HTTP (continuação)

15 de janeiro de 2024 22:26

Exercício

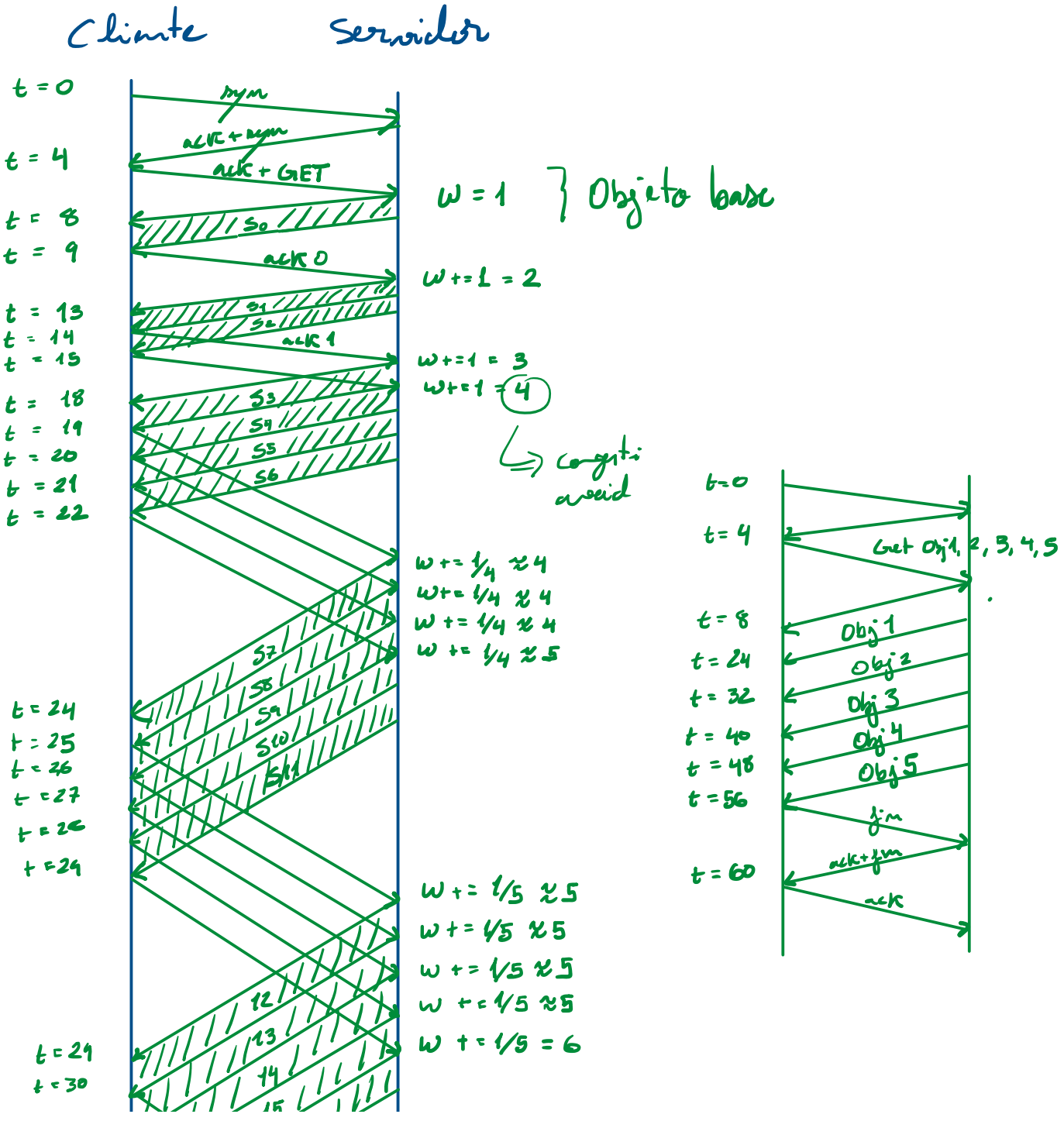
c) HTTP 1.1 → persistente

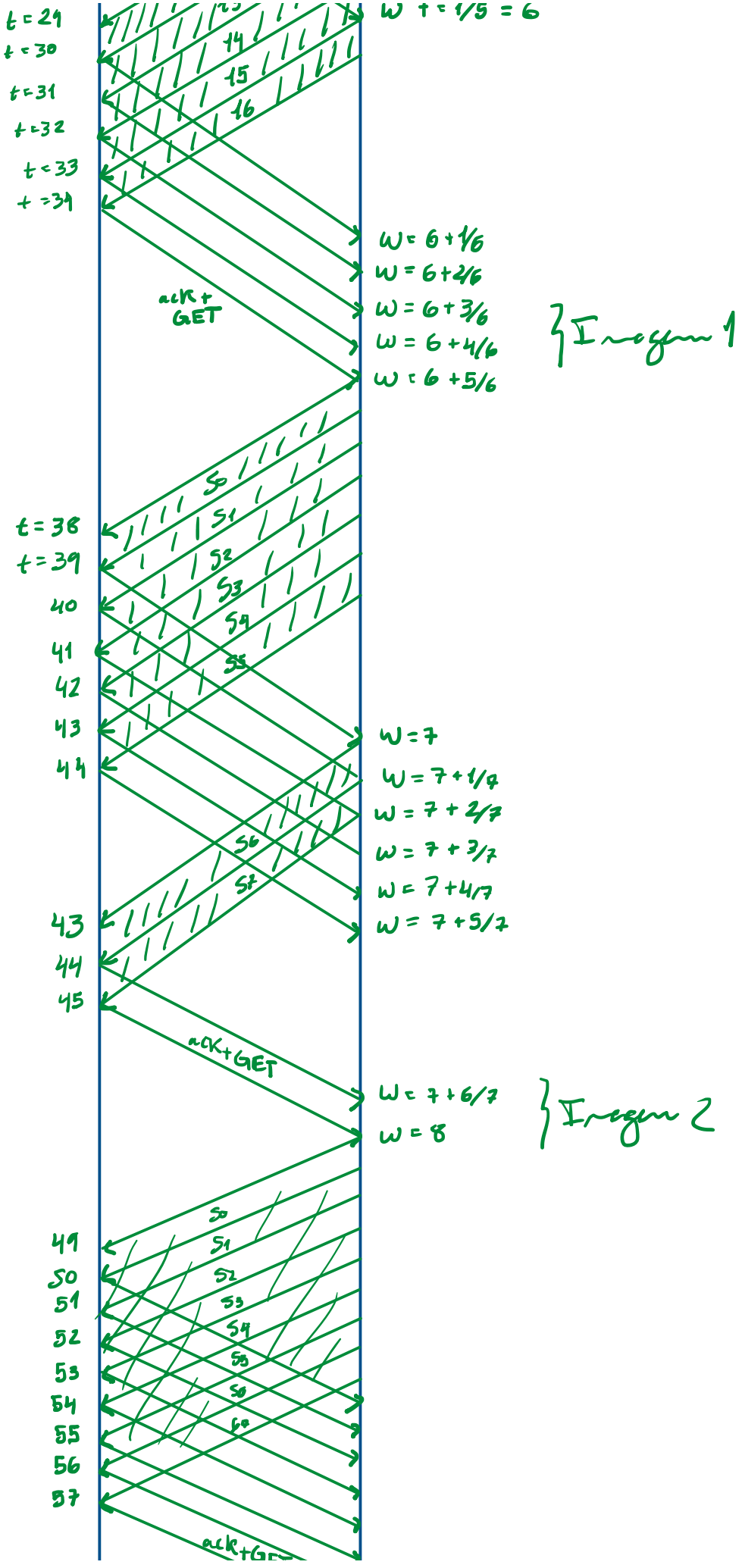
HTTP não persistente

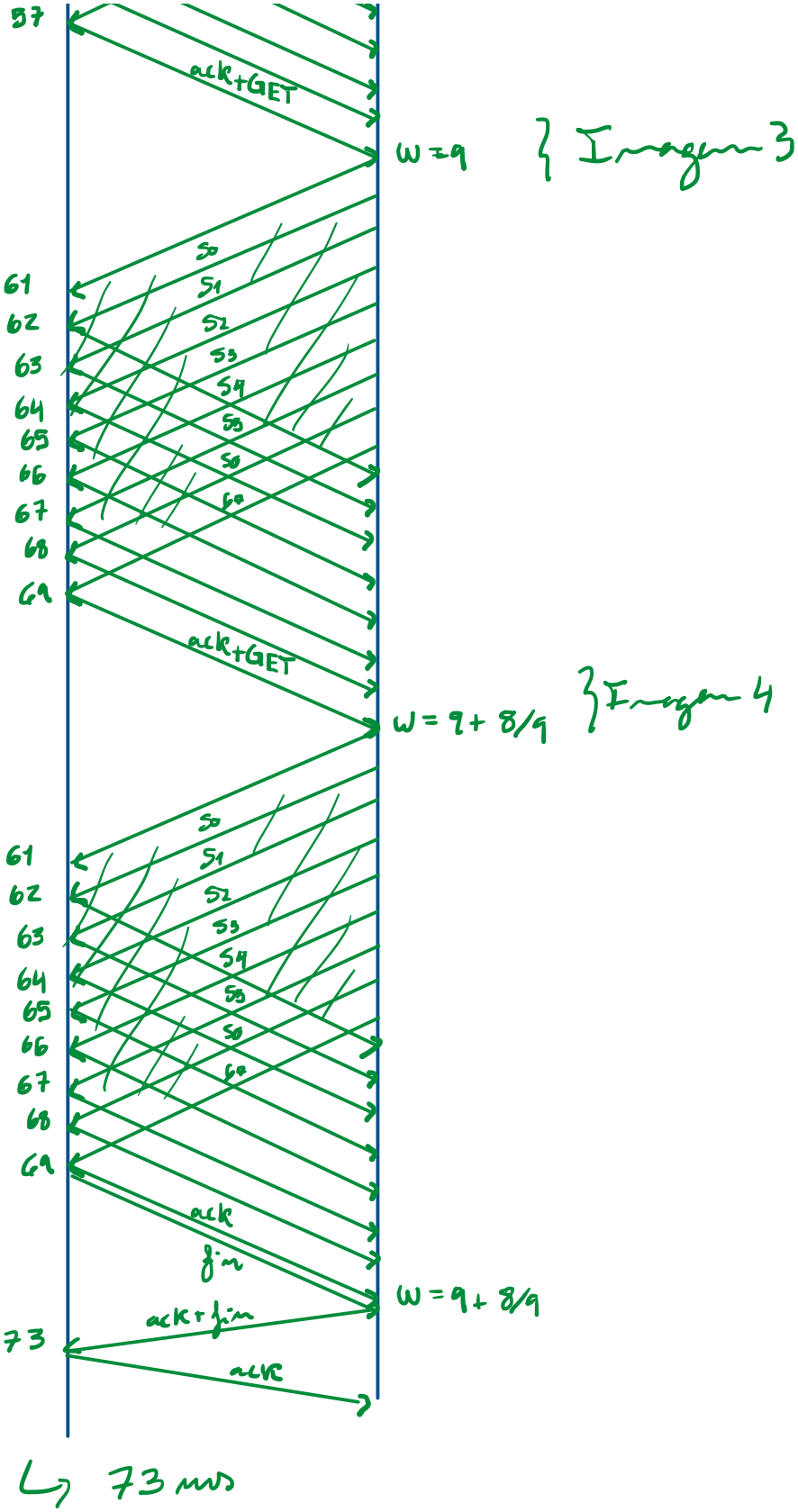
- Só pode ser enviado no máximo um objeto Web por cada conexão estabelecida
- O HTTP/1.0 utiliza HTTP não persistente

HTTP persistente

- Podem ser enviados múltiplos objetos Web por cada ligação estabelecida entre o cliente e o servidor.
- O HTTP/1.1 usa por defeito conexões persistentes



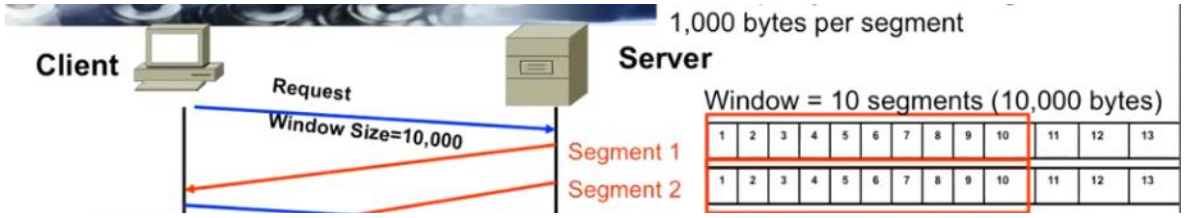

t = 29
t = 30
t = 31
t = 32
t = 33
t = 34
14
15
16
w t = 1/5 = 6
w = 6 + 1/6
w = 6 + 2/6
w = 6 + 3/6
w = 6 + 4/6
w = 6 + 5/6
ack + GET
Imagem 1
S0
S1
S2
S3
S4
S5
S6
S7
t = 38
t = 39
40
41
42
43
44
w = 7
w = 7 + 1/7
w = 7 + 2/7
w = 7 + 3/7
w = 7 + 4/7
w = 7 + 5/7
43
44
45
ack + GET
w = 7 + 6/7
w = 8
Imagem 2
49
50
51
52
53
54
55
56
57

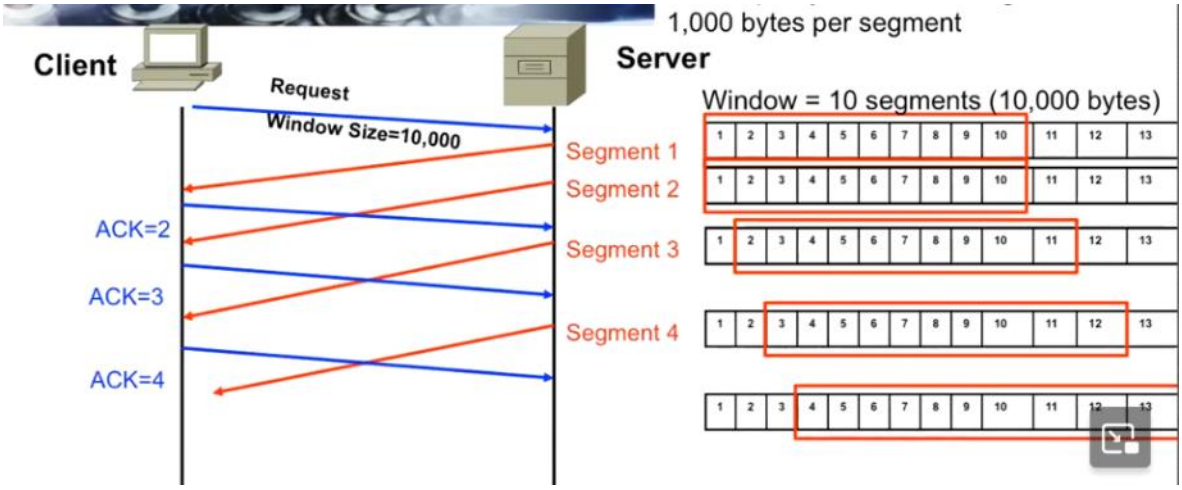


↳ 73 ms

Sliding Window

→





Desempenho HTTP/1.*

- **Como melhorar o desempenho do HTTP/1.*?**
- **Quais as melhores práticas, simples e eficazes, que têm sido usadas com regularidade?**
  - Reduzir o número de consultas ao DNS (*DNS Lookups*)
  - Reutilizar conexões TCP → YT, Netflix, etc.
  - Utilizar CDNs (*Content Delivery Network*)
  - Minimizar o número de redireccionamentos HTTP (*HTTP Redirects*)
  - Eliminar bytes desnecessários nos pedidos HTTP (cabeçalhos)
  - Comprimir os artefactos na transmissão (compressão corpo)
  - Cache dos recursos do lado do cliente
  - Eliminar o envio de recursos desnecessários

Problemas de desempenho

Paralelismo limitado

- O paralelismo está limitado ao número de conexões
- Na prática, mais ou menos 6 conexões por origem

Head-of-line blocking

- Bloqueio do cabeça de fila, acumula pedidos em queue e atrasa a solicitação por parte do cliente
- Servidor obrigado a responder pela ordem (ordem restrita)

Paralelismo limitado

- O paralelismo está limitado ao número de conexões
- Na prática, mais ou menos 6 conexões por origem

Head-of-line blocking

- Bloqueio do cabeça de fila, acumula pedidos em queue e atrasa a solicitação por parte do cliente
- Servidor obrigado a responder pela ordem (ordem restrita)

Overhead protocolar é elevado

- Metadados do cabeçalho não são compactados
- Aproximadamente 800 bytes de metadados por pedido, mais os cookies

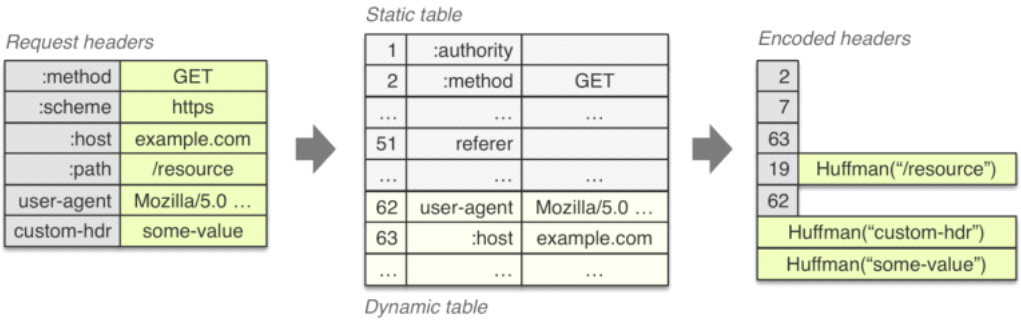
- **Paralelismo é limitado pelo número de conexões...**

- Cada conexão implica overhead de handshake inicial
- Se for HTTPS, ainda tem mais um overhead do handshake TLS
- Cada conexão gasta recursos do lado do servidor
- As conexões competem umas com as outras

- **HTTP2 é uma extensão e não uma substituição do HTTP/1.1**
  - Não se mexe nos métodos, URLs, *headers*, códigos de resposta, etc.
  - Semântica – para a aplicação – deve é a mesma!
  - Não há alterações na API aplicacional...
- **Alvo → as limitações de desempenho das versões anteriores**
  - Primeiras versões do HTTP foram desenhadas para serem de fácil implementação!
  - Clientes HTTP/1.* obrigados a lançar várias conexões em paralelo para baixar a latência.
  - Não há compressão nem prioridades
  - **Mau uso** da conexão TCP de suporte!...

HTTP2 – Compressão do cabeçalho

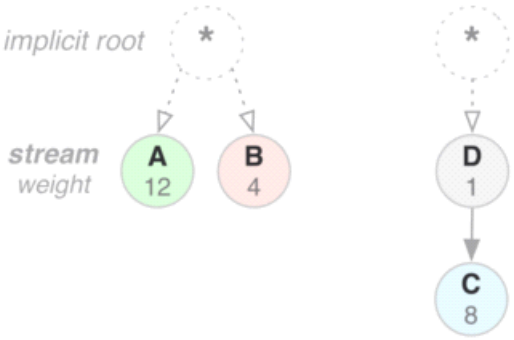
- HPACK



- Valores literais (texto) são codificados com código de Huffman estático
- Tabela indexação estática → por ex: "2" corresponde a "method: GET"
- Tabela indexação dinâmica → Valores enviados anteriormente pode ser indexados!

HTTP2 – pesos e dependências

- Exemplo: stream A deve ter 12/16 e a B 4/16 dos recursos totais
- Exemplo: stream D deve ser entregue antes da stream C

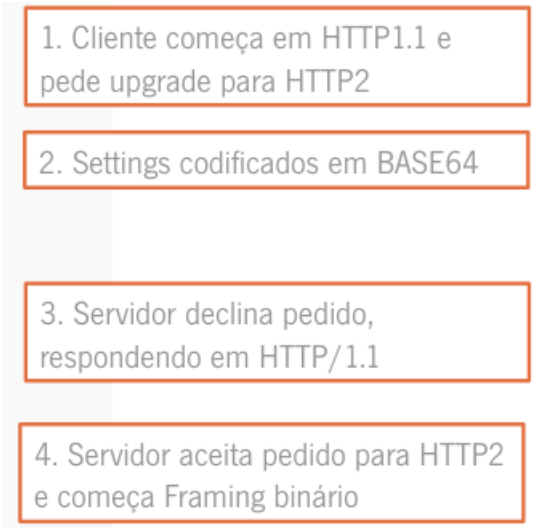


- **Cada stream pode ter um peso**
  - [1-256] integer value
- **Cada stream pode ter uma dependência**
  - ... uma outra stream ID

Negociação protocolar

1. Cliente começa em HTTP1.1 e pede upgrade para HTTP2

2. Settings codificados em BASE64



- **É possível começar logo em HTTP2 se e só se o cliente souber que o servidor fala HTTP2:**
  - Enviar a sequência de 24 octetos: 0x505249202a20485454502f322e300d0a0d0a534d0d0a0d0a
  - Que corresponde a “"PRI * HTTP/2.0\r\n\r\nSM\r\n\r\n")”, logo seguido de uma frame de SETTINGS para definer os parâmetros da conexão HTTP2