

Sistemas Distribuídos

Aula 8

Aula passada

- Arquitetura de sistemas
- Arquitetura de sistemas distribuídos
- Cliente/servidor

Aula de hoje

- DNS
- CDN

Mapeando Nomes

- Pessoas possuem muitos identificadores
 - nome, CPF, passaporte, DRE, etc
- Na Internet, tudo que está conectado possui um identificador único (endereço IP)
 - usado para localizar dispositivo
- Humanos: nomes; computadores: números
 - `www.ufrj.br` → `146.164.170.36`



- Sistema que traduz nome para número?
- Como isto é feito na Internet?

Mapeando Nomes

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

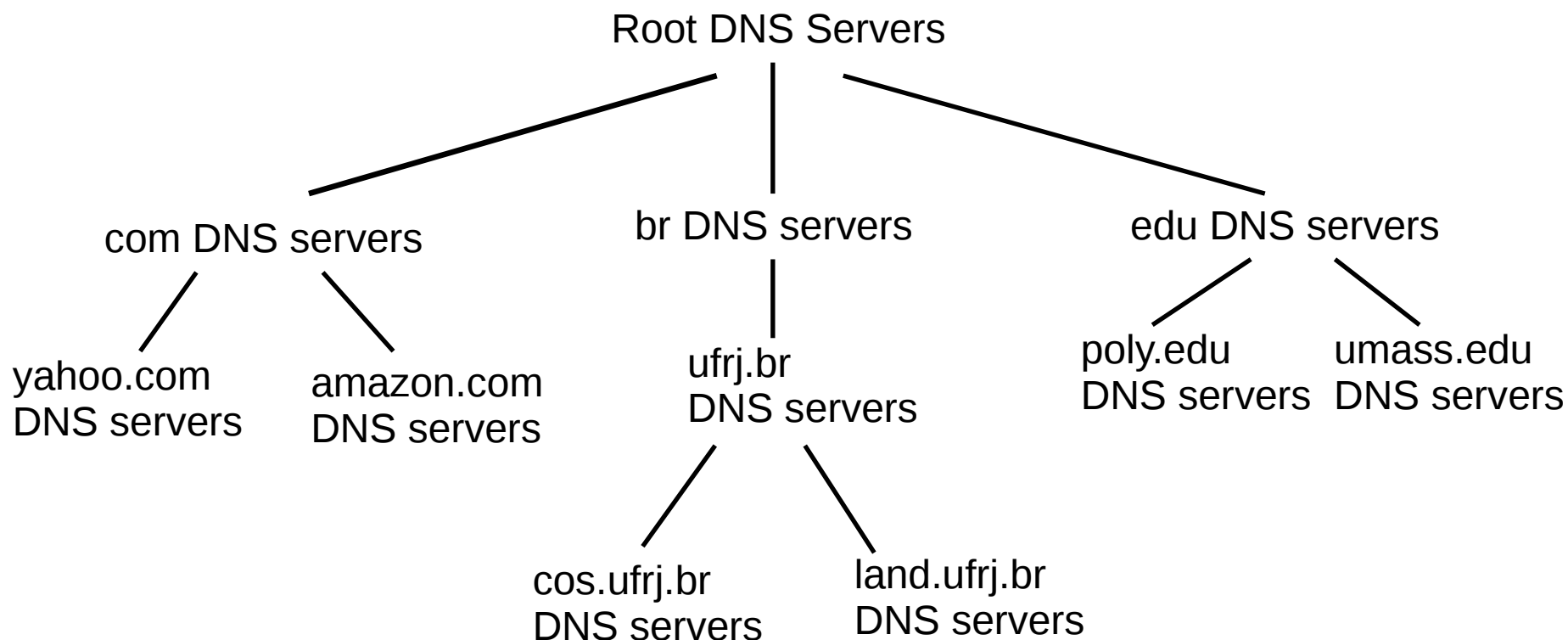


- Problemas?
- Não possui escalabilidade
- Não é tolerante a falhas
- Como fazer controle de acesso
 - quem pode atualizar o que no BD?
- Quem é o responsável pelo sistema
 - quem paga a conta?
- Replicação do BD não resolve os últimos dois problemas

DNS: Domain Name System

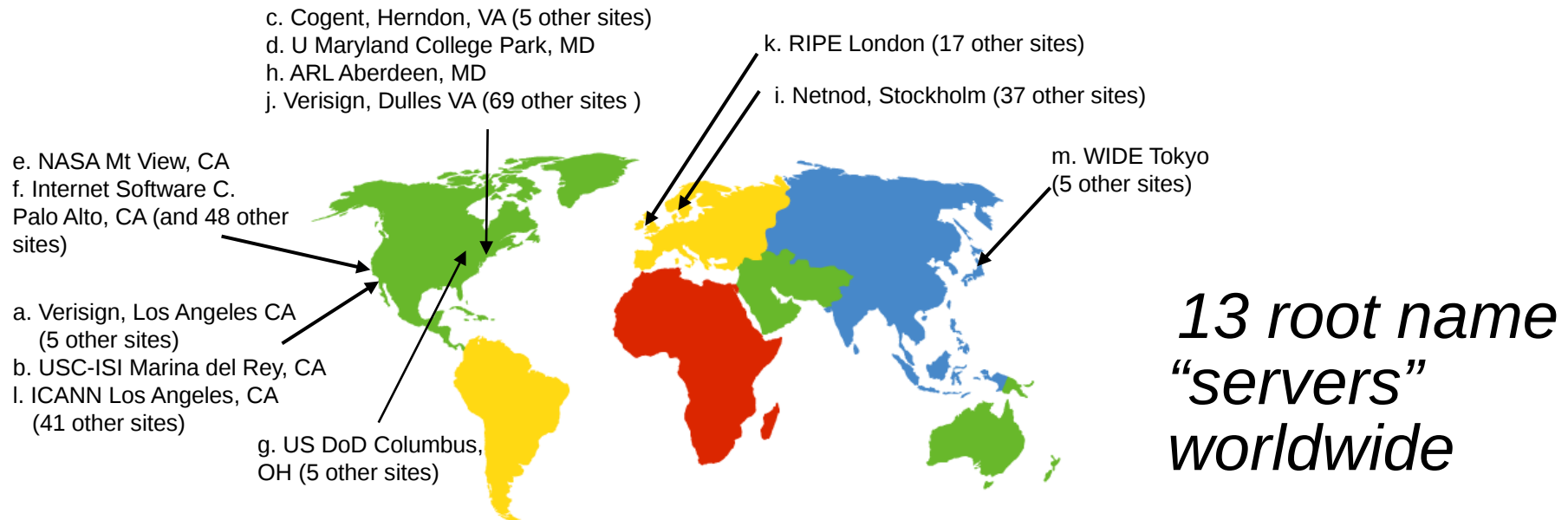
- Sistema de tradução de nomes (e serviços) distribuído, em muitos sentidos
 - projetado e implementado em 1983 (pequenas mudanças desde então)
 - mapeia www.land.ufrj.br → 146.164.47.193
- Hierarquia (árvore) de servidores de nomes (*name server*)
 - servidor de nome sabe endereço dos filhos (pelo nome)
 - folhas guardam mapeamento definitivo: *authoritative name servers*
 - cliente que inicia pedido: *local name server*

Hierarquia no DNS



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

DNS: Root name servers



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

TLD e Authoritative Servers

Top-level domain (TLD) servers:

- responsible for com, org, net, edu, etc, and all top-level country domains uk, fr, ca, jp, br.
- Network Solutions maintains servers for “com” TLD, Educause for “edu”, NIC.br for “br” TLD

Authoritative DNS servers:

- organization's DNS servers, providing authoritative hostname to IP mappings for organization's servers (e.g., Web, mail).
- can be maintained by organization or service provider or CDN

Local Name Server

- receives query from user's computer
- does not strictly belong to hierarchy
- each ISP (residential ISP, company, university) has one
- also called “default name server”
- when host makes DNS query, query is sent to its local DNS server
- acts as proxy, forwards query into hierarchy

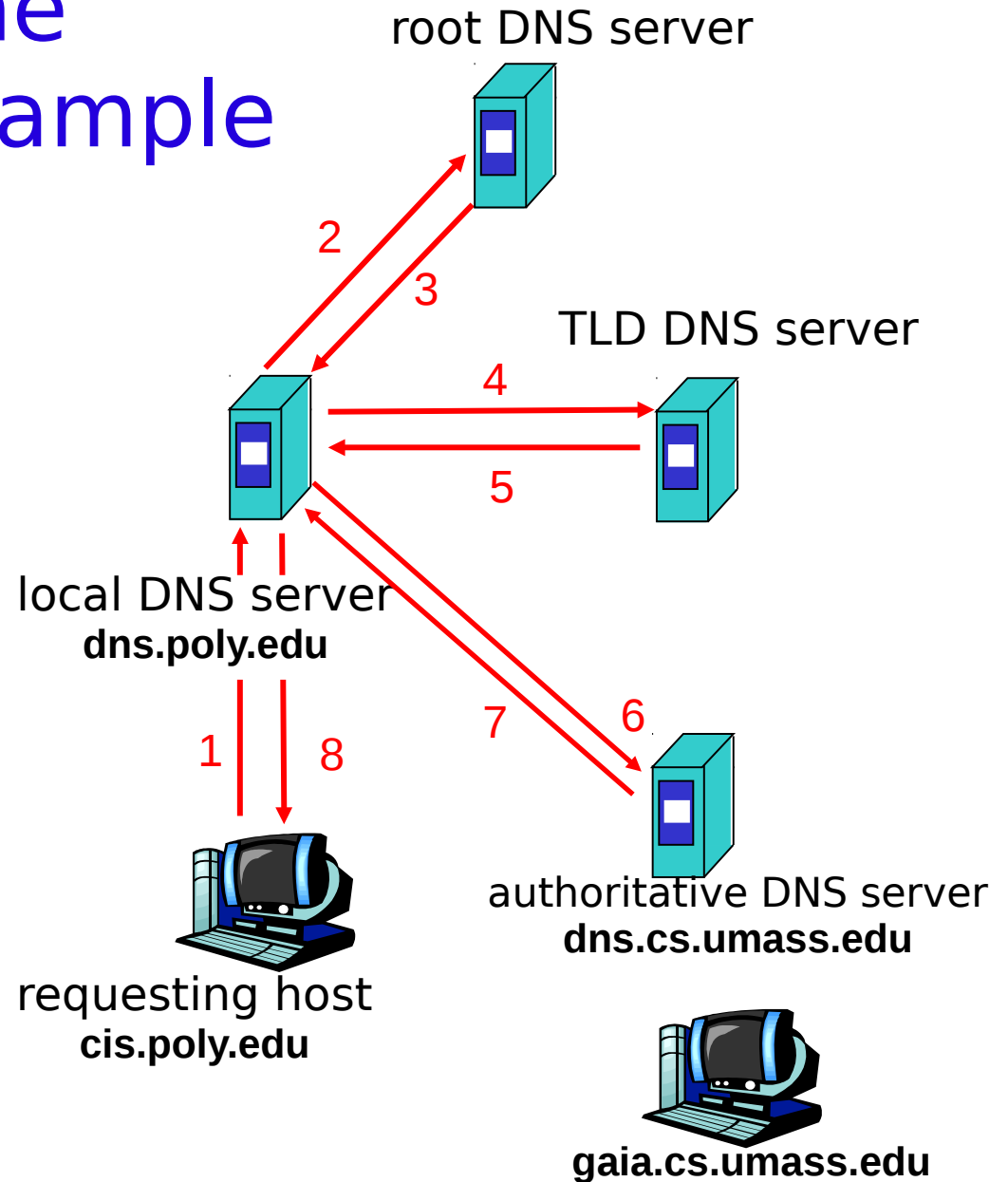
DNS name resolution example

Host at cis.poly.edu
wants IP address for
gaia.cs.umass.edu

iterated query:

contacted server replies
with name of server to
contact

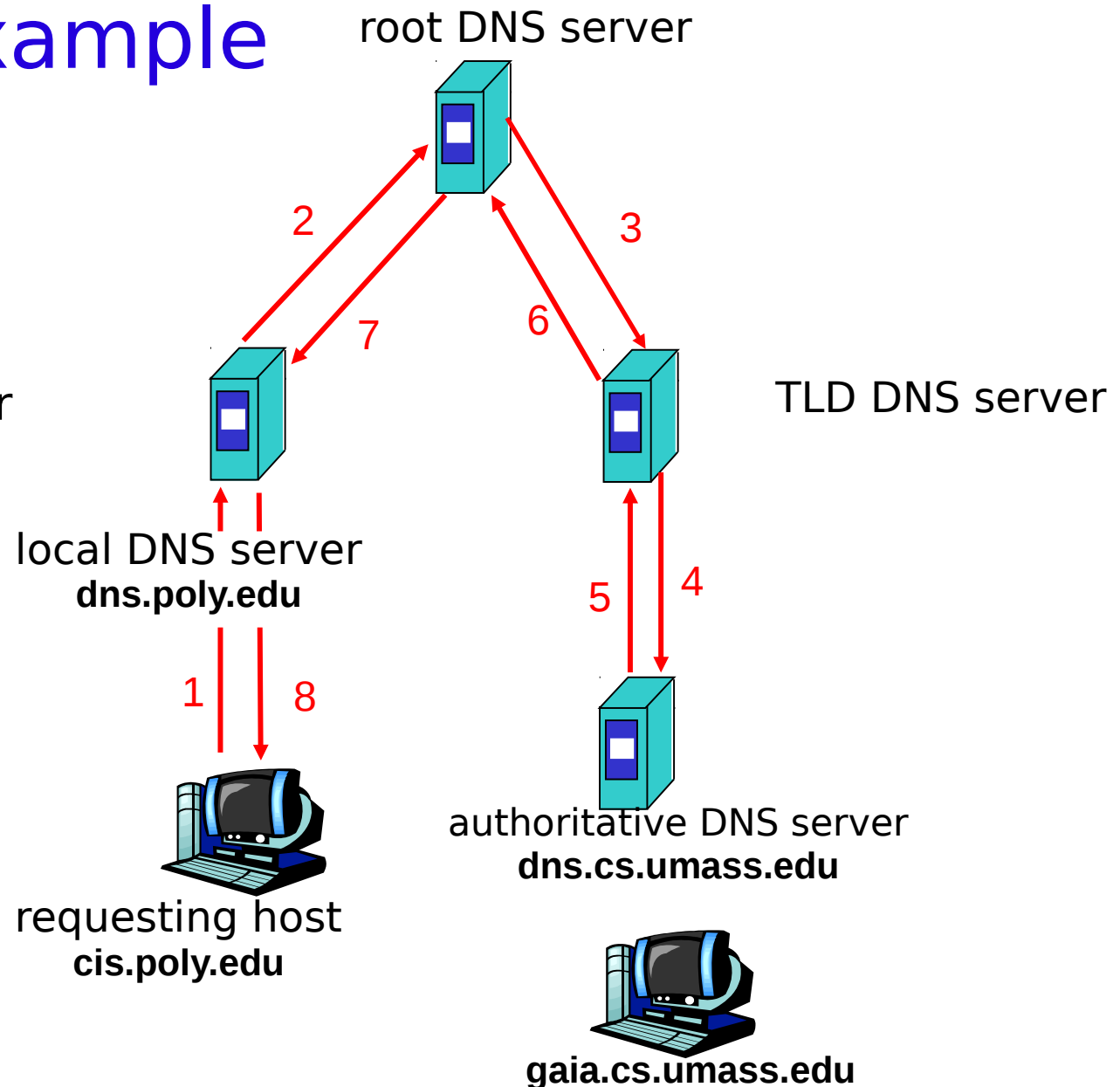
“I don’t know this name,
but ask this server”



DNS name resolution example

recursive query:

puts burden of name resolution on contacted name server
heavy load?



DNS: caching e atualização

- Caching (lembrar mapeamento) usado amplamente por todos servidores
 - entradas expiram depois de algum tempo (TTL)
- TLD tipicamente cached nos local name servers
 - diminui carga nos root name servers
- Cópias em cache podem estar desatualizadas
 - precisa esperar o TTL expirar, não é possível resetar!
- Mecanismo de controle de cache (TTL, redundância) projetado pelo IETF (RFC 2136)

DNS: História de Sucesso

- Um dos sistemas mais importantes da Internet
 - logo, do mundo!
- Coração da internet: sem DNS, internet para
 - usado para muitas funcionalidades, veremos em breve
- Sofre ataques constantemente!

Sistema distribuído verdadeiramente escalável e robusto

- Brincar com *nslookup* (faz queries DNS)

Distribuindo Conteúdo



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

- **Ideia 0:** único, “mega servidor”
 - single point of failure
 - point of network congestion
 - long path to distant clients
 - multiple copies of video sent over outgoing link

....quite simply: this solution *doesn't scale*

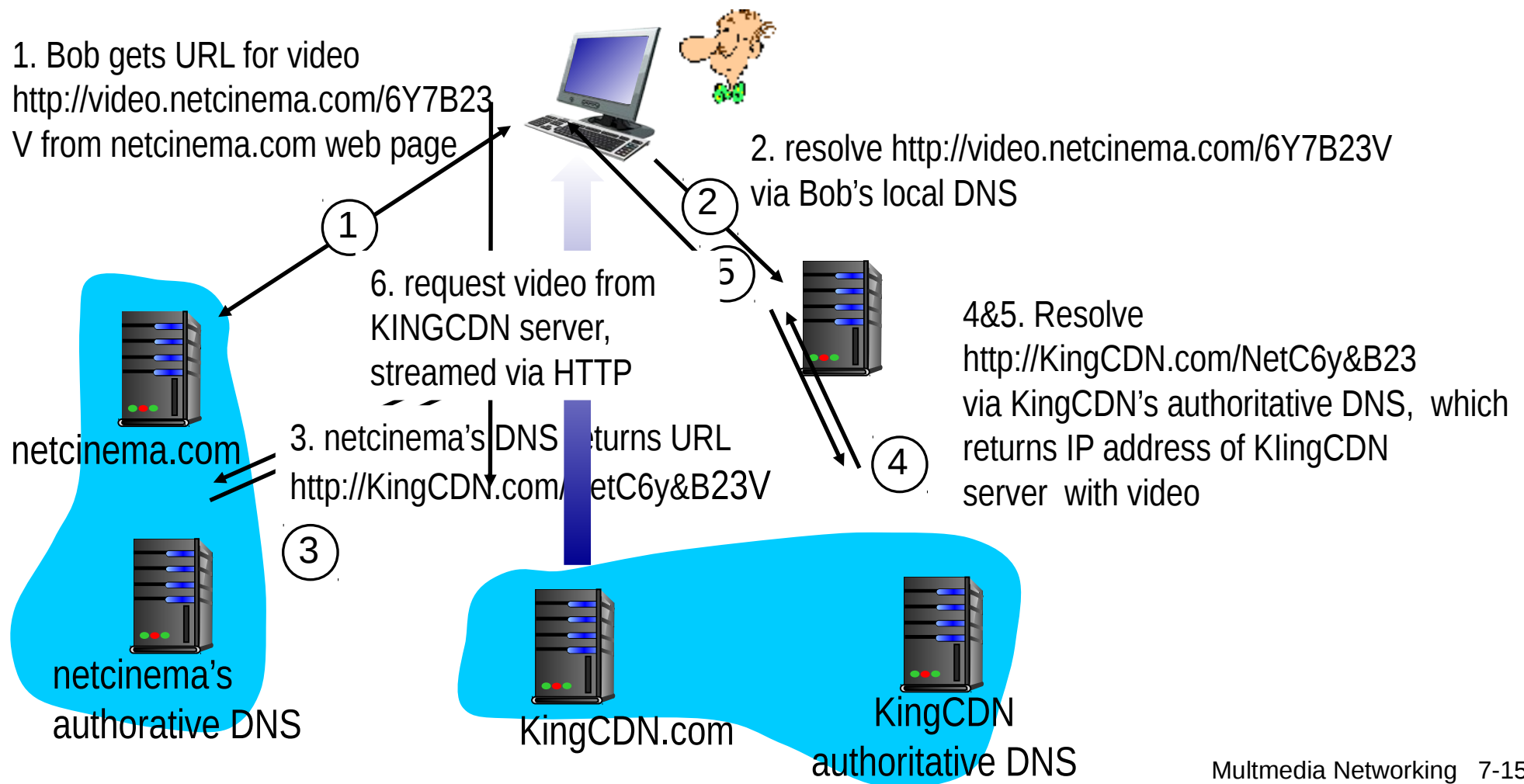
Distribuindo Conteúdo

- **Ideia 1:** store/serve multiple copies of videos at multiple geographically distributed sites (*CDN*)
 - *enter deep:* push CDN servers deep into many access networks
 - close to users
 - used by Akamai, 1700 locations
 - *bring home:* smaller number (10's) of larger clusters in POPs near (but not within) access networks
 - used by Limelight

CDN: “simple” content access scenario

Bob (client) requests video at <http://netcinema.com>

- video stored in CDN at <http://KingCDN.com/NetC6y&B23V>



CDN: cluster selection strategy

- *challenge*: how does CDN DNS select “good” CDN node to stream to client
 - pick CDN node geographically closest to client
 - pick CDN node with shortest delay (or min # hops) to client (CDN nodes periodically ping access ISPs, reporting results to CDN DNS)
- *alternative*: let *client* decide - give client a list of several CDN servers
 - client pings servers, picks “best”
 - Netflix approach

Case study: Netflix

- 30% downstream US traffic in 2011
- owns very little infrastructure, uses 3rd party services:
 - own registration, payment servers
 - Amazon (3rd party) cloud services:
 - Netflix uploads studio master to Amazon cloud
 - create multiple version of movie (different encodings) in cloud
 - upload versions from cloud to CDNs
 - Cloud hosts Netflix web pages for user browsing
 - *three* 3rd party CDNs host/stream Netflix content: Akamai, Limelight, Level-3

Case study: Netflix

