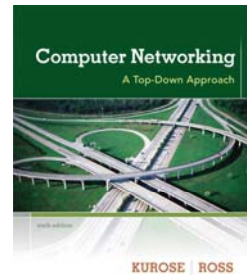


## Chương 2 Tầng Ứng dụng (Application layer)



**Computer  
Networking: A  
Top Down  
Approach**  
6<sup>th</sup> edition  
Jim Kurose, Keith Ross  
Addison-Wesley  
March 2012

A note on the use of these ppt slides:

We're making these slides freely available to all (faculty, students, readers). They're in PowerPoint form so you see the animations; and can add, modify, and delete slides (including this one) and slide content to suit your needs. They obviously represent a lot of work on our part. In return for use, we only ask the following:

- ❖ If you use these slides (e.g., in a class) that you mention their source (after all, we'd like people to use our book!)
- ❖ If you post any slides on a www site, that you note that they are adapted from (or perhaps identical to) our slides, and note our copyright of this material.

Thanks and enjoy! JFK/KWR

©All material copyright 1996-2012  
J.F Kurose and K.W. Ross, All Rights Reserved

Tầng Application 2-1

cuu duong than cong . com

## Chương 2: Nội dung

2.1 Các nguyên lý của  
các ứng dụng mạng

2.2 Web và HTTP

2.3 FTP

2.4 Thư điện tử

▪ SMTP, POP3, IMAP

2.5 DNS

2.6 Các ứng dụng P2P

2.7 Lập trình socket  
với UDP và TCP

cuu duong than cong . com

Tầng Application 2-2

## Chương 2: tầng Ứng dụng

### Mục tiêu:

- ❖ Khái niệm, các phương diện áp dụng của các giao thức ứng dụng mạng
  - Các mô hình dịch vụ tầng transport
  - Mô hình client-server
  - Mô hình peer-to-peer
- ❖ Tìm hiểu về các giao thức thông qua việc xem xét các giao thức phổ biến của tầng ứng dụng
  - HTTP
  - FTP
  - SMTP / POP3 / IMAP
  - DNS
- ❖ Lập trình ứng dụng mạng
  - socket API

Tầng Application 2-3

cuu duong than cong . com

## Một số ứng dụng mạng

- ❖ e-mail
- ❖ web
- ❖ Nhắn tin
- ❖ Đăng nhập từ xa
- ❖ Chia sẻ file P2P
- ❖ Trò chơi trực tuyến với nhiều người cùng tham gia
- ❖ Truyền hình trực tuyến (streaming stored video - Vd: YouTube, Hulu, Netflix)
- ❖ Đàm thoại trên mạng IP (Vd: Skype)
- ❖ Hội thảo video thời gian thực
- ❖ Mạng xã hội
- ❖ Tìm kiếm
- ❖ ...
- ❖ ...

Tầng Application 2-4

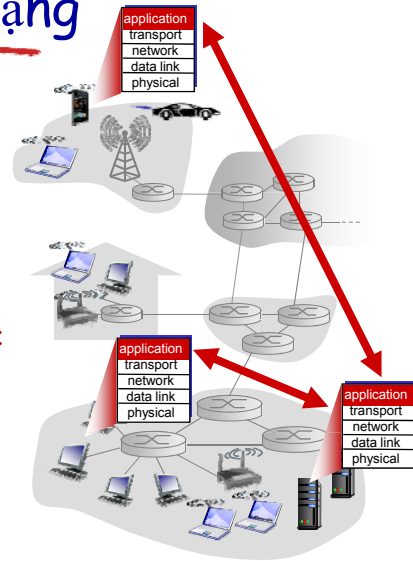
## Tạo một ứng dụng mạng

### Viết chương trình để:

- ❖ Chạy trên các hệ thống đầu cuối (khác nhau)
- ❖ Liên lạc qua mạng
- ❖ Ví dụ: phần mềm web server giao tiếp với trình duyệt

### Không cần viết phần mềm cho các thiết bị trong lõi của mạng

- ❖ Các thiết bị trong lõi mạng không chạy các ứng dụng của người dùng
- ❖ Các ứng dụng trên các hệ thống đầu cuối cho phép phát triển ứng dụng và quảng bá nhanh chóng



Tầng Application 2-5

cuu duong than cong . com

## Các kiến trúc ứng dụng

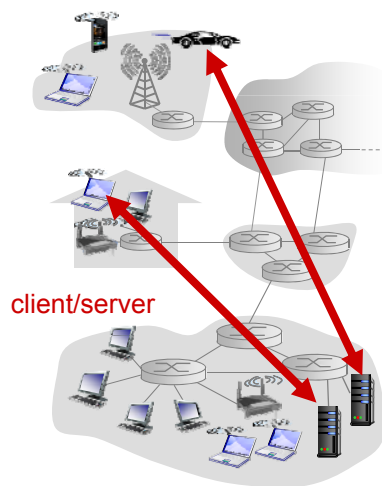
### Kiến trúc phù hợp của các ứng dụng:

- ❖ client-server
- ❖ peer-to-peer (P2P) (Mạng ngang hàng)

cuu duong than cong . com

Tầng Application 2-6

## Kiến trúc Client-server



### server:

- ❖ Máy luôn luôn hoạt động
- ❖ Địa chỉ IP cố định
- ❖ Tổ chức thành các trung tâm dữ liệu để mở rộng quy mô

### clients:

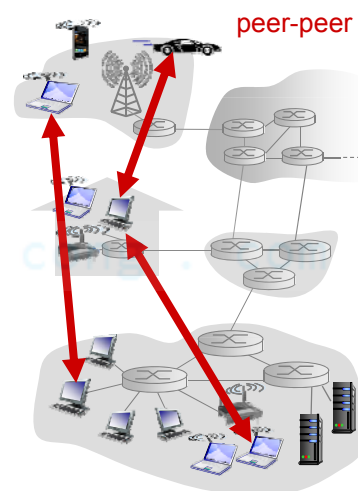
- ❖ Giao tiếp với server
- ❖ Có thể kết nối không liên tục
- ❖ Có thể thay đổi địa chỉ IP
- ❖ Không giao tiếp trực tiếp với các client khác

Tầng Application 2-7

cuu duong than cong . com

## Kiến trúc P2P (ngang hàng)

- ❖ *không có server* luôn luôn hoạt động
- ❖ Các hệ thống đầu cuối bất kỳ truyền thông trực tiếp với nhau
- ❖ Các peer yêu cầu dịch vụ từ các peer khác và cung cấp dịch vụ ngược lại cho các peer khác
  - *Có khả năng tự mở rộng - các peer mới cung cấp thêm dịch vụ mới, cũng như có thêm nhu cầu mới về dịch vụ*
- ❖ Các peer được kết nối không liên tục và có thể thay đổi địa chỉ IP
  - Quản lý phức tạp



Tầng Application 2-8

## Các tiến trình liên lạc

### *Tiến trình (process):*

chương trình đang chạy trong một máy

- ❖ Trong cùng một máy, hai tiến trình giao tiếp với nhau bằng cách sử dụng cơ chế truyền thông liên tiến trình (**inter-process communication**) (được định nghĩa bởi hệ điều hành)
- ❖ Các tiến trình trong các host khác nhau truyền thông với nhau bằng cách trao đổi **các thông điệp (message)**

clients, servers

**Tiến trình client:** tiến trình khởi tạo liên lạc

**Tiến trình server:** tiến trình chờ đợi để được liên lạc

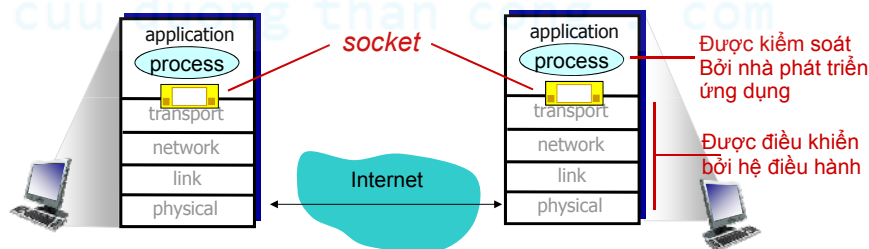
- ❖ Chú ý: các ứng dụng với kiến trúc P2P có cả các tiến trình client và server

Tầng Application 2-9

cuu duong than cong . com

## Sockets

- ❖ Tiến trình gửi/nhận thông điệp đến/ từ **socket** của nó
- ❖ socket tương tự như cửa ra vào
  - Tiến trình gửi đẩy thông điệp ra khỏi cửa
  - Tiến trình gửi dựa trên hạ tầng vận chuyển bên kia của cánh cửa để phân phối thông điệp đến socket tại tiến trình nhận



Tầng Application 2-10

## Xác định tiến trình

- ❖ Để nhận thông điệp, tiến trình phải có **định danh**
- ❖ Thiết bị host device có địa chỉ IP 32-bit duy nhất
- ❖ **Q:** địa chỉ IP của host mà trên tiến trình đang chạy trên đó có đủ để xác định tiến trình đó hay không?
  - **A:** không, có nhiều tiến trình có thể đang được chạy trên cùng một host
- ❖ **Định danh (identifier)** bao gồm cả địa chỉ IP và **số cổng (port numbers)** được liên kết với tiến trình trên host.
- ❖ Ví dụ về số cổng:
  - HTTP server: 80
  - Mail server: 25
- ❖ Để gửi thông điệp HTTP đến web server `gaia.cs.umass.edu` :
  - **IP address:** 128.119.245.12
  - **port number:** 80
- ❖ Còn nữa...

Tầng Application 2-11

cuu duong than cong . com

## Giao thức tầng Ứng dụng định nghĩa

- ❖ **Các loại thông điệp được trao đổi**
    - e.g., yêu cầu (request), đáp ứng (response)
  - ❖ **Cú pháp thông điệp:**
    - Các trường trong thông điệp và cách mà các trường được định nghĩa
  - ❖ **Ngữ nghĩa của thông điệp**
    - Ý nghĩa của thông tin trong các trường
  - ❖ **Các quy tắc (rules)** khi nào và cách mà các tiến trình gửi và đáp ứng các thông điệp
- Các giao thức mở:**
    - ❖ Được định nghĩa trong RFCs
    - ❖ Cung cấp khả năng tương tác cho các ứng dụng thuộc các nhà phát triển khác nhau
    - ❖ Vd: HTTP, SMTP
  - Các giao thức độc quyền:**
    - ❖ Vd: Skype

Tầng Application 2-12

## Dịch vụ vận chuyển nào mà ứng dụng cần?

### Toàn vẹn dữ liệu (data integrity)

- ❖ Một số ứng dụng (ví dụ truyền file, web transactions) yêu cầu độ tin cậy 100% khi truyền dữ liệu
- ❖ Các ứng dụng khác (ví dụ audio) có thể chịu được một số mất mát.

### Định thời (timing)

- ❖ Một số ứng dụng (ví dụ, thoại Internet, game tương tác) yêu cầu độ trễ thấp để đạt được "hiệu quả" thực thi

### Thông lượng (throughput)

- ❖ Một số ứng dụng (vd: đa phương tiện) yêu cầu thông lượng tối thiểu để đạt được "hiệu quả" thực thi
- ❖ Các ứng dụng khác ("ứng dụng mềm dẻo") có thể dùng bất kỳ thông lượng nào cũng được

### An ninh

- ❖ Mã hóa, toàn vẹn dữ liệu, ...

Tăng Application 2-13

cuu duong than cong . com

## Các yêu cầu dịch vụ vận chuyển: các ứng dụng phổ biến

ứng dụng	mất dữ liệu	thông lượng	độ nhạy thời gian
Truyền file	không	mềm dẻo	không
e-mail	không	mềm dẻo	không
Web documents	không	mềm dẻo	không
audio/video thời gian thực	chịu lỗi	audio: 5kbps-1Mbps video: 10kbps-5Mbps	có, 100' s msec
audio/video đã lưu	chịu lỗi	như trên	có, vài giây
Game tương tác	chịu lỗi	Trên một vài kbps	có, 100' s msec
nhắn tin	không	mềm dẻo	có và không

Tăng Application 2-14

## Các dịch vụ thuộc giao thức vận chuyển trên Internet

### Dịch vụ TCP:

- ❖ **Truyền tải có đảm bảo (reliable transport)** giữa tiến trình gửi và nhận
- ❖ **Điều khiển luồng thông tin (flow control)**: bên gửi sẽ không gửi vượt khả năng bên nhận
- ❖ **Điều khiển tắc nghẽn (congestion control)**: điều tiết bên gửi khi mạng quá tải
- ❖ **Không hỗ trợ**: định thời, bảo đảm thông lượng tối thiểu, bảo mật
- ❖ **Hướng kết nối (connection-oriented)**: yêu cầu thiết lập kết nối giữa tiến trình client và server trước khi truyền

### Dịch vụ UDP:

- ❖ **Truyền dữ liệu không đảm bảo (unreliable data transfer)** giữa tiến trình gửi và nhận
- ❖ **Không hỗ trợ**: độ tin cậy, điều khiển luồng, điều khiển tắc nghẽn, định thời, bảo đảm thông lượng, bảo mật, và thiết lập kết nối.

**Q:** Tại sao phải quan tâm? Tại sao có UDP?

Tầng Application 2-15

cuu duong than cong . com

## Ứng dụng Internet: Các giao thức tầng application, transport

application	Giao thức tầng application	Giao thức dưới tầng transport
e-mail	SMTP [RFC 2821]	TCP
remote terminal access	Telnet [RFC 854]	TCP
Web	HTTP [RFC 2616]	TCP
Truyền file	FTP [RFC 959]	TCP
streaming multimedia	HTTP (e.g., YouTube), RTP [RFC 1889]	TCP or UDP
Thoại Internet	SIP, RTP, độc quyền (e.g., Skype)	TCP or UDP

Tầng Application 2-16



## Bảo mật TCP

### TCP & UDP

- ❖ Không mã hóa
- ❖ Mật mã chưa mã hóa được gửi đến socket để đi qua Internet trong dạng nguyên bản

### SSL

- ❖ Hỗ trợ kết nối TCP được mã hóa
- ❖ Toàn vẹn dữ liệu
- ❖ Chứng thực đầu cuối

### SSL là giao thức ở tầng Application

- ❖ Các ứng dụng dùng thư viện SSL, thông qua đó để "nói chuyện" với TCP

### SSL socket API

- ❖ Mật mã dạng không mã hóa được gửi vào trong socket và chuyển đi qua Internet ở dạng mã hóa
- ❖ Xem chương 7

Tầng Application 2-17

cuu duong than cong . com

## Chương 2: Nội dung

2.1 Các nguyên lý của các ứng dụng mạng

2.2 Web và HTTP

2.3 FTP

2.4 Thư điện tử

▪ SMTP, POP3, IMAP

2.5 DNS

2.6 Các ứng dụng P2P

2.7 Lập trình socket với UDP và TCP

cuu duong than cong . com

Tầng Application 2-18

# Web và HTTP

*Ôn lại...*

- ❖ **web page** bao gồm các đối tượng (**objects**)
- ❖ Đối tượng có thể là tập tin HTML, hình ảnh JPEG, Java applet, tập tin audio,...
- ❖ web page bao gồm tập tin HTML bao gồm một số đối tượng được tham chiếu
- ❖ Mỗi đối tượng có thể được xác định bởi một **URL**, ví dụ

`www.someschool.edu/someDept/pic.gif`

tên máy

đường dẫn

Tầng Application 2-19

cuu duong than cong . com

## Tổng quan HTTP

**HTTP: hypertext transfer protocol**

- Giao thức tầng Application của Web
- ❖ Mô hình client/server
  - **client**: trình duyệt gửi yêu cầu, nhận phản hồi (dùng giao thức HTTP) và "hiển thị" các đối tượng Web
  - **server**: Web server gửi (dùng giao thức HTTP) các đối tượng để trả lời yêu cầu



Tầng Application 2-20

## Tổng quan HTTP (tt)

### *dùng TCP:*

- ❖ Client khởi tạo kết nối TCP (tạo socket) đến cổng 80 server
- ❖ server chấp nhận kết nối TCP từ client
- ❖ Các thông điệp HTTP (thông điệp thuộc giao thức tầng application) được trao đổi giữa trình duyệt (HTTP client) và web server (HTTP server)
- ❖ Kết nối TCP được đóng

### *HTTP "không lưu trạng thái"*

- ❖ server không duy trì thông tin về các yêu cầu trước đó của client

*ngoài lề*  
Các giao thức nào duy trì "trạng thái" thì phức tạp!

- ❖ Lịch sử trước đó (trạng thái) phải được duy trì
- ❖ Nếu server/client bị sự cố, cách nhìn về "trạng thái" của nó có thể bị mâu thuẫn, phải được điều chỉnh

Tăng Application 2-21

cuu duong than cong . com

## Các kết nối HTTP

### *HTTP không bền vững*

- ❖ Chỉ tối đa một đối tượng được gửi qua kết nối TCP
  - Kết nối sau đó sẽ bị đóng
- ❖ Tải nhiều đối tượng yêu cầu nhiều kết nối

### *HTTP bền vững*

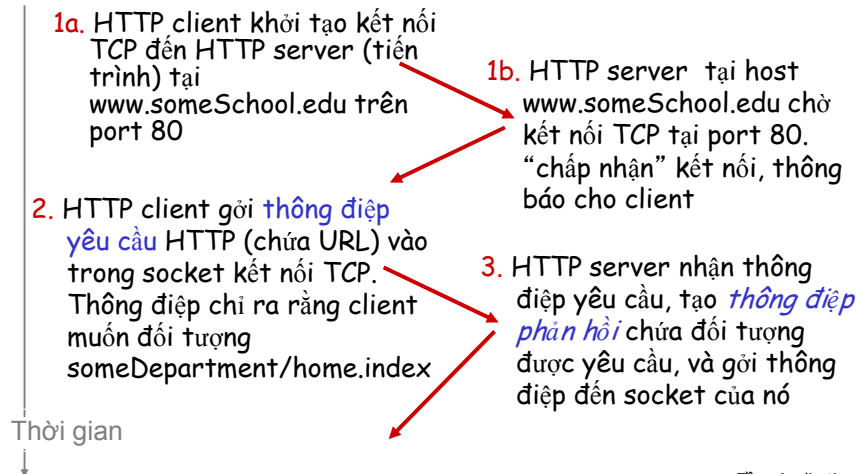
- ❖ Nhiều đối tượng có thể được gửi qua một kết nối TCP giữa client và server

Tăng Application 2-22

## HTTP không bền vững

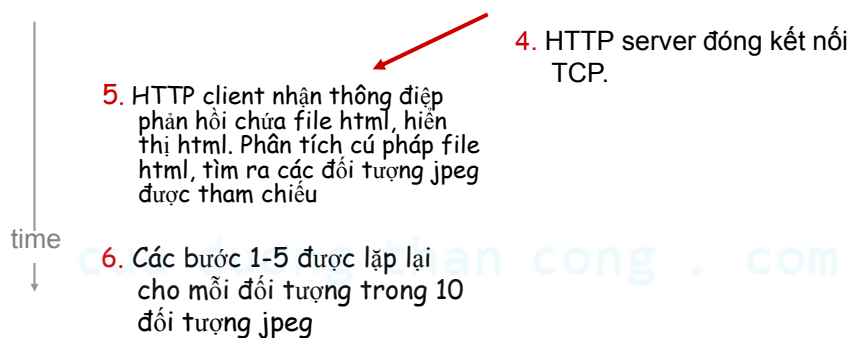
Giả sử người dùng vào URL như sau:

`www.someSchool.edu/someDepartment/home.index` (chứa text, tham chiếu đến 10 hình jpeg)



Tăng Application 2-23

## HTTP không bền vững(tt)



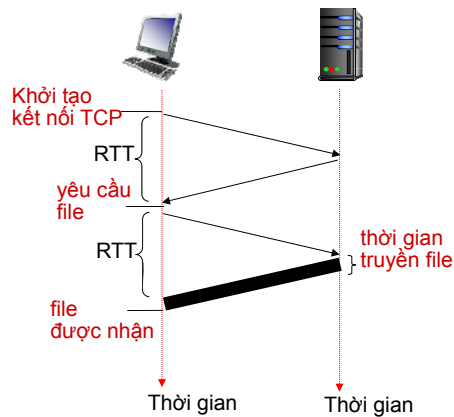
Tăng Application 2-24

## HTTP không bền vững: thời gian đáp ứng

**RTT (round-trip time):** thời gian để cho một gói tin nhô đi từ client đến server và quay ngược lại

**Thời gian đáp ứng HTTP:**

- ❖ Một RTT để khởi tạo kết nối TCP
- ❖ Một RTT cho yêu cầu HTTP và vài byte đầu tiên của phản hồi HTTP được trả về
- ❖ Thời gian truyền file
- ❖ Thời gian đáp ứng HTTP không bền vững =  $2RTT + \text{thời gian truyền file}$



Tăng Application 2-25

cuu duong than cong . com

## HTTP bền vững

**Vấn đề với HTTP không bền vững:**

- ❖ Yêu cầu requires 2 RTTs cho mỗi đối tượng
- ❖ Tốn tài nguyên khi Hệ điều hành xử lý mỗi kết nối TCP
- ❖ Các trình duyệt thường mở các kết nối TCP song song để lấy các đối tượng được tham chiếu

**HTTP bền vững:**

- ❖ Server để kết nối mở sau khi gửi phản hồi
- ❖ Các thông điệp HTTP tiếp theo giữa cùng client/server được gửi trên kết nối đã mở ở trên
- ❖ Client gửi các yêu cầu ngay khi nó gặp một đối tượng tham chiếu
- ❖ Chỉ cần một RTT cho tất cả các đối tượng được tham chiếu

Tăng Application 2-26

## Thông điệp yêu cầu HTTP

- ❖ hai loại thông điệp HTTP: yêu cầu (*request*), phản hồi (*response*)
- ❖ **Thông điệp yêu cầu HTTP:**
  - ASCII (dạng thức con người có thể đọc được)

Dòng yêu cầu  
(các lệnh GET, POST, HEAD)

Các dòng  
header

ký tự xuống dòng,  
về đầu dòng mới chỉ  
điểm cuối cùng  
của thông điệp

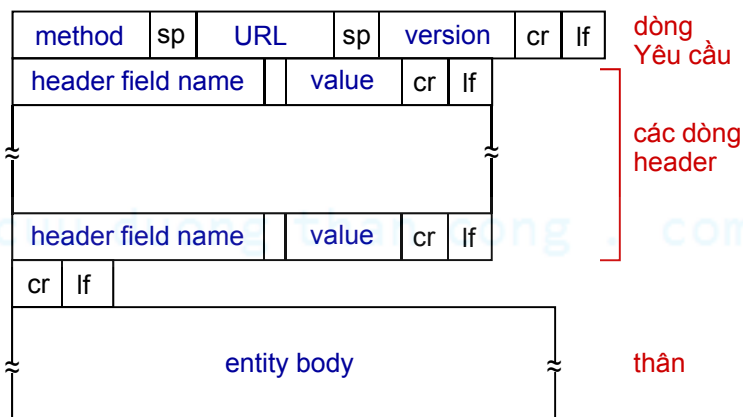
```
GET /index.html HTTP/1.1\r\n
Host: www-net.cs.umass.edu\r\n
User-Agent: Firefox/3.6.10\r\n
Accept: text/html,application/xhtml+xml\r\n
Accept-Language: en-us,en;q=0.5\r\n
Accept-Encoding: gzip,deflate\r\n
Accept-Charset: ISO-8859-1,utf-8;q=0.7\r\n
Keep-Alive: 115\r\n
Connection: keep-alive\r\n
\r\n
```

Ký tự xuống dòng  
Ký tự về đầu dòng

Tăng Application 2-27

cuu duong than cong . com

## Thông điệp yêu cầu HTTP: định dạng tổng quát



Tăng Application 2-28

## Tải lên biểu mẫu nhập liệu

### Phương thức POST:

- ❖ web page thường bao gồm form input
- ❖ dữ liệu nhập được tải lên server trong phần thân đối tượng HTML

### Phương thức URL:

- ❖ Dùng phương thức GET
- ❖ dữ liệu nhập được tải lên trong trường URL của dòng yêu cầu:

`www.somesite.com/animalsearch?monkeys&banana`

Tầng Application 2-29

cuu duong than cong . com

## Các phương thức

### HTTP/1.0:

- ❖ GET
- ❖ POST
- ❖ HEAD
  - Yêu cầu server loại bỏ đối tượng được yêu cầu ra khỏi thông điệp phản hồi

### HTTP/1.1:

- ❖ GET, POST, HEAD
- ❖ PUT
  - Tải file trong thân thực thể đến đường dẫn được xác định trong trường URL
- ❖ DELETE
  - Xóa file được chỉ định trong trường URL

Tầng Application 2-30

## Thông điệp phản hồi HTTP

dòng trạng thái  
(giao thức  
mã trạng thái  
cụm từ trạng thái)

các dòng  
header

Dữ liệu, ví dụ,  
file HTML  
được yêu cầu

```
HTTP/1.1 200 OK\r\n
Date: Sun, 26 Sep 2010 20:09:20 GMT\r\n
Server: Apache/2.0.52 (CentOS)\r\n
Last-Modified: Tue, 30 Oct 2007 17:00:02
GMT\r\n
ETag: "17dc6-a5c-bf716880"\r\n
Accept-Ranges: bytes\r\n
Content-Length: 2652\r\n
Keep-Alive: timeout=10, max=100\r\n
Connection: Keep-Alive\r\n
Content-Type: text/html; charset=ISO-8859-
1\r\n
\r\n
data data data data data ...
```

Tăng Application 2-31

cuu duong than cong . com

## Các mã trạng thái phản hồi HTTP

- ❖ Mã trạng thái xuất hiện trong dòng đầu tiên trong thông điệp đáp ứng từ server tới client
- ❖ Một số mã mẫu:
  - 200 OK**
    - Yêu cầu thành công, đối tượng được yêu cầu sau ở trong thông điệp này
  - 301 Moved Permanently**
    - Đối tượng được yêu cầu đã được di chuyển, vị trí mới được xác định sau trong thông điệp này (Location:)
  - 400 Bad Request**
    - Server không hiểu thông điệp yêu cầu
  - 404 Not Found**
    - Thông tin được yêu cầu không tìm thấy trên server này
  - 505 HTTP Version Not Supported**

Tăng Application 2-32



## Thử kiểm tra HTTP (phía client)

1. Telnet đến Web server yêu thích của bạn:

```
telnet cis.poly.edu 80
```

Mở kết nối TCP ở port 80  
(port server HTTP mặc định) tại cis.poly.edu.  
Mọi thứ nhập vào được gửi đến  
port 80 tại cis.poly.edu

2. Nhập vào yêu cầu trong lệnh GET HTTP:

```
GET /~ross/ HTTP/1.1  
Host: cis.poly.edu
```

Bằng cách gõ những dòng này  
(enter 2 lần), bạn đã gửi yêu cầu  
GET tối thiểu (nhưng đầy đủ)  
đến HTTP server

3. Xem thông điệp phản hồi được gửi bởi HTTP server!

(hoặc dùng Wireshark để xem thông điệp  
yêu cầu và phản hồi của HTTP được bắt lại)

Tăng Application 2-33

cuu duong than cong . com

## Trạng thái User-server: cookies

Nhiều Web site dùng cookies

Ví dụ:

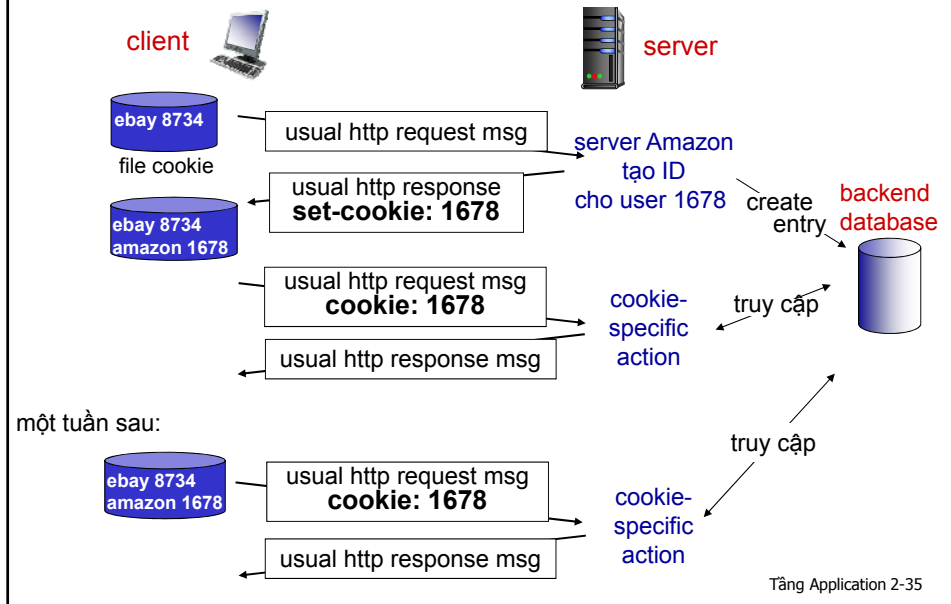
**4 thành phần:**

- 1) cookie header line của thông điệp phản hồi HTTP
- 2) cookie header line trong thông điệp yêu cầu HTTP kế tiếp
- 3) File cookie được lưu trữ trên máy người dùng, được quản lý bởi trình duyệt của người dùng
- 4) Cơ sở dữ liệu tại Web site

- ❖ Susan thường truy cập Internet từ một PC
- ❖ Vào trang thương mại điện tử lần đầu tiên
- ❖ Khi yêu cầu khởi tạo HTTP đến trang web đó, thì trang đó tạo:
  - ID duy nhất
  - Một bản ghi trong cơ sở dữ liệu cho ID đó

Tăng Application 2-34

## Cookies: lưu trữ “trạng thái” (tt.)



cuu duong than cong . com

## Cookies (tt)

*Cookie có thể được sử dụng cho:*

- ❖ Cấp phép
- ❖ Giỏ mua hàng
- ❖ Các khuyến cáo
- ❖ Trạng thái phiên làm việc của user (Web e-mail)

*Làm thế nào để giữ “trạng thái”:*

- ❖ các thời điểm kết thúc giao thức: duy trì trạng thái tại người gửi/nhận thông qua nhiều giao dịch
- ❖ cookies: các thông điệp http mang trạng thái

*cookies và sự riêng tư:*

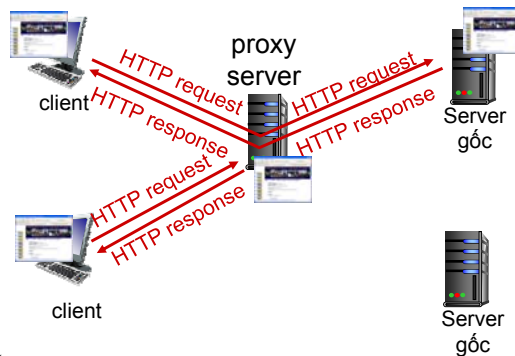
- ❖ cookie cho phép các trang biết nhiều hơn về bạn
- ❖ Bạn có thể cung cấp tên và email cho các trang

Tầng Application 2-36

## Web caches (proxy server)

**Mục tiêu:** thỏa mãn yêu cầu của client không cần liên quan đến server nguồn

- ❖ user thiết lập trình duyệt: truy cập Web thông qua cache
- ❖ Trình duyệt gửi tất cả yêu cầu HTTP đến cache
  - Đối tượng có trong cache: cache trả về đối tượng
  - Ngược lại cache yêu cầu đối tượng từ server gốc, sau đó trả đối tượng đó cho client



Tăng Application 2-37

cuu duong than cong . com

## Thông tin thêm về Web caching

- ❖ Cache hoạt động như là cả client và server
  - server đối với client yêu cầu thông tin
  - client đối với server cung cấp
- ❖ Thông thường cache được cài đặt bởi ISP (trường đại học, công ty, ISP riêng)

### **Tại sao dùng Web caching?**

- ❖ Giảm thời gian đáp ứng cho yêu cầu của client
- ❖ Giảm lưu lượng trên đường link truy cập ra Internet của một tổ chức
- ❖ Internet có rất nhiều caches: cho phép những nhà cung cấp nội dung với lượng tài nguyên "nghèo nàn" vẫn cung cấp nội dung một cách hiệu quả (chia sẻ file P2P cũng vậy)

Tăng Application 2-38

## Ví dụ Caching:

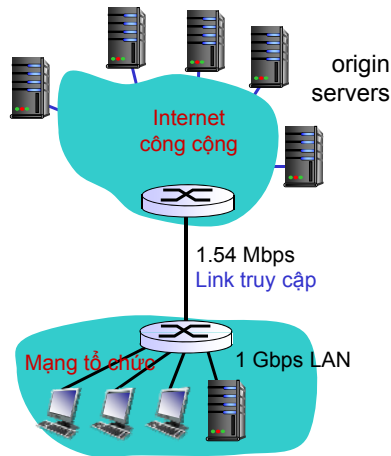
### Giả sử:

- ❖ Kích thước trung bình của đối tượng: 100K bits
- ❖ Số lượng yêu cầu trung bình từ trình duyệt đến server gốc: 15/sec
- ❖ Tốc độ truyền dữ liệu trung bình đến trình duyệt: 1.50 Mbps
- ❖ RTT từ bộ định tuyến của tổ chức đến bất kỳ server gốc: 2 giây
- ❖ Tốc độ link truy cập: 1.54 Mbps

### Kết quả:

- ❖ Độ khả dụng của LAN: 15%
- ❖ Độ khả dụng của link truy cập = 99%
- ❖ Tổng thời gian trễ = trễ Internet + trễ truy cập + trễ LAN  
= 2 giây + minutes +  $\mu$ secs

Vấn đề!



Tăng Application 2-39

cuu duong than cong . com

## Ví dụ Caching: đường link truy cập lớn hơn

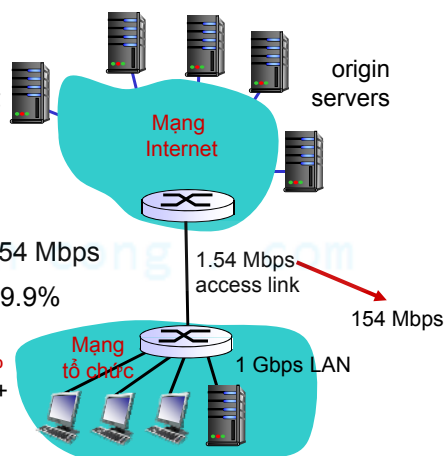
### Giả sử:

- ❖ Kích thước trung bình của đối tượng: 100K bits
- ❖ tốc độ trung bình yêu cầu từ trình duyệt đến server = 15/s
- ❖ Tốc độ trung bình dữ liệu đến trình duyệt: 1.50 Mbps
- ❖ RTT từ bộ định tuyến của tổ chức đến bất kỳ server gốc: 2 giây
- ❖ Tốc độ link truy cập: 1.54 Mbps

### Kết quả:

- ❖ độ khả dụng của LAN = 15%
- ❖ độ khả dụng trên liên kết truy cập = 99%
- ❖ Tổng độ trễ = trễ Internet + trễ truy cập + trễ LAN  
= 2 sec + minutes +  $\mu$ secs

**Chi phí:** tốc độ link truy cập được tăng lên (không rẻ!)



Tăng Application 2-40

## Ví dụ Caching: thiết lập cache

### Giả sử:

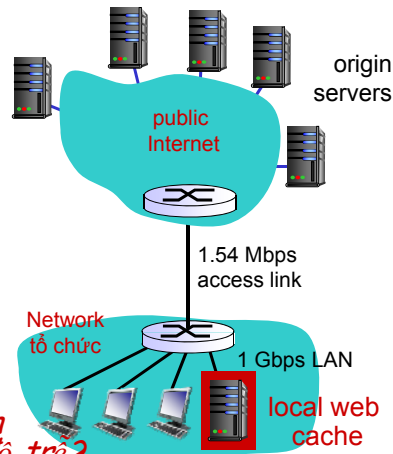
- ❖ Kích thước trung bình của đối tượng: 100K bits
- ❖ tốc độ trung bình yêu cầu từ trình duyệt đến server = 15/s
- ❖ Tốc độ trung bình dữ liệu đến trình duyệt: 1.50 Mbps
- ❖ RTT từ bộ định tuyến của tổ chức đến bất kỳ server gốc: 2 giây
- ❖ Tốc độ link truy cập: 1.54 Mbps

### Kết quả:

- ❖ Độ khả dụng LAN: 15%
- ❖ access link utilization = ?
- ❖ total delay = ?

*Làm cách nào để tính độ khả dụng đường link, độ trễ?*

**Chi phí:** web cache (rẻ!)



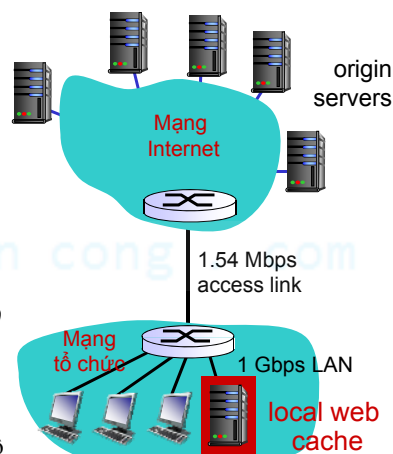
Tăng Application 2-41

cuu duong than cong . com

## Ví dụ Caching: thiết lập cache

### Tính độ khả dụng của đường link truy cập, độ trễ với cache:

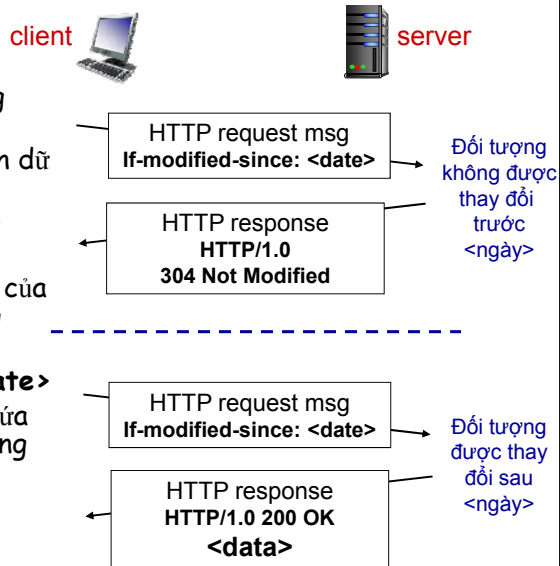
- ❖ Giả sử khả năng đáp ứng của cache là 0.4
  - 40% yêu cầu được cache đáp ứng, 60% yêu cầu được server gốc đáp ứng
- ❖ Độ hiệu dụng của đường kết nối ra ngoài (access link):
  - 60% yêu cầu dùng access link
- ❖ Tốc độ truyền dữ liệu đến trình duyệt trên access link =  $0.6 \times 1.50 \text{ Mbps} = 0.9 \text{ Mbps}$ 
  - Độ khả dụng =  $0.9 / 1.54 = 0.58$
- ❖ Tổng độ trễ
  - =  $0.6 \times (\text{độ trễ từ server gốc}) + 0.4 \times (\text{độ trễ khi được cache đáp ứng})$
  - =  $0.6 \times (2.01) + 0.4 \times (\sim \text{msecs})$
  - =  $\sim 1.2 \text{ secs}$
  - Ít hơn với link 154 Mbps (và cũng rẻ hơn!)



Tăng Application 2-42

## GET có điều kiện

- ❖ **Mục tiêu:** không gửi đối tượng nếu đối tượng trong cache đã được cập nhật
  - Không có độ trễ truyền dữ liệu
  - Mức độ sử dụng đường link thấp hơn
- ❖ **cache:** xác định thời gian của bản sao được cache trong thông điệp yêu cầu HTTP  
**If-modified-since: <date>**
- ❖ **server:** đáp ứng không chứa đối tượng nếu bản sao trong cache đã được cập nhật:  
**HTTP/1.0 304 Not Modified**



Tăng Application 2-43

cuu duong than cong . com

## Chương 2: Nội dung

2.1 Các nguyên lý của các ứng dụng mạng

2.2 Web và HTTP

2.3 FTP

2.4 Thư điện tử

- SMTP, POP3, IMAP

2.5 DNS

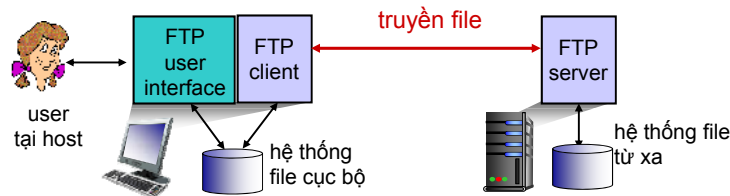
2.6 Các ứng dụng P2P

2.7 Lập trình socket với UDP và TCP

cuu duong than cong . com

Tăng Application 2-44

## FTP: giao thức truyền file



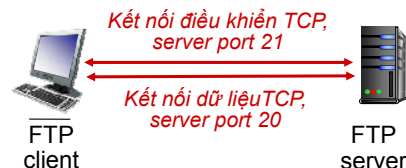
- ❖ Truyền file đến/từ máy ở xa
- ❖ Mô hình client/server
  - *client*: phía khởi tạo phiên truyền (đến/từ máy ở xa)
  - *server*: máy ở xa
- ❖ FTP: RFC 959
- ❖ FTP server: port 21

Tầng Application 2-45

cuu duong than cong . com

## FTP: kết nối điều khiển và kết nối dữ liệu riêng biệt

- ❖ FTP client liên hệ với FTP server tại port 21, dùng TCP
- ❖ client được cấp phép trên kết nối điều khiển
- ❖ client duyệt thư mục từ xa, bằng cách gửi các lệnh trên kết nối điều khiển
- ❖ Khi server nhận lệnh truyền file, *server* mở kết nối dữ liệu TCP thứ 2 (để truyền file) đến client
- ❖ Sau khi truyền một file, server đóng kết nối dữ liệu
- ❖ server mở kết nối dữ liệu TCP khác để truyền file khác
- ❖ Kết nối điều khiển: “*out of band*” (*ngoại tuyến*)
- ❖ FTP server duy trì “trạng thái”: thư mục hiện tại, xác thực trước đó



Tầng Application 2-46

## Các lệnh và phản hồi FTP

### *Các lệnh mẫu:*

- ❖ Gởi văn bản ASCII trên kênh điều khiển
- ❖ **USER *username***
- ❖ **PASS *password***
- ❖ **LIST** trả về danh sách file trên thư mục hiện tại
- ❖ **RETR *filename*** lấy file
- ❖ **STOR *filename*** lưu trữ (đặt) file vào trong máy ở xa

### *Ví dụ mã trả về*

- ❖ Mã trạng thái và cụm từ mô tả (như HTTP)
- ❖ **331 Username OK, password required**
- ❖ **125 data connection already open; transfer starting**
- ❖ **425 Can't open data connection**
- ❖ **452 Error writing file**

Tăng Application 2-47

cuu duong than cong . com

## Chương 2: Nội dung

2.1 Các nguyên lý của các ứng dụng mạng

2.2 Web và HTTP

2.3 FTP

2.4 Thư điện tử

▪ SMTP, POP3, IMAP

2.5 DNS

2.6 Các ứng dụng P2P

2.7 Lập trình socket với UDP và TCP

Tăng Application 2-48



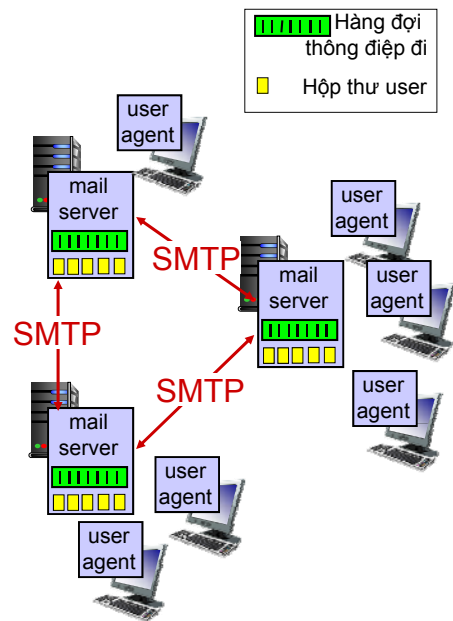
## Thư điện tử

### *Ba thành phần chính:*

- ❖ user agents
- ❖ mail servers
- ❖ simple mail transfer protocol: SMTP

### *User Agent*

- ❖ Còn gọi là “mail reader”
- ❖ Soạn thảo, sửa đổi, đọc các thông điệp email
- ❖ Ví dụ Outlook, Thunderbird, iPhone mail client
- ❖ Các thông điệp đi và đến được lưu trên server



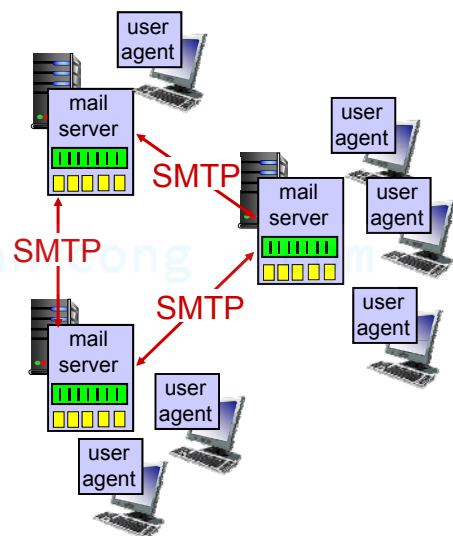
Tăng Application 2-49

cuu duong than cong . com

## Thư điện tử: mail servers

### *mail servers:*

- ❖ *Hộp thư (mailbox)* chứa thông điệp đến user
- ❖ *Hàng thông điệp (message queue)* của các thông điệp mail ra ngoài (chuẩn bị gửi)
- ❖ *Giao thức SMTP* giữa các mail server để gửi các thông điệp email
  - client: mail server gửi
  - “server”: mail server nhận



Tăng Application 2-50

## Thư điện tử: SMTP [RFC 2821]

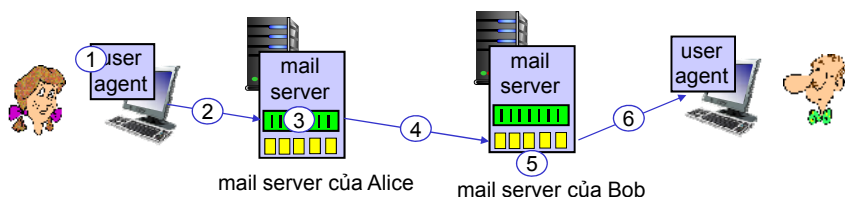
- ❖ Sử dụng TCP để truyền thông điệp email một cách tin cậy từ client đến cổng 25 server
- ❖ Truyền trực tiếp: server gửi đến server nhận
- ❖ 3 giai đoạn truyền
  - bắt tay (chào hỏi)
  - truyền thông điệp
  - đóng
- ❖ Tương tác lệnh/phản hồi (như HTTP, FTP)
  - **Lệnh:** văn bản ASCII
  - **Phản hồi:** mã và cụm trạng thái
- ❖ Thông điệp phải ở dạng mã ASCII 7 bit

Tầng Application 2-51

cuu duong than cong . com

## Tình huống: Alice gửi thông điệp đến Bob

- 1) Alice dùng một UA để soạn thảo thông điệp "gửi đến" bob@some school.edu
- 2) UA của Alice gửi thông điệp đến mail server của cô ta; thông điệp được đặt trong hàng đợi
- 3) Phần client của SMTP server mở kết nối TCP với mail server của Bob
- 4) Phần client của SMTP server gửi thông điệp của Alice trên kết nối TCP
- 5) Mail server của Bob đặt thông điệp đó trong hộp thư của Bob
- 6) Bob kích hoạt user agent của anh ta để đọc thông điệp



Tầng Application 2-52

## Ví dụ tương tác SMTP

```
S: 220 hamburger.edu
C: HELO crepes.fr
S: 250 Hello crepes.fr, pleased to meet you
C: MAIL FROM: <alice@crepes.fr>
S: 250 alice@crepes.fr... Sender ok
C: RCPT TO: <bob@hamburger.edu>
S: 250 bob@hamburger.edu ... Recipient ok
C: DATA
S: 354 Enter mail, end with "." on a line by itself
C: Do you like ketchup?
C: How about pickles?
C: .
S: 250 Message accepted for delivery
C: QUIT
S: 221 hamburger.edu closing connection
```

Tăng Application 2-53

cuu duong than cong . com

## Thử nghiệm tương tác SMNP:

- ❖ **telnet servername 25**
- ❖ Xem trả lời 220 từ server
- ❖ Nhập các lệnh HELO, MAIL FROM, RCPT TO, DATA, QUIT

Lệnh ở trên cho phép bạn gửi email không cần dùng email client (reader)

Tăng Application 2-54

## SMTP: kết luận

- ❖ SMTP dùng kết nối bền vững
- ❖ SMTP yêu cầu thông điệp (header & body) phải ở dạng ASCII 7-bit
- ❖ SMTP server dùng `CRLF.CRLF` để xác định kết thúc thông điệp

### *So sánh với HTTP:*

- ❖ HTTP: pull (kéo)
- ❖ SMTP: push (đẩy)
- ❖ Cả hai đều có tương tác lệnh/phản hồi, các mã trạng thái dạng ASCII
- ❖ HTTP: mỗi đối tượng được đóng gói trong thông điệp phản hồi của nó
- ❖ SMTP: nhiều đối tượng được gửi trong thông điệp chứa nhiều phần

Tăng Application 2-55

cuu duong than cong . com

## Định dạng thông điệp Mail

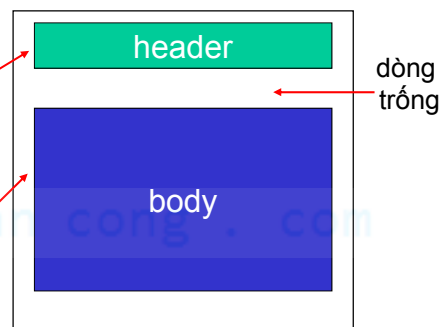
SMTP: giao thức dùng cho trao đổi thông điệp email

RFC 822: chuẩn cho định dạng thông điệp văn bản:

- ❖ Các dòng header, ví dụ
  - To:
  - From:
  - Subject:

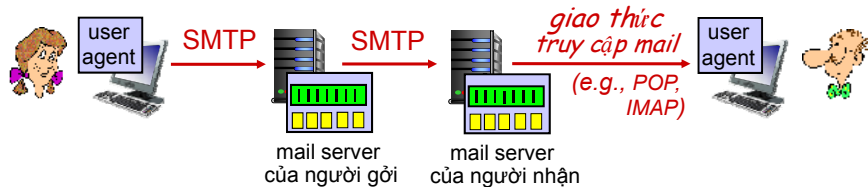
*Khác với các lệnh*  
SMTP MAIL FROM,  
RCPT TO!

- ❖ Body: "thông điệp"
  - Chỉ có các ký tự ASCII



Tăng Application 2-56

## Các giao thức truy cập Mail



- ❖ **SMTP**: truyền dẫn/lưu trữ thư vào server của người nhận
- ❖ Giao thức truy cập mail: trích xuất từ server
  - **POP**: Post Office Protocol [RFC 1939]: xác thực, tải thư về
  - **IMAP**: Internet Mail Access Protocol [RFC 1730]: nhiều tính năng hơn, bao gồm cả các thao tác thay đổi các thông điệp đang được lưu trên server
  - **HTTP**: gmail, Hotmail, Yahoo! Mail...

Tầng Application 2-57

cuu duong than cong . com

## Giao thức POP3

### Giai đoạn kiểm tra đăng nhập (authorization)

- ❖ Các lệnh phía client:
  - **user**: khai báo username
  - **pass**: password
- ❖ Đáp ứng phía server
  - **+OK**
  - **-ERR**

### Giai đoạn giao dịch

- client:
- ❖ **list**: liệt kê các số thông điệp
- ❖ **retr**: trích xuất thông điệp theo số
- ❖ **dele**: xóa
- ❖ **quit**

```

S: +OK POP3 server ready
C: user bob
S: +OK
C: pass hungry
S: +OK user successfully logged on

C: list
S: 1 498
S: 2 912
S: .
C: retr 1
S: <message 1 contents>
S: .
C: dele 1
C: retr 2
S: <message 1 contents>
S: .
C: dele 2
C: quit
S: +OK POP3 server signing off
  
```

Tầng Application 2-58

## POP3 và IMAP

### *Tìm hiểu thêm về POP3*

- ❖ Ví dụ trên sử dụng chế độ “tải xuống và xóa” của POP3
  - Bob không thể đọc lại e-mail nếu anh ta thay đổi client
- ❖ Chế độ “tải xuống-và-giữ” của POP3: sao chép các thông điệp trên các client khác nhau
- ❖ POP3 không giữ trạng thái của các phiên làm việc

### *IMAP*

- ❖ Giữ tất cả các thông điệp ở một nơi: tại server
- ❖ Cho phép người dùng tổ chức, sắp xếp các thông điệp trong các thư mục
- ❖ Giữ trạng thái của người dùng trong suốt phiên làm việc:
  - Các tên của các thư mục và ánh xạ giữa các ID của thông điệp và tên của thư mục

Tăng Application 2-59

cuu duong than cong . com

## Chương 2: Nội dung

2.1 Các nguyên lý của các ứng dụng mạng

2.2 Web và HTTP

2.3 FTP

2.4 Thư điện tử

- SMTP, POP3, IMAP

2.5 DNS

2.6 Các ứng dụng P2P

2.7 Lập trình socket với UDP và TCP

cuu duong than cong . com

Tăng Application 2-60

## DNS: domain name system

*Con người:* nhiều cách nhận dạng:

- Số an sinh xã hội, tên, số hộ chiếu

*Internet hosts, routers:*

- Địa chỉ IP (32 bit) - được dùng cho định địa chỉ gói tin
- “tên”, ví dụ www.yahoo.com - được dùng bởi con người

**Q:** làm cách nào để ánh xạ giữa địa chỉ IP và tên, và ngược lại?

**Domain Name System:**

- ❖ *Cơ sở dữ liệu phân tán* được thực hiện theo tổ chức phân cấp của nhiều *name server*
- ❖ *Giao thức tầng application:* các host, các name server trao đổi để *phân giải* tên (dịch địa chỉ  $\Leftrightarrow$  tên)
  - Lưu ý: chức năng trong phần lõi Internet, được thực hiện như là giao thức tầng application
  - Sự phức tạp ở “biên” của mạng”

Tầng Application 2-61

cuu duong than cong . com

## DNS: các dịch vụ, cấu trúc

**Các dịch vụ DNS**

- ❖ Dịch tên máy ra địa chỉ IP
- ❖ Bí danh máy
  - Lưu các tên gốc, bí danh tương ứng
- ❖ Bí danh mail server
- ❖ Cân bằng tải
  - Các bản sao cho web server: nhiều địa chỉ IP tương ứng cho 1 tên

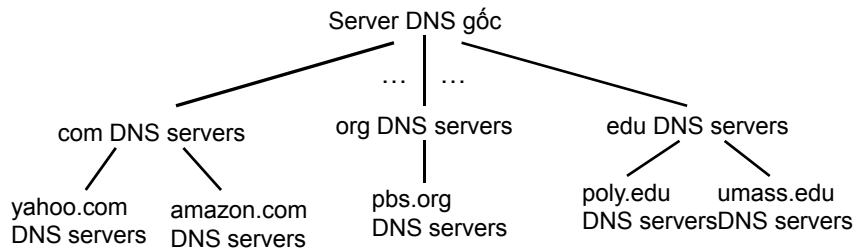
**Tại sao không tập trung hóa DNS?**

- ❖ Một điểm chịu lỗi
- ❖ Lưu lượng
- ❖ Cơ sở dữ liệu tập trung cách xa nơi yêu cầu
- ❖ Bảo trì

**A: không biến đổi được quy mô!**

Tầng Application 2-62

## DNS: cơ sở dữ liệu phân cấp, phân tán



*client muốn địa chỉ IP của www.amazon.com:*

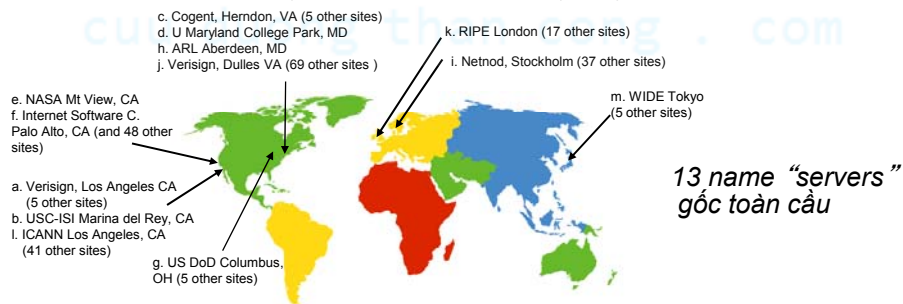
- ❖ client truy vấn server gốc (root) để tìm DNS server quản lý vùng ".com"
- ❖ client truy vấn DNS server ".com" tìm DNS server quản lý vùng "amazon.com"
- ❖ client truy vấn DNS server "amazon.com" để lấy địa chỉ IP của www.amazon.com

Tăng Application 2-63

cuu duong than cong . com

## DNS: các name server gốc

- ❖ Được name server cục bộ liên lạc để hỏi khi không thể phân giải tên
- ❖ name server gốc:
  - Liên lạc với name server có thẩm quyền (authoritative name server) nếu ánh xạ tên không xác định
  - Lấy ánh xạ
  - Trả về ánh xạ đến name server cục bộ



Tăng Application 2-64



## TLD, server có thẩm quyền

### *Các top-level domain (TLD) server :*

- Chịu trách nhiệm cho tên miền com, org, net, edu, aero, jobs, museums, và tất cả các tên miền cấp cao nhất của quốc gia, như là: uk, fr, ca, jp
- Công ty Network Solutions quản lý máy chủ chứa các thông tin của vùng .com TLD
- Tổ chức Educause quản lý .edu TLD

### *Các DNS server có thẩm quyền:*

- Các tổ chức sở hữu các DNS server riêng nhằm cung cấp các tên được cấp phép và ánh xạ địa chỉ IP cho các host được đặt tên của tổ chức đó
- Có thể được quản lý bởi tổ chức hoặc nhà cung cấp dịch vụ

Tăng Application 2-65

cuu duong than cong . com

## DNS name server cục bộ

- ❖ Không hoàn toàn theo cấu trúc phân cấp
- ❖ Mỗi ISP (ISP, công ty, trường đại học) có một server cục bộ như vậy
  - Còn được gọi là “default name server”
- ❖ Khi một host tạo một truy vấn DNS, truy vấn được gửi đến DNS server cục bộ của nó
  - Có bộ nhớ đệm (cache) cục bộ chứa thông tin về các cặp tên-đến-địa chỉ gần đây (nhưng có thể hết hạn!)
  - Hoạt động như một proxy, chuyển truy vấn vào trong hệ thống DNS phân cấp

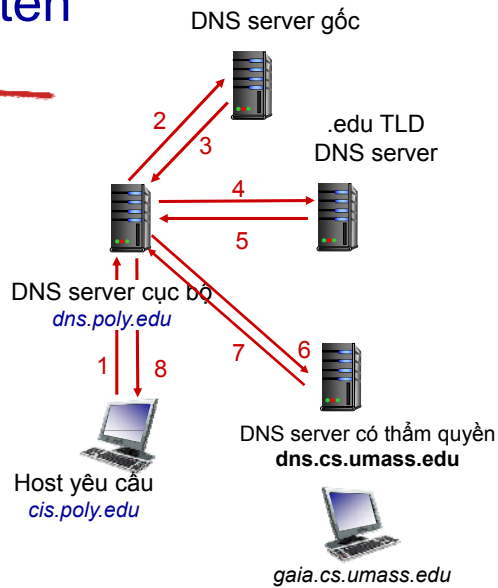
Tăng Application 2-66

## Ví dụ phân giải tên miền DNS

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

### Truy vấn tuần tự:

- ❖ Server được hỏi sẽ trả lời với tên của server quản lý vùng liên quan
- ❖ “tôi không biết tên này, nhưng hãy hỏi thêm thông tin từ máy chủ này”



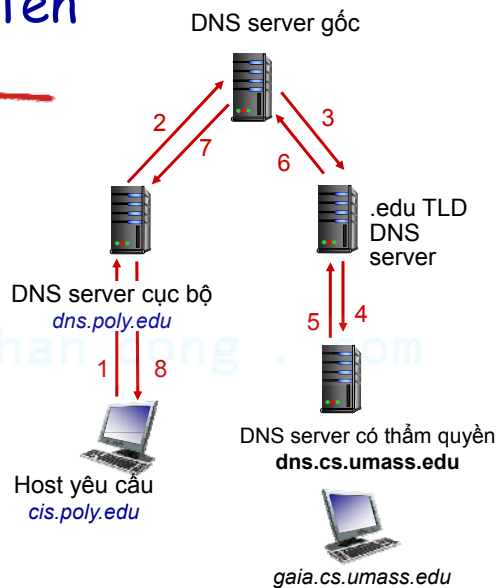
Tăng Application 2-67

cuu duong than cong . com

## Ví dụ phân giải tên DNS

### Truy vấn đệ quy:

- ❖ Đẩy trách nhiệm phân giải tên cho name server được hỏi
- ❖ Tải nặng tại các tầng trên của hệ thống phân cấp?



Tăng Application 2-68

## DNS: caching, cập nhật các bản ghi

- ❖ Một khi name server biết về 1 ánh xạ tên-địa chỉ, nó sẽ *lưu tạm* ánh xạ đó
  - Các mục cache hết hạn (sẽ bị xóa) sau một khoảng thời gian (TTL)
  - Thông tin trong các TLD servers thường được lưu tạm trong các name server cục bộ
    - Do đó các name server gốc không bị truy cập thường xuyên
- ❖ Các mục được lưu tạm có thể hết hạn
  - Nếu cập thông tin tên-địa chỉ IP thay đổi, có thể các máy khác trên Internet không biết được cho đến khi tất cả TTL hết hạn.
- ❖ Cơ chế cập nhật/thông báo được đề xuất trong bộ chuẩn IETF
  - RFC 2136

Tầng Application 2-69

cuu duong than cong . com

## Các bản ghi DNS

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

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

### type=A

- **name** là tên host
- **value** là địa chỉ IP

### type=NS

- **Name** là tên miền (e.g., foo.com)
- **value** là tên host của name server có thẩm quyền quản lý tên miền này

### type=CNAME

- **name** là bí danh của một tên "gốc" (tên thực)
- VD: **www.ibm.com** tên thực là **serveeast.backup2.ibm.com**
- **value** là tên gốc

### type=MX

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

Tầng Application 2-70

## Giao thức và các thông điệp DNS

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

### msg header

- ❖ **identification:** số 16 bit xác định 1 truy vấn, hoặc trả lời cho truy vấn có cùng số này
- ❖ **flags:**
  - Truy vấn hoặc trả lời
  - Mong muốn đệ quy
  - Đệ quy sẵn sàng
  - Trả lời có thẩm quyền

← 2 bytes → ← 2 bytes →	
identification	flags
# questions	# answer RRs
# authority RRs	# additional RRs
questions (variable # of questions)	
answers (variable # of RRs)	
authority (variable # of RRs)	
additional info (variable # of RRs)	

Tăng Application 2-71

cuu duong than cong . com

## Giao thức và các thông điệp DNS

	← 2 bytes →	← 2 bytes →
	identification	flags
	# questions	# answer RRs
	# authority RRs	# additional RRs
Các trường name, type cho một truy vấn	questions (variable # of questions)	
Các RRs để trả lời truy vấn	answers (variable # of RRs)	
Các bản ghi thông tin về các server có thẩm quyền	authority (variable # of RRs)	
thông tin "hữu ích" bổ sung có thể sẽ dùng	additional info (variable # of RRs)	

Tăng Application 2-72

## Thêm các bản ghi vào trong DNS

- ❖ Ví dụ: khởi tạo mới công ty “Network Utopia”
- ❖ Đăng ký tên miền networkutopia.com tại một *DNS registrar - tổ chức nhận đăng ký tên miền* (như là Network Solutions)
  - Cung cấp tên, địa chỉ IP của name server có thẩm quyền quản lý tên miền này (primary và secondary)
  - Tổ chức quản lý tên miền thêm hai bản ghi vào trong server quản lý vùng .com:  
(networkutopia.com, dns1.networkutopia.com, NS)  
(dns1.networkutopia.com, 212.212.212.1, A)
- ❖ Tạo bản ghi type A trong server có thẩm quyền quản lý networkutopia.com cho www.networkutopia.com; và bản ghi type MX cho mail server thuộc networkutopia.com

Tầng Application 2-73

cuu duong than cong . com

## Tấn công DNS

### Tấn công DDoS

- ❖ Đẩy khối lượng dữ liệu lớn tới các server gốc
  - Không thành công cho đến nay
  - Lọc lưu lượng
  - Các DNS server cục bộ lưu tạm các địa chỉ IP của TLD servers, vì thế không cần truy cập server gốc
- ❖ Đẩy khối lượng dữ liệu lớn tới TLD server
  - Nguy hiểm hơn

### Tấn công chuyển hướng

- ❖ Man-in-middle
  - Ngăn chặn các truy vấn
- ❖ Đầu độc DNS
  - Gửi các trả lời giả tạo đến các DNS server, (sẽ được các server lưu lại)

### Khai thác DNS cho tấn công DDoS

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

Application Layer 2-74

## Chương 2: Nội dung

- 2.1 Các nguyên lý của các ứng dụng mạng
- 2.2 Web và HTTP
- 2.3 FTP
- 2.4 Thư điện tử
  - SMTP, POP3, IMAP
- 2.5 DNS

- 2.6 Các ứng dụng P2P
- 2.7 Lập trình socket với UDP và TCP

Tầng Application 2-75

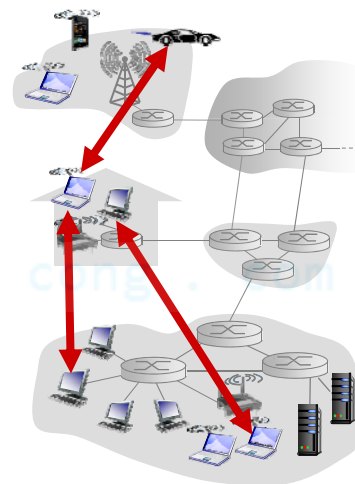
cuu duong than cong . com

## Kiến trúc P2P thuần túy

- ❖ Không có server hoạt động liên tục
- ❖ Các hệ thống đầu cuối bất kỳ giao tiếp trực tiếp với nhau
- ❖ Các máy được kết nối không liên tục và thay đổi địa chỉ IP

*Ví dụ:*

- Phân phối file (BitTorrent)
- Streaming (KanKan)
- VoIP (Skype)

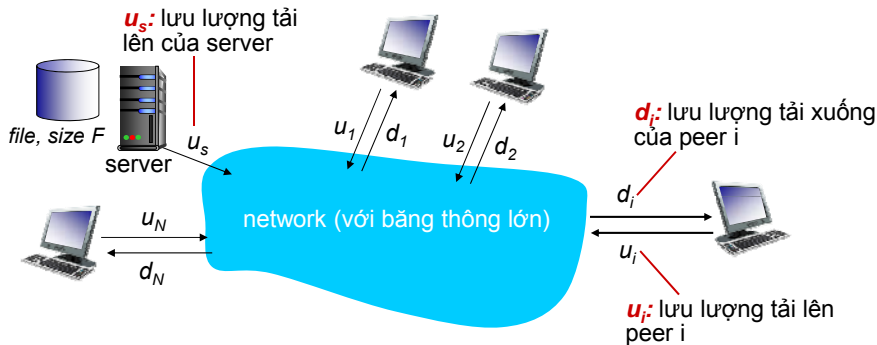


Tầng Application 2-76

## Phân phối tập tin: so sánh giữa mô hình client-server và P2P

**Câu hỏi:** mất bao lâu để phân phối file (kích thước  $F$ ) từ một server đến  $N$  máy?

- Lưu lượng tải lên/tải xuống của peer là tài nguyên giới hạn



Tăng Application 2-77

cuu duong than cong . com

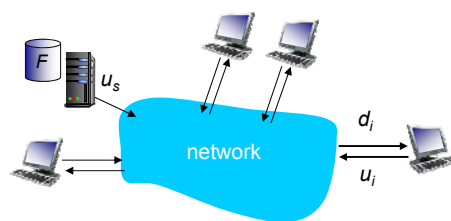
## Thời gian phân phối file: client-server

❖ **Server truyền:** phải gửi (tải lên) tuần tự  $N$  bản sao file:

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

❖ **client:** mỗi client phải tải xuống bản sao của file

- $d_{\min}$  = tốc độ tải xuống của client tải chậm nhất
- Thời gian để client tải chậm nhất tải xong:  $F/d_{\min}$



Thời gian để phân phối tập tin  $F$  đến  $N$  client dùng phương pháp client-server

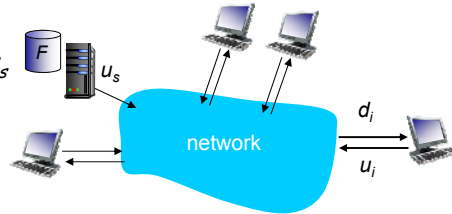
$$D_{c-s} \geq \max\{NF/u_s, F/d_{\min}\}$$

Tăng tuyến tính trong  $N$

Tăng Application 2-78

## Thời gian phân phối file: P2P

- ❖ **Server:** chỉ cần tải lên ít nhất một bản sao
  - Thời gian gửi một bản sao:  $F/u_s$
- ❖ **một client:** client phải download bản sao file
  - Thời gian tối thiểu để client tải xuống:  $F/d_{\min}$
- ❖ **nhiều client:** với  $N$  máy thì tổng dung lượng tải về là  $NF$  bit
  - Tốc độ upload tối đa (không ché tốc độ tải về tối đa) là  $u_s + \sum u_i$



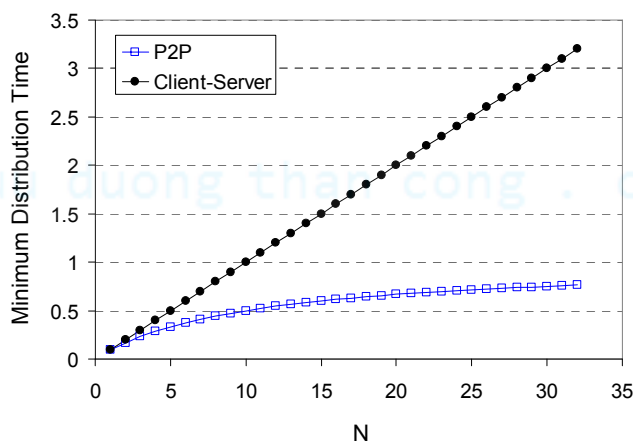
Thời gian phân phối  
 $F$  đến  $N$  client dùng phương pháp P2P  $D_{P2P} \geq \max\{F/u_s, F/d_{\min}, NF/(u_s + \sum u_i)\}$

Tăng tuyến tính trong  $N$  ...  
 ...nhưng mỗi khi thêm peer tham gia sử dụng dịch vụ,  
 chính nó lại gia tăng năng lực dịch vụ

Tăng Application 2-79

## So sánh Client-server với P2P: ví dụ

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



Tăng Application 2-80

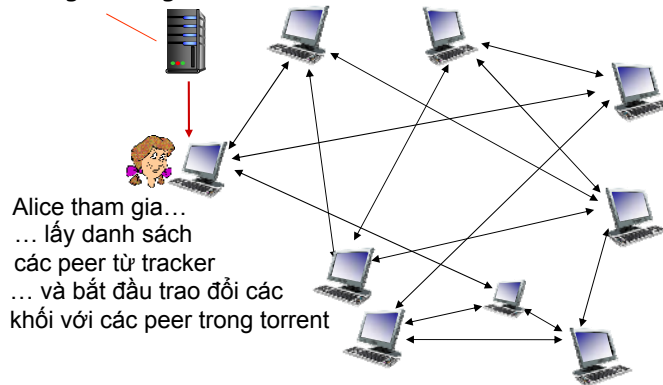


## Phân phối file P2P: BitTorrent

- ❖ Tập tin được chia thành các khối 256Kb (chunks)
- ❖ Các peer trong in torrent gửi/nhận các khối

*tracker*: theo dõi các peers tham gia trong torrent

*torrent*: nhóm các peer trao đổi các khối

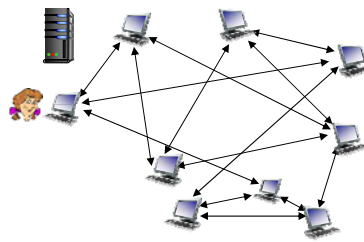


Tăng Application 2-81

cuu duong than cong . com

## Phân phối file P2P: BitTorrent

- ❖ peer tham gia torrent:
  - không có các khối, nhưng sẽ lần lượt tích lũy chúng từ các peer khác
  - đăng ký với tracker để lấy danh sách các peer khác, kết nối với peer khác và cả "láng giềng" của các peer khác
- ❖ trong khi tải xuống, peer tải các khối dữ liệu nó đã có tới các peer khác
- ❖ peer có thể thay đổi các peer mà nó đang trao đổi các khối dữ liệu
- ❖ peers có thể tham gia hay rời bỏ nhóm phân phối
- ❖ Một khi peer có toàn bộ file, nó có thể (ích kỷ) rời khỏi hoặc (vị tha) ở lại trong nhóm



Tăng Application 2-82

## BitTorrent: yêu cầu, gọi các khối

### *Yêu cầu các khối:*

- ❖ tại bất kỳ thời điểm nào, các peer khác nhau có các tập con khác nhau của các khối
- ❖ định kỳ, Alice yêu cầu mỗi peer cho danh sách các khối mà các peer có
- ❖ Alice yêu cầu các khối đang thiếu từ các peer, hiem trước

### *Gọi các khối: tit-for-tat*

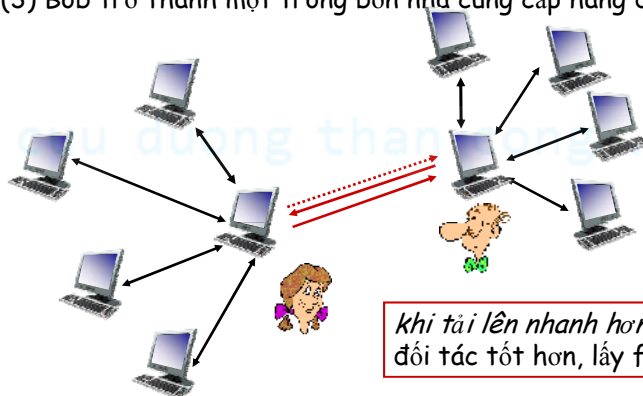
- ❖ Alice gọi các khối cho bốn peer nào hiện tại đang gọi các khối cho cô ở tốc độ cao nhất
  - Alice xiết các yêu cầu từ các peer khác (sẽ không nhận được khối từ Alice)
  - Đánh giá lại top 4 mỗi 10 giây
- ❖ Mỗi 30 giây: chọn ngẫu nhiên một peer khác, bắt đầu gọi các khối
  - Không xiết các yêu cầu của peer này
  - Peer mới được chọn có thể tham gia và top 4

Tăng Application 2-83

cuu duong than cong . com

## BitTorrent: tit-for-tat

- (1) Alice cho Bob kết nối
- (2) Alice trở thành một trong bốn nhà cung cấp hàng đầu của Bob; Bob đáp lại
- (3) Bob trở thành một trong bốn nhà cung cấp hàng đầu của Alice



Tăng Application 2-84

## Distributed Hash Table (DHT)

- ❖ Bảng Hash
- ❖ Mô hình DHT
- ❖ Circular DHT and overlay networks
- ❖ Peer churn

cuu duong than cong . com

## Cơ sở dữ liệu đơn giản

Cơ sở dữ liệu phân tán (*distributed P2P database*) đơn giản với cặp (*key, value*):

- key: số an sinh xã hội; value: tên người

Key	Value
132-54-3570	John Washington
761-55-3791	Diana Louise Jones
385-41-0902	Xiaoming Liu
441-89-1956	Rakesh Gopal
217-66-5609	Linda Cohen
.....	.....
177-23-0199	Lisa Kobayashi

- key: tên phim; value: địa chỉ IP

## Bảng Hash

- thuận tiện hơn để lưu trữ và tìm kiếm trên đại diện số của key
- $\text{key} = \text{hash}(\text{original key})$

Original Key	Key	Value
132-54-3570	8962458	John Washington
761-55-3791	7800356	Diana Louise Jones
385-41-0902	1567109	Xiaoming Liu
441-89-1956	2360012	Rakesh Gopal
217-66-5609	5430938	Linda Cohen
.....		.....
177-23-0199	9290124	Lisa Kobayashi

cuu duong than cong . com

## Distributed Hash Table (DHT)

- ❖ Phân phối các cặp (key, value) qua hàng triệu các peer
  - Các cặp được phân bố đều trên các peer
- ❖ Bất kỳ peer nào cũng có thể **truy vấn** trên DHT bằng cách dùng key
  - Cơ sở dữ liệu trả về giá trị trùng key đó
  - Để giải quyết truy vấn, số lượng nhỏ các thông điệp được trao đổi giữa các peer
- ❖ Mỗi peer chỉ biết một số nhỏ các peer khác

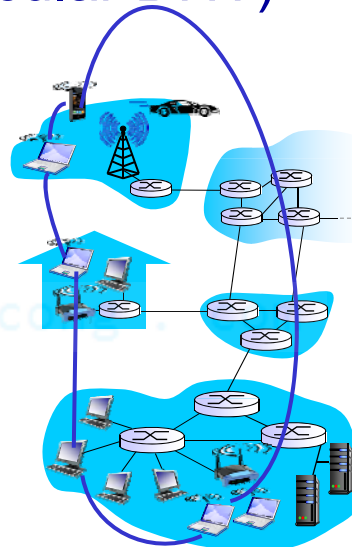
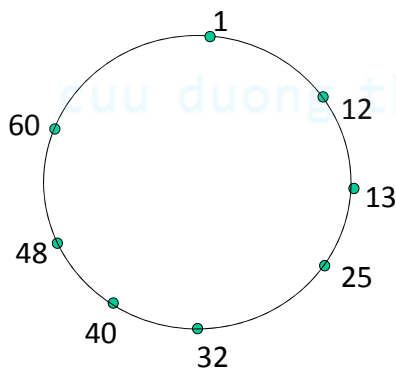
## Chỉ định các cặp key-value cho các peer

- ❖ Quy tắc: chỉ định cặp key-value đến peer mà có ID *gần nhất (closest)*.
- ❖ Quy ước: gần nhất(closest) is *sự kế thừa ngay lập tức (immediate successor)* của khóa (key) đó.
- ❖ Ví dụ: không gian ID  $\{0,1,2,3,\dots,63\}$
- ❖ Giả sử 8 peer: 1,12,13,25,32,40,48,60
  - Nếu key = 51, thì được chỉ định cho peer 60
  - Nếu key = 60, thì được chỉ định cho peer 60
  - Nếu key = 61, thì được chỉ định cho peer 1

cuu duong than cong . com

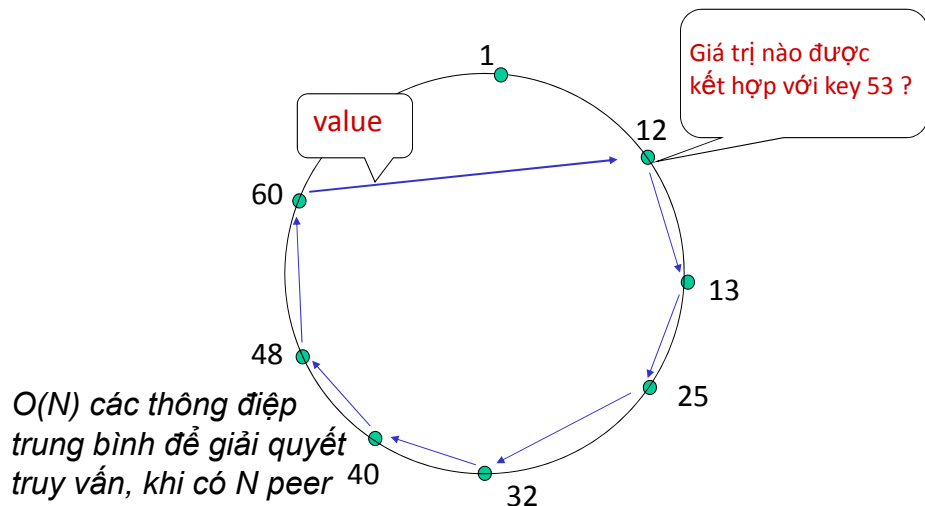
## DHT vòng tròn (Circular DHT)

Mỗi peer chỉ nhận thức được người lập tức kế nhiệm và người tiền nhiệm

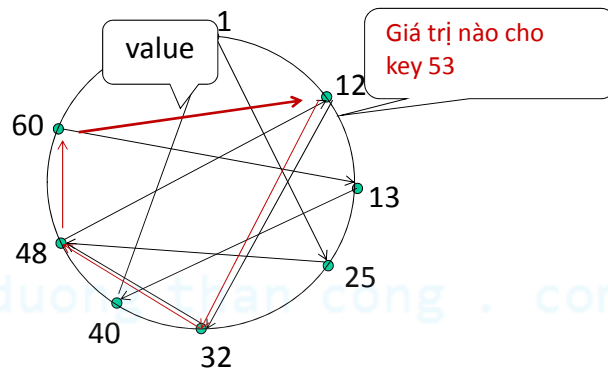


“overlay network”

## Giải quyết một truy vấn

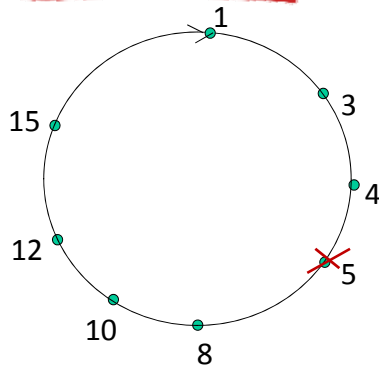


## DHT vòng tròn với đường tắt



- Mỗi peer theo dõi đại chỉ IP của người tiền nhiệm, người kế nhiệm, đường tắt.
- Giảm từ 6 còn 3 thông điệp.
- Có thể thiết kế các đường tắt với  $O(\log N)$  láng giềng,  $O(\log N)$  thông điệp trong truy vấn

## Peer churn



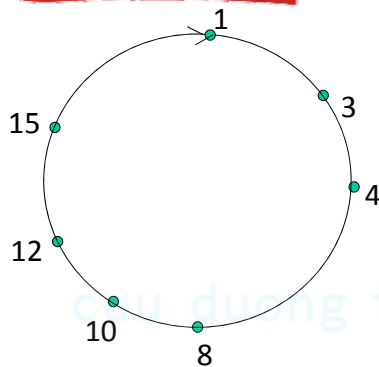
Ví dụ: *peer 5 đột ngột rời khỏi* mới

### Xử lý peer churn:

- ❖ Các peer có thể tham gia và rời khỏi (churn) nhóm
- ❖ Mỗi peer biết địa chỉ của hai kế nhiệm của nó
- ❖ Mỗi peer định kỳ ping hai kế nhiệm của nó để kiểm tra sự tồn tại
- ❖ Nếu người vừa kế nhiệm bỏ đi, thì chọn kế nhiệm kế tiếp như là người kế nhiệm tức thời mới

cuu duong than cong . com

## Peer churn



Ví dụ: *peer 5 đột ngột rời khỏi* mới

### Xử lý peer churn:

- ❖ Các peer có thể tham gia và rời khỏi (churn) nhóm
- ❖ Mỗi peer biết địa chỉ của hai kế nhiệm của nó
- ❖ Mỗi peer định kỳ ping hai kế nhiệm của nó để kiểm tra sự tồn tại
- ❖ Nếu người vừa kế nhiệm bỏ đi, thì chọn kế nhiệm kế tiếp như là người kế nhiệm tức thời mới

- ❖ peer 4 phát hiện sự rời khỏi của peer 5; peer 8 trở thành người kế nhiệm ngay lập tức của nó
- ❖ 4 yêu cầu 8 là người kế nhiệm tức thời của nó; người kế nhiệm tức thời của 8 trở thành người kế nhiệm thứ 2 của 4.

## Chương 2: Nội dung

2.1 Các nguyên lý của các ứng dụng mạng

2.2 Web và HTTP

2.3 FTP

2.4 Thư điện tử

- SMTP, POP3, IMAP

2.5 DNS

2.6 Các ứng dụng P2P

2.7 Lập trình socket với UDP và TCP

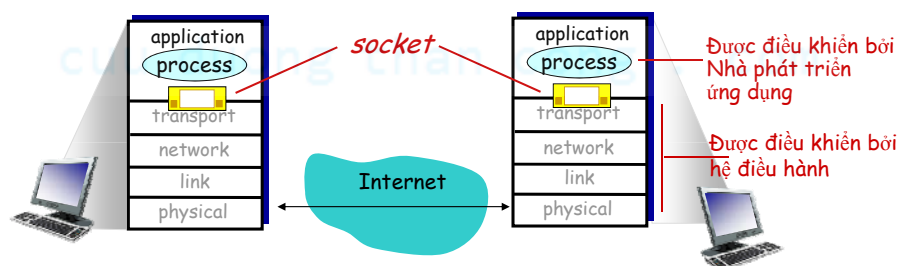
Tầng Application 2-95

cuu duong than cong . com

## Lập trình Socket

**Mục tiêu:** tìm hiểu cách xây dựng các ứng dụng client/server liên lạc bằng sockets

**socket:** một cánh cửa giữa tiến trình ứng dụng và giao thức vận chuyển giữa 2 đầu cuối



Tầng Application 2-96



## Lập trình Socket

*Hai loại socket cho hai dịch vụ transport:*

- **UDP:** chuyển gói tin không đảm bảo
- **TCP:** chuyển luồng tin có đảm bảo (stream-oriented)

*Ứng dụng ví dụ:*

1. Client nhận một dòng các ký tự (dữ liệu) từ bàn phím của nó và gửi dữ liệu đó đến server.
2. Server nhận được dữ liệu đó và chuyển đổi các ký tự sang chữ hoa.
3. Server gửi dữ liệu đã được sửa đổi cho client ở trên.
4. Client này nhận được dữ liệu đã bị sửa đổi và hiển thị dòng đó lên màn hình của nó.

Tăng Application 2-97

cuu duong than cong . com

## Lập trình Socket với UDP

**UDP: không “kết nối” giữa client và server**

- ❖ Không bắt tay trước khi gửi dữ liệu
- ❖ Bên gửi chỉ rõ địa chỉ IP đích và số port cho mỗi packet
- ❖ Bên nhận lấy địa chỉ IP và số port của bên gửi từ packet nhận được

**UDP: dữ liệu được truyền có thể bị mất hoặc được nhận không đúng thứ tự**

**Quan điểm ứng dụng:**

- ❖ UDP cung cấp cơ chế truyền các nhóm bytes (“datagrams”) không tin cậy giữa client và server

Tăng Application 2-98

## Sự tương tác socket Client/server: UDP

### server (chạy với địa chỉ serverIP)

create socket, port= x:  
`serverSocket =  
socket(AF_INET,SOCK_DGRAM)`

↓  
read datagram from  
`serverSocket`

↓  
write reply to  
`serverSocket`  
specifying clientIP,  
port number

### client

create socket:  
`clientSocket =  
socket(AF_INET,SOCK_DGRAM)`

↓  
create datagram with serverIP  
and port=x; send datagram via  
`clientSocket`

↓  
read datagram from  
`clientSocket`

↓  
close  
`clientSocket`

Application 2-99

cuu duong than cong . com

## Ứng dụng ví dụ: UDP client

### *Python UDPClient*

Bao gồm thư viện socket  
của Python' →

```
from socket import *  
serverName = 'hostname'  
serverPort = 12000
```

Tạo socket UDP cho  
server →

```
clientSocket = socket(socket.AF_INET,  
socket.SOCK_DGRAM)
```

Nhận thông điệp từ bàn phím  
người dùng →

```
message = raw_input('Input lowercase sentence:')
```

Đính kèm tên server,  
port đến thông điệp; gửi  
vào socket →

```
clientSocket.sendto(message,(serverName, serverPort))
```

Đọc các ký tự trả lời từ  
socket vào chuỗi →

```
modifiedMessage, serverAddress =  
clientSocket.recvfrom(2048)
```

In ra chuỗi được nhận và  
đóng socket →

```
print modifiedMessage  
clientSocket.close()
```

Tăng Application 2-100

## Ứng dụng ví dụ : UDP server

### *Python UDPServer*

```
from socket import *
serverPort = 12000

Tạo UDP socket → serverSocket = socket(AF_INET, SOCK_DGRAM)
Gắn kết socket đến số port cục bộ 12000 → serverSocket.bind(("", serverPort))
print "The server is ready to receive"

Lặp mãi mãi → while 1:
    Đọc từ UDP socket vào message, lấy địa chỉ IP của client (địa chỉ IP client và port) → message, clientAddress = serverSocket.recvfrom(2048)
    Gởi chuỗi chữ hoa trở lại cho client này → modifiedMessage = message.upper()
    serverSocket.sendto(modifiedMessage, clientAddress)
```

Tăng Application 2-101

cuuduongthancong.com

## Lập trình Socket với TCP

### **client phải liên lạc với server**

- ❖ Tiến trình server phải được chạy trước
- ❖ server phải tạo socket (cửa) để mời client đến liên lạc

### **client tiếp xúc server bằng:**

- ❖ Tạo socket TCP, xác định địa chỉ IP, số port của tiến trình server
- ❖ **Khi client tạo socket:** client TCP thiết lập kết nối đến server TCP

❖ Khi được client liên lạc, **server TCP tạo socket mới** cho tiến trình server để trao đổi với client đó

- Cho phép server nói chuyện với nhiều client
- Số source port được dùng để phân biệt các client (xem tiếp chương 3)

### **Quan điểm ứng dụng:**

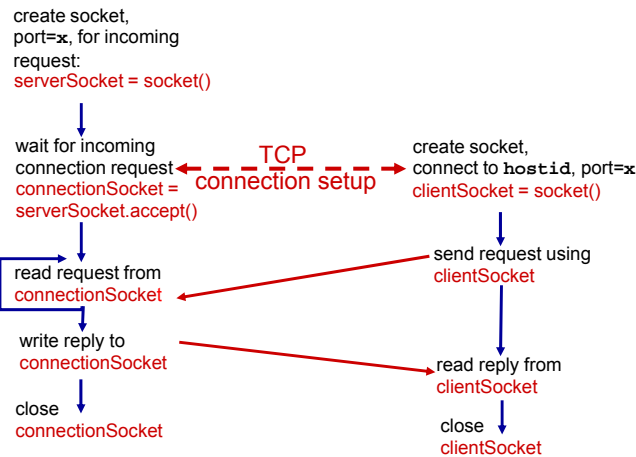
TCP cung cấp việc truyền các byte đáng tin cậy và theo thứ tự giữa client và server

Tăng Application 2-102

## Tương tác socket Client/server: TCP

### server (chạy trên hostid)

### client



Tăng Application 2-103

cuu duong than cong . com

## Ứng dụng ví dụ : TCP client

### *Python TCPClient*

```
from socket import *
serverName = 'servername'
serverPort = 12000
clientSocket = socket(AF_INET, SOCK_STREAM)
clientSocket.connect((serverName, serverPort))
sentence = raw_input('Input lowercase sentence:')
clientSocket.send(sentence)
modifiedSentence = clientSocket.recv(1024)
print 'From Server:', modifiedSentence
clientSocket.close()
```

Tạo TCP socket cho  
server, port 12000

Không cần đính kèm tên  
server, port

Tăng Application 2-104

## Ví dụ ứng dụng: TCP server

### *Python TCPServer*

	from socket import *
	serverPort = 12000
Tạo socket TCP chào đón	→ serverSocket = socket(AF_INET, SOCK_STREAM)
	serverSocket.bind(('', serverPort))
server bắt đầu lắng nghe các yêu cầu TCP đến	→ serverSocket.listen(1)
	print 'The server is ready to receive'
Lặp mãi mãi	→ while 1:
server đợi accept() cho yêu cầu đến, socket mới được tạo trở về	→ connectionSocket, addr = serverSocket.accept()
Đọc các byte từ socket (nhưng không đọc địa chỉ như UDP)	→ sentence = connectionSocket.recv(1024)
	capitalizedSentence = sentence.upper()
	connectionSocket.send(capitalizedSentence)
Đóng kết nối đến client này (nhưng không đóng socket chào đón)	→ connectionSocket.close()

Tăng Application 2-105

cuuduongthancong.com

## Chương 2: tóm tắt

- ❖ Các kiến trúc ứng dụng
  - client-server
  - P2P
- ❖ Các yêu cầu về dịch vụ ứng dụng:
  - Độ tin cậy, băng thông, độ trễ
- ❖ Mô hình dịch vụ vận chuyển Internet
  - Kết nối định hướng, tin cậy: TCP
  - Không tin cậy, datagrams: UDP
- ❖ Các giao thức:
  - HTTP
  - FTP
  - SMTP, POP, IMAP
  - DNS
  - P2P: BitTorrent, DHT
- ❖ Lập trình socket : TCP, UDP sockets

Tăng Application 2-106

## Chương 2: tóm tắt

*Quan trọng: tìm hiểu về các giao thức!*

- ❖ trao đổi thông điệp yêu cầu / trả lời điển hình:
  - client yêu cầu thông tin hoặc dịch vụ
  - server đáp ứng với dữ liệu, mã trạng thái
- ❖ Các định dạng thông điệp:
  - headers: các trường cho biết thông tin về dữ liệu
  - data: thông tin để truyền thông

*Các chủ đề quan trọng:*

- ❖ Điều khiển với các thông điệp dữ liệu
  - in-band, out-of-band
- ❖ Tập trung và không tập trung
- ❖ Không trạng thái và có trạng thái
- ❖ Truyền tin cậy và không tin cậy
- ❖ “sự phức tạp tại mạng biên”

Tăng Application 2-107

cuu duong than cong . com

cuu duong than cong . com