

# Sistemas Distribuídos

## Aula 8

### **Aula de hoje**

- DNS
- CDN

# Mapeando Nomes em Números

- Todo dispositivo conectado à Internet possui um identificador único (endereço IP e porta)
  - necessário para localizar dispositivo
- Humanos preferem nomes; computadores preferem números
  - João da Silva → DRE: 119119119
  - [www.ufrj.br](http://www.ufrj.br) → 146.164.170.36



- Sistema (distribuído) que traduz nome para número
- Como esta tradução é feita na Internet?

# Mapeando Nomes

- **Ideia 0:** Um grande banco de dados com tabela nomes/endereço IP

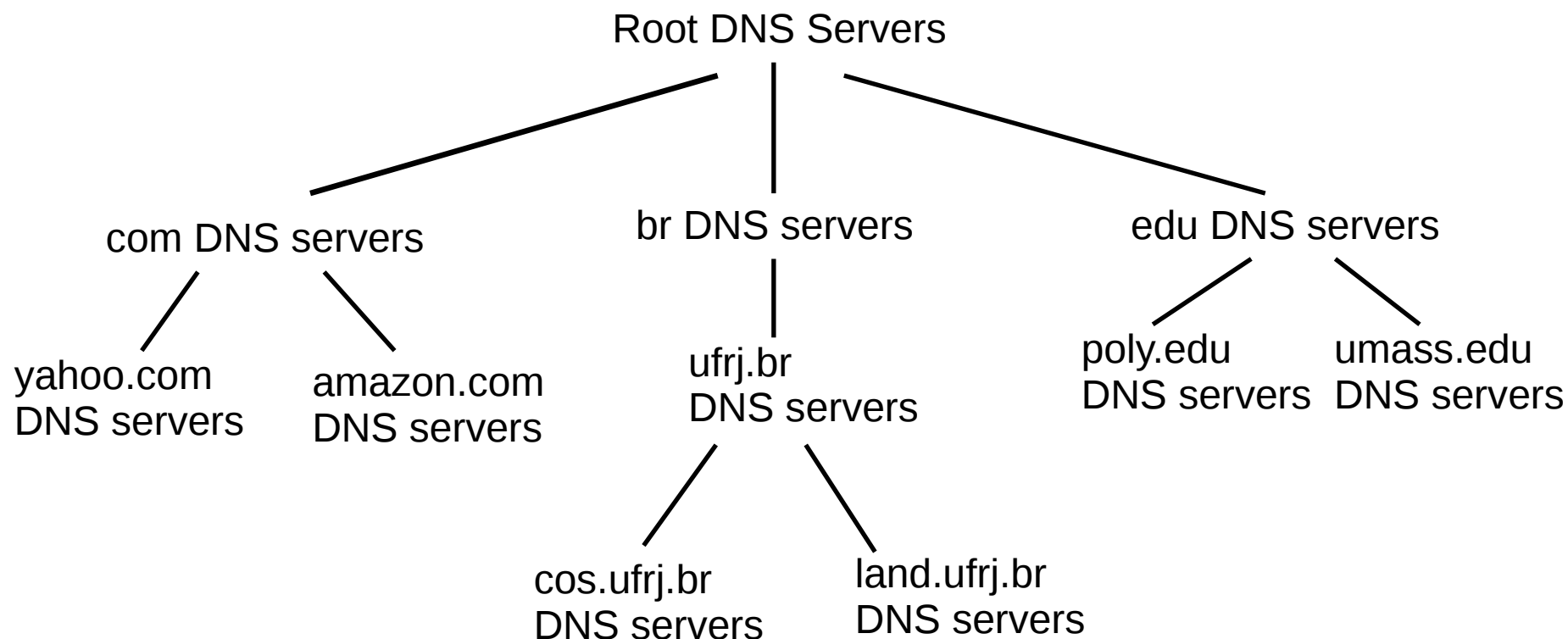


- Não possui escalabilidade
  - Não é tolerante a falhas
  - Como fazer controle de acesso?
    - quem pode atualizar dados do BD?
  - Quem é o responsável pelo sistema?
    - e quem paga a conta?
- 
- Replicação do BD não resolve os últimos dois problemas

# DNS: Domain Name System

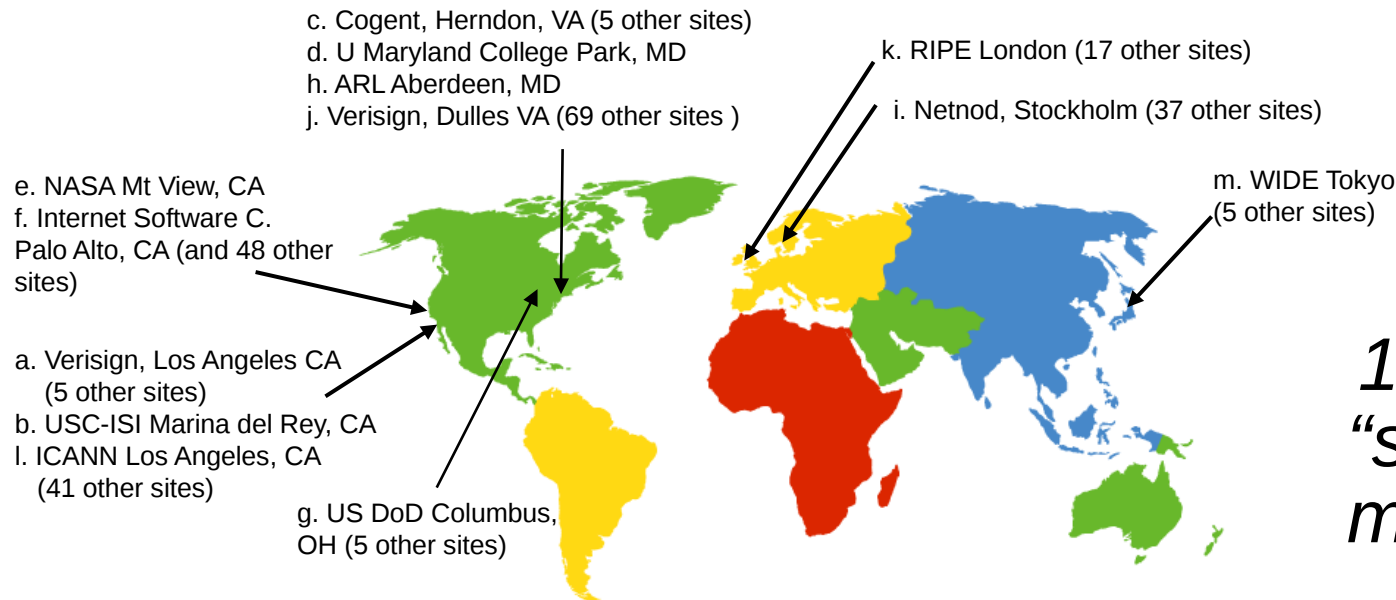
- Sistema distribuído para tradução de nomes (e serviços)
  - projetado e implementado em 1983 (pequenas mudanças desde então)
  - ex. mapeia `www.cos.ufrj.br` → `146.164.34.6`
- Hierarquia (árvore) de servidores de nomes (*name server*)
  - servidor de nome conhece mapeamento entre nome e IP de seus filhos na árvore
  - folhas guardam mapeamento definitivo: *authoritative name servers*
  - servidor que inicia pedido: *local name server*

# Hierarquia no DNS



- Hierarquia segue estrutura de nomes
  - definida pela separação por “.”
- Para cada nível e cada nó na árvore, temos um ou mais servidores

# DNS: Root name servers



*13 root name  
“servers” no  
mundo*

- Sempre disponíveis, conhecem todos os TLD
- Contactados por *local name servers*
  - local name server conhece IP dos root name servers
- Resolvem os nomes (respondem com IP)

# TLD e Authoritative Servers

## Servidores de Top-level Domain (TLD):

- Responsáveis pelo domínio de mais alto nível
- com, org, net, edu, etc, e todos os top-level domain dos países: br, ar, jp, ch, etc
- NIC.br mantém servidores TLD para “br”, Network Solutions para “com”, Educause para “edu”, etc

## Servidores Autoritativos:

- Servidores que oferecem mapeamento definitivo entre nome e IP para um servidor ou serviço (ex. Web, Email, etc)
- Gerenciados pela própria organização, pelo provedor de serviço de internet (ISP), ou CDN (ex. PESC gerencia seu próprio servidor DNS)

# Local Name Server

- Recebe consulta proveniente dos dispositivos finais (ex. seu laptop)
  - dispositivo faz uma consulta DNS através de algum local name server
- Não pertence a hierarquia de servidores
- Cada organização ou ISP possui ao menos um (ex. PESC possui um, UFRJ outros)
- Também chamado de “*default name server*”
- Faz o papel de *proxy*, pois envia a consulta para a hierarquia de servidores e responde aos dispositivos



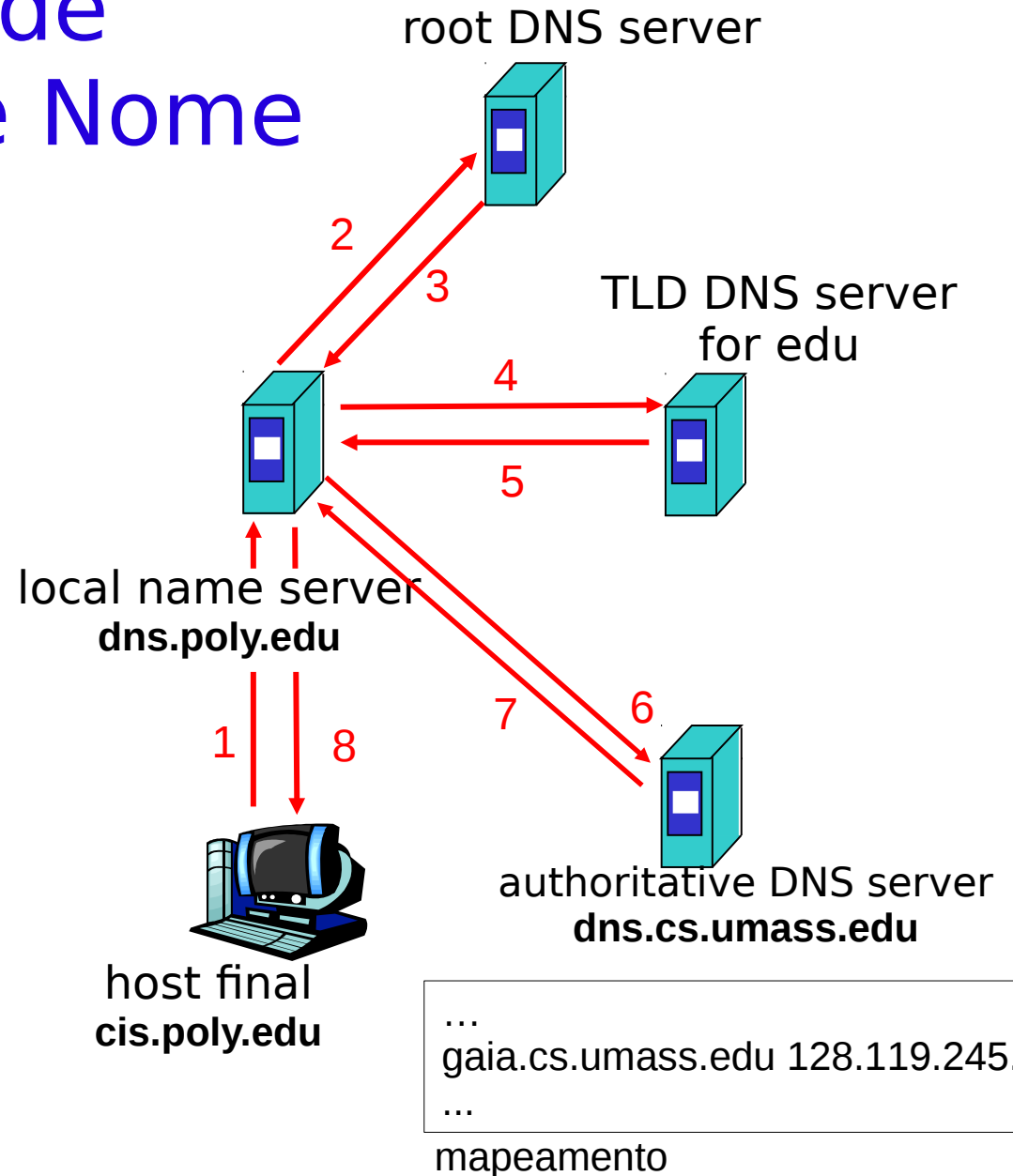
# Exemplo de Resolução de Nome

Host cis.poly.edu quer endereço IP de gaia.cs.umass.edu

## Consulta iterativa:

Servidor contactado responde com IP do servidor para contactar

“Eu não conheço este nome, mas pergunte a este servidor”

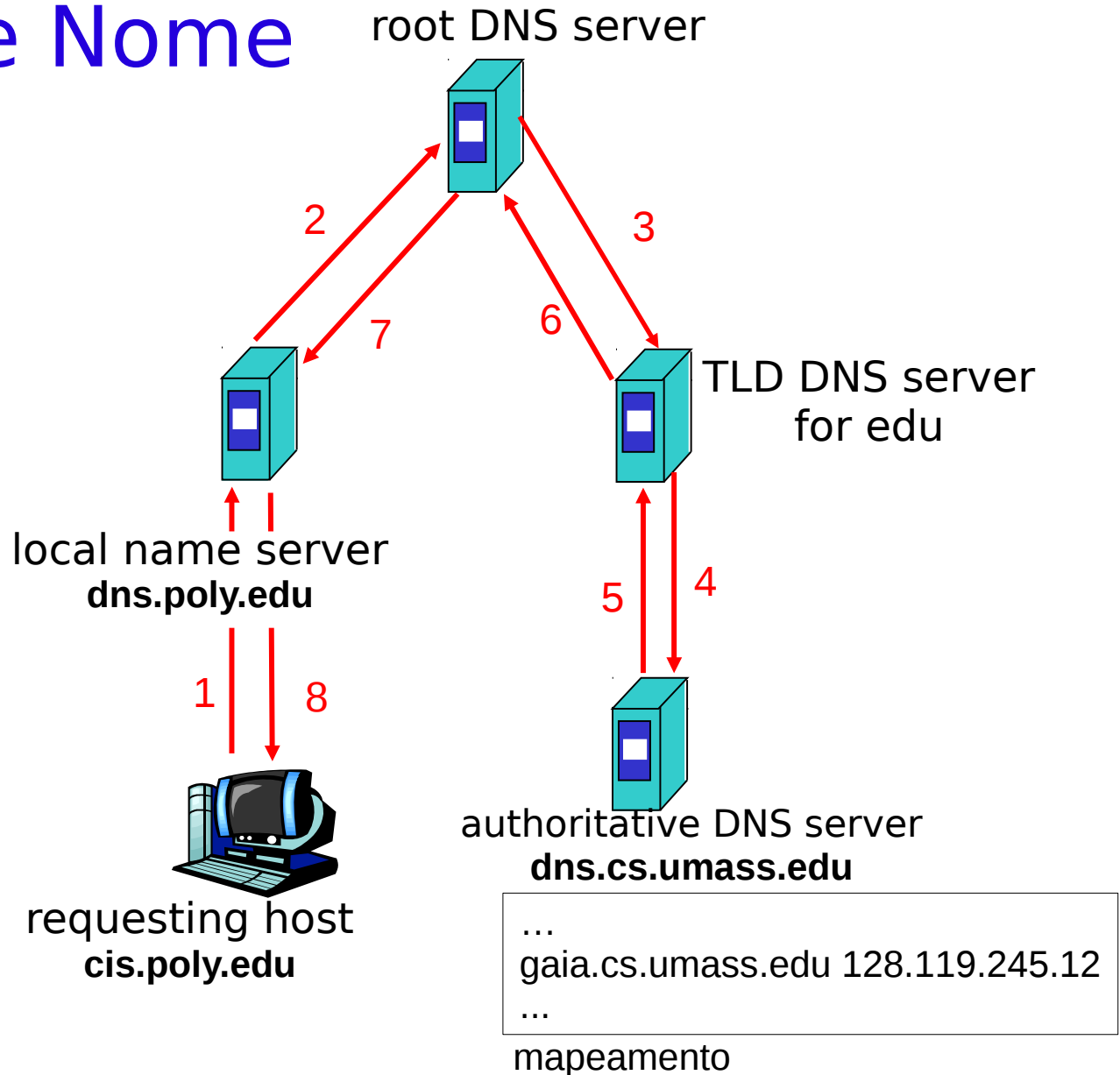


# Exemplo de Resolução de Nome

## Consulta recursiva:

Servidor repassa a consulta para outro servidor e aguarda resposta

“Eu não conheço este nome, mas vou descobrir e te responder”



# DNS: caching e atualização

- *Caching* usado amplamente por todos servidores DNS
  - servidor armazena mapeamento e responde com valores armazenados, finalizando a consulta
- IP dos TLDs tipicamente *cacheado* nos local name servers
  - diminui carga nos root name servers
- Cópias em cache podem estar desatualizadas
  - valores cacheados expiram depois do TTL
- Mecanismo de controle de cache (TTL, redundância) projetado pelo IETF (RFC 2136)

***Caching é fundamental!***

# DNS: História de Sucesso

- Um dos sistemas mais importantes da Internet
  - e consequentemente do mundo!
- Pilar da Internet: sem DNS, Internet pára
  - usado para muitas outras funcionalidades
- Sofre ataques constantemente
  - mas não consegue ser derrubado

**Sistema distribuído verdadeiramente escalável e robusto!**

- Brincar com o programa *nslookup* (faz queries DNS)

# Distribuindo Conteúdo



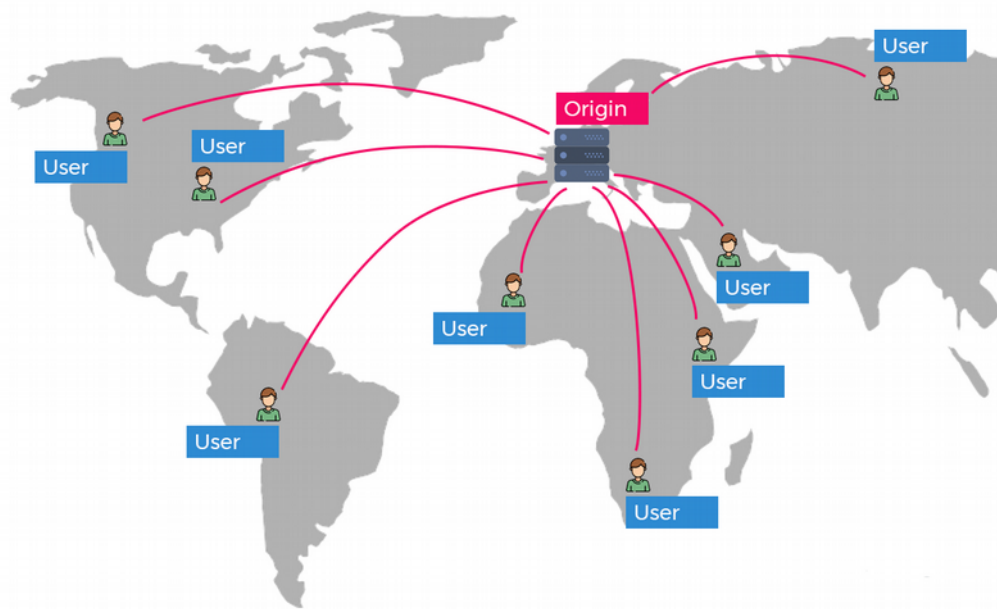
- Como distribuir conteúdo para milhares de pessoas simultaneamente?
  - ex. biblioteca com milhões de filmes

- **Ideia 0:** um único, “mega servidor”
  - ponto único de falha
  - ponto de congestionamento de rede
  - caminho longo até cliente distantes
  - várias cópias do mesmo conteúdo enviadas pelo mesmo enlace
- Esta solução não funciona para atender milhões de clientes

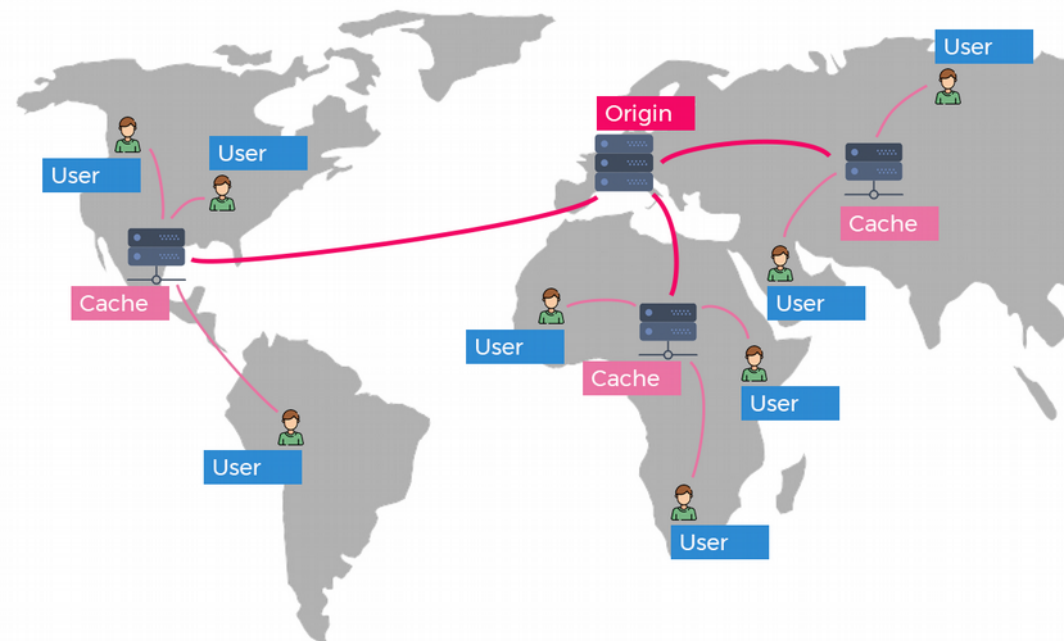
# Distribuindo Conteúdo

- **Ideia 1:** armazenar e servir conteúdo a partir de múltiplos servidores em diferentes locais geográficos

Ideia 0



Ideia 1



- Replicar conteúdo e armazenar mais próximo dos clientes: *Content Distribution Network (CDN)*

# Content Distribution Network

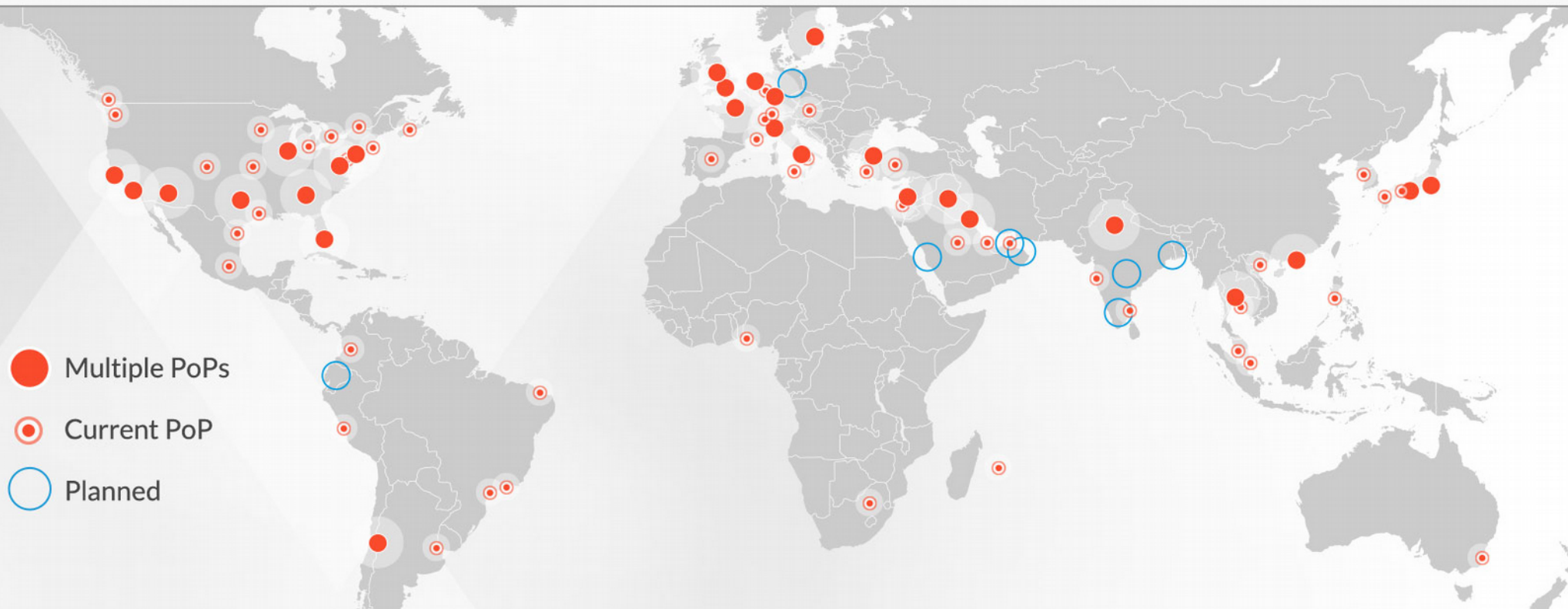
- Dois paradigmas para construir uma CDN
  - *enter deep*: instalar servidores CDN em muitas redes de acesso
    - mais próximo dos usuários
    - usado pela Akamai, presente em 4000+ localidades em 130+ países
  - *bring home*: menor número de cluster de servidores instalados nos POPs
    - próximo das redes de acesso
    - usado pela Limelight, presente em 135 POPs

# Limelight

- Um único sistema autônomo (AS) da Internet
- rede global com servidores em diferentes POPs
- troca tráfego com outras redes (ASes) nos POPs, incluindo Rio e São Paulo

Limelight's Global Private Network

ASN 22822

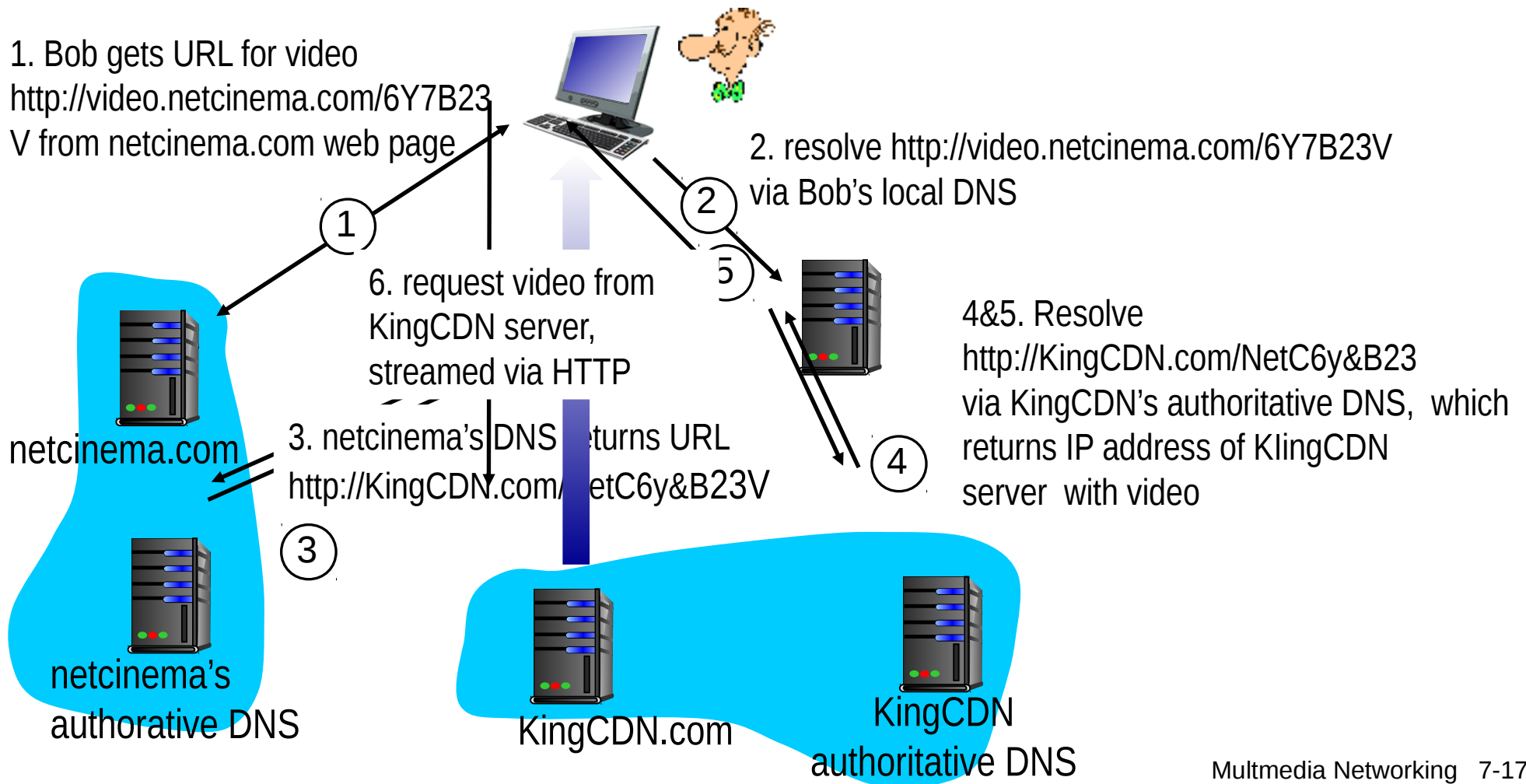




# Exemplo de Acesso via CDN

Bob (cliente) solicita video em `http://netcinema.com`

- video armazenado na CDN em `http://KingCDN.com/NetC6y&B23V`



# Seleção de Servidor CDN

- *desafio*: como escolher um bom servidor CDN para um cliente específico?
  - escolher servidor CDN mais próximo geograficamente do cliente
  - escolher servidor CDN com menor latência de rede até o cliente
- Como determinar esses servidores?
  - 1) geolocalização de endereços IPs, 2) *ping* entre servidores CDN e servidores de nome de ISPs
- *alternativa*: deixar cliente (aplicativo) decidir – DNS retorna lista de servidores CDN
  - cliente pode escolher o melhor (ex. estimar latência com ping)

# Estudo de Caso: Netflix

- 30% tráfego de downstream nos EUA em 2011
- Possui muito pouca infraestrutura, utiliza serviços de terceiros
  - servidores para inscrição e pagamento
- Utiliza serviços de nuvem da Amazon
- Faz upload *master* para nuvem da Amazon
- Cria multiplas versões do vídeo na nuvem
- Upload dos vídeos da nuvem para CDNs
- Serviços web da Netflix na nuvem Amazon (login, navegar, escolher)
- Contrato com 3 CDNs para distribuir vídeos: Akamai, Limelight, Level-3

# Estudo de Caso: Netflix

