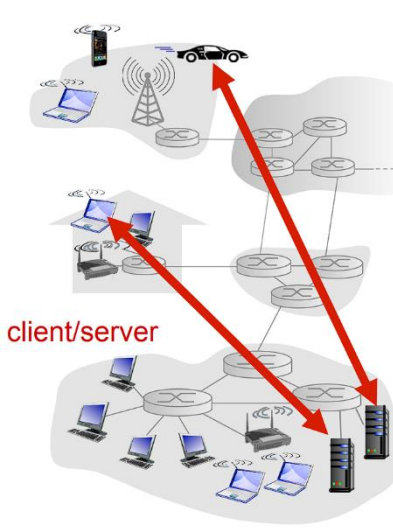
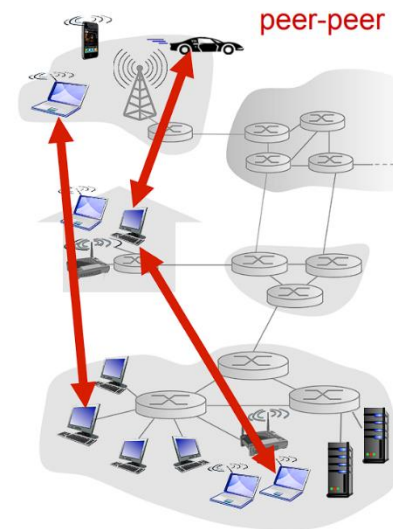




## **NHẬP MÔN MẠNG MÁY TÍNH**

### **Chương 2: Tầng Application**

## Các kiến trúc ứng dụng

|                 | Client – Server   | P2P (Peer to peer)  |
|-----------------|---|---|
| <b>Minh Họa</b> |  |                             |
| <b>Server</b>   | Luôn hoạt động  | Không có Server   |
|                 | Địa chỉ IP cố định.   |   |
|                 | Trung tâm phục vụ và lưu trữ dữ liệu.   |   |
| <b>Client</b>   | Giao tiếp với Server.   | Các hệ thống đầu cuối giao tiếp trực tiếp với nhau.   |
|                 | Có thể kết nối không liên tục.  | Các peer được kết nối không liên tục và có thể thay đổi địa chỉ IP.   |
|                 | Có thể dùng địa chỉ IP động.  | Các peer yêu cầu dịch vụ từ các peer khác và cung cấp dịch vụ ngược lại cho các peer khác ⇒ Quản Lý phức tạp. |
|                 | Không giao tiếp trực tiếp với các Client khác.                                    |   |

## 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 dịch vụ giao thức Transport Internet

### Dịch vụ TCP:

Reliable transport (truyền tải tin cậy) giữa tiến trình gửi và nhận.

Flow control (điều khiển luồng): người gửi sẽ không áp đảo người nhận.

Congestion control (điều khiển tắc nghẽn): điều tiết người gửi khi mạng quá tải.

Connection-oriented (hướng kết nối): thiết lập được yêu cầu giữa tiến trình client và server.

### Dịch vụ UDP:

Truyền tải không tin cậy giữa tiến trình gửi và nhận.

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:

- Là giao thức mạng TCP/IP
- Có chức năng kết nối các máy tính lại với nhau.
- Có thể gửi dữ liệu cho nhau,...

Khác:

| TCP  | UDP  |
|--|--|
| <ul style="list-style-type: none"><li>- Thường dùng cho mạng WAN.</li><li>- Không cho phép mất gói tin.</li><li>- Đảm bảo việc truyền dữ liệu.</li><li>- Tốc độ truyền thấp hơn UDP.</li></ul> | <ul style="list-style-type: none"><li>- Thường dùng cho mạng LAN.</li><li>- Cho phép mất dữ liệu.</li><li>- Khung đảm bảo.</li></ul> |

## 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

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

### **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 ⇒ Nhà cung cấp dịch vụ có thể cung cấp nội dung hiệu quả hơn (chia sẻ file P2P cũng vậy).