Fluxo de Vídeo e CDN: contexto

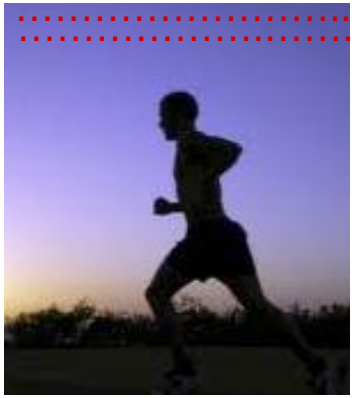
- Tráfego de vídeo: maior consumo de banda da Internet
  - Netflix, YouTube, Amazon Prime: 80% do tráfego dos ISP residenciais (2020)
  - YouTube: bilhões de usuários, Netflix: ~209M (2021)
- ***Desafio 1: como atender ~1 bilhão de usuários?***
  - Um único mega-servidor de vídeo não vai funcionar (Por que?)
- ***Desafio 2: heterogeneidade de usuários***
  - Usuários diferentes tem diferentes recursos disponíveis (cabeado vs móvel; usuários com alta largura de banda vs usuários com baixa largura de banda)

Solução: ***Implementação através de protocolos de aplicação e de uma infraestrutura de milhões de servidores distribuídos***

Multimídia: vídeo

- video: sequência de imagens exibidas a uma taxa constante
  - *Por ex., 24 ou 30 quadros/s*
- Imagem digital: vetor de pixels
  - *Cada pixel (repr. p/ bits) tem uma intensidade luminosa e uma cor*
- codificação: usa redundância ***dentro e entre as*** imagens p/ diminuir o # bits usados na codificação da imagem
  - *espacial (dentro da imagem)*
  - *temporal (de um quadro p/ o outro)*

Ex. de codificação espacial: *ao invés de enviar N valores da mesma cor (todos roxos)*, envia somente 2 valores: o valor da cor e o número valores repetidos (N)



quadro i

quadro i+1

Ex. de codificação temporal: *ao invés de enviar o quadro completo no instante* i+1, envia somente as diferenças de um quadro para o outro

Multimídia: vídeo

- CBR: (constant bit rate): video codificado a taxa cte
- VBR: (variable bit rate): taxa de codificação muda à medida que a codificação temporal e espacial mudam
- exemplos:
  - MPEG-1 (CD-ROM) 1.5 Mbps
  - MPEG-2 (DVD) 3-6 Mbps
  - MPEG-4 (geralmente usada na Internet, 64 kbps - 12 Mbps)

Ex. de codificação espacial: *ao invés de enviar N valores da mesma cor (todos roxos), envia somente 2 valores: o valor da cor e o número valores repetidos (N)*

quadro i

quadro i+1

Ex. de codificação temporal: *ao invés de enviar o quadro completo no instante i+1, envia somente as diferenças de um quadro para o outro*

Multimídia: vídeo

A compressão permite armazenar múltiplas versões do mesmo vídeo, com qualidades diferentes

Ex: 300 kbps, 1 Mbps, 3 Mbps

Usuário decide de acordo com a banda disponível. Por exemplo, um usuário no celular, provavelmente, escolhe 300 kbps

Ex. de codificação espacial:
ao invés de enviar N valores da mesma cor (todos roxos), envia somente 2 valores: o valor da cor e o número valores repetidos (N)

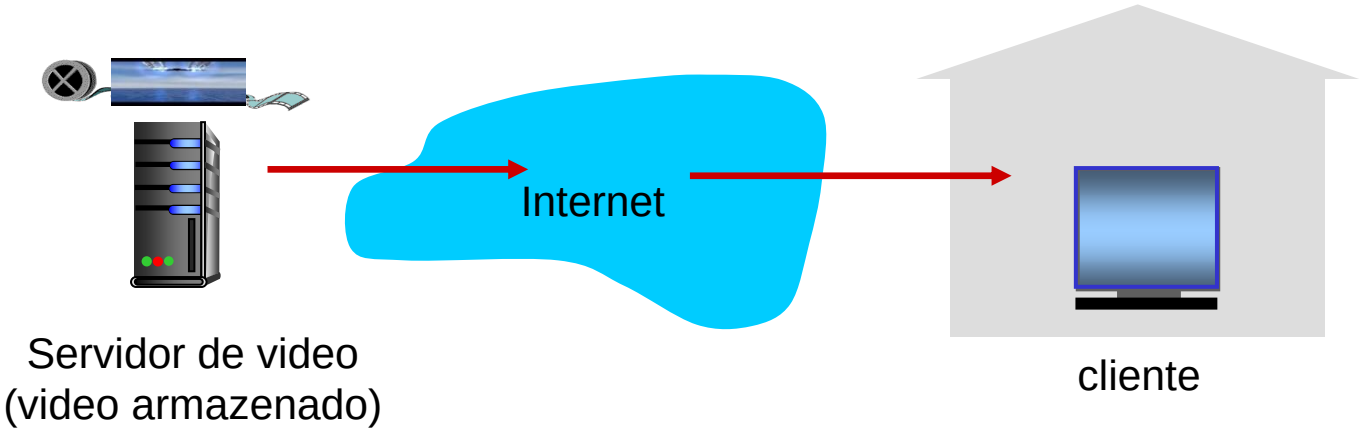
quadro i

quadro i+1

Ex. de codificação temporal:
ao invés de enviar o quadro completo no instante i+1, envia somente as diferenças de um quadro para o outro

Fluxo de vídeo armazenado

Simples cenário:

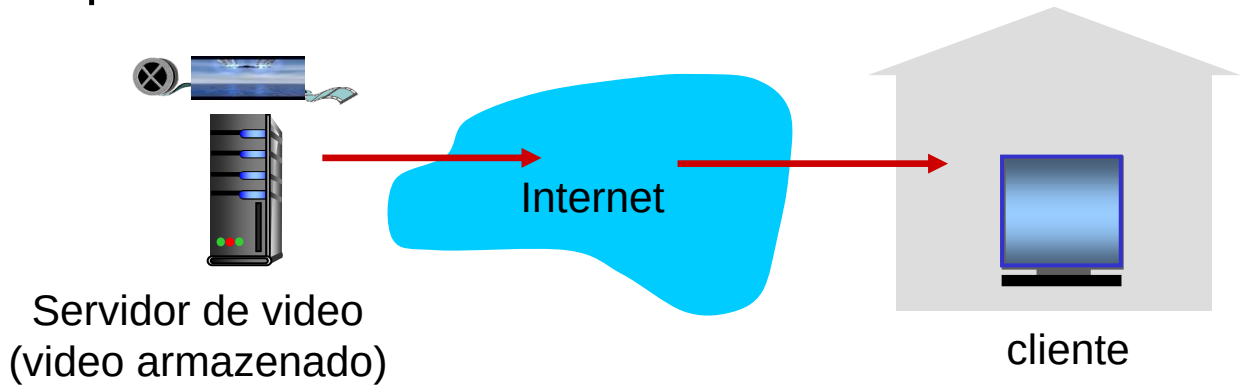


Medida de desempenho principal: ***vazão média fim-a-fim***

- A vazão deve ser maior do que a taxa (bits/seg) do fluxo de vídeo
- Ex. um vídeo armazenado de 67 minutos codificado a uma taxa de 2 Mbps → 1 GB de armazenamento
- Fluxos de vídeo com qualidade 4K requerem uma taxa > 10 Mbps!

Fluxo de vídeo armazenado

Simples cenário:

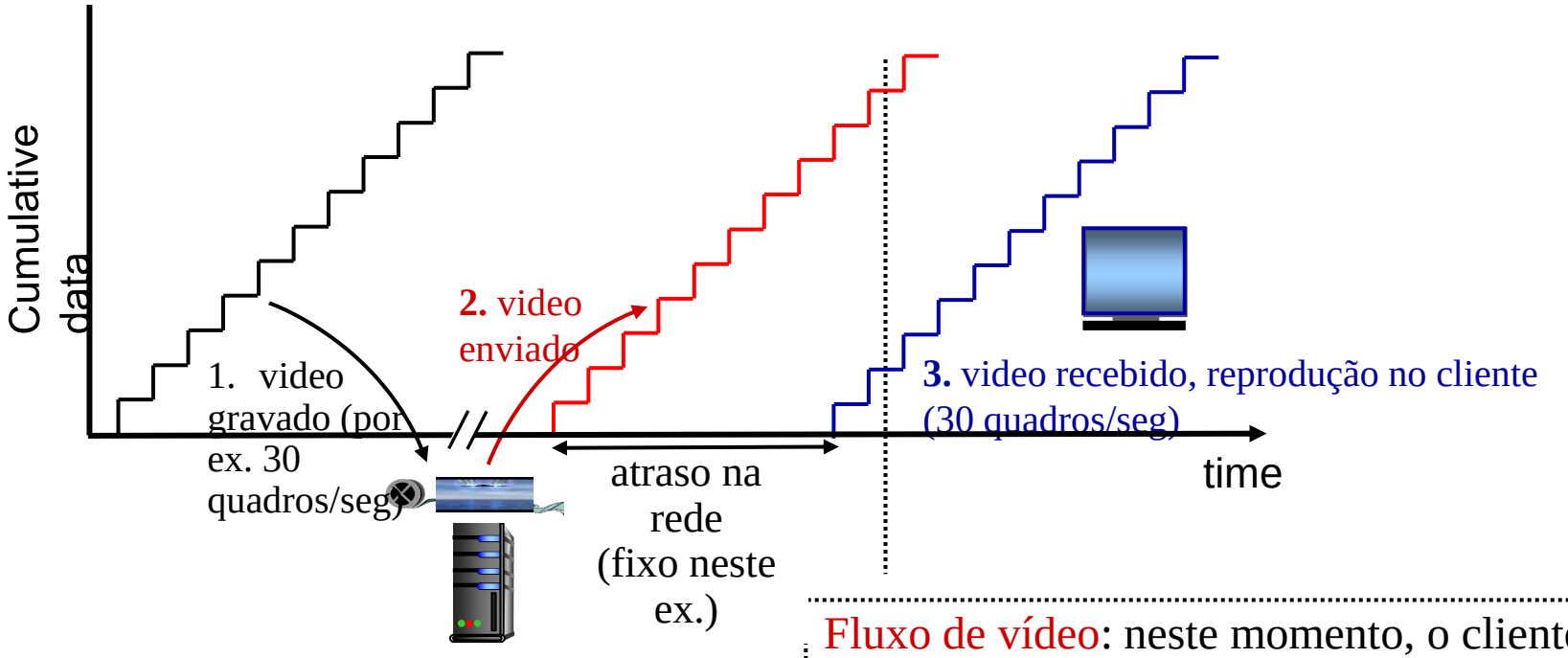


Principais desafios:

- Largura de banda entre o servidor e o cliente vai variar ao longo do tempo, de acordo com os níveis de congestionamento (na rede de acesso, no núcleo da rede e no servidor de vídeo)

- Perda de pacotes e atraso fim-a-fim devido ao congestionamento irão retardar a reprodução do vídeo, resultando numa qualidade de video ruim

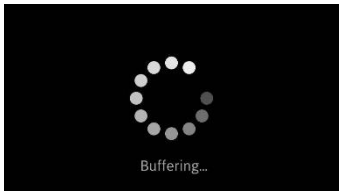
Fluxo de vídeo armazenado



Fluxo de vídeo: neste momento, o cliente começa a reproduzir parte do vídeo, enquanto o servidor ainda está enviando o resto do vídeo

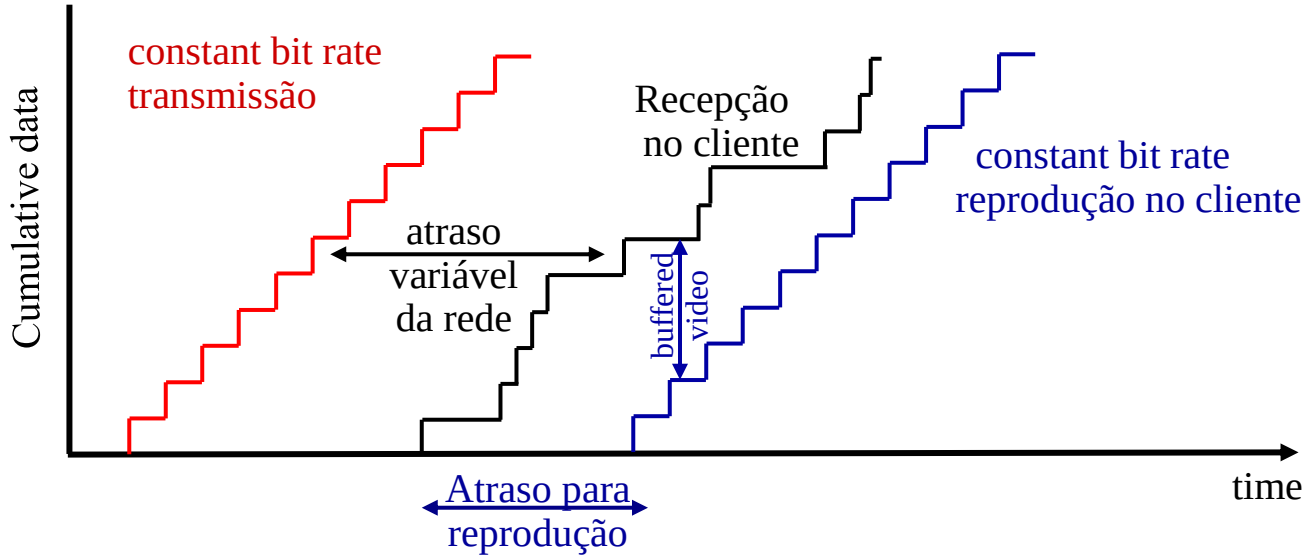
Fluxo de vídeo armazenado: desafios

- Restrição de reprodução contínua:
  uma vez iniciada, deve ser igual à temporização original
  - … mas o atraso na rede é variável (*jitter*), portanto o cliente vai necessitar de um buffer para uma reprodução *fiel* ao video original



- Outros desafios:
  - Interatividade do cliente: pausa, reprodução acelerada, retorno do vídeo, pular para um quadro mais à frente do video
  - Pacotes do video podem ser perdidos e devem ser retransmitidos

Fluxo de vídeo armazenado: buffer de reprodução



Armazenamento no buffer e atraso de reprodução:
compensa o atraso fim-a-fim e o jitter da rede

Streaming: HTTP e DASH

HTTP

- Vídeo é armazenado num servidor http usual como um arquivo com uma URL
- Conforme bytes chegam e são armazenados em um ***buffer***. Quando nº bytes excedem um determinado valor a aplicação inicia a exibição do vídeo obtendo quadros do buffer do cliente, descomprimindo-os e exibindo-os no monitor. ***Ex: Youtube***

Desvantagem/limitação:

todos os usuários recebem a mesma codificação do vídeo, independentemente da banda disponível (e dispositivo usado) de cada um

Streaming: HTTP e DASH

DASH: *D*ynamic, *A*daptive *S*treaming over *H*TTP

Servidor:

- Vídeo é dividido em múltiplos trechos
- Cada trecho é codificado e armazenado em diferentes taxas (versões diferentes)
- Cada versão é armazenada com uma URL diferente no servidor HTTP
- Armazena *o arquivo manifest:*

  Contém URLs diferentes para cada versão (em diferentes taxas) de cada trecho do arquivo de vídeo

Streaming: HTTP e DASH

*DASH: D*ynamic, *A*daptive *S*treaming over *H*TTP

Cliente:

- *Solicita o arquivo manifest*
- *Seleciona um trecho do vídeo por vez, especificando a URL e a quantidade de bytes no pedido HTTP GET*
- *Periodicamente mede a largura de banda entre C-S para selecionar qual pedaço se adequa melhor à banda disponível*

⇒ ***Pode alterar dinamicamente a URL (taxa de codificação) durante o download do video, conforme a banda disponível.***
Interessante para usuários móveis, cuja banda flutua demais

Streaming: HTTP e DASH

*DASH: D*ynamic, *A*daptive *S*treaming over *H*TTP

"inteligência" no cliente: cliente determina

- ***quando*** requisitar um pedaço (de tal forma que o buffer não fique vazio ou ocorra *overflow*)
- Determina ***com qual taxa de codificação*** requisitar um trecho do arquivo (qualidade maior quando houver mais banda disponível)
- Determina ***onde*** requisitar um pedaço: pode requisitar de uma URL cujo servidor esteja mais "próximo" do cliente (nº menor de enlaces) ou de uma URL com maior largura de banda

Redes de Distribuição de Conteúdo (CDN)

Desafio:

Como distribuir conteúdo (selecionado entre milhões de vídeos) para centenas de milhares de usuários simultâneos garantindo continuidade na reprodução e alta interatividade? Ex. Youtube

Opção1: único e enorme "mega-servidor"

- Ponto único de falha
- Ponto único de congestionamento
- Longo caminho para clientes distantes (muitos enlaces, com bandas diferentes e diferentes ISPs) ⇒ enlace de gargalo limita a vazão fim-a-fim ⇒ "congelamento" do vídeo reproduzido
- múltiplas cópias do mesmo vídeo enviadas sobre o enlace de saída do servidor ⇒ desperdício de largura de banda e $$$ ao ISP

… solução não escalável!!!

Redes de Distribuição de Conteúdo (CDN)

Opção 2:

Armazenar e distribuir várias cópias de vídeos em vários sites distribuídos geograficamente (CDN)

requisição do usuário é direcionada ao servidor que provê o melhor serviço

Privado:

- ***Google****: distribui os vídeos do* ***Youtube***

Terceirizado:

- *Akamai, Limelight, Level-3: empresas que distribuem o conteúdo de muitos provedores*

Redes de Distribuição de Conteúdo (CDN)

Existem duas abordagens/arquiteturas para posicionar os conteúdos:

enter deep:

Colocar os servidores CDN dentro das redes de acesso dos ISPs

- ISPs próximos dos usuários
- Diminui o nº de enlaces ⇒ *atraso menor*
- Usado pela Akamai: 240.000 servidores distribuídos em mais de 120 países (2015)

Desvantagem: manutenção e gerenciamento

Redes de Distribuição de Conteúdo (CDN)

bring home:

número menor (dezenas) de clusters maiores em pontos de troca de tráfego (IXP) próximos (não dentro de) às redes de acesso

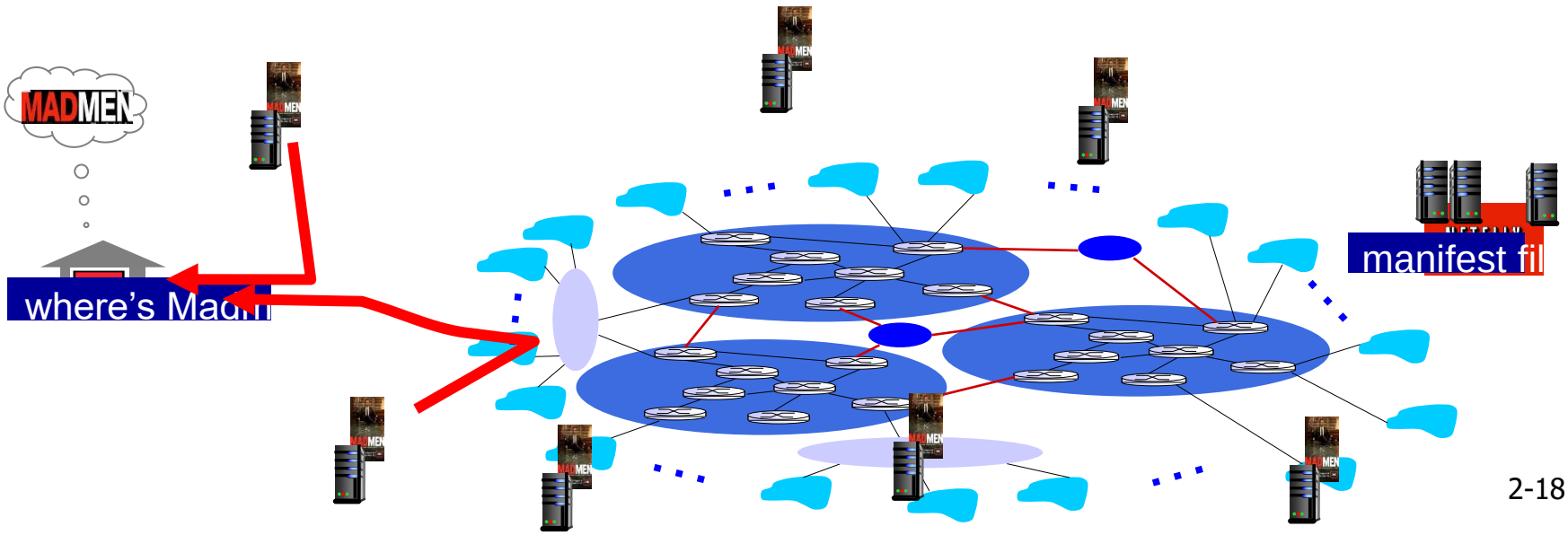
- Usado pela Limelight
- Simplifica a manutenção e o gerenciamento

  …. ao custo de possível aumento do atraso

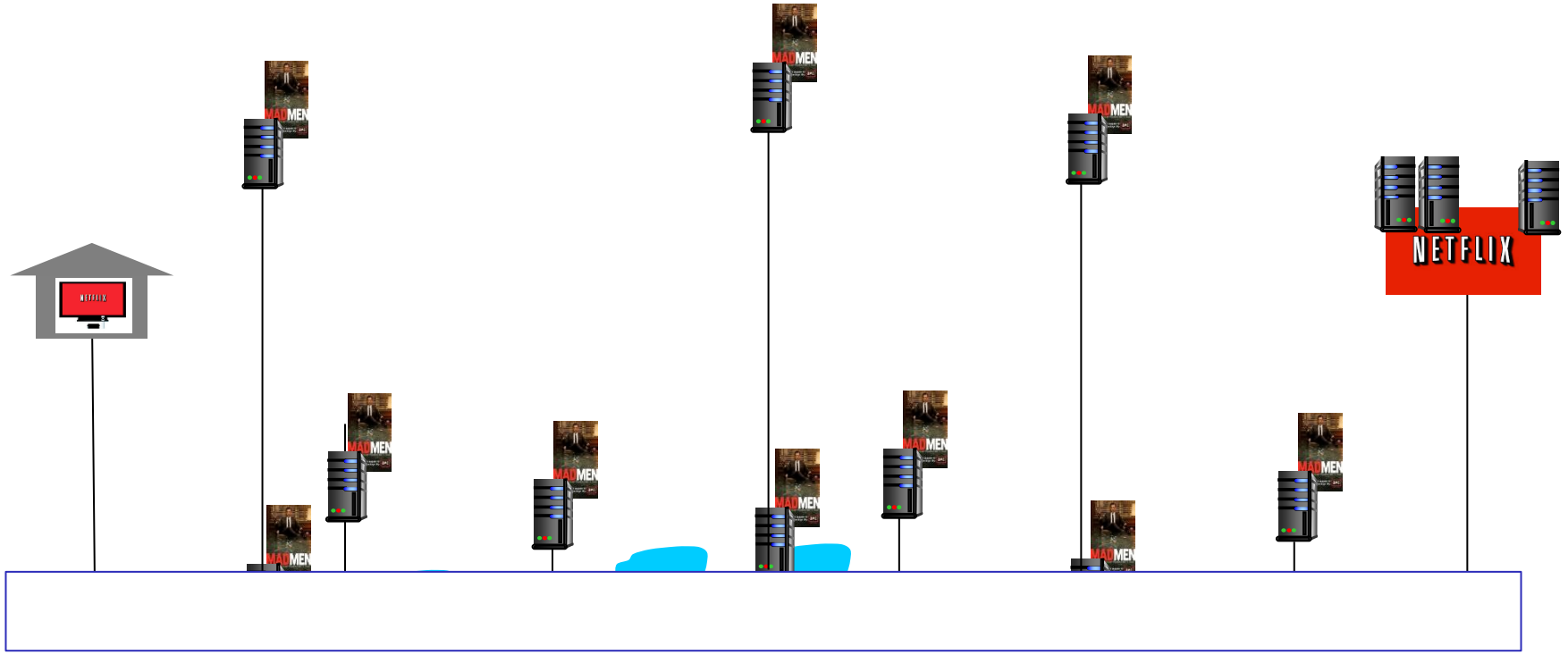
  …. e diminuição da vazão

Redes de Distribuição de Conteúdo (CDN)

- CDN: armazena cópias do conteúdo nos nós CDN
  - Replica conteúdo de vídeos mais populares (“PUSH”)
- Assinante solicita o conteúdo do CDN
  - Acessa o servidor CDN mais próximo e obtém o conteúdo
  - Pode escolher uma cópia de um outro servidor CDN se o caminho para o mais próximo estiver congestionado



Redes de Distribuição de Conteúdo (CDN)

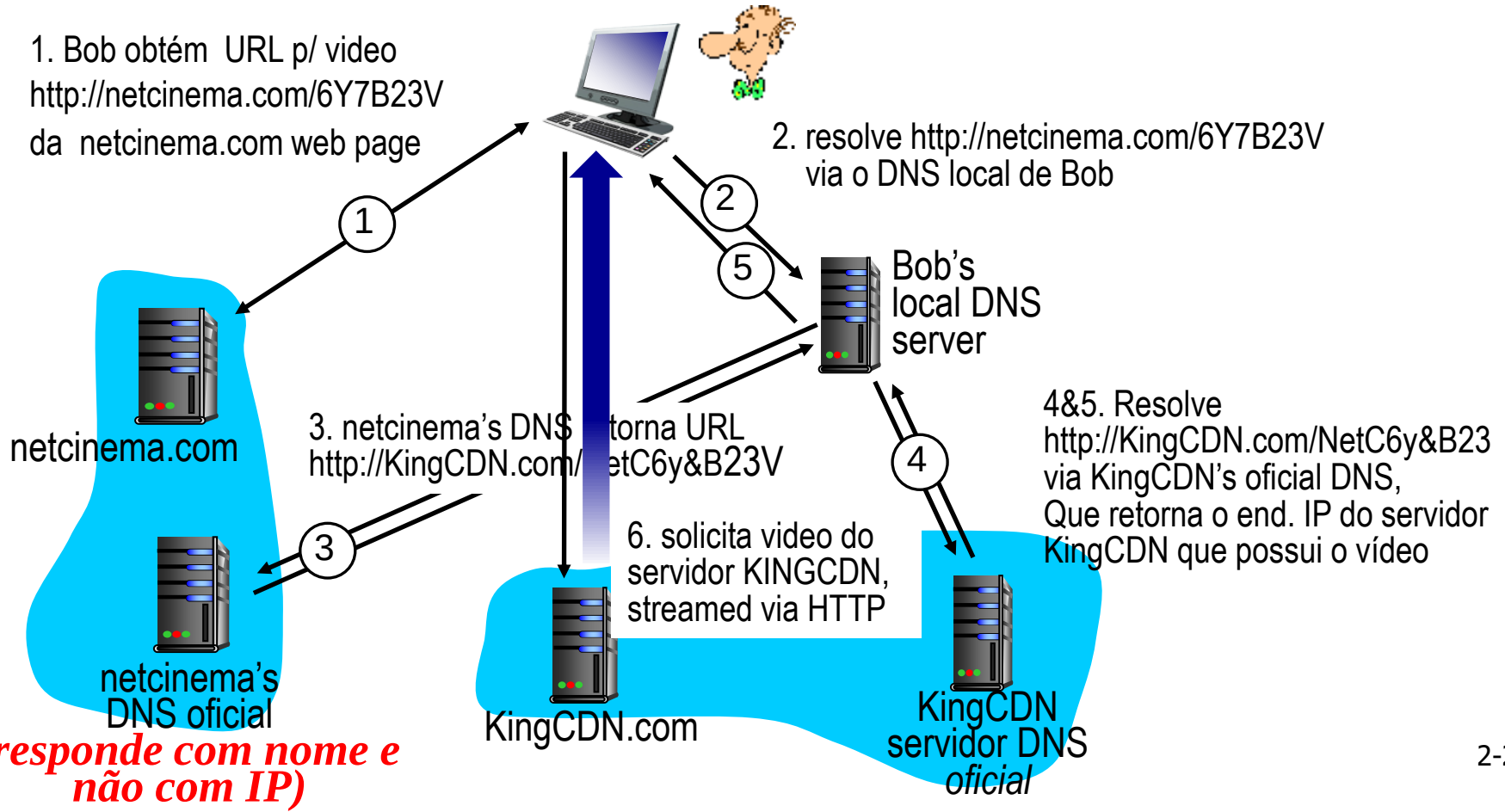


***Desafios para se obter uma cópia numa rede congestionada*:**

- De qual servidor CDN obter o conteúdo?
- Como o usuário se comporta na presença de congestionamento?
- Qual conteúdo guardar em cada um dos servidores CDN?

CDN: acesso ao conteúdo – usa DNS

Bob pede o vídeo http://netcinema.com/6Y7B23V
video armazenado na CDN em http://KingCDN.com/NetC6y&B23V



(responde com nome e não com IP)

CDN: Estratégia de seleção do cluster

- CDN aprende o IP do cliente quando recebe a requisição DNS
- Seleciona o cluster apropriado baseado no IP do servidor DNS local do cliente (LDNS)

1ª estratégia:

- Atribuir o cluster mais próximo geograficamente
- Cada LDNS é mapeado a uma localização geográfica

Problemas:

- Mais próximo geograficamente pode não ser o mais próximo em número de hops. Além disso, o LDNS pode ser remoto, afastado do cliente
- Ignora variações de atraso e banda, sempre atribuindo o mesmo cluster ao cliente

CDN: Estratégia de seleção do cluster

2ª estratégia:

- Fazer medidas em tempo real do atraso e de perdas entre o os clusters e os clientes
- Envia mensagens de ping ou de pedidos DNS para os servidores locais

Problema:

Servidores podem não responder (bloqueia pacotes de PING, por questões de segurança)

Estudo de caso: Netflix

- Desde 2020 é o maior provedor residencial de filmes e séries de TV online da América do Norte
- Dois componentes:
  - Nuvem de servidores da Amazon
  - Infraestrutura própria de CDN privada
- Servidores da Amazon:
  - Web Site para registro, login, cobrança, catálogo de filmes e recomendações
  - Upload dos filmes provenientes dos estúdios (*Content ingestion – obtenção do conteúdo*)
  - Criação das diferentes versões *(bit rate)* e formatos, cada uma delas apropriada para diferentes tipos de dispositivos *(Content processing). Permite Fluxo adaptativo – DASH*
  - *Upload dos filmes processados na infraestrutura de CDNs da Netflix*

Estudo de caso: Netflix

Características:

- Rack de servidores podem se localizar em ISPs residenciais ou em IXPs
  - Mais de 200 racks em IXPs
  - 100´s de racks em ISPs residenciais
  - Cada servidor no rack tem muitas portas Ethernet de 10 Gbps e mais de 100 Terabytes de armazenamento
  - Racks nos IXPs: dezenas de servidores com toda a biblioteca de vídeos (com diferentes versões, para suportar DASH)
  - Os ISPs locais contêm apenas os vídeos mais populares
- Usa o Sistema de PUSH, distribuindo os vídeos durante os horários de menor demanda. Locais sem a biblioteca completa recebem apenas os vídeos mais populares

Estudo de caso: Netflix

Distribuição – interação entre clientes e servidores

- Web pages para escolher os vídeos estão na nuvem da Amazon
- Qdo o usuário seleciona um filme para reprodução:

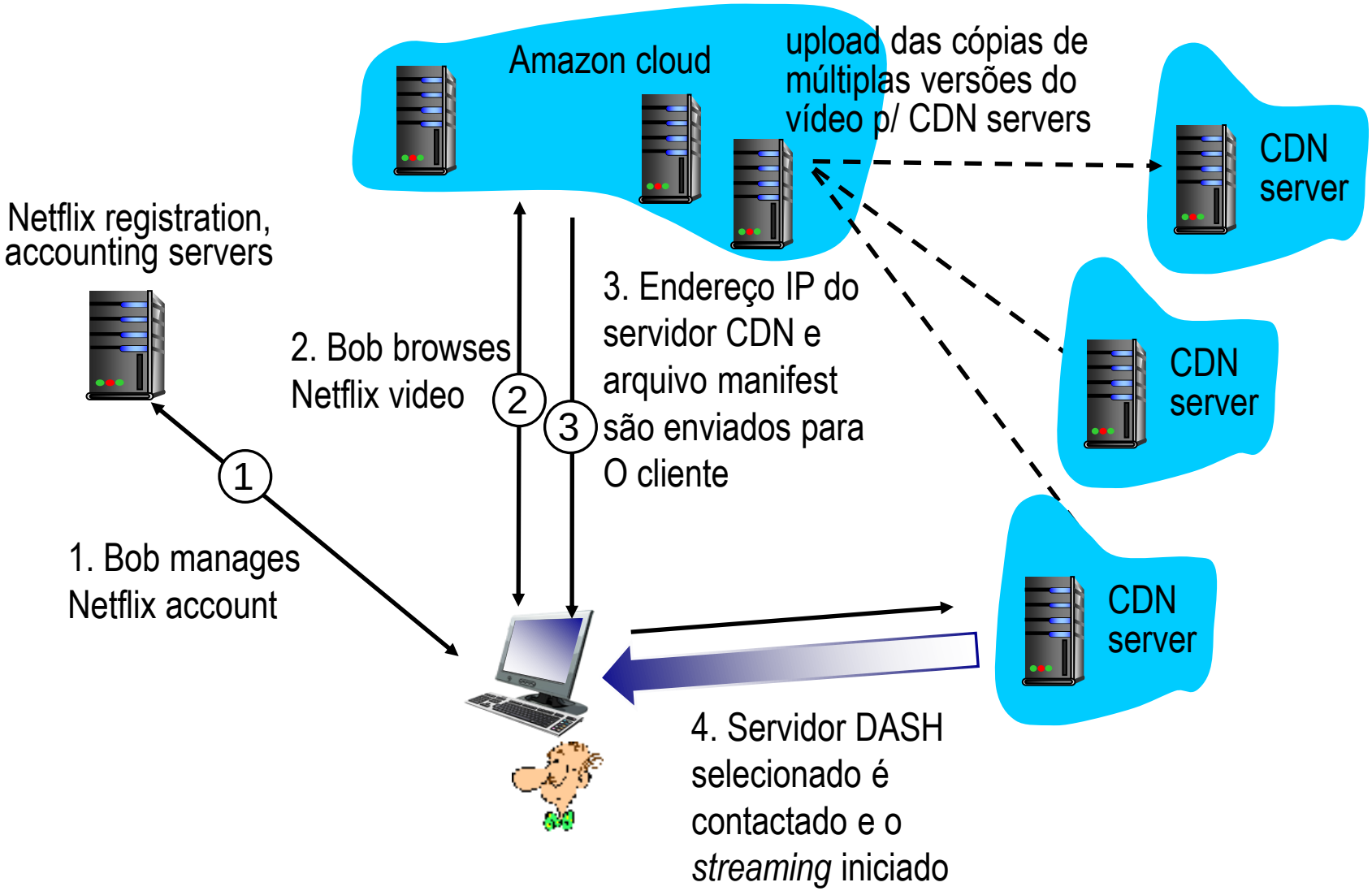
1. SW da Netflix rodando nos servidores da Amazon determina quais servidores CDN possuem cópias do filme e qual deles é o "melhor" para atender à demanda
2. Se cliente usa um ISP que contém um ***rack*** Netflix instalado (e se este tem o filme desejado) é selecionado um servidor neste ***rack***, caso contrário é selecionado um servidor do IXP mais próximo
3. Endereço IP do servidor é enviado ao cliente, assim cmo o arquivo *manifest*, que possui as URLs das diferentes versões do filme
4. Cliente se conecta ao servidor CDN e interage usando DASH

Estudo de caso: Netflix

Distribuição – interação entre clientes e servidores:

- Netflix usa trechos de 4 segundos do filme
- A qualidade dos trechos (bit rate) pode variar conforme as medidas de vazão feitas pelo cliente durante o download
- Netflix não necessita empregar redirecionamento DNS porque ela só trabalha com vídeos e não com páginas Web. O SW da Netflix rodando nos servidores da Amazon já determinam a qual servidor o cliente deve se conectar

Estudo de caso: Netflix



YouTube: arquitetura

Maior *site* existente para compartilhamento de vídeos (2005): comprado pela Google em 2006

- Assim como o Netflix, faz uso extensivo da tecnologia CDN para a distribuição dos vídeos
- A Google também utiliza sua própria rede CDN privada
- Tem instalado ***clusters*** de servidores em centenas de locais de IXPs e ISPs diferentes, além dos enormes ***datacenters*** da própria Google
- Ao contrário da Netflix, a Google usa ***pull-caching*** (armazenagem de informações solicitadas, como caches Web), além do redirecionamento via DNS
- Na estratégia de seleção do ***cluster,*** direciona o cliente para o ***cluster*** cujo RTT entre o cliente e o mesmo seja o menor possível (para balancear a carga, pode redirecioná-lo para outro mais distante)

YouTube: arquitetura

- Emprega o ***streaming*** por HTTP, muitas vezes disponibilizando uma pequena quantidade de versões diferentes do video, com taxas de bits diferentes
- Não utiliza o DASH: usuário escolhe a versão manualmente
- Para economizer banda e recursos do servidor, limita o fluxo de *bytes* transmitidos após uma certa quantidade ter sido obtida pelo cliente
- Produtores também enviam seus vídeos do cliente para o servidor via HTTP

- O YouTube processa cada video recebido, convertendo-o para um formato próprio e criando várias versões em diferentes taxas de bits. Esse processamento é realizado dentro dos ***datacenters*** da Google

Trabalho teórico

1. Descreva como funciona a infraestrutura de rede privada e de servidores CDN da Google, a saber:

- *tipos/tamanho/quantidade de servidores*
- *onde estão localizados*
- *função de cada tipo de servidor*
- *filosofia de localização (Enter Deep ou Bring Home)*
- *o que acontece quando um usuário faz uma solicitação de busca*

Entrega: 30/6/2022

Capítulo 2: Resumo

Nosso estudo sobre aplicações de rede está agora completo!

- Arquiteturas de aplicações
  - cliente-servidor
  - P2P
- Requisitos de serviço das aplicações:
  - confiabilidade, banda, atraso
- Modelos de serviço de transporte da Internet
  - orientado à conexão, confiável: TCP
  - não confiável, datagramas: UDP
- Protocolos específicos:
  - HTTP
  - SMTP, POP, IMAP
  - DNS
  - P2P: BitTorrent
- Streaming de Vídeo: CDNs

Capítulo 2: Resumo

Mais importante: aprendemos sobre *protocolos*

- troca típica de mensagens pedido/resposta
  - cliente solicita info ou serviço
  - servidor responde com dados, código de *status*
- formatos de mensagens:
  - cabeçalhos: campos com info sobre dados (metadados)
  - dados: info sendo comunicada

Temas importantes:

- msgs de controle vs. dados
  - na banda, fora da banda
- centralizado vs. descentralizado
- s/ estado vs. c/ estado
- transferência de msgs confiável vs. não confiável
- "complexidade na borda da rede"