiável** – Não há método de confirmação de entrega dos pacotes enviados;

2 - **Ágil e Rápido** – Não depende de estabelecimento de conexão, ou seja, uma vez que se tem o destino definido, os pacotes serão enviados aumentando assim o desempenho;

3 - **Transferência de Responsabilidades** – O fato de não haver implementação de correção de erros e controle de perdas bem como o controle de fluxo, assume-se que tais recursos serão fornecidos pela camada de aplicação.

- Dessa forma, o serviço UDP é definido como um protocolo da camada de transporte sem conexão fim a fim e não confiável.

2. UDP

- O protocolo UDP (User Datagram Protocol) foi criado com o intuito de dar mais celeridade na troca de informação na rede sem muita burocracia. Desta forma, podemos apontar as seguintes características chaves:

1 - **Não confiável** – Não há método de confirmação de entrega dos pacotes enviados;

2 - **Ágil e Rápido** – Não depende de estabelecimento de conexão, ou seja, uma vez que se tem o destino definido, os pacotes serão enviados aumentando assim o desempenho;

3 - **Transferência de Responsabilidades** – O fato de não haver implementação de correção de erros e controle de perdas bem como o controle de fluxo, assume-se que tais recursos serão fornecidos pela camada de aplicação.

- Dessa forma, o serviço UDP é definido como um protocolo da camada de transporte sem conexão fim a fim e não confiável.

2. UDP

- O protocolo UDP é ideal para situações onde a velocidade de transmissão é mais importante do que a confiabilidade total, sendo usado em aplicações sensíveis ao tempo que toleram pequenas perdas de pacotes, por exemplo:

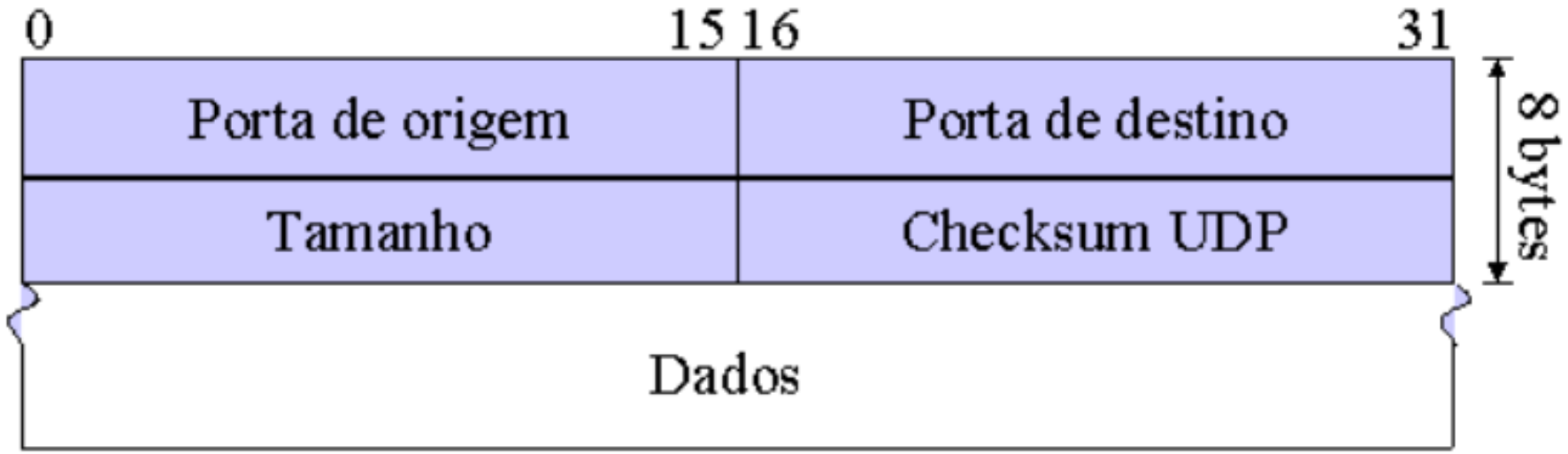
- **Streaming de Mídia ao Vivo**: Plataformas de streaming ao vivo (YouTube Live, Twitch) utilizam UDP para garantir que o vídeo seja transmitido instantaneamente, mesmo que alguns quadros se percam ocasionalmente.

- **Jogos Online (Multiplayer)**: Jogos de ritmo acelerado usam UDP para enviar dados de posição e ações em tempo real, priorizando a baixa latência para evitar "lag".

- **VoIP (Voz sobre IP)**: Aplicações de chamadas de voz e videoconferência, como Skype, Zoom e WhatsApp, usam UDP porque um pequeno chiado ou corte no áudio é melhor do que um atraso longo na conversa.

2. UDP

- A imagem abaixo mostra o cabeçalho do UDP:



2. UDP

- O cabeçalho do UDP é conhecido por ser simples (diferente do TCO que é complexo), tendo tamanho fixo de 8 bytes, por isso ele é tão rápido. O cabeçalho é dividido em apenas 4 campos

1. Porta de Origem: Indica o número da porta do dispositivo que está enviando os dados. É usada para que o receptor saiba para onde enviar uma resposta, se necessário.

2. Porta de Destino: Indica para qual porta o pacote está indo no dispositivo de destino (ex: porta 53 para DNS, porta 123 para NTP). Isso garante que os dados cheguem ao aplicativo correto.

3. Comprimento: Especifica o tamanho total do pacote (cabeçalho + dados). O valor mínimo é 8 bytes (apenas o cabeçalho vazio).

4. Checksum (Soma de Verificação): É um campo usado para verificar se os dados sofreram alguma corrupção durante o trajeto.

Curiosidade: No IPv4, o uso do Checksum no UDP é opcional (embora recomendado), mas no IPv6 ele é obrigatório. Se o receptor detectar um erro no Checksum, ele simplesmente descarta o pacote silenciosamente, sem pedir reenvio.

2. UDP

- Outro conceito importante é o de porta. As portas funcionam como as "portas de um prédio". Se o endereço IP é o número do edifício, as portas são os números dos apartamentos.

- Trazendo para a área de tecnologia, as portas servem para o computador saber para qual programa específico ele deve entregar os dados.

- Por exemplo, se um usuário faz uma consulta DNS, o pacote chegará com a Porta de Destino 53 (porta padrão do DNS).

- O servidor que receber a solicitação sabe que, se chegou na porta 53, ele deve entregar aquele pacote para o software de DNS e não para um banco de dados por exemplo.

3. TCP

- O protocolo TCP (Transmission Control Protocol) é o principal protocolo da camada de transporte. Possui como característica o fato de estabelecer uma conexão (denominada 3-way-handshake) antes de enviar os dados. O protocolo TCP implementa as seguintes técnicas:

1 - Verificação e correção de erros fim a fim;

2 - Recuperação de perda de pacotes e descarte de pacotes duplicados;

3 - Mensagens de confirmação de recebimento por meio de pacotes ACK (Acknowledgement);

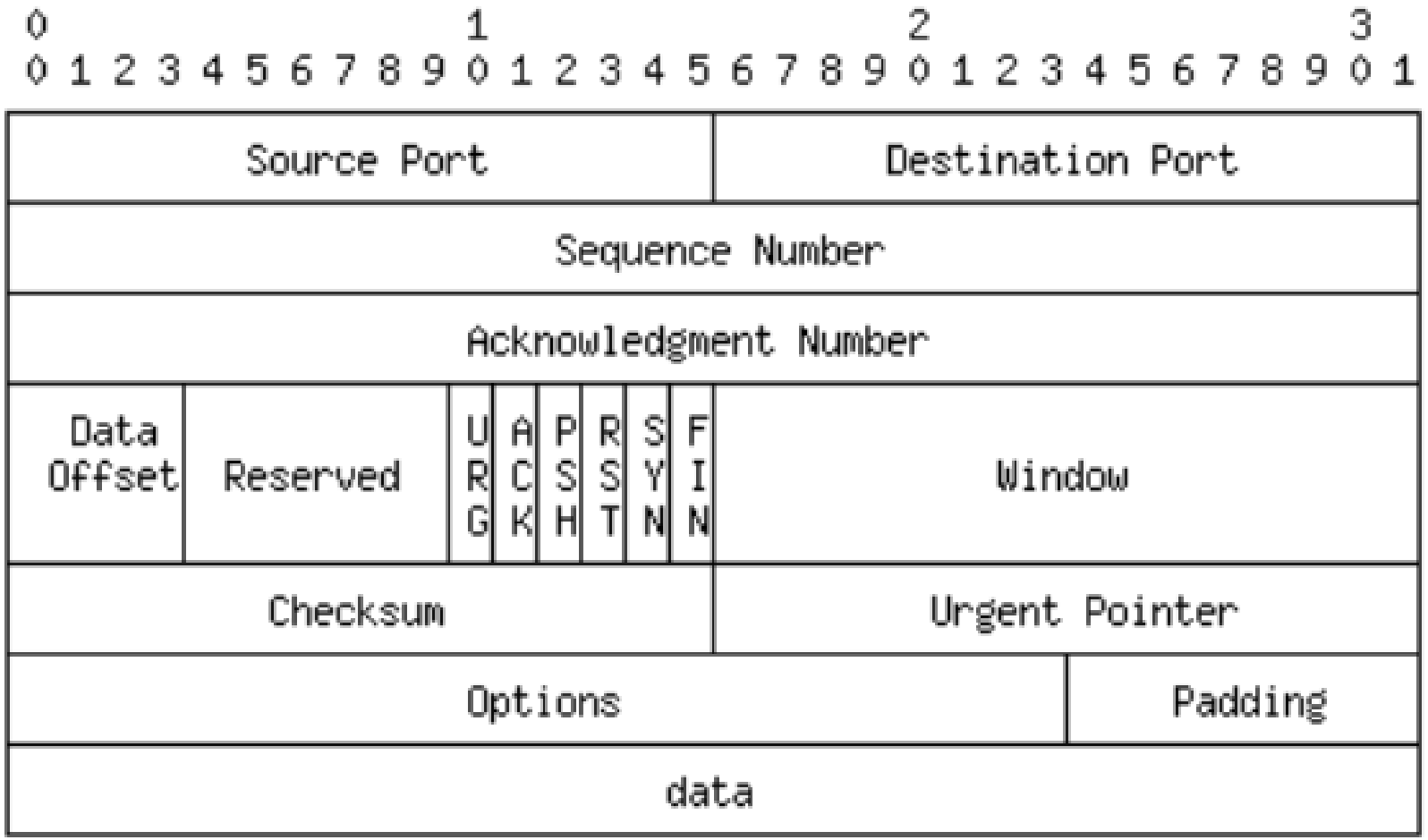
4 - Controle de fluxo através de técnicas como a janela deslizante fim a fim;

5 - Transferência de dados nas duas direções (FullDuplex);

6 - Possibilidade de envio de dados urgentes;

3. TCP

- A imagem abaixo mostra o cabeçalho do protocolo TCP. Podemos observar que ele possui muitos campos:



3. TCP

- O cabeçalho TCP possui muito mais campos quando comparado com o cabeçalho UDP, gerando um overhead maior.

- Para transmitir a mesma informação no protocolo TCP e UDP, fatalmente o TCP consumirá mais tempo e recursos coputacionais (CPU, memória, banda, etc.). Os principais campos são:

- **Source Port e Destination Port:** Campos de 16 bits cada que correspondem às portas de origem e destino.

- **Sequence Number:** Determina o número de sequência do segmento

- **Ack Number:** Corresponde ao número de sequência ESPERADO como próxima informação, ou seja, número de sequência do próximo segmento.

3. TCP

- **Data Offset**: Possui o tamanho do cabeçalho (já que pode variar)
- **Reserved**: Campo reservado de 6 bits. Poderá ser utilizado em implementações futuras;
- **Flags**: São sinais de controle, como mostra a figura abaixo:

SYN → iniciar conexão
ACK → confirmação
FIN → finalizar conexão
RST → resetar conexão
PSH → enviar dados imediatamente
URG → dados urgentes

3. TCP

- **Window Size**: Diz quantos bytes podem ser enviados de uma vez.

- **Checksum**: Campo utilizado para detectar erros em TODO O SEGMENTO. Contempla ainda o cálculo sobre o pseudo cabeçalho IP da camada 3, ferindo o princípio de isolamento das camadas do modelo OS.

- **Urgent Pointer:** Usado quando flag URG está ativa. Indica onde estão os dados urgentes

- **Dados (payload):** neste campo que ficam alocados os dados efetivamente.

4. Conexão do Protocolo TCP

- Como vimos, o TCP é um protocolo orientado à conexão. Para estabelecer uma conexão, deve-se estabelecer uma metodologia padrão para que os dispositivos possam se entender. Esse procedimento no TCP é conhecido como **3-way-handshake.**

- Devemos pensar no 3-way-handshake como um "acordo ou diálogo" entre cliente e servidor: 1- "Você está aí?", 2- "Estou, e você?", 3- "Estou também, podemos começar!".

🥇 1ª etapa — SYN (Sincronizar)

📤 Cliente → Servidor

- Envia um pacote com a flag **SYN = 1**
- Define um número de sequência inicial

📌 Exemplo:

- Seq = 1000

👉 Significa:

> "Quero iniciar uma conexão e começo com esse número"

🥈 2ª etapa — SYN + ACK

📥 Servidor → Cliente

- Responde com:
  - **SYN = 1**
  - **ACK = 1**
- Define seu próprio número de sequência
- Confirma o do cliente

📌 Exemplo:

- Seq = 5000
- ACK = 1001

👉 Significa:

> "Recebi seu pedido e aceito. Aqui está meu número inicial."

🥉 3ª etapa — ACK

📤 Cliente → Servidor

- Envia:
  - **ACK = 1**
- Confirma o servidor

📌 Exemplo:

- ACK = 5001

👉 Significa:

> "Recebi sua resposta. Conexão estabelecida!"