

Chương 2: Nội dung

2.1 Nguyên lý của ứng dụng mạng

- Kiến trúc của ứng dụng
- Các yêu cầu của ứng dụng

2.2 Web và HTTP

2.3 FTP

2.4 Thư điện tử

- SMTP, POP3, IMAP

2.5 DNS

2.6 Ứng dụng P2P

2.7 Lập trình socket với UDP và TCP

Tầng ứng dụng 2-60

DNS: Hệ thống tên miền (domain name system)

Con người: có nhiều định danh:

- CMT, tên, số hộ chiếu

Các host và router trên Internet:

- Địa chỉ IP (32 bit) – được dùng để gán địa chỉ cho các gói tin
- “tên miền”, ví dụ: www.yahoo.com – được con người sử dụng

Hỏi: Làm cách nào để ánh xạ giữa địa chỉ IP và tên miền, và ngược lại?

Hệ thống tên miền:

- ❖ Cơ sở dữ liệu phân tán được cài đặt phân cấp với nhiều server tên miền
- ❖ Giao thức tầng ứng dụng: để các host, các server tên miền truyền thông được thì cần phải *phân giải* các tên miền (diễn dịch địa chỉ/tên miền)
 - Chú ý: chức năng lõi của Internet, được cài đặt như giao thức tầng ứng dụng
 - Phức tạp tại “phần cạnh” của mạng

Tầng ứng dụng 2-61

DNS: các dịch vụ và cấu trúc

Các dịch vụ của DNS

- ❖ Dịch tên host thành địa chỉ IP
- ❖ Bí danh của host
 - Các tên đúng chuẩn, các tên là bí danh
- ❖ Bí danh mail server
- ❖ Phân tán tải
 - Nhìn rộng các máy chủ Web: nhiều địa chỉ IP tương ứng với một tên

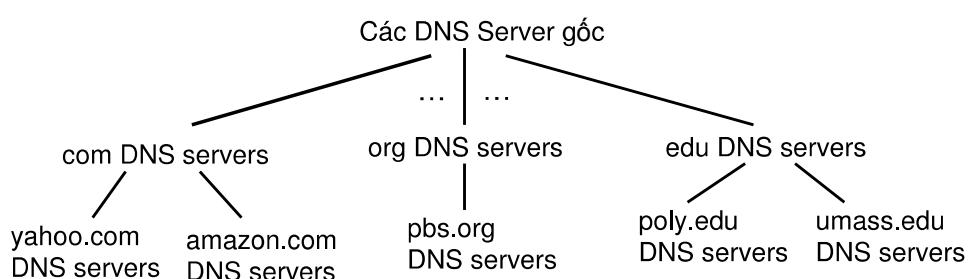
Tại sao không tập trung hóa trong DNS?

- ❖ Một điểm chịu lỗi
- ❖ Lưu lượng
- ❖ Cơ sở dữ liệu tập trung ở xa
- ❖ Bảo trì

Trả lời: Không thể thực hiện với quy mô lớn!

Tầng ứng dụng 2-62

DNS: cơ sở dữ liệu phân tán và phân cấp



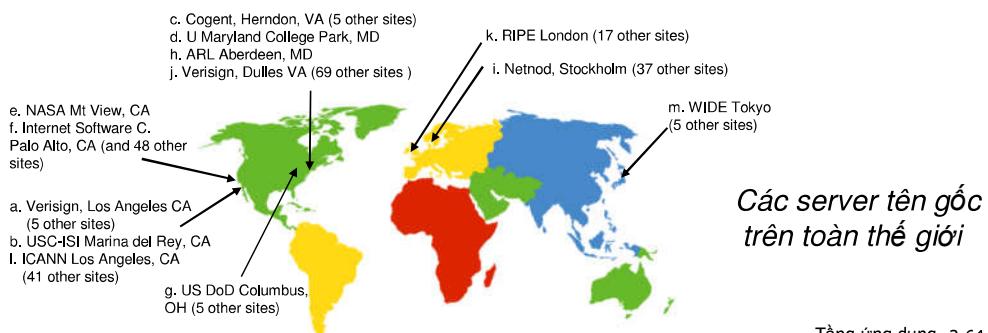
client muốn IP cho www.amazon.com:

- ❖ client truy vấn server gốc để tìm com DNS server
- ❖ client truy vấn .com DNS server để lấy amazon.com DNS server
- ❖ client truy vấn amazon.com DNS server để lấy địa chỉ IP của www.amazon.com

Tầng ứng dụng 2-63

DNS: Các server tên gốc

- ❖ Được tiếp xúc bởi server tên cục bộ mà server này không có khả năng phân giải tên
- ❖ Server tên gốc:
 - Liên lạc với server tên có thẩm quyền nếu việc ánh xạ tên không được xác định
 - Lấy ánh xạ
 - Trả lại ánh xạ cho server tên cục bộ



Tầng ứng dụng 2-64

Các server TLD và server có thẩm quyền

Các server top-level domain (TLD):

- Chịu trách nhiệm cho tên miền com, org, net, edu, aero, jobs, museums, và tất cả các tên miền cấp cao nhất thuộc quốc gia như: uk, fr, ca, jp
- Network Solutions duy trì các server cho .com TLD
- Educause cho .edu TLD

Các server DNS có thẩm quyền:

- Các DNS server của tổ chức, cung cấp các tên host có thẩm quyền cho các ánh xạ IP với các host của tổ chức (ví dụ web và mail).
- Có thể được duy trì bởi các tổ chức hoặc nhà cung cấp dịch vụ

Tầng ứng dụng 2-65

Server tên DNS cục bộ

- ❖ Không hoàn toàn theo cấu trúc phân cấp
- ❖ Mỗi ISP (ISP khu dân cư, công ty, trường học) có một server cục bộ
 - Cũng được gọi là “server tên mặc định”
- ❖ Khi một host tạo một truy vấn DNS, truy vấn này sẽ được gửi tới server DNS cục bộ
 - Có một vùng đệm cục bộ dịch chuyển cặp tên-địa chỉ gần nhất (tuy nhiên vẫn có thể đã bị quá hạn, chưa được cập nhật!)
 - Hoạt động giống như proxy, chuyển tiếp truy vấn vào hệ thống phân cấp

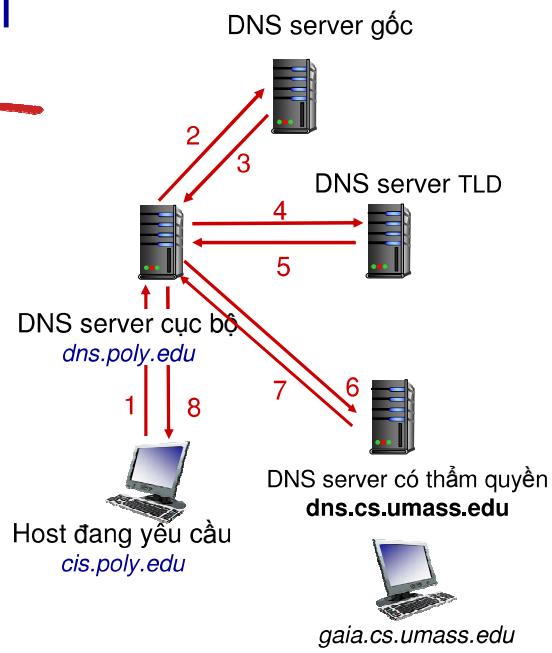
Tầng ứng dụng 2-66

Ví dụ phân giải tên DNS

- ❖ Host tại cis.poly.edu muốn lấy địa chỉ IP của gaia.cs.umass.edu

Truy vấn lặp:

- ❖ Server được hỏi sẽ trả lời với tên của server sẽ có thể hỏi được tiếp
- ❖ “Tôi không biết tên đó, nhưng có thể hỏi server này”

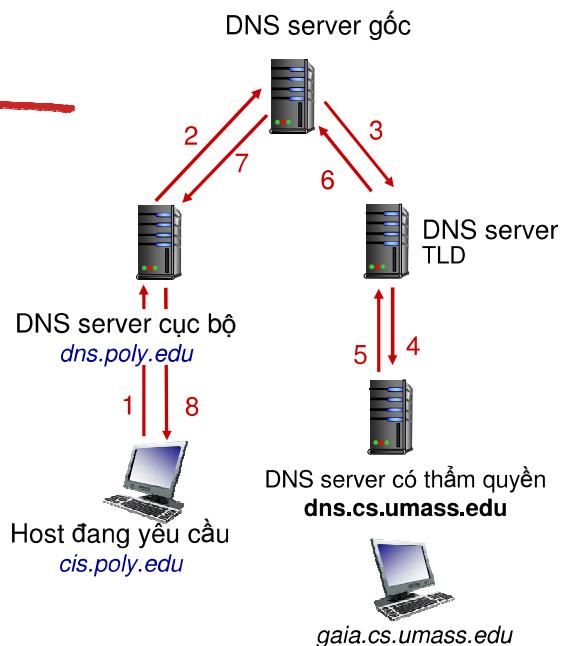


Tầng ứng dụng 2-67

Ví dụ phân giải tên DNS

Truy vấn đệ quy:

- ❖ Đặt trách nhiệm phân giải tên lên server tên được hỏi
- ❖ Tải nặng tại các cấp cao hơn trong hệ thống phân cấp?



Tầng ứng dụng 2-68

DNS: caching và cập nhật các bản ghi

- ❖ Khi server tên học cách ánh xạ, nó sẽ *cache* ánh xạ
 - Mục cache sẽ bị quá hạn (bị xóa) sau một vài lần (TTL)
 - Các server TLD điển hình thường cache trong các server tên cục bộ
 - Do đó, các server tên gốc sẽ không thường xuyên được truy cập
- ❖ Các mục cache có thể bị *quá hạn*
 - Nếu host thay đổi địa chỉ IP, thì Internet sẽ có thể không biết được cho đến khi TTL hết hạn
- ❖ Cơ chế cập nhật, thông báo được đề xuất bởi chuẩn IETF
 - RFC 2136

Tầng ứng dụng 2-69

Bản ghi DNS (DNS records)

DNS: cơ sở dữ liệu phân tán lưu trữ các bản ghi nguồn (resource records - RR)

Định dạng RR: (name, value, type, ttl)

type=A

- **name** là tên máy
- **Value** là địa chỉ IP

type=NS

- **name** là tên miền (ví dụ: foo.com)
- **value** là tên host của server có thẩm quyền cho tên miền này

type=CNAME

- **name** là tên bí danh của tên “chuẩn” (tên thật)
- **www.ibm.com** tên thật là **servereast.backup2.ibm.com**
- **value** là tên chuẩn

type=MX

- **value** là tên của mail server được kết hợp với **name**

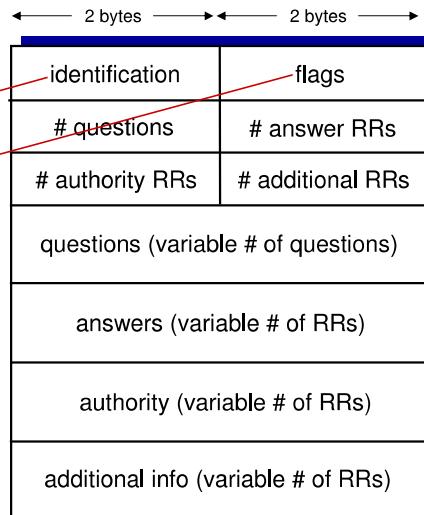
Tầng ứng dụng 2-70

Giao thức, thông điệp DNS

- ❖ Các thông điệp **truy vấn** và **trả lời** đều có cùng **định dạng thông điệp**

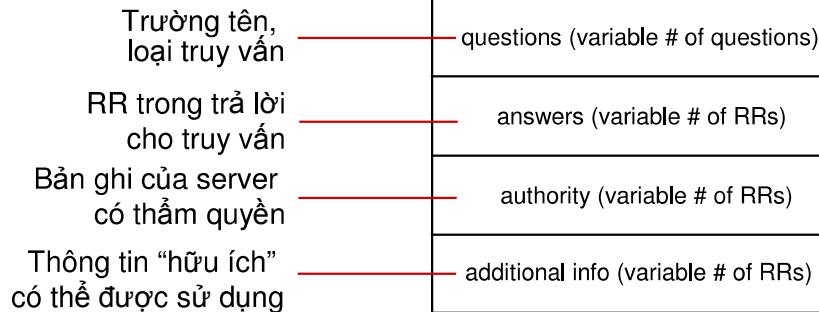
Tiêu đề thông điệp

- ❖ **identification:** 16 bit cho truy vấn/trả lời
- ❖ **flags:**
 - Truy vấn hoặc trả lời
 - Đệ quy chờ
 - Đệ quy sẵn sàng
 - Trả lời có thẩm quyền



Tầng ứng dụng 2-71

Giao thức, thông điệp DNS



Tầng ứng dụng 2-72

Chèn thêm bản ghi vào trong DNS

- ❖ Ví dụ: Tạo mới “Network Utopia”
- ❖ Đăng ký tên miền networkuptopia.com tại **DNS registrar** (Ví dụ: Network Solutions)
 - Cung cấp tên, địa chỉ IP của server tên có thẩm quyền (sơ cấp và thứ cấp) của bạn
 - Registrar sẽ chèn hai bản ghi RR vào trong .com TLD server:
(**networkutopia.com**, **dns1.networkutopia.com**, **NS**)
(**dns1.networkutopia.com**, **212.212.212.1**, **A**)
- ❖ Tạo bản ghi loại A vào trong server có thẩm quyền cho **www.networkutopia.com**; bản ghi loại MX record cho **networkutopia.com**

Tầng ứng dụng 2-73

Tấn công DNS

Tấn công DDoS

- ❖ Tấn công server gốc với lưu lượng lớn
 - Không thành công cho đến nay
 - Lọc lưu lượng
 - DNS server cục bộ cache IP của TLD server, cho phép bỏ qua server gốc
 - Tấn công TLD servers
 - Tiềm tàng nhiều nguy hiểm hơn

Chuyển hướng tấn công

- ❖ Man-in-middle
 - Truy vấn đánh chặn
- ❖ Đầu độc DNS
 - Gửi trả lời giả mạo đến DNS server, mà các server này có thể cache lại

Khai thác DNS cho DDoS

- ❖ Gửi truy vấn với địa chỉ nguồn giả mạo: IP đích
- ❖ Yêu cầu khuếch đại

Tầng ứng dụng 2-74

Chương 2: Nội dung

2.1 Nguyên lý của ứng dụng mạng

- Kiến trúc của ứng dụng
- Các yêu cầu của ứng dụng

2.2 Web và HTTP

2.3 FTP

2.4 Thư điện tử

- SMTP, POP3, IMAP

2.5 DNS

2.6 Ứng dụng P2P

2.7 Lập trình socket với UDP và TCP

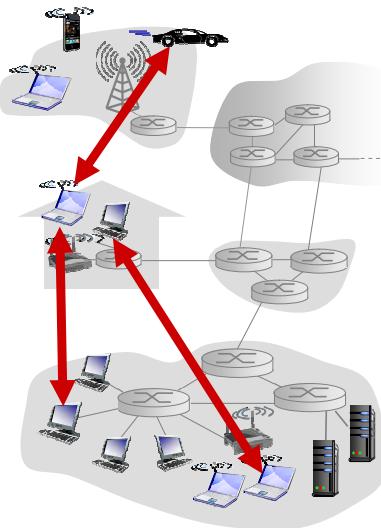
Tầng ứng dụng 2-75

Kiến trúc mạng P2P thuận túy

- ❖ Không cần phải có server luôn hoạt động
- ❖ Các hệ thống cuối tùy ý kết nối trực tiếp
- ❖ Các peer (các nút mạng) không cần kết nối liên tục vào hệ thống mạng và có thể thay đổi địa chỉ IP.

Ví dụ:

- Chia sẻ file (BitTorrent)
- Streaming (KanKan)
- VoIP (Skype)

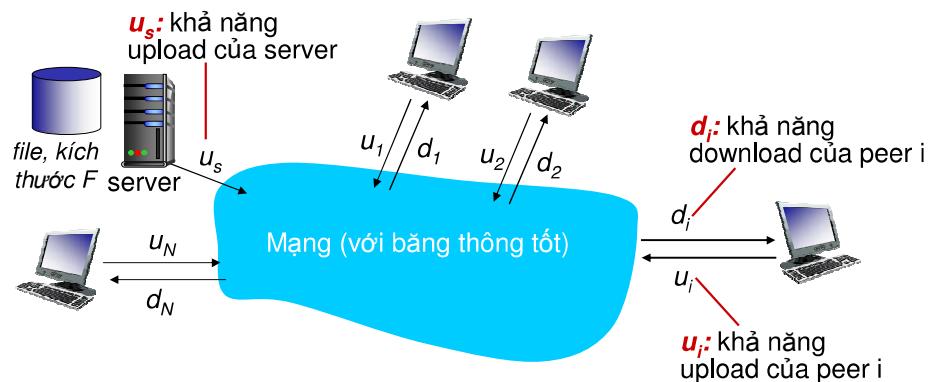


Tầng ứng dụng 2-76

Chia sẻ file: client-server và P2P

Hỏi: Cần mất bao lâu để truyền (phân phối) file (có kích thước F) từ một server đến N peer?

- Khả năng upload/download (tải lên/tải về) của peer bị giới hạn bởi tài nguyên.



Tầng ứng dụng 2-77

Thời gian truyền file: client-server

- ❖ **Việc truyền của server:** phải gửi tuần tự (upload) N bản sao của file:

- Thời gian gửi một bản sao: F/u_s
- Thời gian gửi N bản sao: NF/u_s

- ❖ **Client:** Mỗi client phải download bản sao của file

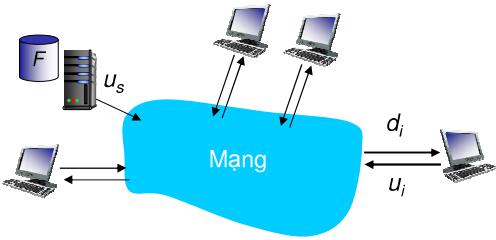
- d_{min} = tốc độ download nhỏ nhất của client
- Thời gian download (ứng với d_{min}) của client: F/d_{min}

Thời gian để phân phối F

$$D_{C-S} \geq \max\{NF/u_s, F/d_{min}\}$$

Tăng tuyến tính theo N

Tầng ứng dụng 2-78



Thời gian truyền file: P2P

- ❖ **Việc truyền của server:** phải upload ít nhất một bản sao của file:

- Thời gian để gửi một bản: F/u_s

- ❖ **Client:** mỗi client phải download bản sao của file

- Thời gian download (ứng với d_{min}) của client : F/d_{min}

- ❖ **Các client:** phải download NF bits

- Tốc độ upload cao nhất (giới hạn) là $u_s + \sum u_i$



Thời gian để phân phối F

$$D_{P2P} \geq \max\{F/u_s, F/d_{min}, NF/(u_s + \sum u_i)\}$$

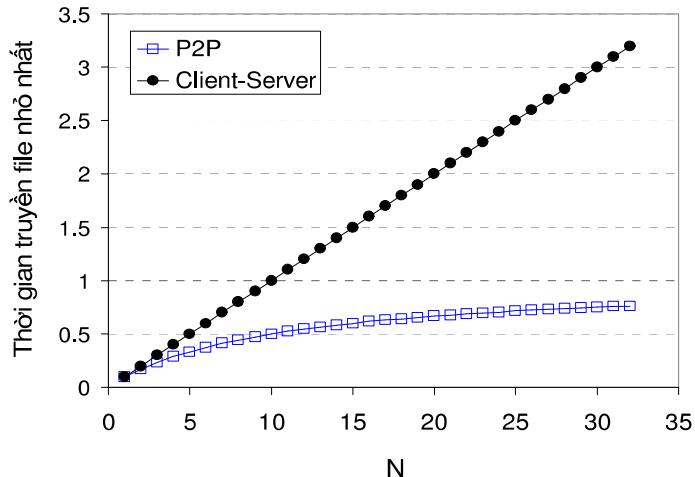
Tăng tuyến tính theo N ...

... nhưng để thực hiện việc này, mỗi peer phải có khả năng phục vụ

Tầng ứng dụng 2-79

Ví dụ so sánh client-server với P2P

Tốc độ upload của client = u , $F/u = 1$ giờ, $u_s = 10u$, $d_{min} \geq u_s$



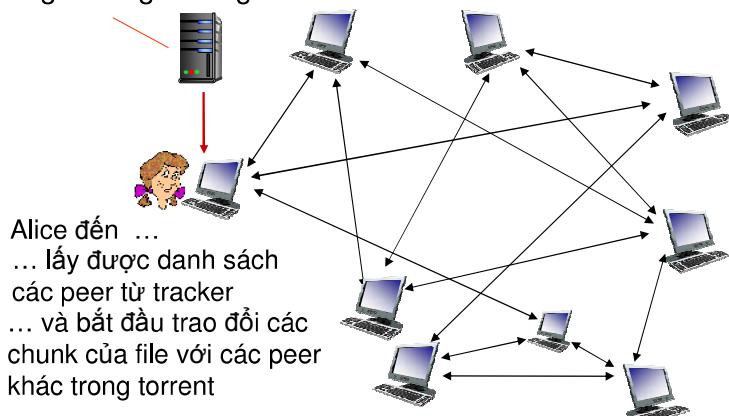
Tầng ứng dụng 2-80

Truyền file P2P: BitTorrent

- File được chia thành các chunk (phần) có kích thước là 256Kb
- Các peer trong torrent gửi/nhận các chunk của file

tracker: kiểm tra các peer đang tham gia trong torrent

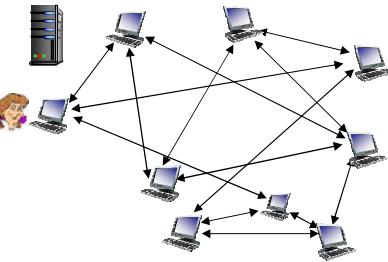
torrent: nhóm các peer trao đổi các chunk của một file



Tầng ứng dụng 2-81

Truyền file P2P: BitTorrent

- ❖ peer tham gia vào torrent:
 - Không có các chunk, nhưng sẽ tích lũy chúng qua thời gian từ các peer khác
 - Đăng ký với tracker để nhận được danh sách các peer, kết nối với tập con của các peer ("các hàng xóm")
- ❖ Trong khi download, peer sẽ upload các chunk tới các peer khác
- ❖ peer có thể thay đổi các peer mà nó sẽ trao đổi các chunk
- ❖ **churn:** các peer có thể đến hoặc đi
- ❖ Khi peer có được toàn bộ file, nó có thể (ích kỷ) rời đi hoặc (vị tha) ở lại trong torrent



Tầng ứng dụng 2-82

BitTorrent: yêu cầu lấy, gửi các chunk của file

Yêu cầu lấy chunk:

- ❖ Tại bất kỳ thời điểm nào, các peer khác nhau đều có các tập con các chunk khác nhau của file.
- ❖ Theo định kỳ, Alice sẽ hỏi mỗi peer về danh sách các chunk mà họ có
- ❖ Alice sẽ yêu cầu các chunk còn thiếu từ các peer, phân hiếm nhất trước

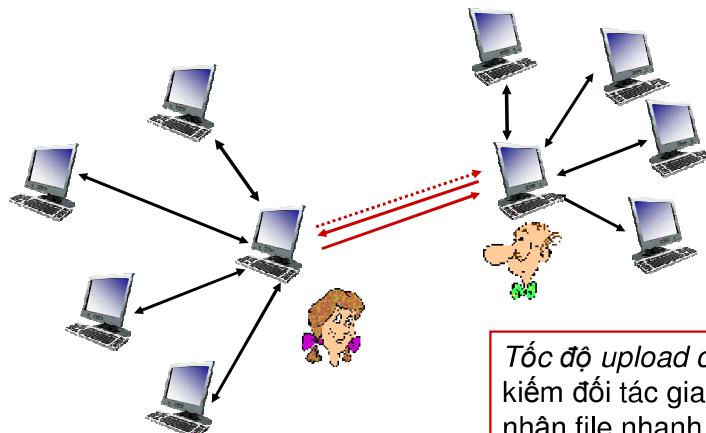
Gửi chunk: tit-for-tat

- ❖ Alice gửi các chunk tới 4 peer hiện đang gửi chunk của cô ấy với **tốc độ cao nhất**
 - Các peer khác đang bị chặn bởi Alice (không nhận được chunk từ cô ấy)
 - Đánh giá lại top 4 sau mỗi 10 giây
- ❖ Cứ mỗi 30 giây: chọn các peer khác một cách ngẫu nhiên, và bắt đầu gửi chunk
 - "Tôi ưu hóa không chặn" peer này
 - Peer được chọn mới sẽ được gia nhập vào top 4

Tầng ứng dụng 2-83

BitTorrent: tit-for-tat

- (1) Alice “tối ưu hóa không chặn” Bob
- (2) Alice trở thành một nhà cung cấp trong top 4 của Bob;
- (3) Ngược lại, Bob trở thành một nhà cung cấp trong top 4 của Alice



Tầng ứng dụng 2-84

Bảng băm phân tán - Distributed Hash Table (DHT)

- ❖ DHT: là một *cơ sở dữ liệu P2P phân tán*
- ❖ Cơ sở dữ liệu chứa các cặp (**khóa, giá trị**) (**key, value**);
Ví dụ:
 - Khóa: Số thứ tự; Giá trị: tên người
 - Khóa: tiêu đề bộ phim; Giá trị: địa chỉ IP
- ❖ Phân tán các cặp (**key, value**) qua (hàng triệu) các peer
- ❖ Mỗi peer **truy vấn** DHT theo khóa
 - DHT trả lại các giá trị tương ứng với khóa
- ❖ Các peer cũng có thể **chèn thêm (insert)** các cặp (**khóa, giá trị**)

Tầng ứng dụng 2-85

Hỏi: Làm cách nào để gán khóa cho các peer?

- ❖ Vấn đề chính:
 - Gán các cặp (khóa, giá trị) cho các peer.
- ❖ Ý tưởng cơ bản:
 - Chuyển mỗi khóa thành một số kiểu số nguyên (integer)
 - Gán số nguyên cho từng peer
 - Đặt cặp (khóa, giá trị) trong một peer mà nó là **gần nhất** với khóa

Tầng ứng dụng 2-86

Định danh DHT

- ❖ Gán định danh số nguyên cho mỗi peer trong dãy $[0, 2^n - 1]$ cho một số n .
 - Mỗi định danh được biểu diễn bởi n bit.
- ❖ Yêu cầu mỗi khóa là một số nguyên trong cùng dãy
- ❖ Để nhận được khóa số nguyên, cần băm khóa gốc
 - **Ví dụ:** khóa = **hash**("Led Zeppelin IV")
 - Đây là lý do tại sao nó được gọi là một **bảng "băm" phân tán**

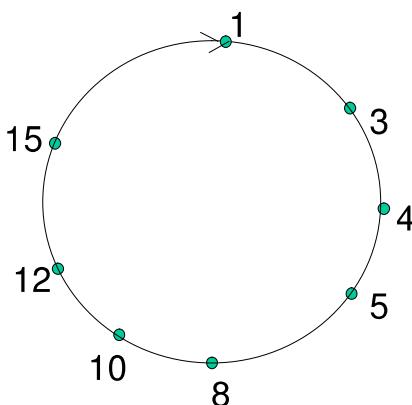
Tầng ứng dụng 2-87

Gán khóa cho các peer

- ❖ Quy tắc: gán khóa cho peer mà có ID **gần nhất**.
- ❖ Trong bài giảng: gần nhất là **giá trị kế liền ngay** với khóa.
- ❖ Ví dụ: $n=4$; các peer: 1,3,4,5,8,10,12,14;
 - khóa = 13, thì peer kế liền với nó là 14
 - khóa = 15, thì peer kế liền là 1

Tầng ứng dụng 2-88

DHT vòng (1)



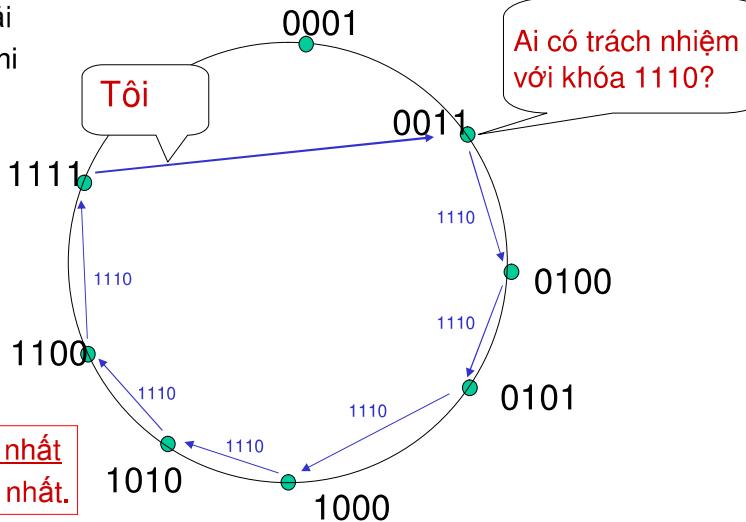
- ❖ Mỗi peer *chỉ biết* về peer kế tiếp và peer tiền nhiệm ngay trước nó
- ❖ Mạng che phủ (“overlay network”)

Tầng ứng dụng 2-89

DHT vòng (1)

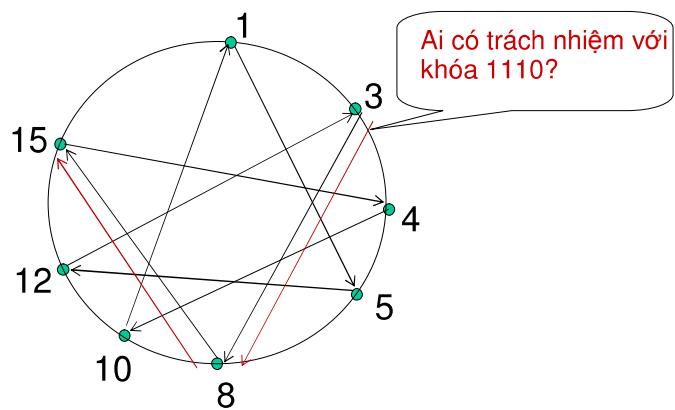
Trung bình cần $O(N)$

thông điệp để giải quyết truy vấn, khi có N peer



Tầng ứng dụng 2-90

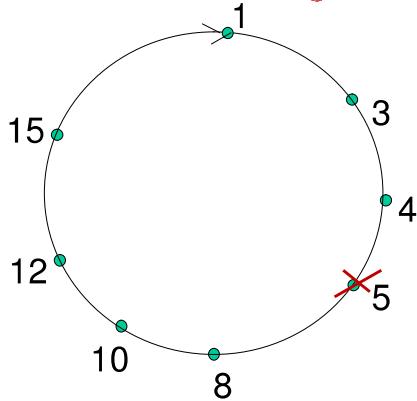
DHT vòng với peer tắt (shortcut)



- ❖ Mỗi peer giữ theo dõi địa chỉ IP của peer kế tiếp, tiền nhiệm và các peer tắt.
- ❖ Sẽ giảm từ 6 xuống còn 2 thông điệp.
- ❖ Có thể thiết kế các peer tắt để có $O(\log N)$ hàng xóm, thì sẽ có $O(\log N)$ thông điệp trong truy vấn

Tầng ứng dụng 2-91

Peer churn



Ví dụ: peer 5 đột ngột rời đi

- ❖ peer 4 phát hiện ra peer 5 rời đi, nó sẽ làm việc với peer 8 kế liền ngay với nó; hỏi peer 8 xem peer nào sẽ làm peer kế liền ngay của nó được; sau đó sẽ làm việc tiếp với peer kế liền ngay với peer 8 này (kế liền ngay thứ cấp).
- ❖ peer 13 muốn kết nối với ai?

Xử lý peer churn:

- ❖ Các peer có thể đến hoặc đi (churn)
- ❖ Mỗi peer biết được địa chỉ của 2 peer kế của nó
- ❖ Định kỳ mỗi peer sẽ ping đến 2 peer kế của nó để kiểm tra sự tồn tại
- ❖ Nếu peer kế liền ngay nó rời đi, thì nó sẽ chọn peer kế tiếp để làm peer kế liền ngay của nó