

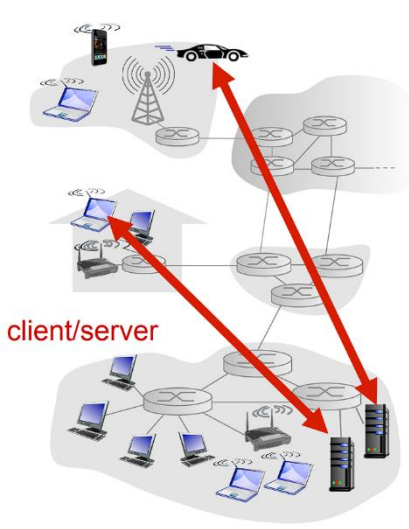
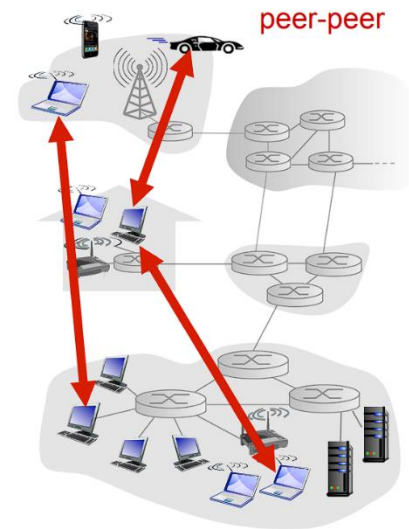


NHẬP MÔN MẠNG MÁY TÍNH

Chương 2: Tầng Application



Các kiến trúc ứng dụng

	Client – Server	P2P (Peer to peer)
Minh Họa	 <p>client/server</p>	 <p>peer-peer</p>
Server	Luôn hoạt động	Không có Server luôn luôn hoạt động
	Địa chỉ IP cố định.	
	Trung tâm phục vụ và lưu trữ dữ liệu.	
Client	Giao tiếp với Server. Không giao tiếp trực tiếp với các Client khác.	Các hệ thống đầu cuối giao tiếp trực tiếp với nhau.
	Có thể kết nối không liên tục, dùng địa chỉ IP động.	Các Peer được kết nối không liên tục và có thể thay đổi địa chỉ IP. ⇒ Quản lý phức tạp
	Clients gửi request và nhận response từ Server. Giao tiếp với Clients khác qua Server	Các Peer vừa có thể tiêu thụ và cung cấp tài nguyên cho các peer khác ⇒ Khả năng mở rộng tốt



Các tiến trình liên lạc

Tiến trình (Process):

Là chương trình chạy trong một Host.

- Trong cùng một host, hai tiến trình giao tiếp với nhau bằng cách sử dụng 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 bởi trao đổi *các thông điệp (Message)*.
- *Clients, Server:*
 - o *Tiến trình Client:* tiến trình khởi tạo truyền thông.
 - o *Tiến trình Server:* tiến trình chờ đợi để được liên lạc.
- **Chú ý:** Các ứng dụng với kiến trúc P2P có cả tiến trình server và client.

Sockets

Là điểm truy cập dịch vụ ở tầng giao vận (transport), tương tự như cổng ra vào.

- Các tiến trình sử dụng socket gọi dịch vụ của tầng giao vận để trao đổi thông điệp.
- Truy cập dịch vụ bằng *định danh (identifier)* – bao gồm **địa chỉ IP** và **số cổng (Port numbers)** được liên kết với tiến trình trên host.

** Một số Port tầng Application*

Protocol	Port Number	Protocol	Port Number
FTP	20	DNS	53
HTTP	80	SSH	22



HTTPS	443	TFTP	69
SMTP	25	Telnet	23
LPD	515	NFS	2049

Các yêu cầu dịch vụ vận chuyển

Ứng dụng	Toàn vẹn dữ liệu	Băng thông	Time sensitive
Truyền file	Không	Mềm dẻo	Không
E-mail	Không	Mềm dẻo	Không
Web ocuments	Không	Mềm dẻo	Không
Audio/video thời gian thực	Chịu mất mát	Audio: 5kbps – 1Mbps Video: 10kbps – 5Mbps	Có, 100's msec
Audio/video đã lưu	Chịu mất mát	Audio: 5kbps – 1Mbps Video: 10kbps – 5Mbps	Có, vài giây
Game tương tác	Chịu mất mát	Trên một vài kbps	Có, 100's msec
Text messaging	Không	Mềm dẻo	Có và Không

Các dịch vụ giao thức Transport Internet

Dịch vụ TCP:



- Là giao thức điều khiển truyền nhận. Giao thức này đảm bảo sự chuyển giao thông tin từ nơi nguồn tới nơi nhận một cách an toàn, bảo đảm toàn vẹn và đúng thứ tự.
- TCP có thể điều tiết được người gửi khi mạng quá tải và thiết lập yêu cầu giữa tiến trình client và server.

Dịch vụ UDP:

- Là một giao thức cho phép gửi các gói tin mà không cần dựng một kết nối hoàn chỉnh giữa các thiết bị kết nối mạng
- Không hỗ trợ độ tin cậy, điều khiển luồng, điều khiển tắc nghẽn, bảo đảm thông lượng, bảo mật, và thiết lập kết nối.

So sánh TCP – UDP:

Giống: TCP và UDP đều là các giao thức được sử dụng để gửi các bit dữ liệu – được gọi là các gói tin – qua Internet. Cả hai giao thức đều được xây dựng trên giao thức IP.

Khác:

	TCP	UDP
Thiết Kế	Định hướng kết nối (connection oriented).	Kém kết nối hơn (connectionless).
Độ tin cậy	Độ tin cậy cao.	Độ tin cậy thấp.
Truyền dữ liệu	Gửi gói tin dạng luồng, xử lý theo thứ tự gói tin.	Dữ liệu không được truyền theo thứ tự.
Hiệu năng	Tốc độ truyền thấp hơn.	Tốc độ truyền cao hơn.
Xử lý vấn đề về gói tin	Gói tin có thể được gửi lại nếu cần hoặc bị mất.	Gói tin không được truyền lại

Web và HTTP

HTTP (Hypertext Transfer Protocol) là giao thức web ở tầng Application.

Mô hình client/server:

- **Client:** trình duyệt yêu cầu và nhận (sử dụng giao thức HTTP) ⇒ Hiển thị các Object của Website.

- **Server**: Web server gửi (sử dụng giao thức HTTP) các Object đáp ứng yêu cầu của Client.
- **RTT (Round Trip Time)**: Thời gian để 1 gói tin nhỏ đi từ Client đến Server và quay ngược lại.

Các kết nối HTTP

HTTP Không Bền Vững (Nonpersistent HTTP)	HTTP Bền Vững (Persistent HTTP)	
<ul style="list-style-type: none"> - Chỉ tối đa 1 đối tượng được gửi qua kết nối TCP, sau đó kết nối sẽ bị đóng. - Tải nhiều đối tượng yêu cầu nhiều kết nối. 	<ul style="list-style-type: none"> - Nhiều đối tượng có thể được gửi qua một kết nối TCP giữa Client và Server. 	
HTTP/1.0 (RFC 1945)	HTTP/1.1 (RFC 2616)	
Thời gian đáp ứng <ul style="list-style-type: none"> - Một RTT để khởi tạo kết nối TCP. - Một RTT cho yêu cầu HTTP và vài byte đầu tiên của đáp ứng HTTP được trả về. - Thời gian đáp ứng = $2RTT + \text{thời gian truyền file}$. 	Thời gian đáp ứng Persistent without pipelining <ul style="list-style-type: none"> - Client chỉ gửi request khi đã nhận được request trước. - 1 RTT cho 1 đối tượng được quan tâm. 	Persistent with pipelining <ul style="list-style-type: none"> - Client gửi request liên tục đến các đối tượng được quan tâm. - Có thể 1 RTT cho tất cả các đối tượng được quan tâm.
Các Phương thức <ul style="list-style-type: none"> - GET - POST - HEAD 	Các Phương thức <ul style="list-style-type: none"> - GET - POST - HEAD - PUT - DELETE 	

HTTP Cookies: lưu trữ trạng thái User-server

- Có thể được sử dụng cho:
 - o Sự cấp phép (authorization).
 - o Giỏ mua hàng (shopping carts).
 - o Các khuyến cáo (recommendations).
 - o Trạng thái phiên làm việc của User (user session state).

Các mã trạng thái đáp ứng HTTP (HTTP response status codes)

Xuất hiện trong dòng đầu tiên trong thông điệp đáp ứng từ Server.

Một vài mã trạng thái thường gặp:

- **200 OK** – Yêu cầu thành công, đối tượng được yêu cầu sau trong thông điệp này.
- **301 Moved Permanently** – Đối tượng được yêu cầu đã di chuyển, vị trí mới được xác định sau trong thông điệp này.
- **304 Not Modified** – Đối tượng được yêu cầu chưa được điều chỉnh (và đã lưu trong cache), client có thể sử dụng đối tượng được phản hồi trong bộ nhớ cache.
- **400 Bad Request** – Thông điệp yêu cầu không được hiểu bởi Server.
- **404 Not Found** – Tài liệu được yêu cầu không tìm thấy trên server này.

Web Caches (Proxy Server)

Mục tiêu:

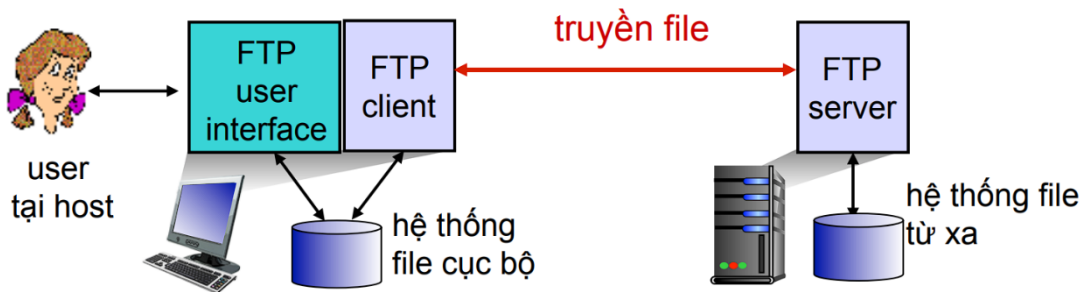
- Đáp ứng yêu cầu của client mà không cần liên quan đến server.
- Giảm thời gian đáp ứng yêu cầu + giảm tải cho client.
- Giảm lưu lượng trên đường liên kết truy cập ra internet.

- Caches dày đặc trên Internet \Rightarrow Nhà cung cấp dịch vụ có thể cung cấp nội dung hiệu quả hơn (chia sẻ file P2P cũng vậy).

HTTP GET có điều kiện (Conditional HTTP)

- Mục tiêu: Không gửi đối tượng nếu dữ liệu trong web caches đã được cập nhật.
 - o Giảm độ trễ.
 - o Nhu cầu sử dụng đường link ít hơn.
- Cache: Xác định mốc thời gian lần cuối dữ liệu lưu trong cache lần cuối được cập nhật trong HTTP request. (**If-modified-since**)
- Server: Không phải hồi lại dữ liệu được yêu cầu nếu như dữ liệu đó không có gì thay đổi kể từ mốc thời gian trong cache nói trên. (**HTTP 304 Not Modified**).

FTP (File Transfer Protocol) – Giao thức truyền file



FTP: Giao thức truyền file đến/từ host từ xa.

Mô hình Client/Server:

- Client: Phía khởi tạo truyền (đến/từ host từ xa).
- Server: host ở xa.

Kết nối điều khiển: TCP ở **port 21**, duy trì trong suốt phiên làm việc.

Kết nối dữ liệu: TCP ở **port 20**, kết nối tự động ngắt khi truyền dữ liệu hoàn tất.

Email (Thư điện tử)

Gồm 3 thành phần chính:

- **User Agent:**



- Còn gọi là “Mail Reader”.
- Soạn thảo, sửa đổi, đọc các thông điệp email.
- VD: Outlook, Thunderbird, iPhone Mail Client,...
- **Mail Server:**
 - Hộp thư (Mailbox) chứa thông điệp đến User.
 - Hàng thông điệp (Message Queue) của các thông điệp mail ra ngoài (chuẩn bị gửi).
- **SMTP (Simple Mail Transfer Protocol):** gửi thư giữa các mail server và giữa user agents và mail server.
 - **RFC 2821.**
 - **TCP, port 25:** chuyển thư từ client đến server và các server với nhau.
 - Gồm **3 giai đoạn truyền:**
 - Thiết lập kết nối.
 - Truyền thông điệp.
 - Đóng kết nối.
 - Tương tác lệnh/phản hồi:
 - Lệnh: ASCII
 - Phản hồi: mã và cụm từ trạng thái
 - Thông điệp phải ở dạng mã ASCII 7 bit.
 - Dùng kết nối không bền vững.



- POP3 và IMAP:

	POP3	IMAP
Độ phức tạp	Đơn giản hơn	Nâng cao hơn
Kết nối	Kết nối trên port 110 và máy chủ POP có bảo mật SSL kết nối trên port 995	Kết nối trên port 143 và máy chủ IMAP có bảo mật SSL kết nối trên port 993
Truy cập	Mỗi lần chỉ có thể truy cập từ một thiết bị duy nhất	Tin nhắn có thể được truy cập trên nhiều thiết bị
Đọc thư	Phải được tải xuống trên hệ thống cục bộ.	Nội dung thư có thể được đọc một phần trước khi tải xuống
Chỉnh sửa	Người dùng phải tải về mới có thể tạo, xóa hoặc đổi tên email trên mail.	Người dùng có thể tạo, xóa hoặc đổi tên email trên mail server.
Chế độ lưu trữ	Có thể tùy chọn giữ lại hoặc xóa thư trên hộp thư sau khi tải về.	Mặc định lưu trữ bản sao trên mail server sau khi tải về.
Tải xuống	Tất cả tin nhắn được tải xuống cùng một lúc	Tiêu đề tin nhắn có thể được xem trước khi tải xuống.
Phạm vi hoạt động	Có thể hoạt động không cần kết nối Internet.	Cần kết nối internet để hoạt động
Tốc độ	Nhanh hơn	Chậm hơn

Các ứng dụng P2P

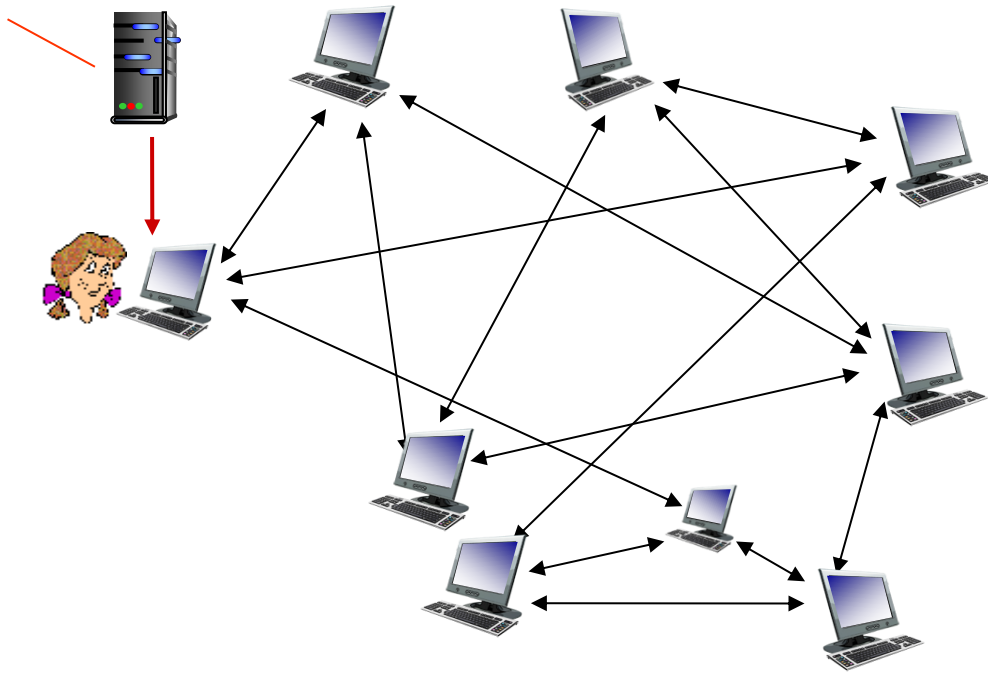
BitTorrent

Các khái niệm:

- **BitTorrent** là một giao thức P2P phổ biến trong việc phân tán tập tin.
- **Torrent** là tập hợp tất cả các Peers tham gia vào việc phân tán một tập tin cụ thể.

- Tracker là một máy tính trung tâm đóng vai trò quản lí giúp 1 Peer kết nối với nhiều “Peer láng giềng” khác để trao đổi dữ liệu.
- Tập tin cần phân tán sẽ được tách thành nhiều đoạn tập tin có kích thước bằng nhau, thông thường là 256KB.
- Các Peers vừa lấy, vừa gửi các đoạn tập tin tới Peers khác.

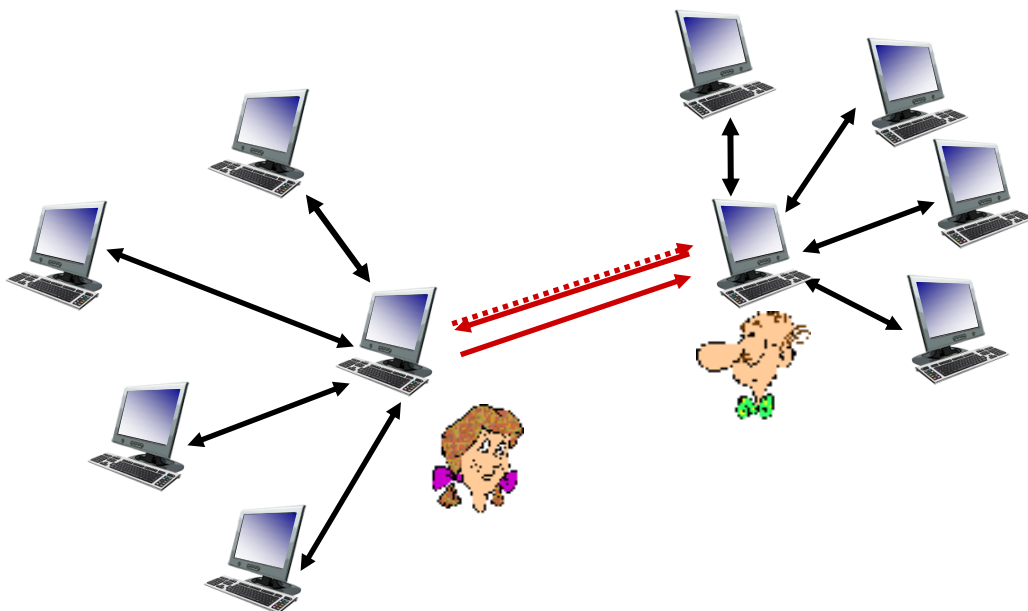
Nguyên lí hoạt động:



- Giả sử, ta có Alice đang có nhu cầu tải 1 tập tin bất kì có dung lượng khá lớn.
- Lúc này, Alice được xem như là 1 peer mới tham gia vào torrent và chưa có đoạn tập tin nào.
- Máy của Alice sẽ đăng ký với Tracker và Tracker sẽ ngẫu nhiên chọn 1 tập hợp các máy ngang hàng đang tham gia vào torrent trước đó và gửi địa chỉ IP của các Peers này đến cho Alice.
- Alice thiết lập kết nối TCP song song tới các máy tính này, gọi chúng là các “Peers láng giềng”. Ở hình minh họa trên thì Alice kết nối với 3 Peers khác.



- Mỗi chu kỳ thời gian, Alice sẽ dựa vào danh sách đoạn tập tin hiện tại của mình mà yêu cầu tới các Peers láng giềng.
- Trong quá trình yêu cầu, Alice sẽ ưu tiên chọn các đoạn tập tin hiếm nhất, có nghĩa là các đoạn tập tin có bản sao lặp lại ít nhất trong số các láng giềng, và tiến hành yêu cầu các đoạn tập tin này trước.
- Về việc gửi khối dữ liệu, cứ mỗi 10 giây, Alice sẽ đánh giá các Peers láng giềng theo tỉ lệ người gửi cho cô với tốc độ nhanh nhất theo top 4. Và 4 máy ngang hàng sẽ được gọi là không nghẽn dữ liệu (unchoked).
- Và mỗi 30 giây, Alice cũng chọn thêm 1 máy ngang hàng mới và bắt đầu giao dịch. Giả sử máy được chọn là Bob. Nếu tỉ lệ tốc độ Bob gửi đến Alice là đủ cao thì kể đó Bob có thể trở thành 1 trong 4 máy tải dữ liệu của Alice. Và Alice cũng có thể trở thành 1 trong 4 máy tải dữ liệu của Bob.



- Và cứ thế Alice và 4 máy tải dữ liệu ngang hàng tiếp tục giao dịch với nhau cho đến khi một bên tìm thấy một đối tác tốt hơn.



- Tất cả các máy ngang hàng khác ngoài 5 máy này (4 máy tải dữ liệu và 1 máy thăm dò) được gọi là tắc nghẽn (choked), nghĩa là chúng sẽ không nhận bất kì tập tin nào từ Alice.

Lập trình Socket

- Mục tiêu: Tìm hiểu cách xây dựng các ứng dụng client/server cái mà truyền thông dùng sockets.
- Socket: là một cánh cổng giữa tiến trình ứng dụng và giao thức transport end-to-end.
- Có 2 loại socket dùng cho 2 dịch vụ transport:
 - o UDP: chuyển giao datagram không tin cậy.
 - o TCP: chuyển giao datagram tin cậy, bytes được định hướng dòng (stream-oriented)

Lập trình socket với UDP

- Không bắt tay trước khi gửi dữ liệu.
- Bên gửi chỉ rõ địa chỉ IP và Port cho mỗi Packet.
- Bên nhận lấy địa chỉ IP và Port của người gửi trong Packet nhận được.
- Dữ liệu truyền có thể bị mất hoặc không đúng thứ tự.

Sự tương tác Socket Client/Server: UDP

Server (máy hostid)



Client





Lập trình socket với TCP

- Tiến trình client/server phải được chạy trước.
- Server phải tạo socket để đợi client đến liên lạc.
- Client tạo socket TCP, xác định địa chỉ IP, số Port của tiến trình server.
- Thiết lập kết nối client TCP đến server TCP khi client tạo socket.
- Khi server đã tiếp xúc với client thì server TCP tạo socket mới cho tiến trình server để truyền thông với client đó.
- Cho phép server nói chuyện được với nhiều client phân biệt bằng số Port.

Sự tương tác Socket Client/Server: TCP

10/11/2023

