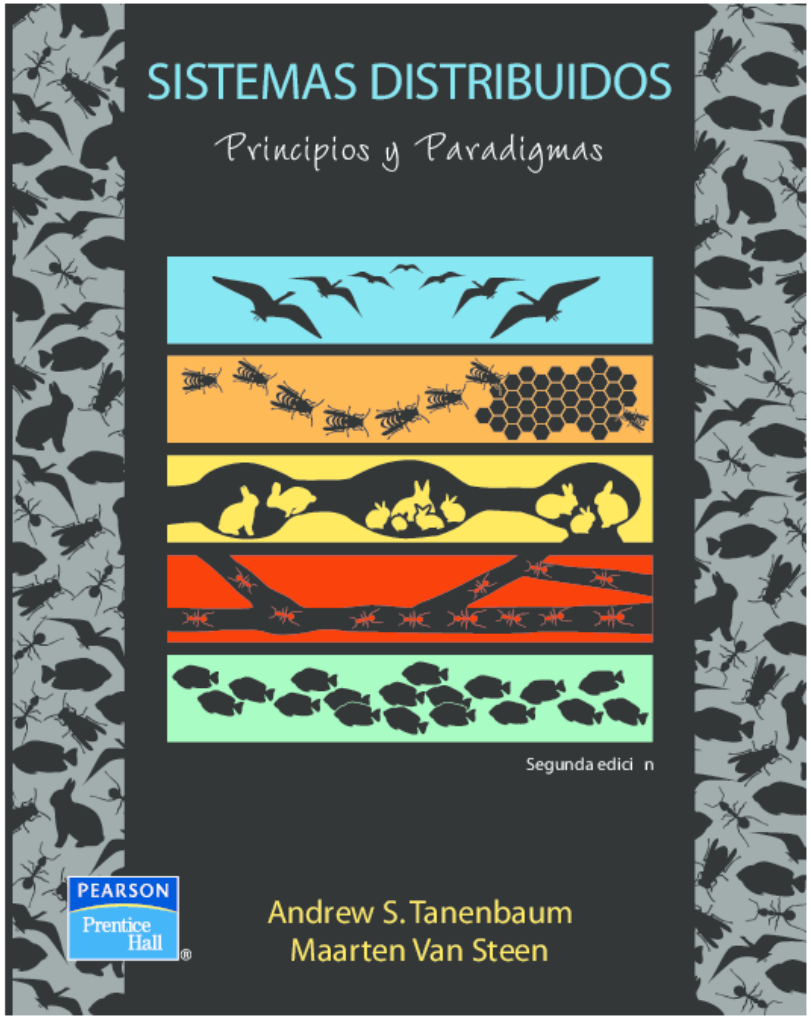
Computação Distribuída

Nomes

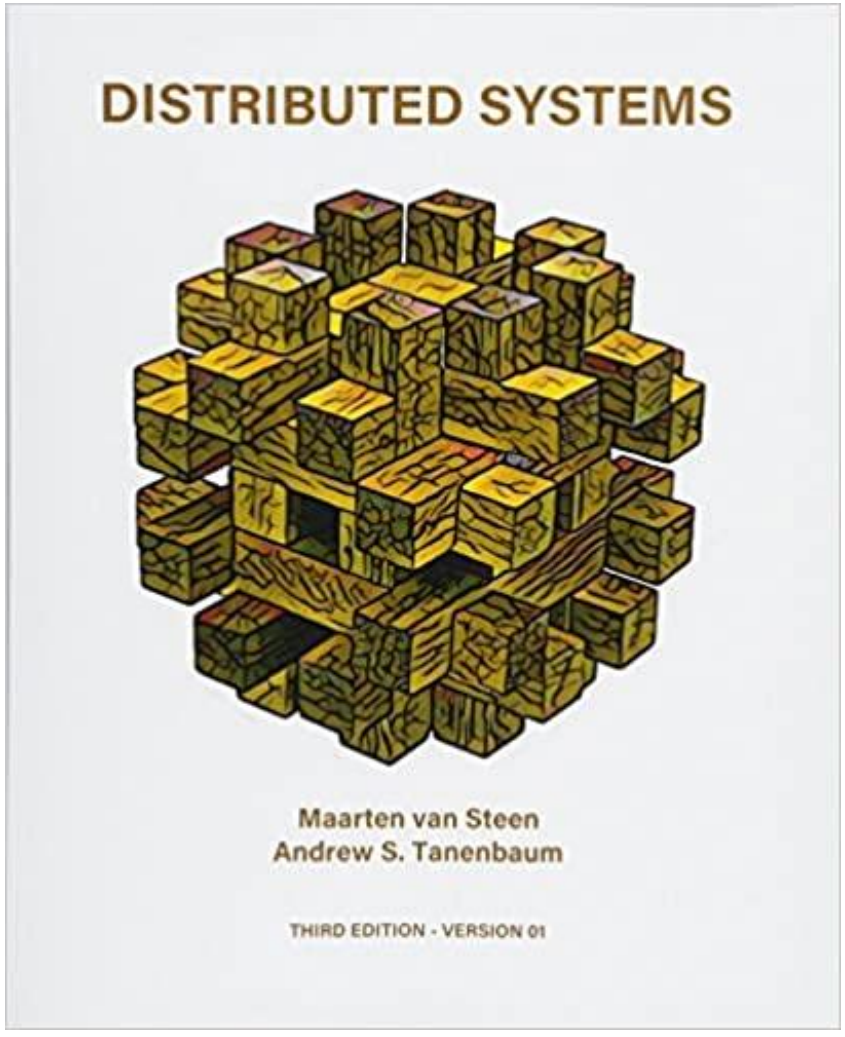
CHAPTER 5

NAMING

Vladimir Rocha (Vladi)
CMCC - Universidade Federal do ABC


SISTEMAS DISTRIBUIDOS
Principios y Paradigmas
Segunda edici n
PEARSON
Prentice
Hall
Andrew S. Tanenbaum
Maarten Van Steen

2007


DISTRIBUTED SYSTEMS
Maarten van Steen
Andrew S. Tanenbaum
THIRD EDITION - VERSION 01

2017

Disclaimer

- Estes slides foram baseados nos do professor **Emilio Francesquini** para o curso de Sistemas Distribuídos na UFABC.
- Este material pode ser usado livremente desde que sejam mantidos, além deste aviso, os créditos aos autores e instituições.
- Estes slides foram adaptados daqueles originalmente preparados (e gentilmente cedidos) pelo professor **Daniel Cordeiro, da EACH-USP** que por sua vez foram baseados naqueles disponibilizados online pelos autores do livro "Distributed Systems", 3ª Edição em: https://www.distributed-systems.net.

Agenda

- Definições
- Resolução de nomes (plano)
- Resolução de nomes (estruturado)
- Resolução de nomes (atributo)

Agenda

- Definições: nomes, identificadores e endereços
- Resolução de nomes (simples)
- Resolução de nomes (estruturado)
- Resolução de nomes (atributo)

Nomes

Essencialmente

Nomes são usados para denotar entidades em um sistema distribuído.

Para realizar operações em uma entidade, é preciso ter acesso a ela usando um ponto de acesso.

Pontos de acessos são identificados por um endereço.

Identificadores

Nomes "puros" ou planos

são nomes que não tem significado próprio; são apenas strings aleatórias. Nomes puros podem ser usados apenas para comparação.

Identificador

Um nome que possui as seguintes propriedades:

1. Cada identificador se refere a, no máximo, uma entidade
2. Cada entidade é referenciada por, no máximo, um identificador
3. Um identificador sempre se refere à mesma entidade (reutilização de identificadores é proibida)

Observação

Um identificador não necessariamente precisa ser um nome puro (pode ter conteúdo).

Agenda

- Definições: nomes, identificadores e endereços
- Resolução de nomes (plano)
- Resolução de nomes (estruturado)
- Resolução de nomes (atributo)

Espaço de nomes plano

Problema

Dado um nome não estruturado (ex: um identificador), como localizar seu ponto de acesso?

1. Solução simples (broadcasting / flooding / ponteiros)
2. Abordagens baseadas em um local pré-determinado (*home-based*)
3. Tabelas de hash distribuídas (P2P estruturado)
4. Serviço hierárquico de nomes

Soluções simples: broadcasting

Broadcasting

Solução: fazer o *broadcast* do ID, requisitando que a entidade devolva seu endereço

- Não escala para além de redes locais
- Requer que todos os processos escutem e processem os pedidos de localização

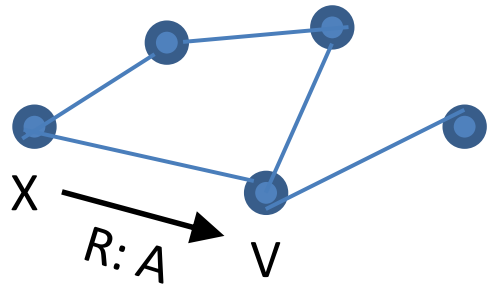
Address Resolution Protocol (ARP)

Para encontrar o endereço MAC associado a um endereço IP, faz o *broadcast* da consulta "quem tem esse endereço IP?" [Plumer/RFC 1982]

Soluções simples: flooding

Flooding (tanenbaum 2018. 3ed pp 85 – section 2.3)

Solução: um nó X envia a requisição R por um arquivo A a alguns vizinhos V. O nó V recebe e realiza o seguinte processo:



- Se V já viu a requisição antes, ignora.
- V procura localmente por A.
- Se V não tem o arquivo:

  encaminha R para um dos seus vizinhos
- Se V tem o arquivo:

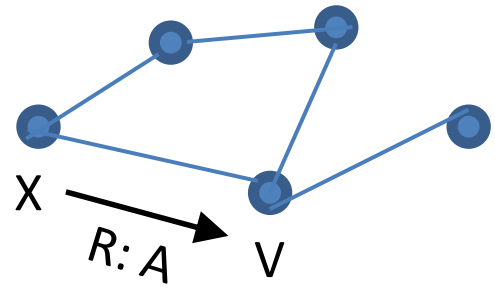
  devolve diretamente para X ou

  devolve para quem fez (encaminhou) a requisição

Soluções simples: flooding

Flooding (tanenbaum 2018. 3ed pp 85 – section 2.3)

Solução: um nó X envia a requisição R por um arquivo A a alguns vizinhos V. O nó V recebe e realiza o seguinte processo:



- Se V já viu a requisição antes, ignora.
- V procura localmente por A.
- Se V não tem o arquivo:

  encaminha R para um dos seus vizinhos
- Se V tem o arquivo:

  devolve diretamente para X ou

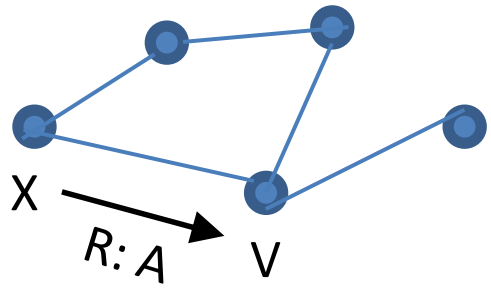
  devolve para quem fez (encaminhou) a requisição

Problema de escalabilidade (de tamanho e geográfica)

Soluções simples: flooding com TTL

Flooding (tanenbaum 2018. 3ed pp 85 – section 2.3)

Solução: um nó X envia a requisição R por um arquivo A a alguns vizinhos V. O nó V recebe e realiza o seguinte processo:



- Se V já viu a requisição antes, ignora.
- V procura localmente por A.
- Se V não tem o arquivo:

  Se TTL da requisição > 0

  encaminha R para um dos seus vizinhos com TTL-1
- Se V tem o arquivo:

  devolve diretamente para X ou

  devolve para quem fez (encaminhou) a requisição

Soluções simples: flooding com TTL

Flooding

[NullSoft 2000]

Gnutella é um sistema distribuído P2P para compartilhamento de arquivos.
A busca era realizada por 'flooding com TTL' com as seguintes características:

- Se V não tem o arquivo, encaminha para 7 vizinhos com TTL=10.
- Se V tem o arquivo, responde de forma inversa pelo mesmo caminho
- Não há loop nas requisições.
- Transferências são realizadas diretamente entre nós via HTTP

Soluções simples: flooding com TTL

Flooding

gnutella

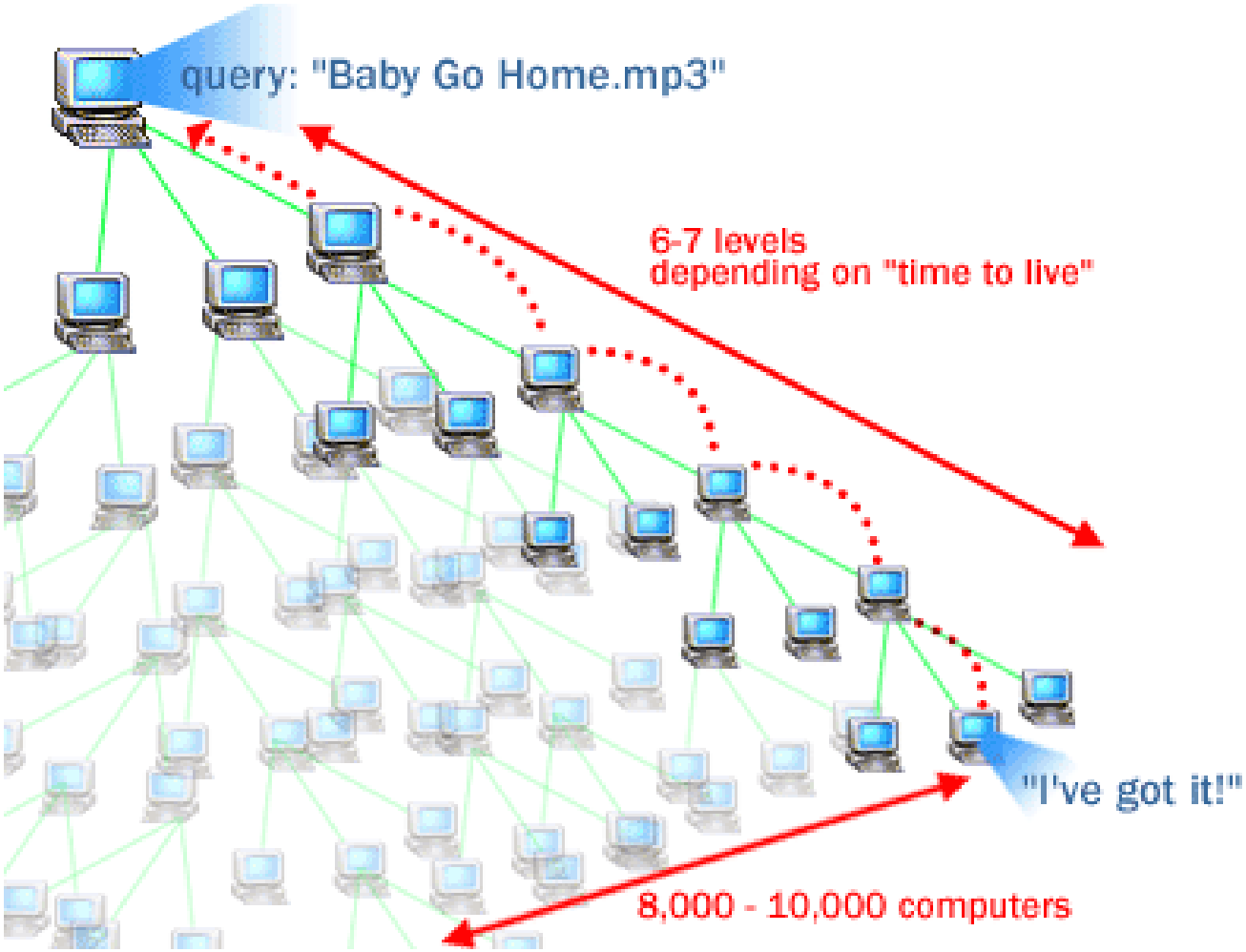


Figure from http://computer.howstuffworks.com/file-sharing.htm

Espaço de nomes plano

Problema

Dado um nome não estruturado (ex: um identificador), como localizar seu ponto de acesso?

1. Solução simples (broadcasting / flooding / ponteiros)
2. Abordagens baseadas em um local pré-determinado (*home-based*)
3. Serviço hierárquico de nomes
4. Tabelas de hash distribuídas (P2P estruturado)

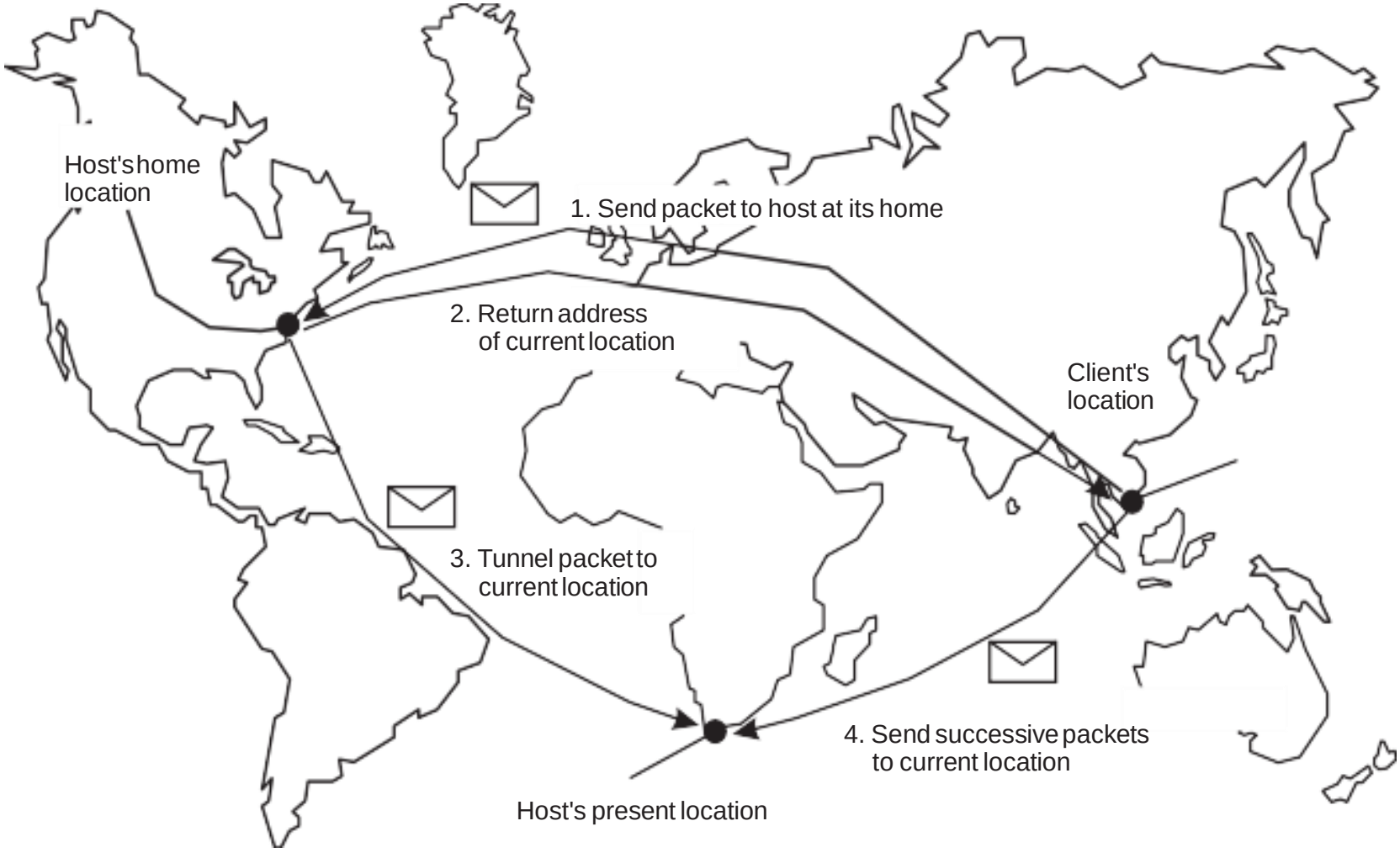
Abordagens baseadas em local pré-determinado

Faça com que um local fixo (sua "casa") sempre saiba onde a entidade está (ideias do Mobile IP) [Johnson 2004]:

- O endereço pré-determinado é registrado em um serviço de nomes
- O endereço mantém um registro do endereço externo da entidade
- O cliente primeiro contacta o endereço pré-determinado e depois continua para seu endereço externo

Abordagens baseadas em local pré-determinado

Princípio do endereçamento IP móvel:



Abordagens baseadas em local pré-determinado

Abordagem em dois níveis

Mantenha um registro das entidades que foram "visitadas":

- Verifique o endereço local primeiro
- Se falhar, só então vá para o endereço pré-determinado

Problemas

- O endereço pré-determinado deve existir enquanto a entidade existir
- O endereço é fixo e pode se tornar desnecessário se a entidade se mover permanentemente
- Escalabilidade geográfica ruim (a entidade pode estar do lado do cliente)

Pergunta: como resolver o problema da mudança permanente? (talvez com outro nível de endereçamento (DNS))

Espaço de nomes plano

Problema

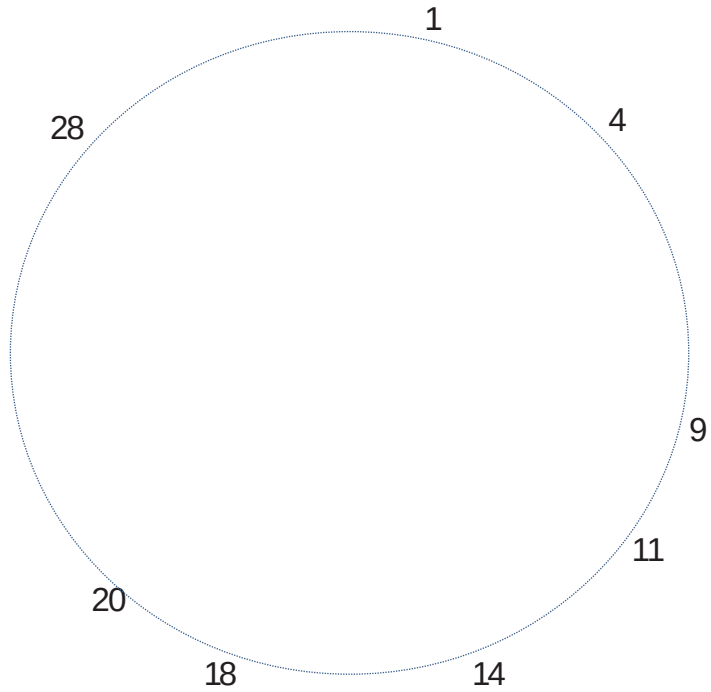
Dado um nome não estruturado (ex: um identificador), como localizar seu ponto de acesso?

1. Solução simples (broadcasting / flooding / ponteiros)
2. Abordagens baseadas em um local pré-determinado (*home-based*)
3. Tabelas de hash distribuídas (P2P estruturado)
3. Serviço hierárquico de nomes

Chord

Considere que os nós estejam organizados em forma de anel lógico [Stoica 2003]

- A cada nó é atribuído um identificador aleatório de *m*-bits
- A cada entidade é atribuída uma única chave de *m*-bits
- Entidades com chave *k* estão sob a jurisdição do nó com o menor *id* $\geq$ *k* (seu sucessor)



Busca por uma informação: solução ruim

Faça com que cada nó mantenha o registro de seus vizinhos e faça uma busca linear ao longo do anel

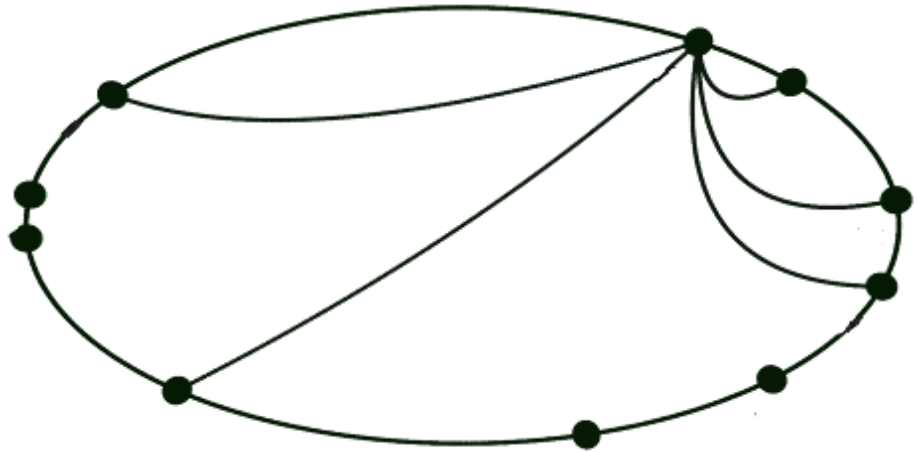
Notação

Chamamos de nó p o nó cujo identificador é p.

Como melhorar o desempenho da busca?

Shortcuts

DHTs: fingertables



Ideia:

- cada nó p mantém um finger table (shortcuts) FT_p [] com um máximo m entradas

$$FT_p[i] = succ(p + 2^{i-1})$$

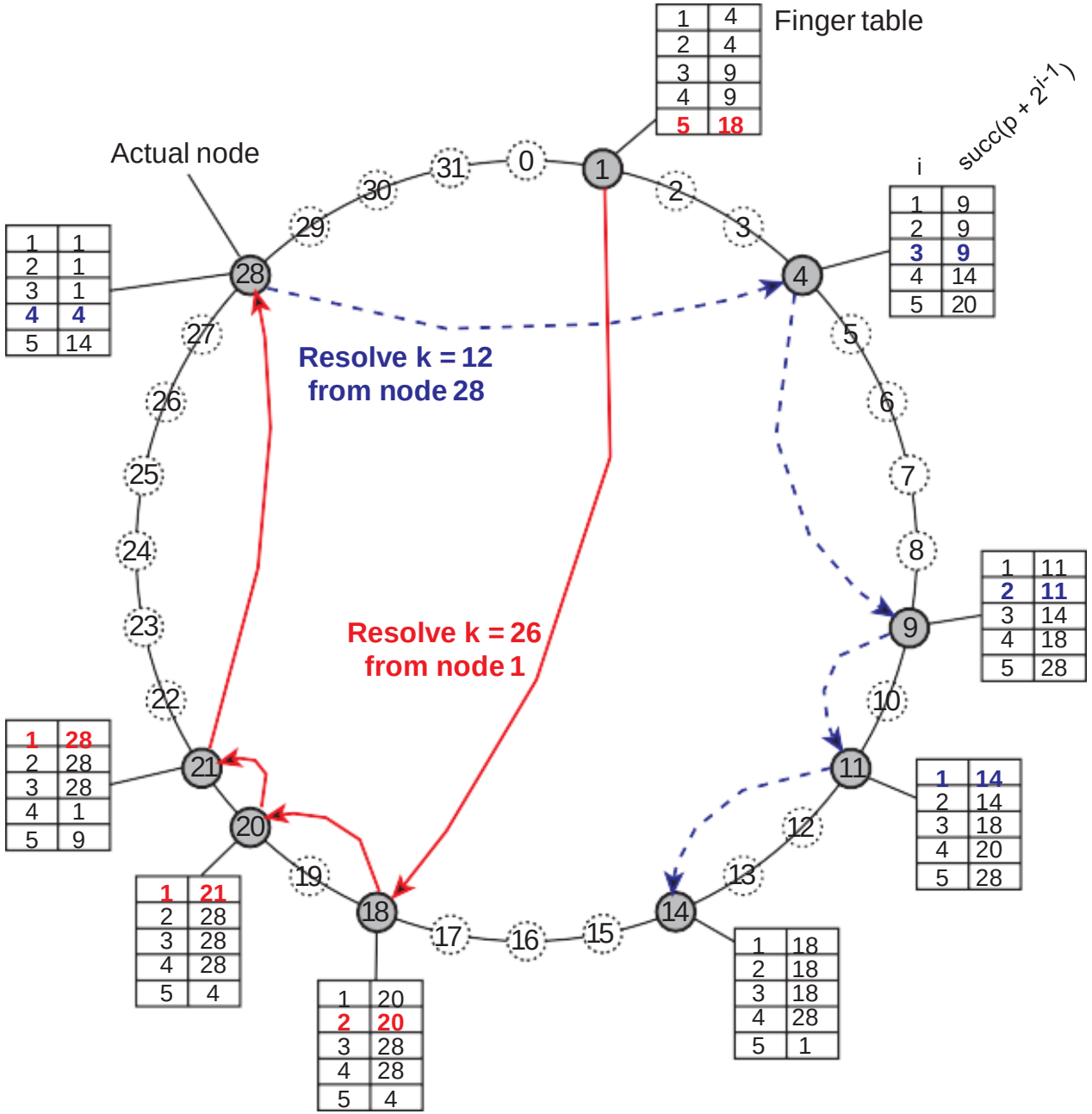
FT_p [i] aponta para o primeiro nó sucessor a p com distância de pelo menos 2^{i-1} posições (distância 2^0, 2^1, 2^2, 2^3, 2^4 ...)

- Para procurar por uma chave k, o nó p encaminha o pedido para o nó com índice j tal que:

$$q = FT_p[j] \leq k < FT_p[j+1]$$

- Se $p < k < FT_p$ [1], a requisição é encaminhada para FT_p [1]

DHTs: fingertables



Espaço de nomes plano

Problema

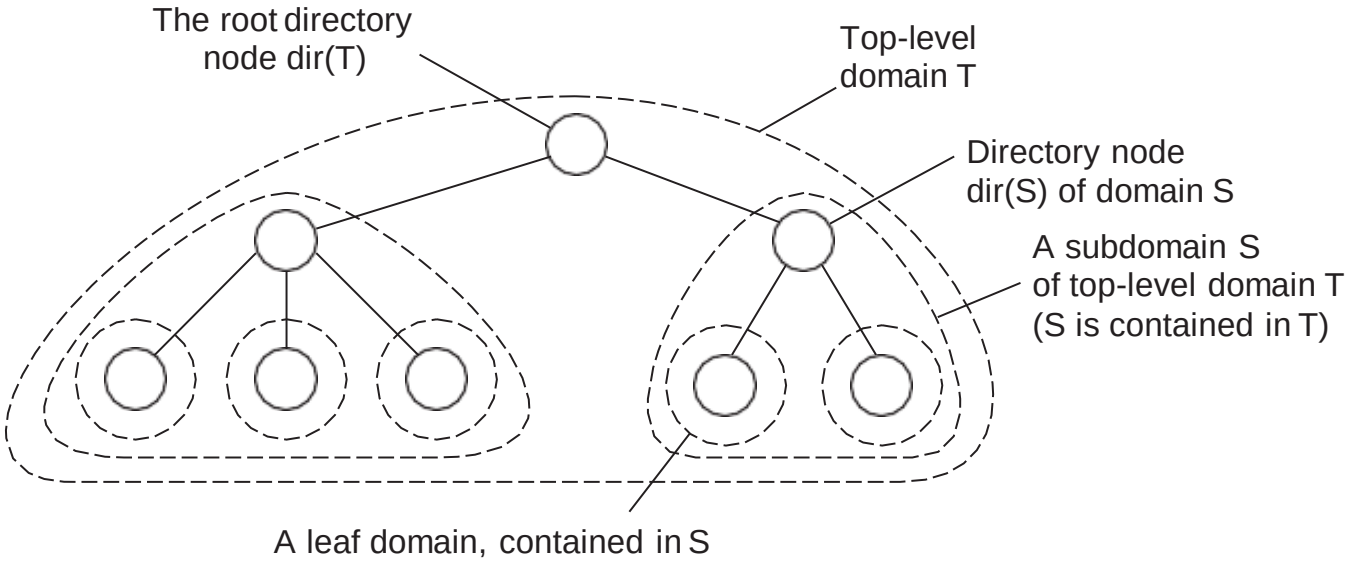
Dado um nome não estruturado (ex: um identificador), como localizar seu ponto de acesso?

1. Solução simples (broadcasting / flooding / ponteiros)
2. Abordagens baseadas em um local pré-determinado (*home-based*)
4. Tabelas de hash distribuídas (P2P estruturado)
4. Serviço hierárquico de nomes

Serviços de localização hierárquicos (HLS)

Ideia:

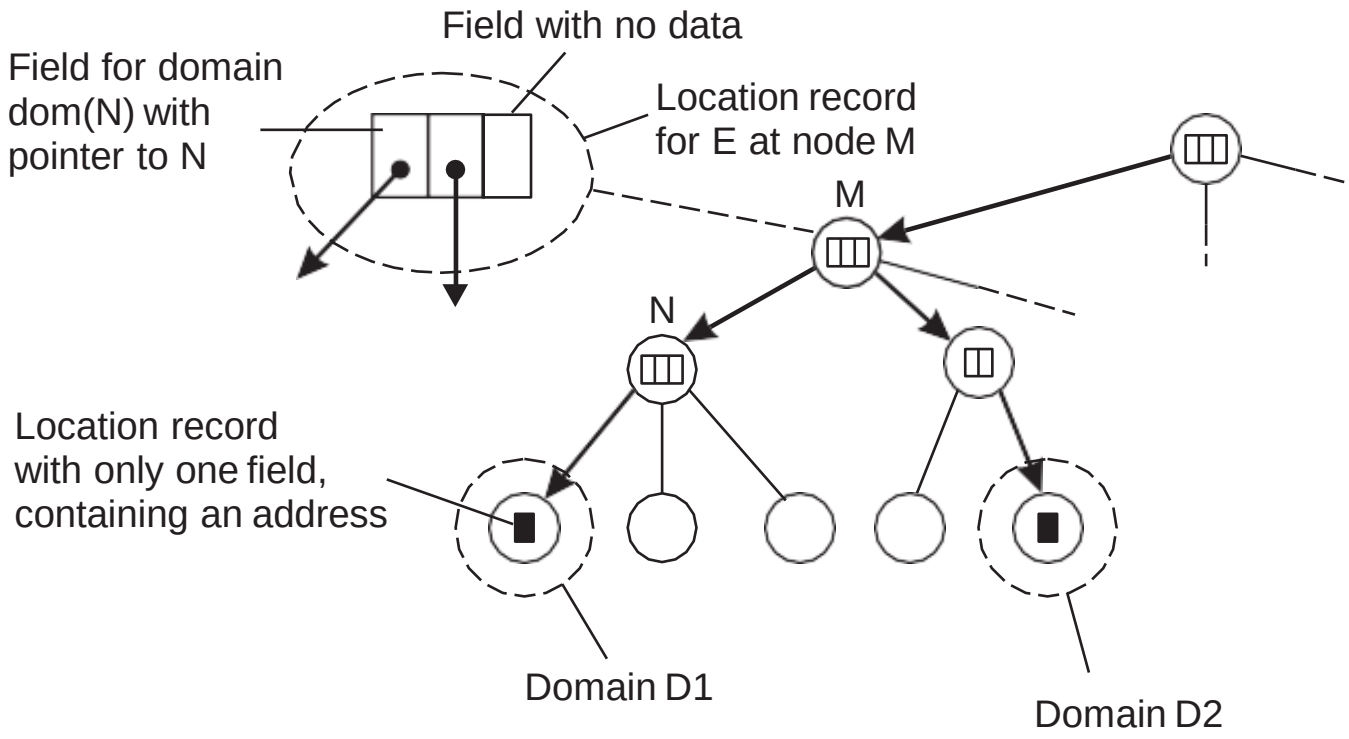
Construir uma árvore de busca em larga escala [van Steen 1998], onde a rede é dividida em domínios hierárquicos.
Cada domínio é representado por um diretório de nós separado.



HLS: organização em árvores

Invariantes

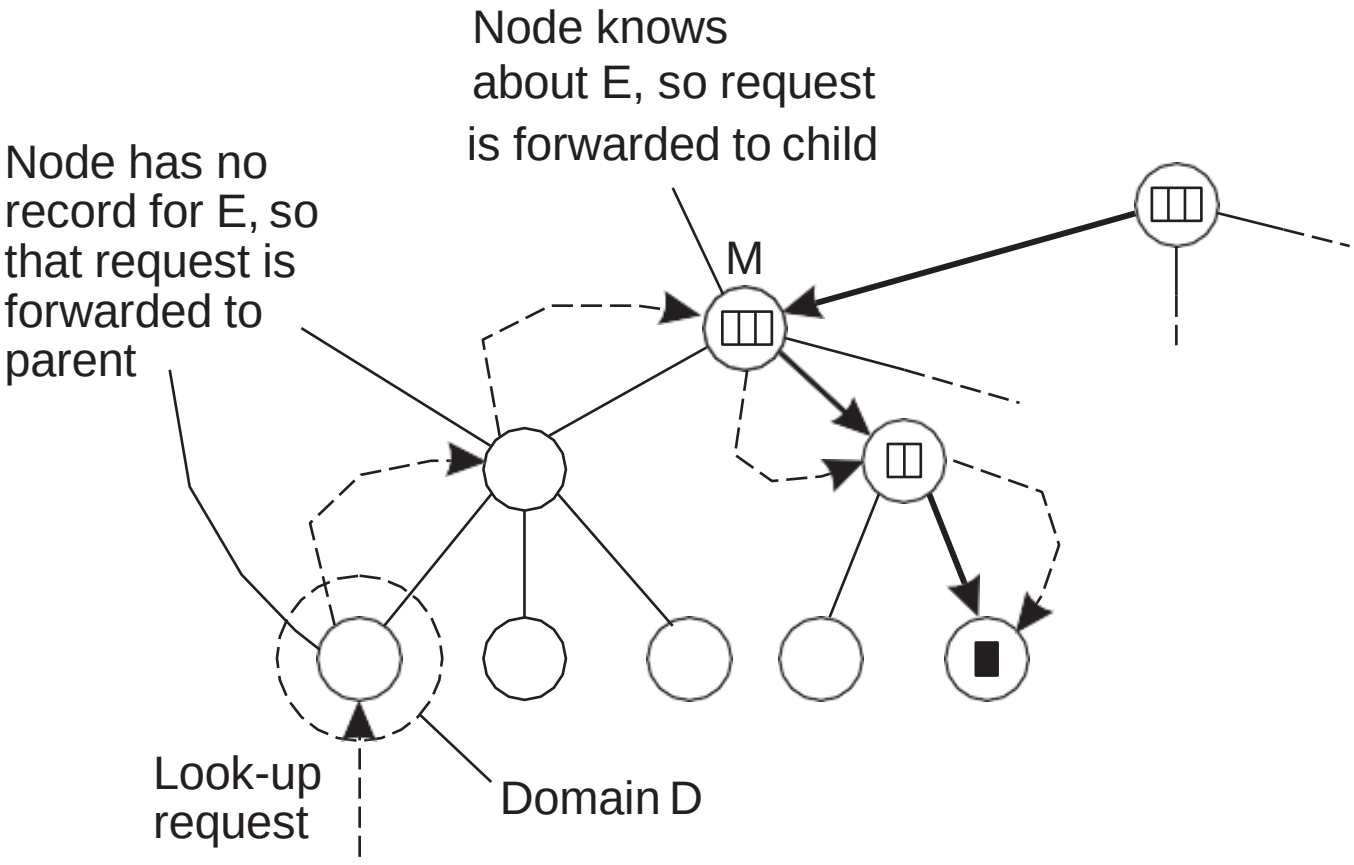
- Endereço da entidade *E* é armazenado numa folha ou nó intermediário
- Nós intermediários contém um ponteiro para um filho *sse* a subárvore, cuja raiz é o filho, contenha o endereço da entidade
- A raiz conhece todas as entidades



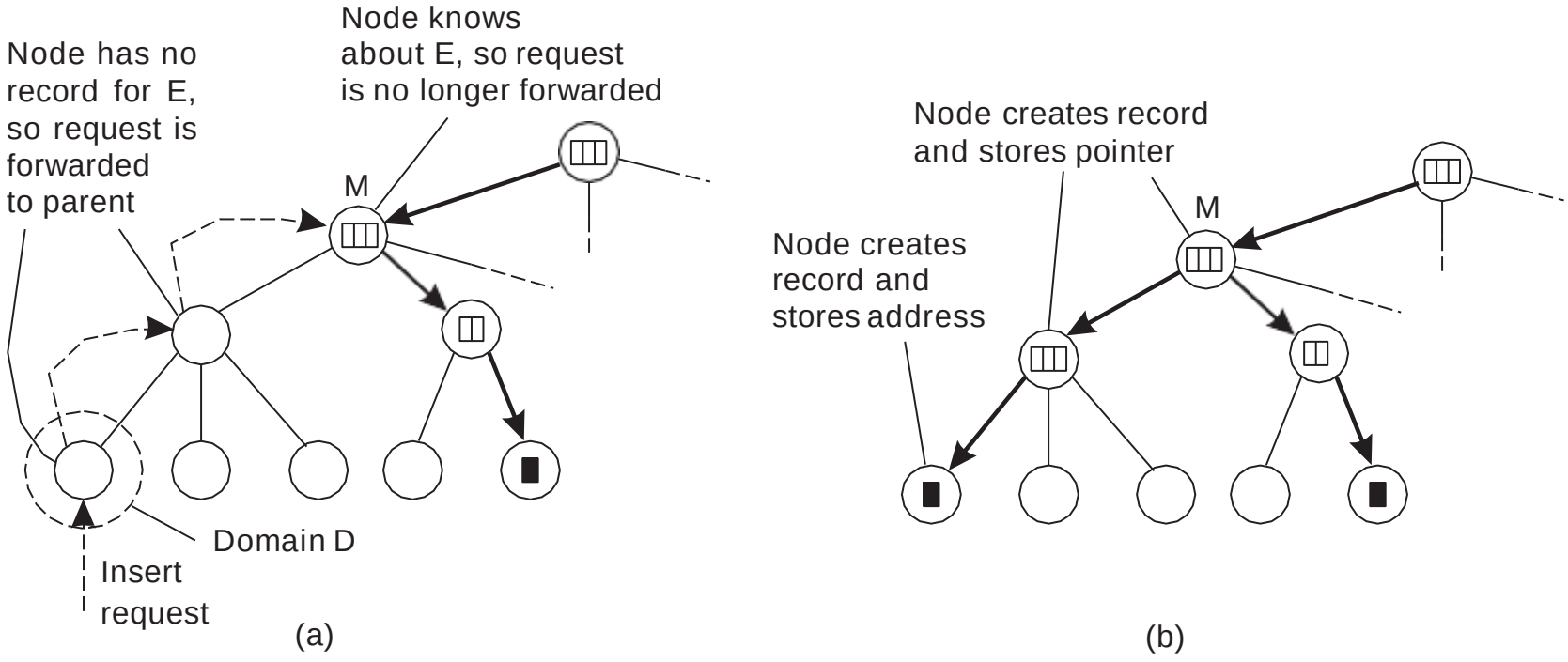
HLS: consulta

Princípios básicos:

- Comece a busca pelo nó folha local
- Se o nó souber sobre *E*, seguir o ponteiro pra baixo, senão suba
- Continue subindo até encontrar a raiz



HLS: inserção



(a) um pedido de inserção é encaminhado ao primeiro nó que conhece a entidade E

(b) uma cadeia de ponteiros de redirecionamento é criada até o nó folha

Agenda

- Definições: nomes, identificadores e endereços
- Resolução de nomes (simples)
- Resolução de nomes (estruturado)
- Resolução de nomes (atributo)

DNS: Implementação de espaços de nomes

A resolução de um nome é o mecanismo para recuperar a entidade que possui o nome

Problema:

Distribuir o processo de resolução de nomes, bem como o gerenciamento do espaço de nomes, em várias máquinas, de forma a distribuir o grafo de nomes para garantir **escalabilidade e disponibilidade.**

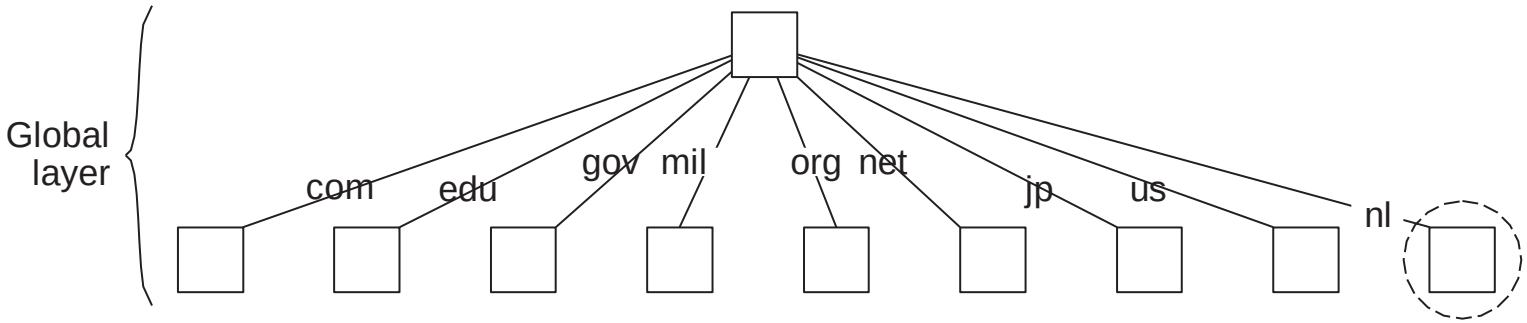
Exemplo a ser analisado:

DNS – Domain Name System [RFC em 1983, implementado em 1984] (transforma um nome a um IP)

DNS: Distribuição

Três níveis distintos:

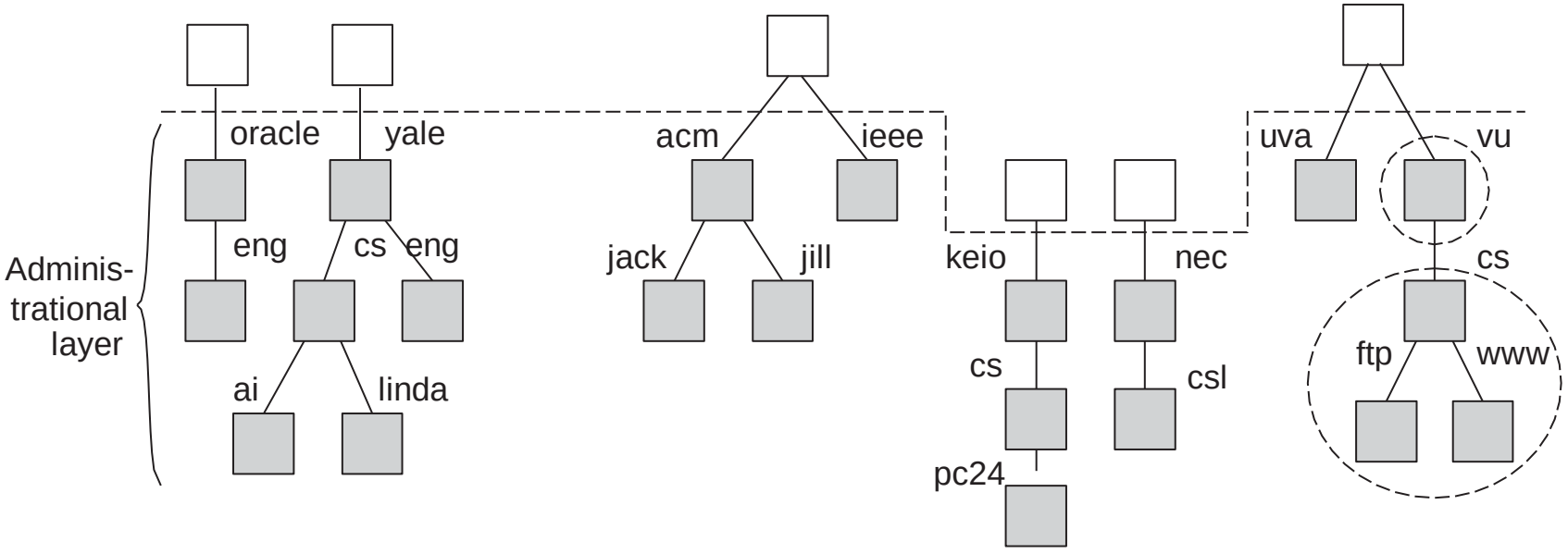
1. Nível global: consiste de um diretório de nós de alto nível. Suas principais características são que os nós do diretório devem ser gerenciados em conjunto por **diferentes administradores** e que há uma relativa **estabilidade nos nomes**, ou seja, as entradas do diretório são modificadas muito raramente



DNS: Distribuição

Três níveis distintos:

2. Nível de administração: nós de diretório de nível intermediário. Podem ser agrupados e **cada grupo pode ser responsabilidade de um administrador diferente.** Apesar de estáveis, mudam com mais frequência do que entradas no nível global

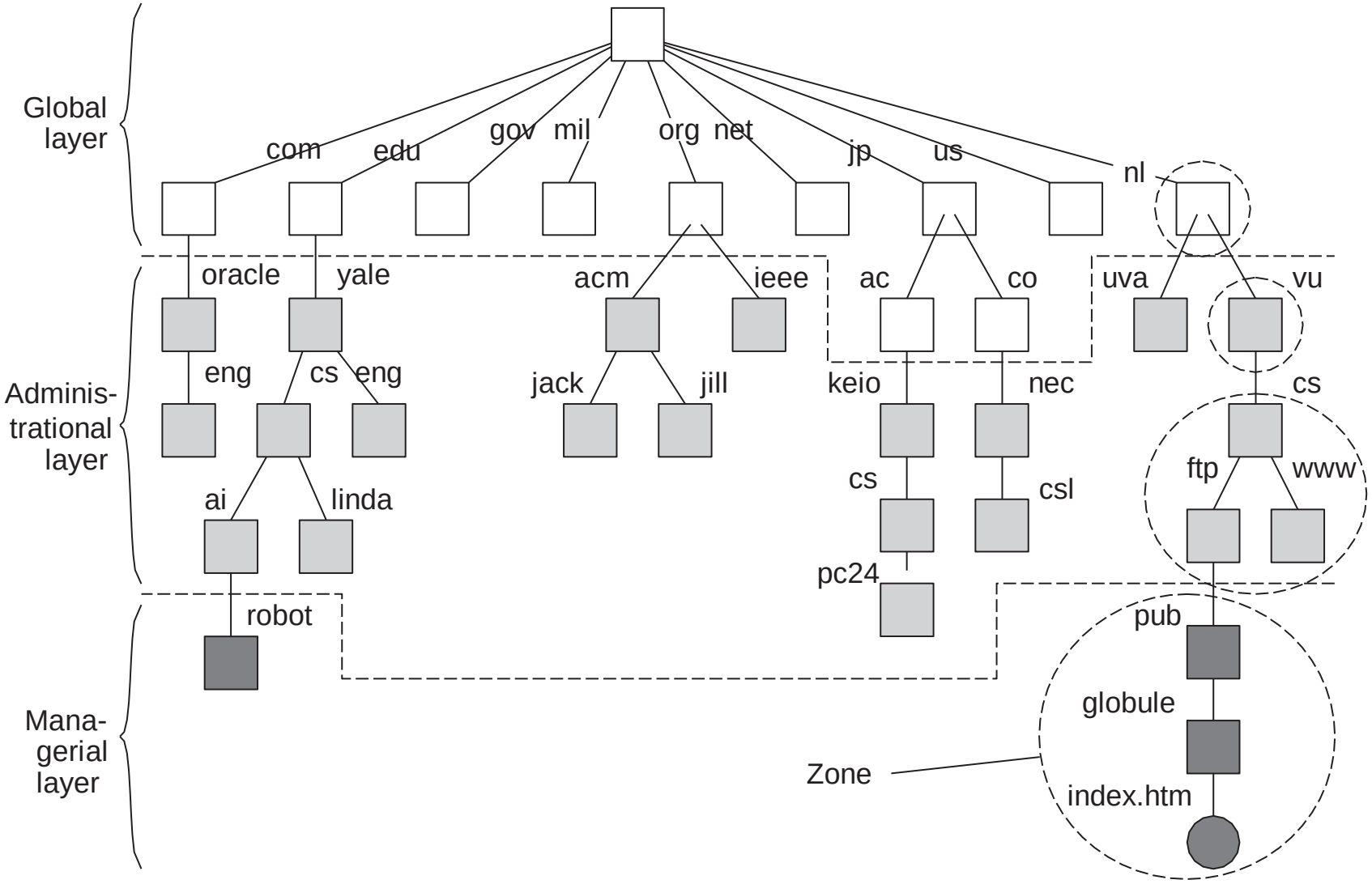


DNS: Distribuição

Três níveis distintos:

3. Nível gerencial: nós de nível inferior que **pertencem a um único administrador.** O problema principal é mapear os nós de diretório aos servidores de nomes local. **Mudanças regulares** ocorrem neste nível.

DNS: Distribuição



DNS: Comparação

Item	Global	Administração	Gerencial
1	Mundial	Organização	Departamento
2			
3			
4			
5			
6			
1: Escala geográfica 2: # Nós 3: Responsividade		4: Propagação de atualizações 5: # Réplicas 6: Cache no lado do cliente?	

DNS: Comparação

Item	Global	Administração	Gerencial
1	Mundial	Organização	Departamento
2	Poucos	Muitos	Qdes. enormes
3			
4			
5			
6			
1: Escala geográfica 2: # Nós 3: Responsividade		4: Propagação de atualizações 5: # Réplicas 6: Cache no lado do cliente?	

DNS: Comparação

Item	Global	Administração	Gerencial
1	Mundial	Organização	Departamento
2	Poucos	Muitos	Qdes. enormes
3	Segundos	Milissegundos	Imediato
4			
5			
6			
1: Escala geográfica 2: # Nós 3: Responsividade		4: Propagação de atualizações 5: # Réplicas 6: Cache no lado do cliente?	

DNS: Comparação

Item	Global	Administração	Gerencial
1	Mundial	Organização	Departamento
2	Poucos	Muitos	Qdes. enormes
3	Segundos	Milissegundos	Imediato
4	Lento	Imediato	Imediato
5			
6			
1: Escala geográfica 2: # Nós 3: Responsividade		4: Propagação de atualizações 5: # Réplicas 6: Cache no lado do cliente?	

DNS: Comparação

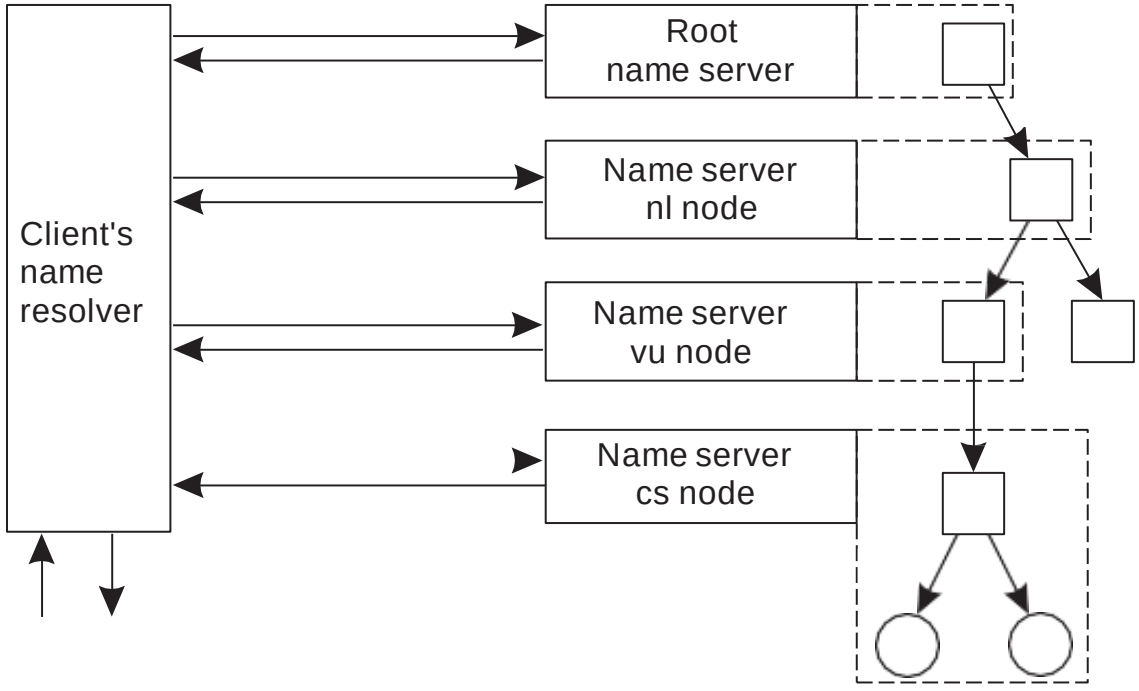
Item	Global	Administração	Gerencial
1	Mundial	Organização	Departamento
2	Poucos	Muitos	Qdes. enormes
3	Segundos	Milissegundos	Imediato
4	Lento	Imediato	Imediato
5	Muitos	Nenhum ou poucos	Nenhum
6			
1: Escala geográfica 2: # Nós 3: Responsividade		4: Propagação de atualizações 5: # Réplicas 6: Cache no lado do cliente?	

DNS: Comparação

Item	Global	Administração	Gerencial
1	Mundial	Organização	Departamento
2	Poucos	Muitos	Qdes. enormes
3	Segundos	Milissegundos	Imediato
4	Lento	Imediato	Imediato
5	Muitos	Nenhum ou poucos	Nenhum
6	Sim	Sim	Às vezes
1: Escala geográfica 2: # Nós 3: Responsividade		4: Propagação de atualizações 5: # Réplicas 6: Cache no lado do cliente?	

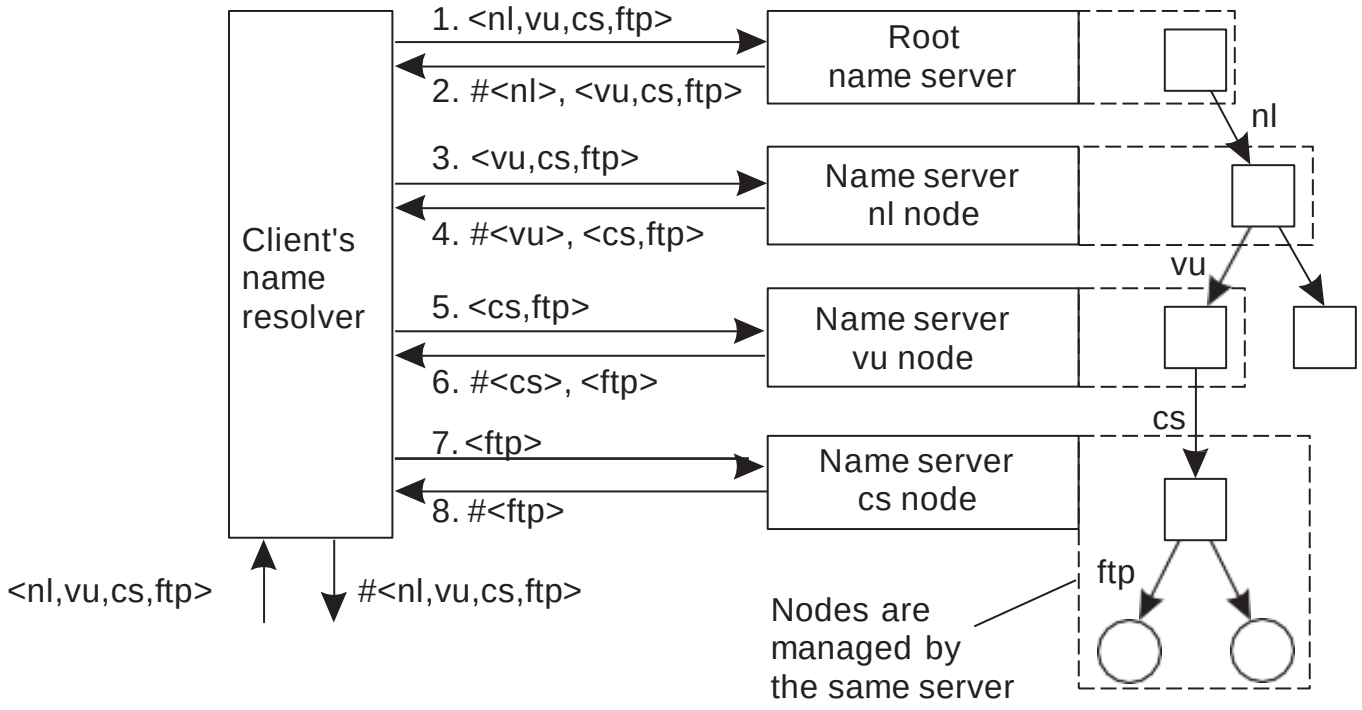
DNS: Resolução de nomes iterativa

1. Cliente envia `resolve(dir,[name1,...,nameK])` para `Server0` responsável por `dir`
2. `Server0` resolve `resolve(dir,name1)` → `dir1`, devolve a identificação (endereço) de `Server1`, que contém `dir1`.
3. Cliente envia `resolve(dir1,[name2,...,nameK])` para `Server1`, etc.



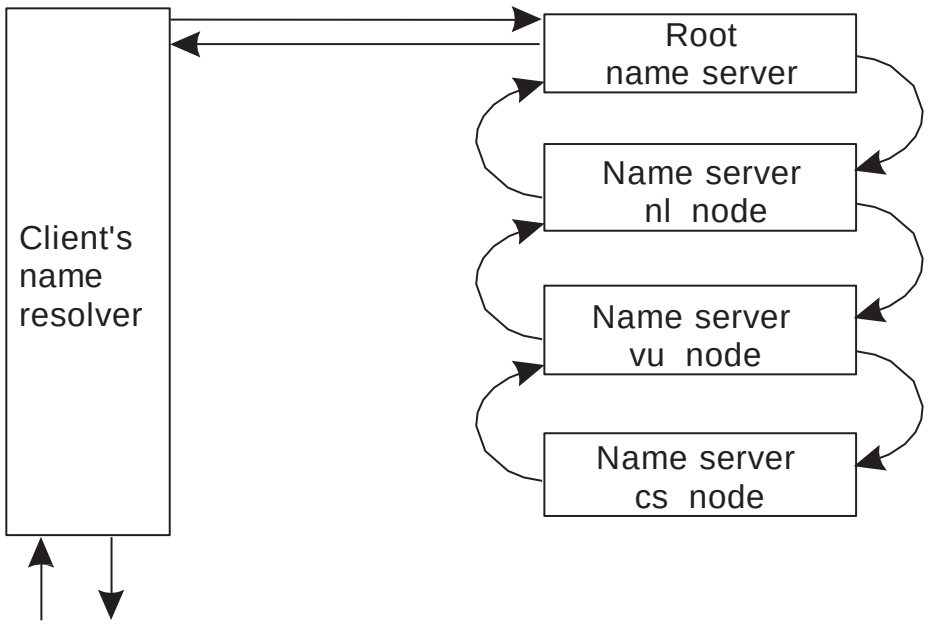
DNS: Resolução de nomes iterativa

1. Cliente envia `resolve(dir,[name1,...,nameK])` para `Server0` responsável por `dir`
2. `Server0` resolve `resolve(dir,name1)` → `dir1`, devolve a identificação (endereço) de `Server1`, que contém `dir1`.
3. Cliente envia `resolve(dir1,[name2,...,nameK])` para `Server1`, etc.



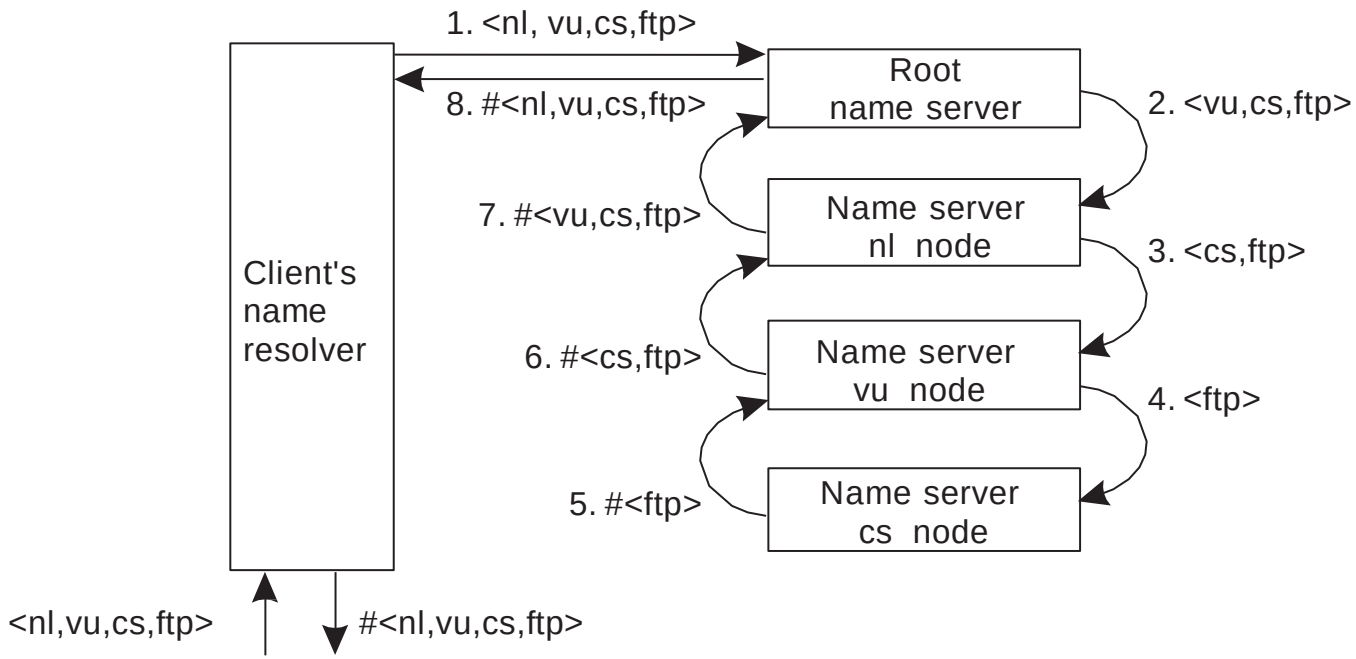
DNS: Resolução de nomes recursiva

1. Cliente envia resolve(dir,[name1,...,nameK]) para Server0, responsável por dir
2. Server0 resolve resolve(dir,name1) → dir1 e envia resolve(dir1,[name2,...,nameK]) para Server1, que contém dir1.
3. Server0 espera pelo resultado de Server1 e o devolve para o cliente.



DNS: Resolução de nomes recursiva

1. Cliente envia `resolve(dir,[name1,...,nameK])` para `Server0`, responsável por `dir`
2. `Server0` resolve `resolve(dir,name1)` → `dir1` e envia `resolve(dir1,[name2,...,nameK])` para `Server1`, que contém `dir1`.
3. `Server0` espera pelo resultado de `Server1` e o devolve para o cliente.



DNS: Resolução de nomes recursiva com cache

Servidor para o nó	Deveria resolver	Procura por	Envia para filho	Recebe e and faz cache	Devolve ao solicitante
cs	<ftp>	#<ftp>	—	—	#<ftp>
vu	<cs,ftp>	#<cs>	<ftp>	#<ftp>	#<cs> #<cs, ftp>
nl	<vu,cs,ftp>	#<vu>	<cs,ftp>	#<cs> #<cs,ftp>	#<vu> #<vu,cs> #<vu,cs,ftp>
root	<nl,vu,cs,ftp>	#<nl>	<vu,cs,ftp>	#<vu> #<vu,cs> #<vu,cs,ftp>	#<nl> #<nl,vu> #<nl,vu,cs> #<nl,vu,cs,ftp>

DNS: Problemas de escalabilidade

Escalabilidade de tamanho

Devemos garantir que os servidores possam lidar com um grande número de requisições por unidade de tempo. Servidores de alto nível podem estar com problemas sérios.

DNS: Problemas de escalabilidade

Escalabilidade de tamanho

Devemos garantir que os servidores possam lidar com um grande número de requisições por unidade de tempo. Servidores de alto nível podem estar com problemas sérios.

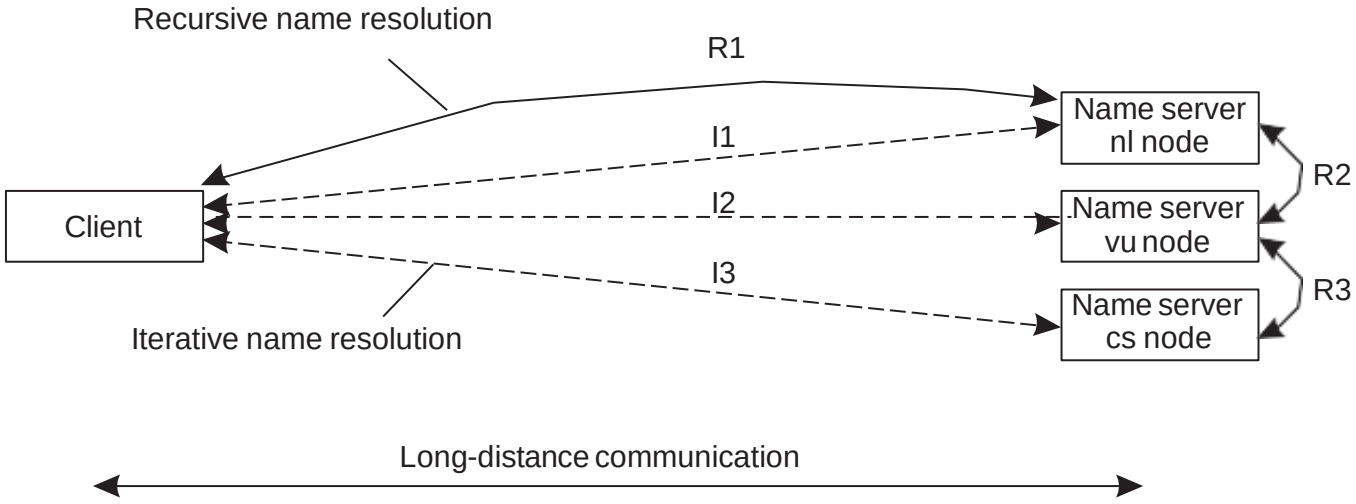
Solução:

Assuma (pelo menos nos níveis global e de administração) que os conteúdos dos nós mudam muito pouco. Assim podemos aplicar replicação extensivamente, mapeando nós a múltiplos servidores e começar a resolução de nomes no servidor mais próximo.

DNS: Problemas de escalabilidade

Escalabilidade geográfica

Precisamos garantir que a resolução de nomes escale mesmo em grandes distâncias geográficas



Porém:

Ao mapear nós em servidores que podem ser localizados em qualquer lugar, nós acabamos introduzindo uma dependência de localização implícita.

DNS: Informação do nó

SOA	Zone	Holds info on the represented zone
A	Host	IP addr. of host this node represents
MX	Domain	Mail server to handle mail for this node
SRV	Domain	Server handling a specific service
NS	Zone	Name server for the represented zone
CNAME	Node	Symbolic link
PTR	Host	Canonical name of a host
HINFO	Host	Info on this host
TXT	Any kind	Any info considered useful

DNS: DIG ufabc.edu.br

```
;; ANSWER SECTION:
ufabc.edu.br.  86400   IN   SOA    ns1.ufabc.edu.br.
                                    root.ufabc.edu.br.
ufabc.edu.br.  86400   IN   NS     ns2.ufabc.edu.br.
ufabc.edu.br.  86400   IN   NS     ns3.ufabc.edu.br.
ufabc.edu.br.  86400   IN   NS     ns1.ufabc.edu.br.
ufabc.edu.br.  86400   IN   A      177.104.50.120
ufabc.edu.br.  86400   IN   MX     smtp.ufabc.edu.br.
ufabc.edu.br.  86400   IN   TXT    v=spf1 mx 177.104.50.110~all

;; ADDITIONAL SECTION:
ns1.ufabc.edu.br  82697   IN   A      177.104.50.101
ns1.ufabc.edu.br. 82697   IN   AAAA   2801:a4:fabc:1034::101
ns2.ufabc.edu.br. 82697   IN   A      177.104.50.102
ns3.ufabc.edu.br. 82697   IN   A      177.104.40.2
smtp.ufabc.edu.br. 86400  IN   A      177.104.50.111
```

Conceitos adquiridos

- Nome e identificador.
- Flooding.
- DHT e fingertables.
- DNS: níveis, resolução iterativa e recursiva.

Agenda

- Definições: nomes, identificadores e endereços
- Resolução de nomes (simples)
- Resolução de nomes (estruturado)
- Resolução de nomes (atributo)

Nomeação baseada em atributos

Observação

Em muitos casos, é mais conveniente nomear e procurar entidades pelos seus atributos $\Rightarrow$ serviços tradicionais de diretórios (ex: páginas amarelas).

Nomeação baseada em atributos

Observação

Em muitos casos, é mais conveniente nomear e procurar entidades pelos seus atributos $\Rightarrow$ serviços tradicionais de diretórios (ex: páginas amarelas).

Problema:

Operações de consulta globais podem ser muito caras, já que necessitam que os valores dos atributos procurados correspondam aos valores reais das entidades. Em princípio, teríamos que inspecionar todas as entidades.

Nomeação baseada em atributos

Observação

Em muitos casos, é mais conveniente nomear e procurar entidades pelos seus atributos $\Rightarrow$ serviços tradicionais de diretórios (ex: páginas amarelas).

Problema:

Operações de consulta globais podem ser muito caras, já que necessitam que os valores dos atributos procurados correspondam aos valores reais das entidades. Em princípio, teríamos que inspecionar todas as entidades.

Solução:

Implementar serviços de diretórios básicos locais (tais como bancos de dados) e combiná-los com os sistemas de nomes estruturados tradicionais.

Implementando um serviço de diretório

Solução para uma busca escalável

Implementar um serviço com diversos bancos de dados locais, combinando-o com o DNS

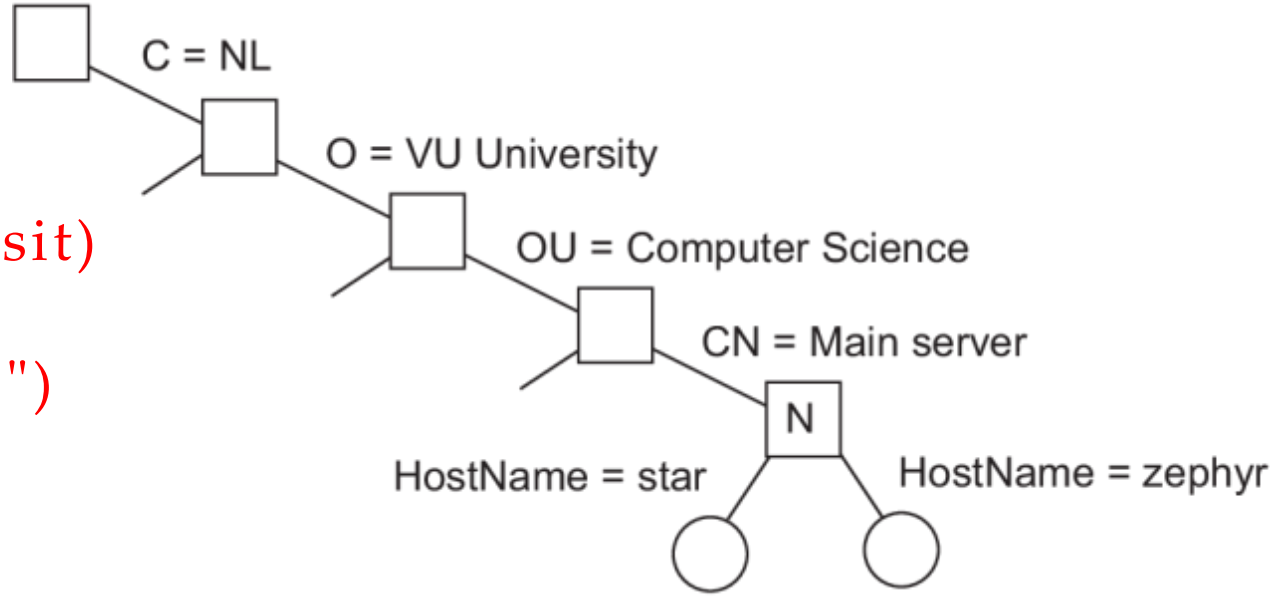
Lightweight Directory Access Protocol (LDAP) [1980]

Cada entrada do diretório consiste em um par (*atributo, valor*), unicamente nomeado para facilitar as buscas.

Attribute	**Abbr.**	**Value**
Country	*C*	NL
Locality	*L*	Amsterdam
Organization	*O*	VU University
OrganizationalUnit	*OU*	Computer Science
CommonName	*CN*	Main server
Mail Servers	–	137.37.20.3, 130.37.24.6, 137.37.20.10
FTP Server	–	130.37.20.20
WWW Server	–	130.37.20.20

Exemplo: LDAP

```
search("\&(C = NL)
        (O = Vrije Universit)
        (OU = *)
        (CN = Main server)")
```



Attribute	Value
Country	NL
Locality	Amsterdam
Organization	Vrije Universit
OrgUnit	Comp. Sc.
CommonName	Main server
Host_Name	star
Host_Address	192.31.231.42

Attribute	Value
Country	NL
Locality	Amsterdam
Organization	Vrije Universit
OrgUnit	Comp. Sc.
CommonName	Main server
Host_Name	zephyr
Host_Address	137.37.20.10