

BỘ THÔNG TIN VÀ TRUYỀN THÔNG  
HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

**BÀI GIẢNG  
INTERNET VÀ GIAO THỨC**

**Chủ trì: TS. Hoàng Trọng Minh**

**Cộng tác viên:**

**ThS. Nguyễn Thanh Trà**

**TS. Phạm Anh Thư**

**TS. Nguyễn Chiến Trinh**

**Hà Nội, 2021**

## **MỤC LỤC**

MỤC LỤC .....	2
DANH MỤC CÁC TỪ VIẾT TẮT .....	6
DANH MỤC HÌNH VẼ .....	6
MỞ ĐẦU .....	11
CHƯƠNG 1. TỔNG QUAN VỀ INTERNET .....	12
1.1 Giới thiệu chung.....	12
1.2 Kiến trúc mạng internet .....	14
1.3 Các thành phần mạng.....	16
1.4 Các giải pháp công nghệ mạng .....	20
1.5 Kết luận chương .....	27
CHƯƠNG 2. CÁC ỨNG DỤNG VÀ GIAO THỨC MẠNG .....	28
2.1 Tổng quan về các ứng dụng và dịch vụ hạ tầng .....	28
2.2 Ứng dụng WEB và các giao thức .....	33
2.2.1 Tổng quan về ứng dụng WEB và giao thức HTTP .....	33
2.2.2 Hoạt động của giao thức HTTP .....	37
2.2.3 Các bản tin HTTP .....	40
2.2.4 Giao thức HTTPS .....	46
2.2.5 Cookie và lưu đệm WEB .....	47
2.3 Ứng dụng truyền tệp và các giao thức .....	56
2.3.1 Dịch vụ truyền tệp .....	56

2.3.2 Giao thức truyền tệp FTP .....	57
2.3.2 Giao thức FTPS .....	60
2.4 Ứng dụng thư điện tử và các giao thức.....	61
2.4.1 Thư điện tử trên Internet.....	61
2.4.2 Giao thức truyền thư điện tử đơn giản SMTP .....	63
2.4.3 Khuôn dạng bản tin thư điện tử .....	68
2.4.4 Các giao thức truy cập thư điện tử .....	69
2.5 Ứng dụng hệ thống tên miền .....	76
2.5.1 Tổng quan DNS .....	76
2.5.2 Hoạt động của DNS .....	80
2.5.3 Bản ghi và bản tin DNS .....	86
2.5.4 Vấn đề an toàn trong DNS.....	91
2.6 Các ứng dụng mạng ngang hàng .....	93
2.6.1 Phân bố tệp P2P .....	93
2.6.2 Bảng hàm băm phân tán DHT .....	101
2.6.3 Ứng dụng thoại Internet.....	109
2.7 Kết luận chương .....	111
<b>CHƯƠNG 3: CÁC GIAO THỨC ỨNG DỤNG ĐA PHƯƠNG TIỆN .....</b>	<b>112</b>
3.1 Giới thiệu chung .....	112
3.2 Các giao thức hỗ trợ thời gian thực luồng đa phương tiện.....	113
3.2.1 Yêu cầu đối với giao thức truyền tải luồng đa phương tiện .....	113
3.2.2 Giao thức truyền tải thời gian thực RTP .....	116
3.2.3 Giao thức điều khiển truyền tải thời gian thực (RTCP) .....	119

3.3 Giao thức báo hiệu và điều khiển phiên đa phương tiện .....	122
3.3.1 Giao thức phát tin trực tuyến thời gian thực RTSP.....	122
3.2.2 Giao thức mô tả phiên SDP .....	126
3.3.3 Giao thức khởi tạo phiên SIP .....	128
3.4 Hoạt động điều hành của SIP .....	142
3.4.1 Hỗ trợ tính di động (SIP Mobility) .....	142
3.4.2 Hoạt động của SIP trong IMS .....	148
3.5 Kết luận chương .....	156
<b>CHƯƠNG 4: GIAO THỨC ỨNG DỤNG INTERNET VẠN VẬT.....</b>	<b>157</b>
4.1 Giới thiệu chung về IoT.....	157
4.1.1 Khái niệm về IoT .....	157
4.1.2 Đặc điểm của IoT .....	158
4.1.3 Tiềm năng và thách thức của hệ thống IoT .....	159
4.2 Kiến trúc và các ứng dụng IoT .....	160
4.2.1 Kiến trúc hệ thống IoT .....	160
4.2.2 Các ứng dụng IoT .....	164
4.3 Các giải pháp công nghệ mạng IoT .....	168
4.3.1 RFID .....	168
4.3.2 Cloud Computing .....	172
4.3.3 ZigBee .....	176
4.4. Các giao thức Request/ Response.....	180
4.4.1 Giao thức HTTP trong IoT .....	180
4.4.2 Giao thức CoAP.....	182

4.5 Các giao thức Publish/Subscriber .....	187
4.5.1 Giao thức MQTT .....	187
4.5.2 Giao thức MQTT-SN .....	220
4.6 Kết luận chương .....	236
KẾT LUẬN .....	237
TÀI LIỆU THAM KHẢO .....	238

# **DANH MỤC CÁC TỪ VIẾT TẮT**

## **DANH MỤC HÌNH VẼ**

<i>Hình 1.1: Kiến trúc internet .....</i>	14
<i>Hình 1.2: Cấu trúc Internet .....</i>	16
<i>Hình 1.3: Internet dưới góc nhìn của người sử dụng .....</i>	20
<i>Hình 1.4: Kết nối tới Internet qua dial-up.....</i>	21
<i>Hình 1.5: Kết nối tới Internet qua ADSL.....</i>	22
<i>Hình 1.6: Kết nối tới Internet qua cable .....</i>	23
<i>Hình 1.7: Kết nối qua vệ tinh .....</i>	24
<i>Hình 1.8: Kết nối không dây.....</i>	25
<i>Hình 1.9: Kết nối Leased - line .....</i>	26
<i>Hình 2.1: Truyền thông giữa các hệ thống đầu cuối ở lớp ứng dụng .....</i>	28
<i>Hình 2.2: Kiến trúc khách/chủ (a) và ngang hàng (b) .....</i>	30
<i>Hình 2.3: Hoạt động yêu cầu-đáp ứng của HTTP .....</i>	35
<i>Hình 2.4: Tính toán thời gian cần thiết để yêu cầu và nhận tệp HTML .....</i>	39
<i>Hình 2.5: Khuôn dạng chung của một bản tin yêu cầu HTTP .....</i>	42
<i>Hình 2.6: Khuôn dạng chung của một bản tin đáp ứng HTTP .....</i>	45
<i>Hình 2.7: Giữ trạng thái người sử dụng với cookie .....</i>	48
<i>Hình 2.8: Máy khách yêu cầu đổi tượng qua máy chủ đệm Web .....</i>	51
<i>Hình 2.9: Nghẽn cổ chai giữa mạng của học viện và Internet.....</i>	53
<i>Hình 2.10: Thêm máy chủ đệm vào mạng của học viện .....</i>	54
<i>Hình 2.11: FTP chuyển tệp giữa các hệ thống tệp cục bộ và ở xa .....</i>	57

<i>Hình 2.12: Kết nối điều khiển và kết nối dữ liệu.....</i>	58
<i>Hình 2.13: Tổng quan về hệ thống thư điện tử Internet.....</i>	62
<i>Hình 2.14: Quá trình hai người gửi thư cho nhau .....</i>	64
<i>Hình 2.15: Các giao thức thư điện tử và những thực thể truyền thông của chúng.....</i>	71
<i>Hình 2.16: Cấu trúc phân cấp của máy chủ DNS .....</i>	81
<i>Hình 2.17: Các máy chủ DNS gốc năm 2009 (tên, tổ chức, vị trí) .....</i>	82
<i>Hình 2.18: Tương tác giữa các máy chủ DNS khác nhau .....</i>	84
<i>Hình 2.19: Truy vấn để quy trong DNS.....</i>	85
<i>Hình 2.20: Khuôn dạng bản tin DNS .....</i>	89
<i>Hình 2.21: Minh họa phân bố tệp. ....</i>	94
<i>Hình 2.22: Thời gian phân bố đối với kiến trúc P2P và client-server. ....</i>	97
<i>Hình 2.23: Phân bổ tệp với BitTorrent.....</i>	99
<i>Hình 2.24: (a) DHT vòng, thiết bị ngang hàng 3 muốn xác định ai chịu trách nhiệm đối với khóa 11. (b) DHT vòng với các đường nối tắt.....</i>	105
<i>Hình 3.1: Phân loại ứng dụng thời gian thực .....</i>	114
<i>Hình 3.2: Header của RTP Packet .....</i>	117
<i>Hình 3.3: Kiến trúc của RTSP .....</i>	122
<i>Hình 3.4: Thủ tục RSTP tương tác giữa máy khách và máy chủ .....</i>	124
<i>Hình 3.5: Các thực thể chức năng của mạng SIP .....</i>	130
<i>Hình 3.6: Cấu trúc bản tin SIP.....</i>	134
<i>Hình 3.7: Ví dụ bản tin yêu cầu (INVITE).....</i>	137
<i>Hình 3.8: Ví dụ bản tin đáp ứng .....</i>	138
<i>Hình 3.9: Thủ tục báo hiệu trong SIP .....</i>	139

<i>Hình 3.10: Hỗ trợ tính di động của thiết bị đầu cuối sử dụng SIP REGISTER .....</i>	143
<i>Hình 3.11: Hỗ trợ tính di động đang trong cuộc gọi bằng cách gửi lại bản tin INVITE</i>	145
<i>Hình 3.12: Hỗ trợ tính di động giữa cuộc gọi sử dụng bản tin INVITE với trường Replaces .....</i>	147
<i>Hình 3.13: Kiến trúc phân lớp của phân hệ IMS .....</i>	149
<i>Hình 3.14: Luồng bản tin báo hiệu đăng ký .....</i>	154
<i>Hình 3.15: Luồng bản tin báo hiệu thiết lập phiên .....</i>	155
<i>Hình 3.16: Luồng bản tin người dùng A lấy thông tin hiện diện người dùng B .....</i>	156
<i>Hình 4. 1 Mô hình kiến trúc hệ thống IoT .....</i>	161
<i>Hình 4. 2: Ví dụ ứng dụng hệ thống RFID .....</i>	169
<i>Hình 4. 3: Điện toán đám mây .....</i>	172
<i>Hình 4. 1: Cấu trúc mạng Zigbee .....</i>	178
<i>Hình 4. 5: Các lớp của CoAP .....</i>	183
<i>Hình 4. 6: Truyền bản tin đáng tin cậy .....</i>	183
<i>Hình 4. 7: Truyền bản tin không đáng tin cậy .....</i>	184
<i>Hình 4. 8: Hai yêu cầu GET có phản hồi ngay lập tức, một thành công, một không tìm thấy .....</i>	184
<i>Hình 4. 9: Yêu cầu GET với phản hồi trì hoãn .....</i>	185
<i>Hình 4. 10: Sơ đồ hệ thống điều khiển năng lượng .....</i>	187
<i>Hình 4. 11: Giao thức MQTT .....</i>	188
<i>Hình 4. 12: Kiến trúc của giao thức MQTT .....</i>	189
<i>Hình 4. 13: Mô hình Xuất bản/Theo dõi .....</i>	190
<i>Hình 4. 14: Publisher và Subscriber .....</i>	190

<i>Hình 4. 15: Các loại đăng ký trong giao thức MQTT</i> .....	191
<i>Hình 4. 16: Định dạng phân cấp của chủ đề</i> .....	192
<i>Hình 4. 17: Cú pháp Wildcard đơn cấp</i> .....	192
<i>Hình 4. 18: Các chủ đề được khai báo theo cú pháp trên sử dụng Wildcard đơn cấp</i> ...	192
<i>Hình 4. 19: Cú pháp Wildcard đa cấp</i> .....	193
<i>Hình 4. 20: Các chủ đề được khai báo theo cú pháp trên sử dụng Wildcard đa cấp.</i> ....	193
<i>Hình 4. 21: Định dạng của gói tin điều khiển</i> .....	194
<i>Hình 4. 22: Cách tính giá trị trường Độ dài còn lại</i> .....	196
<i>Hình 4. 23: Gói tin CONNECT</i> .....	198
<i>Hình 4. 24: Gói tin CONNACK</i> .....	202
<i>Hình 4. 25: Gói tin PUBLISH</i> .....	204
<i>Hình 4. 26: Gói tin PUBACK</i> .....	206
<i>Hình 4. 27: Gói tin PUBREC</i> .....	207
<i>Hình 4. 28: Gói tin PUBREL</i> .....	207
<i>Hình 4. 29: Gói tin PUBCOMP</i> .....	207
<i>Hình 4. 30: Gói tin SUBSCRIBE</i> .....	208
<i>Hình 4. 31: Gói tin SUBACK</i> .....	210
<i>Hình 4. 32: Gói tin UNSUBSCRIBE</i> .....	211
<i>Hình 4. 33: Gói tin UNSUBACK</i> .....	212
<i>Hình 4. 34: Gói tin DISCONNECT, PINGREQ, PINGRESP</i> .....	213
<i>Hình 4. 35: Quy trình thiết lập phiên MQTT và đăng ký theo dõi với cờ cleanSession = 1.</i> .....	214

<i>Hình 4. 36: Quy trình thiết lập phiên MQTT và đăng ký theo dõi với cờ cleanSession = 0.</i>	215
<i>Hình 4. 37: Quy trình xuất bản bản tin với QoS = 0.....</i>	216
<i>Hình 4. 38: Quy trình xuất bản bản tin với QoS mức 1. ....</i>	217
<i>Hình 4. 39: Quy trình xuất bản bản tin với QoS mức 2. ....</i>	218
<i>Hình 4. 40: Kiến trúc giao thức MQTT-SN. ....</i>	223
<i>Hình 4. 41: Transparent Gateway và Aggregating Gateway. ....</i>	223
<i>Hình 4. 42: Định dạng gói tin trong MQTT-SN. ....</i>	224
<i>Hình 4. 43: Tiến trình Thiết lập kết nối từ phía Client.....</i>	228
<i>Hình 4. 44: Sơ đồ chuyển trạng thái của Client. ....</i>	234
<i>Hình 4. 45: Tiến trình chuyển sang trạng thái ngủ của Client. ....</i>	235

## MỎ ĐẦU

# CHƯƠNG 1. TỔNG QUAN VỀ INTERNET

## 1.1 Giới thiệu chung

Internet là một tập hợp các mạng được kết nối theo giao thức chung. Nền tảng của Internet được hình thành bởi sự kết nối toàn cầu của hàng trăm nghìn máy tính, thực thể truyền thông và hệ thống thông tin độc lập. Khả năng kết nối được thực hiện qua việc sử dụng một tập hợp các tiêu chuẩn, thủ tục và định dạng giao tiếp chung giữa các mạng và các thiết bị cũng như phương tiện tính toán khác nhau. Các thủ tục cho phép các máy tính giao tiếp với nhau được gọi là "giao thức". Hạ tầng và các giao thức của Internet ngày nay đã được phát triển đa dạng để đáp ứng môi trường kết nối cho thực thể vật lý hay ảo như Internet của vạn vật IoT/IoE (Internet of Things/ Internet of everything). Tuy nhiên, các giao thức ban đầu được Internet sử dụng dựa trên các giao thức của giao thức "TCP/IP" là tên của hai giao thức đã hình thành cơ sở chính cho hoạt động của Internet.

Vào những năm 1970, một cuộc cách mạng về mạng máy tính đã ra đời khái niệm về Internet xuất phát từ mong muốn có một công nghệ chuyển mạch gói duy nhất để xử lý mọi yêu cầu truyền thông. Tuy nhiên, không thể có công nghệ chuyển mạch gói đơn lẻ nào đáp ứng được mọi nhu cầu và mục tiêu chuyển thành việc kết nối nhiều công nghệ chuyển mạch gói thành một hoạt động tổng thể. Từ đó, bộ tiêu chuẩn cho một kết nối liên thông ra đời và gọi là bộ giao thức internet TCP/IP. Bộ giao thức này đã cung cấp nền tảng giao tiếp cho Internet toàn cầu ngày nay. Một trong những lý do chính cho sự thành công của các tiêu chuẩn TCP/IP nằm ở khả năng đáp ứng được tính không đồng nhất của các giải pháp công nghệ khác nhau. Thay vì cố gắng tích hợp với các công nghệ lớp thấp, TCP/IP thực hiện tiếp cận ảo hóa cho các gói tin độc lập với hạ tầng mạng hay nhận dạng liên kết và thực hiện các ánh xạ để tương thích.

Internet và các giao thức của nó đã là một động lực chính cho sự phát triển liên tục của các công nghệ chuyển mạch gói. Khi Internet phát triển, máy tính trở nên mạnh mẽ hơn và các ứng dụng gửi nhiều dữ liệu hơn, đặc biệt là ảnh và video. Để đáp ứng nhu cầu sử dụng ngày càng tăng, các công nghệ chuyển mạch gói được phát triển để truyền nhiều dữ

liệu hơn và xử lý nhiều gói hơn trong một thời gian nhất định. Các công nghệ mới được kết hợp vào Internet với các công nghệ hiện có một cách đơn giản nhờ việc Internet chấp nhận sự không đồng nhất và các công nghệ mạng mới mà không làm gián đoạn hoạt động của các mạng hiện có.

Mặc dù Internet có thể coi như là một hệ thống lớn, nhưng bao gồm các bộ phận được sở hữu và điều hành bởi các cá nhân hoặc tổ chức. Để giúp làm rõ quyền sở hữu và mục đích, ngành công nghiệp mạng sử dụng các thuật ngữ mạng công cộng và mạng riêng.

Mạng công cộng được vận hành giống như dịch vụ có sẵn cho thuê bao, bất kỳ cá nhân hoặc công ty nào trả phí thuê bao đều có thể sử dụng mạng. Một công ty cung cấp dịch vụ truyền thông được gọi là nhà cung cấp dịch vụ. Nhà cung cấp dịch vụ Internet là một phần của các nhà cung cấp dịch vụ truyền thông. Thuật ngữ “công cộng” đề cập đến tính khả dụng chung của dịch vụ chứ không phải dữ liệu được truyền.

Một mạng riêng được kiểm soát bởi một nhóm cụ thể. Sự phân biệt giữa phần công cộng và riêng tư của Internet có thể không rõ ràng vì quyền kiểm soát không luôn bao hàm quyền sở hữu. Một công ty có thể thuê đường truyền dữ liệu công cộng nhưng áp đặt các chính sách để nó thành mạng riêng của họ.

Truyền thông luôn liên quan đến ít nhất hai thực thể, một thực thể gửi thông tin và một thực thể nhận thông tin đó. Trên thực tế, phần lớn các hệ thống truyền thông đều chứa các thực thể trung gian do giới hạn của các phương tiện truyền dẫn. Để truyền thông thành công, tất cả các thực thể trong mạng phải thống nhất về cách thông tin sẽ được biểu diễn và truyền đạt. Thỏa thuận giao tiếp liên quan đến thủ tục được sử dụng để bắt đầu, thực hiện giao tiếp, và định dạng của bản tin. Từ đó, chúng ta sử dụng thuật ngữ giao thức truyền thông, giao thức mạng hoặc giao thức để chỉ một đặc điểm kỹ thuật cho truyền thông qua mạng. Một trong các khía cạnh quan trọng nhất của giao thức là liên quan tới các tình huống không mong muốn hoặc lỗi trong các thủ tục để đưa ra các phương án xử lý.

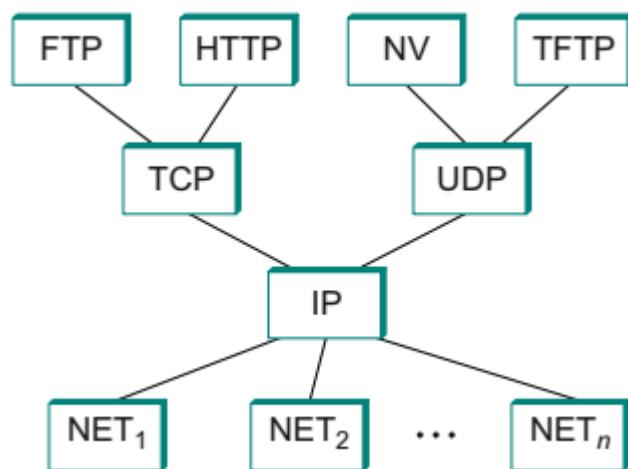
Internet hỗ trợ hai mô hình giao tiếp cơ bản: mô hình luồng và mô hình bản tin.

Truyền tải theo luồng trên Internet. Thuật ngữ luồng biểu thị một mô hình gồm một chuỗi các byte đi từ chương trình ứng dụng này sang chương trình ứng dụng khác. Cơ chế truyền tải luồng thực hiện chuyển một chuỗi các byte theo yêu cầu của ứng dụng. Dịch vụ luồng internet là hướng kết nối và yêu cầu các giai đoạn thiết lập, truyền và giải phóng kết nối.

Cơ chế truyền tải bản tin cho phép bên gửi và bên nhận giao tiếp thông qua một bản tin cho một phiên kết nối và không phân chia hoặc ghép hợp các bản tin. Mô hình này khác với mô hình truyền tải trên khi có nhiều sự lựa chọn giao tiếp như: 1:1, 1: N và N:1.

## 1.2 Kiến trúc mạng internet

Kiến trúc Internet, đôi khi còn được gọi là kiến trúc TCP/IP như mô tả đơn giản như trong Hình 1.1 dưới đây. Kiến trúc Internet đã phát triển dựa trên nền tảng của mạng ARPANET. Cá Internet và ARPANET đều được tài trợ bởi Cơ quan Dự án Nghiên cứu Nâng cao (ARPA), một trong những cơ quan tài trợ nghiên cứu và phát triển của Bộ Quốc phòng Mỹ. Internet và ARPANET đã ra đời trước khi có kiến trúc OSI, và kinh nghiệm thu được từ việc xây dựng chúng có ảnh hưởng lớn đến mô hình tham chiếu OSI.



Hình 1.1: Kiến trúc internet

Mặc dù mô hình rất phổ biến cho các công nghệ mạng tham chiếu, nhưng mô hình 4 lớp thường được sử dụng để mô tả kiến trúc Internet. Ở mức thấp nhất là nhiều loại

giao thức mạng, được ký hiệu là NET1, NET2, v.v. Trên thực tế, các giao thức này được triển khai bằng sự kết hợp giữa phần cứng và phần mềm liên quan tới thiết bị mạng.

Lớp thứ hai bao gồm một giao thức duy nhất - Giao thức Internet (IP). Đây là giao thức hỗ trợ kết nối nhiều công nghệ mạng thành một mạng liên kết logic duy nhất.

Lớp thứ ba chứa hai giao thức chính — Giao thức điều khiển truyền tải (TCP) và Giao thức dữ liệu Người dùng (UDP). TCP và UDP cung cấp các kênh logic thay thế cho các chương trình ứng dụng: TCP cung cấp kênh luồng byte đáng tin cậy và UDP cung cấp kênh phân phối datagram không đáng tin cậy (datagram có thể được coi tương đương với từ bản tin). Trong ngôn ngữ của Internet thì hai giao thức TCP và UDP đôi khi được gọi là giao thức end-to-end thay vì gọi là giao thức lớp truyền tải.

Phía trên lớp truyền tải là một loạt các giao thức ứng dụng, chẳng hạn như HTTP, FTP, Telnet (đăng nhập từ xa) và giao thức chuyển thư đơn giản (SMTP) cho phép tương tác giữa các ứng dụng phổ biến. Ta lưu ý sự khác biệt giữa giao thức lớp ứng dụng và một ứng dụng ở góc độ chức năng vì đôi khi tên gọi tương tự nhau.

Kiến trúc Internet có ba đặc điểm nổi bật như sau:

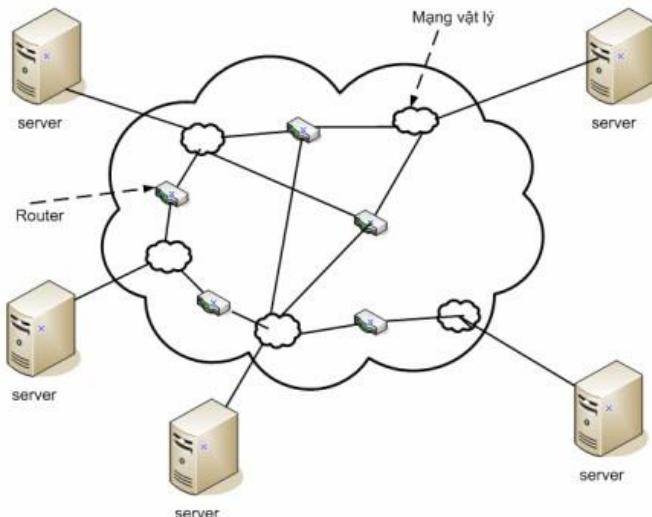
Kiến trúc Internet không hướng tới mục tiêu phân lớp nghiêm ngặt. Các ứng dụng có thể tùy ý bỏ qua các lớp truyền tải để đến trực tiếp lớp IP hoặc một trong các mạng phía dưới. Như vậy, các lập trình viên có thể tự do xác định các ứng dụng hoặc kênh dữ liệu cho bất kỳ giao thức hiện có nào.

Thứ hai, việc tập trung vào lớp IP đơn nhất đã phản ánh triết lý trung tâm của kiến trúc. Nghĩa là, IP đóng vai trò là đầu mối của kiến trúc, xác định một phương pháp chung để trao đổi các gói tin trong tập hợp các mạng kết nối. Phía trên IP có thể có nhiều giao thức truyền tải để cung cấp kênh trùm tượng khác nhau cho các chương trình ứng dụng. Do đó, vấn đề gửi bản tin từ máy chủ đến máy chủ lưu trữ hoàn toàn tách biệt với vấn đề gửi bản tin từ máy chủ tới máy khách. Phía dưới lớp IP, kiến trúc cho phép tùy ý nhiều công nghệ mạng khác nhau nhằm mở rộng và phát huy năng lực kết nối.

Cuối cùng, kiến trúc Internet yêu cầu một giao thức mới chính thức được đưa vào kiến trúc phải có cả đặc tả giao thức và ít nhất một (và tốt nhất là hai) đại diện triển khai các thuộc tính kỹ thuật. Đặc điểm này đảm bảo rằng các giao thức của kiến trúc có thể được triển khai thực tế một cách hiệu quả.

### 1.3 Các thành phần mạng

Internet là một liên mạng kết nối các mạng nhỏ hơn với nhau. Cấu trúc Internet gồm các mạng máy tính được kết nối với nhau thông qua các kết nối viễn thông. Thiết bị dùng để kết nối các mạng máy tính với nhau là cổng nối Internet (Internet Gateway) hoặc bộ định tuyến (Router).



Hình 1.2: Cấu trúc Internet

Tuy nhiên, đối với người dùng, Internet chỉ là một mạng duy nhất thông qua các công nghệ kết nối và chuyển mạch, người dùng sử dụng các dịch vụ một cách trong suốt đối với cấu trúc mạng phía dưới như trên hình 1.3.

Các thiết bị mạng Internet cũng là các thiết bị trên mạng máy tính gồm:

#### **Hub**

Hub kết nối nhiều thiết bị mạng máy tính với nhau. Hub hoạt động như một bộ lặp tín hiệu để khuếch đại các tín hiệu bị suy giảm trong khi truyền và kết nối với nhau. Hub là

thiết bị đơn giản nhất trong họ thiết bị kết nối mang vì nó kết nối các thành phần mạng với cách thức giống hệt nhau. Một Hub có thể được sử dụng với cả dữ liệu kỹ thuật số và dữ liệu tương tự, miễn là cài đặt của nó đã được định cấu hình để chuẩn bị cho việc định dạng dữ liệu đến. Ví dụ, nếu dữ liệu đến ở định dạng kỹ thuật số, hub phải chuyển nó dưới dạng gói; tuy nhiên, nếu dữ liệu đến là tương tự, thì trung tâm chuyển nó ở dạng tín hiệu.

Các Hub không thực hiện chức năng lọc hoặc định địa chỉ gói tin; Hub chỉ gửi các gói dữ liệu đến tất cả các thiết bị được kết nối. Các hub hoạt động ở lớp Vật lý của mô hình Kết nối Hệ thống Mở (OSI).

### **Chuyển mạch**

Các bộ chuyển mạch nói chung có vai trò thông minh hơn các hub. Switch là một thiết bị nhiều cổng kết nối để tăng hiệu quả mạng trong mục tiêu kết nối. Chuyển mạch duy trì thông tin định tuyến chỉ cho các nút trong mạng nội bộ. Các dải mạng LAN thường được kết nối bằng cách sử dụng thiết bị chuyển mạch. Nói chung, bộ chuyển mạch có thể đọc địa chỉ phần cứng của các gói đến để truyền chúng đến đích thích hợp.

Sử dụng thiết bị chuyển mạch cải thiện hiệu quả mạng khi kết hợp với bộ định tuyến để tạo ra khả năng chuyển mạch ảo. Chuyển mạch có thể hoạt động ở lớp Liên kết dữ liệu hoặc lớp Mạng của mô hình OSI. Một bộ chuyển mạch nhiều lớp là một bộ chuyển mạch có thể hoạt động ở cả hai lớp, có nghĩa là nó có thể hoạt động như một bộ chuyển mạch và một bộ định tuyến. Bộ chuyển mạch đa lớp là một thiết bị hiệu suất cao hỗ trợ các giao thức định tuyến giống như bộ định tuyến.

### **Bộ định tuyến**

Bộ định tuyến giúp truyền các gói đến đích bằng cách tạo ra đường đi qua các thiết bị mạng được kết nối với nhau thông qua bảng định tuyến. Bộ định tuyến là thiết bị thông minh và chúng lưu trữ thông tin về mạng mà chúng được kết nối. Hầu hết các bộ định tuyến có thể được cấu hình để hoạt động như tường lửa lọc gói và sử dụng danh sách kiểm soát truy cập (ACL). Bộ định tuyến có thể hoạt động nội miền hoặc liên miền.

Bộ định tuyến cũng được sử dụng để chia mạng nội bộ thành hai hoặc nhiều mạng con. Các bộ định tuyến cũng có thể được kết nối nội bộ với các bộ định tuyến khác, tạo ra các vùng hoạt động độc lập. Bộ định tuyến thiết lập giao tiếp bằng cách duy trì các bảng định tuyến về điểm đến và kết nối cục bộ. Các bộ định tuyến thường giao tiếp định tuyến và thông tin khác bằng cách sử dụng giao thức định tuyến.

Bộ định tuyến là thiết bị kết nối hai hoặc nhiều mạng không đồng nhất với nhau. Bởi vì bộ định tuyến và công kết nối là xương sống của các mạng máy tính lớn như Internet, chúng có các tính năng đặc biệt để linh hoạt đối phó với các địa chỉ mạng và kích thước gói khác nhau thông qua việc phân đoạn. Mỗi giao diện bộ định tuyến có mô-đun Giao thức phân giải địa chỉ (ARP) riêng, địa chỉ LAN và địa chỉ Giao thức Internet (IP). Bộ định tuyếnsử dụng bảng định tuyến để tìm đường. Khi nhận được một gói, bộ định tuyến sẽ loại bỏ các tiêu đề và phân tích tiêu đề IP bằng cách xác định địa chỉ nguồn và đích cũng như kiểu dữ liệu, đồng thời lưu ý thời gian đến. Tiêu đề IP và thông tin thời gian đến được nhập vào bảng định tuyến.

## Cầu nối

Cầu nối được sử dụng để kết nối hai hoặc nhiều máy chủ hoặc phân đoạn mạng với nhau. Vai trò cơ bản của bridge trong kiến trúc mạng là lưu trữ và chuyển tiếp các khung giữa các phân đoạn khác nhau mà bridge kết nối. Chúng sử dụng địa chỉ Điều khiển truy cập phương tiện (MAC) phần cứng để chuyển các khung. Bằng cách xem xét địa chỉ MAC của các thiết bị được kết nối với từng đoạn, các cầu nối có thể chuyển tiếp dữ liệu hoặc chặn không cho nó vượt qua. Các cầu nối cũng có thể được sử dụng để kết nối hai mạng LAN vật lý thành một mạng LAN lôgic lớn hơn. Các cầu nối chỉ hoạt động ở các lớp Liên kết Vật lý và Dữ liệu của mô hình OSI. Các cầu nối được sử dụng để chia các mạng lớn hơn thành các phần nhỏ hơn bằng cách nằm giữa hai phân đoạn mạng vật lý và quản lý luồng dữ liệu giữa hai phân đoạn đó.

## **Modem**

Modem (bộ điều biến-giải điều chế) được sử dụng để truyền tín hiệu kỹ thuật số qua đường dây điện thoại tương tự. Do đó, tín hiệu số được modem chuyển đổi thành tín hiệu tương tự có tần số khác nhau và truyền đến modem tại vị trí nhận. Modem nhận thực hiện chuyển đổi ngược lại và cung cấp đầu ra kỹ thuật số cho thiết bị được kết nối với modem, thường là máy tính. Dữ liệu kỹ thuật số thường được truyền đến hoặc từ modem qua đường nối tiếp thông qua giao diện tiêu chuẩn công nghiệp, RS-232. Nhiều công ty điện thoại cung cấp dịch vụ DSL và nhiều nhà khai thác cáp sử dụng modem làm thiết bị đầu cuối. Modem hoạt động trên cả hai lớp Liên kết vật lý và dữ liệu.

## **Bộ lặp**

Bộ lặp là một thiết bị điện tử khuếch đại tín hiệu mà nó nhận được. Bạn có thể coi bộ lặp là một thiết bị nhận tín hiệu và truyền lại ở mức công suất cao hơn hoặc cao hơn để tín hiệu có thể bao phủ khoảng cách xa hơn, hơn 100 mét đối với cáp LAN tiêu chuẩn. Bộ lặp hoạt động trên lớp Vật lý.

## **Điểm truy cập**

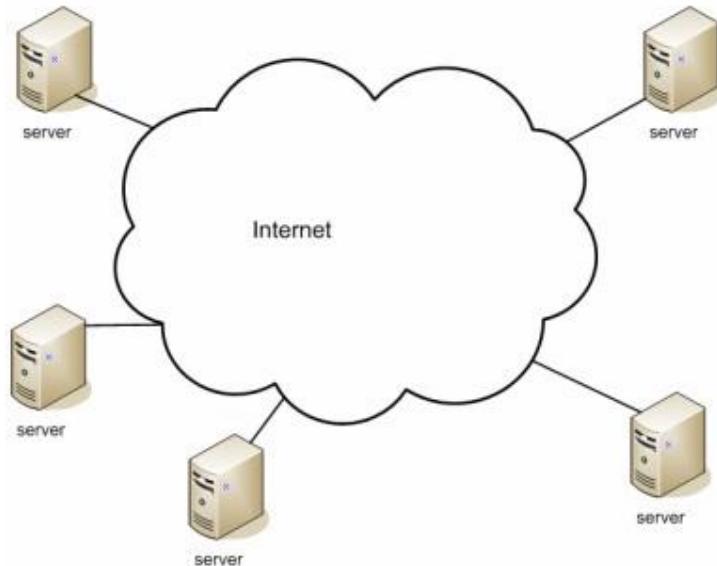
Mặc dù về mặt kỹ thuật, một điểm truy cập (AP) có thể liên quan đến kết nối có dây hoặc không dây, nhưng nó thường có nghĩa là một thiết bị không dây. Một AP hoạt động ở lớp OSI thứ hai, lớp Liên kết dữ liệu và nó có thể hoạt động như một cầu nối kết nối mạng có dây tiêu chuẩn với các thiết bị không dây hoặc như một bộ định tuyến truyền dữ liệu từ điểm truy cập này sang điểm truy cập khác.

Điểm truy cập không dây (WAP) bao gồm thiết bị phát và thiết bị thu (thu phát) được sử dụng để tạo mạng LAN không dây (WLAN). Các điểm truy cập thường là các thiết bị mạng riêng biệt với ăng-ten, bộ phát và bộ điều hợp tích hợp. Các AP sử dụng chế độ mạng cơ sở hạ tầng không dây để cung cấp điểm kết nối giữa WLAN và mạng LAN Ethernet có dây. Họ cũng có một số cổng, cung cấp cho bạn một cách để mở rộng mạng lưới để hỗ trợ các máy khách bổ sung. Tùy thuộc vào quy mô của mạng, một hoặc nhiều AP có thể được yêu cầu để cung cấp phạm vi phủ sóng đầy đủ. Các AP bổ sung được sử dụng để cho phép

truy cập vào nhiều máy khách không dây hơn và mở rộng phạm vi của mạng không dây. Mỗi AP bị giới hạn bởi phạm vi truyền của nó - khoảng cách mà khách hàng có thể đến từ một AP mà vẫn thu được tín hiệu có thể sử dụng và tốc độ xử lý dữ liệu. Khoảng cách thực tế phụ thuộc vào tiêu chuẩn không dây, các vật cản và điều kiện môi trường giữa khách hàng và AP. Các AP cao cấp hơn có ăng-ten công suất cao, cho phép chúng mở rộng khoảng cách tín hiệu không dây có thể truyền đi.

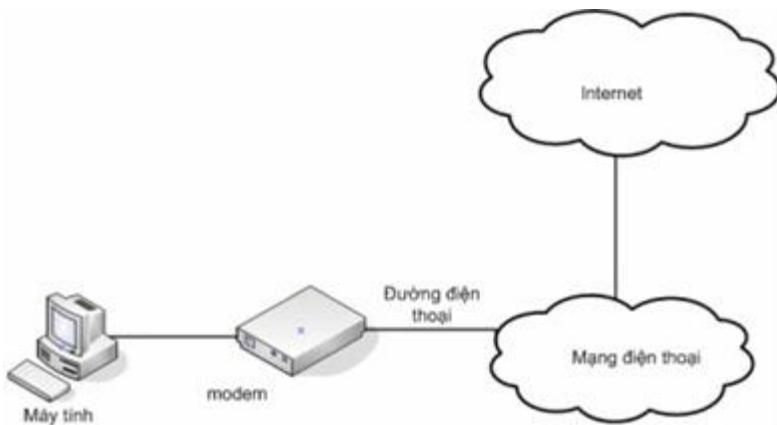
#### 1.4 Các giải pháp công nghệ mạng

Để có thể sử dụng các dịch vụ Internet, người dùng phải kết nối máy tính của mình với Internet. Có nhiều phương thức kết nối, mỗi phương thức có tốc độ truyền nhận dữ liệu khác nhau, tùy thuộc vào nhu cầu sử dụng và điều kiện của người sử dụng. Các phương thức kết nối chính là: điện thoại (dial-up), băng rộng, kết nối qua vệ tinh, kết nối không dây và kết nối thông qua kênh thuê riêng.



Hình 1.3: Internet dưới góc nhìn của người sử dụng

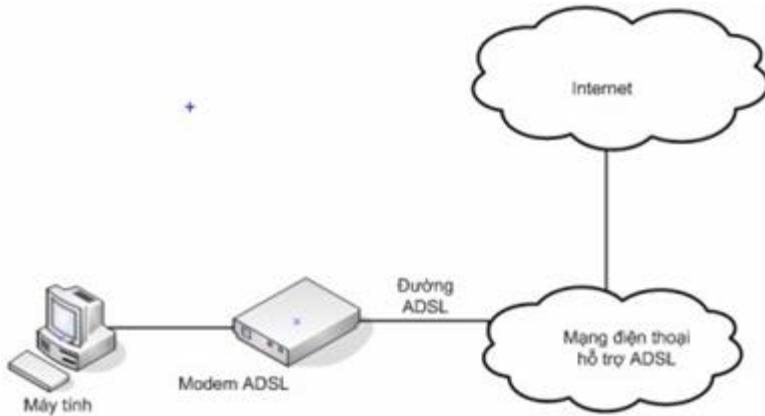
Kết nối quay số qua mạng điện thoại (Dial-up): Người dùng kết nối với Internet thông qua mạng điện thoại. Người dùng cần có một đường điện thoại và một thiết bị kết nối là Modem. Máy tính của người dùng kết nối với Modem và Modem được kết nối tới đường điện thoại.



*Hình 1.4: Kết nối tới Internet qua dial-up*

Đây là phương thức truy cập Internet thông qua đường dây điện thoại bằng cách quay số tới số của nhà cung cấp dịch vụ Internet. Trên lý thuyết, tốc độ kết nối của dial-up dao động từ 20- 56Kbps. Đây là phương thức kết nối chậm nhất trong số các công nghệ truy cập Internet. Truy cập Internet băng rộng: thường được gọi tắt là “Internet băng rộng” hoặc “băng rộng” – là loại hình kết nối Internet tốc độ cao và luôn trong trạng thái kết nối 24/24. Đối với kết nối băng rộng, người ta thường đề cập tới các công nghệ kết nối như DSL (Digital Subscriber Line hay kênh thuê bao số) và “modem cáp” (cable modem) – có khả năng truyền dữ liệu ở tốc độ 521Kbps.

DSL là một tập hợp các công nghệ kết nối Internet tốc độ cao, trong đó có hai công nghệ chính là “DSL bát đối xứng” và “DSL đối xứng”. “DSL bát đối xứng” (ADSL, RADSL, VDSL) có tốc độ tải xuống (download) nhanh nhưng tốc độ tải lên (upload) chậm hơn (nhưng vẫn ở mức có thể chấp nhận được). Trong khi đó, “DSL đối xứng” (SDSL, HDSL, IDSL) có tốc độ download và upload bằng nhau và đều ở mức cao.



*Hình 1.5: Kết nối tới Internet qua ADSL*

ADSL (Asymmetrical DSL - Đường thuê bao số bất đối xứng): Truyền dữ liệu qua đường dây điện thoại có sẵn. ADSL cung cấp một băng thông bất đối xứng trên một đôi dây. Thuật ngữ bất đối xứng ở đây để chỉ sự không cân bằng trong dòng dữ liệu tải xuống và tải lên. Dòng dữ liệu tải xuống có băng thông lớn hơn băng thông dòng dữ liệu tải lên.

ADSL1 cung cấp 1,5Mbps cho đường dữ liệu tải xuống và 16Kbps cho đường dữ liệu tải lên, hỗ trợ chuẩn MPEG-1.

ADSL2 có thể cung cấp băng thông tối 3Mbps cho đường xuống và 16Kbps cho đường lên, hỗ trợ 2 dòng MPEG-1.

ADSL3 có thể cung cấp 6Mbps cho đường xuống và ít nhất 64Kbps cho đường lên, hỗ trợ chuẩn MPEG-2.

RADSL (Rate Adaptive DSL): Là một phiên bản của kết nối ADSL nhưng có khả năng tự điều chỉnh tốc độ kết nối dựa vào chất lượng tín hiệu, ở đó các modem có thể kiểm tra đường truyền khi khởi động và đáp ứng lúc hoạt động theo tốc độ nhanh nhất mà đường truyền có thể cung cấp. RADSL còn được gọi là ADSL có tốc độ biến đổi. Trên thực tế, rất nhiều công nghệ ADSL lại là RADSL.

VDSL/VHDSL (Very High Bit Rate DSL): Tận dụng cáp đồng để kết nối Internet thay cho cáp quang; và cũng giống như ADSL, VDSL có thể chia sẻ chung với đường điện thoại. VDSL là một công nghệ xDSL cung cấp đường truyền bất đối xứng trên một đôi dây

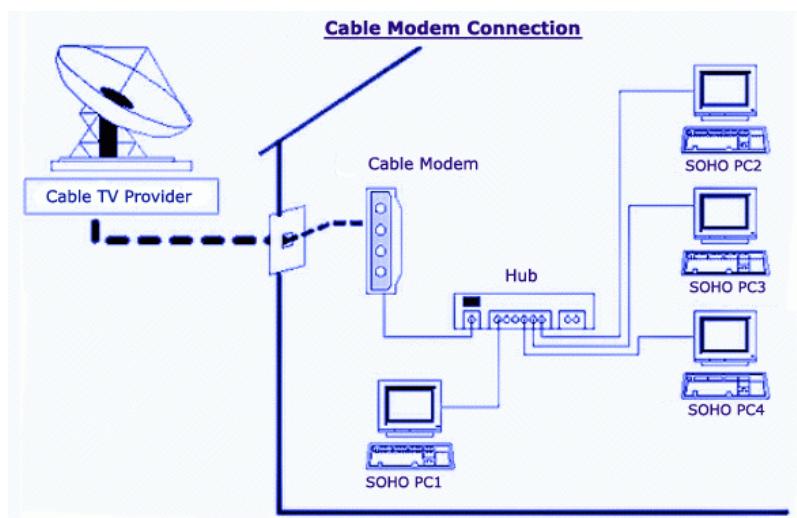
đồng. Dòng bit tải xuống của VDSL là cao nhất trong tất cả các công nghệ của xDSL, đạt tới 52Mbps, dòng tải lên có thể đạt 2,3Mbps. VDSL thường chỉ hoạt động tốt trong các mạng mạch vòng ngắn. VDSL dùng cáp quang để truyền dẫn là chủ yếu, và chỉ dùng cáp đồng ở phía đầu cuối.

HDSL (High Bit Rate DSL): Có tốc độ kết nối cao hơn ADSL nhưng không cho phép chia sẻ chung với đường điện thoại. Tốc độ của HDSL và HDSL-2 dao động từ 668Kbps – 2.048Mbps (E1).

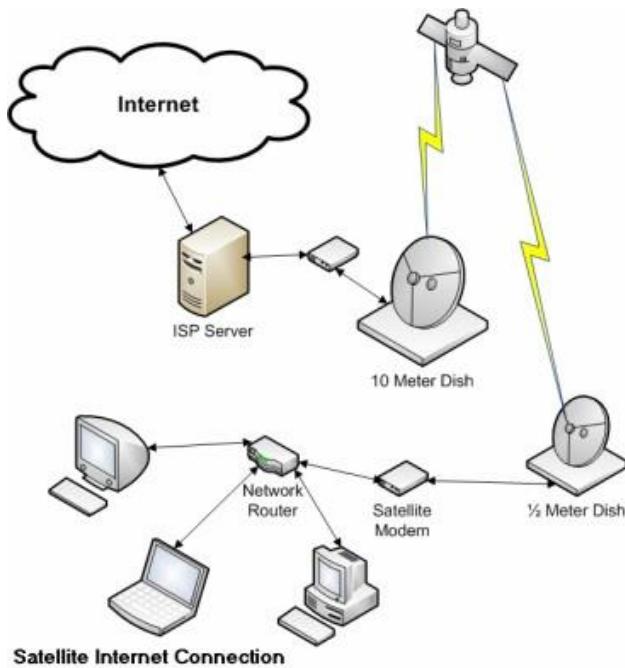
SDSL (Symmetric DSL): Là một phiên bản của HDSL thường có tốc độ truyền tải dữ liệu từ 160Kbps – 1,5Mbps. Cũng giống HDSL, SDSL không chia sẻ đường kết nối với điện thoại.

IDSL (ISDN Digital Subscriber Line) có khả năng truyền tải dữ liệu ở khoảng cách xa hơn so với HDSL và SDSL – 7.924m. Với khoảng cách này, tốc độ truyền dữ liệu của IDSL là 144Mbps.

Cable Modem: Là phương thức kết nối Internet thông qua một loại modem đặc biệt được thiết kế riêng cho việc truyền dữ liệu thông qua mạng truyền hình cáp. Cable modem có thể tăng đáng kể băng thông giữa máy tính người dùng và nhà cung cấp dịch vụ Internet (ISP- Internet service provider), đặc biệt là tốc độ tải xuống (từ ISP tới người dùng).



Hình 1.6: Kết nối tới Internet qua cable



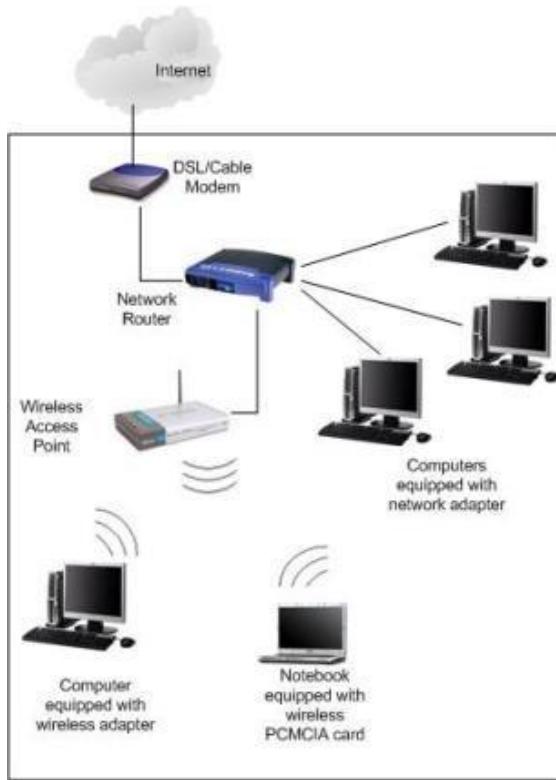
*Hình 1.7: Kết nối qua vệ tinh*

Không giống với loại modem analog, modem cáp kết nối với máy tính thông qua cổng Ethernet nên trạng thái kết nối luôn luôn ở dạng sẵn sàng. Tốc độ dữ liệu truyền đi trong cable modem phụ thuộc vào số lượng người sử dụng truyền và nhận dữ liệu vào cùng một thời điểm.

Dịch vụ Internet vệ tinh thường được sử dụng tại các khu vực mà các phương pháp truy cập Internet bình thường không thể tiếp cận được (vùng sâu, vùng xa, hải đảo...).

### **Kết nối không dây**

Khái niệm về Hotspot: Hotspot là một địa điểm mà tại đó có cung cấp các dịch vụ kết nối không dây và dịch vụ truy cập Internet tốc độ cao, thông qua hoạt động của các thiết bị thu phát không dây (Wireless Access Point).



*Hình 1.8: Kết nối không dây*

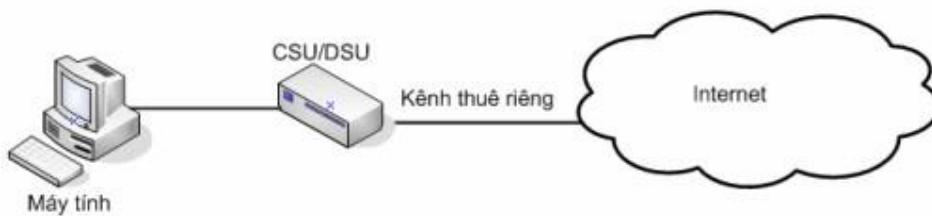
Wi-Fi là tên viết tắt của cụm từ “Wireless Fidelity” - một tập hợp các chuẩn tương thích với mạng không dây nội bộ (WLAN-Wireless Local Area Network) dựa trên đặc tả IEEE 802.11 (802.11a, 802.11b, 802.11g, N, AH...). Wi-Fi cho phép các máy tính hoặc PDA (Personal Digital Assistant, thiết bị cá nhân kỹ thuật số) hỗ trợ kết nối không dây có thể truy cập vào mạng Internet trong phạm vi phủ sóng của điểm truy cập không dây (hay còn gọi là “hotspot”). Tốc độ kết nối của các chuẩn thuộc Wi-Fi rất khác nhau, cụ thể:

- + 802.11: Dùng cho mạng WLAN, có tốc độ truyền tải dữ liệu từ 1-2Mbps.
- + 802.11a: Là phần mở rộng của 802.11, áp dụng cho mạng WLAN, có tốc độ kết nối lên tới 54Mbps.
- + 802.11b (còn gọi là 802.11 High Rate hoặc Wi-Fi): Cũng là phần mở rộng của 802.11 dành cho mạng WLAN, có tốc độ truyền dữ liệu tối đa ở mức 11Mbps.
- + 802.11g: Sử dụng cho mạng WLAN với tốc độ kết nối tối đa trên 20Mbps.
- + 802.11N: Sử dụng cho mạng WLAN với tốc độ kết nối tối đa trên 54Mbps.
- + 802.ah: Sử dụng cho mạng WLAN với tốc độ kết nối tối đa trên 760 Mbps.

## **Kết nối thông qua kênh thuê riêng (Leased-Line)**

Leased-Line, hay còn gọi là kênh thuê riêng, là một hình thức kết nối trực tiếp giữa các node mạng sử dụng kênh truyền dẫn số liệu thuê riêng, là dịch vụ đường truyền Internet có công nghệ riêng biệt dành cho các văn phòng, công ty có yêu cầu cao về chất lượng dịch vụ. Kênh truyền dẫn số liệu thông thường cung cấp cho người sử dụng sự lựa chọn trong suốt về giao thức đấu nối hay nói cách khác, có thể sử dụng các giao thức khác nhau trên kênh thuê riêng như PPP, HDLC, LAPB...

Khác với kết nối Internet thông thường, đường truyền kênh thuê riêng có thể cung cấp mọi tốc độ từ 256Kbps đến hàng chục Gbps với cam kết tốt nhất về độ ổn định và tốc độ kết nối.



*Hình 1.9: Kết nối Leased - line*

Về mặt hình thức, kênh thuê riêng có thể là các đường cáp đồng trực tiếp kết nối giữa hai điểm hoặc có thể bao gồm các tuyến cáp đồng và các mạng truyền dẫn khác nhau. Khi kênh thuê riêng phải đi qua các mạng khác nhau, các quy định về các giao tiếp với mạng truyền dẫn sẽ được quy định bởi nhà cung cấp dịch vụ. Do đó, các thiết bị đầu cuối CSU/DSU (Channel Service Unit/Data Service Unit) cần thiết để kết nối kênh thuê riêng sẽ phụ thuộc vào nhà cung cấp dịch vụ. Một số các chuẩn kết nối chính được sử dụng là HDSL, G703...

Khi sử dụng kênh thuê riêng, người sử dụng cần thiết phải có đủ các giao tiếp trên các bộ định tuyến sao cho có một giao tiếp kết nối WAN (Wireless Local Area Network) cho mỗi kết nối kênh thuê riêng tại mỗi node. Điều đó có nghĩa là, tại điểm node có kết nối kênh thuê riêng đến 10 điểm khác nhau thiết phải có đủ 10 giao tiếp WAN để phục vụ cho các kết nối kênh thuê riêng. Đây là một vấn đề hạn chế về đầu tư thiết bị ban đầu, không

linh hoạt trong mở rộng phát triển, phức tạp trong quản lý, đặc biệt là chi phí thuê kênh lớn đối với các yêu cầu kết nối xa về khoảng cách địa lý.

Giao thức sử dụng với leased-line là HDLC, PPP, LAPB.

- + HDLC (High – level Data Link Control): là giao thức được sử dụng với họ bộ định tuyến Cisco hay nói cách khác chỉ có thể sử dụng HDLC khi cả hai phía của kết nối leased-line đều là bộ định tuyến Cisco.
- + PPP (Point – to – point Protocol): là giao thức chuẩn quốc tế, tương thích với tất cả các bộ định tuyến của các nhà sản xuất khác nhau. Khi đấu nối kênh leased-line giữa một phía là thiết bị của Cisco và một phía là thiết bị của hãng thứ ba thì nhất thiết phải dùng giao thức đấu nối này. PPP là giao thức lớp 2 cho phép nhiều giao thức mạng khác nhau có thể chạy trên nó, do vậy nó được sử dụng phổ biến.
- + LAPB (Link Acess Procedure Balanced): là giao thức truyền thông lớp 2 tương tự như giao thức mạng X.25 với đầy đủ các thủ tục, quá trình kiểm soát truyền dẫn, phát triển và sửa lỗi. LAPB ít được sử dụng.

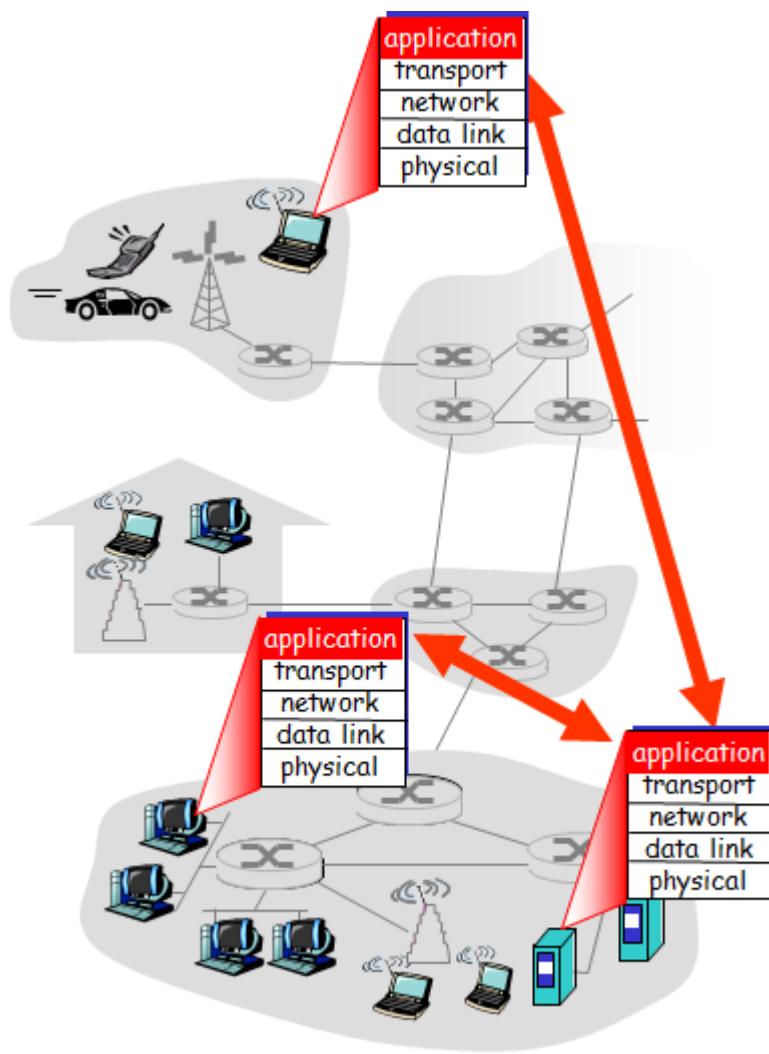
## 1.5 Kết luận chương

Trong chương 1 giới thiệu cho người đọc các vấn đề cơ bản nhất về Internet từ các khái niệm ban đầu tới kiến trúc chung của mô hình kết nối liên mạng hay mạng toàn cầu. Kiến trúc cốt lõi của internet là dựa trên nguyên lý của bộ giao thức TCP/IP cho dù các công nghệ mới đã thay đổi góc nhìn về internet. Các thành phần thiết bị mạng cơ bản nhất của mạng internet được giới thiệu cùng với các giải pháp truy nhập mạng Internet.

## CHƯƠNG 2. CÁC ỨNG DỤNG VÀ GIAO THỨC MẠNG

### 2.1 Tổng quan về các ứng dụng và dịch vụ hạ tầng

Giả sử chúng ta có ý tưởng cho một ứng dụng mạng mới. Ứng dụng này có thể là một dịch vụ rất tốt cho mọi người hoặc đơn giản chỉ là để phát triển. Cho dù động cơ phát triển ứng dụng là gì thì chúng ta hãy cùng tìm hiểu xem cách thực hiện ý tưởng vào môi trường thực của ứng dụng mạng như thế nào.



Hình 2.1: Truyền thông giữa các hệ thống đầu cuối ở lớp ứng dụng

Phần chính của quá trình phát triển ứng dụng mạng là viết chương trình chạy trên các hệ thống đầu cuối khác nhau để thực hiện truyền thông qua mạng. Ví dụ, trong ứng dụng

web có hai chương trình phần mềm riêng biệt truyền thông với nhau: phần mềm trình duyệt chạy trên thiết bị đầu cuối của người sử dụng (máy tính bàn, máy tính xách tay, PDA, điện thoại di động, ...) và phần mềm máy chủ chạy trên máy chủ web. Hoặc ví dụ khác, trong hệ thống chia sẻ tệp ngang hàng (P2P file sharing) có một chương trình chạy ở cả hai trạm đều tham gia vào cộng đồng chia sẻ tệp. Trong trường hợp này, các chương trình trong các trạm khác nhau có thể gần giống hoặc giống hệt nhau.

Vì vậy, khi phát triển một ứng dụng mới thì cần phải xây dựng phần mềm chạy trên nhiều hệ thống cuối. Phần mềm này có thể viết dựa trên C, Java hoặc Python. Điều quan trọng là chúng ta không cần phải viết phần mềm chạy trên các thiết bị mạng lỗi như bộ định tuyến hay bộ chuyển mạch. Ngay cả khi muốn viết phần mềm ứng dụng cho các thiết bị mạng lỗi cũng không cần phải làm như vậy. Như đã biết, các thiết bị mạng lỗi không thực hiện chức năng lớp ứng dụng mà thực hiện chức năng ở các lớp thấp hơn, chủ yếu là ở lớp mạng và các lớp dưới. Thiết kế cơ bản này đẩy phần mềm ứng dụng vào trong hệ thống cuối (**Error! Reference source not found.**), tạo điều kiện cho sự phát triển và triển khai chóng của rất nhiều ứng dụng mạng.

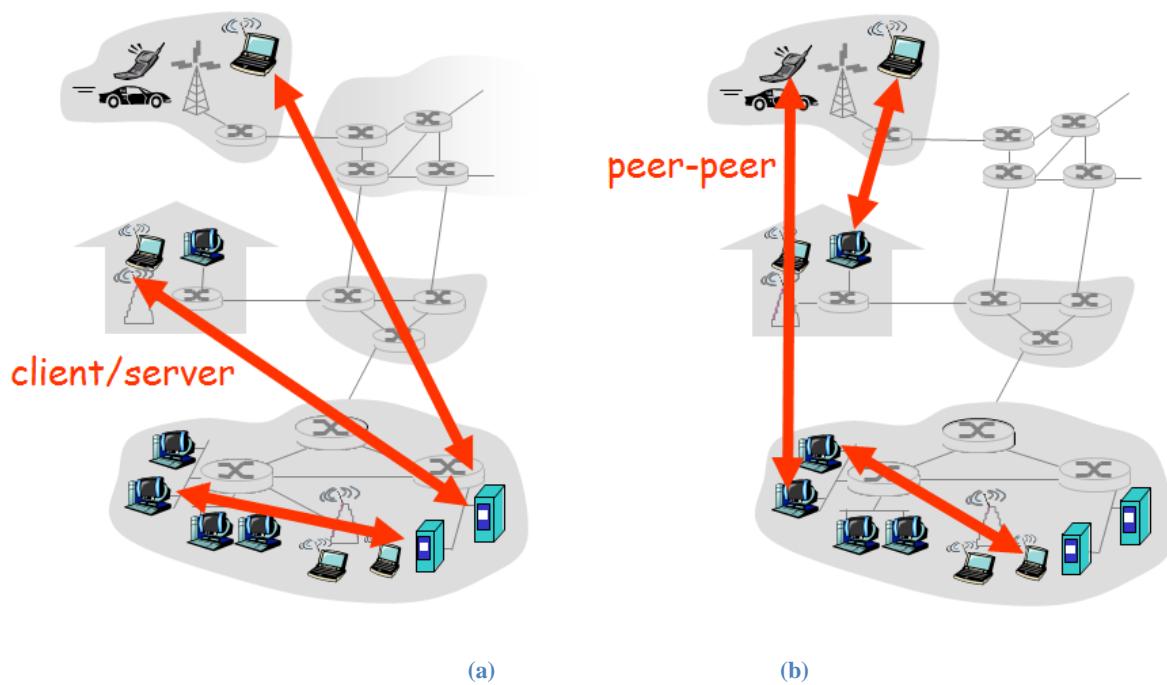
Trước khi đi vào việc mã hóa phần mềm, cần phải có kế hoạch về kiến trúc cho ứng dụng. Lưu ý là kiến trúc ứng dụng khác hoàn toàn với kiến trúc mạng (như kiến trúc phân lớp của Internet). Từ quan điểm của người phát triển ứng dụng, kiến trúc mạng là cố định và cung cấp một tập các dịch vụ cho các ứng dụng. Mặt khác, **kiến trúc ứng dụng** được các nhà phát triển ứng dụng thiết kế và chỉ rõ cách cấu trúc ứng dụng đó trên nhiều hệ thống cuối khác nhau. Khi xây dựng kiến trúc ứng dụng, một nhà phát triển ứng dụng có thể lựa chọn một trong hai mô hình nổi bật được dùng trong các ứng dụng mạng hiện tại: kiến trúc khách/chủ (client/server) hoặc kiến trúc ngang hàng (P2P - peer-to-peer).

### **Kiến trúc khách/chủ:**

Trong **kiến trúc khách/chủ** luôn có một máy trạm hoạt động gọi là máy chủ (server). Nó phục vụ các yêu cầu từ nhiều máy trạm khác (client). Các máy khách có thể hoạt động liên tục hoặc không. Ví dụ về ứng dụng web: luôn có máy chủ web hoạt động để phục vụ

yêu cầu từ các trình duyệt chạy trên trạm khách. Khi máy chủ web nhận một yêu cầu về đối tượng từ trạm khách thì nó đáp ứng thông qua việc gửi đối tượng đó tới trạm khách.

Để ý là với kiến trúc khách/chủ, các máy khách không truyền thông trực tiếp với nhau, ví dụ trong ứng dụng web thì hai trình duyệt không truyền thông trực tiếp với nhau. Một đặc điểm nữa của kiến trúc này là máy chủ có địa chỉ cụ thể và cố định (địa chỉ IP). Chính vì đặc điểm này và việc máy chủ luôn hoạt động nên một máy khách luôn có thể kết nối với máy chủ bằng việc gửi tin tới địa chỉ của máy chủ. Một vài ứng dụng nổi tiếng với kiến trúc khách/chủ là web, FTP, Telnet và e-mail. **Error! Reference source not found.**(a) à minh họa kiến trúc khách/chủ.



Hình 2.2: Kiến trúc khách/chủ (a) và ngang hàng (b)

Thường trong ứng dụng khách/chủ, một trạm chủ đơn lẻ không có khả năng đáp ứng kịp những yêu cầu từ các máy khách của nó. Ví dụ, một điểm kết nối xã hội phổ biến có thể nhanh chóng bị sập nếu nó chỉ có một máy chủ xử lý tất cả các yêu cầu. Vì lý do này, một nhóm trạm lớn – đôi khi được gọi là trung tâm dữ liệu – được dùng để tạo ra một máy chủ ảo mạnh trong kiến trúc khách/chủ. Các dịch vụ ứng dụng dựa trên kiến trúc khách/chủ thường là cần nhiều cơ sở hạ tầng vì chúng yêu cầu nhà cung cấp dịch vụ phải mua, cài đặt

và bảo dưỡng cụm máy chủ. Hơn nữa, các nhà cung cấp dịch vụ phải trả phí cho việc kết nối liên mạng định kỳ và băng thông cho việc gửi và nhận dữ liệu tới và từ Internet. Các dịch vụ phổ biến như công cụ tìm kiếm (ví dụ như Google), thương mại Internet (như Amazon và e-Bay), thư điện tử trên Web (như Yahoo Mail), kết nối mạng xã hội (như MySpace và Facebook) và chia sẻ video (như YouTube) là những dịch vụ cần nhiều cơ sở hạ tầng và tốn chi phí để cung cấp.

### Kiến trúc ngang hàng:

Trong **kiến trúc ngang hàng** (P2P: peer-to-peer) có rất ít (hoặc không có) máy chủ hạ tầng luôn hoạt động. Thay vào đó, ứng dụng khai thác truyền thông trực tiếp giữa cặp trạm kết nối liên tục, gọi là các thiết bị ngang hàng (peer). Các thiết bị ngang hàng này không phải của nhà cung cấp dịch vụ mà là các máy tính bàn, máy tính xách tay do người sử dụng điều khiển và hầu hết thiết bị ngang hàng nằm trong nhà, trong trường học hay trong các công sở. Vì truyền thông giữa các thiết bị ngang hàng không cần qua máy chủ dành riêng nên kiến trúc này được gọi là ngang hàng.

Rất nhiều ứng dụng thông dụng và ứng dụng cần lưu lượng lớn ngày nay dựa trên kiến trúc ngang hàng như phân bổ tệp (như BitTorrent), chia sẻ tệp (như eMule và LimeWire), điện thoại Internet (như Skype) và IPTV (như PPLive). Kiến trúc ngang hàng được minh họa trong **Error! Reference source not found.(b)**. Để ý rằng một số ứng dụng ô kiếng lai ghép, kết hợp cả các phần tử khách/chủ và ngang hàng. Ví dụ, với nhiều ứng dụng gửi tin nhắn tức thời, các máy chủ được dùng để truy vết các địa chỉ IP của người sử dụng, nhưng các bản tin người sử dụng-người sử dụng thì gửi trực tiếp giữa các máy trạm của người sử dụng (mà không cần chuyển tiếp qua các máy chủ trung gian).

Một trong những đặc điểm hấp dẫn nhất của kiến trúc P2P là khả năng tự mở rộng (self-scalability). Ví dụ, trong ứng dụng chia sẻ tệp P2P, mặc dù mỗi thiết bị ngang hàng tạo ra tải trọng bằng việc yêu cầu lấy tệp, mỗi thiết bị ngang hàng cũng tăng thêm dung lượng dịch vụ cho hệ thống bằng cách phân phối tệp tới các thiết bị ngang hàng khác. Kiến trúc P2P cũng hiệu quả về chi phí vì thông thường chúng không cần hạ tầng máy chủ và

băng thông máy chủ. Để giảm chi phí, các nhà cung cấp dịch vụ (MSN, Yahoo...) ngày càng quan tâm đến việc sử dụng kiến trúc P2P cho các ứng dụng của họ.

Tuy nhiên, các ứng dụng P2P trong tương lai gặp phải ba thách thức chính:

1. *Tính thân thiện của các nhà cung cấp dịch vụ Internet (ISP).* Hầu hết các ISP (gồm cả ISP cáp và DSL) đã ước định sử dụng băng thông bất đối xứng, nghĩa là băng thông chiều xuống lớn hơn băng thông chiều lên. Song các ứng dụng phân bổ tệp và video trực tuyến P2P thì lại chuyển lưu lượng lên từ các máy chủ đến các ISP dân thường, do đó sẽ đẩy áp lực đáng kể lên các ISP. Các ứng dụng P2P tương lai cần phải được thiết kế để thân thiện với các ISP.
2. *An toàn.* Vì có đặc tính mở và phân bổ rộng khắp nên các ứng dụng P2P sẽ có thách thức về vấn đề an toàn.
3. *Khích lệ.* Thành công của các ứng dụng P2P còn phụ thuộc vào việc thuyết phục được người sử dụng tình nguyện chia sẻ nguồn lực như băng thông, bộ nhớ và khả năng tính toán cho các ứng dụng, đây cũng là thách thức trong thiết kế cần được lưu ý.

Trong chương này sẽ trình bày các ứng dụng mạng và các giao thức lớp ứng dụng cho cả mô hình khách/chủ và mô hình P2P. Một giao thức lớp ứng dụng định nghĩa các thủ tục của ứng dụng, chạy trên các hệ thống cuối khác nhau, qui định cách thức chuyển các bản tin cho nhau như thế nào. Cụ thể là một giao thức lớp ứng dụng định nghĩa các vấn đề sau:

- + Loại bản tin trao đổi, ví dụ bản tin yêu cầu và bản tin phản hồi;
- + Cú pháp của các loại bản tin khác nhau, như các trường trong bản tin và cách mô tả các trường này;
- + Ngữ nghĩa của các trường, tức là ý nghĩa của trường thông tin;
- + Quy tắc xác định một tiến trình gửi và phản hồi bản tin khi nào và như thế nào.

Một vài giao thức lớp ứng dụng được đặc tả trong các RFC và vì thế nó mang tính công khai. Ví dụ, giao thức lớp ứng dụng của web là HTTP (Hyper-Text Transfer Protocol - giao thức truyền siêu văn bản) [RFC 2616]. Nếu một nhà phát triển trình duyệt tuân theo các quy tắc của RFC HTTP thì trình duyệt này sẽ có khả năng nhận các trang web từ bất kỳ máy chủ web nào cũng tuân theo quy tắc của RFC HTTP. Rất nhiều giao thức lớp ứng dụng

khác là độc quyền và không công khai. Ví dụ, rất nhiều hệ thống chia sẻ tệp P2P sử dụng giao thức lớp ứng dụng độc quyền.

Điều quan trọng là cần phân biệt được ứng dụng mạng với giao thức lớp ứng dụng. Giao thức lớp ứng dụng chỉ là một phần của ứng dụng mạng. Ví dụ, Web là một ứng dụng khách-chủ cho phép người sử dụng lấy dữ liệu từ máy chủ Web theo yêu cầu. Ứng dụng Web gồm rất nhiều phần tử, bao gồm các khuôn dạng tài liệu chuẩn (HTML), các trình duyệt Web (ví dụ Firefox và Microsoft Internet Explorer), các máy chủ Web (ví dụ máy chủ Apache và Microsoft) và một giao thức lớp ứng dụng. Giao thức lớp ứng dụng của Web là HTTP, nó định nghĩa khuôn dạng và trình tự của các bản tin chuyển giữa phần mềm trình duyệt và máy chủ Web. Vì thế HTTP chỉ là một phần (mặc dù là phần quan trọng) của ứng dụng Web.

Một ví dụ khác là ứng dụng thư điện tử Internet cũng có rất nhiều thành phần, bao gồm các máy chủ thư điện tử chứa các hòm thư của người sử dụng, những chương trình đọc thư cho phép người sử dụng đọc và tạo ra các bản tin, một chuẩn định nghĩa cấu trúc của bản tin thư điện tử và các giao thức lớp ứng dụng định nghĩa cách các bản tin chuyển giữa các máy chủ, cách bản tin chuyển giữa máy chủ và chương trình đọc thư, và cách các nội dung của các phần cấu thành bản tin thư (ví dụ như tiêu đề của thư) được biên dịch như thế nào. Giao thức lớp ứng dụng chủ yếu cho thư điện tử là SMTP [RFC 5321]. Tuy nhiên, giao thức này cũng chỉ là một phần (dù là phần quan trọng) của ứng dụng thư điện tử.

## 2.2 Ứng dụng WEB và các giao thức

### 2.2.1 Tổng quan về ứng dụng WEB và giao thức HTTP

Cho tới đầu những năm 1990, Internet chủ yếu được các nhà nghiên cứu và sinh viên các trường đại học dùng để truy nhập vào các trạm ở xa, truyền tệp từ trạm nội bộ tới trạm từ xa và ngược lại, để nhận và gửi tin tức, thư điện tử. Mặc dù những ứng dụng đó là cực kỳ hữu dụng cho tới ngày nay, song ngoài cộng đồng nghiên cứu và học thuật thì Internet vẫn còn xa lạ với mọi người. Sau đó từ những năm 1990 một ứng dụng chủ đạo mới xuất hiện đó là WWW (World Wide Web). Web là ứng dụng Internet đầu tiên có được sự quan

tâm của toàn công chúng. Nó đã thay đổi nhanh chóng (và sẽ còn tiếp tục làm thay đổi) cách con người tương tác bên trong và bên ngoài môi trường làm việc của họ. Web đã thúc đẩy Internet từ vị trí là một trong nhiều mạng truyền dữ liệu thành mạng dữ liệu duy nhất.

Có lẽ hầu hết người sử dụng đều thấy Web vận hành *theo yêu cầu*. Người sử dụng nhận được những gì họ muốn và khi họ muốn. Điều này không giống với truyền thanh và truyền hình quảng bá, chúng buộc người sử dụng phải nhận nội dung mà nhà cung cấp mang lại. Ngoài việc sẵn sàng cung cấp theo yêu cầu, Web còn có rất nhiều đặc tính tuyệt vời mà mọi người thích thú và mong chờ. Việc tạo thông tin trên Web khá dễ dàng đối với bất cứ ai – tất cả mọi người đều có thể trở thành người phát hành thông tin với chi phí cực thấp. Các siêu liên kết và công cụ tìm kiếm giúp chúng ta duyệt vô cùng nhiều các trang Web. Những hình ảnh đồ họa tạo cảm giác thực cho chúng ta. Các khuôn dạng, chương trình Java (Java applets) và các thiết bị khác cho phép chúng ta tương tác với các trang Web. Và hơn nữa, Web cung cấp giao diện trình đơn (menu) tới rất nhiều kho dữ liệu audio và video lưu trữ trên Internet, từ đó có thể truy nhập đa phương tiện theo yêu cầu.

**Giao thức truyền siêu văn bản (HTTP)** - giao thức lớp ứng dụng của Web - là trái tim của ứng dụng Web. Nó được định nghĩa trong RFC 1945 và RFC 2616. HTTP được thực hiện trong hai chương trình: chương trình máy khách và chương trình máy chủ. Chương trình máy khách và chương trình máy chủ thực hiện trên các hệ thống đầu cuối khác nhau, giao tiếp với nhau bằng cách trao đổi các bản tin HTTP. HTTP định nghĩa cấu trúc của các bản tin và phương thức máy khách và máy chủ trao đổi các bản tin. Trước khi trình bày chi tiết về HTTP chúng ta xem xét một số thuật ngữ Web.

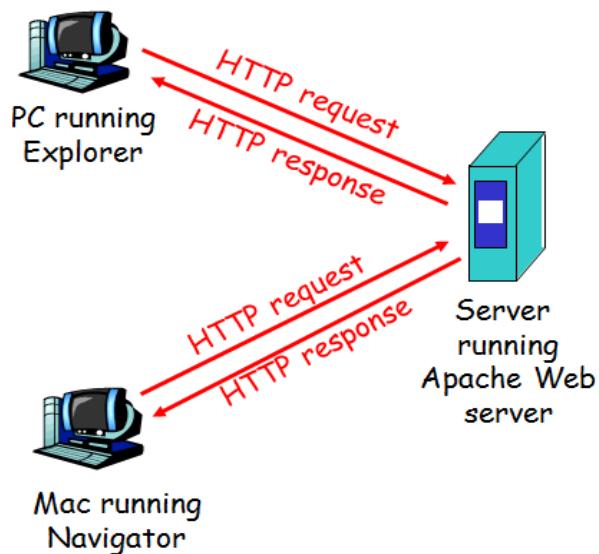
Một trang Web (Web page) - còn được gọi là tài liệu - cấu thành từ các đối tượng (object). Một đối tượng đơn giản chỉ là một tệp (file) - như tệp HTML, hình ảnh JPEG, ứng dụng Java trên trang Web hoặc một đoạn video - có khả năng đánh địa chỉ bằng một URL. Hầu hết các trang Web được cấu thành từ tệp HTML cơ sở và một số đối tượng tham chiếu. Ví dụ, nếu trang Web chứa các câu lệnh HTML và 5 hình ảnh JPEG thì trang Web đó có 6 đối tượng: tệp HTML cơ sở và 5 hình ảnh. Tệp HTML cơ sở tham chiếu các đối tượng khác

trong trang với các URL của đối tượng đó. Mỗi URL có hai thành phần: tên trạm của máy chủ chứa đối tượng và tên đường dẫn tới đối tượng. Ví dụ một URL như sau:

*http://www.someschool.edu/someDepartment/picture.gif*

URL trên có *www.someschool.edu* là tên máy trạm và */someDepartment/picture.gif* là tên đường dẫn. Vì các trình duyệt Web (như Internet Explorer và Firefox) triển khai bên phía máy khách (client) của HTTP trong ngữ cảnh của Web, nên chúng ta sẽ dùng từ trình duyệt (browser) và máy khách (client) thay thế cho nhau. Các máy chủ Web được triển khai bên phía máy chủ (server) của HTTP chứa các đối tượng Web, mỗi đối tượng được đánh một địa chỉ URL.

HTTP xác định các máy khách Web yêu cầu trang Web từ máy chủ Web và các máy chủ truyền các trang Web này tới máy khách như thế nào. Chúng ta thảo luận tương tác giữa máy khách và máy chủ chi tiết sau, song ý tưởng chung về chúng được minh họa trong hình 2.3. Khi người sử dụng yêu cầu một trang Web (ví dụ khi người đó nhấp chuột vào một siêu liên kết - hyperlink), trình duyệt sẽ gửi các bản tin yêu cầu HTTP về các đối tượng trong trang tới máy chủ. Máy chủ nhận được yêu cầu này và đáp ứng bằng bản tin đáp ứng HTTP chứa các đối tượng đó.



Hình 2.3: Hoạt động yêu cầu-đáp ứng của HTTP

HTTP sử dụng TCP làm giao thức lớp giao vận nền (thay vì chạy trên UDP). Đầu tiên máy khách HTTP sẽ khởi tạo kết nối TCP với máy chủ. Một khi kết nối được thiết lập thì các tiến trình duyệt và máy chủ sẽ truy nhập TCP thông qua giao diện socket của nó. Như đã mô tả ở chương trước, giao diện socket phía máy khách là cánh cửa giữa tiến trình máy khách và kết nối TCP, ở bên máy chủ là cánh cửa giữa tiến trình máy chủ và kết nối TCP. Máy khách gửi các bản tin yêu cầu HTTP vào giao diện socket và nhận bản tin đáp ứng HTTP cũng từ giao diện socket của nó. Tương tự như vậy, máy chủ HTTP nhận bản tin yêu cầu từ socket và gửi bản tin đáp ứng vào giao diện socket của nó. Một khi máy khách gửi bản tin vào giao diện socket của nó thì bản tin đó đã ra khỏi sự kiểm soát của máy khách và đi vào phạm vi kiểm soát của TCP. Như trong chương trước đã đưa ra, TCP cung cấp dịch vụ truyền dữ liệu tin cậy cho HTTP. Điều này có nghĩa là mỗi bản tin yêu cầu HTTP do tiến trình máy khách gửi sẽ được chuyển nguyên vẹn tới bên máy chủ và tương tự, mỗi bản tin đáp ứng HTTP do tiến trình máy chủ gửi sẽ được chuyển nguyên vẹn tới máy khách. Ta thấy một trong những ưu điểm lớn của kiến trúc phân lớp ở đây, đó là HTTP không phải quan tâm đến dữ liệu tổn thất, hay chi tiết việc TCP khôi phục hay tái sắp xếp dữ liệu trong mạng như thế nào. Công việc đó là của TCP và của các giao thức lớp dưới trong chồng giao thức đảm nhiệm.

Điều quan trọng cần lưu ý là máy chủ gửi các tệp được yêu cầu tới máy khách mà không lưu trữ thông tin về trạng thái của máy khách. Nếu một máy khách nào đó yêu cầu cùng một đối tượng hai lần trong vài giây, thì máy chủ sẽ không phản hồi là nó vừa gửi đối tượng đó cho máy khách, mà lại tiếp tục gửi lại đối tượng đó vì nó hoàn toàn quên những việc đã làm trước đó. Vì máy chủ HTTP không duy trì thông tin về máy khách nên nó được gọi là **giao thức phi trạng thái** (stateless protocol). Chúng ta cũng nhận thấy là Web sử dụng kiến trúc ứng dụng khách-chủ. Một máy chủ Web thì luôn hoạt động, có địa chỉ IP cố định và nó phục vụ các yêu cầu từ hàng triệu trình duyệt khác nhau.

### **2.2.2 Hoạt động của giao thức HTTP**

Trong nhiều ứng dụng Internet, truyền thông máy khách và máy chủ có thể kéo dài trong một khoảng thời gian, với việc máy khách tạo ra một loạt yêu cầu và máy chủ đáp ứng từng yêu cầu đó. Phụ thuộc vào ứng dụng và cách sử dụng ứng dụng, một loạt các yêu cầu có thể được thực hiện đi thực hiện lại, theo chu kỳ hoặc không liên tục. Khi tương tác máy khách-máy chủ được thực hiện trên TCP thì nhà phát triển ứng dụng cần phải ra quyết định quan trọng – mỗi cặp yêu cầu/đáp ứng gửi trên một kết nối TCP riêng hay là tất cả yêu cầu và phản hồi tương ứng sẽ được gửi trên cùng một kết nối TCP? Nếu gửi trên từng kết nối riêng thì ứng dụng được gọi là sử dụng **kết nối không liên tục** (non-persistent connection) còn trong trường hợp dùng chung kết nối thì được gọi là **kết nối liên tục** (persistent connection). Với giao thức HTTP, có thể dùng cả kết nối không liên tục và liên tục. Mặc dù trong chế độ mặc định HTTP sử dụng kết nối liên tục, song máy khách và máy chủ HTTP có thể được cấu hình để kết nối không liên tục.

#### **HTTP kết nối không liên tục**

Hãy xem xét các bước truyền một trang Web từ máy chủ tới máy khách trong trường hợp kết nối không liên tục. Giả sử trang Web gồm một tệp HTML cơ sở và 10 hình ảnh JPEG, tổng cộng là 11 đối tượng trên cùng máy chủ. URL cho tệp HTML cơ sở là:

```
http://www.someschool.edu/someDepartment/home.index
```

Các hoạt động sẽ xảy ra như sau:

1. Tiên trình máy khách HTTP khởi tạo kết nối TCP tới máy chủ <http://www.someschool.edu> trên cổng 80, số cổng mặc định của HTTP. Kết hợp cùng với kết nối TCP này sẽ có một socket phía máy khách và một socket phía máy chủ.
2. Máy khách HTTP gửi một bản tin yêu cầu HTTP tới máy chủ qua socket của nó. Bản tin yêu cầu có tên đường dẫn /someDepartment/home.index.

3. Tiết trình máy chủ HTTP nhận bản tin yêu cầu qua socket của nó, lấy đối tượng /someDepartment/home.index từ bộ nhớ (RAM hoặc ổ đĩa cứng), đóng gói đối tượng vào một bản tin đáp ứng HTTP và gửi bản tin đáp ứng này tới máy khách qua socket của nó.
4. Tiết trình máy chủ HTTP báo cho TCP đóng kết nối TCP (song TCP không kết thúc kết nối cho tới khi nó chắc chắn là máy khách đã nhận nguyên vẹn bản tin đáp ứng này).
5. Máy khách HTTP nhận bản tin đáp ứng. Kết nối TCP kết thúc. Bản tin chỉ dẫn đối tượng đóng gói là tệp HTML. Máy khách lấy tệp từ bản tin đáp ứng, kiểm tra tệp HTML, và tìm tham chiếu đến 10 đối tượng JPEG.
6. Bốn bước đầu tiên được lặp lại cho mỗi đối tượng JPEG tham chiếu.

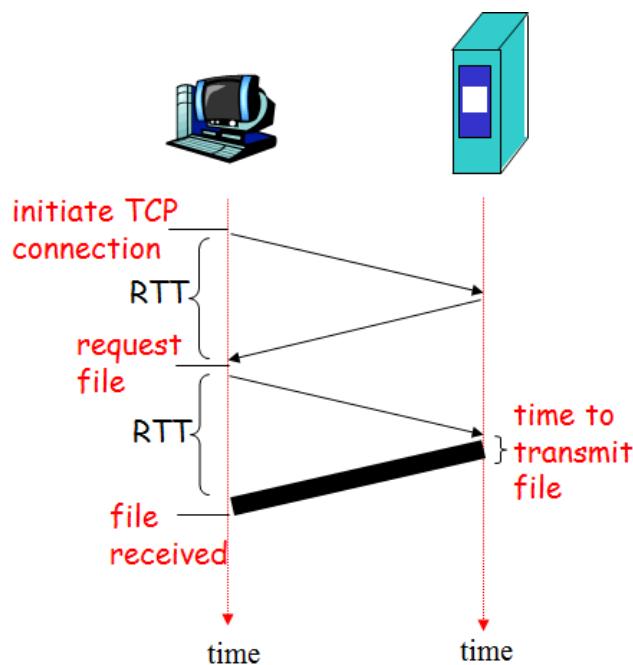
Khi trình duyệt nhận trang Web, nó hiển thị trang Web cho người sử dụng. Hai trình duyệt khác nhau có thể diễn tả (hiển thị cho người sử dụng) một trang Web theo các cách khác nhau. HTTP không thể làm gì với thể hiện trang Web của máy khách. Các đặc tả HTTP (RFC 1945 và RFC 2616) chỉ định nghĩa giao thức truyền thông giữa chương trình HTTP máy khách và chương trình HTTP máy chủ.

Các bước ở trên minh họa việc sử dụng kết nối không liên tục, trong đó mỗi kết nối TCP sẽ bị đóng sau khi máy chủ gửi đối tượng – kết nối này không liên tục cho những đối tượng khác. Lưu ý là mỗi kết nối TCP truyền tải chính xác một bản tin yêu cầu và một bản tin đáp ứng. Vì vậy, trong ví dụ này, khi người sử dụng yêu cầu trang Web thì có 11 kết nối TCP được tạo ra.

Trong những bước mô tả ở trên, ta không rõ về việc liệu máy khách nhận 10 ảnh JPEG qua 10 kết nối TCP nối tiếp, hay một vài ảnh JPEG nhận được qua các kết nối TCP song song. Thực sự là người sử dụng có thể cấu hình các trình duyệt mới để điều khiển mức độ song song này. Trong chế độ mặc định, hầu hết các trình duyệt sẽ mở từ 5 đến 10 kết nối TCP song song, và mỗi kết nối này xử lý một giao dịch yêu cầu – đáp ứng. Nếu người sử dụng muốn thì số lượng kết nối song song tối đa có thể thiết lập bằng một, trong trường

hợp đó 10 kết nối sẽ được thiết lập nối tiếp. Việc sử dụng các kết nối song song sẽ giúp rút ngắn thời gian đáp ứng.

Bây giờ, chúng ta thử tính toán để ước lượng khoảng thời gian từ thời điểm máy khách yêu cầu tệp HTML cơ sở cho tới khi nó nhận được toàn bộ tệp yêu cầu. Chúng ta định nghĩa RTT (round trip time) là thời gian cho một gói tin nhỏ đi từ máy khách tới máy chủ và trở về máy khách. RTT gồm có trễ truyền lan gói tin, trễ hàng đợi ở các bộ định tuyến, chuyển mạch trung gian và trễ xử lý gói tin (xem lại hoạt động của TCP). Ta sẽ xem xét bắt đầu từ lúc người sử dụng nhấp chuột vào một siêu kết nối (một địa chỉ trang Web). Như diễn tả trên hình 2.4, việc nhấp chuột sẽ làm trình duyệt khởi tạo một kết nối TCP giữa trình duyệt và máy chủ Web; nó bao gồm tiến trình bắt tay ba bước – máy khách gửi một đoạn TCP nhỏ tới máy chủ, máy chủ nhận biết và đáp ứng bằng một đoạn TCP nhỏ, và cuối cùng thì máy khách nhận biết trở lại với máy chủ.



Hình 2.4: Tính toán thời gian cần thiết để yêu cầu và nhận tệp HTML

Hai bước đầu trong tiến trình bắt tay ba bước chiếm một RTT. Sau khi hoàn thành hai bước của tiến trình bắt tay ba bước, máy khách sẽ gửi bản tin yêu cầu HTTP kết hợp với bước thứ ba của tiến trình bắt tay (nhận biết) vào kết nối TCP. Một khi bản tin yêu cầu tới

được máy chủ thì máy chủ sẽ gửi tệp HTML vào kết nối TCP. Lần yêu cầu/đáp ứng HTTP này sẽ chiếm một RTT. Vì vậy, sơ bộ tổng thời gian đáp ứng là hai RTT cộng với thời gian truyền dẫn ở máy chủ chứa tệp HTML.

### **Kết nối liên tục:**

Các kết nối không liên tục có một số thiếu sót. Đầu tiên, là phải thiết lập và duy trì kết nối mới cho mỗi đối tượng được yêu cầu. Đối với mỗi một trong những kết nối này, phải cấp phát bộ đệm TCP và phải duy trì các biến TCP trên cả máy khách và máy chủ. Điều này có thể áp đặt tải trọng lên máy chủ Web, do nó có thể phải phục vụ yêu cầu cho hàng trăm máy khách khác nhau đồng thời. Thứ hai, như chúng ta đã thấy mỗi đối tượng sẽ chịu một thời gian trễ chuyển phát khoảng 2 RTT: một RTT để thiết lập kết nối TCP và một RTT để yêu cầu và nhận đối tượng.

Với kết nối liên tục, máy chủ phải để kết nối TCP mở sau khi gửi đáp ứng. Những yêu cầu và đáp ứng liên tiếp giữa cùng một máy khách với máy chủ có thể gửi trên cùng một kết nối. Cụ thể, toàn bộ một trang Web (như ví dụ là tệp HTML cơ sở và 10 hình ảnh) có thể gửi trên duy nhất một kết nối TCP liên tục. Hơn nữa, nhiều trang Web trong cùng một máy chủ có thể được gửi từ máy chủ này tới cùng một máy khách chỉ qua một kết nối TCP liên tục. Các yêu cầu về đối tượng này có thể được thực hiện liên tiếp mà không phải chờ phản hồi giải quyết yêu cầu chính thức (tạo đường ống - pipelining). Thông thường, máy chủ HTTP đóng một kết nối khi nó không được sử dụng trong một khoảng thời gian nhất định (thời gian quá hạn có khả năng thiết lập). Khi máy chủ nhận các yêu cầu liên tiếp, nó cũng gửi các đối tượng liên tiếp. Phương thức mặc định của HTTP là sử dụng kết nối liên tục với đường ống.

#### **2.2.3 Các bản tin HTTP**

Đặc tả HTTP trong RFC 2616 gồm các định nghĩa về khuôn dạng của các bản tin HTTP. Có hai loại bản tin HTTP là bản tin yêu cầu (request) và bản tin đáp ứng (response).

### **Bản tin yêu cầu HTTP:**

Bản tin yêu cầu HTTP có định dạng:

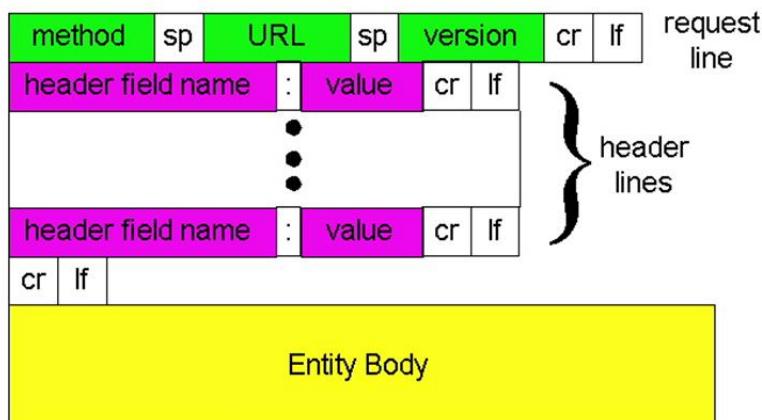
```
GET /somedir/page.html HTTP/1.1  
Host: www.someSchool.edu  
Connection: close  
User-agent: Mozilla/4.0  
Accept-language: fr
```

Chúng ta có thể tìm hiểu được rất nhiều thông qua xem xét chi tiết bản tin yêu cầu đơn giản này. Đầu tiên, ta thấy là bản tin được viết bằng ký tự ASCII thông thường, như vậy các máy tính bình thường có thể đọc được nó. Thứ hai, chúng ta thấy là bản tin gồm 5 dòng, sau mỗi dòng là ký tự xuống dòng và chuyển dòng. Sau dòng cuối cùng cũng vậy. Mặc dù bản tin cụ thể này có 5 dòng, song một bản tin yêu cầu có thể có nhiều dòng hơn hoặc có khi chỉ có một dòng. Dòng đầu tiên của bản tin yêu cầu HTTP được gọi là **dòng yêu cầu**, các dòng tiếp theo là các **dòng tiêu đề**. Dòng yêu cầu có 3 trường: trường phương thức, trường URL, và trường phiên bản HTTP. Trường phương thức có thể lấy một số giá trị khác nhau, bao gồm GET, POST, HEAD, PUT, và DELETE. Các bản tin HTTP chủ yếu sử dụng phương thức GET. Phương thức GET được sử dụng khi trình duyệt yêu cầu một đối tượng, với đối tượng yêu cầu được xác định trong trường URL. Trong ví dụ này trình duyệt yêu cầu đối tượng /somedir/page.html. Phiên bản được chỉ ra rất rõ ràng, trong ví dụ này phiên bản thực hiện trình duyệt là HTTP/1.1.

Hãy nhìn vào dòng tiêu đề trong ví dụ này. Dòng tiêu đề Host: www.someSchool.edu đặc tả trạm chủ chứa đối tượng. Người dùng có thể cho rằng dòng tiêu đề này là không cần thiết vì ở đây đã có kết nối TCP tại trạm chủ. Tuy nhiên, thông tin cung cấp từ dòng tiêu đề trạm chủ (host) được bộ lưu đệm Web (Web proxy cache) yêu cầu (sẽ đưa ra kỹ hơn trong phần 2.5). Bằng việc bao hàm dòng tiêu đề Connection: close, trình duyệt báo cho máy chủ là nó không muốn bận tâm tới các kết nối liên tục, nó muốn máy chủ đóng kết nối sau khi gửi đối tượng yêu cầu. Dòng tiêu đề User-agent:

dòng tiêu đề đặc tả đại lý (agent) người sử dụng, nghĩa là loại trình duyệt thực hiện yêu cầu tới máy chủ. Ở đây đại lý người sử dụng là Mozilla/4.0, trình duyệt của Netscape. Dòng tiêu đề này hữu ích vì máy chủ có thể gửi phiên bản khác của cùng đối tượng tới các loại nhân tố người sử dụng khác nhau (các phiên bản cùng đánh một địa chỉ URL). Cuối cùng là tiêu đề Accept-language: tiêu đề này chỉ rằng người sử dụng mong muốn nhận phiên bản tiếng Pháp của đối tượng nếu như đối tượng như vậy có trên máy chủ, nếu không có máy chủ sẽ gửi phiên bản mặc định. Tiêu đề Accept-language: chỉ là một trong nhiều tiêu đề thỏa thuận nội dung có sẵn trong HTTP.

Tiếp tục ví dụ trên, chúng ta sẽ xem xét khuôn dạng chung của bản tin yêu cầu trong hình 2.5. Ví dụ ở trên tuân thủ khá chặt chẽ theo khuôn dạng này. Tuy nhiên, chúng ta có thể để ý rằng sau các dòng tiêu đề (và kí tự xuống dòng bổ sung và kí tự chuyển dòng) là một “khối thực thể”. Khối thực thể này rỗng với phương thức GET, nhưng nó được sử dụng với phương thức POST. Một máy khách HTTP thường dùng phương thức POST khi người sử dụng điền vào khuôn mẫu (form), ví dụ khi người sử dụng cung cấp từ khoá tìm kiếm cho công cụ tìm kiếm. Với bản tin POST, người sử dụng vẫn yêu cầu trang Web từ máy chủ, nhưng một số nội dung cụ thể của trang Web lại phụ thuộc vào thông tin người sử dụng đưa vào các trường của khuôn mẫu. Nếu giá trị trường phương thức là POST thì khối thực thể chứa toàn bộ thông tin mà người sử dụng đã đưa vào trường khuôn mẫu.



Hình 2.5: Khuôn dạng chung của một bản tin yêu cầu HTTP

Cần chú ý là một yêu cầu được tạo ra với khuôn mẫu có thể không cần sử dụng phương thức POST. Thay vào đó, các khuôn mẫu HTML thường sử dụng phương thức GET và bao hàm dữ liệu đầu vào (trong trường các khuôn mẫu) trên dòng URL được yêu cầu. Ví dụ, nếu khuôn mẫu sử dụng phương thức GET có hai trường, và thông tin đầu vào hai trường là monkeys và bananas thì URL sẽ có cấu trúc

[www.somesite.com/animalsearch?monkeys&bananas](http://www.somesite.com/animalsearch?monkeys&bananas).

Khi lướt Web hàng ngày chúng ta có thể để ý đến kiểu URL mở rộng này.

Phương thức HEAD tương tự với phương thức GET. Khi một máy chủ nhận được yêu cầu với phương thức HEAD, nó đáp ứng bằng bản tin HTTP nhưng bỏ qua đối tượng yêu cầu. Các nhà phát triển ứng dụng thường sử dụng phương thức HEAD để gỡ lỗi (debugging). Phương thức PUT thường được sử dụng kết hợp với các công cụ phát hành Web. Nó cho phép người sử dụng tải lên một đối tượng tới một đường cụ thể (thư mục) trên một máy chủ Web cụ thể. Phương thức PUT cũng được các ứng dụng sử dụng khi cần tải các đối tượng lên các máy chủ Web. Phương thức DELETE cho phép người sử dụng hoặc ứng dụng xoá một đối tượng trên máy chủ Web.

### **Bản tin đáp ứng HTTP:**

Dưới đây là bản tin đáp ứng HTTP điển hình. Đây có thể là bản tin đáp ứng của bản tin yêu cầu trong ví dụ vừa đưa ra ở trên.

```
HTTP/1.1 200 OK
Connection: close
Date: Sat, 07 Jul 2007 12:00:15 GMT
Server: Apache/1.3.0 (Unix)
Last-Modified: Sun, 6 May 2007 09:23:24 GMT
Content-Length: 6821
Content-Type: text/html

(data data data data data ...)
```

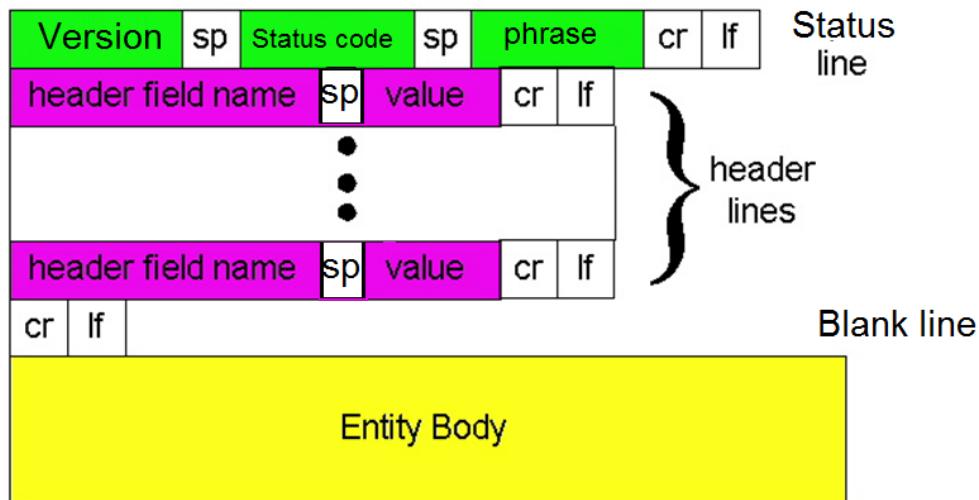
Chúng ta hãy xem xét kỹ bản tin đáp ứng này. Nó có ba phần: **dòng trạng thái** ban đầu, sáu **dòng tiêu đề** và sau cùng là **khối thực thể**. Khối thực thể là thân của bản tin – nó chứa chính đối tượng được yêu cầu (phần data data data data data ...). Dòng trạng thái có 3 trường: trường phiên bản giao thức, mã trạng thái, và bản tin trạng thái tương ứng. Trong ví dụ này dòng trạng thái cho thấy máy chủ sử dụng HTTP/1.1 và mọi thứ là ổn - OK (nghĩa là máy chủ đã tìm thấy, và đang gửi đối tượng yêu cầu).

Bây giờ hãy xem các dòng tiêu đề. Máy chủ sử dụng dòng tiêu đề Connection: close để báo máy khách là nó sẽ đóng kết nối TCP sau khi gửi bản tin. Dòng tiêu đề Date: chỉ thời gian và ngày mà đáp ứng HTTP được máy chủ tạo ra và gửi đi. Lưu ý là đây không phải thời gian tạo ra đối tượng hoặc thay đổi đối tượng lần sau cùng; nó là thời gian khi máy chủ lấy được đối tượng từ hệ thống tệp của nó, chèn đối tượng vào bản tin đáp ứng và gửi bản tin đáp ứng. Dòng tiêu đề Server: chỉ ra là bản tin đã được tạo ra bằng máy chủ Web Apache, nó tương tự như dòng tiêu đề User-agent: trong bản tin yêu cầu HTTP. Dòng tiêu đề Last-Modified: chỉ thời gian và ngày mà đối tượng được tạo ra hay thay đổi sau cùng. Dòng tiêu đề Last-Modified: rất quan trọng để cache (lưu đệm) đối tượng cả ở máy khách cục bộ lẫn ở máy chủ đệm của mạng (còn được gọi là máy chủ proxy). Dòng tiêu đề Content-Length: chỉ số byte của đối tượng được gửi. Dòng tiêu đề Content-Type: chỉ rằng đối tượng ở trong khối thực thể là văn bản HTML. (Kiểu đối tượng thường được chỉ ra chính thức bằng tiêu đề Content-Type: và không ở phần mở rộng tệp).

Bây giờ chúng ta sẽ xem xét khuôn dạng chung của bản tin đáp ứng trong hình 2.4. Khuôn dạng chung này của bản tin đáp ứng phù hợp với bản tin yêu cầu của ví dụ trước. Chúng ta tìm hiểu thêm về các mã trạng thái và mệnh đề của nó. Mã trạng thái và mệnh đề tương ứng chỉ ra kết quả của yêu cầu. Một vài mã trạng thái thông dụng và mệnh đề tương ứng gồm:

- + 200 OK: Yêu cầu thành công và thông tin được trả về trong bản tin đáp ứng.

- + 301 Moved Permanently: Đổi tượng được yêu cầu đã chuyển đi vĩnh viễn; URL mới được xác định trong tiêu đề Location: của bản tin đáp ứng. Phần mềm máy khách sẽ tự động lấy URL mới.
- + 400 Bad Request: Đây là mã lỗi chung chỉ ra là máy chủ không hiểu yêu cầu.
- + 404 Not Found: Tài liệu yêu cầu không tồn tại ở máy chủ này.
- + 505 HTTP Version Not Supported: Phiên bản giao thức HTTP yêu cầu không được máy chủ hỗ trợ.



Hình 2.6: Khuôn dạng chung của một bản tin đáp ứng HTTP

Để có được một bản tin đáp ứng HTTP thực sự, đầu tiên, sử dụng Telnet vào máy chủ Web nào đó. Sau đó gõ một dòng bản tin yêu cầu để tìm đối tượng chứa trong máy chủ đó. Ví dụ, nếu người dùng truy nhập vào dấu nhắc dòng lệnh, gõ:

```
telnet cis.poly.edu 80

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

Nhấn Enter 2 lần sau khi gõ dòng cuối. Các lệnh này sẽ mở kết nối TCP tới cổng 80 của trạm chủ cis.poly.edu và sau đó gửi bản tin yêu cầu HTTP. Bản tin đáp ứng bao gồm tệp HTML cơ sở trang Web của giáo sư Ross sẽ xuất hiện. Nếu chỉ muốn thấy các

dòng bản tin HTTP và không nhận đối tượng thì thay GET bằng HEAD. Cuối cùng, thay /~ross/ bằng /~banana/ và xem bản tin đáp ứng nhận được là gì.

Trong phần này chúng ta thảo luận về một số dòng tiêu đề có thể sử dụng trong bản tin HTTP yêu cầu và đáp ứng. Đặc tả HTTP định nghĩa rất nhiều những dòng tiêu đề mà các trình duyệt, máy chủ Web và các máy chủ đệm mạng có thể chèn vào. Chúng ta chỉ xem xét một số lượng nhỏ trong tổng số các dòng tiêu đề.

Làm thế nào để một trình duyệt quyết định những dòng tiêu đề nào sẽ có trong bản tin yêu cầu? Làm thế nào một máy chủ Web quyết định những dòng tiêu đề nào sẽ có trong bản tin đáp ứng? Một trình duyệt sẽ tạo ra các dòng tiêu đề như một chức năng của loại trình duyệt và phiên bản (ví dụ: trình duyệt HTTP/1.0 sẽ không thể tạo ra dòng tiêu đề phiên bản 1.1), cấu hình của trình duyệt người sử dụng (ví dụ: ngôn ngữ ưa thích) và liệu trình duyệt hiện tại có lưu đệm phiên bản của đối tượng (có thể đã cũ) không. Các máy chủ Web hành động tương tự: Có nhiều sản phẩm khác nhau, phiên bản khác nhau, và cấu hình khác nhau, tất cả những điều này sẽ tác động tới việc những tiêu đề nào sẽ được bao hàm trong bản tin đáp ứng.

#### **2.2.4 Giao thức HTTPS**

HTTPS là giao thức truyền tải siêu văn bản bảo mật. Đây là giao thức được sử dụng để xác thực trang Web được truy cập và bảo vệ quyền riêng tư và tính toàn vẹn của dữ liệu được gửi giữa trình duyệt và trang web. HTTPS thường được sử dụng để bảo vệ các giao dịch trực tuyến có tính bảo mật cao như giao dịch ngân hàng và đặt hàng mua sắm trực tuyến.

Giao thức HTTPS sử dụng giao thức bảo mật tầng giao vận để mã hóa thông tin đó là SSL/TLS. Giao thức bảo mật TLS thực hiện mã hóa một thông điệp HTTP trước khi truyền và giải mã một thông báo khi đến. Nói một cách chính xác, HTTPS không phải là một giao thức riêng biệt, mà để cập đến việc sử dụng HTTP thông thường qua kết nối SSL/TLS được mã hóa. Cả hai giao thức TLS và SSL đều sử dụng hệ thống hạ tầng khóa công khai PKI bất đối xứng. Một hệ thống mật mã bất đối xứng sử dụng hai khóa để mã

hóa thông tin liên lạc, khóa công khai và khóa bí mật. Bất cứ dữ liệu nào được mã hóa bằng khoá công khai (public key) chỉ có thể được giải mã bởi khóa bí mật (private key) và ngược lại.

HTTPS tạo ra một kênh an toàn qua một mạng không an toàn. Điều này đảm bảo sự bảo vệ khỏi những kẻ nghe trộm và các cuộc tấn công ở giữa (MIM), với điều kiện sử dụng các bộ mật mã hóa thích hợp và chứng chỉ máy chủ được xác minh và đáng tin cậy.

### **So sánh HTTPS với HTTP:**

Các URL của HTTPS bắt đầu bằng "https://" và sử dụng cổng 443 theo mặc định, trong khi các URL HTTP bắt đầu bằng "http://" và sử dụng cổng 80 theo mặc định.

HTTP không được mã hóa và do đó dễ bị tấn công MIM và nghe trộm, có thể cho phép những kẻ tấn công có quyền truy cập vào tài khoản trang web và thông tin nhạy cảm, đồng thời sửa đổi trang web để đưa phần mềm độc hại hoặc quảng cáo vào. HTTPS được thiết kế để chống lại các cuộc tấn công như vậy và được coi là an toàn trước chúng.

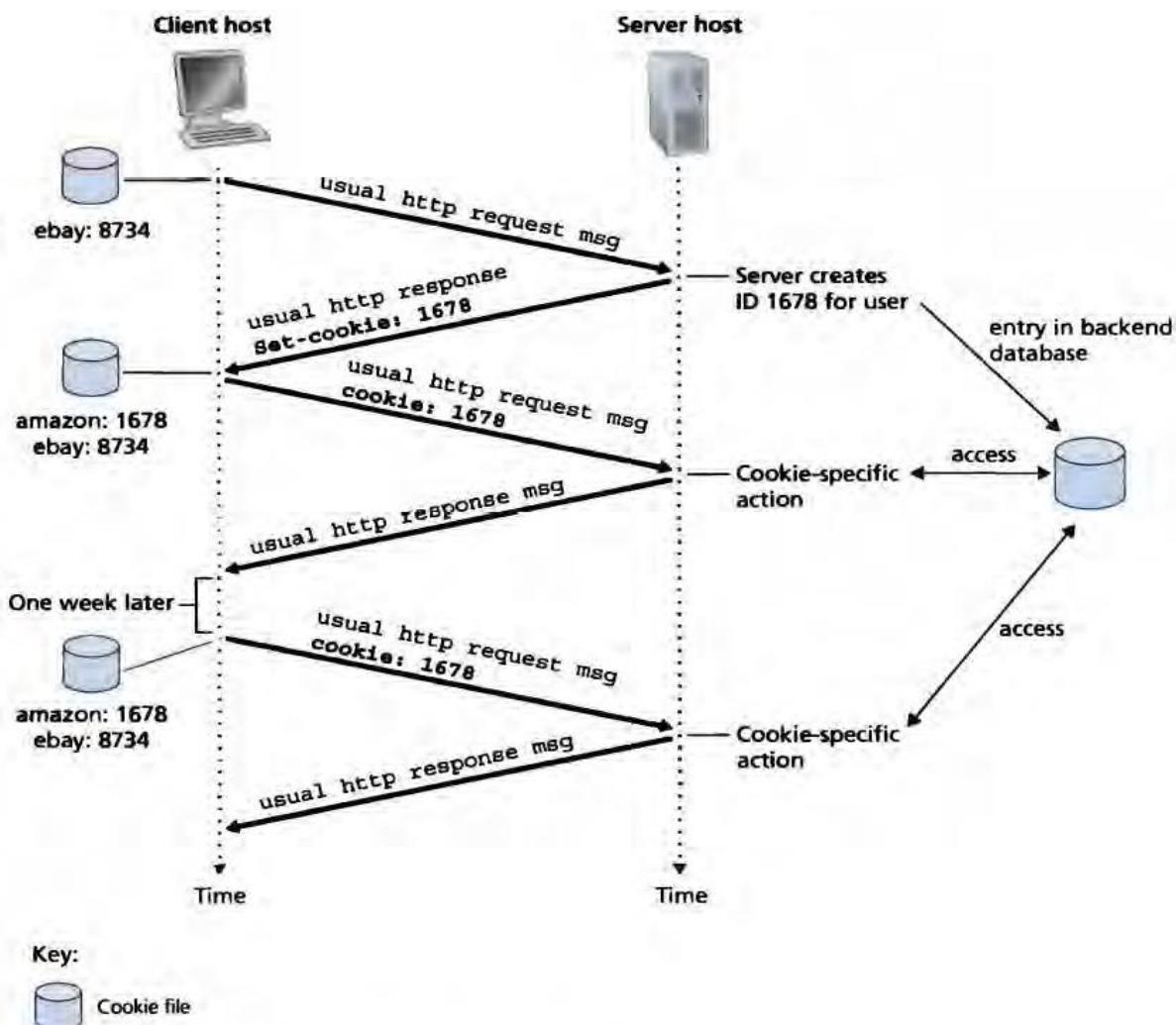
#### **2.2.5 Cookie và lưu đệm WEB**

##### **Cookie:**

Máy chủ HTTP không lưu trạng thái. Điều này làm đơn giản hóa thiết kế và cho phép các kỹ sư phát triển các máy chủ Web hiệu năng cao có thể xử lý đồng thời hàng nghìn kết nối TCP. Tuy nhiên, thường chúng ta mong muốn trang Web nhận dạng được người sử dụng, hoặc bởi vì máy chủ muốn giới hạn truy nhập của người sử dụng, hoặc bởi vì nó muốn dùng nội dung như một hàm nhận dạng người sử dụng. Với những mục đích này, HTTP sử dụng cookie. Các cookie được định nghĩa trong RFC 2965, nó cho phép các điểm truy nhập bám vết người sử dụng. Hầu hết các trang Web thương mại ngày nay đều sử dụng cookie.

Trên hình 2.7 công nghệ cookie có bốn thành phần: (1) dòng tiêu đề cookie trong bản tin đáp ứng HTTP; (2) dòng tiêu đề cookie trong bản tin yêu cầu HTTP; (3) tệp cookie giữ trên hệ thống đầu cuối của người sử dụng và được trình duyệt người sử dụng quản lý; (4)

cơ sở dữ liệu đầu cuối (back-end) tại trang Web. Xem hình 2.5 để thấy hoạt động của cookie. Giả sử Susan, người luôn truy cập Web bằng Internet Explorer từ máy tính ở nhà, kết nối lần đầu tới Amazon.com. Giả sử là trước đây cô ấy đã từng vào trang eBay. Khi yêu cầu tới máy chủ Web Amazon thì máy chủ tạo ra một số nhận dạng duy nhất và tạo ra một mục ở cơ sở dữ liệu đầu cuối sắp xếp theo số nhận dạng. Sau đó máy chủ Web Amazon sẽ đáp ứng tới trình duyệt của Susan, bao gồm tiêu đề Set-cookie: của đáp ứng HTTP, trong đó chứa số nhận dạng. Ví dụ: dòng tiêu đề có thể là Set-cookie: 1678



Hình 2.7: Giữ trạng thái người sử dụng với cookie

Khi trình duyệt của Susan nhận được bản tin đáp ứng HTTP này, nó nhìn vào tiêu đề Set-cookie:. Trình duyệt sẽ thêm một dòng tới tệp cookie đặc biệt mà nó quản lý. Dòng

này gồm tên máy chủ và số nhận dạng trong tiêu đề Set-cookie:. Chú ý là tệp cookie đã có sẵn một mục cho eBay, vì Susan đã vào trang đó trước đây. Khi Susan tiếp tục duyệt trang Amazon, mỗi lần cô ấy yêu cầu một trang Web thì trình duyệt sẽ tham khảo tệp cookie của cô ấy, trích lấy số nhận dạng cho trang này và đưa dòng tiêu đề cookie có chứa số nhận dạng vào yêu cầu HTTP. Đặc biệt, mỗi yêu cầu HTTP của cô ấy tới máy chủ Amazon sẽ có dòng tiêu đề là Cookie: 1678

Theo cách này, máy chủ Amazon có thể bám vết hành động của Susan ở trang Amazon. Mặc dù trang Web Amazon không cần biết tên của Susan, song nó biết chính xác các trang mà người sử dụng 1678 đã ghé thăm và theo thứ tự nào, vào thời điểm nào. Amazon dùng cookie để cung cấp dịch vụ giỏ bán hàng (shopping cart) – Amazon có thể duy trì một danh sách những món hàng mà Susan quan tâm, và vì thế cô ấy có thể trả tiền mua chúng ở cuối phiên.

Nếu Susan trở lại trang Amazon thì một tuần sau trình duyệt của cô sẽ tiếp tục đưa dòng tiêu đề Cookie: 1678 ở các bản tin yêu cầu. Amazon cũng khuyến nghị các sản phẩm với Susan dựa trên các trang Web mà cô ta đã ghé thăm ở Amazon trước đây. Nếu Susan cũng tự mình đăng ký với Amazon – cung cấp đầy đủ tên, địa chỉ email, địa chỉ bưu chính và thông tin thẻ tín dụng thì Amazon có thể có những thông tin này trong cơ sở dữ liệu của mình, và vì thế nó chia sẻ cả tên của Susan với số nhận dạng của cô ấy (và tất cả các trang mà cô ấy đã vào trước đây). Đây là cách mà Amazon và các trang thương mại điện tử khác cung cấp dịch vụ “one-click shopping” (mua bán chỉ qua một cú nhấp chuột) – khi Susan chọn mua bán một vật trong khi ghé thăm, cô không phải gõ lại tên, số thẻ tín dụng hay địa chỉ của mình.

Từ đây ta thấy là cookie dùng để nhận dạng người sử dụng. Khi người sử dụng lần đầu vào một trang thì người sử dụng có thể cung cấp nhận dạng người sử dụng (có thể là tên). Trong những phiên sau đó, trình duyệt sẽ chuyển tiêu đề cookie tới máy chủ, do đó nhận dạng người sử dụng với máy chủ. Do vậy Cookie có thể được dùng để tạo lớp phiên người sử dụng trên đầu của HTTP phi trạng thái. Ví dụ, khi người sử dụng truy cập (log) vào một ứng dụng thư điện tử trên Web (ví dụ Hotmail), trình duyệt sẽ gửi thông tin cookie

tới máy chủ, cho phép máy chủ nhận dạng người sử dụng trong suốt phiên của người sử dụng với ứng dụng đó.

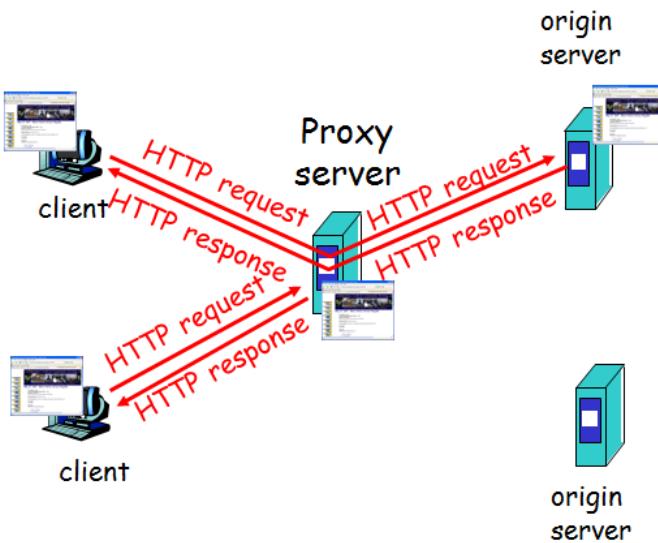
Mặc dù cookie thường làm đơn giản kỹ năng mua sắm trên Internet cho người sử dụng, song chúng gây tranh cãi bởi vì chúng có thể xâm phạm tính riêng tư. Như chúng ta vừa thấy, sử dụng kết hợp các cookie và thông tin tài khoản do người sử dụng cung cấp thì một trang Web có thể biết được rất nhiều về thói quen người sử dụng và có thể bán thông tin này cho bên thứ ba.

### **Lưu đệm WEB:**

Một máy chủ đệm Web (Web Cache) còn được gọi là máy chủ proxy, là một phần tử mạng thoả mãn các yêu cầu HTTP khi đại diện cho máy chủ Web gốc. Máy chủ đệm Web có kho lưu trữ riêng và giữ các bản sao của các đối tượng được yêu cầu gần đây trong kho lưu trữ này. Trong hình 2.8, trình duyệt người sử dụng có thể được cấu hình sao cho tất cả yêu cầu HTTP của người sử dụng đầu tiên sẽ được hướng tới máy chủ đệm Web. Ví dụ, giả sử trình duyệt yêu cầu đối tượng <http://www.someschool.edu/campus.gif>. Sau đây là trình tự những hoạt động sẽ xảy ra:

1. Trình duyệt thiết lập kết nối TCP tới máy chủ đệm Web và gửi yêu cầu HTTP về đối tượng tới máy chủ đệm Web.
2. Máy chủ đệm Web kiểm tra xem liệu có bản sao của đối tượng trong kho lưu trữ của mình không. Nếu có, máy chủ đệm Web sẽ gửi trả đối tượng trong bản tin đáp ứng HTTP tới trình duyệt máy khách.
3. Nếu máy chủ đệm Web không có đối tượng này, nó sẽ mở một kết nối TCP tới máy chủ gốc (origin server), nghĩa là tới [www.someschool.edu](http://www.someschool.edu). Sau đó máy chủ đệm Web sẽ gửi một yêu cầu HTTP về đối tượng vào kết nối TCP từ bộ đệm tới máy chủ. Sau khi nhận được yêu cầu này, máy chủ gốc sẽ gửi đối tượng trong đáp ứng HTTP tới máy chủ đệm Web.

4. Khi máy chủ đệm Web nhận đối tượng này, nó lưu bản sao trong bộ lưu trữ nội bộ của mình và gửi một bản sao trong bản tin đáp ứng HTTP tới trình duyệt khách (trên kết nối TCP sẵn có giữa trình duyệt khách và máy chủ đệm Web).



*Hình 2.8: Máy khách yêu cầu đối tượng qua máy chủ đệm Web*

Chú ý là máy chủ đệm vừa là khách vừa là chủ ở cùng thời điểm. Khi nó nhận yêu cầu từ trình duyệt và gửi lại đáp ứng thì nó là máy chủ. Khi nó gửi yêu cầu và nhận đáp ứng từ máy chủ gốc thì nó là máy khách.

Thông thường, máy chủ đệm Web được ISP mua và cài đặt. Ví dụ, một trường đại học có thể cài đặt máy chủ đệm trên mạng cục bộ của mình và cấu hình toàn bộ trình duyệt trong trường kết nối tới máy chủ đệm đó. Hoặc một nhà cung cấp dịch vụ Internet dân dụng lớn (như VNPT) có thể cài đặt một hoặc nhiều máy chủ đệm trong mạng và cấu hình trước những trình duyệt trỏ tới các máy chủ đệm này.

Đệm Web đã được triển khai trong Internet bởi hai lý do. Thứ nhất, máy chủ đệm Web có thể làm giảm đáng kể thời gian đáp ứng yêu cầu từ máy khách, đặc biệt là khi băng thông nghẽn cỗ chai giữa máy khách và máy chủ gốc nhỏ hơn băng thông nghẽn cỗ chai giữa máy khách và bộ đệm. Nếu có kết nối tốc độ cao giữa máy khách và máy chủ đệm, như thường xảy ra, và nếu máy chủ đệm có đối tượng được yêu cầu, thì nó có khả năng

nhanh chóng chuyển đổi tượng đó tới máy khách. Thứ hai, các máy chủ đệm Web có thể giảm đáng kể lưu lượng trên đường kết nối mạng của một tổ chức tới Internet (xem trong ví dụ sau). Bằng việc giảm bớt lưu lượng, tổ chức (hoặc một công ty hay một trường đại học) không cần phải tăng băng thông quá nhanh, như thế sẽ làm giảm chi phí mua thêm băng thông kết nối. Hơn nữa, các máy chủ đệm Web có thể làm giảm đáng kể lưu lượng Web trên tổng thể mạng Internet, do vậy nó sẽ cải thiện hiệu năng cho tất cả các ứng dụng.

Để hiểu sâu hơn về lợi ích của các máy chủ đệm, hãy xem ví dụ trong hình 2.9. Trong hình có hai mạng: mạng của học viện và mạng Internet công cộng. Mạng trong học viện là mạng LAN tốc độ cao. Một bộ định tuyến (router) trong mạng của học viện và một bộ định tuyến Internet được kết nối thông qua đường 15 Mbit/s. Các máy chủ gốc kết nối vào Internet nhưng được đặt trên các vị trí khắp toàn cầu. Giả sử kích thước đói tượng trung bình là 1 Mbit và tốc độ yêu cầu trung bình từ các trình duyệt trong học viện tới các máy chủ gốc là 15 yêu cầu/giây. Giả sử các bản tin yêu cầu HTTP có kích thước rất nhỏ và sẽ không tạo lưu lượng đáng kể nào trong mạng hay trong các liên kết truy nhập (từ bộ định tuyến của học viện tới bộ định tuyến của Internet). Cũng giả sử là khoảng thời gian từ khi bộ định tuyến phía Internet của kết nối truy nhập (trong hình 2.9) chuyển tiếp một yêu cầu HTTP (trong một gói tin IP) cho tới khi nó nhận được đáp ứng (thường gửi trên nhiều gói tin IP) trung bình là 2 giây. Khoảng thời gian này thường được coi là “trễ Internet”.

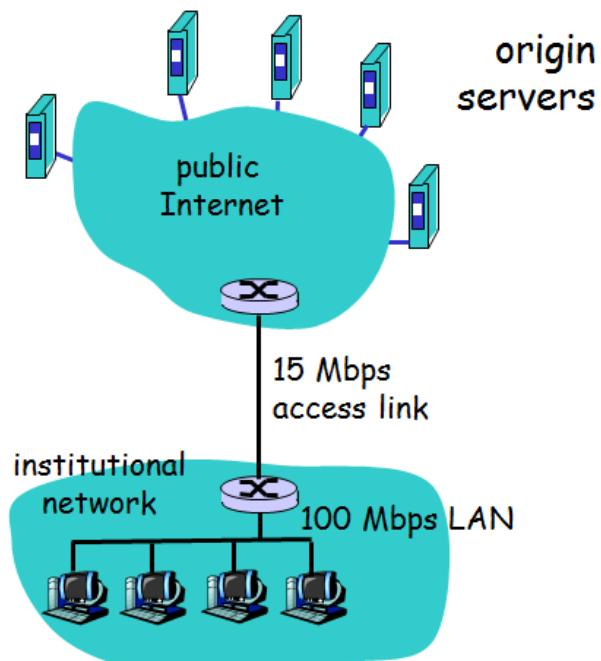
Như vậy, tổng thời gian đáp ứng - khoảng thời gian từ khi trình duyệt yêu cầu đói tượng cho tới lúc trình duyệt nhận được đói tượng đó - là tổng thời gian trễ trong LAN, thời gian trễ truy nhập (trễ giữa 2 bộ định tuyến) và trễ Internet. Hãy tính toán sơ bộ để ước lượng tổng trễ này. Tải trọng lưu lượng trên LAN là:

$$(15 \text{ yêu cầu/giây}) \times (1\text{Mbit/yêu cầu}) / (100\text{Mbit/s}) = 0,15$$

trong đó tải trọng lưu lượng trên liên kết truy nhập (từ bộ định tuyến Internet đến bộ định tuyến của học viện) là:

$$(15 \text{ yêu cầu/giây}) \times (1\text{Mbit/yêu cầu}) / (15\text{Mbit/s}) = 1$$

Tải trọng lưu lượng 0,15 trên mạng LAN thường dẫn đến nhiều nhất là mười giây trễ, vì thế ta có thể bỏ qua trễ của mạng LAN. Tuy nhiên với điều kiện mạng như trong hình 2.7 thì tải trọng lưu lượng là 1 (trong trường hợp kết truy nhập), như vậy trễ trên liên kết trở nên rất lớn và có thể tăng không giới hạn. Vì vậy, thời gian đáp ứng trung bình để thỏa mãn các yêu cầu sẽ lên tới hàng phút và điều đó là không thể chấp nhận với người sử dụng trong mạng học viện. Rõ ràng là phải có biện pháp để xử lý tình huống này.

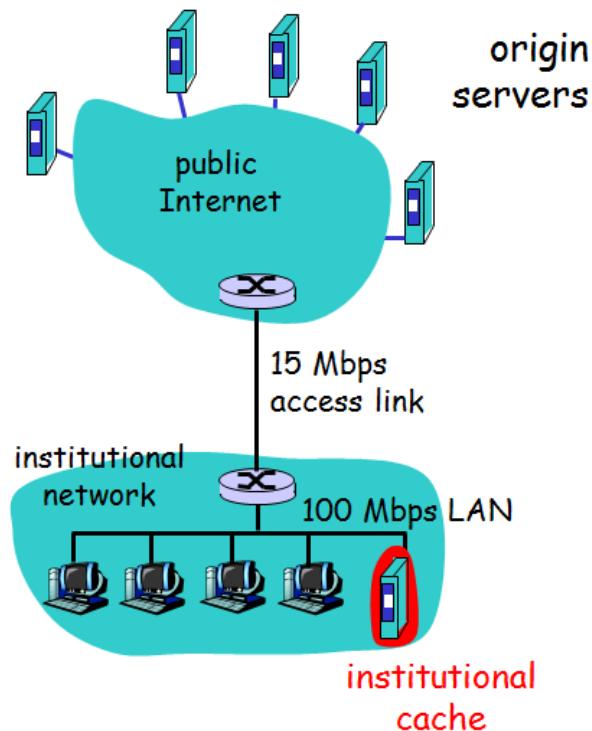


Hình 2.9: Nghẽn cổ chai giữa mạng của học viện và Internet

Một giải pháp khả thi là tăng tốc độ truy nhập từ 15Mbit/s đến 100Mbit/s. Điều này sẽ làm giảm tải trọng lưu lượng trên đường truy nhập xuống còn 0,15 và làm trễ giữa hai bộ định tuyến giảm đáng kể. Trong trường hợp này, tổng thời gian đáp ứng xấp xỉ 2 giây, tức là bằng trễ Internet. Song giải pháp này đồng nghĩa với việc học viện phải nâng cấp đường truy nhập từ 15Mbit/s lên 100Mbit/s, chi phí cho việc này khá tốn kém.

Bây giờ hãy cân nhắc giải pháp thay thế việc nâng cấp đường truy nhập bằng cách cài đặt thêm một máy chủ đệm Web trong mạng của học viện. Giải pháp này được minh họa trong hình 2.10. Tỷ lệ đáp ứng yêu cầu do máy chủ đệm giải quyết được trong thực tế

thường từ 0,2 đến 0,7. Để minh họa cụ thể, giả sử tỉ lệ này là 0,4. Vì các máy khách và máy chủ đệm được kết nối trong cùng mạng LAN tốc độ cao nên 40% yêu cầu có thể đáp ứng ngay lập tức, nghĩa là trong vòng 10 giây thông qua máy chủ đệm Web. Tuy vậy, 60% yêu cầu còn lại vẫn cần máy chủ gốc đáp ứng. Nhưng với chỉ 60% đối tượng yêu cầu phải chuyển qua liên kết truy nhập, tải trọng lưu lượng trên đường truy nhập giảm từ 1,0 xuống còn 0,6. Thông thường, tải trọng lưu lượng nhỏ hơn 0,8 nghĩa là độ trễ tương ứng nhỏ, khoảng 10 miligiây trên liên kết 15Mbit/s. Trễ này là không đáng kể so với trễ 2 giây trên Internet. Như vậy, trễ trung bình là:  $0,4 \times (0,01\text{giây}) + 0,6 \times (2,01\text{ giây}) = 1,21\text{ giây}$ . Giải pháp thứ hai này cho thời gian đáp ứng thấp đáng kể so với giải pháp thứ nhất và nó không yêu cầu học viện phải nâng cấp kết nối tới Internet. Dĩ nhiên là giải pháp này yêu cầu học viện phải mua và cài đặt máy chủ đệm Web song chi phí đó thấp, rất nhiều máy chủ đệm sử dụng các phần mềm tên miền công cộng chạy trên các máy tính cá nhân có chi phí không đắt.



Hình 2.10: Thêm máy chủ đệm vào mạng của học viện

### **Bản tin GET có điều kiện:**

Mặc dù việc đệm Web giảm được thời gian đáp ứng cho người sử dụng nhưng nó lại đặt ra một khó khăn mới, đó là bản sao của các đối tượng nằm trong các máy chủ đệm có thể bị cũ. Nói cách khác, đối tượng ở các máy chủ Web đã thay đổi so với thời điểm mà máy chủ đệm sao chép nó và chuyển tới máy khách. Rất may là HTTP có cơ chế cho phép máy chủ đệm tra cứu việc đối tượng đã được cập nhật hay chưa. Cơ chế này được gọi là cơ chế GET có điều kiện (conditional GET). Một bản tin yêu cầu HTTP được gọi là bản tin conditional GET nếu (1) bản tin yêu cầu sử dụng phương thức GET và (2) bản tin yêu cầu bao gồm một dòng tiêu đề `If-Modified-Since`:

Ví dụ sau minh họa hoạt động của cơ chế này. Đầu tiên, máy chủ đệm proxy đại diện cho trình duyệt yêu cầu, gửi bản tin yêu cầu tới máy chủ Web:

```
GET /fruit/kiwi.gif HTTP/1.1
```

```
Host: www.exotiquecuisine.com
```

Thứ hai, máy chủ Web gửi một bản tin đáp ứng cùng với đối tượng yêu cầu đến máy chủ đệm:

```
HTTP/1.1 200 OK
```

```
Date: Sat, 7 Jul 2007 15:39:29
```

```
Server: Apache/1.3.0 (Unix)
```

```
Last-Modified: Wed, 4 Jul 2007 09:23:24
```

```
Content-Type: image/gif
```

```
(data data data data data ...)
```

Máy chủ đệm chuyển tiếp đối tượng tới trình duyệt yêu cầu nhưng cũng lưu đối tượng lại. Điều quan trọng là máy chủ đệm cũng lưu trữ ngày thay đổi cuối cùng của đối tượng. Thứ ba, một tuần sau, một trình duyệt khác yêu cầu đối tượng đó qua máy chủ đệm, và đối tượng vẫn còn ở trong bộ đệm. Vì đối tượng này có thể đã bị thay đổi ở máy chủ Web trong

tuần trước đó, nên máy chủ đệm thực hiện kiểm tra cập nhật bằng cách tạo ra một bản tin conditional GET. Cụ thể là máy chủ đệm sẽ gửi:

```
GET /fruit/kiwi.gif HTTP/1.1  
Host: www.exotiquecuisine.com  
If-Modified-Since: Wed, 4 Jul 2007 09:23:24
```

Chú ý là giá trị của dòng tiêu đề `If-Modified-Since`: giống hệt như giá trị của dòng tiêu đề `Last-Modified`: do máy chủ gửi một tuần trước đó. Bản tin conditional GET này báo cho máy chủ gửi đối tượng chỉ khi đối tượng đó đã thay đổi từ thời điểm chỉ ra trong bản tin. Giả sử đối tượng không thay đổi từ 9 giờ 23 phút 24 giây ngày 4 tháng 7 năm 2007 thì máy chủ sẽ gửi tới máy chủ đệm bản tin đáp ứng:

```
HTTP/1.1 304 Not Modified  
Date: Sat, 14 Jul 2007 15:39:29  
Server: Apache/1.3.0 (Unix)  
  
(empty entity body)
```

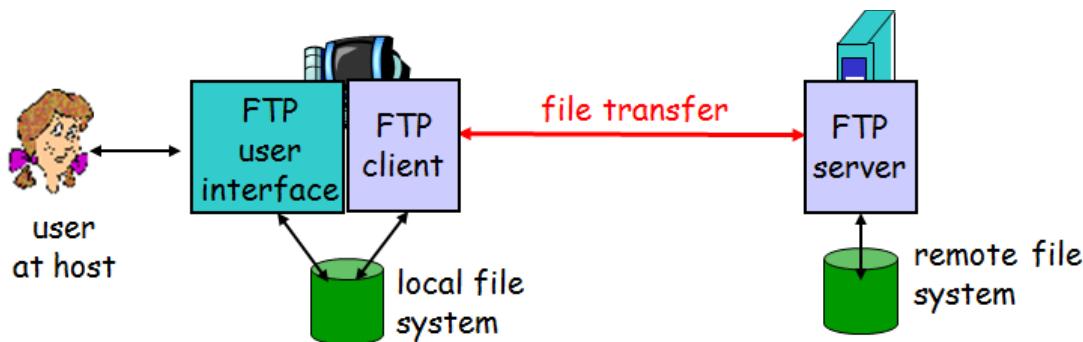
Chúng ta thấy là trong bản tin đáp ứng với conditional GET, máy chủ Web vẫn gửi bản tin đáp ứng nhưng không kèm theo đối tượng yêu cầu trong bản tin đáp ứng. Việc đính kèm đối tượng yêu cầu chỉ làm phí băng thông và làm tăng thời gian đáp ứng yêu cầu của người sử dụng, đặc biệt là khi đối tượng có kích thước lớn. Chú ý là bản tin đáp ứng cuối cùng này có dòng trạng thái là 304 `Not Modified`, điều đó có nghĩa là máy chủ đệm có thể chuyển tiếp bản sao được lưu đệm (của máy chủ đệm proxy) của đối tượng tới trình duyệt yêu cầu.

## 2.3 Ứng dụng truyền tệp và các giao thức

### 2.3.1 Dịch vụ truyền tệp

Một dịch vụ truyền tệp điển hình đó là người sử dụng ngồi trước màn hình trạm chủ (trạm cục bộ) và muốn truyền tệp tới một trạm chủ ở xa hoặc nhận tệp từ một trạm chủ ở

xa. Để truy nhập vào tài khoản ở xa thì người sử dụng phải cung cấp nhận dạng cá nhân và mật khẩu. Sau khi cung cấp thông tin uỷ quyền này, người sử dụng có thể truyền tệp từ hệ thống tệp cục bộ tới hệ thống tệp ở xa và ngược lại. Hình 2.11 minh họa việc người sử dụng tương tác với FTP thông qua đại lý người sử dụng FTP. Đầu tiên người sử dụng cung cấp tên trạm chủ (hostname) của trạm ở xa, buộc tiến trình máy khách FTP trên trạm chủ cục bộ thiết lập một kết nối TCP với tiến trình máy chủ FTP ở đầu xa. Sau đó người sử dụng cung cấp nhận dạng cá nhân và mật khẩu, thông tin này được gửi qua kết nối TCP như một phần của lệnh FTP. Một khi máy chủ đã cho phép người sử dụng thì người sử dụng sẽ sao chép một hoặc nhiều tệp lưu trữ trong hệ thống tệp cục bộ vào hệ thống tệp ở xa (hoặc ngược lại).

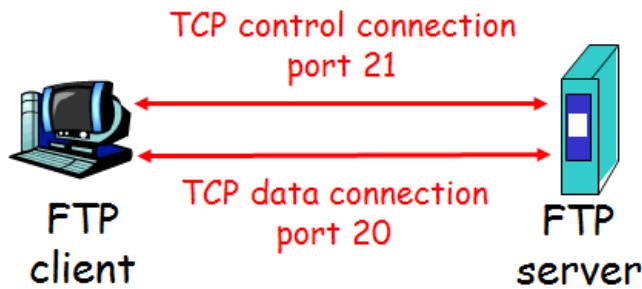


Hình 2.11: FTP chuyển tệp giữa các hệ thống tệp cục bộ và ở xa

### 2.3.2 Giao thức truyền tệp FTP

HTTP và FTP đều là các giao thức truyền tệp và có rất nhiều đặc tính chung, ví dụ: cả hai giao thức đều chạy trên TCP. Tuy nhiên, hai giao thức lớp ứng dụng này có một số khác biệt quan trọng. Điểm khác biệt lớn nhất là FTP sử dụng hai kết nối TCP song song để truyền tệp, một **kết nối điều khiển** và một **kết nối dữ liệu**. *Kết nối điều khiển* được sử dụng để gửi thông tin điều khiển giữa hai trạm chủ như nhận dạng người sử dụng, mật khẩu, lệnh thay đổi thư mục từ xa, lệnh đưa vào (put) hay lấy (get) tệp. *Kết nối dữ liệu* được sử dụng để gửi tệp. Vì FTP sử dụng kết nối điều khiển riêng nên nó được gọi là gửi thông tin điều khiển **ngoài băng (out-of-band)**. Trong chương 6 chúng ta sẽ xem xét giao thức RTSP được sử dụng để truyền dòng phương tiện liên tục như video và audio cũng gửi thông tin

điều khiển ngoài băng. HTTP gửi các dòng tiêu đề yêu cầu và đáp ứng vào cùng một kết nối TCP, kết nối này cũng dùng để chuyển tệp. Vì lý do này, HTTP được gọi là gửi thông tin điều khiển **trong băng (in-band)**. Phản tiếp theo chúng ta sẽ xem xét SMTP, giao thức chủ yếu cho thư điện tử; đây cũng là một giao thức gửi thông tin điều khiển trong băng. Hình 2.12 minh họa các kết nối dữ liệu và điều khiển FTP.



Hình 2.12: Kết nối điều khiển và kết nối dữ liệu

Khi người sử dụng bắt đầu phiên FTP với máy chủ ở xa, bên máy khách FTP (người sử dụng) sẽ khởi tạo kết nối TCP với bên máy chủ (trạm chủ ở xa) trên cổng 21 của máy chủ. Bên máy khách FTP gửi nhận dạng người sử dụng và mật khẩu trên kết nối điều khiển này. Bên máy khách cũng gửi lệnh để thay đổi thư mục từ xa trên kết nối điều khiển. Khi máy chủ nhận được lệnh truyền tệp trên kết nối điều khiển (hoặc tới hoặc từ trạm chủ ở xa) thì máy chủ sẽ khởi tạo một kết nối dữ liệu TCP tới máy khách. FTP sẽ gửi chính xác một tệp qua kết nối dữ liệu và đóng kết nối dữ liệu này. Nếu trong cùng một phiên người sử dụng muốn truyền một tệp khác thì FTP lại mở một kết nối dữ liệu khác. Vì vậy, với FTP kết nối điều khiển duy trì mở suốt phiên của người sử dụng, song cứ mỗi lần truyền một tệp trong phiên thì lại tạo ra một kết nối dữ liệu mới (nghĩa là kết nối dữ liệu là không liên tục).

Trong suốt một phiên, máy chủ FTP cần phải duy trì trạng thái về người sử dụng. Đặc biệt, máy chủ phải liên kết kết nối điều khiển với tài khoản người sử dụng cụ thể và máy chủ phải duy trì theo dõi thư mục hiện thời của người sử dụng khi người sử dụng truy nhập vào cây thư mục từ xa. Duy trì theo dõi thông tin trạng thái này cho từng phiên đang diễn ra của người sử dụng sẽ làm hạn chế đáng kể tổng số phiên mà FTP có thể duy trì đồng

thời. Mặt khác, HTTP là phi trạng thái, nó không duy trì theo dõi bất kỳ trạng thái người sử dụng nào.

### Các lệnh và phản hồi của giao thức FTP:

Hãy xem xét một số lệnh và phản hồi thông dụng của FTP. Các lệnh từ máy khách tới máy chủ và các phản hồi từ máy chủ về máy khách được gửi trên kết nối điều khiển dưới khuôn dạng mã ASCII 7 bit. Vì vậy, cũng giống như lệnh HTTP, mọi người có thể dễ dàng đọc được lệnh FTP. Để phân định các lệnh liên tiếp cần phải xuống dòng và trở về đầu dòng sau mỗi lệnh. Mỗi lệnh gồm bốn ký tự ASCII viết hoa, một số lệnh có các đối số (argument) tuỳ chọn. Một vài lệnh thông dụng được đưa ra như sau:

- + USER username: Dùng để gửi nhận dạng của người sử dụng tới máy chủ.
- + PASS password: Dùng để gửi mật khẩu của người sử dụng tới máy chủ.
- + LIST: Dùng để yêu cầu máy chủ gửi lại danh sách tất cả các tệp trong thư mục ở xa hiện thời. Danh mục tệp này được gửi trên kết nối dữ liệu (kết nối mới và không liên tục) chứ không phải kết nối điều khiển TCP.
- + RETR filename: Dùng để lấy một tệp từ thư mục hiện thời của trạm chủ ở xa. Lệnh này buộc trạm chủ ở xa khởi tạo một kết nối dữ liệu và gửi tệp được yêu cầu qua kết nối dữ liệu này.
- + STOR filename: Dùng để đưa một tệp vào thư mục hiện thời của trạm chủ ở xa.

Thường có một tương quan một-một giữa lệnh người sử dụng đưa ra và lệnh FTP gửi trên kết nối điều khiển. Tiếp theo mỗi lệnh sẽ có một phản hồi gửi từ máy chủ tới máy khách. Những phản hồi này là một số có 3 chữ số, theo sau là một bản tin tuỳ chọn. Cấu trúc này tương tự như mã trạng thái và mệnh đề trong dòng trạng thái của bản tin đáp ứng HTTP. Sau đây là một vài phản hồi điển hình, cùng với nó có thể là các thông điệp:

- + 331 Username OK, password required (tên máy chủ OK, yêu cầu mật khẩu)
- + 125 Data connection already open; transfer starting (kết nối dữ liệu đã mở; bắt đầu truyền)

- + *425 Can't open data connection* (không thể mở kết nối dữ liệu)
- + *452 Error writing file* (lỗi ghi tệp)

Nếu chúng ta quan tâm chi tiết về các lệnh và phản hồi FTP khác, có thể đọc trong RFC 959.

### **2.3.2 Giao thức FTPS**

FTPS (File Transfer Protocol Secure) là giao thức được sử dụng cùng loại trao đổi dữ liệu với FTP. FTPS bổ sung thêm giao thức hỗ trợ bảo mật lớp vận chuyển TLS/SSL để tăng tính bảo mật, hạn chế nhược điểm của FTP.

Cũng giống như FTP, giao thức FTPS cũng sử dụng kênh điều khiển và dữ liệu riêng biệt. Nhưng giao thức tiên tiến này đặc biệt hơn. Đó là sau khi kết nối với máy chủ, máy khách FTPS sẽ tiến hành tra chứng chỉ xem có đáng tin không. Chứng chỉ được coi là đáng tin cậy nếu là 1) được ký bởi cơ quan chứng nhận bên thứ ba có uy tín hoặc 2) nếu nó được ký bởi đối tác và có một bản sao giấy chứng nhận công khai của họ. Nếu đáng tin cậy thì hai bên có thể tiến hành trao đổi tệp. FTPS có thể được gọi qua 2 phương thức:

- + Giao thức FTPS ẩn: kết nối qua cổng 990. Nếu một máy khách kết nối qua cổng 990, máy chủ giả định rằng SSL sẽ được gọi và tự động tìm kiếm thông tin xác thực. Nếu những thông tin này không được cung cấp, kết nối sẽ bị ngắt.
- + Giao thức FTPS hiện: kết nối trên cổng 21. FTPS sẽ yêu cầu máy khách thông báo rõ ràng rằng máy có ý định sử dụng SSL không. Khi máy chủ nhận được lệnh này, nó sẽ tìm thông tin xác thực. FTPS hiện cho thấy khả năng linh hoạt hơn và cho phép máy khách đạt được mức độ bảo mật tăng lên khi cần thiết hoặc tốc độ cao hơn khi bảo mật ít có nguy cơ gặp sự cố.

FTPS có các ưu điểm sau:

- + Được biết đến và sử dụng rộng rãi
- + Giao tiếp có thể được đọc và hiểu bởi một con người
- + Cung cấp dịch vụ chuyển tin từ máy chủ đến máy chủ

- + SSL / TLS có cơ chế xác thực tốt (tính năng chứng chỉ X.509)
- + Hỗ trợ FTP và SSL / TLS được tích hợp vào nhiều khung giao tiếp internet

Bên cạnh đó, một số nhược điểm của FTPS là:

- + Không có định dạng danh sách thư mục thống nhất
- + Yêu cầu kênh DATA thứ cấp, khiến nó khó sử dụng phía sau tường lửa
- + Không xác định tiêu chuẩn cho bộ ký tự tên tệp (mã hóa)
- + Không phải tất cả các máy chủ FTP đều hỗ trợ SSL / TLS
- + Không có cách chuẩn để nhận và thay đổi thuộc tính tệp hoặc thư mục

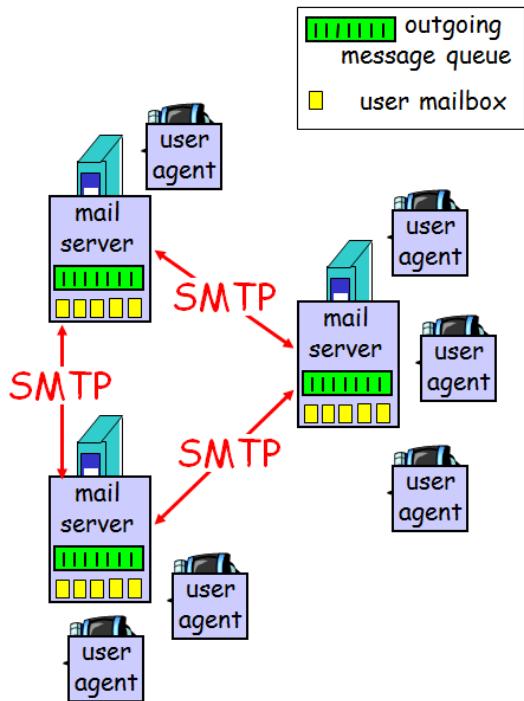
## 2.4 Ứng dụng thư điện tử và các giao thức

### 2.4.1 Thư điện tử trên Internet

Thư điện tử xuất hiện từ khi Internet ra đời. Nó là ứng dụng phổ biến nhất thủa sơ khai của Internet và ngày càng phát triển mạnh mẽ. Cho tới nay, nó vẫn còn là một trong những ứng dụng Internet quan trọng và được sử dụng nhiều nhất.

Cũng như thư tín thông thường (bưu chính), thư điện tử là phương tiện truyền thông không đồng bộ - con người gửi và đọc thư vào thời điểm thuận tiện mà không phụ thuộc vào lịch biểu của người khác. Khác với thư bưu chính, thư điện tử được gửi nhanh hơn, phân phát dễ dàng hơn và có chi phí rẻ. Thư điện tử hiện đại có nhiều đặc tính mạnh mẽ. Sử dụng danh mục thư, thư điện tử và spam (thư rác) có thể gửi tới hàng nghìn người nhận cùng thời điểm. Các thư điện tử hiện đại thường bao gồm các gắn kèm, siêu liên kết, văn bản dạng HTML và ảnh.

Trong phần này chúng ta sẽ tìm hiểu về các giao thức lớp ứng dụng là trái tim của thư điện tử Internet. Nhưng trước khi đi sâu vào các giao thức này, hãy nhìn nhận một cách tổng thể hệ thống thư điện tử Internet và các thành phần chủ chốt của nó.



Hình 2.13: Tổng quan về hệ thống thư điện tử Internet

Hình 2.13 cho thấy bức tranh tổng thể của hệ thống thư điện tử Internet. Chúng ta thấy ở đây có 3 thành phần chính: **đại lý người sử dụng** (user agent), **các máy chủ thư**, và **giao thức truyền thư đơn giản SMTP** (Simple Mail Transfer Protocol). Chúng ta cùng mô tả mỗi thành phần này ở ngữ cảnh người gửi thư – Alice, gửi thư tới người nhận là Bob. Các đại lý người sử dụng cho phép người sử dụng đọc, trả lời, chuyển tiếp, lưu và soạn thảo thư. (Các đại lý người sử dụng cho thư điện tử có khi được gọi là bộ đọc thư –mail reader). Khi Alice viết xong thư, đại lý người sử dụng của cô ấy gửi bản tin tới máy chủ thư của cô ta, và bản tin được đặt ở hàng đợi bản tin đầu ra của máy chủ thư. Khi Bob muốn đọc thư, đại lý người sử dụng của anh ấy lấy thư từ hòm thư của anh ta trên máy chủ thư của anh ấy. Vào cuối những năm 1990, đại lý người sử dụng có giao diện đồ họa người sử dụng (GUI) trở nên phổ biến, nó cho phép người sử dụng xem và soạn bản tin đa phương tiện. Ngày nay, Microsoft's Outlook, Apple Mail và Mozilla Thunderbird là những đại lý người sử dụng GUI thông dụng nhất cho thư điện tử. Ngoài ra cũng có rất nhiều các giao diện người sử dụng thư điện tử dạng văn bản trong miền công cộng, cũng như các giao diện dựa trên Web.

Các máy chủ thư tạo thành cơ sở hạ tầng lõi của thư điện tử. Mỗi người nhận (như Bob) có một hòm thư đặt ở một trong những máy chủ thư. Hòm thư của Bob quản lý và duy trì thư đã gửi cho anh ấy. Một thư thông thường sẽ bắt đầu hành trình của mình từ đại lý người sử dụng của người gửi, đi tới máy chủ thư người gửi, và tới máy chủ thư người nhận, tại đó nó được ký gửi trong hòm thư người nhận. Khi Bob muốn truy cập vào thư trong hòm thư của mình, máy chủ thư chứa hòm thư của anh ta sẽ xác thực Bob (với tên và mật khẩu). Máy chủ thư của Alice cũng phải giải quyết lỗi xảy ra trên máy chủ thư của Bob. Nếu máy chủ thư của Alice không thể chuyển thư tới máy chủ thư của Bob thì máy chủ của Alice sẽ giữ thư trong hàng đợi thư và sau đó cố gắng gửi thư đi. Nỗ lực gửi lại sẽ được thực hiện sau mỗi khoảng 30 phút, nếu không thành công trong vài ngày thì máy chủ sẽ xoá bỏ thư và thông báo bằng thư điện tử tới người gửi (Alice).

SMTP là giao thức lớp ứng dụng chủ yếu của thư điện tử Internet. Nó sử dụng dịch vụ truyền dữ liệu tin cậy của TCP để gửi thư từ máy chủ thư người gửi đến máy chủ thư người nhận. Như hầu hết các giao thức ứng dụng khác, SMTP có hai phía: phía khách – thực hiện trên máy chủ thư người gửi, và phía chủ – thực hiện trên máy chủ thư người nhận. Cả hai bên khách và chủ chạy SMTP trên mỗi máy chủ thư. Khi một máy chủ thư gửi thư tới các máy chủ thư khác thì nó hoạt động như máy khách SMTP. Khi một máy chủ thư nhận thư từ máy chủ thư khác thì nó hoạt động như máy chủ SMTP.

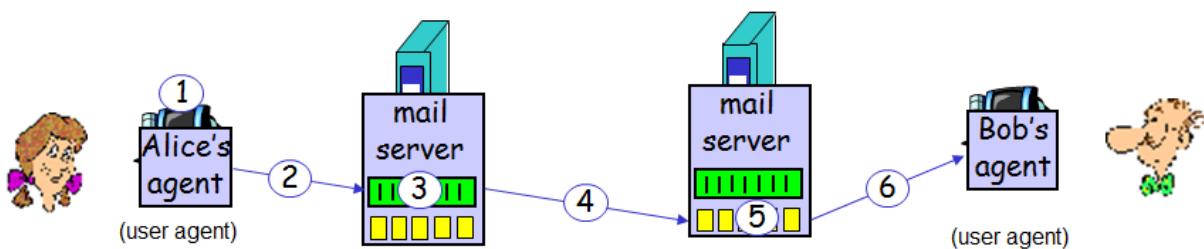
#### **2.4.2 Giao thức truyền thư điện tử đơn giản SMTP**

SMTP là trái tim của thư điện tử Internet (được định nghĩa trong RFC 5321). Như chúng ta đã biết, SMTP chuyển bản tin từ máy chủ thư bên gửi tới máy chủ thư bên nhận. SMTP ra đời trước HTTP (RFC đầu tiên của SMTP có từ năm 1982 và giao thức SMTP đã xuất hiện khá lâu trước đó). Mặc dù SMTP có rất nhiều đặc tính quan trọng, bằng chứng là nó đã tồn tại rộng khắp trên Internet, tuy nhiên nó là công nghệ mang tính kế thừa với nhiều đặc tính xưa cũ. Ví dụ, nó giới hạn toàn bộ bản tin (không chỉ các tiêu đề) phải là chữ và ký tự trong bảng mã ASCII 7 bit. Giới hạn này thích hợp trong những năm đầu thập kỷ 1980 khi dung lượng truyền dẫn còn khan hiếm và không ai gửi thư kèm tệp, hình ảnh,

audio hay video có kích thước lớn. Nhưng ngày nay, trong kỷ nguyên đa phương tiện hạn chế của ASCII 7 bit dẫn đến một số điều không phù hợp: nó yêu cầu dữ liệu đa phương tiện số phải được mã hóa thành ASCII trước khi gửi qua SMTP, và nó yêu cầu bản tin ASCII tương ứng được giải mã trở lại nhị phân sau khi truyền tải trên SMTP. Chúng ta nhớ lại là HTTP không yêu cầu dữ liệu đa phương tiện phải được mã hóa trước khi truyền đi.

Để minh họa hoạt động cơ bản của SMTP, hãy xem xét một trường hợp sau (hình 2.14). Giả sử Alice muốn gửi tới Bob một bản tin mã ASCII đơn giản.

1. Alice gọi đại lý người sử dụng của cô ấy để gửi thư điện tử, gõ vào địa chỉ đích là hộp thư của Bob (ví dụ bob@someschool.edu), viết bản tin và lệnh cho đại lý người sử dụng gửi bản tin này.
2. Đại lý người sử dụng của Alice gửi bản tin tới máy chủ thư của cô ta và bản tin sẽ được đặt trong hàng đợi bản tin.
3. Phía máy khách của SMTP (chạy trên máy chủ thư của Alice) thấy bản tin trong hàng đợi bản tin. Nó mở một kết nối TCP tới một máy chủ thư SMTP (chạy trên máy chủ thư của Bob).
4. Sau quá trình bắt tay SMTP ban đầu, máy khách SMTP gửi bản tin của Alice vào kết nối TCP.
5. Tại máy chủ thư của Bob, phía máy chủ SMTP nhận bản tin. Máy chủ thư của Bob sẽ đặt bản tin này vào hộp thư của Bob.
6. Bob gọi đại lý người sử dụng của anh ấy để đọc bản tin khi thuận tiện.



Hình 2.14: Quá trình hai người gửi thư cho nhau

Một điều quan trọng cần lưu ý là SMTP thường không sử dụng các máy chủ thư trung gian để gửi bản tin, ngay cả khi hai máy chủ thư ở hai đầu trái đất. Nếu máy chủ thư của Alice ở Việt Nam và máy chủ thư của Bob ở St. Louis thì kết nối TCP là kết nối trực tiếp giữa máy chủ Việt Nam và St. Louis. Đặc biệt, nếu máy chủ của Bob mà hỏng thì bản tin sẽ ở lại trong máy chủ thư của Alice và chờ để cố gắng gửi tiếp – bản tin sẽ không lưu ở bất cứ máy chủ thư trung gian nào.

Bây giờ chúng ta sẽ đi vào chi tiết phương thức SMTP truyền bản tin từ máy chủ thư gửi đến máy chủ thư nhận. Chúng ta sẽ thấy giao thức SMTP có rất nhiều điểm tương đồng với các giao thức được dùng trong tương tác mặt đối mặt của con người. Thứ nhất, máy khách SMTP (chạy trên trạm chủ thư gửi) thiết lập kết nối TCP tới cổng 25 ở máy chủ SMTP (chạy trên trạm chủ thư nhận). Nếu như máy chủ thư hỏng thì máy khách sẽ lại cố gửi lại sau đó. Một khi kết nối được thiết lập, máy chủ và máy khách thực hiện một vài bước bắt tay lớp ứng dụng – giống như con người thường giới thiệu nhau trước khi truyền thông tin cho nhau, các máy khách và máy chủ SMTP giới thiệu nhau trước khi truyền thông tin. Trong pha bắt tay SMTP này, máy khách SMTP chỉ ra địa chỉ email của người gửi (người tạo ra bản tin) và địa chỉ email của người nhận. Một khi máy khách và máy chủ SMTP đã giới thiệu nhau xong thì máy khách sẽ gửi bản tin. SMTP có thể dựa vào dịch vụ truyền dữ liệu tin cậy của TCP để lấy bản tin từ máy chủ mà không bị lỗi. Sau đó máy khách sẽ lặp lại tiến trình này trên cùng kết nối TCP đó nếu nó có những bản tin khác gửi tới máy chủ, nếu không thì nó sẽ lệnh cho TCP đóng kết nối này.

Hãy xem xét ví dụ của các bản tin trao đổi giữa máy khách SMTP (C) và máy chủ thư SMTP (S). Tên của máy khách là `crepes.fr` và tên máy chủ là `hamburger.edu`. Các dòng văn bản ASCII bắt đầu với C : chính là các dòng máy khách gửi vào socket TCP và dòng văn bản ASCII bắt đầu với S : chính là các dòng máy chủ gửi vào socket TCP. Đoạn对话 sau bắt đầu ngay khi kết nối TCP được thiết lập.

```
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 hambuger.edu closing connection
```

Trong ví dụ trên máy khách gửi một bản tin (“Do you like ketchup? How about pickles?”) từ máy chủ thư crepes.fr tới máy chủ thư hamburger.edu. Là một phần của đoạn hội thoại, máy khách đưa ra 5 lệnh: HELO (từ viết tắt của HELLO), MAIL FROM, RCPT TO, DATA và QUIT. Máy khách cũng gửi dòng chứa dấu chấm câu đơn lẻ, chỉ thị kết thúc một bản tin tới máy chủ. (Trong thuật ngữ ASCII, mỗi bản tin kết thúc bằng CRLF. CRLF, trong đó CR và LF là xuống dòng và về đầu dòng – nghĩa là dòng có một dấu chấm). Máy chủ đưa ra trả lời từng lệnh, mỗi lệnh có một mã trả lời và một vài dòng giải thích bằng tiếng Anh (tùy chọn). Chúng ta để ý ở đây là SMTP sử dụng kết nối liên tục: Nếu máy chủ thư bên gửi có vài bản tin gửi tới cùng máy chủ thư bên nhận

thì nó có thể gửi toàn bộ bản tin trên cùng kết nối TCP đó. Với mỗi bản tin, máy khách bắt đầu tiến trình với một MAIL FROM: crepes.fr, chỉ định kết thúc bản tin với một dấu chấm câu độc lập, và đưa ra QUIT chỉ sau khi toàn bộ bản tin đã được gửi.

Người dùng có thể sử dụng Telnet để thực hiện một đoạn giao tiếp trực tiếp với máy chủ SMTP, để làm điều này thực hiện:

```
telnet serverName 25
```

trong đó serverName là tên của máy chủ thư nội bộ. Khi làm điều này, người dùng đang thiết lập một kết nối TCP giữa trạm chủ cục bộ và máy chủ thư. Sau khi gõ dòng lệnh này, người dùng sẽ ngay lập tức nhận được bản tin “220” trả lời từ máy chủ. Sau đó đưa ra các lệnh SMTP HELO, MAIL FROM, RCPT TO, DATA, CRLF.CRLF và QUIT vào những thời điểm tương ứng.

Hãy so sánh sơ bộ SMTP và HTTP. Cả hai giao thức đều được dùng để truyền tệp từ một trạm chủ tới một trạm chủ khác: HTTP truyền tệp (còn được gọi là đối tượng) từ máy chủ Web tới máy khách Web (thường là trình duyệt); SMTP truyền tệp (bản tin thư điện tử) từ một máy chủ thư tới một máy chủ thư khác. Khi truyền tệp, cả HTTP và SMTP liên tục đều sử dụng các kết nối liên tục. Vì vậy hai giao thức này có nhiều điểm chung. Tuy nhiên, cũng có những khác biệt quan trọng. Thứ nhất là HTTP chủ yếu là giao thức kéo (pull) – người nào đó tải thông tin lên máy chủ Web và người sử dụng dùng HTTP để kéo thông tin đó từ máy chủ khi thuận tiện. Cụ thể, kết nối TCP được khởi tạo bằng máy chủ muốn nhận tệp. Mặt khác, SMTP bản chất là giao thức đẩy (push), máy chủ thư gửi đẩy tệp tới máy chủ thư nhận. Cụ thể, kết nối TCP được khởi tạo bằng máy chủ muốn gửi tệp.

Khác biệt thứ hai là SMTP yêu cầu mỗi bản tin, gồm toàn bộ thân bản tin, phải có khuôn dạng ASCII 7 bít. Nếu bản tin này chứa các ký tự không phải ASCII 7 bít (ví dụ: tiếng Việt có dấu) hoặc chứa dữ liệu nhị phân (như tệp ảnh) thì bản tin phải được mã hoá thành ASCII 7 bít. Dữ liệu HTTP không áp đặt hạn chế này.

Điểm khác biệt quan trọng thứ ba liên quan đến việc xử lý một tài liệu chứa cả văn bản và ảnh (hoặc với một loại phương tiện nào khác) như thế nào. Như đã biết, HTTP đóng gói đối tượng trong mỗi bản tin đáp ứng HTTP của nó. Thư điện tử Internet đưa toàn bộ đối tượng của thư vào trong một bản tin.

#### 2.4.3 Khuôn dạng bản tin thư điện tử

Khi Alice viết bức thư tay gửi Bob, cô ấy có thể bao hàm toàn bộ các thẻ loại thông tin tiêu đề bên ngoài ở đầu thư, như địa chỉ của Bob, địa chỉ trả về của cô ấy và ngày tháng. Tương tự, khi bản tin thư điện tử được gửi từ người này tới người khác thì tiêu đề chứa thông tin sẽ đi trước phần nội dung tin. Những thông tin bên ngoài này được chứa trong các dòng tiêu đề, chúng được định nghĩa trong RFC 5322. Các dòng tiêu đề và thân của bản tin được tách biệt bằng một dòng trống (tức là bằng CRLF). RFC 5322 đặc tả khuôn dạng chính xác của các dòng tiêu đề thư cũng như những biến dịch ngữ nghĩa. Cùng với HTTP, mỗi dòng tiêu đề chứa văn bản đọc được, bao gồm một từ khoá, tiếp theo sau là một giá trị và dấu hai chấm. Một vài từ khoá là bắt buộc, còn lại là tùy chọn. Mỗi tiêu đề phải có dòng tiêu đề `From:` và dòng tiêu đề `To:`, tiêu đề có thể có dòng tiêu đề `Subject:` cũng như các dòng tiêu đề tùy chọn khác. Cần lưu ý là những dòng tiêu đề này khác với lệnh SMTP mà chúng ta nghiên cứu ở phần 3.2.1 (mặc dù chúng chứa một vài từ giống nhau như “from” và “to”). Lệnh trong phần trước là một phần của giao thức bắt tay SMTP, còn dòng tiêu đề xem xét ở đây là một phần trong bản tin thư.

Một tiêu đề tin điển hình giống như thế này:

```
From: alice@crepes.fr  
To: bob@hamburger.edu  
Subject: Searching for the meaning of life.
```

Sau tiêu đề bản tin là một dòng trống, sau đó là thân bản tin (dạng ASCII). Ta có thể dùng Telnet để gửi bản tin từ máy chủ thư chứa một vài dòng tiêu đề, bao gồm dòng tiêu

đề `Subject`: Để làm điều đó, dùng lệnh `telnet serverName 25` như mô tả ở phần trên.

#### 2.4.4 Các giao thức truy cập thư điện tử

Một khi SMTP chuyển xong thư từ máy chủ thư của Alice đến máy chủ thư của Bob thì thư được đặt trong hòm thư của Bob. Chúng ta ngầm định là Bob đọc thư bằng cách đăng nhập vào máy chủ thư và thực hiện việc đọc thư trên máy chủ đó. Cho tới tận đầu thập kỷ 1990 đây là cách thức tiêu chuẩn để thực hiện điều này. Ngày nay, việc truy nhập thư sử dụng kiến trúc khách-chủ: người sử dụng thường đọc thư điện tử bằng máy khách thực hiện ở hệ thống cuối phía người sử dụng, ví dụ như là máy tính ở công ty, máy tính xách tay hay là PDA. Bằng cách thực hiện vai trò máy khách thư trên máy tính nội bộ, người sử dụng có được rất nhiều đặc tính, bao gồm khả năng xem các bản tin đa phương tiện và các tệp đính kèm.

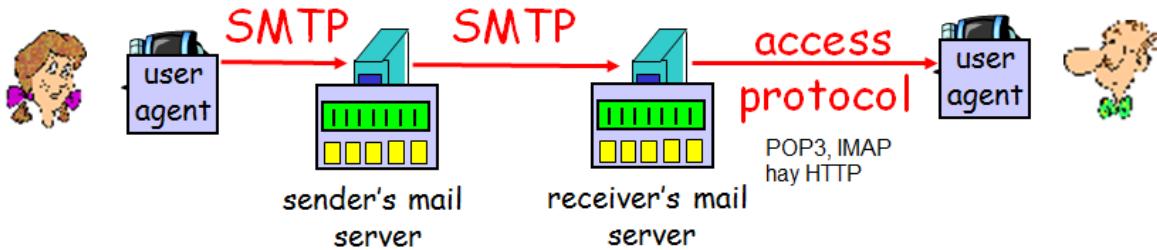
Giả sử Bob (người nhận) chạy chương trình đại lý người sử dụng trên máy tính nội bộ của mình, theo lẽ tự nhiên có thể xem xét đặt máy chủ thư tại máy tính của anh ấy. Với giải pháp này, máy chủ thư của Alice sẽ đổi thoại trực tiếp với máy tính của Bob. Tuy nhiên giải pháp này cũng có các khó khăn. Chúng ta nhớ lại là một máy chủ thư quản lý các hòm thư và hoạt động ở cả hai phía khách và chủ của SMTP. Nếu máy chủ thư của Bob nằm trên máy tính nội bộ của anh ấy thì máy tính đó sẽ phải luôn ở trạng thái hoạt động và phải kết nối vào Internet để nhận được những thư mới tới vào bất kỳ thời điểm nào. Điều này là không thực tế với nhiều người sử dụng Internet. Thay vào đó, một người sử dụng thường chạy chương trình đại lý người sử dụng trên máy tính nội bộ song lại truy nhập vào hòm thư của mình lưu trên một máy chủ thư chia sẻ luôn hoạt động. Máy chủ thư này được rất nhiều người sử dụng khác chia sẻ và thường được một ISP của người sử dụng duy trì (ví dụ: ISP là một trường đại học hoặc một công ty).

Bây giờ hãy xét đường đi của thư điện tử khi gửi từ Alice đến Bob. Chúng ta mới học được là ở một điểm nào đó trên đường đi, thư cần phải được gửi vào máy chủ thư của Bob. Điều này có thể thực hiện dễ dàng bằng cách đại lý người sử dụng của Alice gửi thư trực

tiếp tới máy chủ thư của Bob. Và thực sự điều này có thể thực hiện được với SMTP, nó được thiết kế để đẩy thư điện tử từ trạm chủ này tới trạm chủ khác. Tuy nhiên, thông thường đại lý người sử dụng của người gửi không đàm thoại trực tiếp với máy chủ thư của người nhận. Thay vào đó, như đưa ra trong hình 2.15, đại lý người sử dụng của Alice sử dụng SMTP để đẩy thư vào máy chủ thư của cô ấy, sau đó máy chủ thư của Alice sử dụng SMTP (như một máy khách SMTP) chuyển tiếp thư này vào máy chủ thư của Bob. Tại sao lại là thủ tục hai bước này? Chủ yếu bởi vì nếu không có sự chuyển tiếp qua máy chủ thư của Alice thì đại lý người sử dụng của Alice không có bất cứ cách nào truy cập đến máy chủ thư đích đang mất liên hệ. Bằng cách đầu tiên gửi e-mail của Alice vào chính máy chủ thư của cô ấy, máy chủ thư của Alice có thể lặp lại việc gửi bản tin tới máy chủ thư của Bob (sau mỗi khoảng 30 phút) cho tới khi máy chủ thư của Bob hoạt động trở lại. (Và nếu máy chủ thư của Alice bị hỏng thì cô ấy có thể phàn nàn với nhà quản trị hệ thống của cô ấy). RFC SMTP định nghĩa cách dùng các lệnh SMTP để chuyển tiếp thư qua nhiều máy chủ SMTP.

Tuy nhiên vẫn còn vấn đề chưa ổn ở đây. Làm thế nào để người nhận như Bob với đại lý người sử dụng trên máy tính cá nhân của anh ấy sẽ nhận được thư này trong khi thư lại nằm trong máy chủ thư thuộc ISP của Bob? Chú ý là đại lý người sử dụng của Bob không thể dùng SMTP để lấy thư vì lấy thư là hoạt động kéo (pull), trong khi SMTP là giao thức đẩy (push). Vấn đề này được giải quyết bằng cách đưa ra các giao thức truy nhập thư đặc biệt để truyền thư từ máy chủ thư của Bob đến máy tính cá nhân của anh ấy. Hiện nay có khá nhiều giao thức truy nhập thư thông dụng, bao gồm POP3 (Post Office Protocol Version 3), IMAP (Internet Mail Access Protocol) và HTTP.

Hình 2.15 đưa ra tổng kết về các giao thức sử dụng cho thư Internet: SMTP được dùng để truyền thư từ máy chủ thư người gửi đến máy chủ thư người nhận; SMTP cũng được dùng để truyền thư từ đại lý người sử dụng bên gửi tới máy chủ thư người gửi. Một giao thức truy nhập thư, ví dụ POP3, được dùng để truyền thư từ máy chủ thư người nhận tới đại lý người sử dụng của người nhận.



Hình 2.15: Các giao thức thư điện tử và những thực thể truyền thông của chúng

### Giao thức POP3:

POP3 là giao thức truy nhập thư rất đơn giản. Nó được định nghĩa trong RFC 1939, nó khá ngắn gọn và dễ đọc. Vì giao thức này rất đơn giản nên chức năng của nó khá hạn chế. POP3 bắt đầu khi đại lý người sử dụng (máy khách) mở một kết nối TCP tới máy chủ thư (máy chủ) trên cổng 110. Với kết nối vừa thiết lập này, POP3 tiến triển qua ba pha: uỷ quyền, giao dịch và cập nhật. Trong pha đầu tiên - pha uỷ quyền - đại lý người sử dụng gửi tên và mật khẩu (rõ ràng) để xác thực người sử dụng. Trong pha thứ hai - pha giao dịch - đại lý người sử dụng lấy các bản tin, cũng trong pha này đại lý người sử dụng có thể đánh dấu các bản tin để xoá, bỏ đánh dấu xoá và lấy thông kê thư. Pha thứ ba - pha cập nhật – diễn ra sau khi máy khách đã thực hiện lệnh `quit`, kết thúc phiên POP3; ở thời điểm này máy chủ thư xoá các bản tin đã bị đánh dấu để xoá.

Trong một phiên giao dịch POP3, đại lý người sử dụng đưa ra các lệnh và máy chủ đáp ứng từng lệnh bằng bản tin trả lời. Có hai khả năng đáp ứng: `+OK` (đôi khi còn kèm theo dữ liệu từ máy chủ tới máy khách), do máy chủ sử dụng để chỉ ra là lệnh trước đó là tốt và `-ERR` do máy chủ sử dụng để chỉ ra là lệnh trước đó bị lỗi.

Pha uỷ quyền có hai lệnh cơ bản: `user <username>` và `pass <password>`. Để minh họa hai lệnh này, hãy thử Telnet trực tiếp vào máy chủ POP3, sử dụng cổng 110 và đưa ra các lệnh này. Giả sử `mailServer` là tên của máy chủ thư của người dùng. Người dùng sẽ thấy như sau:

```
telnet mailServer 110
```

```
+OK POP3 server ready  
user bob  
+OK  
pass hungry  
+OK user successfully logged on
```

Bây giờ hãy nhìn vào pha giao dịch. Một đại lý người sử dụng POP3 có thể được người sử dụng cấu hình để “tải về và xoá” hoặc “tải về và lưu giữ”. Trình tự lệnh do đại lý người sử dụng POP3 đưa ra phụ thuộc vào việc đại lý người sử dụng cấu hình theo phương thức hoạt động nào. Trong phương thức tải về-và-xoá, đại lý người sử dụng sẽ đưa ra các lệnh `list`, `retr` và `dele`. Ví dụ, giả sử người sử dụng có hai bản tin trong hộp thư của mình. Trong đoạn hội thoại sau, **C:** (máy khách) là đại lý người sử dụng và **S:** (máy chủ) là máy chủ thư. Giao dịch sẽ như sau:

```
C: list  
  
S: 1 498  
  
S: 2 912  
  
S: .  
  
C: retr 1  
  
S: (blah blah ...  
S: .....  
S: .....blah)  
  
S: .  
  
C: dele 1  
  
C: retr 2
```

```
S: (blah blah ...  
S: .....  
S: .....blah)  
S: .  
C: dele 2  
C: quit  
S: +OK POP3 server signing off
```

Đầu tiên, đại lý người sử dụng sẽ yêu cầu máy chủ thư liệt kê kích thước của mỗi bản tin lưu trữ. Sau đó đại lý người sử dụng sẽ lấy và xoá từng bản tin khỏi máy chủ. Chú ý là sau pha uỷ quyền, đại lý người sử dụng chỉ dùng bốn lệnh: list, retr, dele và quit. Cú pháp của các lệnh này được định nghĩa trong RFC 1939. Sau khi xử lý lệnh quit, máy chủ POP3 đi vào pha cập nhật và xoá bỏ các bản tin 1 và 2 khỏi hòm thư.

Có một khó khăn với phương thức tải về-và-xoá đó là người nhận (Bob) có thể nay đây mai đó và muốn truy cập bản tin từ nhiều máy tính khác nhau như máy để bàn ở cơ quan, máy tính ở nhà hoặc máy tính xách tay của anh ấy. Phương thức tải về-và-xoá này sẽ phân chia những bản tin thư của Bob trên 3 máy này, cụ thể là nếu đầu tiên Bob đọc một bản tin trên máy tính cơ quan thì buổi tối khi về nhà anh ấy sẽ không thể đọc lại bản tin này từ máy xách tay của mình. Trong phương thức tải về-và-giữ, đại lý người sử dụng vẫn giữ bản tin trên máy chủ thư sau khi tải chúng về. Trong trường hợp này, Bob có thể đọc lại bản tin từ các máy tính khác nhau, anh ấy có thể truy nhập một bản tin từ cơ quan và truy nhập chính bản tin đó ở nhà sau đó một tuần.

Trong phiên POP3 giữa đại lý người sử dụng và máy chủ thư, máy chủ POP3 sẽ duy trì một vài thông tin trạng thái; cụ thể là nó theo dõi bản tin người sử dụng nào đã được đánh dấu xoá. Tuy nhiên, máy chủ POP3 không mang thông tin trạng thái xuyên qua các

phiên POP3. Thiếu thông tin trạng thái này qua các phiên làm đơn giản hoá việc thực hiện máy chủ POP3.

### **Giao thức IMAP:**

Với truy nhập POP3, một khi Bob đã tải các bản tin về máy tính nội bộ, anh ấy có thể tạo các thư mục thư và chuyển các bản tin tải về vào các thư mục đó. Sau đó Bob có thể xoá bản tin, di chuyển bản tin vào các thư mục và tìm kiếm các bản tin (qua tên người gửi hoặc chủ đề thư). Nhưng mô hình này (các thư mục và bản tin trong máy nội bộ) sẽ đặt ra một khó khăn cho người sử dụng hay di chuyển, khi họ mong muốn duy trì phân cấp thư mục trên máy chủ ở xa và có thể truy nhập từ bất kỳ máy tính nào. Điều này là không thể với POP3 – giao thức POP3 không cung cấp công cụ nào cho người sử dụng để tạo thư mục từ xa và sắp xếp bản tin vào các thư mục.

Để giải quyết khó khăn này và những vấn đề khác nữa, giao thức IMAP (RFC 3501) đã được tạo lập. Cũng giống POP3, IMAP là giao thức truy nhập thư. Nó có nhiều đặc tính hơn so với POP3 và tất nhiên là nó cũng phức tạp hơn nhiều. (Và vì thế việc thực thi ở phía máy khách và máy chủ cũng phức tạp hơn nhiều).

Một máy chủ IMAP sẽ liên kết mỗi bản tin với một thư mục, khi bản tin lần đầu đến máy chủ, nó được gắn với thư mục INBOX của người nhận. Sau đó người nhận có thể chuyển bản tin này vào một thư mục mới do người sử dụng tạo ra, đọc bản tin, xoá bản tin... Giao thức IMAP cung cấp các lệnh để cho người sử dụng tạo thư mục và chuyển bản tin từ thư mục này tới thư mục khác. IMAP cũng cung cấp các lệnh cho phép người sử dụng tìm kiếm thư trong thư mục từ xa theo những tiêu chí riêng. Chú ý là khác với POP3, máy chủ IMAP duy trì thông tin trạng thái người sử dụng xuyên suốt các phiên IMAP – ví dụ, tên của các thư mục và những bản tin nào được liên kết với những thư mục nào.

Một đặc tính quan trọng khác của IMAP là nó có những lệnh cho phép đại lý người sử dụng lấy các thành phần của bản tin. Ví dụ, đại lý người sử dụng có thể chỉ rút ra tiêu đề bản tin của bản tin hay chỉ một phần của bản tin MIME nhiều phần dữ liệu. Đặc tính này rất hữu dụng khi chỉ có kết nối băng thông thấp (ví dụ: kết nối modem tốc độ thấp) giữa đại

lý người sử dụng và máy chủ thư của nó. Với kết nối băng thông thấp, người sử dụng có thể không muốn tải về toàn bộ bản tin trong hòm thư, đặc biệt là tránh những thư kích thước lớn có thể chứa đoạn audio và video.

### **Thư điện tử dựa trên Web (Web-Based E-mail)**

Tháng 12 năm 1995, chỉ vài năm sau khi Web được phát minh, Sabeer Bhatia và Jack Smith đã tới thăm nhà tư bản kinh doanh Internet Draper Fisher Jurvetson và đề xuất phát triển một hệ thống thư điện tử trên Web miễn phí. Ý tưởng này cấp tài khoản thư điện tử miễn phí cho bất kỳ người nào mong muốn, và tạo ra các tài khoản truy cập được từ Web. Để trao đổi lấy 15% công ty, Draper Fisher Jurvetson đã tài trợ cho Bhatia và Smit để gây dựng nên công ty Hotmail. Với nhân lực gồm 3 nhân viên chính thức và 14 nhân viên cộng tác, họ đã phát triển được và ra mắt dịch vụ vào tháng 7 năm 1996. Trong vòng một tháng sau khi bắt đầu, họ đã có 100.000 thuê bao (khách hàng). Vào tháng 12 năm 1997, chỉ sau không đến 18 tháng ra mắt dịch vụ, Hotmail đã có trên 12 triệu thuê bao và được Microsoft mua lại với giá 400 triệu đô la Mỹ (theo báo cáo). Thành công của Hotmail được cho là do lợi thế đi tiên phong và tiếp thị quảng bá của thư điện tử.

Thư điện tử trên Web tiếp tục phát triển mạnh, trở nên thông minh và mạnh mẽ hơn qua từng năm. Một trong những dịch vụ phổ biến nhất ngày nay là gmail của Google, nó cung cấp bộ nhớ miễn phí hàng gigabyte, lọc thư rác nâng cao và phát hiện virus, tính năng mã hoá thư điện tử (SSL), lấy thư từ dịch vụ thư điện tử bên thứ ba, và giao diện hướng tìm kiếm.

Ngày càng có nhiều người sử dụng đang gửi và truy nhập thư điện tử qua trình duyệt Web. Hotmail giới thiệu truy nhập thư điện tử qua Web vào giữa thập kỷ 1990, hiện nay thư điện tử dựa trên Web cũng được Yahoo, Google cũng như nhiều tập đoàn và trường đại học lớn sử dụng. Với dịch vụ này, đại lý người sử dụng là một trình duyệt Web bình thường và người sử dụng truyền thông với hòm thư từ xa của họ thông qua HTTP. Khi người nhận (Bob) muốn truy nhập thư trong hòm thư của mình, thì thư điện tử được gửi từ máy chủ thư của Bob tới trình duyệt của Bob sử dụng giao thức HTTP chứ không phải là POP3 hay

IMAP. Khi người gửi (Alice) muốn gửi thư điện tử, thư điện tử của cô ấy sẽ được gửi từ trình duyệt của cô ta tới máy chủ thư của cô ấy trên HTTP chứ không phải SMTP. Tuy nhiên, máy chủ thư của Alice vẫn gửi bản tin tới và nhận bản tin từ những máy chủ thư khác qua SMTP.

## 2.5 Ứng dụng hệ thống tên miền

Nhiệm vụ của ứng dụng hệ thống tên miền là chuyển đổi giữa tên miền và địa chỉ IP. Người sử dụng sử dụng ứng dụng này dùng tên chứ không dùng địa chỉ IP.

### 2.5.1 Tổng quan DNS

Con người có thể được nhận dạng bằng nhiều cách thức khác nhau. Ví dụ, chúng ta có thể được nhận dạng bằng tên trong giấy khai sinh, chúng ta cũng có thể được nhận dạng thông qua số chứng minh thư, hoặc số bằng lái xe. Mặc dù mỗi định danh trên đều có thể sử dụng để nhận dạng con người, song trong một ngữ cảnh cụ thể, một định danh này có thể phù hợp hơn các định danh khác. Ví dụ, máy tính ở IRS (đại lý thu thập phí của Mỹ) ưa sử dụng các số an ninh xã hội có độ dài cố định hơn là tên trong giấy khai sinh. Mặt khác, những người bình thường thích tên trong giấy khai sinh (dễ nhớ) hơn là số an ninh xã hội hoặc chứng minh thư. (Bạn có thể tưởng tượng thế này, bạn sẽ nói: “Xin chào, tên tôi là 132-67-9875. Hãy làm quen với chồng tôi, 178-87-1146”).

Có thể nhận dạng con người bằng nhiều cách khác nhau, các máy tính trên Internet cũng vậy. Một định danh cho máy trạm là tên trạm chủ (hostname). Tên trạm chủ, ví dụ `cnn.com`, `www.yahoo.com`, `gaia.cs.umass.edu`, `cis.poly.edu`, là những tên dễ nhớ và do đó phù hợp với mọi người. Tuy nhiên, tên trạm chủ cung cấp rất ít (nếu có) thông tin về vị trí của nó trong Internet. (Một tên trạm chủ như `www.eurecom.fr` có kết thúc bằng mã quốc gia `.fr` cho ta thấy là trạm chủ này có thể ở Pháp, song không có thêm thông tin gì khác). Thêm nữa, vì tên trạm chủ có thể bao gồm các ký tự chữ và số có độ dài thay đổi nên sẽ khó xử lý trên các bộ định tuyến (router). Vì những lý do này, các trạm chủ cũng được nhận dạng bằng địa chỉ IP.

Một địa chỉ IP (phiên bản 4) có độ dài 4 byte và có cấu trúc phân cấp cứng. Ví dụ, địa chỉ IP là 121.7.106.83, mỗi byte tách biệt nhau bằng một dấu chấm và có giá trị số thập phân từ 0 đến 255. Địa chỉ IP là phân cấp vì nếu chúng ta quét địa chỉ từ trái qua phải, chúng ta nhận được nhiều thông tin hơn về vị trí trạm được đặt ở đâu trong Internet (nghĩa là, nó đang ở trong mạng nào, trong mạng của các mạng). Tương tự, khi chúng ta quét một địa chỉ bưu chính từ trên xuống dưới, chúng ta có nhiều thông tin cụ thể hơn về địa chỉ của thư.

Như vậy, có hai cách để nhận dạng một trạm chủ, bằng tên trạm chủ hoặc bằng địa chỉ IP. Con người thích định danh theo tên trạm chủ vì nó dễ nhớ, trong khi các bộ định tuyến thì lại ưu tiên sử dụng địa chỉ IP có cấu trúc phân cấp và độ dài cố định. Để hài hoà những ý muốn này, chúng ta cần một dịch vụ thư mục để phiên dịch tên trạm chủ sang địa chỉ IP. Đây là nhiệm vụ chính của **hệ thống tên miền (DNS)** của Internet. DNS là (1) cơ sở dữ liệu phân tán thực hiện trên cấu trúc phân cấp các **máy chủ DNS** và (2) là một giao thức lớp ứng dụng cho phép các trạm chủ truy vấn cơ sở dữ liệu phân tán. Các máy chủ DNS thường là các máy UNIX chạy phần mềm BIND (Berkeley Internet Name Domain). Giao thức DNS chạy trên UDP và sử dụng cổng 53.

DNS thường được các giao thức ứng dụng khác sử dụng - bao gồm HTTP, SMTP và FTP - để phiên dịch tên trạm chủ do người sử dụng đưa ra thành địa chỉ IP. Ví dụ, hãy xem điều gì xảy ra khi khi một trình duyệt (máy khách HTTP) chạy trên máy tính của người sử dụng nào đó, yêu cầu trang [www.someschool.edu/index.html](http://www.someschool.edu/index.html). Để máy trạm của người sử dụng có thể gửi một bản tin yêu cầu HTTP tới máy chủ Web [www.someschool.edu](http://www.someschool.edu) thì máy trạm của người sử dụng trước tiên phải có được địa chỉ IP của [www.someschool.edu](http://www.someschool.edu). Điều này được thực hiện như sau:

1. Máy tính của người sử dụng đó hoạt động như bên máy khách của ứng dụng DNS.
2. Trình duyệt lấy tên trạm chủ [www.someschool.edu](http://www.someschool.edu) từ URL và chuyển tên trạm chủ tới bên máy khách của ứng dụng DNS.
3. Máy khách DNS gửi một truy vấn chứa tên trạm chủ tới máy chủ DNS.

4. Máy khách DNS nhận được bản tin trả lời chứa địa chỉ IP cho tên trạm chủ.
5. Một khi trình duyệt nhận được địa chỉ IP từ DNS, nó sẽ bắt đầu một kết nối TCP tới tiến trình máy chủ HTTP thông qua cổng 80 của máy chủ có địa chỉ IP đó.

Từ ví dụ này ta thấy DNS gây thêm một phần trễ - đôi khi khá lớn – vào các ứng dụng Internet phải sử dụng đến nó. Rất may là địa chỉ IP yêu cầu thường được lưu đệm ở một máy chủ DNS gần đó, giúp giảm lưu lượng mạng DNS cũng như giảm trễ DNS trung bình.

DNS còn cung cấp một số dịch vụ quan trọng khác ngoài việc phiên dịch tên trạm chủ thành địa chỉ IP như:

- **Host aliasing** (Bí danh trạm chủ). Một trạm chủ có tên trạm phức tạp có thể có một hay nhiều bí danh. Ví dụ, tên trạm như `relay1.west-coast.enterprise.com` có thể có hai bí danh là `enterprise.com` và `www.enterprise.com`. Trong trường hợp này, tên trạm chủ `relay1.west-coast.enterprise.com` được gọi là tên trạm chính tắc (canonical hostname). Tên trạm bí danh, nếu có, thường dễ nhớ hơn là tên trạm chính tắc. DNS có thể được một ứng dụng gọi tới để lấy tên trạm chính tắc cho một tên trạm bí danh được cung cấp, cũng như địa chỉ IP của trạm chủ đó.
- **Mail server aliasing** (Bí danh máy chủ thư). Rõ ràng là chúng ta cũng mong muốn địa chỉ thư điện tử cần phải dễ nhớ. Ví dụ, nếu Bob có một tài khoản Hotmail, địa chỉ email của Bob có thể đơn giản là `bob@hotmail.com`. Tuy nhiên, tên trạm máy chủ thư Hotmail có thể phức tạp hơn và khó nhớ hơn cái tên `hotmail.com` (ví dụ tên chính tắc có thể là `relay1.west-coast.hotmail.com`). DNS có thể được một ứng dụng thư gọi tới để lấy tên chính tắc cho một tên trạm bí danh được cung cấp, cũng như địa chỉ IP của trạm chủ đó. Thực tế là, bản ghi MX (xem ở phần sau) cho phép một máy chủ thư và máy chủ web của một công ty có cùng tên trạm (bí danh). Ví dụ, một máy chủ Web và máy chủ thư của công ty có thể có cùng tên gọi là `enterprise.com`.

- **Phân bổ tải.** DNS cũng được sử dụng để thực hiện phân bổ tải giữa các máy chủ nhân rộng (replicated), như các máy chủ nhân rộng Web. Các trang Web bận rộn, như `cnn.com` thường được nhân rộng trên nhiều máy chủ, với mỗi máy chủ chạy trên một hệ thống đầu cuối khác nhau và có địa chỉ IP khác nhau. Đối với các máy chủ Web nhân rộng, một tập địa chỉ IP sẽ được gắn với một tên chính tắc (canonical). Cơ sở dữ liệu DNS chứa tập địa chỉ IP này. Khi máy khách thực hiện truy vấn DNS đối với tên ánh xạ tới tập địa chỉ, máy chủ sẽ đáp ứng toàn bộ tập địa chỉ IP này, nhưng xoay vòng thứ tự những địa chỉ này với mỗi lần trả lời. Vì một máy khách thường gửi bản tin yêu cầu HTTP của nó đến địa chỉ IP đúng đầu tập danh sách nên việc xoay vòng DNS sẽ phân tải lưu lượng giữa các máy chủ nhân rộng. Việc xoay vòng DNS cũng được sử dụng cho ứng dụng e-mail sao cho nhiều máy chủ thư có thể có cùng bí danh. Gần đây, các công ty phân phối nội dung như Akamai đã sử dụng DNS theo cách phức tạp hơn để cung cấp phân bổ nội dung Web.

Giống như HTTP, FTP và SMTP, giao thức DNS là một giao thức lớp ứng dụng vì (1) nó chạy giữa các hệ thống đầu cuối truyền thông sử dụng mô hình client-server và (2) nó dựa trên giao thức lớp giao vận end-to-end để truyền các bản tin DNS giữa các hệ thống truyền thông đầu cuối. Tuy nhiên, ở một ý nghĩa khác, vai trò của DNS khác với ứng dụng web, truyền file và thư điện tử. Khác với những ứng dụng này, DNS không phải là ứng dụng mà người sử dụng tương tác trực tiếp. Thay vào đó, DNS cung cấp chức năng Internet cốt lõi, đó là phiên dịch tên trạm chủ thành địa chỉ IP cho các ứng dụng người sử dụng và các phần mềm khác trên Internet. Ta đã biết là kiến trúc Internet phức tạp, tập trung chủ yếu ở biên mạng. DNS thực hiện tiến trình phiên dịch tên thành địa chỉ sử dụng máy khách và máy chủ đặt tại biên của mạng, là một ví dụ của tính phức tạp này.

DNS được đặc tả trong RFC 1034, RFC 1035 và được cập nhật trong một vài RFC bổ sung. Đó là một hệ thống phức tạp, chúng ta chỉ đề cập những khía cạnh chủ yếu trong hoạt động của nó ở đây.

### **2.5.2 Hoạt động của DNS**

Hãy xem xét hoạt động của DNS ở mức tổng quan và thảo luận về dịch vụ phiên dịch tên-địa chỉ IP.

Giả sử một ứng dụng nào đó (như trình duyệt web) chạy trên trạm chủ người sử dụng cần phiên dịch từ tên trạm chủ thành địa chỉ IP. Ứng dụng này sẽ kích hoạt phía máy khách của DNS, đặc tả tên trạm chủ cần phải phiên dịch. (Trên nhiều máy tính chạy trên UNIX, `gethostbyname()` là hàm gọi ứng dụng này để thực hiện việc phiên dịch này). Sau đó, DNS ở trạm chủ người sử dụng sẽ tiếp tục quá trình, gửi bản tin truy vấn vào trong mạng. Tất cả bản tin truy vấn và phản hồi được gửi trên dữ liệu đồ UDP tới cổng 53. Sau một khoảng thời gian trễ từ mili giây đến hàng giây, DNS trong trạm chủ người sử dụng sẽ nhận được bản tin phản hồi DNS cung cấp ánh xạ yêu cầu. Ánh xạ này sau đó được chuyển đến ứng dụng đưa ra. Vì vậy, từ khía cạnh của ứng dụng đưa ra trong trạm chủ người sử dụng, DNS là hộp đen cung cấp dịch vụ phiên dịch đơn giản và dễ dàng. Song thực tế thì hộp đen này thực hiện dịch vụ khá phức tạp, bao gồm một lượng lớn máy chủ DNS phân bố khắp nơi trên thế giới, cũng như một giao thức lớp ứng dụng đặc tả các máy chủ DNS và các trạm chủ truy vấn truyền thông với nhau như thế nào.

Một thiết kế đơn giản cho DNS có thể có một máy chủ DNS chứa tất cả ánh xạ. Trong thiết kế tập trung này, các máy khách gửi truy vấn trực tiếp tới một máy chủ DNS duy nhất và máy chủ DNS sẽ đáp ứng trực tiếp tới các máy khách truy vấn. Mặc dù tính đơn giản của thiết kế này rất hấp dẫn nhưng nó lại không phù hợp với Internet ngày nay, với số lượng trạm rất lớn và vẫn gia tăng từng ngày. Các khó khăn mà thiết kế tập trung gặp phải là:

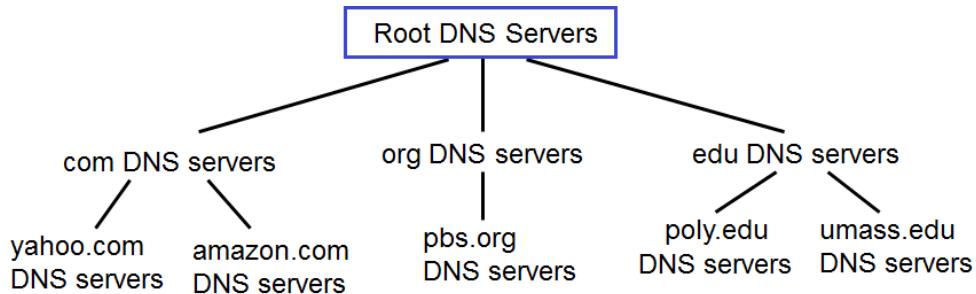
- **Một điểm lỗi.** Nếu máy chủ DNS trực trặc thì toàn bộ mạng Internet bị trực trặc.
- **Lưu lượng.** Một máy chủ DNS đơn sẽ phải xử lý toàn bộ truy vấn DNS (cho toàn bộ yêu cầu HTTP và email gửi từ hàng trăm triệu máy tính).
- **Cơ sở dữ liệu tập trung ở xa.** Một máy chủ DNS không thể “gắn” với tất cả máy khách đang truy vấn. Nếu chúng ta đặt máy chủ DNS đơn này ở Thành phố New York thì tất cả truy vấn từ Việt nam sẽ phải đi tới phía bên kia của trái đất, có lẽ trên cả những kết nối chậm và tắc nghẽn. Điều này làm tăng đáng kể độ trễ.

- **Bảo trì.** Một máy chủ DNS đơn sẽ phải lưu giữ các bản ghi cho tất cả các trạm trên Internet. Không những cơ sở dữ liệu tập trung này sẽ phải có kích thước rất lớn mà nó còn phải thường xuyên cập nhật cho từng trạm mới xuất hiện.

Tóm lại, cơ sở dữ liệu tập trung trong một máy chủ DNS là không phù hợp quy mô. Dẫn đến là DNS được thiết kế phân tán. Thực tế, DNS là một ví dụ tuyệt vời cho việc thực hiện cơ sở dữ liệu phân tán trên Internet.

### **Phân bố cơ sở dữ liệu DNS:**

Để giải quyết vấn đề quy mô (scale), DNS sử dụng nhiều máy chủ, tổ chức theo kiểu phân cấp và phân tán trên khắp địa cầu. Không có máy chủ nào chứa toàn bộ ánh xạ cho các máy tính trên Internet. Thay vào đó, các ánh xạ sẽ được phân bố giữa các máy chủ DNS. Trong mức tiệm cận ban đầu, có ba lớp máy chủ DNS: máy chủ DNS gốc (root), máy chủ DNS miền mức cao (top-level domain - TLD) và máy chủ DNS thẩm quyền. Chúng được tổ chức phân cấp như hình 2.16. Để hiểu các lớp máy chủ tương tác như thế nào, giả sử một máy khách DNS muốn xác định địa chỉ IP cho tên miền `www.amazon.com`.



*Hình 2.16: Cấu trúc phân cấp của máy chủ DNS*

Với tiệm cận ban đầu, những sự kiện sau sẽ xảy ra. Đầu tiên máy khách sẽ kết nối với một máy chủ gốc, máy chủ gốc này sẽ trả về những địa chỉ của các máy chủ TLD cho tên miền mức cao nhất là `.com`. Sau đó máy khách kết nối tới một trong những máy chủ TLD, máy chủ TLD này sẽ trả về địa chỉ IP của máy chủ thẩm quyền cho `amazon.com`. Cuối cùng, máy khách kết nối tới các máy chủ thẩm quyền cho `amazon.com`, máy chủ thẩm quyền này trả về địa chỉ IP cho tên trạm chủ `www.amazon.com`. Chúng ta sẽ xem xét chi tiết tiến trình DNS này sau. Trước tiên ta sẽ nhìn vào ba lớp của máy chủ DNS:

- **Các máy chủ DNS gốc.** Trong mạng Internet có 13 máy chủ DNS gốc (dán nhãn từ A đến M), hầu hết đặt ở Bắc Mỹ. Hình 2.17 là bản đồ máy chủ DNS gốc tháng 10 năm 2009. Mặc dù chúng ta coi từng máy chủ trong 13 máy chủ DNS này là một máy chủ đơn, song mỗi máy chủ thực nhất là một nhóm máy chủ nhau rộng với mục đích là tăng độ an toàn và tin cậy.



Hình 2.17: Các máy chủ DNS gốc năm 2009 (tên, tổ chức, vị trí)

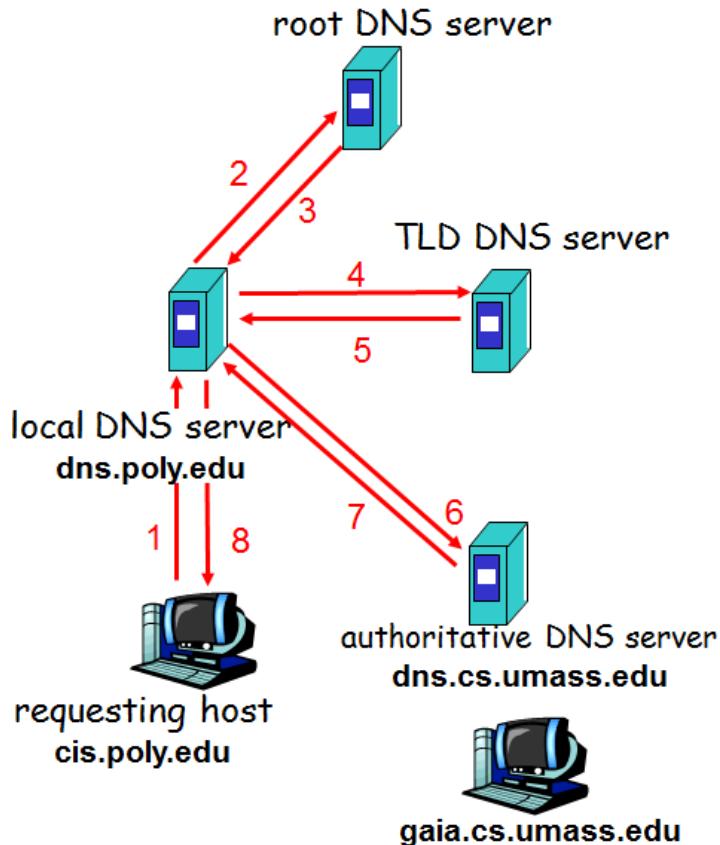
- **Các máy chủ tên miền mức cao (TLD).** Những máy chủ này chịu trách nhiệm về các miền mức cao như com, org, net, edu và gov và tất cả các miền mức cao của quốc gia như us, uk, fr, ca, cn, jp và vn. Công ty Network Solutions (các giải pháp mạng) duy trì các máy chủ TLD cho tên miền mức cao com và công ty Educause duy trì các máy chủ TLD tên miền mức cao edu.
- **Các máy chủ DNS thẩm quyền.** Bất cứ tổ chức nào có các máy truy nhập công cộng (như máy chủ web và máy chủ thư điện tử) trên Internet phải cung cấp các bản ghi DNS có khả năng truy cập ra công chúng, với ánh xạ tên của các trạm chủ này sang địa chỉ IP. Máy chủ DNS thẩm quyền của tổ chức sẽ chứa những bản ghi DNS này. Tổ chức có thể lựa chọn triển khai máy chủ DNS thẩm quyền riêng để giữ các bản ghi này; hoặc tổ chức đó có thể phải trả tiền để lưu trữ các bản ghi trong máy chủ DNS thẩm quyền của một nhà cung cấp dịch vụ nào đó. Hầu hết các trường đại học và những công ty lớn đều triển

khai và duy trì máy chủ DNS thẩm quyền sơ cấp và thứ cấp (dự phòng) của riêng mình.

Các máy chủ gốc, TLD và thẩm quyền đều thuộc về cấu trúc phân cấp máy chủ DNS như trong hình 2.16. Còn có một loại máy chủ DNS quan trọng khác nữa được gọi là **máy chủ DNS cục bộ**. Một máy chủ DNS cục bộ không bắt buộc thuộc phân cấp máy chủ tuy nhiên nó là trung tâm của kiến trúc DNS. Mỗi ISP – như ISP của trường đại học, viện nghiên cứu, công ty hay là một ISP khu vực dân cư – đều có một máy chủ DNS cục bộ (còn gọi là máy chủ tên mặc định). Khi một trạm chủ kết nối tới một ISP, ISP cung cấp các địa chỉ IP của một hoặc nhiều máy chủ DNS cục bộ cho trạm chủ đó (thường là qua DHCP). Người dùng có thể dễ dàng xác định địa chỉ IP của máy chủ DNS cục bộ của mình bằng cách truy nhập cửa sổ trạng thái mạng trong Window hay UNIX. Một máy chủ DNS cục bộ của một trạm thường “gần” với trạm đó. Đối với ISP của một tổ chức, máy chủ DNS cục bộ có thể nằm trên cùng mạng LAN với máy trạm; đối với ISP của khu dân cư thì nó thường chỉ tách biệt với máy chủ DNS cục bộ qua một hoặc vài bộ định tuyến. Khi một trạm chủ truy vấn DNS, truy vấn sẽ được gửi tới máy chủ DNS cục bộ - hoạt động như một proxy - chuyển tiếp truy vấn đó vào cây phân cấp máy chủ DNS.

Lấy một ví dụ đơn giản. Giả sử trạm chủ cis.poly.edu muốn có địa chỉ IP của gaia.cs.umass.edu. Cũng giả sử là máy chủ DNS cục bộ của trường bách khoa này gọi là dns.poly.edu và một máy chủ DNS thẩm quyền cho gaia.cs.umass.edu là dns.umass.edu. Trên hình 2.18, đầu tiên trạm chủ cis.poly.edu gửi bản tin truy vấn DNS tới máy chủ DNS cục bộ của nó là dns.poly.edu. Bản tin truy vấn này chứa tên cần phiên dịch là gaia.cs.umass.edu. Máy chủ DNS cục bộ chuyển tiếp bản tin truy vấn tới máy chủ DNS gốc. Máy chủ DNS gốc đánh dấu hậu tố tên miền .edu và trả về cho máy chủ DNS cục bộ một danh sách địa chỉ IP của các máy chủ TLD chịu trách nhiệm về tên miền .edu. Sau đó máy chủ DNS cục bộ gửi lại bản tin truy vấn tới một trong các máy chủ TLD. Máy chủ TLD sẽ đánh dấu hậu tố umass.edu và đáp ứng với địa chỉ của máy chủ DNS thẩm quyền cho trường đại học Massachusetts, có tên là dns.umass.edu. Cuối cùng, máy chủ DNS cục bộ gửi lại truy vấn trực tiếp tới

`dns.umass.edu`, và máy chủ này đáp ứng lại cùng với địa chỉ IP của `gaia.cs.umass.edu`. Chú ý là trong ví dụ này, để lấy được ánh xạ cho một tên trạm chủ thì cần truyền tới tám bản tin DNS: bốn bản tin truy vấn và bốn bản tin trả lời. Sau này ta sẽ thấy việc đệm DNS sẽ làm giảm lưu lượng truy vấn này.

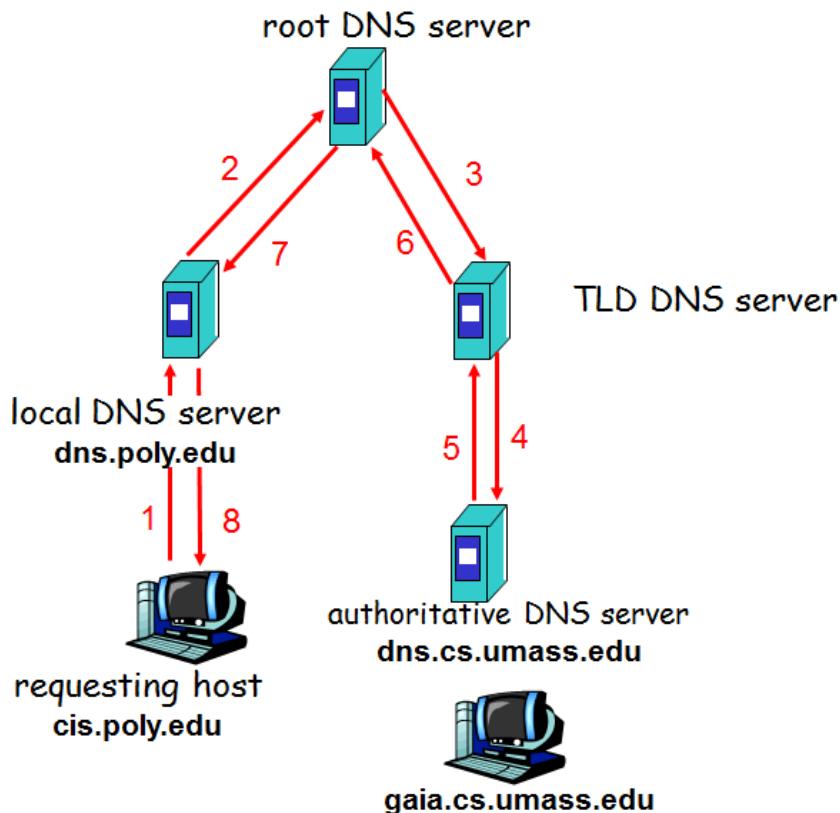


Hình 2.18: Tương tác giữa các máy chủ DNS khác nhau

Ví dụ trên đã giả sử là máy chủ TLD biết tên máy chủ DNS thẩm quyền cho tên trạm chủ. Nhìn chung thì điều này không phải là luôn luôn đúng. Thay vào đó, máy chủ TLD có thể chỉ biết một máy chủ DNS trung gian biết máy chủ DNS thẩm quyền cho tên trạm chủ. Ví dụ, giả sử là trường đại học Massachusetts có máy chủ DNS của trường có tên là `dns.umass.edu`. Cũng giả sử là mỗi khoa trong trường có máy chủ DNS riêng và mỗi máy chủ DNS của khoa có thẩm quyền cho toàn bộ các trạm chủ trong khoa. Trong trường hợp này, khi máy chủ DNS trung gian `dns.umass.edu` nhận một truy vấn tới một trạm có tên trạm kết thúc bằng `cs.umass.edu`, nó trả về `dns.poly.edu` địa chỉ IP của

`dns.cs.umass.edu` là máy chủ thẩm quyền cho tất cả các trạm có tên trạm kết thúc là `.umass.edu`. Khi đó, máy chủ DNS cục bộ `dns.poly.edu` sẽ gửi một truy vấn tới máy chủ DNS thẩm quyền, máy chủ này trả ánh xạ mong muốn tới máy chủ DNS cục bộ, đến lượt máy chủ này trả ánh xạ tới trạm chủ yêu cầu. Trong trường hợp này tổng cộng có tới 10 bản tin DNS được gửi đi.

Hình 4.3 cho thấy ví dụ sử dụng cả hai loại truy vấn là đệ quy và lặp. Truy vấn gửi từ `cis.poly.edu` tới `dns.poly.edu` là truy vấn đệ quy, vì truy vấn yêu cầu `dns.poly.edu` lấy ánh xạ thay mặt nó. Nhưng ba truy vấn tiếp theo là lặp vì tất cả phản hồi được trả trực tiếp về `dns.poly.edu`. Trên lý thuyết, bất kỳ truy vấn DNS nào đều có thể là lặp hoặc đệ quy. Ví dụ, hình 2.19 cho thấy chuỗi truy vấn DNS mà tất cả các truy vấn là đệ quy. Trên thực tế, các truy vấn thường tuân theo mô hình trong hình 2.18: Truy vấn từ trạm chủ yêu cầu tới máy chủ DNS cục bộ là đệ quy và các truy vấn khác là lặp.



Hình 2.19: Truy vấn đệ quy trong DNS

### **Lưu đệm DNS:**

Lưu đệm DNS (DNS cache) là một đặc tính quan trọng của hệ thống DNS. Thực tế là DNS sử dụng lưu đệm DNS để cải thiện hiệu năng trễ và giảm số lượng bản tin DNS xuất hiện trên Internet. Ý tưởng của đệm DNS rất đơn giản. Trong một chuỗi truy vấn, khi một máy chủ DNS nhận tin trả lời DNS (ví dụ, chứa ánh xạ từ tên trạm chủ sang địa chỉ IP), nó có thể lưu đệm ánh xạ này ở bộ nhớ nội bộ. Ví dụ, ở hình 4.3, mỗi khi máy chủ DNS cục bộ dns.poly.edu nhận một phản hồi từ một máy chủ DNS nào đó, nó có thể lưu đệm bất kỳ thông tin nào trong phản hồi đó. Nếu một cặp địa chỉ IP/tên trạm chủ được lưu đệm trong một máy chủ DNS và có một yêu cầu nữa tới máy chủ DNS này để lấy chính tên trạm chủ đó thì máy chủ DNS có thể cung cấp địa chỉ IP yêu cầu, ngay cả khi nó không có thẩm quyền về tên trạm chủ đó. Vì các trạm và ánh xạ giữa tên trạm chủ và địa chỉ IP không phải luôn cố định, máy chủ DNS sẽ hủy thông tin đệm sau một khoảng thời gian (thường đặt là 2 ngày).

Lấy ví dụ, giả sử là trạm apricot.poly.edu truy vấn dns.poly.edu về địa chỉ IP của tên trạm chủ cnn.com. Thêm nữa, giả sử là một vài giờ sau, một trạm khác của Đại học Bách khoa, như kiwi.poly.fr cũng truy vấn dns.poly.edu với cùng tên trạm chủ. Vì có lưu đệm nên máy chủ DNS cục bộ sẽ có thể trả lời tức thì về địa chỉ IP của cnn.com cho trạm yêu cầu thứ hai này mà không phải truy vấn bất kỳ một máy chủ DNS nào khác. Máy chủ DNS cục bộ cũng có thể lưu đệm địa chỉ IP của các máy chủ TLD, do đó cho phép máy chủ cục bộ bỏ qua các máy chủ DNS gốc trong chuỗi truy vấn (điều này thường xảy ra).

### **2.5.3 Bản ghi và bản tin DNS**

#### **Bản ghi DNS:**

Các máy chủ DNS cùng triển khai cơ sở dữ liệu phân tán lưu giữ các **bản ghi nguồn** (RRs), bao gồm các RRs cung cấp ánh xạ tên trạm chủ-tới-địa chỉ IP. Mỗi bản tin trả lời DNS mang một hoặc nhiều bản ghi nguồn. Phần này cung cấp một cái nhìn bao quát về các

bản ghi nguồn và bản tin DNS, chi tiết có thể xem trong các RFC DNS (RFC 1034 và RFC 1035).

Một bản ghi nguồn có bốn trường sau:

(Name, Value, Type, TTL)

TTL là thời gian sống của bản ghi nguồn; nó xác định khi nào nguồn phải được bỏ ra khỏi lưu đệm. Trong ví dụ về bản ghi dưới đây, chúng ta bỏ qua trường TTL. Nghĩa của trường Name và Value phụ thuộc vào Type:

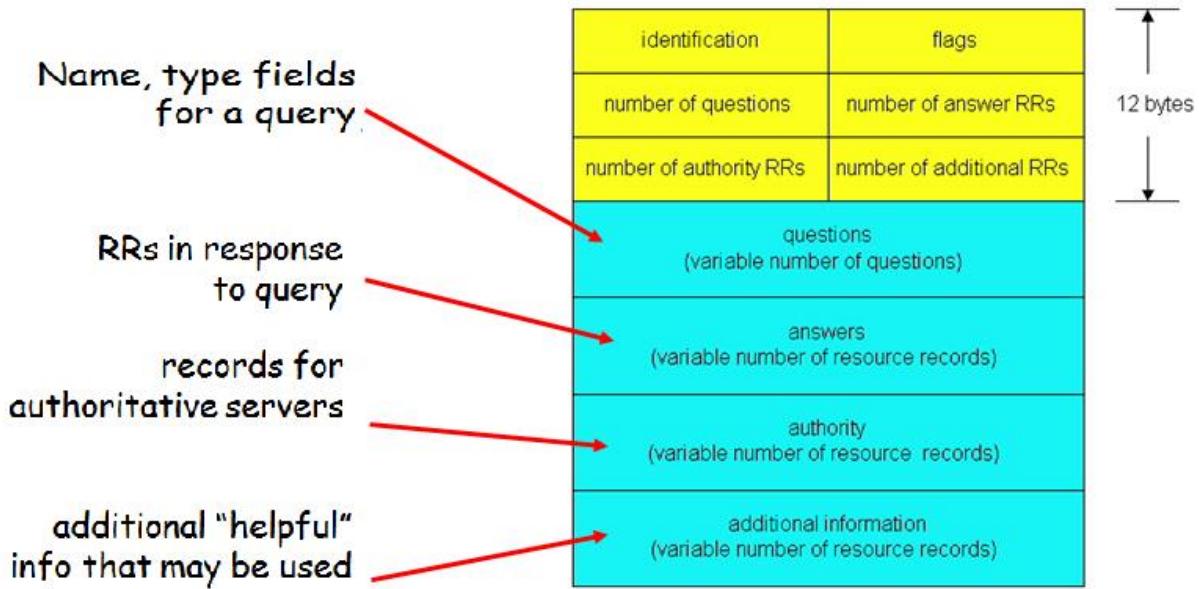
- Nếu Type=A thì Name (tên) là tên trạm và Value (giá trị) là địa chỉ IP của tên trạm đó. Vì vậy, bản ghi Type A cung cấp ánh xạ tên trạm chủ-tới-địa chỉ IP chuẩn. Ví dụ (relay1.bar.foo.com, 145.37.93.126, A) là bản ghi Type A.
- Nếu Type=NS thì Name là tên miền (như foo.com) và Value là tên trạm của máy chủ DNS thẩm quyền, máy chủ này biết cách lấy địa chỉ IP của các trạm chủ trong miền đó. Bản ghi này được dùng để định tuyến các truy vấn DNS trong một chuỗi truy vấn. Ví dụ (foo.com, dns.foo.com, NS) là bản ghi Type NS.
- Nếu Type=CNAME thì Value là tên trạm chính tắc cho tên trạm bí danh Name. Bản ghi này có thể cung cấp cho các trạm truy vấn tên chính tắc cho tên trạm. Ví dụ (foo.com, relay1.bar.foo.com, CNAME) là một bản ghi CNAME.
- Nếu Type=MX thì Value là tên chính tắc của máy chủ thư có tên trạm bí danh Name. Ví dụ (foo.com, mail.bar.foo.com, MX) là bản ghi MX. Các bản ghi MX cho phép các tên trạm của máy chủ thư có bí danh đơn giản. Chú ý là bằng cách sử dụng bản ghi MX, một công ty có thể có cùng bí danh cho máy chủ thư và một máy chủ khác (như máy chủ Web của nó). Để nhận được tên chính tắc cho máy chủ thư, một máy khách DNS cần truy vấn bản ghi MX; để nhận tên chính tắc cho máy chủ khác, máy khách DNS sẽ phải truy vấn bản ghi CNAME.

Nếu một máy chủ DNS có thẩm quyền cho một tên trạm cụ thể, thì máy chủ DNS này sẽ chứa bản ghi Type A cho tên trạm đó. (Ngay cả khi nếu máy chủ DNS không thẩm quyền, nó vẫn có thể chứa bản ghi Type A trong bộ đệm của nó). Nếu một máy chủ không có thẩm quyền về tên trạm, nó sẽ chứa một bản ghi loại NS cho tên miền chứa tên trạm, nó cũng sẽ có bản ghi Type A cung cấp địa chỉ IP của máy chủ DNS trong trường Value của bản ghi NS. Ví dụ, giả sử là một máy chủ TLD .edu không có thẩm quyền cho trạm gaia.cs.umass.edu, ví dụ (umass.edu, dns.umass.edu, NS). Máy chủ TLD .edu cũng sẽ chứa bản ghi Type A, ánh xạ máy chủ DNS dns.umass.edu với địa chỉ IP, ví dụ (dns.umass.edu, 128.119.40.111, A).

### Bản tin DNS:

Lúc trước chúng ta đã đề cập đến các bản tin truy vấn và trả lời DNS. Chúng chỉ là hai loại bản tin của DNS. Hơn nữa, cả bản tin truy vấn và trả lời đều có khuôn dạng chung (hình 2.20). Ngữ nghĩa của các trường trong bản tin DNS như sau:

- 12 byte đầu tiên là *phần tiêu đề* (*header*), bao gồm một số trường. Trường đầu tiên là một số dài 16 bit nhận dạng truy vấn. Định dạng này được sao chép vào bản tin trả lời truy vấn, cho phép máy khách đối chiếu bản tin trả lời nhận được với truy vấn đã gửi đi. Có một số cờ trong trường cờ. Cờ truy vấn/trả lời 1 bit cho biết bản tin là truy vấn (0) hay là trả lời (1). Cờ thẩm quyền 1 bit được thiết lập trong bản tin trả lời khi máy chủ DNS là máy chủ thẩm quyền đối với tên được truy vấn. Một cờ yêu cầu để quy 1 bit được thiết lập khi một máy khách (trạm chủ hay máy chủ DNS) mong muốn máy chủ DNS thực hiện để quy khi nó không có bản ghi. Trường khả để quy 1 bit được thiết lập trong bản tin trả lời nếu máy chủ DNS hỗ trợ để quy. Trong tiêu đề này còn có bốn trường số nữa. Những trường này chỉ ra số lần xuất hiện của bốn loại dữ liệu tiếp theo tiêu đề.
- *Phần câu hỏi* (*question*) chứa thông tin về truy vấn đang được thực hiện. Phần này gồm (1) trường tên chứa tên đang được truy vấn, (2) trường loại chỉ ra loại câu hỏi về tên, ví dụ: một địa chỉ trạm gắn với tên (Type A) hay máy chủ thư đổi với tên (Type MX).



Hình 2.20: Khuôn dạng bản tin DNS

- Trong một bản tin trả lời từ máy chủ DNS, *phản trả lời (answer)* chứa bản ghi nguồn cho tên được truy vấn ban đầu. Nhớ là trong mỗi bản ghi nguồn có trường Type (ví dụ: A, NS, CNAME, MX), Value và TTL. Một bản tin trả lời có thể trả về nhiều RRs vì một tên trạm có thể có nhiều địa chỉ IP (ví dụ, với các máy chủ nhân rộng).
- Phản *thẩm quyền (authority)* chứa các bản ghi của các máy chủ thẩm quyền khác.
- Phản *bổ trợ (additional)* chứa những bản ghi hỗ trợ khác. Ví dụ, trường trả lời trong bản tin trả lời một truy vấn MX chứa bản ghi nguồn cung cấp tên trạm chính tắc của máy chủ thư. Trường bổ trợ này chứa bản ghi Type A cung cấp địa chỉ IP cho tên trạm chính tắc của máy chủ thư này.

Để gửi bản tin yêu cầu DNS trực tiếp từ một trạm của người dùng tới một máy chủ DNS, điều này có thể dễ dàng thực hiện với chương trình **nslookup**, chương trình này thường có sẵn trên các hệ điều hành Windows và UNIX. Ví dụ, từ một trạm chủ Window, mở Command Promt (nhắc lệnh) và kích hoạt chương trình nslookup bằng cách gõ “nslookup”. Sau khi kích hoạt nslookup, người dùng có thể gửi một truy vấn DNS tới bất

kỳ máy chủ DNS nào (máy chủ gốc, TLD hoặc thẩm quyền). Sau khi nhận bản tin trả lời từ máy chủ DNS, nslookup sẽ hiển thị bản ghi trong bản tin trả lời (dưới dạng đọc được). Có một giải pháp khác chạy nslookup từ trạm của người dùng, là người dùng có thể vào một trong nhiều trang Web cho phép người dùng triển khai nslookup từ xa. (Chỉ cần gõ nslookup vào một công cụ tìm kiếm và như vậy người dùng có thể tới được những trang đó).

### **Chèn bản ghi DNS vào cơ sở dữ liệu DNS:**

Ở trên chúng ta đã tập trung vào việc làm thế nào để lấy bản ghi từ cơ sở dữ liệu DNS. Chúng ta có thể tự hỏi là làm thế nào để đưa bản ghi vào cơ sở dữ liệu trong lần đầu tiên. Hãy nhìn vào cách thực hiện điều này từ ví dụ cụ thể. Giả sử chúng ta vừa tạo lập một công ty mới gọi là Network Utopia. Chắc chắn điều đầu tiên mà chúng ta muốn làm là đăng ký tên miền networkutopia.com với công ty đăng ký. Một công ty đăng ký là một cơ quan thương mại có nhiệm vụ xác minh tính duy nhất của tên miền, đưa tên miền vào cơ sở dữ liệu DNS và thu phí của chúng ta cho dịch vụ này. Trước năm 1999, một công ty đăng ký duy nhất là Network Solutions chiếm độc quyền về việc đăng ký tên miền cho tên miền .com, .net và .org. Song ngày nay có nhiều công ty đăng ký cạnh tranh khách hàng, và Công ty ICANN (Internet Corporation for Assigned Names and Numbers - <http://www.icann.org/>) cũng đã được công nhận là những công ty đăng ký. Danh sách các công ty đăng ký được công nhận có ở trang <http://www.internic.net>.

Khi chúng ta đăng ký tên miền networkutopia.com với một công ty đăng ký, chúng ta cũng cần phải cung cấp cho công ty này tên và địa chỉ IP của máy chủ DNS thẩm quyền sơ cấp và thứ cấp. Giả sử tên và địa chỉ IP đó là dns1.networkutopia.com, dns2.networkutopia.com, 212.212.212.1 và 212.212.212.2. Với mỗi máy chủ DNS thẩm quyền này, công ty đăng ký sẽ đảm bảo là bản ghi Type NS và Type A được nhập vào máy chủ com TLD. Đặc biệt là với máy chủ thẩm quyền sơ cấp cho networkutopia.com, công ty đăng ký phải chèn hai bản ghi nguồn sau vào hệ thống DNS:

(networkutopia.com, dns1.networkutopia.com, NS)

(dns1.networkutopia.com, 212.212.212.1, A)

Chúng ta cũng phải đảm bảo là bản tin nguồn Type A cho máy chủ Web của chúng ta www.networkutopia.com và bản ghi nguồn Type MX cho máy chủ thư của chúng ta mail.networkutopia.com được nhập vào máy chủ DNS thẩm quyền của chúng ta. (Mãi cho tới gần đây, nội dung của mỗi máy chủ DNS mới được cấu hình tĩnh, ví dụ từ tệp cấu hình do nhà quản lý hệ thống tạo ra. Gần đây, tùy chọn OPTION mới được bổ sung vào giao thức DNS để cho phép bổ sung hoặc xoá dữ liệu linh động từ cơ sở dữ liệu thông qua các bản tin DNS. RFC2136 và RFC3007 đặc tả việc cập nhật động DNS).

Một khi tất cả các bước trên đã hoàn tất, mọi người mới có thể vào trang Web của chúng ta và gửi email tới nhân viên công ty chúng ta. Nay giờ chúng sẽ xác minh tuyên bố này là đúng. Việc xác minh này cũng giúp cung cấp điều chúng ta đã học được về DNS. Giả sử là Alice ở Australia muốn xem trang Web www.networkutopia.com. Đầu tiên trạm chủ của cô ấy sẽ gửi truy vấn DNS tới máy chủ DNS cục bộ. Máy chủ DNS cục bộ sẽ kết nối tới máy chủ com TLD. (Máy chủ DNS cục bộ cũng sẽ phải kết nối tới máy chủ DNS gốc nếu địa chỉ của máy chủ com TLD không được đệm). Máy chủ TLD này chứa các bản ghi nguồn Type NS và Type A đưa ra trên, bởi vì công ty đăng ký đã có bản ghi nguồn này chèn vào tất cả máy chủ com TLD. Máy chủ com TLD gửi trả lời tới máy chủ DNS cục bộ của Alice, trong bản tin trả lời có chứa hai bản ghi nguồn. Sau đó máy chủ DNS cục bộ gửi một truy vấn DNS tới 212.212.212.1, yêu cầu bản ghi Type A tương ứng với www.netorkutopia.com. Bản ghi này cung cấp địa chỉ IP của máy chủ Web yêu cầu, là 212.212.71.4, máy chủ DNS cục bộ chuyển ngược bản ghi này về máy trạm của Alice. Nay giờ trình duyệt của Alice có thể khởi tạo kết nối TCP tới trạm 212.212.71.4 và gửi yêu cầu HTTP trên kết nối này.

#### **2.5.4 Vấn đề an toàn trong DNS**

Chúng ta đã thấy là DNS là thành phần quan trọng trong cơ sở hạ tầng Internet với nhiều dịch vụ quan trọng, như Web và email sẽ không thể thực hiện được nếu thiếu nó. Vì

vậy, chúng ta thường hỏi DNS có thể bị tấn công như thế nào? Nếu DNS bị tấn công và không cung cấp dịch vụ, thì hầu hết các ứng dụng Internet cũng bị loại bỏ cùng với nó.

Kiểu tấn công đầu tiên đó là tấn công tràn ngập băng thông DDoS (từ chối dịch vụ) chống lại các máy chủ DNS. Ví dụ, một kẻ tấn công có thể cố gửi vào từng máy chủ gốc DNS một lượng lớn gói tin làm tràn lụt máy chủ, rất nhiều hoặc hầu hết các truy vấn DNS hợp pháp không bao giờ được trả lời. Cuộc tấn công DDoS quy mô lớn như vậy chống lại các máy chủ gốc DNS thực sự xảy ra vào ngày 21 tháng 10 năm 2002. Trong vụ tấn công này, những kẻ tấn công tận dụng mạng ma (bonet) gửi những lượng bản tin ICMP không lồ vào toàn bộ 13 máy chủ gốc DNS. Thật may là vụ tấn công quy mô lớn này chỉ gây thiệt hại nhỏ, gần như không ảnh hưởng tới trải nghiệm người dùng Internet. Những kẻ tấn công đã thành công trong việc phát lượng gói tin rất lớn vào các máy chủ gốc. Song rất nhiều máy chủ gốc được bảo vệ bởi các bộ lọc gói, được cấu hình luôn chặn toàn bộ các bản tin ICMP ping trực tiếp vào các máy chủ gốc. Vì vậy, những máy chủ được bảo vệ này vẫn còn rỗi và hoạt động bình thường. Hơn nữa, hầu hết máy chủ DNS cục bộ đệm địa chỉ IP của các máy chủ TLD nên nó vẫn xử lý truy vấn mà không cần phải chuyển qua máy chủ gốc DNS.

Một khả năng tấn công DDoS khác hiệu quả hơn chống lại DNS là gửi lượng truy vấn DNS rất lớn tới các máy chủ TLD, ví dụ tới tất cả các máy chủ TLD xử lý tên miền .com. Sẽ rất khó lọc những truy vấn DNS trực tiếp tới máy chủ DNS, và các máy chủ TLD thường không dễ dàng chuyển qua như máy chủ gốc. Song mức độ nghiêm trọng của những kiểu tấn công này thường được giảm nhẹ phần nào do việc đệm ở các máy chủ DNS cục bộ.

DNS cũng có thể bị tấn công theo nhiều cách khác. Trong kiểu tấn công người trung gian (man-in-the-middle), kẻ tấn công chặn truy vấn từ các trạm và trả về bản tin trả lời giả. Trong kiểu tấn công đầu độc, kẻ tấn công gửi trả lời giả (không có thật) tới máy chủ DNS, lừa máy chủ nhập bản ghi giả vào bộ lưu đệm của nó. Cho dù là kiểu tấn công nào thì cũng là để chuyển hướng người sử dụng trang Web không đề phòng tới trang Web của kẻ tấn

công. Tuy nhiên, những kiểu tấn công này khó thực hiện vì chúng đòi hỏi đánh chặn các gói tin hoặc điều chỉnh máy chủ.

Một kiểu tấn công quan trọng khác là không tấn công vào dịch vụ DNS mà lại khai thác cơ sở hạ tầng DNS để khởi động tấn công DDoS chống lại trạm chủ mục tiêu (ví dụ như máy chủ thư của trường bạn). Trong kiểu tấn công này, kẻ tấn công gửi truy vấn DNS tới rất nhiều máy chủ DNS thẩm quyền, mỗi truy vấn có địa chỉ nguồn giả của máy trạm mục tiêu. Các máy chủ DNS sẽ gửi trả lời trực tiếp tới trạm mục tiêu đó. Nếu những truy vấn này được gia công bằng cách nào đó sao cho đáp ứng có kích thước lớn hơn nhiều (lượng byte) so với bản tin truy vấn (gọi là khuyếch đại), thì kẻ tấn công có thể tràn ngập mục tiêu mà không phải tự tạo ra nhiều lưu lượng. Những kiểu tấn công phản hồi khai thác DNS kiểu này ít thành công cho tới nay.

Tóm lại, DNS đã tự chứng minh sức mạnh đáng kinh ngạc chống lại những cuộc tấn công. Ngày nay, chưa có kiểu tấn công nào cản trở thành công dịch vụ DNS. Có nhiều kẻ tấn công phản hồi thành công, tuy nhiên những cuộc tấn công này có thể (và đang) được giải quyết bằng cấu hình tương ứng của các máy chủ DNS.

## 2.6 Các ứng dụng mạng ngang hàng

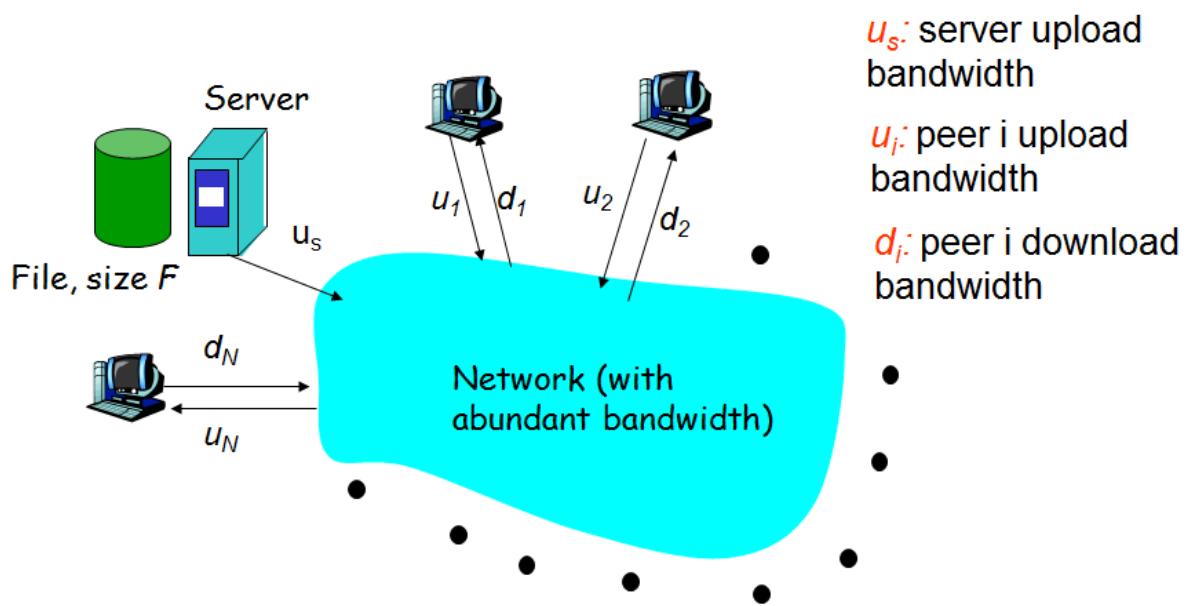
### 2.6.1 Phân bố tệp P2P

Trước tiên, chúng ta xem xét một ứng dụng điển hình của mạng ngang hàng đó là phân bố tệp rất lớn từ một trạm chủ đơn tới một số các trạm chủ (gọi là các thiết bị ngang hàng). Tệp này có thể là phiên bản mới của hệ điều hành Linux, tệp âm nhạc MP3, hay tệp video MPEG. Trong kiến trúc phân bố tệp client-server, máy chủ phải gửi bản sao của tệp tới từng thiết bị ngang hàng, áp đặt một tải trọng khổng lồ lên máy chủ và tiêu tốn lượng băng thông máy chủ rất lớn. Trong phân bố tệp ngang hàng, mỗi thiết bị ngang hàng có thể phân bố lại bất cứ lượng tệp nào mà nó nhận được tới bất cứ thiết bị ngang hàng khác, do đó hỗ trợ máy chủ trong quá trình phân bố. Giao thức phân bố tệp P2P thông dụng nhất là BitTorrent. Giao thức này ban đầu được Bram Cohen phát triển, và hiện nay nhiều máy khách BitTorrent độc lập nhau đã thích ứng với giao thức BitTorrent, cũng tương tự

như các máy khách của trình duyệt Web thích ứng với giao thức HTTP. Chúng ta sẽ xem xét khả năng tự mở rộng qui mô của kiến trúc P2P trong ngữ cảnh phân bổ tệp. Phần này cũng sẽ mô tả BitTorrent chi tiết hơn, và nhấn mạnh các đặc tính và đặc điểm quan trọng của nó.

### Kiến trúc P2P trong phân bổ tệp:

Để so sánh kiến trúc máy khách- máy chủ và kiến trúc ngang hàng và minh họa khả năng tự mở rộng qui mô của P2P, chúng ta xem xét mô hình định lượng đơn giản nhằm phân bổ tệp tới một tập cố định các thiết bị ngang hàng cho cả hai kiến trúc. Trên hình 2.21 máy chủ và các thiết bị ngang hàng được kết nối tới Internet thông qua các liên kết truy nhập.



Hình 2.21: Minh họa phân bổ tệp.

Biểu diễn tốc độ tải lên của liên kết truy nhập máy chủ là  $u_s$ , tốc độ tải lên của liên kết truy nhập thiết bị ngang hàng thứ  $i$  là  $u_i$ , và tốc độ tải xuống của liên kết truy nhập thiết bị ngang hàng thứ  $i$  là  $d_i$ . Chúng ta cũng ký hiệu kích cỡ của tệp được phân bổ (tính bằng bit) là  $F$  và số lượng thiết bị ngang hàng muốn nhận được bản sao của tệp là  $N$ . Thời gian

phân bố là thời gian thực hiện lấy được bản sao của tệp đối với tất cả  $N$  thiết bị ngang hàng. Trong phân tích thời gian phân bố dưới đây, đối với cả hai kiến trúc client-server và P2P, chúng ta sẽ đơn giản hóa (tuy nhiên vẫn chính xác) bằng giả thiết rằng lõi Internet có đủ thura băng thông, sao cho tất cả nghẽn cỗ chai chỉ xảy ra tại phần truy nhập mạng. Chúng ta cũng giả sử rằng máy chủ và máy khách không tham gia vào bất cứ ứng dụng khác nào, để cho tất cả băng thông truy nhập đường lên và đường xuống của họ có thể hoàn toàn dành riêng cho việc phân bố tệp.

Đầu tiên chúng ta xác định thời gian phân bố cho kiến trúc client-server, được biểu diễn bằng  $D_{cs}$ . Trong kiến trúc client-server, không có thiết bị ngang hàng nào hỗ trợ phân bố tệp. Chúng ta có các quan sát sau:

- Máy chủ phải truyền một bản sao của tệp tới tất cả  $N$  thiết bị ngang hàng. Do đó, máy chủ phải truyền  $NF$  bits. Do tốc độ tải lên của máy chủ là  $u_s$ , thời gian phân bố tệp phải ít nhất là  $NF/u_s$ .
- Ký hiệu  $d_{min}$  là tốc độ tải xuống thấp nhất của thiết bị ngang hàng, tức là  $d_{min} = \min(d_1, d_2, \dots, d_N)$ . Thiết bị ngang hàng với tốc độ tải xuống thấp nhất không thể nhận được tất cả  $F$  bit của tệp trong khoảng thời gian nhỏ hơn  $F/d_{min}$  giây. Vì vậy thời gian phân bố tối thiểu là  $F/d_{min}$  giây.

Kết hợp hai quan sát trên, chúng ta có:

$$D_{cs} \geq \max\left\{\frac{NF}{u_s}, \frac{F}{d_{min}}\right\} \quad (2.1)$$

Đây chính là giới hạn dưới của thời gian phân bố tối thiểu đối với kiến trúc client-server. Máy chủ có thể lập lịch truyền dẫn của nó để đạt được giới hạn này. Do đó có thể lấy giới hạn này như là thời gian phân bố tệp thực tế, tức là

$$D_{cs} = \max\left\{\frac{NF}{u_s}, \frac{F}{d_{min}}\right\} \quad (2.2)$$

Chúng ta thấy từ phương trình (2.2) nếu  $N$  đủ lớn, thời gian phân bố tệp client-server được xác định bằng  $NF/u_s$ . Do đó, thời gian phân bố tăng tuyến tính với số lượng thiết bị ngang hàng  $N$ . Ví dụ, nếu số lượng thiết bị ngang hàng từ tuần này sang tuần sau tăng một

nghìn lần, từ một nghìn lên một triệu, thì thời gian yêu cầu để phân bố tệp cho tất cả các thiết bị ngang hàng tăng 1000 lần.

Bây giờ chúng ta sẽ thực hiện phân tích tương tự cho kiến trúc P2P, mà mỗi thiết bị ngang hàng có thể hỗ trợ máy chủ trong quá trình phân bố tệp. Cụ thể, khi thiết bị ngang hàng nhận một số dữ liệu tệp nào đó, nó có thể sử dụng dung lượng tải lên của bản thân nó để phân bổ lại dữ liệu cho các thiết bị ngang hàng khác. Tính toán thời gian phân bố cho kiến trúc P2P phức tạp hơn so với kiến trúc client-server, do thời gian phân bố phụ thuộc vào từng thiết bị ngang hàng phân bố một phần tệp cho các thiết bị ngang hàng khác như thế nào. Tuy nhiên, có thể nhận được một biểu diễn đơn giản cho thời gian phân bố tối thiểu như sau đây. Trước hết chúng ta thực hiện các quan sát sau:

- Tại thời điểm bắt đầu phân bố, chỉ có máy chủ có tệp. Để lấy tệp này cho cộng đồng thiết bị ngang hàng, máy chủ phải gửi từng bit của tệp ít nhất một lần ra liên kết truy nhập. Do đó, thời gian phân bố tệp phải ít nhất là  $F/u_s$ . (Khác với mô hình client-server, mỗi bit được máy chủ gửi một lần có thể không được máy chủ gửi lại nữa, do các thiết bị ngang hàng có thể phân bổ lại các bit giữa chúng với nhau).
- Như với kiến trúc client-server, thiết bị ngang hàng với tốc độ tải xuống thấp nhất không thể nhận được tất cả  $F$  bit của tệp trong khoảng thời gian nhỏ hơn  $F/d_{min}$  giây. Vì vậy thời gian phân bố tối thiểu là  $F/d_{min}$  giây.
- Cuối cùng, nhận thấy rằng tổng số dung lượng tải lên của hệ thống tổng thể bằng tốc độ tải lên của máy chủ cộng với các tốc độ tải lên của từng thiết bị ngang hàng riêng rẽ, có nghĩa là,  $u_{total} = u_s + u_1 + u_2 + \dots + u_N$ . Hệ thống phải phân phát (tải lên)  $F$  bit tới từng thiết bị trong  $N$  thiết bị ngang hàng, vì vậy nó phân phát tổng cộng  $NF$  bit. Việc này không thể thực hiện tại tốc độ cao hơn  $u_{total}$ . Do đó, thời gian phân bố tối thiểu cũng ít nhất là  $NF/(u_s + u_1 + u_2 + \dots + u_N)$ .

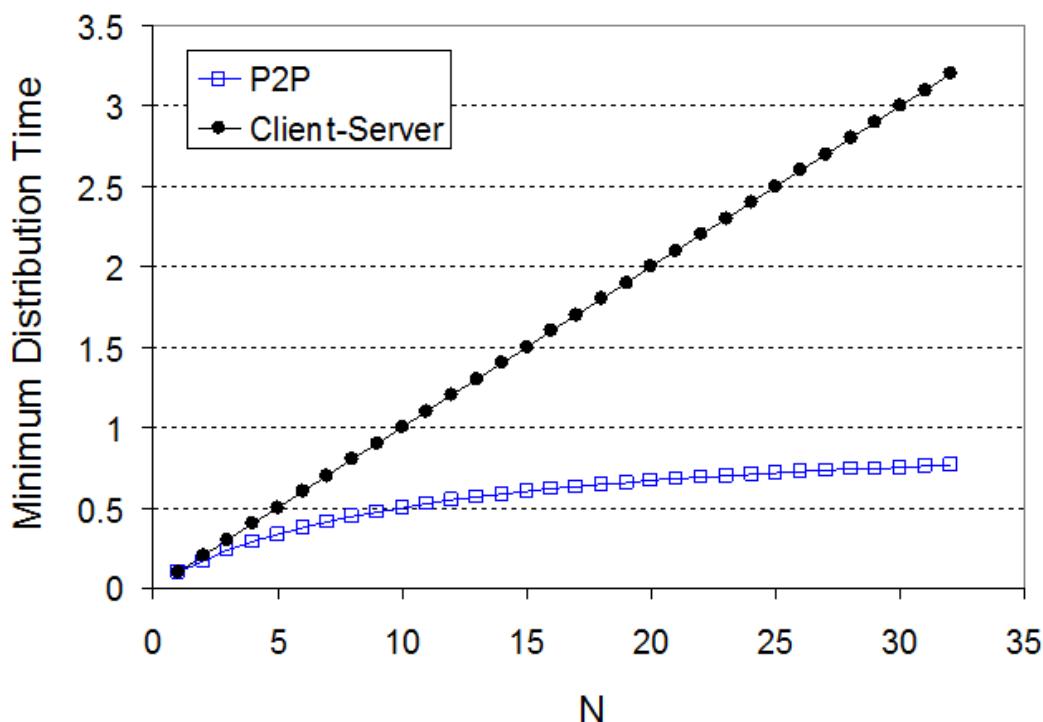
Kết hợp các quan sát trên, chúng ta nhận được thời gian phân bố tối thiểu đối với P2P, được ký hiệu là  $D_{P2P}$ .

$$D_{P2P} \geq \max \left\{ \frac{F}{u_s}, \frac{F}{d_{min}}, \frac{NF}{u_s + \sum_{i=1}^N u_i} \right\} \quad (2.3)$$

Phương trình (2.3) đưa ra giới hạn dưới của thời gian phân bổ tối thiểu đối với kiến trúc P2P. Nếu chúng ta hình dung mỗi thiết bị ngang hàng có thể phân bổ lại các bit ngay khi nó nhận được, thì sẽ tồn tại mô hình phân bổ thực sự đạt đến giá trị giới hạn dưới này. Trên thực tế, thường là khúc dữ liệu của tệp được phân bổ lại chứ không phải từng bit, phương trình (2.3) là một ước lượng tốt đối với thời gian phân bổ tối thiểu thực tế. Do đó, lấy giới hạn dưới của phương trình (2.3) cho thời gian phân bổ tối thiểu như sau

$$D_{P2P} = \max \left\{ \frac{F}{u_s}, \frac{F}{d_{min}}, \frac{NF}{u_s + \sum_{i=1}^N u_i} \right\} \quad (2.4)$$

Hình 2.22 so sánh thời gian phân bố tối thiểu cho kiến trúc client-server và P2P với giả thiết rằng tất cả các thiết bị ngang hàng có cùng tốc độ tải lên  $u$ . Trong hình 5.2 chúng ta có  $F/u=1$  giờ,  $u_s = 10u$ , và  $d_{min} \geq u_s$ . Do đó, thiết bị ngang hàng có thể truyền toàn bộ tệp trong 1 giờ, tốc độ truyền dẫn của máy chủ gấp 10 lần tốc độ tải lên của thiết bị ngang hàng và (để đơn giản) tốc độ tải xuống của thiết bị ngang hàng được thiết lập đủ lớn để không bị ảnh hưởng.



Hình 2.22: Thời gian phân bố tối thiểu với kiến trúc P2P và client-server.

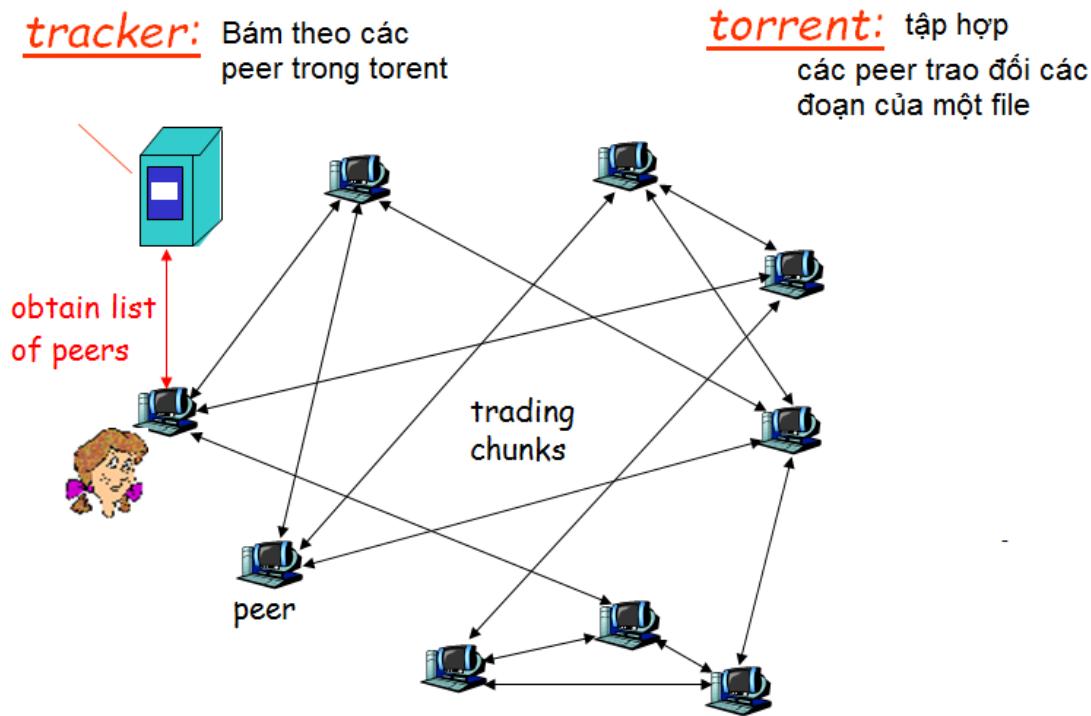
Chúng ta thấy từ Hình 2.22 đối với kiến trúc client-server, thời gian phân bổ tăng tuyến tính và không giới hạn khi số lượng thiết bị ngang hàng tăng lên. Tuy nhiên, đối với kiến trúc P2P, thời gian phân bổ tối thiểu không chỉ bao giờ cũng nhỏ hơn thời gian phân bổ đối với kiến trúc client-server; mà nó cũng nhỏ hơn 1 giờ cho bất cứ số lượng thiết bị ngang hàng  $N$  nào. Do đó, ứng dụng với kiến trúc P2P có thể tự mở rộng. Khả năng mở rộng qui mô này là hệ quả trực tiếp của các thiết bị ngang hàng trở thành thiết bị phân bố cũng như tiêu thụ thông tin (bit).

### **Giao thức phân bố tệp Bit Torrent:**

BitTorrent là giao thức P2P thông dụng để phân bố tệp. Theo ngôn ngữ chuyên môn BitTorrent, tập hợp tất cả các thiết bị ngang hàng tham gia phân bố một tệp cụ thể được gọi là *torrent* (dòng chảy). Các thiết bị trong torrent tải xuống các khía dữ liệu kích cỡ bằng nhau của tệp từ một thiết bị khác, với kích thước khía dữ liệu điển hình là 256 Kbyte. Khi thiết bị ngang hàng tham gia vào torrent lần đầu, nó không có khía dữ liệu nào. Theo thời gian nó dần dần thu thập nhiều hơn và nhiều hơn các khía dữ liệu. Trong khi tải xuống các khía dữ liệu, nó cũng tải lên các khía dữ liệu đến các thiết bị ngang hàng khác. Một khi thiết bị ngang hàng đã nhận được toàn bộ tệp, nó có thể (tự bản thân) rời bỏ torrent, hoặc vẫn ở lại trong torrent và tiếp tục tải lên các khía dữ liệu tới các thiết bị ngang hàng khác. Hơn nữa, bất kỳ thiết bị ngang hàng nào có thể rời bỏ torrent tại bất kỳ thời điểm nào với một tập khía dữ liệu, và sau đó lại gia nhập torrent.

Bây giờ chúng ta sẽ xem xét kĩ hơn các hoạt động của BitTorrent. Do BitTorrent là một giao thức và hệ thống phức tạp, chúng ta sẽ chỉ mô tả các cơ cấu quan trọng nhất của nó, và chỉ xem xét chi tiết một số cơ cấu; điều này sẽ cho phép chúng ta nắm bắt được vấn đề một cách tổng thể. Mỗi torrent có một nút hạ tầng gọi là bộ theo dõi (tracker). Khi thiết bị ngang hàng gia nhập torrent, nó tự đăng ký với bộ theo dõi và định kì thông báo với bộ theo dõi nó vẫn còn trong torrent. Bằng cách này, bộ theo dõi duy trì giám sát các thiết bị ngang hàng đang tham gia vào torrent. Một torrent cho trước có thể có ít hơn 10 hay nhiều hơn 1000 thiết bị ngang hàng tại bất cứ thời điểm nào.

Trong hình 2.23, khi thiết bị ngang hàng mới Alice gia nhập torrent, bộ theo dõi lựa chọn ngẫu nhiên một tập nhỏ các thiết bị ngang hàng (giả thiết cụ thể là 50) từ tập các thiết bị ngang hàng đang tham gia, và gửi địa chỉ IP của 50 thiết bị ngang hàng này cho Alice. Có danh sách các thiết bị ngang hàng này, Alice thử thiết lập đồng thời các kết nối TCP với tất cả các thiết bị ngang hàng trong danh sách. Gọi tất cả các thiết bị ngang hàng thiết lập kết nối TCP thành công với Alice là “thiết bị ngang hàng lân cận” (neighboring peers). (Trong hình 2.23 Alice chỉ có 3 thiết bị ngang hàng lân cận. Thông thường, số lượng thiết bị ngang hàng lân cận phải có nhiều hơn). Theo thời gian, một số thiết bị ngang hàng này có thể rời bỏ và các thiết bị ngang hàng khác (ngoài 50 thiết bị ban đầu) có thể thử thiết lập kết nối TCP với Alice. Như vậy thiết bị ngang hàng lân cận sẽ thay đổi theo thời gian.



Hình 2.23: Phân bổ tệp với BitTorrent.

Tại bất cứ thời gian cho trước nào, mỗi thiết bị ngang hàng sẽ có một tập nhỏ các khung dữ liệu của tệp, và các thiết bị ngang hàng khác nhau sẽ có các tập nhỏ khác nhau. Định kỳ theo thời gian, Alice sẽ hỏi các thiết bị lân cận (qua kết nối TCP) danh sách các khung dữ liệu mà họ có. Nếu Alice có  $L$  lân cận khác nhau, nó sẽ nhận được  $L$  danh sách các khung

dữ liệu. Với nhận biết này, Alice sẽ đưa ra yêu cầu (through qua kết nối TCP) đối với các khúc dữ liệu hiện thời nó chưa có.

Tại bất cứ thời điểm cho trước nào, mỗi thiết bị ngang hàng sẽ có một tập nhỏ các khúc dữ liệu và sẽ biết khúc dữ liệu nào lân cận (neighbor) của nó đang có. Với thông tin này, Alice sẽ có phải đưa ra hai quyết định quan trọng. Thứ nhất, khúc dữ liệu nào nó phải yêu cầu đầu tiên từ các lân cận của nó? Và thứ hai, nó phải gửi yêu cầu về khúc dữ liệu đến lân cận nào? Để quyết định yêu cầu khúc dữ liệu nào, Alice sử dụng kỹ thuật được gọi là *đầu tiên là hiếm nhất* (rarest first). Ý tưởng của nó là xác định khúc dữ liệu, trong số các khúc dữ liệu nó chưa có, đang có ít nhất từ các lân cận của nó (nghĩa là, các khúc dữ liệu có ít bản sao nhất trong các lân cận) và sau đó yêu cầu các khúc dữ liệu hiếm nhất trước tiên. Bằng cách này, các khúc dữ liệu hiếm nhất được phân bổ lại nhanh hơn, dẫn đến (tương đối) cân bằng số lượng các bản sao của từng khúc dữ liệu trong torrent.

Để xác định Alice đáp ứng yêu cầu nào, BitTorrent sử dụng giải thuật giao dịch khôn khéo. Ý tưởng cơ bản là Alice đưa ta mức ưu tiên cho các lân cận hiện tại đang cung cấp dữ liệu cho nó với tốc độ cao nhất. Cụ thể, đối với các lân cận của cô ta, Alice liên tục đo tốc độ nó nhận dữ liệu (bit) và xác định bốn thiết bị ngang hàng cung cấp bit với tốc độ cao nhất. Sau đó nó đáp lại bằng cách gửi các khúc dữ liệu tới bốn thiết bị ngang hàng này. Mỗi 10 giây, nó tính toán lại tốc độ và có thể thay đổi tập bốn thiết bị ngang hàng. Trong ngôn ngữ BitTorrent, bốn thiết bị ngang hàng này được gọi là mở (unchoked). Một điểm quan trọng nữa là cứ mỗi 30 giây, nó cũng chọn ngẫu nhiên một lân cận bổ sung và gửi khúc dữ liệu. Giả sử thiết bị ngang hàng được chọn ngẫu nhiên là Bob. Trong ngôn ngữ BitTorrent, Bob được gọi là được mở khả quan (optimistically unchoked). Vì rằng Alice gửi dữ liệu cho Bob, cô ấy có thể trở thành một trong bốn người tải lên đứng đầu của Bob, trong trường hợp này Bob có thể bắt đầu gửi dữ liệu cho Alice. Nếu tốc độ dữ liệu Bob gửi cho Alice đủ cao, đến lượt Bob sau đó có thể trở thành một trong bốn người tải lên đứng đầu của Alice. Nói một cách khác, cứ mỗi 30 giây, Alice sẽ chọn ngẫu nhiên một đối tác giao dịch mới và khởi tạo giao dịch với đối tác này. Nếu hai thiết bị khách hàng cùng thỏa mãn với giao dịch, chúng sẽ đưa nhau lên danh sách bốn người đứng đầu và tiếp tục giao dịch với nhau cho

đến khi một bên tìm thấy đối tác khác tốt hơn. Ở đây hiệu quả nhận được là các thiết bị ngang hàng có khả năng tải lên với tốc độ phù hợp tiến đến nhau. Việc lựa chọn lân cận ngẫu nhiên cũng cho phép các thiết bị ngang hàng mới nhận được khúc dữ liệu, do đó nó cũng có khả năng giao dịch. Tất cả các thiết bị ngang hàng khác ngoài năm thiết bị ngang hàng này (bốn thiết bị ngang hàng đứng đầu và một thiết bị thử nghiệm) đều “đóng” (choked), có nghĩa là chúng không nhận bất cứ khúc dữ liệu nào từ Alice. Ngoài ra, BitTorrent còn có một số cơ cấu khác, bao gồm các mảnh (khúc nhỏ), đường ống, lựa chọn ngẫu nhiên đầu tiên, chế độ kết thúc trò chơi, và chống rút ống (anti-snubbing).

Cơ cấu thúc đẩy giao dịch được mô tả trên thường đưa đến “ăn miếng trả miếng”. Các tài liệu chỉ ra rằng có thể tránh được mô hình thúc đẩy này. Tuy nhiên, hệ sinh thái BitTorrent đang rất thành công, với hàng triệu thiết bị ngang hàng hoạt động chia sẻ tệp đồng thời trên hàng trăm hay hàng nghìn torrent. Nếu BitTorrent được thiết kế không cần “ăn miếng trả miếng” (hay biến thể của nó), nhưng các điểm khác hoàn toàn giống như vậy, BitTorrent thậm chí có thể không tồn tại như hiện nay, do phần lớn người sử dụng có thể đã là người ăn theo.

### **2.6.2 Bảng hàm băm phân tán DHT**

Thành phần quan trọng của rất nhiều ứng dụng P2P và các ứng dụng phân tán khác là chỉ số (tức là cơ sở dữ liệu đơn giản), hỗ trợ các hoạt động tìm kiếm và cập nhật. Khi các chỉ số này được phân tán, các thiết bị ngang hàng có thể tạo lập lưu đệm nội dung và định tuyến thông minh các truy vấn giữa chúng với nhau. Do đánh chỉ số và tìm kiếm thông tin là thành phần quan trọng trong các hệ thống như vậy, chúng ta sẽ tìm hiểu một kỹ thuật đánh chỉ số và tìm kiếm thông dụng, gọi là Bảng hàm băm phân tán DHT.

Xem xét quá trình xây dựng cơ sở dữ liệu phân tán đơn giản trên số lượng lớn (có thể lên đến hàng triệu) thiết bị ngang hàng hỗ trợ đánh chỉ số và truy vấn. Thông tin được lưu giữ trong cơ sở dữ liệu của chúng ta sẽ bao gồm các cặp (khóa, giá trị). Ví dụ, khóa có thể là số chứng minh thư và các giá trị có thể tương ứng với tên người; trong trường hợp này, cặp khóa-giá trị điển hình là (156-45-7081, Nguyen). Hay khóa có thể là tên nội dung (như

tên bộ phim, album, phần mềm), và giá trị có thể là địa chỉ IP tại đó nội dung được lưu trữ; trong trường hợp này ví dụ cặp khóa-giá trị là (God father, 203.17.123.38). Thiết bị ngang hàng truy vấn cơ sở dữ liệu của chúng ta bằng cách cung cấp khóa: nếu có cặp (khóa, giá trị) trong cơ sở dữ liệu phù hợp với khóa, cơ sở dữ liệu trả lại cặp phù hợp cho thiết bị ngang hàng truy vấn. Như vậy, ví dụ, nếu cơ sở dữ liệu lưu trữ số chứng minh thư và tên người tương ứng, thiết bị ngang hàng có thể truy vấn số chứng minh thư cụ thể, và cơ sở dữ liệu trả lại tên của người có số chứng minh thư này. Các thiết bị ngang hàng cũng có khả năng chèn cặp (khóa, giá trị) vào cơ sở dữ liệu. Một bảng băm (hash table) là một cấu trúc dữ liệu ánh xạ giữa khóa và giá trị, tức là tương ứng với một khóa bảng băm sẽ trả về một giá trị. Để thực hiện việc ánh xạ bảng băm sử dụng hàm băm (hash function) tính toán vị trí lưu giá trị dựa trên khóa. Hàm băm phải đảm bảo: tránh xung đột và dễ dàng thực hiện. Tránh xung đột nghĩa là ánh xạ giữa không gian khóa và không gian địa chỉ phải đều và ngẫu nhiên đến mức có thể, ngoài ra thuật toán băm phải đơn giản, hiệu quả.

Xây dựng một cơ sở dữ liệu như vậy là đơn giản đối với kiến trúc client-server, mà tất cả cặp (khóa, giá trị) được lưu trữ trong một máy chủ trung tâm. Phương án tập trung cũng đã được thực thi trong hệ thống P2P trước kia như Napster. Nhưng vấn đề trở nên thách thức hơn và thú vị hơn nhiều trong hệ thống phân tán bao gồm từ hàng triệu thiết bị ngang hàng kết nối với nhau không có ủy quyền trung tâm. Trong hệ thống P2P, chúng ta muốn phân tán cặp (khóa, giá trị) thông qua tất cả các thiết bị ngang hàng. Sao cho mỗi thiết bị ngang hàng chỉ giữ một tập nhỏ của tổng số các cặp (khóa, giá trị). Một phương án đơn giản xây dựng cơ sở dữ liệu P2P như vậy là (1) rải ngẫu nhiên các cặp (khóa, giá trị) đến các thiết bị ngang hàng và (2) từng thiết bị ngang hàng bảo quản danh sách địa chỉ IP của tất cả các thiết bị ngang hàng tham gia. Bằng cách này, thiết bị ngang hàng truy vấn có thể gửi truy vấn tới tất cả các thiết bị ngang hàng khác, và thiết bị ngang hàng chứa cặp (khóa, giá trị) phù hợp với khóa có thể đáp ứng với cặp phù hợp của chúng. Tất nhiên do tính qui mô mạng, phương pháp này hoàn toàn không thể thực hiện được vì nó có thể yêu cầu mỗi thiết bị ngang hàng phải theo dõi tất cả các thiết bị ngang hàng khác (có thể hàng triệu) và tồi tệ hơn nữa, mỗi truy vấn được gửi đến tất cả các thiết bị ngang hàng.

Chúng ta sẽ mô tả một phương án thực tế khác để thiết kế cơ sở dữ liệu P2P. Đầu tiên gán một định danh cho mỗi thiết bị ngang hàng, mỗi định danh là một số nguyên trong dài  $[0, 2^n - 1]$  với một số  $n$  cố định. Để ý rằng mỗi định danh như thế có thể được biểu diễn bằng  $n$ -bit. Yêu cầu mỗi khóa là một số nguyên trong dài này. Chúng ta có thể quan sát rằng các khóa điển hình mô tả ở trên (số chứng minh thư và tên nội dung) không phải là các số nguyên. Để tạo các số nguyên cho các khóa này, chúng ta sẽ sử dụng hàm băm ánh xạ từng khóa (như số chứng minh thư) tới số nguyên trong dài  $[0, 2^n - 1]$ . Hàm băm là hàm nhiều-đến-một (many-to-one) mà hai đầu vào khác nhau có thể có cùng một đầu ra (cùng một số nguyên), nhưng khả năng có cùng một đầu ra là vô cùng nhỏ. Hàm băm được giả thiết là sẵn sàng công khai cho tất cả các thiết bị ngang hàng trong hệ thống. Từ giờ trở về sau, khi chúng ta đề cập đến “khóa”, chúng ta đề cập đến băm của khóa nguyên bản. Như vậy, ví dụ, nếu khóa nguyên bản là “God father”, khóa sẽ là số nguyên bằng băm của “God father”. Cũng vậy, do chúng ta sử dụng băm của khóa, thay cho bản thân khóa, chúng ta từ giờ sẽ đề cập đến cơ sở dữ liệu phân tán như là *bảng hàm băm phân tán DHT*.

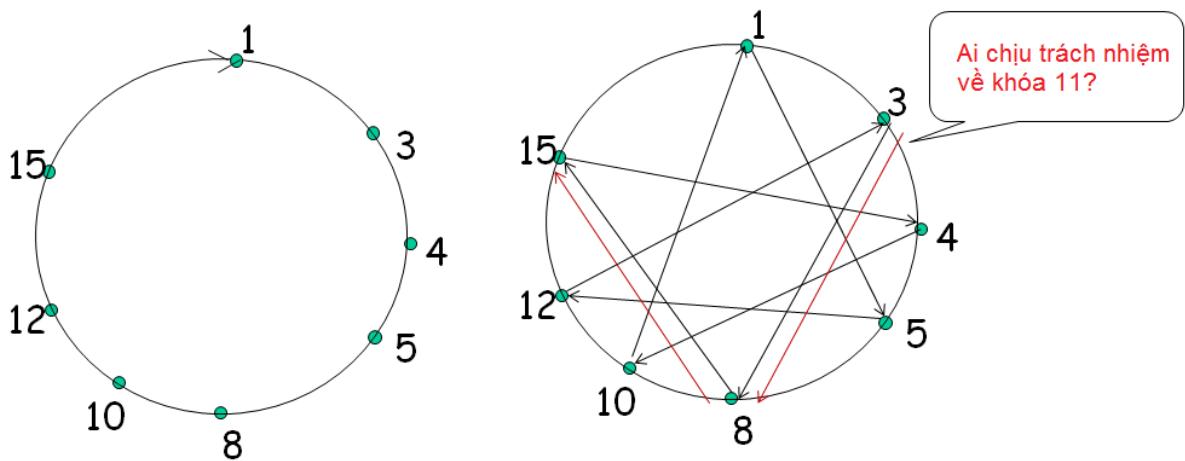
Bây giờ chúng ta sẽ xem xét vấn đề lưu trữ cặp (khóa, giá trị) trong DHT. Vấn đề trọng tâm ở đây là xác định qui luật gán khóa cho thiết bị ngang hàng. Giả định mỗi thiết bị ngang hàng có một định danh số nguyên và mỗi khóa cũng là một số nguyên trên cùng một dài, giải pháp tự nhiên là gán mỗi cặp (khóa, giá trị) tới thiết bị ngang hàng có định danh gần khóa nhất. Để thực thi phương án này, chúng ta sẽ cần định nghĩa “gần nhất” có nghĩa là gì, ở đây có thể có rất nhiều quy ước. Để thuận tiện, định nghĩa thiết bị ngang hàng gần nhất như thiết bị ngay sau của khóa. Để hiểu sâu hơn, chúng ta cùng lấy một ví dụ. Giả sử  $n = 4$  sao cho tất cả thiết bị ngang hàng và định danh khóa trong dài  $[0, 15]$ . Tiếp tục giả sử có tám thiết bị ngang hàng trong hệ thống có định danh 1, 3, 4, 8, 10, 12, và 15. Cuối cùng, giả sử chúng ta muốn lưu trữ cặp khóa-giá trị (11, Nguyen) vào trong một từ tám thiết bị ngang hàng. Nhưng trong thiết bị ngang hàng nào? Sử dụng quy ước “gần nhất”, do thiết bị ngang hàng 12 là thiết bị ngay sau đối với khóa 11, do đó chúng ta lưu trữ cặp (11, Nguyen) trong thiết bị ngang hàng 12. Để hoàn thiện định nghĩa của chúng ta về “gần nhất”, nếu khóa bằng chính xác một định danh thiết bị ngang hàng, chúng ta sẽ lưu trữ cặp (khóa,

giá trị) trong thiết bị ngang hàng trùng lặp đó; và nếu khóa lớn hơn tất cả định danh thiết bị ngang hàng, chúng ta sử dụng quy ước modulo- $2^n$ , lưu trữ cặp (khóa, giá trị) trong thiết bị ngang hàng với định danh nhỏ nhất.

Giả sử thiết bị ngang hàng, Alice, muốn chèn một cặp (khóa, giá trị) vào DHT. Về mặt quan niệm, điều này là đơn giản: đầu tiên xác định thiết bị ngang hàng mà định danh của nó gần khóa nhất, sau đó gửi bản tin tới thiết bị ngang hàng đó, chỉ dẫn nó lưu trữ cặp (khóa, giá trị). Nhưng Alice xác định thiết bị ngang hàng gần khóa nhất như thế nào? Nếu Alice duy trì theo dõi của tất cả các thiết bị ngang hàng trong hệ thống (ID của thiết bị ngang hàng và địa chỉ IP tương ứng), nó có thể xác định tại chỗ thiết bị ngang hàng gần nhất. Nhưng giải pháp như vậy yêu cầu mỗi thiết bị ngang hàng phải duy trì theo dõi tất cả các thiết bị ngang hàng khác trong DHT – điều này hoàn toàn không thực tế đối với hệ thống qui mô lớn với hàng triệu thiết bị ngang hàng.

### **DHT vòng:**

Để giải quyết vấn đề quy mô, chúng ta sẽ xem xét tổ chức các thiết bị ngang hàng thành một vòng. Trong sắp xếp dạng vòng, mỗi thiết bị ngang hàng chỉ duy trì theo dõi thiết bị ngay sau nó (modulo  $2^n$ ). Ví dụ một vòng như vậy được đưa ra trong Hình 2.24 (a). Trong ví dụ này,  $n$  bằng 4 và cũng có tám thiết bị ngang hàng như ví dụ trước. Mỗi thiết bị ngang hàng chỉ nhận biết thiết bị ngay sau của nó; ví dụ, thiết bị ngang hàng 5 biết địa chỉ IP và định danh của thiết bị ngang hàng 8 nhưng không cần thiết phải biết về các thiết bị ngang hàng khác có thể có trong DHT. Việc bố trí vòng các thiết bị ngang hàng như vậy là trường hợp đặc biệt của *mạng che phủ* (overlay network). Trong mạng che phủ, các thiết bị ngang hàng tạo thành mạng logic trừu tượng lưu trú bên trên mạng máy tính “nền” bao gồm từ các liên kết vật lý, bộ định tuyến, và máy chủ. Các liên kết trong mạng che phủ không phải là các liên kết vật lý, mà chỉ đơn giản là các liên kết ảo giữa các cặp thiết bị ngang hàng. Trong mạng che phủ Hình 2.24 (a) có 8 thiết bị ngang hàng và 8 liên kết che phủ; trong mạng che phủ Hình 2.24 (b) có 8 thiết bị ngang hàng và 16 liên kết che phủ. Một liên kết che phủ đơn thường sử dụng nhiều liên kết vật lý và các bộ định tuyến trong mạng nền.



Hình 2.24: (a) DHT vòng, thiết bị ngang hàng 3 muốn xác định ai chịu trách nhiệm đối với khóa 11. (b) DHT vòng với các đường nối tắt.

Sử dụng che phủ vòng Hình 2.24 (a), giả sử rằng thiết bị ngang hàng 3 muốn xác định thiết bị ngang hàng nào trong DHT chịu trách nhiệm đối với khóa 11 (hoặc để chèn hoặc để truy vấn cặp khóa-giá trị). Sử dụng che phủ vòng, thiết bị ngang hàng gốc (thiết bị ngang hàng 3) tạo ra bản tin “Ai chịu trách nhiệm đối với khóa 11?” và gửi bản tin này tới thiết bị ngay sau của nó là thiết bị ngang hàng 4. Bất cứ khi nào một thiết bị ngang hàng nhận được bản tin như vậy, do nó biết định danh của thiết bị ngay sau của nó, nên nó có thể xác định nó có trách nhiệm (có nghĩa là nó là gần nhất) đối với khóa đang được hỏi hay không. Nếu thiết bị ngang hàng không chịu trách nhiệm đối với khóa, nó chỉ đơn giản gửi bản tin đến thiết bị ngay sau của nó. Ví dụ, khi thiết bị ngang hàng 4 nhận được bản tin hỏi về khóa 11, nó xác định rằng nó không chịu trách nhiệm đối với khóa (vì rằng thiết bị ngay sau của nó gần với khóa hơn), như vậy nó chỉ chuyển bản tin tiếp đến thiết bị ngay sau của nó, gọi là thiết bị ngang hàng 5. Quá trình này tiếp tục cho đến khi bản tin đến thiết bị ngang hàng 12, xác định nó gần khóa 11 nhất. Tại điểm này, thiết bị ngang hàng 12 có thể gửi bản tin trả lại thiết bị ngang hàng gốc số 3, chỉ thị rằng nó chịu trách nhiệm đối với khóa 11.

DHT vòng cung cấp giải pháp rất thông minh nhằm giảm số lượng thông tin che phủ mỗi thiết bị ngang hàng phải quản lý. Đặc biệt, mỗi thiết bị ngang hàng chỉ nhận thức hai thiết bị ngang hàng, thiết bị ngay sau của nó và thiết bị ngay trước nó. (Ngầm định thiết bị ngang hàng biết được thiết bị ngay trước của nó, do thiết bị ngay trước này gửi bản tin).

Nhưng giải pháp nay đưa ra vấn đề mới. Mặc dù mỗi thiết bị ngang hàng chỉ nhận biết 2 thiết bị ngang hàng lân cận, để tìm nút chịu trách nhiệm đối với khóa (trong trường hợp xấu nhất) tất cả  $N$  nút trong DHT sẽ phải chuyển tiếp bản tin xung quanh vòng, do đó có trung bình  $N/2$  bản tin được gửi đi.

Do vậy, trong thiết kế DHT, có điểm cân bằng giữa số lượng lân cận mỗi thiết bị ngang hàng phải theo dõi và số lượng bản tin DHT cần phải gửi để giải quyết một truy vấn. Một mặt, nếu mỗi thiết bị ngang hàng theo dõi tất cả các thiết bị ngang hàng khác (che phủ dạng lưới), thì chỉ có một bản tin được gửi đi cho mỗi truy vấn, nhưng mỗi thiết bị ngang hàng phải theo dõi  $N$  thiết bị ngang hàng. Mặt khác, với DHT vòng, mỗi thiết bị ngang hàng chỉ nhận biết hai thiết bị ngang hàng, nhưng trung bình  $N/2$  bản tin được gửi đi cho mỗi truy vấn. May mắn là chúng ta có thể hoàn thiện thiết kế DHT sao cho số lượng lân cận của một thiết bị ngang hàng cũng như số lượng của bản tin cho mỗi truy vấn trong phạm vi chấp nhận được. Một trong những hoàn thiện như vậy là sử dụng che phủ vòng như nền tảng, nhưng bổ sung các “đường tắt” sao cho mỗi thiết bị ngang hàng không chỉ theo dõi thiết bị ngay sau của nó, mà cả một số lượng nhỏ các thiết bị ngang hàng được nối tắt rái rác trên vòng tròn. Ví dụ một DHT vòng như vậy với một số đường nối tắt được đưa ra trên Hình 2.24 (b). Các đường tắt được sử dụng để tiến hành định tuyến các bản tin truy vấn. Đặc biệt, khi thiết bị ngang hàng nhận bản tin truy vấn khóa, nó chuyển tiếp bản tin tới lân cận (lân cận ngay sau hay một trong những lân cận nối tắt) gần khóa nhất. Vì vậy, trong Hình 2.24 (b) khi thiết bị ngang hàng 4 nhận bản tin hỏi về khóa 11, nó xác định rằng thiết bị ngang hàng gần khóa nhất (trong các lân cận) là lân cận nối tắt 10 và sau đó chuyển tiếp bản tin thẳng đến thiết bị ngang hàng 10. Rõ ràng là các đường nối tắt có thể giảm đáng kể số lượng bản tin sử dụng để xử lý truy vấn.

Vấn đề tiếp theo là “Bao nhiêu lân cận nối tắt mỗi thiết bị ngang hàng cần phải có, và thiết bị ngang hàng nào phải là lân cận nối tắt?”. Vấn đề này đã nhận được sự quan tâm của cộng đồng nghiên cứu. Một điều quan trọng cần lưu ý là các nghiên cứu đã chứng minh được rằng DHT có thể được thiết kế sao cho cả số lượng lân cận của mỗi thiết bị ngang hàng lẫn số lượng bản tin mỗi truy vấn là  $O(\log N)$ , trong đó  $N$  là số lượng thiết bị ngang

hàng. Các thiết kế này cố gắng thỏa hiệp giữa giải pháp cực trị sử dụng topo dạng lưới và topo che phủ vòng.

### **Peer churn:**

Trong hệ thống P2P, thiết bị ngang hàng có thể vào và ra không báo trước. Do đó, khi thiết kế DHT, chúng ta phải quan tâm đến duy trì che phủ DHT bằng sự có mặt của *peer churn*. Để đạt được bức tranh toàn cảnh thấu hiểu điều này có thể được thiết lập như thế nào chúng ta sẽ lại xem xét DHT vòng trong Hình 2.24 (a). Để xử lý peer churn, chúng ta sẽ yêu cầu mỗi thiết bị ngang hàng theo dõi (tức là biết địa chỉ IP) của thiết bị ngay sau thứ nhất và thứ hai của nó; ví dụ, thiết bị ngang hàng 4 bây giờ theo dõi cả thiết bị ngang hàng 5 và 8. Chúng ta cũng sẽ yêu cầu mỗi thiết bị ngang hàng định kỳ kiểm tra hai thiết bị ngay sau của nó còn sống (ví dụ, bằng cách định kỳ gửi các bản tin ping đến chúng và yêu cầu chúng phản hồi). Tiếp tục xem xét DHT được duy trì như thế nào khi một thiết bị ngang hàng đột ngột rời bỏ. Ví dụ, giả sử thiết bị ngang hàng 5 trong Hình 2.24 (a) đột ngột rời bỏ. Trong trường hợp này, hai thiết bị ngang hàng đãng trước thiết bị ngang hàng bỏ đi (4 và 3) biết được 5 đã bỏ đi, do nó không còn phản hồi bản tin ping. Các thiết bị ngang hàng 4 và 3 vì vậy cần cập nhật thông tin trạng thái thiết bị ngay sau của chúng. Bây giờ chúng ta sẽ xem xét thiết bị ngang hàng 4 cập nhật trạng thái của nó như thế nào:

- Thiết bị ngang hàng 4 thay thế thiết bị ngay sau thứ nhất của nó (thiết bị ngang hàng 5) bằng thiết bị ngay sau thứ hai (thiết bị ngang hàng 8).
- Thiết bị ngang hàng 4 sau đó yêu cầu thiết bị ngay sau thứ nhất mới (thiết bị ngang hàng 8) về định danh và địa chỉ IP của thiết bị ngay sau của nó (thiết bị ngang hàng 10). Thiết bị ngang hàng 4 sau đó lấy thiết bị ngang hàng 10 làm thiết bị ngay sau thứ hai.

Chúng ta đã đưa ra ngắn gọn cần làm gì khi một thiết bị ngang hàng rời đi, bây giờ chúng ta sẽ xem xét điều gì xảy ra khi một thiết bị ngang hàng muốn gia nhập vào DHT. Giả sử thiết bị ngang hàng với định danh 13 muốn gia nhập DHT, và tại thời điểm gia nhập, nó chỉ biết về sự tồn tại của thiết bị ngang hàng 1 trong DHT. Thiết bị ngang hàng đầu tiên sẽ gửi cho thiết bị ngang hàng 13 một bản tin, như “Thiết bị ngay trước và thiết bị ngay sau

của thiết bị ngang hàng 13 là ai?" Bản tin này được chuyển tiếp qua DHT cho đến khi nó tới được thiết bị ngang hàng 12, người thực sự biết rằng nó sẽ là thiết bị ngay trước của thiết bị ngang hàng 13 và thiết bị ngay sau của nó, thiết bị ngang hàng 15, sẽ trở thành thiết bị ngay sau của thiết bị ngang hàng 13. Thiết bị ngang hàng 13 bây giờ có thể gia nhập DHT bằng cách lấy thiết bị ngang hàng 15 là thiết bị ngay sau và thông báo thiết bị ngang hàng 12 là nó phải đổi thiết bị ngay sau thành 13.

DHT đã được sử dụng vô cùng rộng rãi trong thực tế. Ví dụ, BitTorrent sử dụng Kademia DHT để tạo lập bộ theo dõi phân tán. Trong BitTorrent, khóa là định danh torrent và giá trị là địa chỉ IP của tất cả các thiết bị ngang hàng hiện đang tham gia trong torrent. Theo cách này, bằng truy vấn DHT với định danh torrent, thiết bị ngang hàng BitTorrent mới tham gia có thể xác định thiết bị ngang hàng chịu trách nhiệm đổi với định danh (có nghĩa là theo dõi các thiết bị ngang hàng trong torrent). Sau khi đã tìm được thiết bị ngang hàng này, thiết bị ngang hàng mới tham gia có thể truy vấn nó về danh sách các thiết bị ngang hàng khác trong torrent. DHT cũng được sử dụng rộng rãi trong hệ thống chia sẻ tệp eMule để xác định vị trí nội dung trong các thiết bị ngang hàng.

Các mạng P2P DHT có khả năng cung cấp một kiến trúc định tuyến hiệu quả, tự tổ chức, quy mô lớn, kết hợp với khả năng chịu lỗi, cân bằng tải và xác định vị trí rõ ràng. Tuy nhiên định tuyến trên DHT cũng có một số vấn đề cần quan tâm. Do các nút mạng gia nhập và rời mạng vào các thời điểm bất kỳ, không dự báo trước được, dẫn đến mạng không ổn định (churn). Điều này ảnh hưởng rất lớn đến hiệu năng định tuyến: do mạng thay đổi liên tục, chi phí duy trì ổn định cho các bảng định tuyến tại các nút sẽ tăng; hơn nữa do định tuyến dựa trên DHT vì vậy thiếu hỗ trợ cho việc truy tìm từ khóa và các truy vấn phức tạp. Tuy nhiên với việc phát triển một nền tảng thống nhất cho các DHT khác nhau sẽ làm cho mạng có cấu trúc ngày càng hấp dẫn hơn. Một nền tảng như vậy sẽ cung cấp một API dựa trên định tuyến trên cơ sở khóa (key based routing), kết hợp với mô hình dịch vụ DHT cơ bản để triển khai các ứng dụng DHT thuận tiện và dễ dàng.

### **2.6.3 Ứng dụng thoại Internet**

Thoại Internet Skype là một ứng dụng P2P thông dụng, thường hàng chục triệu người sử dụng kết nối tại bất cứ thời điểm nào. Ngoài việc cung cấp dịch vụ thoại Internet PC-to-PC Skype còn đưa ra dịch vụ thoại PC-to-phone, dịch vụ thoại phone-to-PC và dịch vụ hội nghị video PC-to-PC.

Skype sử dụng kỹ thuật P2P bằng nhiều cách sáng tạo, minh họa một cách độc đáo P2P có thể sử dụng trong các ứng dụng khác ngoài phân bố nội dung và chia sẻ tệp như thế nào. Cũng như nhắn tin, thoại Internet PC-to-PC vốn dĩ từ bản chất bên trong đã là P2P, vì trong tâm điểm của ứng dụng, cặp người sử dụng (thiết bị ngang hàng) truyền thông thời gian thực với nhau. Nhưng Skype cũng sử dụng các kỹ thuật P2P cho hai chức năng quan trọng khác: xác định vị trí người sử dụng và hỗ trợ chuyển đổi địa chỉ mạng NAT.

Không chỉ các giao thức Skype là quyền sở hữu riêng, mà tất cả các gói tin được truyền đi (các gói tin thoại và điều khiển) đều được mã hóa. Tuy nhiên, chúng ta vẫn có thể tìm hiểu Skype làm việc tổng thể như thế nào. Giống như FastTrack, các nút mạng Skype được tổ chức thành mạng che phủ phân cấp, mỗi thiết bị ngang hàng được phân loại thành siêu thiết bị ngang hàng hay thiết bị ngang hàng gốc. Skype bao gồm chỉ số ánh xạ tên người sử dụng Skype tới địa chỉ IP hiện thời (và số của cổng). Chỉ số này được phân bố trên các siêu thiết bị ngang hàng. Khi Alice muốn gọi cho Bob, máy khách Skype của Alice tìm kiếm chỉ số phân bố để xác định địa chỉ IP hiện thời của Bob. Bởi vì giao thức Skype là thuộc quyền sở hữu riêng, hiện nay vẫn chưa rõ ánh xạ chỉ số được tổ chức trên các siêu thiết bị ngang hàng như thế nào, mặc dù có một số dạng tổ chức DHT có thể là giải pháp tiềm năng.

Các kỹ thuật P2P cũng được sử dụng trong chuyển tiếp Skype, nó rất hữu dụng để thiết lập cuộc gọi giữa các trạm chủ trong mạng nhà. Nhiều cấu hình mạng nhà cung cấp truy nhập vào Internet thông qua bộ định tuyến (thường là bộ định tuyến không dây). Các bộ định tuyến này thực tế còn hơn cả bộ định tuyến, và thường bao hàm cả bộ chuyển đổi địa chỉ mạng NAT (Network Address Translator). NAT ngăn chặn trạm chủ bên ngoài mạng nhà khỏi tạo kết nối với trạm chủ bên trong mạng nhà. Nếu cả hai chủ gọi Skype đều

có NAT, thì sẽ xuất hiện vấn đề - không ai có thể chấp nhận cuộc gọi được khởi tạo bởi người khác, làm cho cuộc gọi có vẻ như không thể thực hiện. Có thể khéo léo sử dụng các siêu thiết bị ngang hàng và các bộ chuyển tiếp để giải quyết vấn đề này. Giả sử khi Alice báo vào mạng, cô ấy được gán vào một siêu thiết bị ngang hàng không có NAT. Alice có thể khởi tạo phiên đến siêu thiết bị ngang hàng của cô ấy do NAT của Alice chỉ không cho phép các phiên khởi tạo từ ngoài mạng nhà của Alice. Điều này cho phép Alice và siêu thiết bị ngang hàng của cô ta trao đổi các bản tin điều khiển trên phiên này. Điều tương tự cũng xảy ra với Bob khi anh ta báo vào mạng. Bây giờ, khi Alice muốn gọi cho Bob, cô ấy thông báo cho siêu thiết bị ngang hàng của cô ấy, và đến lượt siêu thiết bị ngang hàng của Alice thông báo cho siêu thiết bị ngang hàng của Bob, sau đó siêu thiết bị ngang hàng này thông báo cho Bob cuộc gọi đến của Alice. Nếu Bob chấp nhận cuộc gọi, hai siêu thiết bị ngang hàng lựa chọn siêu thiết bị ngang hàng không có NAT thứ ba – nút chuyển tiếp – công việc của nó sẽ là chuyển dữ liệu giữa Alice và Bob. Các siêu thiết bị ngang hàng của Alice và Bob sau đó chỉ dẫn cho Alice và Bob khởi tạo phiên với bộ chuyển tiếp. Sau đó Alice gửi dữ liệu thoại đến bộ chuyển tiếp trên kết nối Alice-đến-bộ chuyển tiếp (nó đã được Alice khởi tạo), và bộ chuyển tiếp truyền các gói tin này trên kết nối bộ chuyển tiếp-đến-Bob (nó đã được Bob khởi tạo); các gói tin từ Bob đến Alice lưu thông trên cùng hai kết nối này theo chiều ngược lại. Như vậy, Bob và Alice có kết nối toàn trình theo yêu cầu thậm chí không bên nào có thể chấp nhận phiên bắt nguồn từ bên ngoài mạng LAN của nó. Việc sử dụng bộ chuyển tiếp minh họa thiết kế ngày càng thông minh của hệ thống P2P, ở đó các thiết bị ngang hàng thực hiện các dịch vụ hệ thống lõi cho các bên khác (dịch vụ chỉ số và chuyển tiếp là hai ví dụ) trong khi đồng thời chúng cũng sử dụng dịch vụ người sử dụng cuối cho bản thân (như tải tệp, thoại IP) được cung cấp bởi hệ thống P2P.

Skype là một ứng dụng Internet thành công rộng rãi, với hàng chục triệu người sử dụng. Việc áp dụng nhanh và rộng rãi của Skype, cũng như chia sẻ tệp P2P, Web, nhắn tin là minh chứng cho sự thông minh của thiết kế kiến trúc tổng thể Internet. Bản thân thiết kế Internet này không thể nhìn thấy trước sự đa dạng và ngày càng mở rộng của các ứng dụng Internet được phát triển. Các dịch vụ mạng cung cấp cho các ứng dụng Internet – truyền tải

gói dữ liệu vô hướng (UDP), chuyên tải gói dữ liệu tin cậy có hướng (TCP), giao diện socket, đánh địa chỉ, và đặt tên (DNS), và các dịch vụ khác – đã chứng minh là có thể cho phép hàng nghìn ứng dụng được phát triển. Do các ứng dụng này đã được xếp lớp trên cùng của bốn lớp tồn tại bên dưới của chồng giao thức Internet, chúng chỉ liên quan đến phát triển client-server mới như một phần mềm peer-to-peer để sử dụng tại hệ thống đầu cuối. Đến lượt điều này lại cho phép các ứng dụng này triển khai và áp dụng nhanh chóng.

## 2.7 Kết luận chương

Các ứng dụng mạng đa dạng là lý do tồn tại của Internet. Từ khi Internet ra đời, đã có rất nhiều giao thức được thiết kế để hỗ trợ sự phát triển và kiến tạo ra các ứng dụng mạng. Những ứng dụng đầu tiên dựa trên nền kỹ tự như e-mail thuận, truy nhập máy tính từ xa, truyền tệp và chát kỹ tự đã trở thành thông dụng từ thập kỷ 1970. Từ giữa thập kỷ 1990 những ứng dụng điển hình là WWW (World Wide Web), tìm kiếm và thương mại điện tử. Tiếp đến là những ứng dụng như nhắn tin tức thời với danh sách lưu sẵn và chia sẻ tệp P2P. Trong chương này đã trình bày chi tiết một số ứng dụng điển hình như Web, e-mail, DNS, phân phối tệp P2P và điện thoại Internet P2P, cũng như các giao thức được sử dụng để cung cấp các ứng dụng đó.

## **CHƯƠNG 3: CÁC GIAO THỨC ỨNG DỤNG ĐA PHƯƠNG TIỆN**

### **3.1 Giới thiệu chung**

Thuật ngữ đa phương tiện được sử dụng để mô tả một dịch vụ có nhiều hình thức nội dung, là sự kết hợp văn bản (text), âm thanh (audio), ảnh tĩnh, hoạt hình, video hoặc các dạng nội dung tương tác khác. Việc đưa kỹ thuật đa phương tiện (multimedia) vào các ứng dụng truyền thông trên mạng giúp tạo ra nhiều ứng dụng phong phú. Chẳng hạn hộp thư điện tử ngày nay có thể không chỉ là văn bản mà còn bao gồm tiếng nói, hình ảnh. Các trang web cũng trở nên sinh động, hấp dẫn hơn hẳn khi kèm theo kỹ thuật multimedia...

Các ứng dụng đa phương tiện — những ứng dụng liên quan đến video, âm thanh và dữ liệu — phổ biến hiện nay được chia thành hai loại: ứng dụng tương tác và các ứng dụng phát trực tuyến (streaming).

*Các ứng dụng tương tác* yêu cầu nghiêm ngặt về thời gian thực. Lớp ứng dụng này cho phép mọi người sử dụng audio/video để truyền thông với nhau trên thời gian thực. Audio tương tác thời gian thực trên Internet thường được xem như thoại Internet. Thoại Internet có thể cung cấp dịch vụ chuyển mạch PBX, thoại nội bộ và đường dài với chi phí rất thấp. Nó cũng tạo điều kiện để triển khai các dịch vụ mới mà mạng chuyển mạch truyền thống không dễ dàng hỗ trợ như tìm kiếm hiện diện, truyền thông nhóm, lọc chủ gọi, tích hợp thoại Web, ... Hiện nay đã sẵn có rất nhiều các sản phẩm thoại Internet. Ví dụ, người sử dụng Skype có thể thực hiện cuộc gọi thoại PC-to-phone và PC-to-PC. Cùng với đó còn có video tương tác thời gian thực, cũng còn gọi là hội nghị video, cá nhân có thể truyền thông, nghe và nhìn. Hiện nay đang sẵn có rất nhiều sản phẩm video tương tác thời gian thực cho Internet bao gồm Microsoft's Netmeeting, Skype video, và các sản phẩm Polycom.

*Các ứng dụng phát trực tuyến* thường được sử dụng trong lĩnh vực giải trí hoặc dạy học, dùng để lưu trữ các tệp tin (file video) hoặc các bài học, cung cấp cho người dùng các tiện ích như tìm kiếm, liệt kê, và khả năng hiện thị hoặc hiện thị lại các dữ liệu video theo yêu cầu. Nó cung cấp các luồng âm thanh hoặc video từ một máy chủ cho một máy khách và được đặc trưng bởi các sản phẩm thương mại như RealAudior. Phát trực tuyến video, được định hình bởi YouTube, đã trở thành một trong những cách thức lưu lượng truy cập thống trị trên Internet.

*Live Streaming:* là các dữ liệu video được convert trực tiếp từ các nguồn cung cấp dữ liệu theo thời gian thực (máy camera, microphone, các thiết bị phát dữ liệu video...). Các dữ liệu này sẽ được multimedia phát quảng bá thành các kênh người dùng sẽ có

quyền truy nhập bất kỳ kênh ưa thích nào để hiện thị dữ liệu mà không được thực hiện các thao tác, tua, dừng,...trên các dữ liệu đó (giống như tivi truyền thống). Ta có thể xem Live Streaming thông qua Facebook Live, Youtube Live,...

*Video on Demand (VoD):* Đây là các file đa phương tiện được lưu trên máy chủ và sẵn sàng hiển thị khi người dùng truy cập. Hiện nay có hàng nghìn địa điểm cung cấp trực tuyến audio và video, bao gồm CNN, YouTube, Microsoft video, NetFlix.... Người dùng có thể thực hiện các thao tác như tua nhanh/chậm, chuyển tập,... đối với loại streaming video này.

Vì các ứng dụng phát trực tuyến thiếu sự tương tác giữa người với người, nên yêu cầu ít hơn một chút về thời gian thực cho các giao thức hạ tầng. Tuy nhiên, yêu tố thời gian thực vẫn quan trọng. Ví dụ khi taấn phím “play” là muốn video bắt đầu phát ngay và khi video bắt đầu phát, các gói tin mà bị trễ sẽ khiến nó bị đình trệ hoặc tạo ra một số loại suy giảm hình ảnh. Vì vậy, các ứng dụng phát trực tuyến không hoàn toàn theo thời gian thực nhưng chúng vẫn có đủ điểm chung với các ứng dụng đa phương tiện tương tác để đảm bảo xem xét một giao thức hỗ trợ thời gian thực chung cho cả hai loại ứng dụng.

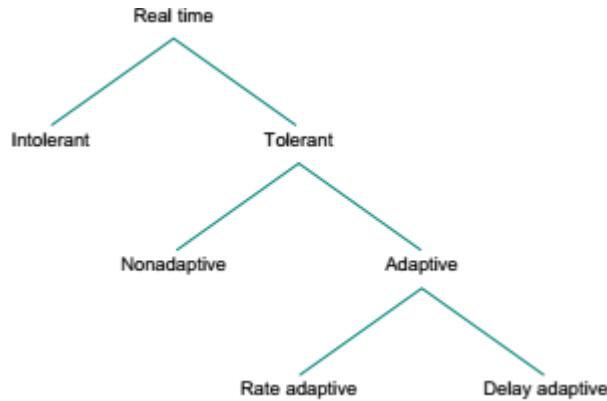
### **3.2 Các giao thức hỗ trợ thời gian thực luồng đa phương tiện**

#### **3.2.1 Yêu cầu đối với giao thức truyền tải luồng đa phương tiện**

Các yêu cầu của ứng dụng đa phương tiện khác nhiều so với các ứng dụng truyền thống trên mạng dữ liệu (hay còn gọi với một thuật ngữ là co dãn-elastic) như telnet, FPT, duyệt Web.. đã đề cập ở chương 2. Cụ thể, không giống như các ứng dụng co dãn, các ứng dụng đa phương tiện rất nhạy cảm với trễ đầu cuối và biến động trễ nhưng có thể chịu được việc mất gói dữ liệu thỉnh thoảng xảy ra.

Đặc điểm đầu tiên mà chúng ta có thể phân loại các ứng dụng thời gian thực là khả năng chịu mất dữ liệu, trong đó "mất mát" có thể xảy ra vì một gói đến quá muộn để được phát lại cũng như phát sinh từ những nguyên nhân thông thường trong mạng. Mặt khác, một mẫu âm thanh bị mất có thể được nội suy từ các mẫu xung quanh mà tương đối ít ảnh hưởng đến chất lượng đa phương tiện cảm nhận. Nhưng khi ngày càng nhiều mẫu bị mất thì chất lượng giảm sút đến mức không thể hiểu được bài phát biểu. Hoặc một chương trình điều khiển robot có thể là một ví dụ về một ứng dụng thời gian thực không thể chịu được mất gói. Không thể chấp nhận lệnh hướng dẫn cánh tay robot bị dừng lại. Do đó ta có thể phân loại các ứng dụng thời gian thực là dung nạp hoặc không dung nạp tùy thuộc vào việc chúng có chịu được sự mất mát thường xuyên hay không. (Lưu ý là có nhiều ứng dụng thời gian thực chịu được sự mất mát nhiều hơn các ứng dụng không thời gian thực; ví dụ: so

sánh ứng dụng âm thanh với ứng dụng FTP mà việc mất một bit không được điều chỉnh có thể hiển thị một tệp hoàn toàn vô dụng.)



*Hình 3.1: Phân loại ứng dụng thời gian thực*

Ta có thể chia tiếp ứng dụng thời gian thực theo khả năng thích ứng của chúng. Ví dụ: một ứng dụng âm thanh có thể thích ứng với lượng thời gian trễ mà các gói phải trải qua khi chúng đi qua mạng. Nếu ta nhận thấy các gói hầu như luôn đến trong vòng 300ms kể từ khi được gửi đi thì ta đặt điểm phát lại của mình cho phù hợp, lưu vào bộ đệm bất kỳ gói nào đến trong thời gian dưới 300ms. Giả sử rằng sau đó ta lại thấy tất cả các gói sẽ đến trong vòng 100 mili giây kể từ khi được gửi đi. Nếu ta chuyển lên điểm phát lại của chúng là 100ms, khi đó người dùng ứng dụng có lẽ sẽ nhận thấy một sự cải thiện. Quá trình chuyển điểm phát lại sẽ yêu cầu phát các mẫu với tốc độ tăng lên trong một khoảng thời gian. Với ứng dụng thoại, điều này có thể được thực hiện theo cách hầu như không nhận ra, chỉ đơn giản bằng cách rút ngắn khoảng thời gian im lặng giữa các từ. Do đó, điều chỉnh điểm phát lại khá dễ dàng và hiệu quả cho một số ứng dụng thoại như thoại hội nghị. Việc làm sai lệch tín hiệu phát lại trong khoảng thời gian điều chỉnh và ảnh hưởng của sự biến dạng này sẽ phụ thuộc rất nhiều vào cách người dùng sử dụng dữ liệu. Nếu ta đặt điểm phát lại của mình dựa trên giả định rằng tất cả các gói sẽ đến trong vòng 100 mili giây và sau đó nhận thấy rằng một số gói đến hơi muộn thì những gói đó sẽ bị mất. Trong khi gói sẽ không bị mất nếu điểm phát lại ở 300 mili giây. Vì vậy, chúng ta chỉ nên nâng cao điểm phát lại khi nó mang lại lợi thế có thể nhận thấy được và chỉ khi chúng ta có một số bằng chứng cho thấy số lượng các gói trễ sẽ có kích thước nhỏ có thể chấp nhận được. Các ứng dụng có thể điều chỉnh điểm phát lại là được gọi là ứng dụng tương thích trễ (*delay adaptive*). Một lớp ứng dụng thích ứng khác là tương thích tốc độ (*rate adaptive*). Ví dụ: nhiều thuật toán mã hóa video có thể đánh đổi tốc độ bit với chất lượng. Do đó, nếu mạng hiếm băng thông, thì có thể đặt các thông số mã hóa cho phù hợp. Nếu sau này có thêm băng thông, thì có thể thay đổi các thông số để tăng chất lượng.

Qua phân tích trên ta thấy các nhà thiết kế của một giao thức truyền tải đối với thời gian thực và các ứng dụng đa phương tiện phải đổi mới với một thách thức thực sự trong việc xác định các yêu cầu đủ rộng để đáp ứng các nhu cầu rất khác nhau của các ứng dụng. Họ cũng phải chú ý đến sự tương tác giữa các ứng dụng khác nhau, chẳng hạn như đồng bộ hóa âm thanh và dòng video. Chúng ta sẽ xem bên dưới những lo ngại này đã ảnh hưởng như thế nào đến thiết kế của giao thức truyền tải thời gian thực chính được sử dụng ngày nay là giao thức RTP.

#### *Yêu cầu thiết kế:*

Các ứng dụng nên sử dụng cùng một phương pháp mã hóa và nén; nếu không, dữ liệu do một bên gửi sẽ không thể hiểu được đối với bên nhận. Vì có khá nhiều chương trình mã hóa khác nhau cho âm thanh cũng như hình ảnh, mỗi phương án đều có sự đánh đổi riêng về chất lượng, yêu cầu băng thông và chi phí tính toán. Sẽ là một ý tưởng tồi khi ra quyết định chỉ sử dụng một chương trình mã hóa. Thay vào đó, giao thức phải cung cấp một cách mà người gửi có thể nói với người nhận rằng chương trình mã hóa nó muốn sử dụng và có thể thương lượng cho đến khi một chương trình có sẵn cho cả hai bên được xác định.

Một yêu cầu quan trọng khác là cho phép bên nhận luồng dữ liệu xác định mối quan hệ thời gian giữa các dữ liệu nhận được. Các ứng dụng thời gian thực phải lưu đệm dữ liệu nhận được vào một bộ đệm phát để khắc phục tình trạng chập chờn khi truyền dữ liệu qua mạng. Do đó, cần có nhãn thời gian của dữ liệu kích hoạt phía nhận phát dữ liệu vào thời điểm thích hợp.Thêm nữa liên quan đến thời gian của một luồng đa phương tiện là vấn đề đồng bộ hóa nhiều phương tiện trong một hội nghị. Ví dụ như đồng bộ hóa luồng âm thanh và video có nguồn gốc từ cùng một người gửi. Vấn đề này phức tạp hơn việc xác định thời gian phát lại cho một luồng đơn lẻ.

Một chức năng khác được cung cấp là một chỉ báo về việc mất gói. Lưu ý rằng một ứng dụng có giới hạn chật chẽ về độ trễ thường không thể sử dụng kiểu truyền tải đáng tin cậy như TCP vì việc sửa lỗi bằng cách truyền lại dữ liệu mất mát khiến gói tin đến đích quá muộn. Do đó, ứng dụng phải có khả năng xử lý các gói bị thiếu và bước đầu tiên trong việc đối phó với chúng là phải nhận thấy sự mất mát. Ví dụ, một ứng dụng video sử dụng mã hóa MPEG có thể thực hiện các hành động khác nhau khi một gói bị mất, tùy thuộc gói đó đến từ khung I, khung B hoặc khung P. Mất gói cũng là một chỉ báo tiềm ẩn của tắc nghẽn. Vì ứng dụng đa phương tiện không chạy qua TCP, chúng cũng bỏ lỡ tính năng tránh tắc nghẽn của TCP. Tuy nhiên, các ứng dụng đa phương tiện có khả năng đáp ứng với tình trạng tắc nghẽn như thay đổi các tham số của thuật toán mã hóa để giảm băng thông tiêu

thu. Để thực hiện công việc này, người nhận cần thông báo cho người gửi biết rằng tồn thât đang xảy ra để người gửi có thể điều chỉnh các tham số mã hóa của nó.

Một chức năng phổ biến khác trên các ứng dụng đa phương tiện là khái niệm về chỉ thị ranh giới khung. Khung ở đây là đặc trưng ứng dụng. Ví dụ: là để thông báo cho một ứng dụng video rằng một tập hợp các gói nhất định tương ứng với một khung duy nhất.

Một chức năng cuối cùng có thể muốn đưa vào giao thức là một số cách xác định người gửi thì thân thiện hơn so với dùng địa chỉ IP. Các ứng dụng hội nghị truyền hình và âm thanh có thể hiển thị các người dùng dưới dạng chuỗi (`user@domain.com`) do đó giao thức ứng dụng sẽ hỗ trợ chuỗi như vậy với một luồng dữ liệu.

Bên cạnh những chức năng được yêu cầu từ giao thức còn có một số lưu ý bổ sung đó là phải sử dụng hiệu quả băng thông. Nói một cách khác là không mong muốn gói tin có tiêu đề dài. Lý do là các gói âm thanh, một trong những gói phổ biến nhất của dữ liệu đa phương tiện thì có xu hướng nhỏ. Gói âm thanh dài có nghĩa là độ trễ cao và ảnh hưởng tiêu cực đến chất lượng cuộc trò chuyện. Vì bản thân các gói dữ liệu ngắn mà có tiêu đề lớn có nghĩa là một lượng lớn băng thông là sử dụng cho tiêu đề, do đó làm giảm dung lượng khả dụng cho dữ liệu “hữu ích”. Chúng ta sẽ thấy một số khía cạnh của thiết kế RTP đã bị ảnh hưởng bởi sự cần thiết của việc giữ tiêu đề ngắn gọn.

Vậy mọi tính năng vừa được mô tả có thực sự cần phải có trong một giao thức truyền tải thời gian thực hoặc có thể có thêm một số tính năng khác nữa? Ý tưởng chính ở đây là để dễ dàng hơn cho các nhà phát triển ứng dụng bằng cách cung cấp một tập trùu tượng và các khối xây dựng sẵn hữu ích cho các ứng dụng của họ. Ví dụ, bằng cách đặt một cơ chế đánh dấu thời gian vào RTP, đã giúp các nhà phát triển ứng dụng thời gian thực không phải phát minh ra ứng dụng thời gian thực của riêng mình. Và điều này cũng tăng cơ hội để hai ứng dụng thời gian thực khác nhau có thể tương tác với nhau.

Như vậy chúng ta đã thấy danh sách khá dài các yêu cầu đối với giao thức truyền tải cho đa phương tiện. Tiếp theo ta chuyển sang xem xét chi tiết giao thức đã được chỉ định để đáp ứng các yêu cầu đó.

### **3.2.2 Giao thức truyền tải thời gian thực RTP**

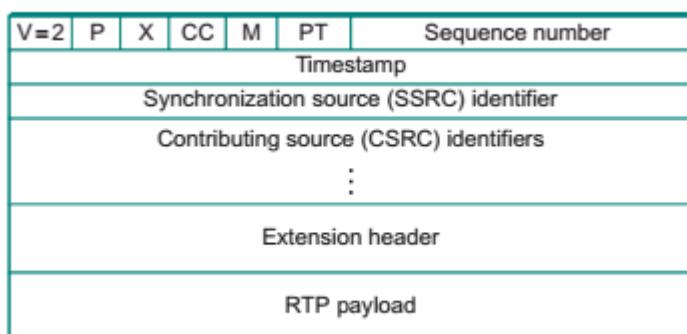
RTP thực sự bắt nguồn từ chức năng giao thức ban đầu được nhúng vào chính ứng dụng. RTP có thể chạy trên nhiều giao thức lớp thấp hơn, nhưng vẫn thường chạy qua UDP. Như vậy ta đang chạy một giao thức truyền tải trên một giao thức truyền tải khác. Nhưng không có quy tắc nào chống lại điều đó. Và trên thực tế UDP cung cấp mức chúc năng tối

thiếu và phân kênh cơ bản dựa trên số cổng. Vì vậy, thay vì tạo lại số cổng trong RTP, RTP chuyển chức năng phân kênh cho UDP.

Giao thức RTP được phát triển trong IETF và đang được sử dụng rộng rãi. Chuẩn RTP xác định một cặp giao thức: giao thức truyền tải thời gian thực (RTP) và giao thức điều khiển truyền tải thời gian thực (RTCP). RTP được sử dụng để trao đổi đa phương tiện dữ liệu, trong khi RTCP được sử dụng để gửi thông tin kiểm soát định kì liên kết với một luồng dữ liệu nhất định. Khi chạy qua UDP, luồng dữ liệu RTP và luồng điều khiển RTCP liên quan sử dụng các cổng tầng vận chuyển nối tiếp nhau. Dữ liệu RTP sử dụng chỉ số cổng chẵn và thông tin điều khiển RTCP sử dụng chỉ số cổng lẻ liền theo.

Vì RTP được thiết kế để hỗ trợ nhiều ứng dụng khác nhau, nó cung cấp một cơ chế linh hoạt mà theo đó các ứng dụng mới có thể được phát triển mà không cần lặp lại việc sửa đổi giao thức RTP. Đối với mỗi lớp ứng dụng (ví dụ: âm thanh), RTP xác định một hồ sơ và một hoặc nhiều định dạng. Hồ sơ cung cấp một loạt thông tin đảm bảo những hiểu biết về các trường trong tiêu đề RTP cho lớp ứng dụng đó. Còn định dạng thì giải thích dữ liệu làm thế nào mà dữ liệu sau tiêu đề RTP đã diễn giải. Ví dụ: tiêu đề RTP được theo sau bởi chuỗi các byte, mỗi byte đại diện cho một mẫu âm thanh được lấy một khoảng thời gian xác định.Thêm nữa, định dạng của dữ liệu có thể phức tạp hơn nhiều. Ví dụ luồng video được mã hóa MPEG sẽ cần phải có nhiều cấu trúc để đại diện cho các loại thông tin khác nhau.

Hình 3.2 chỉ ra khuôn dạng tiêu đề của gói tin RTP.



Hình 3.2: Header của RTP Packet

Kích thước nhỏ nhất của một header của gói tin RTP là 12 bytes. Sau phần header chính, là phần header mở rộng và không cần thiết phải có phần header này. Chi tiết các trường trong một header như sau:

- *Phiên bản (2 bits)*: Cho biết phiên bản của giao thức này. Phiên bản hiện tại là phiên bản 2.
- *P (Padding) (1 bit)*: Cho biết số các byte mở rộng cần thêm vào cuối của gói tin RTP. Ví dụ trong trường hợp ta muốn sử dụng các thuật toán mã hóa, ta có thể thêm vào một số byte vào phần kết thúc của gói tin để tiến hành mã hóa frame trên đường truyền.
- *X (Extension) (1bit)*: Cho biết có thêm phần header mở rộng vào sau phần header chính hay không.
- *CC (CSRC Count) (4 bit)*: Chứa con số định danh CSRC cho biết kích thước cố định của header.
- *M (Marker) (1 bit)*: Cho biết mức của ứng dụng và được định nghĩa bởi một profile. Nếu được thiết lập, có nghĩa là dữ liệu hiện tại đã được tính toán chi phí một cách thích hợp
- *Loại tải trọng - PT (Payload Type) (7 bit)*: Đây là một đặc tả được định nghĩa bởi một profile RTP. Đối với dòng audio, trường loại tải trọng được sử dụng để chỉ loại mã hóa audio (ví dụ PCM, điều chế delta thích nghi, mã hóa tuyến tính) được sử dụng. Nếu bên gửi quyết định thay đổi mã hóa trong phiên, bên gửi có thể thông báo bên nhận về thay đổi này thông qua trường loại tải trọng. Bên gửi có thể muốn thay đổi mã hóa để tăng chất lượng audio hay giảm tốc độ bit dòng RTP. Đối với dòng video loại tải trọng được sử dụng để chỉ loại mã hóa video (ví dụ JPEG hình ảnh động, MPEG 1, MPEG 2, H. 261). Cũng như audio, bên gửi có thể thay đổi mã hóa video trong phiên đang diễn ra.
- *Trường số thứ tự (Sequence Number -16 bits)*: Trường số thứ tự có độ dài 16 bit. Số thứ tự tăng thêm một đơn vị mỗi lần có gói tin RTP được gửi đi, và bên nhận có thể sử dụng để phát hiện mất gói và lưu lại số thứ tự gói. Ví dụ, nếu bên nhận ứng dụng nhận được dòng gói tin RTP với khoảng trống số thứ tự giữa 86 và 89 thì bên nhận sẽ biết ngay gói tin số 87 và 88 bị mất. Sau đó bên nhận có thể cố gắng phục hồi dữ liệu mất này.
- *Trường nhãn thời gian (Timestamp - 32 bits)*: Trường nhãn thời gian có độ dài 32 bit. Nó phản ánh thời điểm lấy mẫu của byte đầu tiên trong gói tin dữ liệu RTP. Như chúng ta đã thấy trong phần trên, bên nhận có thể sử dụng nhãn thời gian để loại bỏ rung pha do mạng gây ra và cung cấp phát lại đồng bộ tại bên nhận. Nhãn thời gian được lấy từ đồng hồ lấy mẫu tại bên gửi. Ví dụ, đối với audio đồng hồ nhãn thời gian tăng thêm một với mỗi chu kì lấy mẫu (ví dụ mỗi khoảng 125 µs cho đồng hồ lấy mẫu 8 kHz): nếu ứng dụng audio tạo các khung từ

160 mẫu được mã hóa thì nhãm thời gian tăng thêm 160 đồi với mỗi gói tin RTP khi nguồn hoạt động. Đồng hồ nhãm thời gian tiếp tục tăng với tốc độ không đổi thậm chí nếu nguồn không hoạt động nữa.

- *Định danh nguồn đồng bộ hóa (SSRC - 32 bits)*: Trường SSRC có độ dài 32 bit. Nó định danh nguồn của dòng RTP. Thông thường mỗi dòng trong phiên RTP có SSRC phân biệt. SSRC không phải địa chỉ IP của bên gửi, mà là một số được nguồn gán ngẫu nhiên khi một dòng mới được bắt đầu. Xác suất hai dòng được gán cùng một SSRC là vô cùng nhỏ. Nếu điều này xảy ra, hai nguồn chọn một giá trị SSRC mới. Nó cũng cho phép một nút duy nhất có nhiều nguồn (ví dụ: nhiều camera) thì phân biệt các nguồn đó. Khi một nút tạo các luồng phương tiện khác nhau (ví dụ: âm thanh và video), nó không bắt buộc để sử dụng cùng một SSRC trong mỗi luồng, vì có các cơ chế trong RTCP (được mô tả bên dưới) để cho phép đồng bộ hóa giữa các phương tiện.
- *Định danh nguồn đóng góp (CSRC)* chỉ được sử dụng khi một số RTP các dòng đi qua một bộ trộn. Bộ trộn có thể được sử dụng để giảm yêu cầu băng thông cho dịch vụ hội nghị bằng cách nhận dữ liệu từ nhiều nguồn và gửi nó dưới dạng một luồng duy nhất. Ví dụ: các luồng âm thanh từ một số thiết bị đồng thời có thể được giải mã và mã hóa thành một luồng âm thanh. Trong trường hợp này, bộ trộn tự liệt kê là đồng bộ hóa nguồn và liệt kê các nguồn đóng góp là các giá trị của SSRC.

### 3.2.3 Giao thức điều khiển truyền tải thời gian thực (RTCP)

Giao thức điều khiển truyền tải thời gian thực (RTCP) có nhiệm vụ giám sát và đánh giá quá trình truyền tin dựa trên việc truyền một cách định kỳ các gói tin điều khiển tới các thành viên tham gia hội thoại với cùng cơ chế truyền dữ liệu. RTCP cung cấp một luồng điều khiển được liên kết với một luồng dữ liệu RTP cho một ứng dụng đa phương tiện. Luồng kiểm soát này cung cấp ba chức năng:

1. Phản hồi về hiệu suất của ứng dụng và mạng
2. Tương quan và đồng bộ hóa các luồng phương tiện khác nhau đến từ cùng một người gửi
3. Truyền đạt danh tính của người gửi để hiển thị trên người dùng giao diện

Chức năng đầu tiên có thể hữu ích để phát hiện và ứng phó với tắc nghẽn. Một số ứng dụng có thể hoạt động ở tốc độ và hiệu suất dữ liệu khác nhau từ đó quyết định sử dụng tính năng

nén để giảm tắc nghẽn. Ví dụ để gửi một luồng dữ liệu với chất lượng cao hơn và ít tắc nghẽn. Phản hồi về hiệu suất cũng hữu ích trong việc chẩn đoán các sự cố mạng.

Ta nghĩ rằng chức năng thứ hai đã được cung cấp bởi mã nguồn đồng bộ hóa (SSRC) của RTP, nhưng thực tế không phải vậy. Nhiều camera từ một nút có thể có các giá trị SSRC khác nhau. Hơn nữa, không có yêu cầu rằng âm thanh và luồng video từ cùng một nút sử dụng cùng một SSRC. Vì có thể xảy ra tranh chấp nên cần thay đổi giá trị SSRC của luồng. Để giải quyết vấn đề này, RTCP sử dụng khái niệm tên (CNAME) được chỉ định cho người gửi, sau đó được liên kết với các giá trị SSRC khác nhau có thể được người gửi đó sử dụng bằng cơ chế RTCP

Tương quan giữa hai luồng chỉ là một phần của vấn đề đồng bộ hóa giữa các phương tiện. Bởi vì các luồng khác nhau hoàn toàn có thể có đồng hồ khác nhau do vậy cần phải có một cách để đồng bộ hóa chính xác các luồng với nhau. RTCP giải quyết vấn đề này bằng cách chuyển tải thông tin thời gian tương quan với thời gian thực tế cùng với nhãn thời gian theo đồng hồ được mang trong gói dữ liệu RTP.

RTCP định nghĩa một số loại gói tin khác nhau, bao gồm

- + Bên gửi báo cáo: là các thông kê về việc gửi và nhận của một phiên từ đối tượng đóng vai trò chủ động gửi
- + Bên nhận báo cáo: là các dữ liệu thống kê việc nhận của bên nhận thông tin.
- + Mô tả nguồn, mang CNAME và thông tin mô tả người gửi khác
- + Các gói điều khiển dành riêng cho ứng dụng

Mỗi gói tin RTCP bắt đầu với một phần cố định giống như thành phần của gói tin dữ liệu RTP, tiếp theo là các thành phần cấu trúc có độ dài biến thiên theo kiểu gói tin nhưng trong giới hạn là 32-bit. Các yêu cầu liên kết và trường độ dài nằm trong phần cố định của mỗi gói tin giúp cho các gói tin RTCP có khả năng sắp xếp theo ngăn xếp (stackable). Nhiều gói RTCP có thể được nối để tạo thành một gói tin hợp nhất và gói tin hợp nhất này được gửi như một gói tin riêng lẻ của giao thức tầng thấp hơn như giao thức UDP. Mỗi gói tin RTCP riêng lẻ trong gói tin hợp nhất có thể được xử lý độc lập mà không cần các yêu cầu về thứ tự cũng như sự kết hợp giữa các gói tin.

Đối với mỗi dòng RTP mà bên nhận nhận như một phần của phiên, bên nhận tạo ra báo cáo quá trình nhận. Bên nhận thu thập các báo cáo quá trình nhận của nó vào một gói tin RTCP duy nhất. Gói tin này được gửi đến cây multicast kết nối tất cả các bên tham gia của phiên. Báo cáo quá trình nhận bao gồm một số trường, các trường quan trọng nhất được liệt kê dưới đây:

- + SSRC của dòng RTP mà báo cáo quá trình nhận của nó được tạo lập.
- + Phần gói tin bị mất trong dòng RTP. Mỗi bên nhận tính số gói tin RTP bị mất chia cho số gói tin được gửi trên một phần của dòng. Nếu bên gửi nhận được các báo cáo quá trình nhận chỉ ra rằng bên nhận chỉ nhận được phần nhỏ các gói tin đã gửi đi thì nó có thể chuyển sang tốc độ mã hóa thấp hơn, nhằm làm giảm nghẽn mạng và cải thiện tốc độ nhận.
- + Số thứ tự cuối cùng nhận được trong dòng gói tin RTP.
- + Rung pha giữa các khoảng đến, là ước lượng gần đúng của phương sai thời gian đến giữa các gói tin liên tiếp trong dòng RTP.

Đối với mỗi dòng RTP mà bên gửi đang truyền đi, bên gửi tạo và truyền các gói tin báo cáo RTCP của bên gửi. Các gói tin này bao gồm thông tin về dòng RTP:

- + SSRC của dòng RTP.
- + Nhãn thời gian và thời gian thực (đồng hồ) của gói tin mới nhất được tạo trong dòng RTP.
- + Số gói tin được truyền đi trên dòng.
- + Số byte được gửi đi trên dòng.

Các báo cáo của bên gửi có thể được sử dụng để đồng bộ các dòng phương tiện trong cùng một phiên RTP. Ví dụ, xem xét ứng dụng hội nghị video trong đó mỗi bên gửi ra hai dòng RTP độc lập, một dòng video và một dòng audio. Nhãn thời gian trong các gói tin RTP này liên kết chặt chẽ với đồng hồ lấy mẫu video và audio, và không liên kết với thời gian thực (đồng hồ thông thường). Đối với gói tin mới nhất được tạo ra trong dòng RTP liên quan, mỗi báo cáo bên gửi RTCP chứa nhãn thời gian của gói tin RTP và thời gian thực khi gói tin được tạo ra. Do đó các gói tin báo cáo bên gửi RTCP liên kết đồng hồ lấy mẫu với đồng hồ thời gian thực. Bên nhận có thể sử dụng liên kết này trong các báo cáo bên nhận RTCP để đồng bộ phát lại audio và video.

Đối với mỗi dòng RTP bên gửi đang truyền đi, bên gửi cũng tạo và truyền các gói tin mô tả nguồn. Các gói tin này chứa thông tin về nguồn, như địa chỉ e-mail của bên gửi, tên của bên gửi, và ứng dụng tạo ra dòng RTP. Nó cũng bao gồm SSRC của dòng RTP liên quan. Các gói tin này cung cấp ánh xạ giữa định danh nguồn (tức là SSRC) và tên người sử dụng/trạm chủ.

Các gói tin RTCP là có khả năng xếp chồng; nghĩa là các báo cáo quá trình nhận của bên nhận, các báo cáo bên gửi, và mô tả nguồn có thể ghép với nhau trong cùng một gói tin. Sau đó gói tin tổng hợp này được đóng gói vào đoạn UDP và chuyển vào cây multicast.

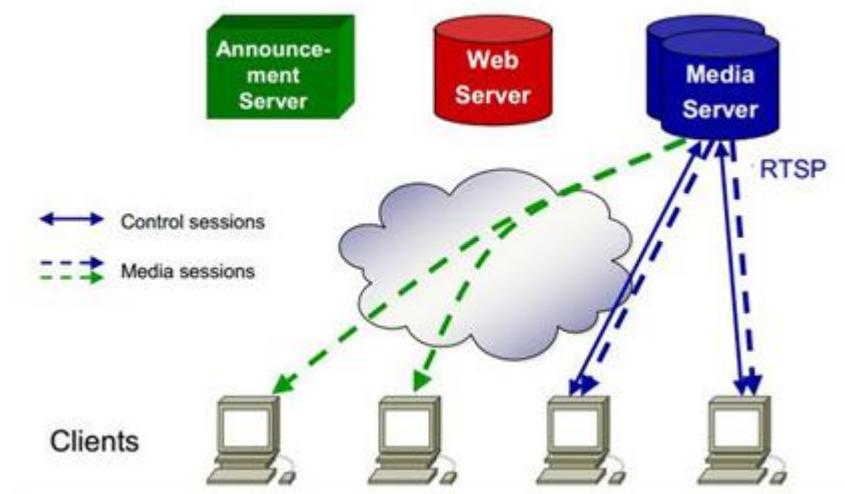
RTP và RTCP là một cặp giao thức khá phức tạp. Sự phức tạp phần lớn đến từ mong muốn làm cho công việc của các nhà thiết kế ứng dụng được dễ dàng hơn. Bởi vì có vô số ứng dụng nên thách thức trong việc thiết kế một giao thức truyền tải là làm cho nó chung nhất để đáp ứng các nhu cầu đa dạng của nhiều ứng dụng khác nhau. RTP đã chứng minh rất thành công trong lĩnh vực này, tạo cơ sở cho phần lớn truyền thông đa phương tiện thời gian thực qua Internet ngày nay.

### 3.3 Giao thức báo hiệu và điều khiển phiên đa phương tiện

#### 3.3.1 Giao thức phát tin trực tuyến thời gian thực RTSP

Giao thức RTSP đã có từ khá lâu được gọi là giao thức phát tin trực tuyến thời gian thực. Giao thức này hỗ trợ người sử dụng đa phương tiện Internet muốn điều khiển phát lại phương tiện liên tục bằng cách tạm dừng phát lại, tìm lại vị trí phát lại về phía trước hay phía sau từ điểm hiện thời, tua trước có hiển thị, tua lại có hiển thị ... Tính năng này tương tự như những gì người sử dụng có với đầu phát DVD video hay đầu phát CD khi nghe nhạc CD. RTSP kiểm soát các máy chủ phát trực tiếp sử dụng trong các hệ thống giải trí và truyền thông, đưa ra các lệnh “phát”, “tạm dừng” và “tua lại”, kiểm soát kết nối máy chủ đến máy khách và các luồng video sẽ được truyền theo yêu cầu người dùng.

Năm 1998 RTSP được tiêu chuẩn hóa với tên gọi RFC 2326. Ngay lập tức, giao thức này trở nên hữu ích và phổ biến vì nó cho phép người dùng điều khiển trực tiếp các nội dung đa phương tiện trên Internet (âm nhạc hoặc video) mà không phải tải tệp xuống thiết bị như trước đó.



Hình 3.3: Kiến trúc của RTSP

RTSP là giao thức ở tầng ứng dụng của chồng giao thức TCP/IP. Giao thức này độc lập với các giao thức ở tầng thấp hơn, do đó nó có thể được thực hiện trên TCP, UDP hoặc giao thức khác ở tầng giao vận.

Ở đây không có khái niệm về một kết nối RTSP; thay vào đó, một máy chủ duy trì một phiên làm việc được đánh dấu bởi một định danh. Một phiên làm việc RTSP không gắn với kết nối vận chuyển như một kết nối TCP. Trong một phiên làm việc RTSP, một máy khách RTSP có thể mở và đóng nhiều kết nối vận chuyển tin cậy với máy chủ để tạo ra các yêu cầu RTSP. Ngoài ra, nó có thể sử dụng một giao thức vận chuyển không kết nối như UDP. Các dòng dữ liệu được điều khiển bởi RTSP có thể sử dụng RTP, nhưng các hoạt động của RTSP không phụ thuộc vào cơ chế truyền tải được sử dụng để truyền dữ liệu truyền thông liên tục.

RTSP có định hướng tương tự như cú pháp và hoạt động với HTTP 1.1, vì vậy những cơ chế mở rộng với HTTP trong nhiều trường hợp có thể cũng được thêm vào RTSP.

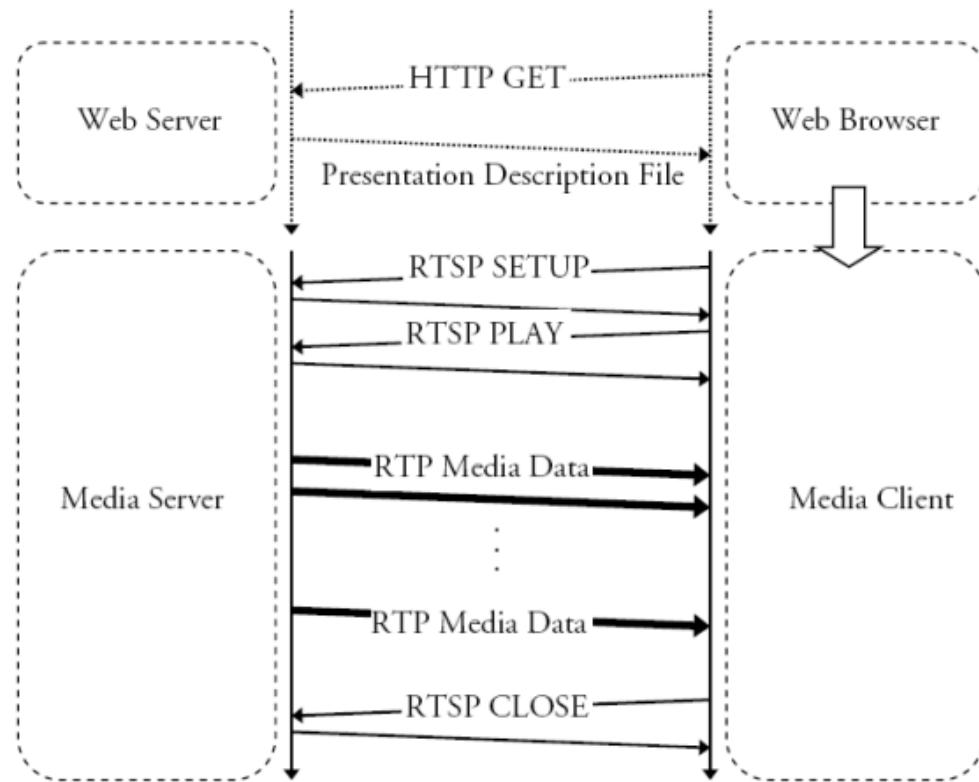
RTSP hỗ trợ các hoạt động sau đây:

- + Lấy lại những dữ liệu truyền thông từ máy chủ truyền thông: Các máy khách có thể yêu cầu một mô tả biểu diễn thông qua HTTP hoặc một số phương pháp khác. Nếu biểu diễn là multicast, mô tả biểu diễn chứa các địa chỉ multicast và các cổng được sử dụng cho dữ liệu truyền thông liên tục. Nếu biểu diễn chỉ được gửi đến máy khách thông qua unicast, máy khách cung cấp đích đến vì lý do an ninh.
- + Mời một máy chủ truyền thông cùng tham gia: Một máy chủ truyền thông có thể được "mời" tham gia vào một kết nối hiện tại, hoặc để phát lại dữ liệu truyền thông vào biểu diễn hay để ghi lại tất cả hoặc một tập con của các dữ liệu truyền thông trong một bài biểu diễn. Chế độ này rất hữu ích cho các ứng dụng giảng dạy phân tán. Những đối tượng khác trong kết nối có thể thay phiên nhau "ấn nút điều khiển từ xa".
- + Bổ sung dữ liệu truyền thông cho một biểu diễn hiện có: Riêng đối với các biểu diễn trực tiếp, nó rất hữu ích nếu máy chủ có thể nói với máy khách về dữ liệu truyền thông bổ sung đã sẵn sàng.

Những yêu cầu RTSP có thể được xử lý bởi các thành phần trung gian, đường hầm và bộ nhớ tạm thời như trong HTTP 1.1.

Chúng ta sẽ xem xét một ví dụ streaming thông thường dùng RTSP đơn giản, được minh họa trên hình 3.4. Trình duyệt trước tiên yêu cầu tệp mô tả trình diễn từ máy chủ Web.

Tệp mô tả trình diễn có thể có các tham chiếu đến một loạt tệp phương tiện liên tục cũng như lệnh để đóng bộ hóa các tệp phương tiện liên tục. Mỗi tham chiếu tới tệp phương tiện liên tục bắt đầu bằng phương thức URL, rtsp://. Tệp mô tả trình diễn này mô tả một hoặc nhiều sự trình diễn, mỗi trình diễn bao gồm một hoặc nhiều dòng dữ liệu đa phương tiện được đóng bộ với nhau. File mô tả trình diễn cũng chứa các thuộc tính của các dòng dữ liệu như định dạng nén để client lựa chọn và chuẩn bị play media.



Hình 3.4: Thủ tục RTSP tương tác giữa máy khách và máy chủ

Máy chủ đóng gói tệp mô tả trình diễn trong bản tin đáp ứng HTTP và gửi bản tin tới trình duyệt. Khi trình duyệt nhận được bản tin đáp ứng HTTP, trình duyệt gọi bộ phát phương tiện (tức là ứng dụng hỗ trợ) dựa trên trường loại nội dung của bản tin. Tệp mô tả trình diễn bao gồm các tham chiếu đến dòng phương tiện, sử dụng phương thức URL rtsp:// như trong ví dụ trên. Sau khi hoàn tất yêu cầu SETUP, cấu hình được các luồng dữ liệu để chuẩn bị streaming, máy client sẽ gửi yêu cầu PLAY để thực hiện truyền các frame dữ liệu thật sự từ máy server tới máy client, và các frame dữ liệu này sẽ được lưu trong một bộ đệm của máy client, các frame này sẽ được giải mã (decode), rồi được hiển thị bởi trình chơi file video và âm thanh (VLC).

RTSP có một số đặc điểm cơ bản sau:

- + **Khả năng mở rộng:** Các phương pháp và các thông số mới có thể dễ dàng thêm vào RTSP.
- + **Dễ dàng phân tích:** RTSP có thể được phân tích theo bộ phân tích cú pháp HTTP hoặc MIME.
- + **An toàn:** RTSP tái sử dụng những cơ chế bảo mật web. Tất cả các cơ chế chứng thực HTTP như chứng thực cơ bản và chứng thực băm được trực tiếp áp dụng. Ngoài ra RTSP cũng có thể tái sử dụng những cơ chế an ninh tầng giao vận hoặc tầng mạng.
- + **Độc lập với tầng vận chuyển:** RTSP có thể sử dụng hoặc một giao thức truyền tin không tin cậy như UDP, hoặc một giao thức truyền tin tin cậy như TCP.
- + **Khả năng đa máy chủ:** Mỗi luồng dữ liệu truyền thông trong một biểu diễn có thể nằm trên một máy chủ khác. Máy khách sẽ tự động thiết lập một số phiên kiểm soát đồng thời với các máy chủ truyền thông khác. Đồng bộ dữ liệu truyền thông được thực hiện ở mức độ vận chuyển.
- + **Kiểm soát các thiết bị ghi:** Các giao thức có thể kiểm soát cả các thiết bị ghi và phát, cũng như thiết bị có thể thay đổi giữa hai chế độ.
- + **Tách kiểm soát dòng và khởi tạo kết nối:** Kiểm soát dòng đã được tách ra khỏi việc mời một máy chủ truyền thông tham gia một kết nối. Yêu cầu duy nhất là giao thức khởi tạo kết nối có thể cung cấp hoặc được sử dụng để tạo ra một nhận dạng kết nối duy nhất. Trong một số trường hợp đặc biệt, tiêu chuẩn SIP (Session Initiation Protocol – Giao thức khởi tạo phiên) hoặc tiêu chuẩn H.323 có thể được sử dụng để mời một máy chủ tham gia một kết nối.
- + **Thích hợp cho các ứng dụng chuyên nghiệp:** RTSP hỗ trợ độ chính xác khung hình thông qua tem thời gian, cho phép chỉnh sửa ảnh kỹ thuật số từ xa.
- + **Mô tả biểu diễn trung tính:** Giao thức không áp đặt một biểu diễn đặc biệt mô tả hoặc định dạng tập tin đặc tả (metafile) và có thể truyền tải loại định dạng được sử dụng. Tuy nhiên, các mô tả trình bày phải có ít nhất một URI RTSP.
- + **Thành phần trung gian và tường lửa thân thiện:** Các giao thức nên sẵn sàng xử lý bởi cả tường lửa ứng dụng và tầng giao vận.
- + **Thân thiện với HTTP:** Trường hợp cần thiết, RTSP tái sử dụng khái niệm HTTP, do đó cơ sở hạ tầng hiện tại có thể được tái sử dụng. Cơ sở hạ tầng này bao gồm PICS (Platform for Internet Content Selection – Nền tảng để lựa chọn nội dung Internet) cho gắn nhãn với nội dung. Tuy nhiên, RTSP không

chỉ thêm các phương pháp vào HTTP bởi vì việc kiểm soát dữ liệu truyền thông liên tục đòi hỏi trạng thái máy chủ trong hầu hết các trường hợp.

- + Kiểm soát máy chủ thích hợp: Nếu một máy khách có thể bắt đầu một luồng dữ liệu, nó phải có khả năng ngăn chặn một luồng dữ liệu. Máy chủ không nên bắt đầu truyền dữ liệu cho các máy khách trong trường hợp máy khách không thể ngăn chặn luồng dữ liệu.
- + Đàm phán vận chuyển: Máy khách có thể đàm phán phương pháp vận chuyển trước khi thực sự cần phải xử lý một luồng dữ liệu truyền thông liên tục.
- + Khả năng đàm phán: Nếu tắt các tính năng cơ bản, phải có một cơ chế an toàn cho máy khách để xác định phương pháp không tiếp tục thực hiện. Điều này cho phép các máy khách trình bày giao diện người dùng phù hợp. Ví dụ, nếu tìm kiếm không cho phép, giao diện người sử dụng phải có khả năng không cho phép di chuyển một thanh trượt chỉ báo vị trí.

### 3.2.2 Giao thức mô tả phiên SDP

SDP, từ viết tắt của Session Description Protocol (Giao thức Mô tả Phiên), là một định dạng để mô tả các thông số khởi tạo dòng thông tin phương tiện (streaming media). SDP được ban hành bởi IETF trong RFC 4566. Dòng thông tin phương tiện là những nội dung được xem hoặc nghe trong khi truyền. SDP là một giao thức khá chung chung có thể được sử dụng trong nhiều tình huống và thường được sử dụng cùng với một hoặc nhiều giao thức khác (ví dụ: SIP). Nó truyền đạt những thông tin sau:

- + Tên và mục đích của phiên
- + Thời gian bắt đầu và kết thúc phiên
- + Các loại phương tiện (ví dụ: âm thanh, video) trong phiên
- + Thông tin chi tiết cần thiết để nhận phiên (ví dụ: địa chỉ đa hướng mà dữ liệu sẽ được gửi đến, giao thức truyền tải được sử dụng, số cổng, lược đồ mã hóa)

SDP cung cấp thông tin được định dạng trong ASCII bằng cách sử dụng một chuỗi các dòng văn bản, mỗi dòng có dạng “<type> = <value>.” Sau đây là một ví dụ về SDP:

v=0

o=larry 2890844526 2890842807 IN IP4 128.112.136.10

s=Networking 101

i=A class on computer networking

u=http://www.cs.princeton.edu/

*e=larry@cs.princeton.edu*  
*c=IN IP4 224.2.17.12/127*  
*t=2873397496 2873404696*  
*m=audio 49170 RTP/AVP 0*  
*m=video 51372 RTP/AVP 31*  
*m=application 32416 udp wb*

Lưu ý rằng SDP, giống như HTML, khá dễ đọc đối với con người nhưng có các quy tắc định dạng nghiêm ngặt giúp máy móc có thể diễn giải dữ liệu rõ ràng. Ví dụ: đặc điểm kỹ thuật SDP xác định tất cả các loại thông tin có thể được phép xuất hiện, thứ tự xuất hiện và định nghĩa định dạng và các từ dành riêng.

Điều đầu tiên cần chú ý là mỗi loại thông tin được xác định bởi một ký tự đơn. Ví dụ: dòng v = 0 cho chúng ta biết rằng "phiên bản" có giá trị bằng không; nghĩa là, thông báo này được định dạng theo phiên bản 0 của SDP. Dòng tiếp theo cung cấp "nguồn gốc" của phiên chứa đủ thông tin để xác định phiên là duy nhất. Larry là một tên người dùng của người tạo ra phiên và 128.112.136.10 là địa chỉ IP của máy tính của anh ta. Số tiếp theo là mã định danh phiên được chọn để là duy nhất cho máy đó. Tiếp theo là số "phiên bản" cho thông báo SDP; nếu thông tin phiên được cập nhật muộn hơn thông báo, số phiên bản sẽ được tăng lên.

Ba dòng tiếp theo (s, i và u) cung cấp tên phiên, mô tả một phiên mô tả và URI của phiên. Đây là những thông tin hữu ích cho người dùng trong việc quyết định có tham gia vào phiên này không. Thông tin như vậy có thể được hiển thị trong giao diện người dùng của một công cụ hiển thị thư mục phiên và các sự kiện sắp tới đã được quảng cáo bằng SDP. Dòng tiếp theo (e =...) chứa địa chỉ email của một người để liên hệ.

Tiếp theo, chúng ta đi đến các chi tiết kỹ thuật để cho phép một chương trình ứng dụng tham gia vào phiên. Dòng bắt đầu c =... cung cấp địa chỉ IP multicast mà dữ liệu của phiên này sẽ được gửi đến; một người dùng sẽ cần tham gia nhóm multicast này để nhận phiên. Tiếp theo là thời gian bắt đầu và kết thúc cho phiên (được mã hóa dưới dạng số nguyên theo Giao thức thời gian mạng NTP). Cuối cùng là thông tin về các phương tiện truyền thông cho phiên này. Phiên này có sẵn ba loại phương tiện: âm thanh, video và một ứng dụng bảng trắng được chia sẻ được gọi là "wb". Mỗi loại phương tiện có một dòng thông tin được định dạng như sau:

*m=<media> <port> <transport> <format>*

Media chỉ rõ loại phương tiện và chỉ số cổng là các cổng UDP. Khi nhìn vào trường "transport", ta thấy ứng dụng wb chạy trực tiếp qua UDP, trong khi âm thanh và video được vận chuyển bằng cách sử dụng "RTP / AVP." Điều này có nghĩa là chúng chạy qua RTP và sử dụng hồ sơ ứng dụng được gọi là AVP. Hồ sơ ứng dụng xác định một số lược đồ mã hóa khác nhau cho âm thanh và video; trong trường hợp này chúng ta có thể thấy rằng âm thanh đang sử dụng mã hóa 0 (là mã hóa sử dụng tốc độ lấy mẫu 8 kHz và 8 bit trên mỗi mẫu) và video đang sử dụng mã hóa 31, đại diện cho cơ chế mã hóa H.26. Những con số cho các lược đồ mã hóa được xác định trong RFC xác định cấu hình AVP. Cuối cùng là mô tả về loại phương tiện "wb". Tất cả các mã hóa thông tin cho dữ liệu này dành riêng cho ứng dụng wb và vì vậy chỉ cần cung cấp tên của ứng dụng trong trường "format" là đủ.

Ví dụ trên đã cho biết cách mô tả các phiên. SDP được sử dụng để thông báo về hội nghị đa phương tiện bằng cách gửi bản tin SDP đến một địa chỉ multicast mặc định. Công cụ trên thiết bị người dùng sẽ hoạt động bằng cách tham gia nhóm multicast đó và hiển thị thông tin mà nhóm thu thập được từ các tin nhắn SDP đã nhận. SDP cũng được sử dụng trong việc phân phối video giải trí của IP (thường được gọi là IPTV) để cung cấp thông tin về nội dung video trên mỗi kênh truyền hình.

SDP cũng đóng một vai trò quan trọng cùng với giao thức khởi tạo phiên (SIP). Với việc phát triển mạnh mẽ của các ứng dụng đa phương tiện SIP hiện là một trong những thành viên quan trọng của bộ giao thức Internet.

### **3.3.3 Giao thức khởi tạo phiên SIP**

SIP do nhóm làm việc MMUSIC (Multiparty Multimedia Session Control) của IETF phát triển và chuẩn hóa trong RFC 3261. Theo định nghĩa của IETF, SIP là "giao thức báo hiệu thực hiện việc khởi tạo, thay đổi và huỷ các phiên kết nối tương tác đa phương tiện điểm - điểm cho đến đa điểm giữa những người sử dụng". SIP có thể sử dụng cho rất nhiều dịch vụ khác nhau trong mạng IP như dịch vụ thông điệp, thoại, hội nghị thoại, e-mail, dạy học từ xa, quảng bá, truy nhập Web và hội nghị video...

SIP dựa trên ý tưởng và cấu trúc của HTTP (Hypertext Transfer Protocol), giao thức trao đổi thông tin của World Wide Web. Nó được định nghĩa như một giao thức Client-Server, trong đó các yêu cầu được bên gọi (client) đưa ra và bên bị gọi (server) trả lời. SIP sử dụng một số kiểu bản tin và các trường mào đầu của HTTP, xác định nội dung luồng thông tin theo mào đầu thực thể (mô tả nội dung - kiểu loại) và cho phép xác nhận các phương pháp sử dụng giống nhau được sử dụng trên Web.

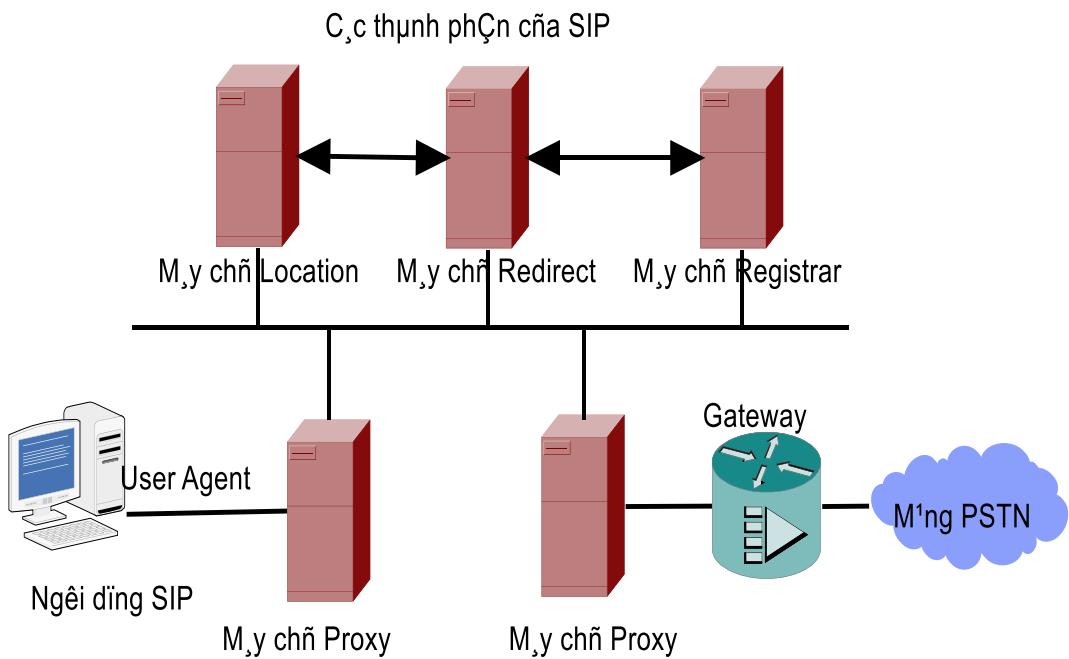
SIP là một giao thức điều khiển nằm trên lớp ứng dụng mà có thể thiết lập, sửa đổi và kết thúc các phiên truyền thông đa phương tiện. SIP có thể mời các thành viên tham gia vào các phiên truyền thông đơn hướng hoặc đa hướng. SIP hỗ trợ việc ánh xạ tên và các dịch vụ chuyển tiếp một cách trong suốt, vì thế cho phép thực hiện các dịch vụ thuê bao điện thoại của mạng thông minh và mạng ISDN. SIP hỗ trợ 5 khía cạnh của việc thiết lập và kết thúc các truyền thông đa phương tiện:

- + Định vị người dùng (User location): xác định hệ thống đầu cuối được sử dụng trong truyền thông.
- + Các khả năng người dùng (User capabilities): xác định phương tiện và các thông số phương tiện được sử dụng.
- + Tính khả dụng người dùng (User Availability): xác định sự sẵn sàng của bên được gọi để tiến hành truyền thông.
- + Thiết lập cuộc gọi (Call setup): thiết lập các thông số của cuộc gọi tại cả hai phía bị gọi và chủ gọi.
- + Xử lý cuộc gọi (Call handling): bao gồm truyền tải và kết thúc cuộc gọi.

### **3.2.3.1 Thành phần chức năng mạng báo hiệu SIP**

Các thực thể giao tiếp với nhau bằng cách sử dụng giao thức SIP được gọi là mạng SIP. Hình 3.5 mô tả khung logic của mạng SIP và các thực thể chức năng của nó.

Các phần tử chức năng của mạng SIP gồm SIP User Agent và máy chủ SIP. SIP xác định ba máy chủ: SIP proxy, máy chủ đăng ký và máy chủ chuyển hướng SIP. Máy chủ định vị và các loại máy chủ ứng dụng, không được hiển thị trong 3.5, không thuộc về SIP. Trong bối cảnh SIP, các phương tiện của phiên sử dụng giao thức RTP / RTCP được kiểm soát bởi SIP; các giao thức bảo mật SIP dùng giao thức bảo mật lớp truyền tải (TLS); và các giao thức truyền tải SIP dùng TCP, UDP và SCTP được truyền qua mạng IP, các máy chủ ứng dụng SIP là máy chủ TC / VTC / VC, máy chủ định vị, máy chủ hội nghị web, máy chủ cầu nối đa phương tiện, máy chủ trò chuyện và máy chủ chia sẻ ứng dụng.



*Hình 3.5: Các thực thể chức năng của mạng SIP*

**(i) Tác nhân người dùng SIP (User Agent):**

Thiết bị đầu cuối hỗ trợ SIP được gọi là tác nhân người dùng SIP (UA). Mục đích của SIP là thiết lập phiên làm việc giữa các tác nhân người dùng. Như cái tên của nó đã chỉ ra, tác nhân người dùng nhận chỉ đạo chỉ đạo từ người dùng và hoạt động như một tác nhân thay mặt họ để thiết lập và hủy bỏ các phiên truyền thông với các tác nhân người dùng khác. Trong hầu hết các trường hợp người dùng sẽ là một con người, nhưng người dùng cũng có thể là một giao thức khác như trong trường hợp gateway. UA phải có khả năng thiết lập phiên truyền thông với UA khác. UA phải duy trì trạng thái mà nó khởi tạo hoặc tham gia.

Tác nhân người dùng SIP đóng cả vai trò tác nhân người dùng khác (UAC) và tác nhân người dùng chủ (UAS). UAC khởi tạo yêu cầu trong khi UAS tạo phản hồi. Trong một phiên, tác nhân người dùng thường sẽ hoạt động như cả UAC và UAS. Tác nhân người dùng SIP cũng phải hỗ trợ giao thức mô tả phiên (SDP) để mô tả phương tiện. UA nên quảng cáo khả năng và tính năng của mình trong bất kỳ yêu cầu nào nó gửi đi. Điều này cho phép các UA khác hiểu biết về nó mà không cần phải thực hiện một truy vấn. Ví dụ các phương thức mà UA hỗ trợ nên được liệt kê trong trường tiêu đề *Allow*. Các mở rộng của SIP phải được liệt kê trong trường tiêu đề *Supported*. UA thường đăng ký với một máy chủ proxy trong miền của họ.

Gateway trong SIP là một ứng dụng giao tiếp giữa mạng SIP với mạng sử dụng giao thức báo hiệu khác. Về mặt giao thức SIP, gateway là một loại tác nhân người dùng đặc biệt, trong đó tác nhân người dùng hoạt động thay mặt cho một giao thức khác chứ không phải là một con người. Một gateway sẽ là kết cuối cho một đường truyền báo hiệu và cũng có thể kết cuối cả đường truyền phương tiện, mặc dù điều này không phải lúc nào cũng đúng. Một sự khác biệt khác giữa tác nhân người dùng và gateway là số lượng người dùng được hỗ trợ. Trong khi tác nhân người dùng thường hỗ trợ một người dùng, một gateway có thể hỗ trợ hàng trăm hoặc hàng nghìn người dùng.

### (ii) **Máy chủ SIP**

Máy chủ SIP sử dụng SIP để quản lý giao tiếp thời gian thực giữa các máy khách SIP. Trên thực tế, các máy chủ SIP là những thực thể chủ chốt cho phép liên lạc giữa các máy khách SIP bằng cách định tuyến các bản tin SIP qua việc phân giải địa chỉ và là cốt lõi của mạng SIP. Máy chủ SIP hoạt động dựa trên các yêu cầu được gửi đi từ các máy khách SIP, xử lý các bản tin SIP và hoạt động trên các quy tắc tuân theo tiêu chuẩn kỹ thuật được xác định trong giao thức điều khiển cuộc gọi SIP. SIP xác định có máy chủ đại diện (proxy), máy chủ đăng ký (registrar) và máy chủ chuyển hướng (redirect).

*Máy chủ SIP proxy* nhận bản tin yêu cầu từ tác nhân người dùng hoặc từ proxy khác và thay mặt cho tác nhân người dùng trong việc chuyển tiếp hoặc phản hồi yêu cầu. Cũng như một bộ định tuyến chuyển tiếp các gói IP, một SIP proxy thực hiện chuyển tiếp các bản tin SIP. Proxy không phải là B2BUA vì nó chỉ được phép sửa đổi yêu cầu và phản hồi theo các quy tắc nghiêm ngặt được nêu trong RFC 3261. Các quy tắc này duy trì tính minh bạch đầu cuối của báo hiệu SIP trong khi vẫn cho phép máy chủ proxy thực hiện các dịch vụ và chức năng có giá trị cho các tác nhân người dùng. Máy chủ proxy thường có quyền truy cập vào cơ sở dữ liệu hoặc dịch vụ vị trí để hỗ trợ cho quá trình xử lý các yêu cầu (xác định bước tiếp theo). Cơ sở dữ liệu gồm thông tin đăng ký SIP, thông tin hiện diện hoặc bất kỳ thông tin về vị trí của người dùng. Để chuyển tiếp một yêu cầu, proxy không cần phải hiểu bản tin đó. Bất kỳ một yêu cầu không xác định nào thì được giả định sử dụng mô hình giao dịch không phải INVITE. Một proxy không được thay đổi thứ tự của các trường tiêu đề hoặc sửa đổi hay xóa các trường tiêu đề.

Một máy chủ proxy có thể là proxy không trạng thái (stateless) hoặc proxy trạng thái (statefull). Một proxy không trạng thái thì không giữ mọi thông tin trạng thái của cuộc gọi hoặc giao dịch. Máy chủ proxy không trạng thái xử lý từng yêu cầu hoặc phản hồi SIP chỉ dựa trên nội dung tin nhắn. Sau khi thư đã được phân tích cú pháp, xử lý và chuyển tiếp hoặc phản hồi, không có thông tin nào (chẳng hạn như thông tin hộp thoại) về tin nhắn được

lưu trữ. Một proxy không trạng thái không bao giờ truyền lại tin nhắn và không sử dụng bất kỳ bộ hẹn giờ SIP nào. Một máy chủ proxy trạng thái theo dõi các yêu cầu và phản hồi nhận được trong quá khứ và sử dụng thông tin đó để xử lý các yêu cầu và phản hồi trong tương lai. Ví dụ: một máy chủ proxy trạng thái hẹn giờ khi một yêu cầu được chuyển tiếp. Nếu không nhận được phản hồi cho yêu cầu trong khoảng thời gian hẹn giờ, proxy sẽ truyền lại yêu cầu, giải phóng tác nhân người dùng của tác vụ này. Ngoài ra, một proxy trạng thái có thể yêu cầu xác thực tác nhân người dùng.

*Máy chủ chuyển hướng* là một loại máy chủ SIP thực hiện việc đáp ứng chứ không chuyển tiếp các yêu cầu. Giống như một máy chủ proxy, một máy chủ chuyển hướng sử dụng cơ sở dữ liệu hoặc dịch vụ vị trí để tra cứu người dùng. Tuy nhiên thông tin vị trí được gửi lại cho phía chủ gọi trong bản tin phản hồi chuyển hướng (3xx) chỉ ra rằng cuộc gọi nên được thử ở một địa điểm khác. Mục đích chính là giải quyết vấn đề thay đổi vị trí tạm thời hoặc vĩnh viễn của người dùng.

*Máy chủ đăng ký SIP* chấp nhận các yêu cầu đăng ký REGISTER. Tất cả những yêu cầu khác sẽ nhận được phản hồi 501 với ý nghĩa là không được triển khai. Thông tin có được từ yêu cầu đăng ký này sẽ được cung cấp cho các máy chủ SIP khác trong cùng một miền quản trị như proxy hay máy chủ chuyển hướng. Một máy chủ đăng ký sẽ lưu tất cả những thông tin để liên lạc nhận được từ bản tin REGISTER vào máy chủ định vị. Giao thức trao đổi giữa máy chủ đăng ký SIP và máy chủ định vị không thuộc phạm vi của SIP. Máy chủ đăng ký thường yêu cầu tác nhân người dùng đăng ký phải được xác thực để các cuộc gọi đến không thể bị người dùng trái phép chiếm đoạt. Điều này có thể xảy ra bởi một người dùng trái phép đăng ký SIP URI của người khác để trả đến UA của chính họ. Các cuộc gọi đến đến URI đó sau đó sẽ đổ chuông sai UA.

*Máy chủ định vị* được mường tượng trong mạng SIP sẽ giữ các thông tin liên lạc và các thông tin khác trong cơ sở dữ liệu. Các máy chủ đăng ký SIP lưu trữ danh bạ và các thông tin khác của người dùng trong máy chủ định vị. Các máy chủ SIP giao tiếp với máy chủ định vị lấy thông tin *địa chỉ* để định tuyến các bản tin SIP cho người dùng hoặc máy chủ khác. Tuy nhiên, giao thức giao tiếp giữa máy chủ định vị và các máy chủ SIP không phải là một phần của SIP.

*Máy chủ ứng dụng SIP* hoạt động như SIP UA có thể gửi SIP yêu cầu và nhận tin nhắn phản hồi SIP. Trên thực tế, các máy chủ ứng dụng SIP nổi lên là một phần quan trọng nhất đối với việc tạo ra và sử dụng các dịch vụ đa phương tiện bằng SIP. Với sự ra đời của SIP, các dịch vụ đa phương tiện giàu tính năng mới được tích hợp với các dịch vụ khác như

dịch vụ web và đã mở ra một biên giới mới cho việc tạo ra các dịch vụ đa phương tiện thời gian thực.

### 3.2.2.2 Địa chỉ SIP

Địa chỉ SIP là mã định danh duy nhất cho mỗi người dùng trên mạng, giống như số điện thoại xác định từng người dùng trên mạng điện thoại toàn cầu hoặc địa chỉ email. Nó còn được gọi là SIP URI (Uniform Resource Identifier).

Địa chỉ SIP là những gì nhận được khi đăng ký một tài khoản SIP và nó hoạt động như một trình xử lý truyền thông mà mọi người sử dụng để liên lạc với nhau. Thông thường, thông qua ENUM, các địa chỉ SIP được dịch sang số điện thoại. Bằng cách này, ta có thể có một tài khoản SIP có địa chỉ SIP được dịch sang số điện thoại; số điện thoại được chấp nhận nhiều hơn đối với những người thường làm việc với số liên lạc so với địa chỉ SIP.

Địa chỉ SIP giống với địa chỉ email. Cấu trúc giống như thế này:

*sip: user @ domain: cổng*

"Sip" biểu thị giao thức và không thay đổi. Nó bắt đầu mọi địa chỉ SIP. Một số địa chỉ SIP được truyền mà không có phần 'sip' vì nó được hiểu rằng phần này sẽ tự động thay thế. Kiểu "sip" là khi sử dụng giao thức truyền tải không đảm bảo an ninh. "sips" là cho trường hợp trao đổi thông tin trên giao thức truyền tải đảm bảo an ninh.

"User" là phần bạn chọn khi đăng ký địa chỉ SIP. Nó có thể là một chuỗi số hoặc chữ cái. Ở các địa chỉ khác, nó có thể là số điện thoại (được sử dụng cho SIP trunking cho các hệ thống PBX) hoặc bất kỳ kết hợp chữ cái và số nào khác.

Dấu @ là bắt buộc ở đây giữa người dùng và miền, như trường hợp có địa chỉ email.

"Tên miền" là tên miền của dịch vụ mà ta đang đăng ký. Nó có thể là một miền hoàn toàn đủ điều kiện hoặc đơn giản là một địa chỉ IP.

"Cổng" là tham số tùy chọn và phần lớn ít được sử dụng trong địa chỉ SIP. Nó biểu thị cổng truy cập trên máy chủ proxy hoặc bất kỳ máy chủ nào khác dành riêng cho hoạt động SIP.

Một số ví dụ địa chỉ SIP

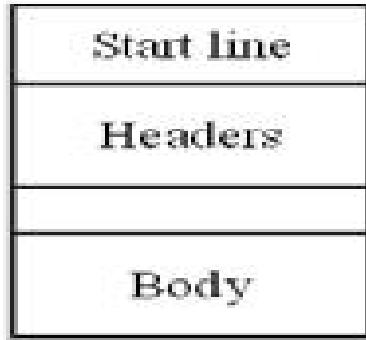
*sip: 500@ekiga.net*

*sip: 8508355@vp.mdbserv.sg*

*sip: 12345@14.18.10.23: 5090*

### 3.2.2.3 Các bản tin SIP

SIP là giao thức dạng văn bản, sử dụng bộ ký tự ISO 10646 trong mã hoá UTF-8 trong RFC 2279. Điều này tạo cho SIP tính linh hoạt, mở rộng và dễ thực thi các ngôn ngữ lập trình cấp cao như Java, Tol, Perl. SIP được thiết kế là một giao thức máy khách-máy chủ theo một cú pháp gần như giống hệt với HTTP / 1.1, mặc dù SIP không phải là HTTP hoặc các phần mở rộng của nó. Theo đó, một bản tin SIP là một yêu cầu từ một máy khách đến một máy chủ để gọi một hoạt động cụ thể hoặc phản hồi từ máy chủ tới máy khách để chỉ ra trạng thái của yêu cầu. Cá hai loại bản tin này đều bao gồm một dòng bắt đầu, một hoặc nhiều trường tiêu đề, một dòng trống để chỉ thị đã hết phần tiêu đề và phần thân bản tin tùy chọn.



Hình 3.6: Cấu trúc bản tin SIP

*Dòng bắt đầu (Start line):* Mỗi bản tin SIP được bắt đầu với một Start Line, Start Line vận chuyển loại bản tin (phương thức trong các Request, và mã đáp ứng trong các bản tin đáp ứng), URI yêu cầu và phiên bản của giao thức. Start line có thể là Request-Line (trong các yêu cầu) hoặc là Status-Line (trong các đáp ứng).

*Tiêu đề (Headers):* Các trường Header của SIP được sử dụng để vận chuyển các thuộc tính của bản tin và để thay đổi ý nghĩa của bản tin. Chúng tương tự như các trường tiêu đề của bản tin HTTP theo cả cú pháp và ngữ nghĩa. Tiêu đề bản tin bao gồm bốn loại: tiêu đề chung, tiêu đề yêu cầu, tiêu đề đáp ứng và tiêu đề thực thể.

*Thân bản tin (Body):* Thân bản tin được sử dụng để mô tả phiên được khởi tạo (ví dụ: trong một phiên multimedia phần này sẽ mang loại mã hóa audio và video, tốc độ lấy mẫu ...), hoặc nó có thể được sử dụng để mang dữ liệu dưới dạng text hoặc nhị phân (không được dịch) mà liên quan đến phiên đó. Phần thân bản tin có thể xuất hiện trong cả bản tin yêu cầu và đáp ứng. Các loại Body bao gồm: giao thức mô tả phiên SDP, mở rộng thư điện

tử internet đa mục đích MIME (Multipurpose Internet Mail Extentions) và các phần định nghĩa trong IETF.

(i) *Bản tin yêu cầu:*

Yêu cầu SIP được gửi từ máy khách đến máy chủ để yêu cầu một hoạt động. Các yêu cầu SIP được thể hiện bằng dòng yêu cầu trên Start line của bản tin. Dòng yêu cầu chứa tên phương thức, URI yêu cầu và phiên bản của giao thức (hình 3.7). RFC 3261 định nghĩa 6 kiểu bản tin yêu cầu cho phép UA và proxy có thể xác định người dùng, khởi tạo, sửa đổi, hủy một phiên, ngoài ra trong các phiên bản RFC khác còn định nghĩa thêm một số bản tin mở rộng của SIP.

**INVITE:** Được UAC (user agent client) sử dụng để yêu cầu thiết lập một cuộc gọi với UAS (user agent server). Khi một UA (user agent) muốn thiết lập một cuộc gọi với một UA khác, nó sẽ gửi bản tin INVITE tới địa chỉ của UA kia, trong bản tin INVITE này sẽ chứa các thông tin như địa chỉ SIP (SIP URI) của phía gọi, hoặc các thông tin mô tả phiên nằm trong phần message body các thông tin này thường được mô tả bằng giao thức SDP(Session Description Protocol), các thông tin mô tả phiên thường là các thông tin như các chuẩn mã hóa âm thanh, hình ảnh (codecs), các thông tin cần thiết để mở các kênh logic.

**REGISTER:** Bản tin Register được UA sử dụng để thực hiện thủ tục đăng ký với SIP server, trong bản tin Register này sẽ chứa các thông tin về UA như: địa chỉ hiện tại, số port mà UA đang lắng nghe.... Nếu nhận được bản tin Response 200 OK từ phía SIP server nghĩa là quá trình đăng ký đã thành công.

**BYE:** Bản tin BYE được gửi từ phía user agent để kết thúc một cuộc gọi. Khi một user agent muốn kết thúc cuộc gọi, nó sẽ gửi bản tin BYE tới phía còn lại để thông báo rằng nó muốn kết thúc cuộc gọi. Một bản tin Response 200 OK được gửi trả lại để thông báo rằng yêu cầu kết thúc được chấp nhận và cuộc gọi được kết thúc.

**ACK:** Bản tin ACK được gửi từ phía user agent để cho biết rằng nó đã nhận được bản tin response cuối cùng cho bản tin INVITE trước đó. Các bản tin Response cuối cùng là các bản tin thuộc lớp 2xx, 3xx, 4xx, 5xx, 6xx.

**CANCEL:** UA1 muốn thực hiện cuộc gọi tới UA2, ban đầu nó sẽ gửi bản tin INVITE tới user agent2. Nếu user agent1 không nhận được bản tin response cuối cùng từ phía user agent2 trong khoảng thời gian time out, nó sẽ gửi bản tin CANCEL tới user agent2 để hủy bỏ cuộc gọi. Bản tin BYE sẽ không được sử dụng trong trường hợp này vì cuộc gọi vẫn

chưa được thiết lập. Một cuộc gọi được thiết lập khi các bản tin INVITE, 200 OK, ACK được trao đổi.

**OPTIONS:** Được sử dụng để truy vấn về khả năng của một UA. Khả năng ở đây có thể là khả năng mã hóa và giải mã âm thanh, hình ảnh, các message header được user agent hỗ trợ.

**REFER:** Được gửi từ UA để yêu cầu một UA khác truy cập vào một địa chỉ URI hoặc URL trong trường header field Refer-To nằm trong bản tin REFER. Địa chỉ URI hoặc URL có thể là một địa chỉ SIP URI hoặc một địa chỉ của một trang web. Nếu là một địa chỉ SIP URI, user agent dường như đang thực hiện một dịch vụ chuyển cuộc gọi (transfer call), hoặc là đang tạo ra một cuộc gọi với nhiều bên tham gia (conferencing). Khi nhận được bản tin REFER nếu UA đồng ý truy cập vào địa chỉ URI hoặc URL được cung cấp nó sẽ gửi trả lại bản tin response 202 Accepted. Ví dụ UA nhận được bản tin REFER với địa chỉ SIP URI trong trường Refer-To, UA sau đó sẽ gửi một bản tin Invite tới địa chỉ SIP URI vừa được cung cấp trong bản tin Refer-To

**SUBSCRIBE:** Được gửi từ một UA để yêu cầu nhận một cảnh báo về sự thay đổi của một sự kiện (event) nào đó, ví dụ như sự kiện chuyển cuộc gọi, hoặc sự kiện thiết lập hội nghị (cuộc gọi có nhiều bên tham gia). Bản tin SUBSCRIBE được gửi tới server với trường Event chứa giá trị là tên của sự kiện mà client muốn được nhận thông báo, và trường Expires là khoảng thời gian timeout của bản tin SUBSCRIBE. Nếu server hỗ trợ sự kiện mà client mô tả trong trường Event, server gửi trả lời bằng bản tin response 200 OK, lúc này cứ mỗi lần có một sự thay đổi nào đó của sự kiện mà client mong muốn nhận, server sẽ gửi bản tin NOTIFY thông báo tới client, bản tin NOTIFY sẽ được gửi cho tới khi khoảng thời gian timeout của bản tin SUBSCRIBE hết hạn, lúc này nếu muốn tiếp tục nhận các thông báo từ server, client phải gửi lại bản tin SUBSCRIBE

**NOTIFY:** Được sử dụng bởi UA để gửi một thông báo về sự thay đổi của một sự kiện tới UA phát ra bản tin SUBSCRIBE. Nội dung của phần thông báo chứa trong phần message body của bản tin NOTIFY và thường được định dạng theo kiểu XML.

**MESSAGE:** Được sử dụng để mang nội dung của các tin nhắn nhanh (Instant message) được gửi giữa các UA. Nội dung của IM được chứa trong phần message body của bản tin MESSAGE. Bản tin quan trọng nhất trong SIP là INVITE (hình 3.7) được sử dụng để thiết lập một phiên giữa những người tham gia. Một phiên là một tập hợp những người tham gia và các luồng phương tiện truyền thông giữa chúng.

Request Line	INVITE sip:bob@biloxi.example.com SIP/2.0
Headers	Via: SIP/2.0/TCP client.atlanta.example.com:5060;branch=z9hG4bK74bf9 Max-Forwards: 70 From: Alice <sip:alice@atlanta.example.com>;tag=9fxced76sl To: Bob <sip:bob@biloxi.example.com> Call-ID: 3848276298220188511@atlanta.example.com CSeq: 1 INVITE Contact: <sip:alice@client.atlanta.example.com;transport=tcp> Content-Type: application/sdp Content-Length: 151
Empty Line	
Message Body	v=0 o=alice2890844526 2890844526 IN IP4 client.atlanta.example.com S=- c=IN IP4 192.0.2.101 t=00 m=audio 49172 RTP/AVP 0 a=rtpmap:0 PCMU/8000

Hình 3.7: Ví dụ bản tin yêu cầu (INVITE)

#### (ii) Bản tin đáp ứng

Các đáp ứng SIP được gửi từ máy chủ đến máy khách cho biết trạng thái của yêu cầu và được phân biệt với bản tin yêu cầu bởi dòng trạng thái làm dòng bắt đầu của bản tin. Dòng trạng thái bao gồm phiên bản giao thức, theo sau là mã trạng thái và dòng text giải thích cho mã này (hình 3.8).

Mã trạng thái là một số nguyên gồm ba chữ số cho biết đáp ứng của một yêu cầu. Tiếp theo là viết tắt mô tả bằng văn bản của mã trạng thái. Mã Trạng thái là nhằm mục đích sử dụng cho tự động hóa, trong khi phần văn bản dành cho người dùng con người. Các phản hồi SIP cũng tương tự như HTTP, nhưng được định nghĩa trong ngữ cảnh của SIP. Chữ số thứ nhất của trạng thái xác định loại của phản ứng. Hai chữ số tiếp theo không có vai trò gì. Vì vậy, bất kỳ phản hồi nào có mã trạng thái giữa 100 và 199 được gọi là phản hồi 1xx, bất kỳ phản hồi nào với mã trạng thái từ 200 đến 299 dưới dạng phản hồi 2xx, v.v. SIP định nghĩa sáu lớp của các bản tin đáp ứng, các lớp từ 1xx tới 5xx hầu như là tương tự với các bản tin response của giao thức HTTP, riêng lớp 6xx được định nghĩa riêng cho SIP.

- 1xx - Là các bản tin tạm thời hoặc là các bản tin thông báo các thông tin phản hồi. Khi một UA nhận được một bản tin yêu cầu nó sẽ gửi ngay lập tức lại một bản tin ở lớp 1xx để thông báo rằng nó đã nhận được bản tin và đang xử lý bản tin đó.

- 2xx – Là bản tin báo thành công: Bản tin yêu cầu đã được nhận thành công, đã hiểu và được chấp nhận.
- 3xx - Là các bản tin chuyển hướng, nó được gửi bởi SIP server có chức năng như là Redirect server. Ví dụ, nếu một Proxy server nhận được một bản tin Invite và nó không thể định vị được địa chỉ của phía nhận, khi đó nó sẽ gửi trả lại phía gọi một bản tin lớp 3xx để thông báo cho phía gọi sử dụng một địa chỉ khác.
- 4xx – Là các bản tin thông báo thất bại, có nghĩa rằng phía nhận (server) không thể xử lý được bản tin yêu cầu.
- 5xx – Là bản tin báo sự cố của server: Máy chủ không thể hoàn thành một yêu cầu đường như hợp lệ
- 6xx – Là bản tin báo sự cố toàn mạng: Yêu cầu không thể được thực hiện ở bất kỳ máy chủ nào

Status Line	SIP/2.0 200 OK
Headers	<p>Via: SIP/2.0/TCP client.atlanta.example.com:5060;branch=z9hG4bK74bf9;received=192.0.2.101</p> <p>From: Alice &lt;sip:alice@atlanta.example.com&gt;;tag=9fxced76sl</p> <p>To: Bob &lt;sip:bob@biloxi.example.com&gt;;tag=8321234356</p> <p>Call-ID: 3848276298220188511@atlanta.example.com</p> <p>CSeq: 1 INVITE</p> <p>Contact: &lt;sip:bob@client.biloxi.example.com;transport=tcp&gt;</p> <p>Content-Type: application/sdp</p> <p>Content-Length: 147</p>
Empty Line	
Message Body	<p>v=0</p> <p>o=bob 2890844527 2890844527 IN IP4 client.biloxi.example.com</p> <p>s=-</p> <p>c=IN IP4 192.0.2.201</p> <p>t=00</p> <p>m=audio 3456 RTP/AVP 0</p> <p>a=rtpmap:0 PCMU/8000</p>

Hình 3.8: Ví dụ bản tin đáp ứng

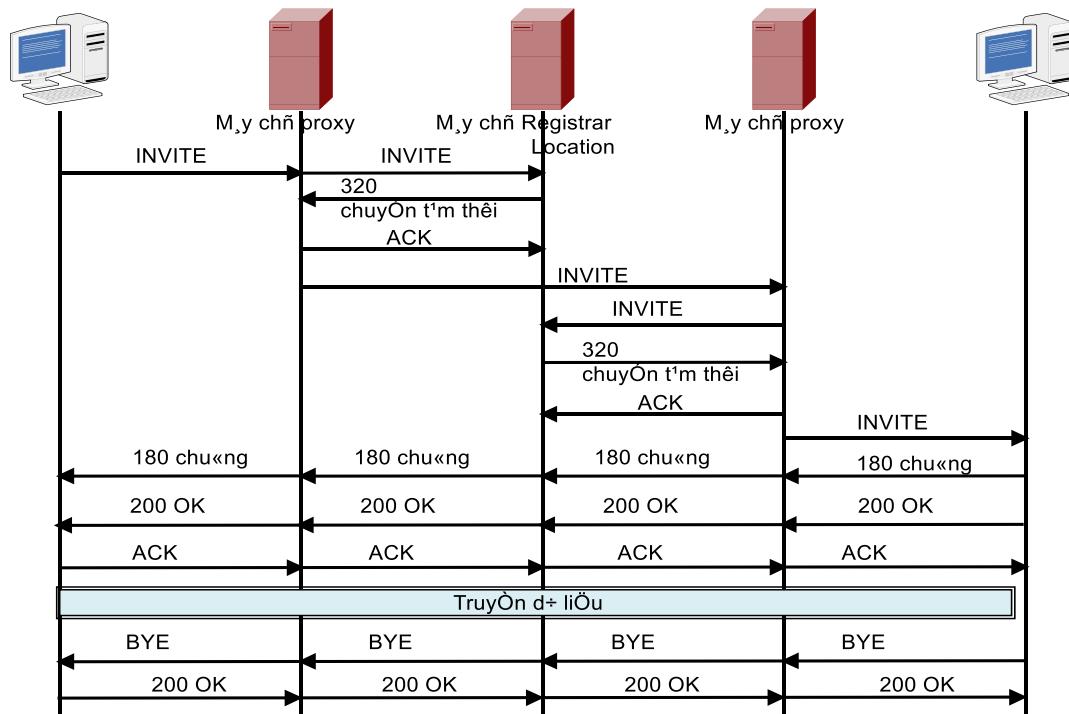
#### 3.2.2.4 Thủ tục báo hiệu SIP

Trong mạng SIP quá trình thiết lập và huỷ một phiên kết nối thường gồm có 6 bước như sau:

1. Đăng ký, khởi tạo và định vị đầu cuối.

2. Xác định media của cuộc gọi, tức là mô tả phiên mà đầu cuối được mời tham dự.
3. Xác định mong muốn của đầu cuối bị gọi, trả lời hay không. Phía bị gọi phải gửi bản tin xác nhận chấp thuận cuộc gọi hoặc từ chối.
4. Thiết lập cuộc gọi.
5. Thay đổi hay điều khiển cuộc gọi (thí dụ như chuyển cuộc gọi).
6. Huỷ cuộc gọi.

Quá trình thiết lập và huỷ cuộc gọi SIP cơ bản được mô tả trên hình 3.9. Trong hội thoại SIP, mỗi bên tham gia (bên chủ gọi và bên bị gọi) được gắn một địa chỉ SIP. Người sử dụng phải đăng ký vị trí của họ với máy chủ SIP. Để tạo một cuộc gọi SIP, phía chủ gọi định vị tới máy phục vụ thích ứng và sau đó gửi một yêu cầu SIP. Hoạt động SIP thường xuyên nhất là lời mời các thành viên tham gia hội thoại. Máy chủ đăng ký đóng vai trò tiếp nhận các yêu cầu đăng ký từ UA và lưu trữ các thông tin này tại một dịch vụ phi SIP (Non-SIP) là máy chủ định vị.



Hình 3.9: Thủ tục báo hiệu trong SIP

(i) Quá trình định vị tới máy chủ SIP

Khi một máy khách muốn gửi đi một yêu cầu, máy khách sẽ gửi bản tin yêu cầu đó tới SIP máy chủ Proxy, hoặc tới địa chỉ IP và cổng tương ứng trong địa chỉ của yêu cầu SIP

(Request-URI). Trường hợp đầu, yêu cầu được gửi tới máy chủ Proxy không phụ thuộc vào địa chỉ của yêu cầu. Với trường hợp sau, máy khách phải xác định giao thức, cổng và địa chỉ IP của máy chủ mà yêu cầu được gửi đến. Một máy khách thực hiện các bước tiếp theo để có được những thông tin này. Máy khách cố gắng liên lạc với máy chủ theo số cổng được chỉ ra trong địa chỉ yêu cầu SIP (Request-URI). Nếu không có số cổng nào chỉ ra trong Request-URI, máy khách sẽ sử dụng địa chỉ cổng mặc định là 5060. Nếu Request-URI chỉ rõ là sử dụng giao thức TCP hay UDP, máy khách sẽ làm việc với máy chủ theo giao thức đó. Nếu không có giao thức nào được chỉ ra thì Client cố gắng dùng giao thức UDP (nếu không hỗ trợ TCP) hoặc sử dụng giao thức TCP cho hoạt động của mình (chỉ được hỗ trợ TCP mà không được hỗ trợ UDP). Máy khách cố gắng tìm một hay nhiều địa chỉ cho máy chủ SIP bằng việc truy vấn DNS (Domain Name System) theo các thủ tục sau:

- Nếu địa chỉ Host trong địa chỉ Request-URI là một địa chỉ IP thì máy khách làm việc với máy chủ bằng địa chỉ được đưa ra. Nếu đó không phải là một địa chỉ IP, máy khách thực hiện bước tiếp theo
- Máy khách đưa ra câu hỏi tới DNS Server về bản ghi địa chỉ cho địa chỉ Host trong địa chỉ Request-URI. DSN sẽ trả về một bản ghi danh sách các địa chỉ. Lúc đó việc lựa chọn một trong các địa chỉ này là tùy ý. Còn nếu DNS Server không đưa ra bản ghi địa chỉ, máy khách sẽ kết thúc hoạt động, có nghĩa nó không thực hiện được việc định vị máy chủ. Nhờ bản ghi địa chỉ, sự lựa chọn tiếp theo cho giao thức mạng của máy khách có nhiều khả năng thành công hơn. Một quá trình thực hiện thành công là quá trình có một bản ghi chứa trong phần trả lời và máy chủ làm việc ở một trong những địa chỉ chứa trong trả lời đó

#### (ii) Giao dịch SIP

Khi có địa chỉ IP của máy chủ SIP thì yêu cầu sẽ được gửi đi theo tầng vận chuyển bằng giao thức TCP hay UDP. Máy khách gửi một hoặc nhiều yêu cầu SIP đến máy chủ đó và nhận lại một hoặc nhiều các phúc đáp từ máy chủ. Một yêu cầu cùng với các phúc đáp được tạo ra bởi yêu cầu đó tạo thành một giao dịch SIP. Tất cả các phúc đáp cho một yêu cầu mang cùng các giá trị trong các trường: Call – ID, Cseq, To, và From. Yêu cầu ACK xác định sự nhận một phúc đáp INVITE không là một phần của giao dịch vì nó có thể di chuyển giữa một tập các host khác nhau. Mỗi cuộc gọi trong SIP được định danh bởi một trường định danh cuộc gọi (Call-ID).

Một yêu cầu phải cần có thông tin gửi đi từ đâu (From) và tới đâu (To). Trường CSeq lưu trữ thông tin về phương thức sử dụng trong phiên, trường CSeq có dạng: CSeq = “CSeq”: “DIGIT Method”. Trong đó DIGIT là số nguyên không dấu 32 bit.

Nếu một giao thức điều khiển luồng tin cây được sử dụng, yêu cầu và các phúc đáp trong một giao dịch đơn lẻ được mang trên cùng kết nối. Một vài yêu cầu SIP từ cùng máy khách đến cùng máy chủ có thể sử dụng cùng kết nối hoặc có thể sử dụng một kết nối mới cho mỗi yêu cầu.

Nếu một máy khách gửi yêu cầu thông qua một giao thức datagram đơn hướng như UDP thì các UA thu sẽ định hướng phúc đáp theo thông tin chứa trong các trường mào đầu Via. Mỗi proxy server trong tuyến chuyển tiếp của yêu cầu chuyển tiếp phúc đáp sử dụng các trường mào đầu Via này.

#### *(iii) Lời mời SIP*

Một lời mời SIP thành công gồm hai yêu cầu INVITE và ACK. Yêu cầu INVITE thực hiện lời mời một thành viên tham gia hội thoại. Khi phía bị gọi đồng ý tham gia, phía chủ gọi xác nhận đã nhận một bản tin đáp ứng bằng cách gửi đi một yêu cầu ACK. Nếu phía chủ gọi không muốn mời thành viên tham gia cuộc gọi nữa nó sẽ gửi yêu cầu BYE thay cho ACK.

Thông điệp INVITE chứa thành phần mô tả phiên SDP và phương thức tiến hành trao đổi ứng với phiên đó. Với các phiên đa hướng, phần mô tả phiên liệt kê kiểu và khuôn dạng của các phương tiện (Media) để phân phối cho phiên hội thoại. Với một phiên đơn hướng, phần mô tả phiên liệt kê kiểu và khuôn dạng của các phương tiện mà phía chủ gọi muốn sử dụng và nêu những dữ liệu muốn gửi đi.

#### *(iv) Định vị người dùng*

Một đối tượng bị gọi có thể di chuyển giữa một số các hệ thống đầu cuối khác nhau theo thời gian. Một máy chủ định vị cũng có thể sử dụng một hay nhiều giao thức khác nhau để xác định hệ thống đầu cuối mà tại đó một người sử dụng có thể liên lạc. Một máy chủ định vị có thể đưa ra một vài vị trí vì người sử dụng được đăng nhập vào tại một vài host đồng thời hoặc bởi vì máy chủ định vị lỗi. Máy chủ SIP kết hợp các kết quả để đưa ra một danh sách các vị trí.

Đối với từng kiểu SIP Server thì hoạt động sau khi nhận được danh sách các vị trí khác nhau là khác nhau. Một SIP Redirect Server sẽ trả lại danh sách địa chỉ cho Client với các mào đầu Contact. Một SIP proxy server có thể thử lần lượt hoặc song song các địa chỉ cho đến khi cuộc gọi thành công (phúc đáp 2xx) hoặc bên bị gọi từ chối cuộc gọi (phúc đáp 6xx).

Nếu một proxy server chuyển tiếp một yêu cầu SIP, nó phải bổ sung địa chỉ của nó vào vị trí bắt đầu của danh sách các trạm chuyển tiếp được ghi trong các mào đầu Via. Đầu

vết Via đảm bảo rằng các trả lời có thể đi theo cùng tuyến đó theo hướng ngược lại, việc đảm bảo hoạt động chính xác nhờ tuân theo các tường lửa và tránh lặp lại yêu cầu. Ở hướng phúc đáp, mỗi host phải xoá bỏ Via của nó, do đó thông tin định tuyến nội bộ được che khuất đối với phía bị gọi và các mạng bên ngoài.

(v) *Thay đổi một phiên hiện tại*

Trong một vài trường hợp, cần phải thay đổi các thông số của phiên hội thoại hiện tại. Việc đó được thực hiện bởi việc phát lại các yêu cầu INVITE. Các yêu cầu INVITE đó có cùng trường Call-ID nhưng có trường mào đầu và trường bản tin khác với yêu cầu ban đầu để mang thông tin mới. Các bản tin INVITE đó phải có chỉ số CSeq cao hơn các yêu cầu trước. Ví dụ: có hai thành viên đang hội thoại và muốn có thêm một người thứ ba tham gia. Một trong hai thành viên sẽ mời thành viên thứ ba tham gia với một địa chỉ đa hướng (Multicast) mới và đồng thời gửi một bản tin INVITE đến thành viên thứ hai với trường miêu tả phiên đa hướng nhưng có trường Call-ID cũ.

### 3.4 Hoạt động điều hành của SIP

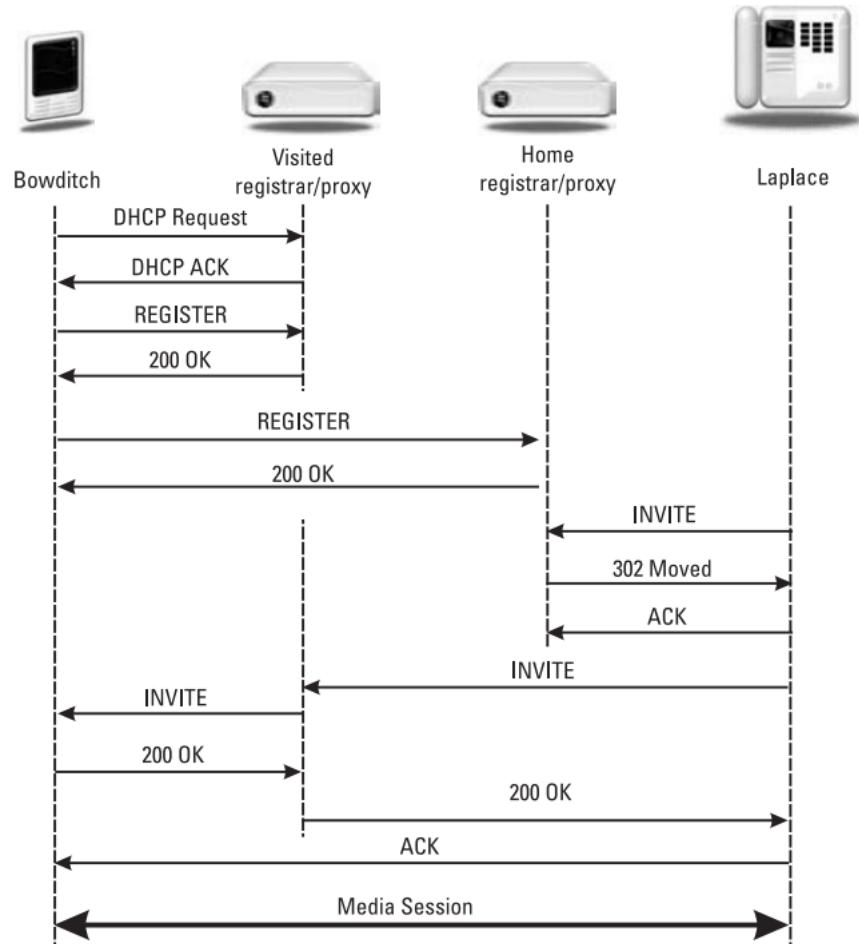
Trong phần này sẽ trình bày một số ứng dụng điều hành tiêu biểu của SIP bao gồm SIP trong thiết bị di động và SIP trong IMS.

#### 3.4.1 Hỗ trợ tính di động (SIP Mobility)

Khả năng di động cá nhân là khả năng có một địa chỉ (nhận dạng) không đổi trên một số lượng thiết bị. SIP URI có chứa thuộc tính này và về cơ bản được hỗ trợ bởi SIP. SIP cũng có thể hỗ trợ tính di động của dịch vụ (khả năng người dùng giữ dịch vụ khi di động) mặc dù một số quy ước và tiện ích mở rộng đã được đề xuất để cung cấp điều này trong một số kiến trúc nhất định.

Tính di động cá nhân được hỗ trợ SIP thông qua việc sử dụng các bản tin đăng ký (REGISTER), nó cho phép thiết bị di động thay đổi địa chỉ IP và điểm kết nối của nó tới mạng Internet và vẫn có thể nhận các cuộc gọi đến. Bản tin đăng ký trong SIP tạm thời liên kết một AOR URI của người dùng với contactURI của một thiết bị cụ thể. Khi địa chỉ IP của thiết bị thay đổi, việc đăng ký cho phép thông tin này được cập nhật tự động trong mạng SIP. Một thiết bị đầu cuối cũng có thể di chuyển giữa các nhà cung cấp dịch vụ bằng cách sử dụng nhiều lớp đăng ký, trong đó việc đăng ký thực sự được thực hiện bằng cách sử dụng một contact làm địa chỉ bản ghi với nhà cung cấp dịch vụ khác. Ví dụ, hãy xem xét UA trong 3.10, đã tạm thời lấy được một SIP URI mới với một nhà cung cấp dịch vụ mới. (Các lý do để làm như vậy có thể bao gồm bảo mật, truyền tải qua NAT / tường lửa

hoặc chính sách cục bộ.) Sau đó, UA thực hiện đăng ký kép như thể hiện trong hình 3.10. Bản tin đăng ký đầu tiên là với nhà cung cấp dịch vụ mới, từ đó liên kết contact URI của thiết bị với AOR URI của nhà cung cấp dịch vụ mới. Bản tin đăng ký thứ 2 được chuyển trả lại nhà cung cấp dịch vụ ban đầu và cung cấp AOR của nhà cung cấp dịch vụ mới làm contact URI. Tiếp đó, khi có yêu cầu thực hiện cuộc gọi (bản tin INVITE) đến mạng của nhà cung cấp dịch vụ ban đầu, INVITE sẽ được chuyển hướng đến nhà cung cấp dịch vụ mới, sau đó định tuyến cuộc gọi đến người dùng.



Hình 3.10: Hỗ trợ tính di động của thiết bị đầu cuối sử dụng SIP REGISTER

Bản tin SIP REGISTER thứ nhất chứa URI thiết bị như sau:

```

REGISTER sip:registrar.capetown.org SIP/2.0
Via: SIP/2.0/TLS 128.5.2.1:5060;branch=z9hG4bK382112
Max-Forwards: 70
To: Nathaniel Bowditch <sip:bowditch321@capetown.org>
From: Nathaniel Bowditch <sip:bowditch321@capetown.org>
;tag=887865
Call-ID: 54-34-19-87-34-ar-gr
CSeq: 3 REGISTER
Contact: <sip:nat@128.5.2.1>
Content-Length: 0

```

Và bản tin SIP REGISTER thứ hai chứa URI chuyển vùng sẽ như sau:

```

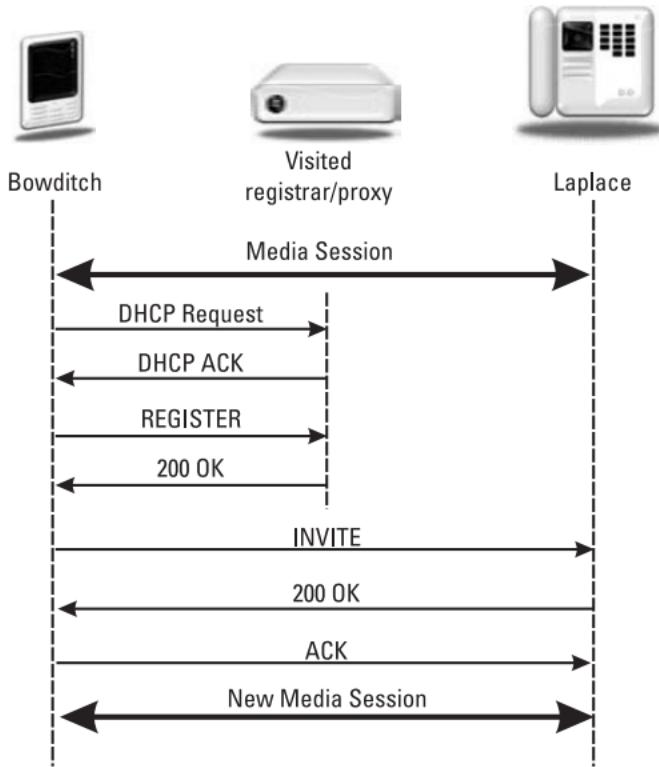
REGISTER sip:registrar.salem.ma.us SIP/2.0
Via: SIP/2.0/TLS 128.5.2.1:5060;branch=z9hG4bK1834
Max-Forwards: 70
To: Nathaniel Bowditch <sip:n.bowditch@salem.ma.us>
From: Nathaniel Bowditch <sip:n.bowditch@salem.ma.us>
;tag=344231
Call-ID: 152-45-N-32-23-W3-45-43-12
CSeq: 6421 REGISTER
Contact: <sip:bowditch321@capetown.org>
Content-Length: 0

```

Bản tin INVITE thứ nhất trong hình 3.10 sẽ được gửi tới địa chỉ [sip:n.bowditch@salem.ma.us](mailto:sip:n.bowditch@salem.ma.us); bản tin INVITE thứ hai sẽ được gửi tới địa chỉ [sip:bowditch321@capetown.org](mailto:sip:bowditch321@capetown.org), được chuyển tiếp tới [sip:nat@128.5.2.1](mailto:sip:nat@128.5.2.1). Bản tin SIP INVITE được chuyển tới Bowditch và cho phép phiên được thiết lập. Cả hai bản tin đăng ký đều cần được làm mới định kỳ.

Một nhược điểm của phương pháp này là SIP hiện không lấy được URI cục bộ. Điều này sẽ phải được thực hiện bằng phương pháp không phải SIP chẳng hạn như đăng ký trang Web, sẽ được kết hợp với các cơ chế xác thực, ủy quyền và tính cước thích hợp.

Bộ đăng ký cục bộ có thể tối ưu hóa điều này bằng cách chuyển tiếp thông tin đăng ký trên UA chuyển vùng tới bộ đăng ký thường trú (home registrar). Không có thay đổi nào đối với các bản tin SIP mà chỉ cần một quy ước được các bộ đăng ký chấp nhận để công nhận việc đăng ký chuyển vùng và thực hiện hành động thích hợp. Có thể các quy ước này được chuẩn hóa nếu các hệ thống xác thực và tính cước được yêu cầu để xử lý các quá trình đăng ký đó được tiêu chuẩn hóa trong tương lai.



Hình 3.11: Hỗ trợ tính di động đang trong cuộc gọi bằng cách gửi lại bản tin INVITE

Trong một phiên, thiết bị di động cũng có thể thay đổi địa chỉ IP khi nó chuyển đổi giữa mạng không dây này với mạng không dây khác. SIP cũng hỗ trợ tình huống này, vì có thể sử dụng lại bản tin INVITE để cập nhật contact URI và thay đổi thông tin phương tiện trong SDP. Điều này được thể hiện bởi các luồng báo hiệu cuộc gọi trong hình 3.11. Tại đây, Bowditch phát hiện một mạng không dây mới, sử dụng DHCP để lấy địa chỉ IP mới, sau đó thực hiện INVITE lại để thực hiện luồng báo hiệu và phương tiện đến địa chỉ IP mới. Nếu UA trong giây lát có thể nhận phương tiện từ cả hai mạng, sự gián đoạn là hầu như không đáng kể. Nếu không cập nhật kịp luồng báo hiệu, một vài gói dữ liệu phương tiện có thể bị mất, dẫn đến cuộc gọi bị gián đoạn đôi chút. Bản tin INVITE gửi lại sẽ xuất hiện như sau:

```

INVITE sip:laplace@client.mathematica.org SIP/2.0
Via: SIP/2.0/UDP 65.32.21.2:5060;branch=z9hG4bK34213
Max-Forwards: 70
To: Marquis de Laplace <sip:laplace@mathematica.org>
;tag=90210
From: Nathaniel Bowditch <sip:n.bowditch@salem.ma.us>
;tag=4552345
Call-ID: 413830e4leoi34ed4223123343ed21
CSeq: 5 INVITE
Contact: <sip:nat@65.43.21.2>
Content-Type: application/sdp
Content-Length: 143

v=0
o=bowditch 2590844326 2590944533 IN IP4 65.32.21.2
s=Bearing
c=IN IP4 65.32.21.2
t=0 0
m=audio 32852 RTP/AVP 96
a=rtpmap:96 iLBC/8000

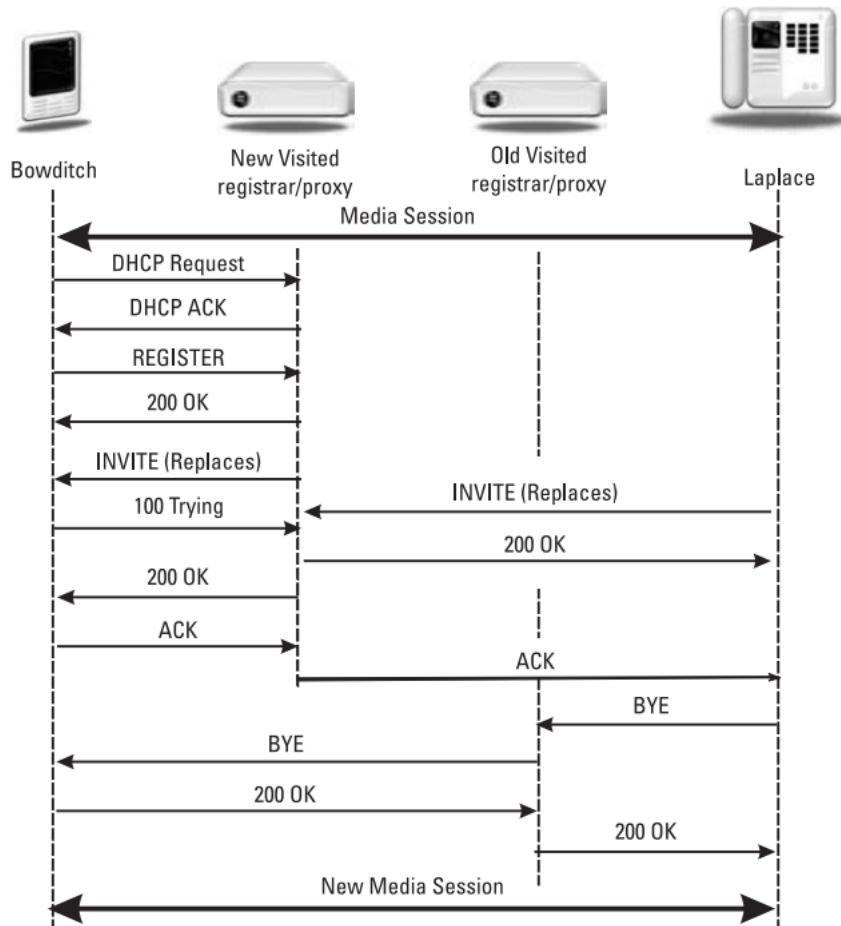
```

Bản tin INVITE gửi lại này có chứa địa chỉ IP mới của Bowditch trong trường tiêu đề Via và contact và thông tin phương tiện SDP.

Lưu ý rằng cả hai tình huống di động trong hình 3.10 và 3.11 không yêu cầu sự hợp tác giữa hai mạng không dây. Do đó, đây là một tình huống hữu ích trong đó UA có thể thực hiện cuộc gọi giữa mạng không dây thương mại và mạng không dây 802.11 gia đình hoặc công ty.

Đối với kịch bản thiết bị di động di chuyển khi đang trong cuộc gọi, trong đó một tập các bộ định tuyến thực tế (tập hợp các SIP proxy mà các bản tin SIP phải đi qua) phải thay đổi, bản tin INVITE gửi lại không thể được sử dụng. Ví dụ, nếu một proxy được yêu cầu cho việc truyền tải NAT/firewall, thì nhiều hơn một contact URI phải được thay đổi, khi đó luồng báo hiệu sẽ thay đổi. Một giải pháp đó là gửi một bản tin INVITE mới với tiêu đề Replaces, để nhận dạng phiên hiện có. Luồng báo hiệu được thể hiện trong hình 3.12. Luồng báo hiệu này tương tự như trong hình 3.11 ngoại trừ việc một bản tin BYE được tạo tự động để kết thúc cuộc gọi hiện có khi bản tin INVITE với trường Replaces được chấp nhận.

Kết quả là Bowditch gửi một bản tin INVITE với trường Replaces để tạo ra cuộc gọi qua máy chủ proxy mới mà không có sự tham gia của máy chủ proxy cũ. Khi Laplace chấp nhận INVITE, một bản tin BYE sẽ tự động được gửi đi để chấm dứt cuộc gọi cũ được định tuyến qua máy chủ proxy cũ. Như vậy phiên truyền thông được thiết lập bằng cách sử dụng địa chỉ IP mới của Bowditch từ SDP trong INVITE.



Hình 3.12: Hỗ trợ tính di động giữa cuộc gọi sử dụng bản tin INVITE với trường Replaces

Các dịch vụ trong SIP có thể được cung cấp trong proxy hoặc UA. Nếu dịch vụ thuộc UA, thì dịch vụ di động sẽ không có vấn đề gì vì người dùng di chuyển xung quanh. Tuy nhiên, việc kết hợp tính di động của dịch vụ và tính di động của cá nhân là đầy thách thức trừ khi các thiết bị của người dùng được cấu hình y hệt nhau với cùng dịch vụ. Ngoài ra, dịch vụ thuộc điểm đầu cuối sẽ chỉ sẵn sàng khi có kết nối mạng. Ví dụ như dịch vụ chuyển tiếp cuộc gọi trên điểm đầu cuối sẽ không thực hiện được nếu đầu cuối này tạm thời bị mất kết nối Internet. Vì vậy một số dịch vụ triển khai trong mạng bằng máy chủ proxy SIP. Đối với những dịch vụ này, dịch vụ tính di động cho UA có nghĩa là sử dụng cùng một bộ proxy để định tuyến và gửi yêu cầu gửi đi khi di động.

Do đặc tính tự nhiên của mạng Internet thì không có lý do gì khiến UA không thể sử dụng cùng một proxy khi được kết nối với Internet tại một điểm khác. Một UA ở Hoa Kỳ và được cấu hình sử dụng tập proxy ở Hoa Kỳ vẫn có thể sử dụng những proxy đó khi chuyển vùng sang Châu Âu chẳng hạn. Có lẽ các bước nhảy SIP sẽ có độ trễ cao hơn một

chút do nhảy qua nhiều của bộ định tuyến hơn và yêu cầu thiết lập cuộc gọi có thể lâu hơn một hoặc hai giây. Tuy nhiên, điều này không ảnh hưởng đến chất lượng của phiên truyền thông vì phương tiện luôn chảy trực tiếp giữa hai UA và không phải truyền qua các máy chủ proxy SIP. Do đó, SIP có thể dễ dàng hỗ trợ tính di động của dịch vụ qua mạng Internet.

Các khả năng di động SIP này rất thích hợp để sử dụng qua mạng không dây như 802.11 trong nhà, ở cơ quan hay nơi công cộng. Các thỏa thuận chuyển vùng cho phép các “điểm phát sóng” không dây được liên kết trong các khu vực đô thị cung cấp một dịch vụ không dây. Tuy nhiên, các nhà cung cấp dịch vụ không dây thương mại có kế hoạch cung cấp mạng điện thoại không dây sử dụng SIP thì SIP cần được mở rộng và phát triển.

### **3.4.2 Hoạt động của SIP trong IMS**

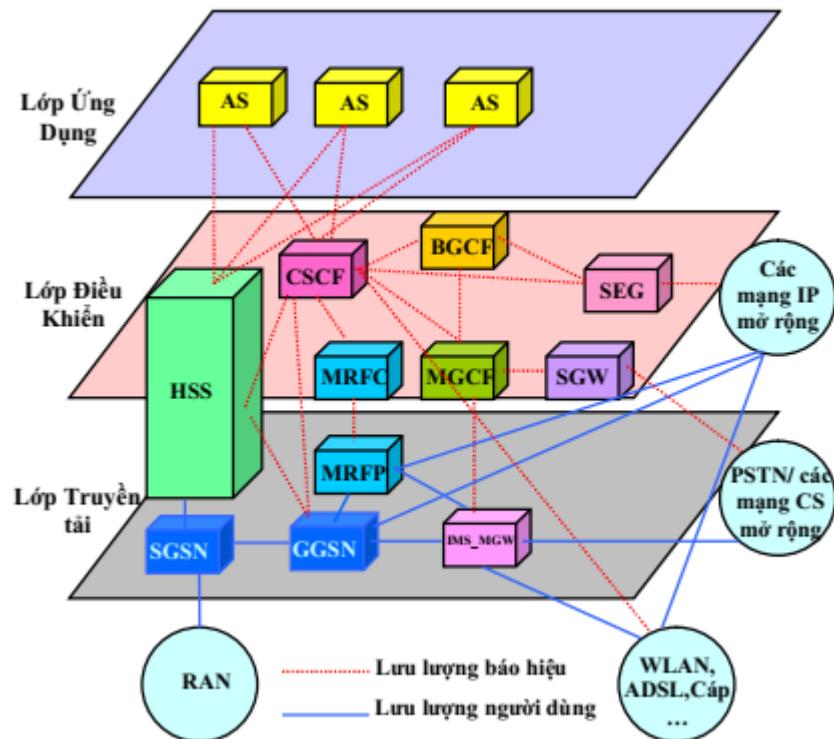
#### **3.4.2.1 Kiến trúc IMS**

Tiếp cận phân hệ đa phương tiện IP (IMS) đóng vai trò then chốt trong quá trình hội tụ giữa mạng cố định, di động và internet hiện nay. IMS là một kiến trúc mạng dùng để thao tác, quản lý và điều khiển các dịch vụ đa phương tiện đến người dùng cố định và di động. IMS hỗ trợ nhiều phương thức truy nhập như GSM, UMTS, CDMA2000, truy nhập hữu tuyến băng rộng như cáp xDSL, cáp quang, cáp truyền hình, cũng như truy nhập vô tuyến băng rộng WLAN, WiMAX. IMS định nghĩa một lớp quản lý dịch vụ chung cho tất cả các loại hình dịch vụ đa phương tiện, độc lập với loại hình mạng truy nhập mà người dùng đang kết nối. IMS xây dựng trên nền mạng lõi IP và cho phép nhiều mạng truy nhập khác, bao gồm cả mạng di động lẫn mạng cố định, kết nối với nhau thông qua lớp dịch vụ chung để cung cấp các gói dịch vụ hội tụ. Một mạng IMS được định nghĩa trong một kiến trúc mặt phẳng ngang gồm 3 lớp chức năng

Lớp đầu tiên là lớp mang. Lớp này truyền tải dung lượng báo hiệu và các luồng lưu lượng đa phương tiện. Lớp này bao gồm các thiết bị phần cứng như thiết bị chuyển mạch, bộ định tuyến và các thực thể xử lý phương tiện như cổng đa phương tiện hay máy chủ phương tiện. IMS đóng vai trò như một lớp truy nhập không phụ thuộc mạng để kết nối đến nhiều loại mạng khác nhau hiện có.

Lớp thứ hai trong kiến trúc IMS là lớp điều khiển. Bao gồm các phần tử của mạng báo hiệu như CSCF, HSS, MGCF... để hỗ trợ điều khiển phiên chung, điều khiển phương tiện và chức năng điều khiển truy nhập qua các giao thức báo hiệu như SIP, Diameter, H248. Lớp điều khiển là chức năng cốt lõi của IMS để truyền thông báo hiệu và các yêu cầu điều khiển tới các thành phần thiết bị trong phiên.

Lớp thứ 3 trong kiến trúc IMS là lớp dịch vụ. Lớp này bao gồm các Server ứng dụng như server ứng dụng SIP, Server truy nhập dịch vụ mở bên thứ 3 và các điểm điều khiển dịch vụ mở kế thừa từ các hệ thống truyền thông. IMS đưa ra các điều khiển dịch vụ thông qua mạng thuê bao nhà, các thành phần của mạng báo hiệu được phân phối trong lớp dịch vụ và lớp điều khiển.



Hình 3.13: Kiến trúc phân lớp của phân hệ IMS

#### 3.4.2.2 Triển khai SIP trong IMS

Kiến trúc IMS sử dụng SIP làm giao thức thiết lập và điều khiển cuộc gọi đa phương tiện. Cùng với việc điều khiển cuộc gọi, SIP cũng hỗ trợ các chức năng như di động của người sử dụng và chuyển hướng cuộc gọi trong IMS. Khi triển khai SIP trong IMS các nhà phát triển nhận ra rằng có sự khác biệt so với phiên bản SIP cho Internet. Một số các mở rộng được định nghĩa trong các RFC bổ sung thêm các tính năng mới và làm cho SIP trở thành giao thức báo hiệu khá phức tạp. Việc sử dụng SIP cho việc thiết lập phiên trên những liên kết bằng thông hạn chế như các giao diện vô tuyến hoặc các liên kết nối tiếp tốc độ thấp dẫn đến thời gian thiết lập cuộc gọi dài. Để khắc phục yếu điểm đó cơ chế nén báo hiệu gọi là SigComp đã được phát triển bởi tổ chức IETF. Tiêu đề riêng P-Header (RFC3329) như P-preferred-identity, th P-access-network-info, P-asserted-identity,

Pcalledparty- id được bổ sung thêm cho mạng IMS để cung cấp các dịch vụ riêng biệt. Các tiêu đề này được định nghĩa thêm để chuyển các thông tin xác đáng vào mạng nhưng nó chưa đủ để phát triển các phần tử chuẩn mực trong IMS. Các chuẩn mở rộng khác như chuẩn thỏa thuận bảo mật (RFC 3329), xác thực phương tiện (RFC 3313), dành trước tài nguyên trong IMS (RFC 3312), SDP mở rộng được đề xuất hỗ trợ thêm cho SIP trong IMS. So sánh với SIP của trong IETF mà ở đó chủ gọi sử dụng SIP yêu cầu một con đường cụ thể trong tiêu đề Route. Trong IMS, PCSCF loại bỏ con đường này và đảm bảo tuân theo việc định tuyến SIP IMS. Các yêu cầu SIP luôn được định tuyến đến S-CSCF mạng nhà ở cả mạng khởi tạo và kết cuối. S-CSCF sử dụng cơ sở dữ liệu người dùng (download xuống trong quá trình đăng ký) để liên kết với các AS SIP xử lý các yêu cầu SIP. Các tiêu chí lọc khởi tạo lúc đầu IFC (The Initial Filter Criteria) trong cơ sở dữ liệu bao cung cấp một logic đơn giản để quyết định sẽ liên kết với AS nào. Các luật này mang tính ổn định tức là nó không thay đổi trong một chu kỳ.

SIP được triển khai trong lõi IMS trên thực thể chức năng điều kiện phiên gọi CSCF. CSCF thực chất là một máy chủ SIP và đóng vai trò trung tâm của IMS. CSCF có nhiệm vụ xử lý báo hiệu SIP trong IMS. Có ba loại chức năng điều khiển phiên khác nhau: CSCF uỷ quyền (ProxyCSCF: P-CSCF); CSCF phục vụ (Serving-CSCF: S-CSCF) và CSCF tham vấn (Interrogating-CSCF: I-CSCF). Mỗi CSCF có nhiệm vụ riêng. Thường thì tất cả các CSCF tham gia trong suốt quá trình đăng ký thiết lập phiên và định hình cơ chế định tuyến SIP. Ngoài ra, tất cả các chức năng đều có khả năng gửi số liệu tính cước tới bộ chức năng tính cước offline. Có vài chức năng thông thường mà P-CSCF và S-CSCF có thể thực hiện. Các thực thể có khả năng giải phóng phiên thay cho thuê bao (ví dụ khi S-CSCF phát hiện ra một phiên đang treo - không sử dụng, hoặc PCSCF nhận được thông báo kênh mang truyền thông bị mất) và có khả năng kiểm tra nội dung của giao thức mô tả phiên (SDP) hoặc kiểm tra các loại hoặc các mã truyền thông trong giao thức này. Khi SDP đang sử dụng không phù hợp với chính sách của nhà khai thác, CSCF từ chối yêu cầu và gửi bản tin thông báo lỗi SIP tới UE.

### *CSCF đại diện (uỷ quyền)*

P-CSCF là điểm kết nối, giao tiếp đầu tiên của các thuê bao trong hệ thống IMS. Có nghĩa là tất cả lưu lượng báo hiệu SIP từ UE sẽ được gửi tới P-CSCF. Ngược lại, tất cả các kết cuối báo hiệu SIP từ mạng được gửi từ P-CSCF tới UE. Bốn chức năng cơ bản của P-CSCF bao gồm: nén SIP, kết hợp bảo mật IP (IPSec), tương tác với chức năng quyết định chính sách (PDF) và xác định phiên khẩn cấp. Có thể có một hoặc nhiều P-CSCF trong một mạng. P-CSCF thực hiện các chức năng sau:

- + Chuyển tiếp các yêu cầu SIP REGISTER tới CSCF truy vấn (I-CSCF) dựa trên tên miền do UE cung cấp.
- + Chuyển tiếp các yêu cầu và đáp ứng SIP của UE tới CSCF phục vụ (S-CSCF).
- + Chuyển tiếp các yêu cầu và đáp ứng SIP tới UE.
- + Phát hiện các yêu cầu thiết lập phiên.
- + Tạo thông tin tính cước để gửi cho nút tính cước CCF.
- + Bảo vệ toàn vẹn báo hiệu SIP và duy trì liên kết bảo mật giữa UE và P-CSCF.
- + Chức năng này được cung cấp bởi giao thức bảo mật IPsec và tải tin bảo mật đóng gói ESP.
- + Nén và giải nén các bản tin SIP từ UE. P-CSCF hỗ trợ nén bản tin dựa trên ba RFC: [RFC3320], [RFC3485] và [RFC3486].
- + Chức năng kiểm tra phương tiện. P-CSCF có thể kiểm tra nội dung tải tin giao thức mô tả phiên (SDP) và kiểm tra xem nó chứa các loại phương tiện hay codec. Khi SDP không phù hợp với chính sách của nhà khai thác thì P-CSCF sẽ loại bỏ yêu cầu và gửi bản tin báo lỗi SIP tới UE.
- + Duy trì bộ định thời phiên. Các bộ định thời phiên cho phép P-CSCF phát hiện và giải phóng tài nguyên do các phiên đang bị treo chiếm dụng.
- + Tương tác với chức năng quyết định chính sách (PDF). PDF chịu trách nhiệm triển khai chính sách vùng theo dịch vụ (SBLP). Trong Release 5, PDF là một thực thể logic của P-CSCF, còn trong Release 6 PDF đứng riêng một mình.

Thông thường một mạng IMS sẽ có nhiều P-CSCF tùy thuộc vào quy mô và độ dư của mạng. Mỗi P-CSCF chỉ phục vụ một số lượng các đầu cuối IMS nhất định

### *CSCF truy vấn*

CSCF truy vấn (I-CSCF) là một SIP Proxy nằm tại biên giới của vùng quản lý. Địa chỉ của các I-CSCF trong một miền sẽ được liệt kê trong các bản ghi DNS của miền đó. Khi muốn xác định bước nhảy tiếp theo cho một bản tin nào đó của thủ tục SIP thì máy chủ SIP phải biết được địa chỉ của ít nhất là một I-CSCF của miền mà bản tin đó cần đến. Có thể có nhiều I-CSCF bên trong một mạng. I-CSCF thực hiện các chức năng sau:

- + Liên lạc với HSS để thu được tên của S-CSCF đang phục vụ khách hàng.
- + Đăng ký (gán) một S-CSCF dựa trên dung lượng nhận được từ HSS.
- + Tạo và gửi thông tin tính cước tới nút tính cước CCF.

- + Cung cấp chức năng che giấu. I-CSCF có chứa một tính năng gọi là THIG –công liên mạng che giấu cấu hình. THIG được sử dụng để che cấu hình và dung lượng của mạng từ phía bên ngoài mạng của nhà khai thác.

Số lượng I-CSCF trong một mạng tùy thuộc vào quy mô và độ dư của mạng đó.

#### *CSCF phục vụ*

CSCF phục vụ (S-CSCF) là một máy chủ SIP đóng vai trò trung tâm của mặt bằng báo hiệu với chức năng chủ yếu là điều khiển phiên. Ngoài tư cách là một máy chủ thì S-CSCF còn hoạt động như một bộ đăng ký SIP, có nghĩa nó chứa một ràng buộc giữa vị trí khách hàng (là địa chỉ IP của thiết bị đầu cuối nơi khách hàng đăng nhập) và địa chỉ SIP của bản ghi thuộc về khách hàng đó (còn gọi là nhận dạng chung cho khách hàng). Có thể có nhiều S-CSCF bên trong mạng. S-CSCF thực hiện các chức năng sau:

- + Điều khiển các yêu cầu đăng ký như một register. S-CSCF nhận biết được địa chỉ IP của UE và P-CSCF nào đang được UE sử dụng như một điểm truy cập IMS.
- + Nhận thực người dùng bằng cơ chế nhận thực và đồng thuận khoá IMS (AKA) giữa UE và mạng nhà.
- + Tải thông tin người dùng và dữ liệu liên quan đến dịch vụ từ HSS trong suốt quá trình đăng ký hoặc khi xử lý một yêu cầu tới người dùng không được đăng ký.
- + Định tuyến lưu lượng đầu cuối di động tới P-CSCF và định tuyến lưu lượng khởi xướng từ di động tới I-CSCF, thực thể chức năng điều khiển công thoát (BGCF) hay máy chủ ứng dụng (AS).
- + Thực hiện chức năng điều khiển phiên. S-CSCF có thể hoạt động giống như một máy chủ đại diện.
- + Tương tác với các nền tảng dịch vụ.
- + Phiên dịch số E.164 tới URI dùng để nhận dạng tài nguyên hợp nhất sử dụng cơ chế phiên dịch hệ thống tên miền (DNS). Chức năng này là cần thiết do việc định tuyến cho một bản tin SIP trong IMS chỉ sử dụng các SIP URI, nghĩa là trong trường hợp một khách hàng quay một số điện thoại thay vì sử dụng SIP URI thì S-CSCF phải sử dụng các dịch vụ phiên dịch số.
- + Giám sát bộ định thời đăng ký và có thể đăng ký lại khi cần.
- + Thực hiện kiểm tra phương tiện. S-CSCF có thể kiểm tra nội dung tải tin SDP và kiểm tra xem nó chứa các loại phương tiện hay codec. Khi SDP không phù hợp với chính sách của nhà điều hành hoặc yêu cầu dịch vụ của khách hàng thì SCSCF sẽ loại bỏ yêu cầu và gửi đi bản tin báo lỗi SIP.

- + Duy trì bộ định thời phiên. Nó cho phép S-CSCF phát hiện và giải phóng các tài nguyên do các phiên đang chiếm dụng.
- + Tạo và gửi thông tin tính cước tới nút tính cước CCF để tính cước offline và tới hệ thống OCS để tính cước online.

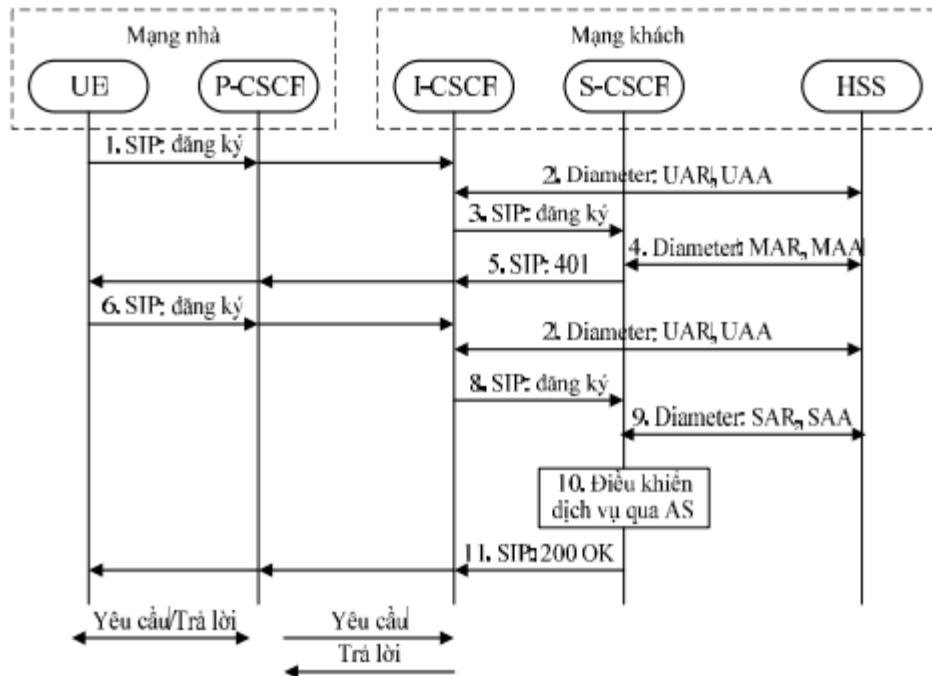
Số lượng S-CSCF trong một mạng phụ thuộc vào quy mô và độ dư của mạng đó. Mỗi S-CSCF chỉ phục vụ cho một số lượng thiết bị đầu cuối IMS nhất định. Khác với P-CSCF và I-CSCF, S-CSCF luôn nằm ở mạng nhà.

#### *3.4.2.3 Một số thủ tục báo hiệu SIP trong IMS*

Để nắm được hoạt động của SIP trong IMS ta xem xét các thủ tục báo hiệu thông qua các ví dụ tương ứng với một số kịch bản có thể xảy ra.

(i) *Đăng ký*: ví dụ thứ nhất chỉ ra một thủ tục khởi tạo đăng ký, cho rằng người dùng đã chuyển mạng sang mạng khách. Thủ tục này bắt đầu với yêu cầu đăng ký SIP người dùng được gửi từ P-CSCF của mạng khách. Vì băng thông vô tuyến hạn chế, bản tin được nén trước khi gửi đi bởi người dùng và được giải nén ở P-CSCF. Nếu có nhiều S-CSCF tồn tại trong mạng nhà của người sử dụng, một I-CSCF cần thiết để triển khai lựa chọn một S-CSCF phục vụ phiên của người dùng đó. Trong trường hợp này P-CSCF quyết định một địa chỉ của I-CSCF mạng nhà của người dùng bằng cách sử dụng tên miền mạng nhà người dùng và chuyển bản tin REGISTER tới I-CSCF. Sau khi I-CSCF gửi đáp ứng nhận thực người dùng (UAR) tới HSS, HSS trả lại địa chỉ khả thi của S-CSCF. I-CSCF lựa chọn một S-CSCF và chuyển bản tin đăng ký.

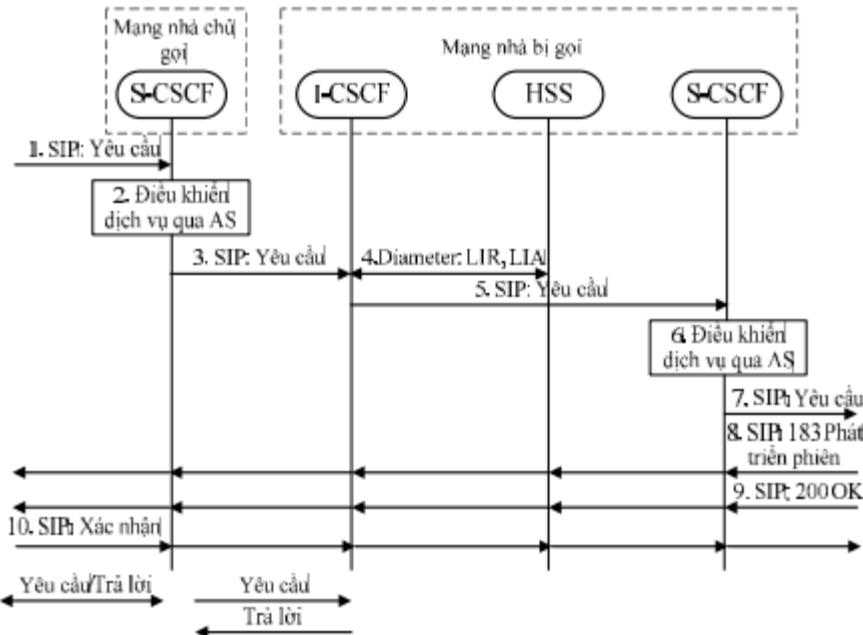
Trong lúc xác nhận đăng ký, S-CSCF lấy lại vector nhận dạng từ HSS qua giao thức Diameter. Đáp ứng nhận thực đa phương tiện MAR và trả lại người dùng bản tin SIP 401 không được nhận thực mà có thể mang số liệu hỏi đáp nhận thực. Sau khi tính toán đáp ứng nhận thực, người dùng gửi đến S-CSCF một bản tin đăng ký khác được mang bởi đáp ứng hỏi đáp. S-CSCF xác nhận lại đáp ứng và nếu đáp ứng đúng, nó tải xuống thuộc tính thuê bao từ HSS qua một đáp ứng yêu cầu chỉ định máy chủ SAR Diameter. S-CSCF có thể liên lạc với một Server ứng dụng để điều khiển dịch vụ như trong thuộc tính của thuê bao trước khi trả lại bản tin 200 OK tới người sử dụng.



Hình 3.14: Luồng bản tin báo hiệu đăng ký

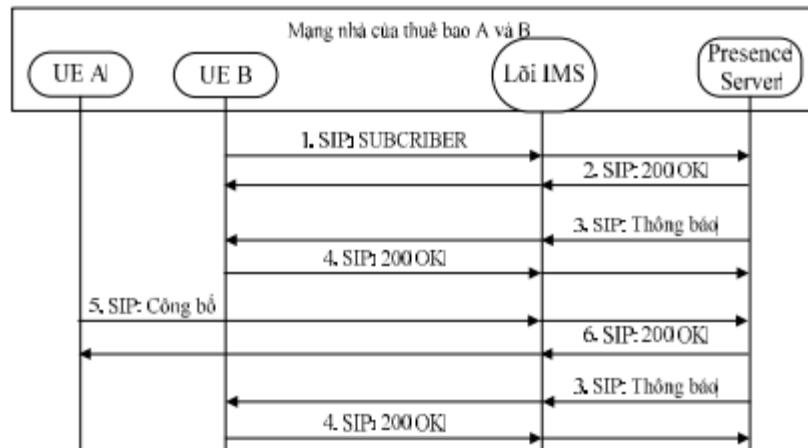
(ii) Thiết lập phiên: Hình 3.15 chỉ ra luồng báo hiệu một thiết lập phiên giữa hai người dùng IMS, cho rằng có nhiều S-CSCF được triển khai. Một thủ tục thiết lập phiên là một quá trình của việc tìm ra các phần tử mạng và các thành phần báo hiệu. Khi định tuyến bản tin đăng ký, I-CSCF của người bị gọi truy vấn HSS của người bị gọi để tìm địa chỉ của một S-CSCF được chỉ định qua bản tin Diameter yêu cầu thông tin vị trí LIR. HSS đáp ứng lại bằng bản tin Diameter trả lời thông tin vị trí LIA

Trước khi gửi bản tin đăng ký, S-CSCF của chủ gọi và người bị gọi có thể liên lạc với Server ứng dụng để điều khiển dịch vụ và tính cước cho dịch vụ tái xuất trong khi đăng ký người dùng. Kỹ thuật phân giải địa chỉ và định tuyến bản tin SIP chuẩn được sử dụng để định tuyến bản tin đăng ký từ chủ gọi tới UE bị gọi trên tất cả các con đường. Các con đường nhận được là UE chủ gọi, PCSCF mạng khách chủ gọi, S-CSCF mạng khách chủ gọi, I-CSCF bị gọi, SCSCF bị gọi, P-CSCF mạng khách bị gọi và UE bị gọi. Bản tin trả lại từ UE bị gọi đi theo đường ngược lại. Thủ tục thỏa thuận một phiên cung cấp trả lời cơ bản cũng được kiểm soát trong thời điểm này. Điều này được hoàn thành qua giao thức mô tả phiên SDP được mang bởi thân của bản tin SIP (ví dụ bản tin đăng ký với một mời gọi và bản tin 200 OK với một trả lời).



Hình 3.15: Luồng bản tin báo hiệu thiết lập phiên

(iii) *Phân phối dịch vụ IMS*: Kiến trúc phân phối dịch vụ IMS bao gồm S-CSCF, Server ứng dụng AS, chức năng điều khiển tài nguyên đa phương tiện MRF và HSS. Trong đó S-CSCF đóng vai trò như một điểm điều khiển phiên trung tâm, các server ứng dụng và MRF là các điểm thi hành dịch vụ. Để thi hành điều khiển logic dịch vụ cho một thuê bao, S-CSCF kiểm tra yêu cầu SIP nhận được. Thông tin được kiểm tra bao gồm kiểu báo hiệu SIP, tiêu đề, URI yêu cầu và mô tả phiên. Nếu điểm chót phù hợp, S-CSCF sẽ lựa chọn một Server ứng dụng và định tuyến yêu cầu SIP tới AS trong dịch vụ được thực thi. Server ứng dụng SIP (SIP AS), cư trú trong mạng nhà và cung cấp dịch vụ dựa trên giao thức SIP. Trong một ví dụ được minh họa trong hình dưới đây chỉ ra luồng bản tin cho thủ tục liên quan đến server có mặt, nơi người dùng A lấy thông tin hiện diện của người dùng B, và dịch vụ hiện diện đóng vai trò như một UA.



*Hình 3.16: Luồng bản tin người dùng A lấy thông tin hiện diện người dùng B*

Các bước trong thủ tục này như sau: 1-2: A lấy thông tin hiện diện của người dùng B; 3-4: Sever hiện diện thông báo cho A về hiệu lực hiện tại của B; 5-6: B thay đổi trạng thái hiệu lực của mình; 7-8: Server hiện diện thông báo cho A bản cập nhật hiệu lực của B.

### 3.5 Kết luận chương

Internet ngày nay cung cấp rất nhiều các dịch vụ tiện ích và trong đó sự chuyển đổi mạnh mẽ các dịch vụ mạng viễn thông truyền thống được tích hợp vào một nền tảng chung. Các dịch vụ đa phương tiện thời gian thực được thực hiện kết nối qua môi trường internet qua các giao thức cơ bản như trình bày trong chương 3. Chương này đã thể hiện các giao thức chính để tạo ra các thủ tục thiết lập kết nối, đảm bảo chất lượng kết nối cũng như quản lý các phiên kết nối đa phương tiện qua nền tảng Internet. Các ứng dụng cụ thể và đặc trưng kỹ thuật đã được đưa ra trong chương này như một chỉ dẫn kết nối cho người đọc.

## CHƯƠNG 4: GIAO THỨC ỨNG DỤNG INTERNET VẠN VẬT

### 4.1 Giới thiệu chung về IoT

#### 4.1.1 Khái niệm về IoT

Viễn cảnh về một thế giới mạng kết nối với các đối tượng thông minh có khả năng biết tất cả mọi thứ về sự vật và về môi trường xung quanh mà không cần đến sự can thiệp của con người. Chỉ bằng việc tính toán, phân tích các dữ liệu được thu thập, các thiết bị tự nó có khả năng điều chỉnh để đưa ra các dự đoán, giao tiếp với các thiết bị khác và đưa ra các quyết định hành động thích hợp, làm giảm đáng kể lãng phí, mất mát và chi phí. Internet of Things (IoT) được kì vọng sẽ trở thành một nền tảng công nghệ sẽ tác động và thúc đẩy mạnh mẽ những thay đổi tích cực trong phương thức sản xuất, thay đổi hoàn toàn cách con người sinh hoạt và tận hưởng cuộc sống.

Thuật ngữ Internet of Things (IoT) được đề cập lần đầu tiên bởi Kevin Ashton, nhà đồng sáng lập trung tâm Auto-ID của đại học MIT, trong một bài thuyết trình của ông về việc ứng dụng RFID vào việc giải quyết một số vấn đề của chuỗi cung ứng tại P&G, nhằm thu hút sự chú ý của các nhà điều hành cấp cao của P&G, Kevin Ashton đã lấy một thuật ngữ đang là xu hướng mới của năm 1999: “the internet” làm tiêu đề bài thuyết trình của mình.

IoT là một mạng lưới mà vạn vật được kết nối với nhau thông qua Internet. Chúng bao gồm các đồ vật, con người được cung cấp một định danh của riêng mình. Tất cả có khả năng truyền tải hay trao đổi thông tin, dữ liệu qua một mạng duy nhất mà không cần đến sự can thiệp, tương tác trực tiếp giữa người với người, hay giữa người với máy tính. IoT đã phát triển từ sự hội tụ của công nghệ không dây, công nghệ vi cơ điện tử. Đặc biệt quan trọng hơn là sự có mặt của Internet. Nói đơn giản hơn thì IoT là một tập hợp các thiết bị có khả năng kết nối mọi thứ lại với nhau thông qua Internet và với thế giới bên ngoài để thực hiện một công việc nào đó.

Hệ thống IoT được phát triển từ giao tiếp máy-máy (Machine to machine - M2M) tức là, các máy kết nối với nhau thông qua mạng mà không có sự tương tác của con người. Kết hợp với công nghệ truyền thông không dây, hệ thống vi cơ điện tử (MEMS), microservice, internet, đồng thời cũng là phần mở rộng của hệ thống SCADA, hệ thống giám sát, thu thập dữ liệu trong thời gian thực và điều khiển các cơ cấu chấp hành từ khoảng cách xa. Khi việc giao tiếp không chỉ dừng ở việc truyền thông giữa các đối tượng hay thiết bị với nhau thông qua internet, mà còn giữa sự vật và con người dựa trên một kiến trúc mang tính kết nối, hệ thống IoT được hình thành.

#### **4.1.2 Đặc điểm của IoT**

Internet vạn vật có những đặc điểm cơ bản sau:

- + **Tính thông minh:** IOT có mạng lưới các thực thể thông minh, chúng có khả năng tự tổ chức và hoạt động riêng lẻ, tùy vào các tình huống và môi trường khác nhau, để từ đó chúng có thể tự liên lạc với nhau để trao đổi thông tin và dữ liệu. Việc tích hợp ứng dụng IOT và hợp trí thông minh có thể giúp các thiết bị, máy móc và phần mềm thu thập được phân tích các dấu vết điện tử của con người khi mà chúng ta tương tác với những thứ thông minh, và cũng từ đó chúng ta có thể phát hiện ra các tri thức mới liên quan đến cuộc sống và hành vi của con người.
- + **Kiến trúc dựa vào sự kiện:** IOT sẽ có thể phản hồi dựa vào các sự kiện được diễn ra trong lúc chúng hoạt động theo thời gian thực.
- + **IOT là hệ thống phức tạp:** IOT chính là một lượng lớn các liên kết giữa các thiết bị, máy móc và dịch vụ với nhau, ngoài ra IOT còn có khả năng thêm vào các nhân tố mới khác.
- + **Kích thước:** IOT là mạng lưới có khả năng chứa đến 50 đến 100 nghìn tỉ các đối tượng kết nối và mạng lưới này có thể theo dõi sự di chuyển của từng đối tượng.

- + **Không gian và thời gian:** IoT có thể thu thập được nhiều dữ liệu với nhau, và đặc biệt là có thể có các dữ liệu thừa về địa điểm, việc xử lý dữ liệu đó được xem như là không hiệu quả, bởi việc xử lý một khối lượng lớn các dữ liệu trong thời gian sẽ có thể đáp ứng cho các hoạt động của các đối tượng khác, và đây cũng chính là một thách thức lớn hiện nay.
- + **Tính không đồng nhất:** Vì các thiết bị trong IoT có phần cứng khác nhau cũng như network khác nhau. Nhờ vào sự liên kết của các network mà các thiết bị giữa các network có thể tương tác với nhau.
- + **Tính kết nối liên thông (interconnectivity):** Với hệ thống IoT thì bất cứ một điều gì, vật gì hay máy móc gì cũng có thể được kết nối với nhau thông qua mạng lưới thông tin và cả cơ sở hạ tầng liên lạc tổng thể.

#### **4.1.3 Tiềm năng và thách thức của hệ thống IoT**

IoT đang dần len lỏi vào cuộc sống của con người. Ý tưởng về một thế giới mà mọi thứ trong cuộc sống được kết nối với Internet để truyền tải, trao đổi dữ liệu, từ đó người dùng có thể tương tác, điều khiển và kiểm soát mọi hoạt động trong cuộc sống thông qua những thiết bị thông minh như điện thoại hoặc máy tính bảng đang dần trở thành sự thật.

Khả năng hàng triệu người kết nối với nhau qua thiết bị thông minh với sức mạnh xử lý, dung lượng lưu trữ và sự tiếp cận tri thức chưa từng có tiền lệ là không giới hạn. Thậm chí, những khả năng đó còn được nhân lên gấp bội nhờ vào tốc độ gia tăng theo cấp số nhân, phạm vi ứng dụng toàn cầu và sự tác động mạnh mẽ của những đột phá về công nghệ mới nổi trong các lĩnh vực như trí thông minh nhân tạo, robot, mạng Internet, phương tiện độc lập, điện toán đám mây, tích hợp thông tin, phân tích số liệu tự động...

IoT sẽ không dừng lại ở những ngôi nhà và ngành nghề thông minh. Con người sẽ đi trên những đường cao tốc và thiết bị vận chuyển thông minh, nhà máy và cánh đồng thông minh, những mạng lưới năng lượng và tiện ích (cung cấp nước, gas) thông minh. Đây không phải là chuyện ảo tưởng, hay là một khả năng có thể xảy ra. Nó chắc chắn sẽ xảy ra trong thời gian tới, vấn đề là sớm hay muộn.

IoT hiện đang có 2 thách thức lớn nhất:

- Bảo mật: IoT là trung tâm dữ liệu mà tất cả các thiết bị / hệ thống kết nối hoạt động dựa trên dữ liệu có sẵn. Khi nói đến luồng dữ liệu giữa các thiết bị, luôn có khả năng dữ liệu có thể bị truy cập hoặc đọc trong quá trình truyền. Do đó, cần kiểm tra xem liệu dữ liệu có được bảo vệ/mã hóa trong quá trình chuyển từ thiết bị này sang thiết bị khác hay không. Bất cứ nơi nào có giao diện người dùng đều phải có mật khẩu bảo vệ trên đó.
- Tính tương tích: Hiện tại có hàng loạt các công nghệ trên mạng có thể được xác định chính xác là dành cho IoT. Ở cấp độ mạng đã có Bluetooth, Bluetooth LE, ZigBee, RFID, Wi-Fi, Z-Wave, 6LowPAN, NFC, Sigfox, Neul, LoRaWAN, Alljoyn, IoTivity, Weave, Homekit, MQTT, CoAP, JSON-LD và nhiều công nghệ khác. Tất cả các tiêu chuẩn này đều là các tiêu chuẩn kỹ thuật và có những sự chòng chéo nhất định trong chức năng chức năng của chúng. Điều đó có nghĩa là bất kỳ thiết bị cụ thể nào cũng có thể hoạt động với một hoặc một vài hoặc không thiết bị nào khác. Vì vậy khả năng tương tác là một vấn đề lớn cần quan tâm.

## **4.2 Kiến trúc và các ứng dụng IoT**

### **4.2.1 Kiến trúc hệ thống IoT**

Mặc dù không có một kiến trúc IoT được thống nhất trên toàn cầu, nhưng định dạng cơ bản và được chấp nhận rộng rãi nhất là kiến trúc ba lớp. Mô hình được giới thiệu lần đầu tiên ở các nghiên cứu sớm nhất về IoT gồm ba tầng: lớp thiết bị, lớp mạng, và lớp ứng dụng như chỉ ra trong hình 4.1.

#### **Lớp thiết bị:**

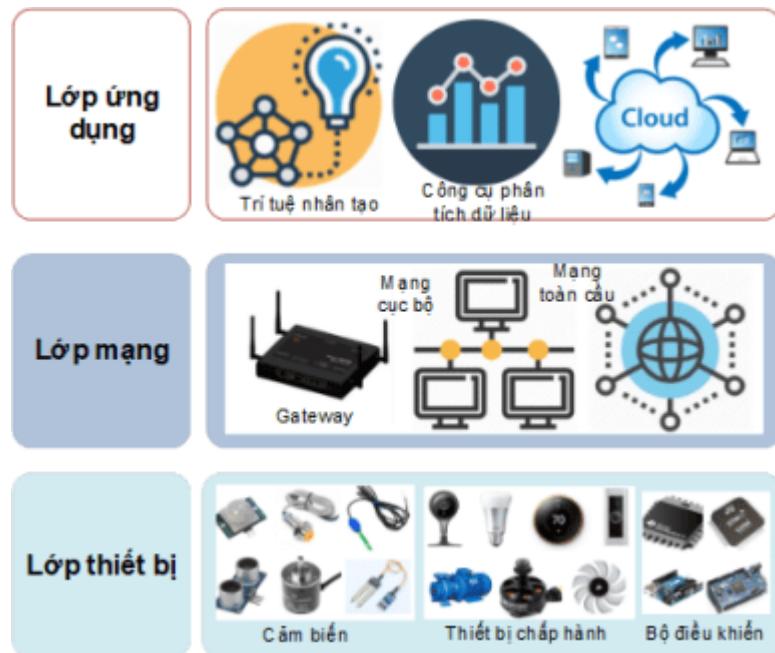
- + Lớp này bao gồm các cảm biến, thiết bị chấp hành và các bộ điều khiển như vi xử lý/vi điều khiển, PLC, FPGA để các máy tính nhúng.
- + Lớp thiết bị thực hiện đo lường và thu thập dữ liệu các đại lượng vật lý thông qua các cảm biến, điều khiển các thiết bị chấp hành và có thể truyền và nhận dữ liệu từ các thiết bị khác qua mạng.

#### **Lớp mạng:**

- + Chức năng lớp mạng xác định các giao thức truyền thông khác nhau được sử dụng cho việc kết nối mạng và thực hiện điện toán biên.
- + Lớp mạng bao gồm các thiết bị liên kết mạng như Hub, Switch, Router; các thiết bị chuyển đổi giao thức mạng như Gateways; đến các thiết bị có khả năng lưu trữ, xử lý cục bộ trước khi gửi dữ liệu lên Server trung tâm.
- + Các “Things” ở lớp thiết bị được kết nối với thiết bị Gateway ở lớp mạng thông qua các mạng cục bộ như WiFi, Zigbee, Bluetooth, LoRaWAN ... đến các mạng có dây như CAN, Modbus, Profibus, RS485, Ethernet,... Sau đó, thiết bị ở lớp mạng thực hiện xử lý và gửi lên trung tâm dữ liệu qua mạng toàn cầu như Internet, 3G/4G/LTE, GSM.

### Lớp ứng dụng:

- + Đây là trung tâm lưu trữ dữ liệu hay đám mây điện tử.
- + Lớp này thực hiện thu nhận dữ liệu từ lớp mạng, lưu trữ, xử lý dữ liệu và ra quyết định dựa trên các thuật toán AI/ML hoặc các công cụ phân tích dữ liệu hiện đại.



Hình 4. 1 Mô hình kiến trúc hệ thống IoT

Một hệ thống IoT sẽ bao gồm 4 thành phần chính là:

- + Thiết bị hay còn được gọi là Things;
- + Trạm kết nối hay là cổng kết nối (Gateways);
- + Hạ tầng mạng hay là các điện toán đám mây (Network and Cloud);
- + Bộ phân tích và xử lý dữ liệu (Services-creation and Solution Layers).

Thông thường cảm biến sẽ có nhiệm vụ chính là cảm nhận các tín hiệu từ môi trường vì dụ như nhiệt độ, áp suất, ánh sáng, chuyển động, tia nhiệt... Sau đó chuyển chúng thành các dạng dữ liệu trong môi trường internet. Tiếp theo thì các tín hiệu sẽ được xử lý và đưa ra các thay đổi theo mong muốn của người tiêu dùng. Quá trình này diễn ra khá nhanh chóng. Hiện nay chúng thường được ứng dụng thông qua các app ở trên điện thoại, máy vi tính, laptop...

Mặc dù mỗi hệ thống IoT đều khác nhau, nhưng nền tảng cho mỗi kiến trúc Internet of Things cũng như luồng quy trình dữ liệu chung của chúng gần như giống nhau bao gồm 4 trạng thái:

### **1. Kết nối các cảm biến, bộ truyền động với vật vạn vật (things):**

Là cơ sở cho mọi hệ thống IoT, các thiết bị được kết nối có trách nhiệm cung cấp dữ liệu. Để thu nhận các thông số vật lý ở thế giới bên ngoài hoặc trong bản thân vật thể, chúng cần các cảm biến được nhúng vào chính thiết bị hoặc được triển khai dưới dạng các đối tượng độc lập để đo và thu thập dữ liệu từ xa. Ví dụ dùng để đóng van khi nước đạt đến một mức nhất định hoặc đơn giản để tắt đèn khi mặt trời mọc.

Một yếu tố không thể thiếu khác của lớp này là các bộ truyền động. Cộng tác chặt chẽ với các cảm biến, chúng có thể chuyển đổi dữ liệu thành hành động. Chẳng hạn một hệ thống tưới cây thông minh có thể dựa trên các thông tin được cung cấp bởi các cảm biến để ra lệnh cho bộ truyền động mở các van nước ở những nơi có độ ẩm đất dưới giá trị cài đặt. Các van được giữ mở cho đến khi các cảm biến báo rằng các giá trị được khôi phục về mặc định.

Điều quan trọng nữa là các đối tượng được kết nối không chỉ cần giao tiếp với hệ thống thu thập dữ liệu trên Internet mà còn cần chia sẻ thông tin và cộng tác với nhau. Việc này khiến các thiết bị tiêu tốn nhiều năng lượng và băng thông, do đó, một kiến trúc mạnh mẽ phải sử dụng các giao thức truyền thông phù hợp với mục đích, an toàn và nhẹ, chẳng hạn như Lightweight M2M mà đã trở thành một giao thức tiêu chuẩn hàng đầu để quản lý các thiết bị nhẹ năng lượng thấp.

## **2. Internet gateways**

Sau khi các cảm biến gửi dữ liệu, các cổng kết nối Internet (Internet Gateways) tổng hợp và chuyển đổi thành dạng kỹ thuật số để có thể xử lý, kết nối với phần còn lại của hệ thống. Hơn nữa, người dùng có thể kiểm soát, lọc và chọn dữ liệu để giảm thiểu khối lượng thông tin cần được chuyển tiếp lên đám mây, giúp tiết kiệm băng thông và giảm thời gian phản hồi.

Một khía cạnh khác mà các cổng hỗ trợ là bảo mật. Bởi vì chúng chịu trách nhiệm quản lý luồng thông tin theo cả hai hướng, do đó có thể ngăn chặn rò rỉ dữ liệu cũng như giảm nguy cơ bị tấn công từ bên ngoài vào các thiết bị IoT.

## **3. Hệ thống biên (edge IT system)**

Mặc dù không phải là một thành phần tất yếu của mọi kiến trúc IoT, nhưng các thiết bị biên có thể mang lại những lợi ích đáng kể, đặc biệt là đối với các dự án IoT quy mô lớn. Vì các hệ thống IoT thu thập một lượng dữ liệu do đó đòi hỏi nhiều băng thông, các hệ thống biên đóng vai trò quan trọng trong việc giảm tải cho cơ sở hạ tầng IT cốt lõi. Chỉ những phần dữ liệu lớn hơn thực sự cần sức mạnh của Data center, Cloud platform mới được chuyển tiếp đến đó, giảm thiểu tối đa tiếp xúc với mạng cục bộ, tăng cường bảo mật, đồng thời giảm tiêu thụ điện năng và băng thông góp phần tận dụng hiệu quả hơn các nguồn lực kinh doanh.

## **4. Data center / Cloud platform**

Dữ liệu cần được lưu trữ để phân tích sâu hơn, đó là lý do tại sao lưu trữ dữ liệu là một giai đoạn quan trọng của kiến trúc IoT. Lưu trữ đám mây là phương pháp ưa thích và

được sử dụng rất nhiều trong triển khai IoT, nguyên nhân chính là do quá trình xử lý chuyên sâu không yêu cầu phản hồi ngay lập tức. Ở đó, các hệ thống máy chủ với cấu hình cao có thể quản lý, phân tích và lưu trữ dữ liệu an toàn hơn, ngoài ra hệ thống cũng hỗ trợ cảm biến có thể được kết nối với các nguồn dữ liệu khác.

#### **4.2.2 Các ứng dụng IoT**

IoT có rất nhiều ứng dụng. Sau đây là 5 ứng dụng nổi bật nhất.

##### **Ứng dụng trong việc xây dựng nhà thông minh**

Nhà thông minh (home automation, domotics, smart home hoặc Intellihome) là kiểu nhà được lắp đặt các thiết bị điện, điện tử có thể được điều khiển hoặc tự động hóa hoặc bán tự động, thay thế con người trong thực hiện một hoặc một số thao tác quản lý, điều khiển. Hệ thống này giao tiếp với người dùng thông qua bảng điện tử đặt trong nhà, ứng dụng trên điện thoại di động, máy tính bảng hoặc giao diện web.

Công nghệ nhà thông minh cung cấp cho chủ nhà sự an toàn, thoải mái, tiện lợi và tiết kiệm năng lượng. Các hệ thống và thiết bị nhà thông minh thường hoạt động cùng nhau, chia sẻ dữ liệu người dùng và tự động hóa các hành động dựa trên quyền ưu tiên của chủ nhà.

Các chức năng chính của nhà thông minh:

- Điều khiển chiếu sáng (on/off, dimmer, scence, timer, logic,...)
- Điều khiển mành, rèm, cửa, cổng
- Hệ thống an ninh, báo động, báo cháy
- Điều khiển điều hòa, máy lạnh
- Hệ thống âm thanh đa vùng
- Camera, chuông hình
- Hệ thống bảo vệ nguồn điện

Nhà thông minh là ứng dụng được tìm kiếm nhiều nhất trên Google. Sự xuất hiện của nó được dự đoán sẽ trở nên phổ biến trong tương lai như điện thoại thông minh hiện nay.

##### **Ứng dụng trong thành phố thông minh/Đô thị thông minh**

Hiện nay, thế giới đang trở nên đô thị hóa hơn bao giờ hết, số lượng người dân tập trung sinh sống và làm việc tại các thành phố lớn ngày càng tăng. Việc này đã ảnh hưởng

đáng kể đến sự phát triển của các thành phố, gây ra tình trạng ô nhiễm môi trường, thiếu thốn các nguồn lực như nước sạch, đất đai, không gian và năng lượng... Để giải quyết các vấn đề nói trên, việc xây dựng thành phố thông minh là hoàn toàn cần thiết.

Đến nay, thế giới vẫn chưa có một định nghĩa thống nhất về Đô thị thông minh. Nhưng về cơ bản, đó là mô hình thành phố ứng dụng công nghệ thông tin, trí tuệ nhân tạo để quản lý, nâng cao tiêu chuẩn cuộc sống đô thị, cải thiện chất lượng phục vụ của chính quyền thành phố và sử dụng hiệu quả các nguồn năng lượng, tài nguyên thiên. Nếu so sánh Đô thị thông minh như một cơ thể người thì Trí tuệ nhân tạo sẽ là bộ não, các Hệ thống cảm biến là các giác quan và Mạng viễn thông số là hệ dây thần kinh. Nói một cách ngắn gọn, Smart City là mô hình thành phố áp dụng công nghệ mới nhất để nâng cao chất lượng thành phố về mọi mặt

Ứng dụng Internet of Thing ở đây chính là việc cài đặt cảm biến (RFID, IR, GPS, máy quét laser, v.v.) cho mọi thứ và kết nối chúng với Internet thông qua các giao thức truyền thông để trao đổi thông tin và truyền thông nhằm truyền đạt được vị trí, theo dõi thông minh, giám sát và quản lý. Những lợi ích tiêu biểu mà IoT đem lại bao gồm:

- Cung cấp nước hiệu quả hơn: IoT sẽ biến đổi cách thức các thành phố tiêu thụ nước. Với sự xuất hiện của những chiếc đồng hồ thông minh có thể ngăn ngừa hiện tượng rò rỉ, thoát nước. Những dữ liệu liên quan sẽ được phân tích nhanh chóng, giảm thời gian khắc phục sự cố. Người dùng có thể thường xuyên tiếp cận thông tin tiêu thụ của mình. Với sự xuất hiện của IoT chắc chắn sẽ làm giảm hiện tượng mất nước nhiều ngày tại các thành phố vào mùa cao điểm như hiện nay, mặt khác chất lượng nước cũng được đảm bảo nhờ các cảm biến kiểm tra chất lượng nước tự động.
- Giải quyết vấn nạn ùn tắc giao thông: Với xu thế phổ biến hiện nay là di dân từ nông thôn ra thành thị, các nhà chức trách không khỏi băn khoăn tìm kiếm giải pháp cho vấn nạn ùn tắc giao thông, ô nhiễm môi trường và cơ sở hạ tầng hiện chưa đủ đáp ứng. Những dịch vụ công hiện nay còn khá ỏi, khó có thể đáp ứng nhu cầu của lượng lớn người dân thành thị. Những giải pháp bức thiết được đặt ra bao gồm quản lý bãi đỗ xe, quản lý giao thông, thiết bị hỗ trợ, tối ưu hóa hiệu suất các bãi đỗ xe, quản lý bến cảng, sân bay. Các giải pháp được áp dụng sẽ làm hạn chế nguy cơ xảy ra tai nạn giao thông, kiểm soát được việc di chuyển và lưu lượng các phương tiện.

- Nâng cao độ tin cậy, chất lượng của giao thông công cộng: Các phương tiện giao thông công cộng có thể bị gián đoạn bất cứ khi nào khi có hiện tượng ùn tắc hay gặp phải sự cố trên đường, IoT sẽ cung cấp cho cơ quan giao thông những thông tin cần thiết để thực hiện các kế hoạch dự phòng, đảm bảo người dân có thể tiếp cận các phương tiện giao thông công cộng an toàn và hiệu quả.

### ***Ứng dụng trong sản xuất công nghiệp***

Việc ứng dụng IoT trong ngành công nghiệp sản xuất được gọi là IIoT (Industrial Internet of Things). IIoT sẽ cách mạng hóa việc sản xuất nhờ việc thu nhận và truy cập vào nguồn dữ liệu không lò với tốc độ lớn hơn và hiệu quả hơn nhiều trước đây. Nhiều công ty tiên phong đã bắt đầu áp dụng IIoT bằng cách sử dụng các thiết bị có kết nối mạng và trí tuệ nhân tạo trong nhà máy.

Như những nhà máy trước đây đã thừa hưởng sự tiến bộ của công nghệ năng lượng hơi nước, năng lượng điện, công nghệ thông tin, và tự động hóa để đạt được một lợi thế cạnh tranh, các nhà máy trong tương lai sẽ thừa hưởng công nghệ Internet để thúc đẩy cuộc cách mạng công nghiệp tiếp theo – Một cuộc cách mạng Internet trong công nghiệp.

IIoT có thể cải thiện đáng kể khả năng kết nối, hiệu quả, tầm ảnh hưởng, tiết kiệm thời gian, chi phí cho các tổ chức. Các công ty nhận được lợi ích từ việc áp dụng IIoT qua cắt giảm chi phí là nhờ các khoản bảo trì có thể xác định trước, mức độ an toàn cao hơn và nhiều tiêu chí hiệu quả về vận hành khác.

Mạng lưới các thiết bị thông minh của IIoT cho phép các tổ chức truy cập nguồn dữ liệu không lò, kết nối mọi người, dữ liệu, quy trình từ các nhà máy tới người quản lý. Những người đứng đầu doanh nghiệp có được cái nhìn chính xác và đầy đủ về công việc của cả công ty, và đưa ra quyết định chính xác hơn. IIoT hứa hẹn sẽ cách mạng hóa việc sản xuất nhờ việc thu nhận và truy cập vào nguồn dữ liệu không lò với tốc độ lớn hơn.

IIoT hứa hẹn sẽ cách mạng hóa việc sản xuất nhờ việc thu nhận và truy cập vào nguồn dữ liệu không lò với tốc độ lớn hơn.

### ***Ứng dụng trong chăm sóc sức khỏe và y tế***

Internet of Things (IoT) mở rộng tiềm năng sử dụng công nghệ để hỗ trợ chăm sóc sức khỏe bằng cách kết nối không chỉ con người, ứng dụng và dữ liệu mà còn cả các cảm biến và thiết bị thu thập dữ liệu. Các hệ sinh thái chăm sóc sức khỏe này có thể bao gồm các thiết bị thông minh có khả năng sử dụng dữ liệu này để thực hiện các hành động, như

cung cấp cảnh báo hoặc gửi thông báo, khi giá trị đo đạt đến một mức nhất định, đưa ra khuyến nghị cho bệnh nhân hoặc chuyên gia chăm sóc sức khỏe dựa trên phân tích dữ liệu và có khả năng cung cấp thuốc dựa trên dữ liệu bệnh nhân được theo dõi và hướng dẫn y tế.

Các hệ thống dựa trên IoT có thể được sử dụng để giải quyết một loạt các vấn đề sức khỏe từ khỏe mạnh đến ốm yếu, thể chất đến tinh thần, chăm sóc, phòng ngừa đến điều trị hoặc phục hồi chức năng. Điện thoại thông minh, đồng hồ và các thiết bị thông minh khác có thể được kết nối với mạng IoT để cung cấp thông tin về bệnh nhân, các hoạt động của bệnh nhân và phạm vi hoặc môi trường của bệnh nhân. Bằng cách sử dụng các công nghệ IoT, hệ thống y tế thông minh có thể trao quyền cho mọi người chủ động tham gia chăm sóc sức khỏe cũng như quản lý sự phục hồi của họ khi có bệnh hoặc bị chấn thương.

### ***Ứng dụng trong nông nghiệp***

Dân số liên tục tăng lên đồng nghĩa với việc nhu cầu sử dụng lương thực cũng tăng. Do đó, nông dân cần áp dụng các kỹ thuật mới, công nghệ tiên tiến để tăng sản lượng sản xuất nông nghiệp. Nông nghiệp thông minh có thể nói là lĩnh vực phát triển nhanh nhất với IoT. Mặc dù IoT nông nghiệp thông minh, cũng như IoT công nghiệp, không phổ biến như các thiết bị kết nối với người tiêu dùng, thị trường vẫn rất năng động. Việc áp dụng các giải pháp IoT cho nông nghiệp không ngừng phát triển. Cụ thể, BI Intelligence dự đoán rằng số lượng cài đặt thiết bị IoT nông nghiệp sẽ đạt 75 triệu vào năm 2020, tăng 20% mỗi năm. Đồng thời, quy mô thị trường nông nghiệp thông minh toàn cầu dự kiến sẽ tăng gấp ba vào năm 2025, đạt 15,3 tỷ đô la (so với mức hơn 5 tỷ đô la một chút vào năm 2016).

IoT có thể được sử dụng trong các trường hợp sau:

- Giám sát điều kiện khí hậu: Có lẽ các thiết bị nông nghiệp thông minh phổ biến nhất là trạm thời tiết, kết hợp các cảm biến khác được đặt trên cánh đồng, chúng thu thập dữ từ môi trường rồi gửi nó lên đám mây. Các phép đo được cung cấp có thể được sử dụng để lập bản đồ các điều kiện khí hậu, chọn các loại cây trồng phù hợp.
- Tự động hóa nhà kính: Nhà kính ban đầu ra đời với mục đích giúp cách ly cây trồng với điều kiện thời tiết bên ngoài. Sau đó, chúng được bổ sung thêm các hệ thống kiểm soát khí hậu bên trong nhà kính, chẳng hạn như nhiệt độ, độ ẩm, ánh sáng,... và hệ thống điều khiển tưới. Hệ thống điều khiển tưới bao gồm các thiết bị như đầu tưới nhỏ giọt hoặc đầu tưới phun sương/mưa, bộ châm phân, bộ điều khiển tưới...

giúp tưới nước/phân một cách tiết kiệm, hiệu quả và đạt năng suất cao. Hệ thống điều khiển khí hậu bao gồm các cảm biến nhiệt độ, độ ẩm bên trong và bên ngoài nhà kính, hệ thống quạt thông gió để đối lưu không khí, hệ thống đèn chiếu sáng để có thể tăng cường ánh sáng khi cần thiết, trạm đo thời tiết để biết các thông số như cường độ bức xạ mặt trời, cảnh báo mưa, tốc độ gió, lưu lượng mưa,... nhằm duy trì nhà kính ở trong điều kiện mong muốn.

- Quản lý cây trồng: Cũng giống như các trạm thời tiết, các thiết bị đo nên được đặt trên cánh đồng để thu thập dữ liệu cụ thể cho canh tác cây trồng, từ nhiệt độ và lượng mưa đến sức khỏe cây trồng, tất cả đều có thể được sử dụng để dễ dàng thu thập dữ liệu và thông tin cho các biện pháp canh tác được cải thiện. Do đó, người nông dân có thể theo dõi sự tăng trưởng của cây trồng và bất kỳ sự bất thường nào để ngăn chặn hiệu quả các bệnh hoặc nhiễm trùng có thể gây hại cho năng suất của họ.
- Giám sát và quản lý gia súc: Cũng giống như giám sát cây trồng, có các cảm biến nông nghiệp IoT có thể được gắn vào động vật trong trang trại để theo dõi sức khỏe và hiệu suất nhật ký của chúng.
- Hệ thống quản lý trang trại đầu cuối: Các hệ thống này thường bao gồm một số thiết bị và cảm biến IoT nông nghiệp, được cài đặt tại cơ sở cũng như bảng điều khiển mạnh mẽ với khả năng phân tích và các tính năng báo cáo / kế toán được xây dựng giúp cung cấp khả năng giám sát trang trại từ xa và cho phép người dùng hợp lý hóa hầu hết các hoạt động kinh doanh.

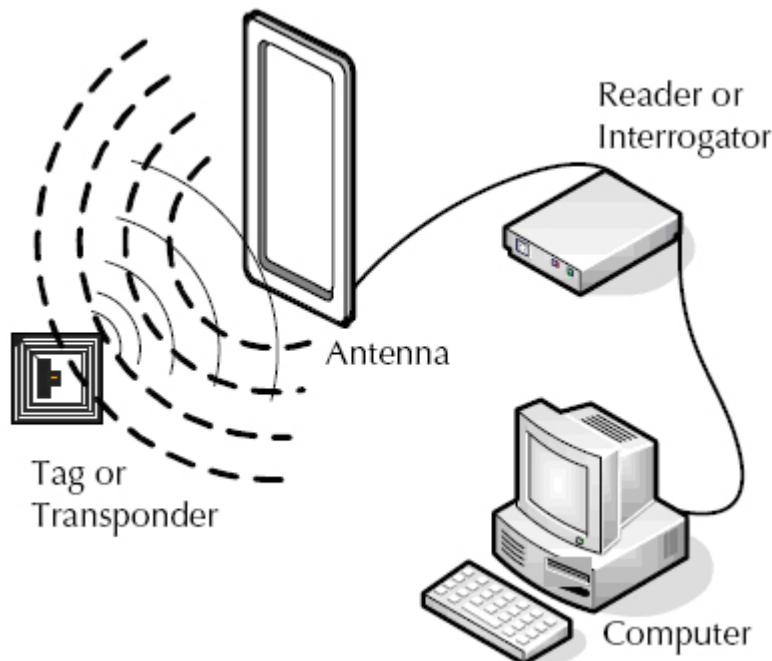
### 4.3 Các giải pháp công nghệ mạng IoT

#### 4.3.1 RFID

RFID (“Radio Frequency Identification” hay “Nhận diện bằng sóng vô tuyến”) là công nghệ nhận dạng đối tượng bằng sóng vô tuyến. Công nghệ này sử dụng sóng vô tuyến để tự động nhận diện và theo dõi vật thể. Vật thể này có thể là bất cứ thứ gì, chẳng hạn như một cuốn sách trong thư viện, một món đồ trong siêu thị, hàng để trong kho, thậm chí là một chiếc ô tô hay động vật.

Kỹ thuật RFID sử dụng truyền thông không dây trong dải tần sóng vô tuyến để truyền dữ liệu từ các tag (thẻ) đến các reader (bộ đọc). Tag có thể được đính kèm hoặc gắn vào đối tượng được nhận dạng chẳng hạn sản phẩm, hộp hoặc (pallet). Reader scan dữ liệu của

tag và gửi thông tin đến cơ sở dữ liệu có lưu trữ dữ liệu của tag. Chẳng hạn, các tag có thể được đặt trên kính chắn gió xe hơi để hệ thống thu phí đường có thể nhanh chóng nhận dạng và thu tiền trên các tuyến đường (hình 4.2). Dạng đơn giản nhất được sử dụng hiện nay là hệ thống RFID bị động làm việc như sau: reader truyền một tín hiệu tần số vô tuyến điện từ qua anten của nó đến một con chip. Reader nhận thông tin trả lại từ chip và gửi nó đến máy tính điều khiển đầu đọc và xử lý thông tin lấy được từ chip. Các chip không tiếp xúc không tích điện, chúng hoạt động bằng cách sử dụng năng lượng nhận từ tín hiệu được gửi bởi reader.



*Hình 4. 2: Ví dụ ứng dụng hệ thống RFID*

### **Thành phần:**

- Tag: là một thẻ gắn chip + antenna, được lập trình điện tử với thông tin duy nhất. Thẻ RFID lưu trữ và truyền dữ liệu đến một đầu đọc trong một môi trường tiếp xúc bằng sóng vô tuyến. Thẻ RFID mang dữ liệu một vật một sản phẩm (item...) nào đó và gắn lên sản phẩm đó. Mỗi thẻ có các phần lưu trữ dữ liệu bên trong và cách giao tiếp với dữ liệu đó. Vài thẻ RFID giống như những nhãn giấy và được ứng dụng để bôi vào hộp và đóng gói. Một số khác được sáp nhập thành vách của thùng chứa plastic được đúc. Còn một số khác được xây dựng thành miếng da bao cổ tay. Mỗi thẻ được lập trình với một nhận dạng duy nhất cho phép theo dõi không dây

đối tượng hoặc con người đang gần thẻ đó. Thông thường mỗi thẻ RFID có một cuộn dây hoặc anten nhưng không phải tất cả RFID đều có vi chip và nguồn năng lượng riêng.

- Reader: là thiết bị kết nối không dây với tag để dễ dàng nhận dạng đối tượng được gắn tag. Nó là một thiết bị đọc và ghi dữ liệu lên tag RFID tương thích. Thời gian mà reader có thể phát năng lượng RF để đọc tag được gọi là chu trình làm việc của reader. Đầu đọc có nhiệm vụ kích hoạt thẻ, truyền dữ liệu bằng sóng vô tuyến với thẻ, thực hiện giải điều chế và giải mã tín hiệu nhận được từ thẻ ra dạng tín hiệu cần thiết để chuyển về máy chủ, đồng thời cũng nhận lệnh từ máy chủ để thực hiện các yêu cầu truy vấn hay đọc ghi thẻ. Đầu đọc thẻ là hệ thám kinh trung ương của toàn bộ hệ thống phần cứng RFID thiết lập việc truyền với thành phần này và điều khiển nó, là thao tác quan trọng nhất của bất kỳ thực thể nào muốn liên kết với thiết bị phần cứng này.
- Antenna thu, phát sóng vô tuyến: Là thiết bị liên kết giữa thẻ và thiết bị đọc. Thiết bị đọc phải xa tín hiệu sóng để kích hoạt và truyền nhận với thẻ.
- Máy chủ và hệ thống phần mềm: Về mặt lý thuyết, một hệ thống RFID có thể hoạt động độc lập không có thành phần này. Thực tế, một hệ thống RFID gần như không có ý nghĩa nếu không có thành phần này.
- Cơ sở tầng truyền thông: Là thành phần bắt buộc, nó là một tập gồm cả hai mạng có dây và không dây và các bộ phận kết nối tuân tự để kết nối các thành phần trong hệ thống RFID với nhau để chúng truyền (giao tiếp) với nhau hiệu quả.

### **Nguyên lý hoạt động:**

Một hệ thống RFID có ba thành phần cơ bản: tag, đầu đọc (reader), và máy chủ.

Tag RFID gắn vào sản phẩm được tích hợp chip bán dẫn và ăng-ten thu sóng. Đầu đọc thẻ nhận tín hiệu từ thẻ RFID từ xa, có thể lên đến 50m tùy vào nguồn năng lượng được cung cấp cho thẻ RFID, chuyển dữ liệu đến máy chủ để phân tích và xử lý thông tin về đối tượng đó.

Vài thẻ RFID giống như những nhãn giấy và được ứng dụng để bồi vào hộp và đóng gói. Một số khác được sáp nhập thành các vách của các thùng chứa plastic được đúc. Còn một số khác được xây dựng thành miếng da bao cổ tay. Mỗi thẻ được lập trình với một nhận dạng duy nhất cho phép theo dõi không dây đối tượng hoặc con người đang gắn thẻ

đó. Bởi vì các chip được sử dụng trong thẻ RFID có thể giữ một số lượng lớn dữ liệu, chúng có thể chứa thông tin như chuỗi số, hướng dẫn cấu hình, dữ liệu kỹ thuật, sổ sách y học, và lịch trình. Cũng như phát sóng tivi hay radio, hệ thống RFID cũng sử dụng bốn băng thông tần số chính: tần số thấp (LF), tần số cao (HF), siêu cao tần (UHF) hoặc sóng cực ngắn (viba). Các hệ thống trong siêu thị ngày nay hoạt động ở băng thông UHF, trong khi các hệ thống RFID cũ sử dụng băng thông LF và HF.

Có 3 loại tag RFID:

- Active tag (chủ động): có nguồn riêng bên trong, truyền dữ liệu bằng nguồn đó
- Passive tag (thụ động): không có nguồn bên trong, sử dụng nguồn nhận được từ reader để hoạt động và truyền dữ liệu được lưu trữ trong nó cho reader
- Semi-passive tag (bán thụ động): có nguồn riêng bên trong, tuy nhiên nó sử dụng nguồn từ reader để truyền dữ liệu, truyền được khoảng cách xa hơn so với tag thụ động

### Ứng dụng của RFID trong IoT

Trong IoT, RFID được sử dụng cùng với camera, GPS, cảm biến thông minh để định vị và xác định các đối tượng.

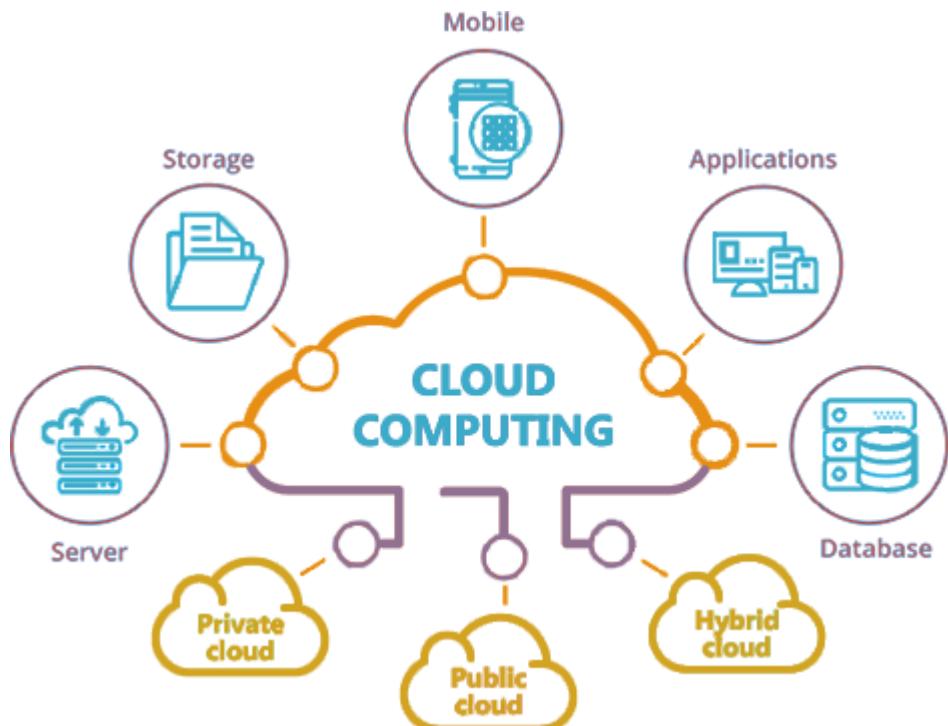
- Nhà thông minh: RFID là một cách tiết kiệm để làm cho những vật dụng trong nhà trở nên “thông minh” hơn. Việc gắn chip RFID trên các vật thể thông thường không tiêu tốn thời gian hay tiền bạc. Với một máy giặt thông minh có thể đọc nhãn RFID, người dùng có thể kiểm soát chu trình giặt và sấy khô cho chiếc áo đó. Một trong những ứng dụng thương mại của RFID là Nest Tag, một chiếc móc khóa giá \$25 (580.000VND) có thể kích hoạt hệ thống báo động Nest của Google.
- Chăm sóc sức khỏe: Bệnh nhân cần được chăm sóc đặc biệt tại bệnh viện hiện sẽ được cấy chip RFID. Dữ liệu sức khỏe của họ được tự động ghi lại và cập nhật trong hệ thống EHR. Tuy nhiên, việc này cũng dấy lên những lo ngại về sự xâm phạm quyền riêng tư của cơ thể.
- Giao thông thông minh: Những đoàn xe thông minh ở các thành phố khác nhau trên thế giới sử dụng tag RFID để kiểm soát giao thông. Tag RFID có thể đọc dữ liệu hành khách cũng như giữ các cảnh báo về hệ thống giao thông để cập nhật. Tag RFID có một số vấn đề trong ứng dụng thực tế. Ví dụ, chúng dễ bị nhiễu và có thể gây ra xung đột giữa các đối tượng. Hơn nữa, mạng RFID hiện dễ bị ảnh hưởng

bởi những vấn đề bảo mật IoT. Đã có một số trường hợp tấn công DDoS và nghe lén được ghi nhận.

Phải nói rằng, phạm vi lớn và chi phí thấp của các hệ thống RFID đã khiến chúng trở thành một đề xuất lý tưởng cho cả người tiêu dùng và IIoT.

#### 4.3.2 Cloud Computing

Cloud Computing (Điện toán đám mây) là mô hình cung cấp các tài nguyên máy tính cho người dùng thông qua Internet. Nguồn tài nguyên này bao gồm rất nhiều thứ liên quan đến điện toán và máy tính. Ví dụ như: phần mềm, dịch vụ, phần cứng,... và sẽ nằm tại các máy chủ ảo (đám mây) trên mạng. Người dùng có thể truy cập vào bất cứ tài nguyên nào trên đám mây. Vào bất kỳ thời điểm nào và ở bất kỳ đâu, chỉ cần kết nối với hệ thống internet.



Hình 4. 3: Điện toán đám mây

Hiện nay, có 4 mô hình triển khai điện toán đám mây chính đang được sử dụng phổ biến. Đó là: Public Cloud, Private Cloud, Hybrid Cloud và Community Cloud.

**Public Cloud** là mô hình triển khai **điện toán đám mây** sử dụng phổ biến nhất hiện nay. Các dịch vụ, ứng dụng trên **Public Cloud** đều nằm trên cùng một **hệ thống Cloud**.

Tức là tất cả người dùng sẽ dùng chung tài nguyên. Nhà cung cấp dịch vụ sẽ trực tiếp quản lý và bảo vệ dữ liệu trên đám mây.

- **Ưu điểm:**
  - + Phục vụ được nhiều đối tượng người dùng, không bị giới hạn về không gian, thời gian.
  - + Đặc biệt **Public Cloud** có chi phí đầu tư thấp. Tiết kiệm được hệ thống máy chủ, giảm gánh nặng quản lý, cơ sở hạ tầng.
  - + Đám mây công cộng còn có thể co giãn theo nhu cầu thực tế của người sử dụng.
- **Nhược điểm:**
  - + Mất an toàn và khó kiểm soát dữ liệu

**Private Cloud** là dịch vụ điện toán đám mây riêng thường được cung cấp cho các doanh nghiệp để đảm bảo an toàn dữ liệu. **Private cloud** sẽ được bảo vệ bên trong tường lửa của công ty và doanh nghiệp trực tiếp quản lý.

- **Ưu điểm:**
  - + Chủ động hơn trong việc sử dụng và quản lý dữ liệu
  - + Bảo mật thông tin tốt hơn
- **Nhược điểm:**
  - + Gặp khó khăn trong việc triển khai công nghệ
  - + Tốn chi phí để xây dựng, duy trì hệ thống
  - + Chỉ phục vụ trong nội bộ doanh nghiệp. Những người dùng khác bên ngoài không thể tiếp cận và sử dụng.

**Đám mây lai (Hybrid Cloud)** là sự kết hợp giữa đám mây công cộng và đám mây riêng. Nó cho phép người dùng khai thác được điểm mạnh của 2 mô hình trên. Và đồng thời hạn chế được điểm yếu của 2 mô hình đó. Đám mây lai thường sẽ do doanh nghiệp tạo ra và việc quản lý thông tin. Dữ liệu sẽ được phân chia giữa doanh nghiệp và nhà cung cấp **Public Cloud**.

- **Ưu điểm:**
  - + Đảm bảo được an toàn cho các dữ liệu quan trọng
  - + Sử dụng được nhiều dịch vụ **điện toán đám mây** mà không bị giới hạn

- Nhược điểm:
  - + Khó khăn khi triển khai và quản lý hệ thống
  - + Tốn nhiều chi phí để xây dựng cơ sở hạ tầng

**Đám mây cộng đồng (Community Cloud)** được xây dựng nhằm mục đích chia sẻ hạ tầng, dữ liệu cho nhiều tổ chức, người dùng khác nhau. Ví dụ, các doanh nghiệp cùng hoạt động trong ngành giáo dục có thể chia sẻ chung một đám mây để trao đổi dữ liệu cho nhau.

- Ưu điểm:
  - + Các tổ chức/doanh nghiệp/cá nhân chung lĩnh vực hoạt động có thể chia sẻ dữ liệu, thông tin dễ dàng để phục vụ cho công việc của chính họ.
  - + Đảm bảo sự riêng tư, an ninh và tuân thủ các chính sách tốt hơn.
- Nhược điểm:
  - + Việc điều hành, quản lý tương đối khó khăn.
  - + Cần tốn nhiều chi phí để xây dựng, triển khai.

Hiện nay, có 3 mô hình cung cấp điện toán đám mây cơ bản:

- Iaas (Infrasructure as a service) là mô hình dịch vụ pay-per-use (tức là trả tiền cho những gì sử dụng). Chi phí sử dụng dịch vụ này được tính dựa trên chức năng và lượng tài nguyên mà khách hàng dùng. Theo Amazon thì đây là mức độ cơ bản nhất của điện toán đám mây. Nhà cung cấp dịch vụ Iaas sẽ bán cho khách hàng các server (máy chủ), thiết bị mạng, bộ nhớ, CPU, storage (không gian lưu trữ), máy tính (có thể máy thật hoặc máy ảo, tùy nhu cầu), trang thiết bị trung tâm dữ liệu và một số tính năng bảo vệ an ninh nâng cao. Với hạ tầng mà Iaas tạo ra, bạn cần vào đó và thiết lập. Và cài thêm những phần mềm cần thiết khác như web server, database,... Iaas không được tạo ra để phục vụ cho người dùng cuối. Mà nó để cho các công ty, đơn vị phát hành web sử dụng với mục đích triển khai phần mềm.
- Paas (Platform as a service) là mô hình dịch vụ giúp các developer có thể phát triển. Nó cho phép triển khai các ứng dụng, website trên đám mây. Paas về cơ bản cũng khá giống với Iaas nhưng cấp độ cao hơn một chút. Paas được trang bị thêm các công cụ phát triển doanh nghiệp thông minh (BI), middleware và nhiều tool khác.

Với Paas, bạn sẽ có một nền tảng (Platform) được cài đặt sẵn để phù hợp cho việc phát triển ứng dụng.

- Saas (Software as a service) là một mô hình dịch vụ điện toán đám mây cao nhất hiện nay. Cho phép người dùng sử dụng được các ứng dụng dễ dàng trên nền tảng đám mây thông qua internet. Đơn giản hơn, Saas sẽ cung cấp phần mềm/ứng dụng chạy trên internet. Từ đó người dùng cuối (end-user) có thể sử dụng ngay. Nhà cung cấp dịch vụ Saas có thể lưu trữ trên server của họ. Hoặc cho phép người dùng tải xuống và vô hiệu hóa nó khi hết hạn. Ví dụ điển hình cho mô hình dịch vụ này là Microsoft Office 365. Đôi khi các web email (Gmail, Outlook, Yahoo Mail,...) cũng dùng dịch vụ này. Đây đều là các sản phẩm hoàn chỉnh. Người dùng có thể sử dụng ngay lập tức mà không cần phải thiết lập server để quản lý. Tương tự, OneDrive, Dropbox cũng là mô hình điện toán đám mây kiểu Saas. Các trang web (phần mềm) này cung cấp không gian lưu trữ cần thiết để bạn có thể upload/download dữ liệu thông qua internet.

**Điện toán đám mây** đem lại rất nhiều lợi ích cho người dùng, cụ thể như sau:

- **Tiết kiệm chi phí:** Giúp giảm thiểu chi phí. Bạn sẽ không tốn tiền đầu tư cơ sở hạ tầng ban đầu. Ví dụ như: mua phần cứng, phần mềm, lắp đặt hệ thống,...
- **Tiện lợi:** Người dùng có thể nhanh chóng truy cập, sử dụng tài nguyên thông qua internet mà không cần cài đặt phức tạp
- **An toàn và liên tục:** Mọi dữ liệu được đồng bộ hóa trên đám mây. Giúp đảm bảo độ an toàn cao hơn, tránh trường hợp mất dữ liệu do hư hỏng ổ cứng. Ngoài ra, nhà cung cấp sẽ sao lưu định kỳ và có các phương thức bảo mật để bảo vệ dữ liệu tốt hơn.
- **Triển khai nhanh chóng ở bất kỳ đâu:** Chỉ với một vài thao tác đơn giản để triển khai chúng mọi nơi. Điều này đồng nghĩa với việc người dùng sẽ có được trải nghiệm tốt hơn với độ trễ thấp hơn.

## Ứng dụng của Cloud Computing trong IoT

Vì những công việc như lưu trữ và xử lý dữ hay được thực hiện ở Cloud nhiều hơn ở thiết bị, điều này có liên quan mật thiết đến IoT. Nhiều hệ thống IoT sử dụng lượng lớn cảm biến để thu thập dữ liệu rồi đưa ra những quyết định thông minh.

Cloud quan trọng cho việc gộp dữ liệu và vẽ ra insight từ dữ liệu đó. Ví dụ, một công ty nông nghiệp thông minh sẽ có thể so sánh những cảm biến độ ẩm đất từ Kansas và Colorado sau khi trồng cùng một loại hạt giống. Nếu thiếu Cloud, việc so sánh dữ liệu trong khu vực rộng hơn là khó khăn hơn rất nhiều.

Cloud có khả năng mở rộng lớn. Khi người dùng có hàng trăm nghìn, thậm chí hàng triệu cảm biến, việc cung cấp khả năng tính toán cho mỗi cảm biến sẽ tiêu tốn rất nhiều chi phí và năng lượng. Thay vào đó, dữ liệu từ cảm biến có thể được đưa đến Cloud và xử lý ở đó khi được gộp lại.

Đối với IoT, thường thì đầu não của hệ thống là ở Cloud. Cảm biến và thiết bị thu thập dữ liệu và thực hiện các hành động, nhưng việc xử lý/chỉ huy/phân tích (những thứ thông minh) thường diễn ra ở Cloud.

#### 4.3.3 ZigBee

Hiện nay, rất nhiều công nghệ không dây (wireless) có thể truyền dữ liệu với tốc độ cao giữa các thiết bị với nhau như BlueTooth hay Wi-Fi. Nhưng đối với những mạng quản lý các sensor trong các ứng dụng điều khiển - tự động hóa của các thiết bị trong nhà hay bệnh viện thì Wi-Fi hay BlueTooth lại không thể đáp ứng được vì chúng có nhiều khuyết điểm như sử dụng băng thông rộng làm tiêu hao nhiều điện năng không cần thiết, sử dụng các nguồn điện trực tiếp, ít sử dụng pin, phạm vi kết nối nhỏ hẹp, độ trễ cao, cơ chế bảo mật đơn giản (BlueTooth), yêu cầu về các thiết bị phần cứng cao, chi phí lớn.

Để giải quyết những khuyết điểm đó, Zigbee đã được ra đời. Đối tượng mà Zigbee hướng đến là mạng điều khiển dành cho nhà thông minh (SmartHome), tự động hóa các hoạt động theo dõi, tiếp nhận và xử lý thông tin trong lĩnh vực y tế (Health Care), quản lý năng lượng một cách hiệu quả (Smart Energy),... Khi được sử dụng trong các hệ thống này, Zigbee được phát huy tất cả những điểm mạnh của nó như độ trễ truyền tin thấp, tiêu hao ít năng lượng, giá thành thấp, ít lỗi, dễ mở rộng và thời gian sử dụng pin dài (1 cặp pin AA có thể hoạt động trong vòng 2 năm).

Zigbee là một giao thức được xây dựng theo chuẩn IEEE 802.15.4. Giao thức này được tạo ra nhằm phục vụ cho những ứng dụng yêu cầu giá thành và công suất thấp nhưng phải có khả năng linh động trong phạm vi rộng. Chuẩn Zigbee được phát triển và xúc tiến bởi hãng Zigbee Alliance, với sự hỗ trợ từ hơn 200 công ty trên thế giới như: SIEMENS, ATMEL, NI, NEC, TEXAS INSTRUMENTS, EPSON....

Về bản chất Zigbee cũng một chuẩn giao tiếp không dây như những chuẩn không dây khác: UWB, Wi-Fi, IrDA, 3G, Bluetooth...nhưng nó mang những đặc tính kỹ thuật và đặc tính vật lý riêng và do đó sẽ chỉ phù hợp với một mảng ứng dụng nhất định.

### ***Ưu điểm:***

- Giá thành thấp
- Tiêu thụ công suất nhỏ
- Kiến trúc mạng linh hoạt
- Được hỗ trợ bởi nhiều công ty
- Số lượng các nút lớn (65k)

### ***Nhược điểm:***

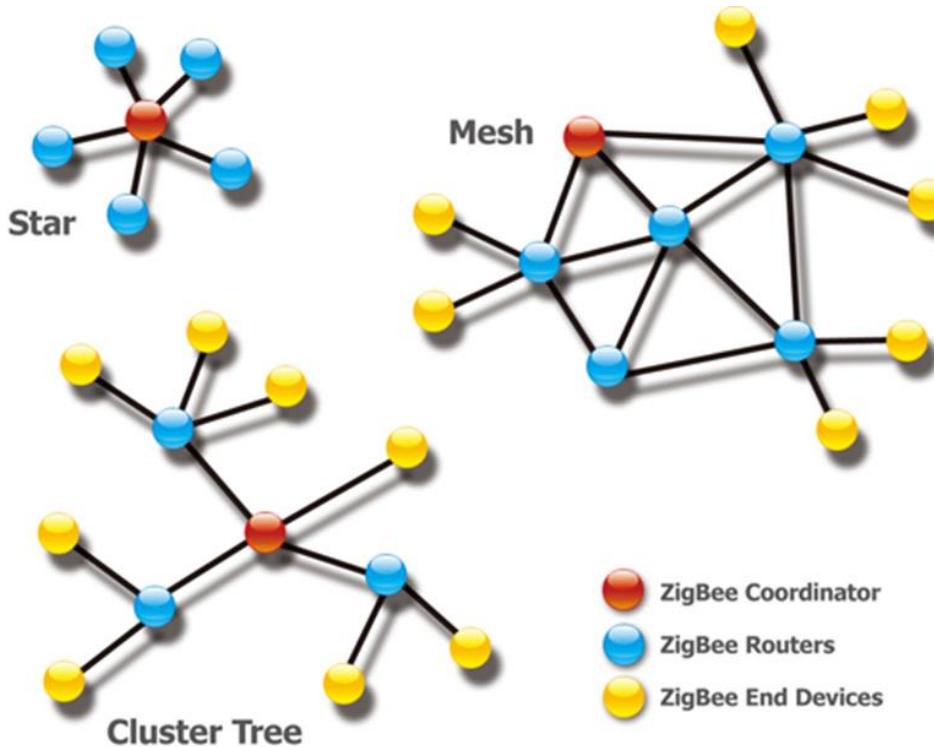
- Lỗi ở một điểm chính có thể gây lỗi hệ thống
- Tốc độ truyền thấp
- Chưa có đầy đủ các thiết bị để phát triển

Bảng 4.1 So sánh Zigbee với các chuẩn không dây khác

Chuẩn	Zigbee	Wifi	Bluetooth	GSM/GPGS/CDMA
Chuẩn IEEE	802.15.4	802.11a,b,g	802.15.1	-
Ứng dụng	Truyền thông không dây vi điều khiển	Web, video, email..	Thay thế cắp nối	Dữ liệu trên phạm vi rộng
Tài nguyên hệ thống	4-32KB	1MB+	250KB+	16MB+
Tuổi thọ pin/ngày)	100-1000+	3-5	1-7	1-7
Tốc độ (KB/s)	20-250	11000s	720	64-128+
Phạm vi truyền (m)	1-100+	1-100	1-10+	1000+
Đặc điểm nổi bật	Đáng tin cậy, tiết kiệm năng lượng, giá thành rẻ	Tốc độ cao, có tính mềm dẻo	Giá rẻ, thuận tiện	Phạm vi rộng, chất lượng tốt

ZigBee có 3 dạng hình mạng được hỗ trợ: dạng hình sao, hình lưới, và hình cây. Mỗi dạng hình đều có những ưu điểm riêng và được ứng dụng trong các trường hợp khác nhau.

- Hình sao (Star): Các nút hình sao liên kết với nút trung tâm.
- Hình lưới (Mesh): Mạng hình lưới có tính tin cậy cao, mỗi nút trong mạng lưới đều có khả năng kết nối với nút khác, nó cho phép truyền thông liên tục giữa các điểm nút với nhau và bền vững. Nếu có sự tác động cản trở, hệ thống có khả năng tự xác định lại cấu hình bằng cách nhảy từ nút này sang nút khác.
- Hình cây (Cluster Tree): là 1 dạng đặc biệt của mạng hình lưới, dạng mạng này có khả năng phủ sóng và mở rộng cao.



Hình 4. 4: Cấu trúc mạng Zigbee

Các thành phần chính trong mạng Zigbee:

- Thiết bị điều phối trung tâm – Zigbee Coordinator (ZC)
  - + Chọn một kênh giao tiếp và một PAN ID để tạo ra mạng
  - + Cho phép router và end device tham gia mạng này
  - + Định tuyến đường truyền trong mạng

- + Luôn hoạt động, không được phép ở trong chế độ SLEEP
- + Bảo quản dữ liệu cho end device đã ở chế độ SLEEP cho tới khi chúng thức và gọi dữ liệu ra
- Thiết bị Router – Zigbee Router (ZR)
  - + Phải tham gia mạng trước khi truyền dữ liệu
  - + Cho phép các router khác và end device tham gia vào mạng, sau khi nó tham gia một mạng
  - + Định tuyến đường truyền trong khi tham gia mạng
  - + Luôn hoạt động, không được phép ở trong chế độ SLEEP
  - + Bảo quản dữ liệu cho end device đã ở chế độ SLEEP cho tới khi chúng thức và gọi dữ liệu ra
- Thiết bị End Device – Zigbee EndDevice (ZED)
  - + Phải tham gia mạng trước khi truyền dữ liệu
  - + Không cho phép các thiết bị khác tham gia mạng
  - + Truyền và nhận thông qua node cha, không thể định tuyến trong mạng
  - + Hỗ trợ dùng pin và chế độ SLEEP

Quá trình thiết lập trong một mạng Zigbee như sau:

- Quét mạng (Network Scan): Các thiết bị trong mạng sẽ quét các kênh tín hiệu, ví dụ nếu dùng dải tần 2,4GHz thì sẽ có 16 kênh để quét, sau đó thiết bị sẽ chọn kênh phù hợp nhất để giao tiếp trong mạng. Ta gọi đó là sự chiếm chỗ: occupancy.
- Thiết lập/Gia nhập mạng: Thiết bị có thể tạo ra một mạng trên một kênh hoặc gia nhập vào một mạng đã tồn tại sẵn.
- Phát hiện thiết bị: Thiết bị sẽ yêu cầu mạng phát hiện ra địa chỉ của mình trên các kênh được kích hoạt.
- Phát hiện dịch vụ: Thiết bị quét các dịch vụ được hỗ trợ trên thiết bị trong phạm vi mạng.
- Liên kết: Thiết bị giao tiếp với nhau thông qua các lệnh và các tin nhắn điều khiển.

### **Ứng dụng của ZigBee trong IoT**

Một số ứng dụng của ZigBee trong IoT:

- Tự động hóa nhà: Công nghệ ZigBee chứng tỏ là công nghệ đáng tin cậy nhất trong việc hiện thực hóa tự động hóa nhà (Smarthome). Các ứng dụng khác nhau như kiểm soát và giám sát mức tiêu thụ năng lượng, quản lý nước, kiểm soát ánh sáng, vv đã được thực hiện dễ dàng hơn thông qua tự động hóa bằng công nghệ ZigBee.
- Tự động hóa công nghiệp: Các thiết bị RFID dựa trên ZigBee giúp cung cấp quản lý truy cập đáng tin cậy trong các ngành công nghiệp. Các ứng dụng khác trong các ngành công nghiệp bao gồm kiểm soát quá trình, quản lý năng lượng, theo dõi nhân sự, v.v.
- Tự động hóa chăm sóc sức khỏe: Một ví dụ phổ biến của tự động hóa chăm sóc sức khỏe là theo dõi sức khỏe từ xa. Một người đeo thiết bị ZigBee với cảm biến đo thông số cơ thể thu thập thông tin sức khỏe. Thông tin này được truyền trên mạng ZigBee đến mạng Giao thức Internet (IP) và sau đó đến nhân viên chăm sóc sức khỏe (bác sĩ hoặc y tá), người sau đó sẽ kê đơn thuốc phù hợp dựa trên thông tin nhận được.

#### **4.4. Các giao thức Request/ Response**

##### **4.4.1 Giao thức HTTP trong IoT**

HTTP là một giao thức yêu cầu/phản hồi không trạng thái, nơi các Client yêu cầu thông tin từ Server và Server sẽ phản hồi các yêu cầu này theo đó (mỗi yêu cầu độc lập với yêu cầu khác), đã được trình bày chi tiết trong chương 2.

Giao thức HTTP được sử dụng cho dịch vụ Web để truyền dữ liệu dưới dạng văn bản, âm thanh, hình ảnh và video từ Server đến trình duyệt web của người dùng và ngược lại. HTTP hiện là nền tảng truyền dữ liệu của ứng dụng duyệt web ngày nay và được sử dụng rộng rãi trong hệ thống IoT. Mặc dù giao thức HTTP có nhiều nhược điểm trong việc truyền dữ liệu và không phù hợp bằng các giao thức tối ưu khác như MQTT, CoAP, AMQP sử dụng cho IoT, nhưng giao thức này vẫn phổ biến trong lĩnh vực nhà thông minh cũng như việc sử dụng nhiều trong bộ vi điều khiển và vi xử lý tiên tiến. Ví dụ, các hệ thống dựa trên thế hệ Raspberry Pi mới sử dụng giao thức HTTP để truyền dữ liệu. Arduino cũng sử dụng giao thức này để giao tiếp với các thiết bị khác. Bên cạnh đó, có thêm 3 phiên bản mới của

HTTP ra đời nhằm bù đắp những nhược điểm của phiên bản cũ và được áp dụng cho những trường hợp người điều khiển mong muốn.

### **Ưu và nhược điểm của giao thức HTTP trong IoT:**

#### ***Ưu điểm***

- + **Khả năng tìm kiếm:** Mặc dù HTTP là một giao thức nhắn tin đơn giản, nó bao gồm khả năng tìm kiếm cơ sở dữ liệu với một yêu cầu duy nhất. Điều này cho phép giao thức được sử dụng để thực hiện các tìm kiếm SQL và trả về kết quả được định dạng thuận tiện trong tài liệu HTML;
- + **Dễ lập trình:** HTTP được mã hóa dưới dạng văn bản thuận túy và do đó dễ theo dõi và triển khai hơn các giao thức sử dụng mã yêu cầu tra cứu. Dữ liệu được định dạng dưới dạng dòng văn bản chứ không phải dưới dạng chuỗi biến hoặc trường;
- + **Bảo mật:** HTTP 1.0 tải xuống từng tệp qua một kết nối độc lập và sau đó đóng kết nối. Vì vậy, điều này làm giảm đáng kể nguy cơ bị đánh chặn trong quá trình truyền.

#### ***Nhược điểm***

- + **Không phù hợp với các thiết bị nhỏ:** Vì các thiết bị nhỏ, chẳng hạn như cảm biến, không yêu cầu nhiều tương tác và chúng tiêu thụ rất ít năng lượng, HTTP quá nặng nên không phù hợp cho các thiết bị này. Một yêu cầu HTTP yêu cầu tối thiểu chín gói TCP, thậm chí nhiều hơn khi bạn xem xét mất gói do kết nối kém và tiêu đề văn bản thuận túy có thể rất dài dòng;
- + **Không được thiết kế cho giao tiếp dựa trên sự kiện:** Hầu hết các ứng dụng IoT đều dựa trên sự kiện. Các thiết bị cảm biến đo một số biến như nhiệt độ, chất lượng không khí và có thể cần đưa ra các quyết định theo hướng sự kiện như tắt công tắc. HTTP được thiết kế cho giao tiếp dựa trên phản hồi yêu cầu thay vì giao tiếp theo hướng sự kiện. Ngoài ra, việc lập trình các hệ thống dựa trên sự kiện này sử dụng giao thức HTTP trở thành một thách thức lớn, đặc biệt là do tài nguyên máy tính trên các thiết bị cảm biến có hạn;

- + Vấn đề thời gian thực: Sau khi yêu cầu một tài nguyên đến máy chủ, Client phải đợi Server phản hồi, dẫn đến việc truyền dữ liệu chậm. Cảm biến IoT là các thiết bị nhỏ với tài nguyên máy tính rất hạn chế và do đó không thể hoạt động đồng bộ một cách hiệu quả. Tất cả các giao thức IoT được sử dụng rộng rãi đều dựa trên mô hình không đồng bộ.

#### **4.4.2 Giao thức CoAP**

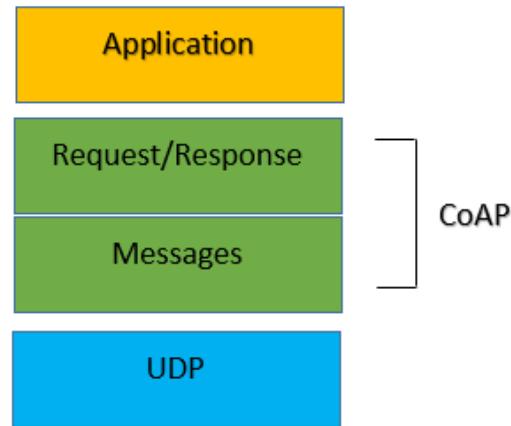
CoAP (Constrained Application Protocol) là một giao thức đơn giản chi phí thấp được thiết kế riêng cho các thiết bị hiệu năng thấp (chẳng hạn như vi điều khiển) và nơi mạng có băng thông thấp. Giao thức này được sử dụng để trao đổi dữ liệu M2M và rất giống với HTTP.

**CoAP có các tính năng chính sau:**

- + Giao thức web nhỏ gọn được sử dụng trong M2M;
- + Bảo mật bằng DTLS;
- + Trao đổi thông điệp không đồng bộ;
- + Header gói tin nhỏ, dễ tách thông tin;
- + Hỗ trợ URI và loại nội dung;
- + Khả năng proxy và bộ nhớ đệm;
- + Tuỳ chọn khai thác tài nguyên;
- + Liên kết UDP với độ tin cậy tùy chọn hỗ trợ các yêu cầu Unicast và Multicast.

#### **Mô hình cấu trúc CoAP**

Mô hình tương tác CoAP tương tự như mô hình Client/Server của HTTP. CoAP sử dụng cấu trúc 2 lớp. Lớp dưới là lớp bản tin được thiết kế liên quan đến UDP và chuyển tiếp không đồng bộ, Lớp yêu cầu/phản hồi liên quan đến phương thức giao tiếp và xử lý bản tin yêu cầu/phản hồi.

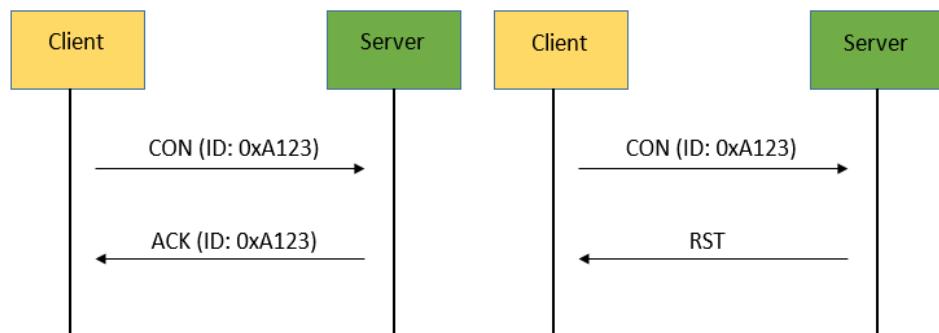


Hình 4. 5: Các lớp của CoAP

### Mô hình bản tin CoAP

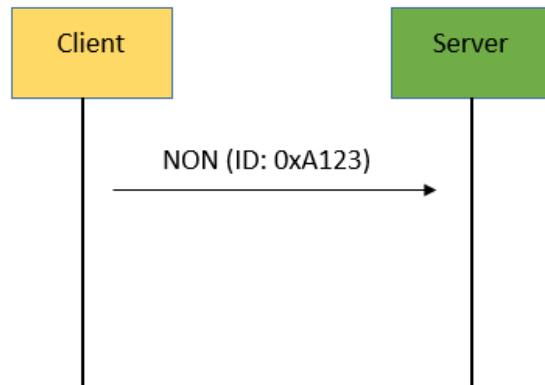
Lớp bản tin hỗ trợ 4 loại bản tin: CON (có thẻ xác nhận), NON (không thẻ xác nhận), ACK (đã xác nhận), RST (đặt lại).

- + Truyền tải bản tin tin cậy: Một bản tin có thẻ xác nhận (CON) được truyền đi truyền lại cho đến khi Server gửi lại bản tin xác nhận (ACK) với cùng một ID. Sử dụng thời gian chờ mặc định và giảm thời gian đếm theo cấp số nhân khi truyền bản tin CON. Nếu Server không thể xử lý bản tin truyền đến, nó sẽ phản hồi bằng cách thay thế bản tin xác nhận (ACK) bằng bản tin đặt lại (RST).



Hình 4. 6: Truyền bản tin đáng tin cậy

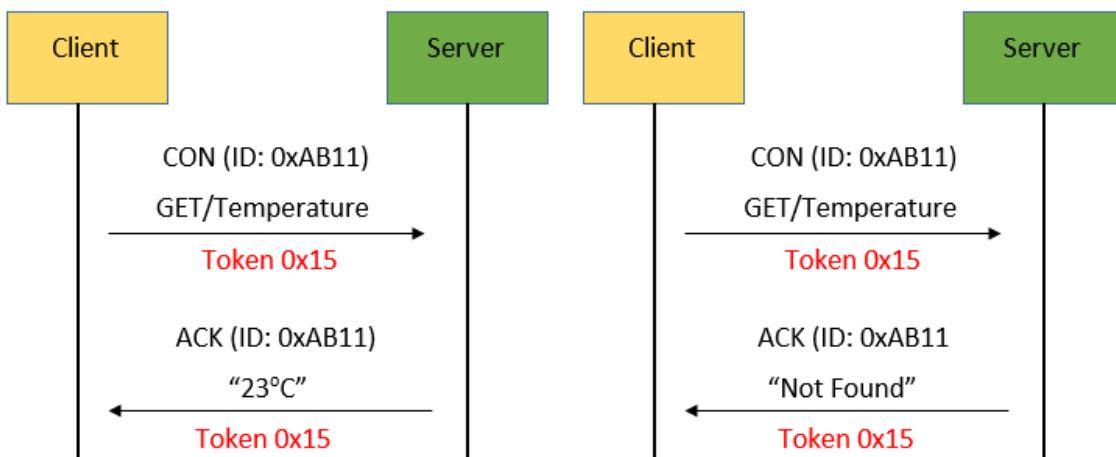
Truyền tải bản tin không tin cậy: Một bản tin không yêu cầu gửi tin cậy, có thể được gửi bằng bản tin không tin cậy. Nó sẽ không được xác nhận, nhưng nó vẫn có ID để phát hiện trùng lặp.



Hình 4. 7: Truyền bản tin không đáng tin cậy

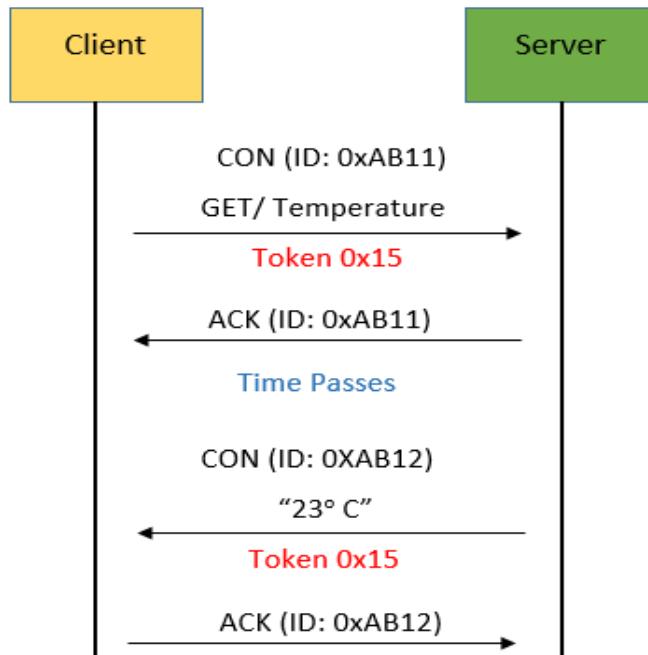
### Mô hình yêu cầu/phản hồi CoAP

Piggy-backed: Một yêu cầu được gửi bằng bản tin có thẻ xác nhận (CON) hoặc không thẻ xác nhận (NON), Server sẽ phản hồi bản tin xác nhận (ACK) ngay lập tức cho yêu cầu của Client nếu nó có sẵn. Với phản hồi thành công, ACK chứa mã bản tin phản hồi (được nhận diện bởi mã bản tin), với phản hồi thất bại, ACK chứa mã phản hồi thất bại.



Hình 4. 8: Hai yêu cầu GET có phản hồi ngay lập tức, một thành công, một không tìm thấy

Phản hồi trì hoãn: Nếu Server nhận được bản tin CON nhưng không thể phản hồi yêu cầu này ngay lập tức, nó sẽ gửi một bản tin ACK trống tránh trường hợp Client gửi lại bản tin này. Khi Server sẵn sàng đáp ứng yêu cầu này, nó sẽ gửi một bản tin CON mới đến Client và Client trả lời một bản tin CON kèm theo xác nhận. Bản tin ACK từ Client chỉ để xác nhận bản tin CON từ Server.



Hình 4. 9: Yêu cầu GET với phản hồi trì hoãn

## Ưu điểm và nhược điểm của CoAP

### Ưu điểm

- + Đây là giao thức đơn giản và header nhỏ gọn hơn do hoạt động qua UDP. Nó cho phép thời gian wake-up ngắn và trạng thái sleep dài. Điều này giúp đạt được tuổi thọ pin dài để sử dụng;
- + Nó sử dụng IPSEC (IP Security) hoặc DTLS (Datagram Transport Layer Security) để cung cấp giao tiếp an toàn;
- + Giao tiếp đồng bộ không cần thiết trong giao thức CoAP;

- + Nó có độ trễ thấp hơn so với HTTP;
- + Nó tránh được việc truyền lại không cần thiết, nên nó tiêu thụ năng lượng ít hơn so với HTTP;
- + Giao thức CoAP được sử dụng như một lựa chọn giao thức tốt nhất cho các mạng trong các thiết bị thông tin, thiết bị truyền thông và thiết bị điều khiển trong mạng nhà thông minh.

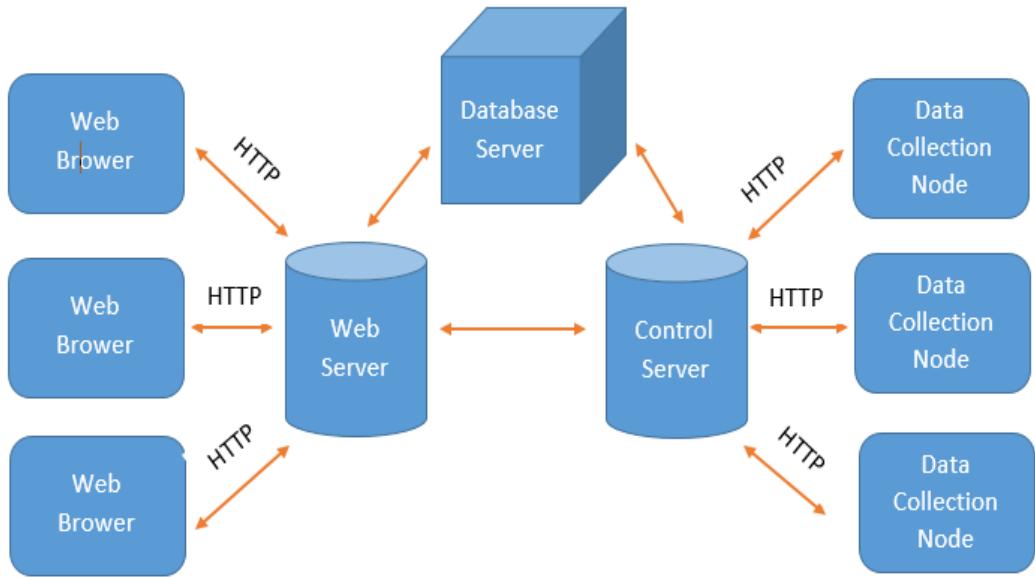
### **Nhược điểm**

- + CoAP là giao thức không tin cậy lầm do sử dụng UDP. Do đó, các thông điệp CoAP đến không có thứ tự hoặc sẽ bị lạc khi chúng đến đích;
- + Nó xác nhận mỗi lần nhận bản tin và do đó tăng thời gian xử lý. Hơn nữa, nó không xác minh xem bản tin nhận đã được giải mã đúng cách hay chưa;
- + Đây là giao thức không được mã hóa như MQTT và sử dụng DTLS để cung cấp bảo mật;
- + CoAP gặp vấn đề giao tiếp khi các thiết bị nằm sau NAT.

### **Ứng dụng CoAP cho nhà thông minh**

Thiết bị thông tin, thiết bị điều khiển và thiết bị truyền thông trong mạng nhà thông minh có đặc điểm là chi phí thấp và nhẹ. Do đó, CoAP có thể được coi là sự lựa chọn giao thức tốt nhất cho mạng truyền thông gia đình.

Mạng nhà thông minh cung cấp khả năng điều khiển và giám sát năng lượng tiêu thụ của các thiết bị trong nhà. Hệ thống kiểm soát năng lượng sử dụng ô cảm thông minh để giám sát thiết bị tiêu thụ điện năng để cung cấp thông tin điện áp, dòng điện và năng lượng khác. Nó có thể nhận ra cảnh báo mất an toàn, điều khiển từ xa và tiết kiệm năng lượng chủ động. Mọi nút thu thập dữ liệu với Client, CoAP có thể trao đổi thông tin với các nút khác. CoAP có thể được cài đặt trong mạng LAN hoặc Internet. Không giống như nhiều giao thức không dây cho các thiết bị tự động trong gia đình, CoAP được thiết kế không bị giới hạn trong mạng cục bộ mà cung cấp nền tảng cơ bản của web.



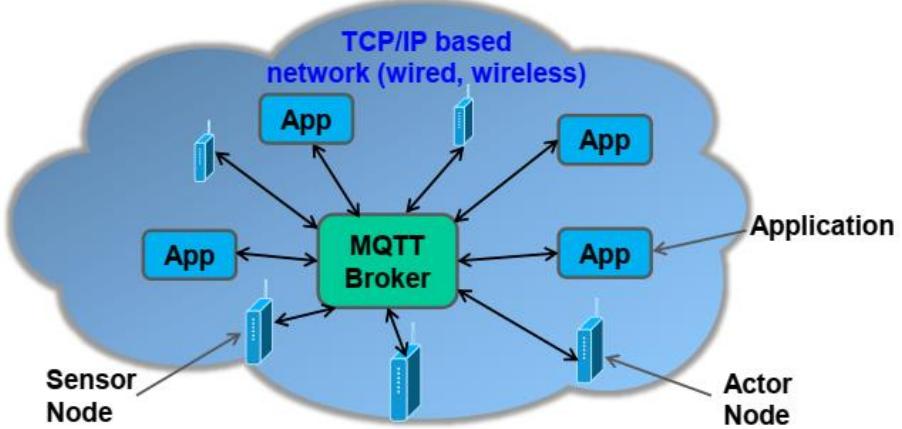
*Hình 4. 10: Sơ đồ hệ thống điều khiển năng lượng*

Trong hệ thống trên, các nút thu thập dữ liệu bao gồm một proxy, ổ cảm thông minh và mô-đun thu thập dữ liệu không dây. Thông tin năng lượng và thông tin môi trường của thiết bị được ổ cảm thông minh thu thập và chuyển đến mô-đun thu thập dữ liệu thông qua kênh không dây, sau đó gửi dữ liệu nối tiếp đến proxy để xử lý và đóng gói dữ liệu. Server điều khiển phân tích tất cả dữ liệu và lưu trữ chúng trong cơ sở dữ liệu. Hệ thống tích hợp mạng gia đình và Internet, người dùng có thể truy cập trang web của hệ thống để điều khiển từ xa công tắc, quản lý cấu hình, truy vấn mức tiêu thụ năng lượng, v.v.

## 4.5 Các giao thức Publish/Subscriber

### 4.5.1 Giao thức MQTT

Giao thức MQTT là một giao thức truyền thông điệp mở, và nhỏ gọn. Nó hoạt động theo mô hình Xuất bản/Theo dõi (Publish/Subscribe) sử dụng băng thông thấp, độ tin cậy cao, có khả năng hoạt động trong điều kiện đường truyền không ổn định phù hợp với các thiết bị có tài nguyên giới hạn, (ví dụ: mạng cảm biến không dây).



Hình 4. 11: Giao thức MQTT.

Giao thức MQTT là lần đầu tiên được giới thiệu bởi Andy Standford-Clark thuộc IBM và Arlen Nipper của Arcom vào năm 1999 để phục vụ dự án quản lý ống dẫn dầu qua sa mạc. Mục tiêu đặt ra là một giao thức có hiệu quả về mặt băng thông và sử dụng công suất pin thấp vì các thiết bị này sử dụng truyền thông vệ tinh mà tại thời điểm đó chi phí rất đắt đỏ. Hai nhà phát minh đã đặt ra các tiêu chí cụ thể cho giao thức MQTT:

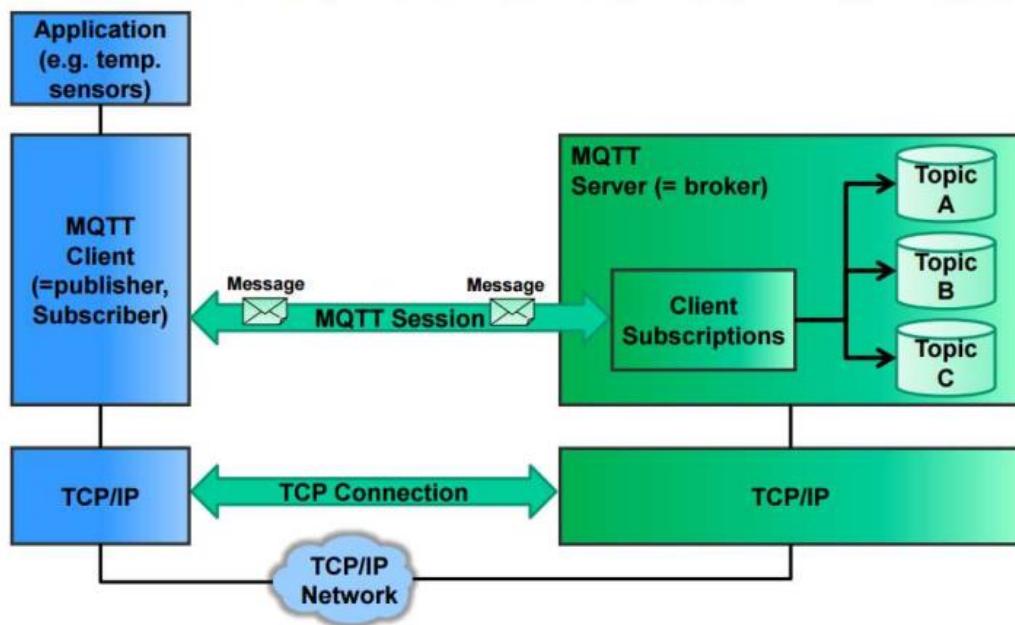
- + Khả năng thực hiện đơn giản.
- + Truyền dữ liệu với sự đảm bảo.
- + Giao thức nhẹ và sử dụng băng thông hiệu quả.
- + Không thể dự đoán dữ liệu.
- + Khả năng nhận thức các phiên MQTT liên tiếp.

Những mục tiêu này vẫn là cốt lõi của giao thức MQTT hiện tại. Tuy nhiên, trọng tâm chính của giao thức đã thay đổi từ các hệ thống nhúng độc quyền sang các trường hợp sử dụng Internet of Things (IoT) mở. Sau này, giao thức MQTT được chuẩn hoá bởi tổ chức OASIS vào năm 2013. Giao thức MQTT được sử dụng để kết nối các thiết bị nhúng, các mạng truyền thông với các ứng dụng hoặc phần mềm trung gian. Quá trình kết nối sử dụng cơ chế định tuyến (một-một, một-nhiều, nhiều-nhiều), làm cho MQTT trở thành giao thức kết nối tối ưu cho các hệ thống IoT và M2M. Giao thức MQTT hoạt động trên nền giao thức TCP, nó vận chuyển các bản tin theo ba mức chất lượng dịch vụ (QoS) khác nhau. Giao thức MQTT có một biến thể là giao thức MQTT-SN (hay còn gọi là MQTT-S) dành

cho các mạng cảm biến không dây. Khác với giao thức MQTT, giao thức MQTT-SN hoạt động trên nền giao thức UDP và sử dụng cờ định danh (ID) thay cho tên của các chủ đề. Các phiên bản hiện tại của giao thức MQTT và MQTT-SN tương ứng là v5.0 và v1.2. Tuy nhiên, đối với giao thức MQTT, phiên bản v3.1.1 vẫn đang được sử dụng rộng rãi nhất.

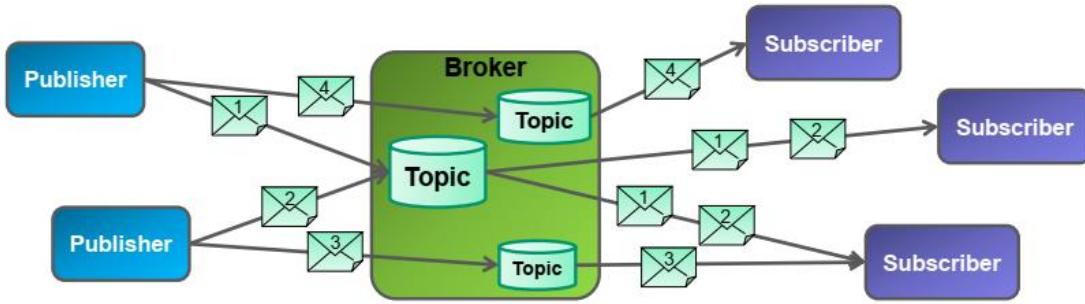
### Kiến trúc giao thức MQTT

MQTT sử dụng mô hình Client-Server, trong đó, các Clients (ví dụ các thiết bị IoT như cảm biến, cơ cấu chấp hành, v.v.) kết nối tới một server (được gọi là Broker) trên nền giao thức TCP. MQTT được coi là giao thức định hướng bản tin, và nó sử dụng các “chủ đề” thay thế cho địa chỉ (ví dụ địa chỉ IP, cổng) mà bản tin được gửi đến. Giao thức MQTT bao gồm 2 thành phần chính: MQTT Client, MQTT Broker, phiên MQTT, khối đăng kí, và các chủ đề.



Hình 4. 12: Kiến trúc của giao thức MQTT.

Một thiết bị có thể theo dõi nhiều chủ đề khác nhau, và nó sẽ nhận được tất cả các bản tin được xuất bản tới các chủ đề đó.



Hình 4. 13: Mô hình Xuất bản/Theo dõi.

### **MQTT Client**

Một MQTT Client có thể là một thiết bị bất kỳ có khả năng gửi và nhận dữ liệu, (ví dụ, cảm biến, server, v.v.). Có 2 loại MQTT Client là “Publisher” và “Subscriber”. Trong đó, thiết bị đóng vai trò là publisher có khả năng xuất bản các bản tin, ngược lại, thiết bị đóng vai trò là subscriber có khả năng nhận các bản tin thuộc chủ đề mà nó theo dõi. Các MQTT Client kết nối tới MQTT Broker thông qua một mạng truyền thông. Nói theo cách khác, tất cả các thiết bị thực thi giao thức MQTT qua TCP/IP đều được gọi là MQTT Client. Việc thực hiện giao thức MQTT một cách dễ dàng là một trong những nguyên nhân làm cho MQTT phù hợp cho các thiết bị nhỏ. Thư viện cho các MQTT Client được cung cấp cho nhiều loại ngôn ngữ khác nhau, ví dụ Android, Arduino, C/C++, C#, iOS, Java, Javascript, .Net, v.v.



Hình 4. 14: Publisher và Subscriber.

### **MQTT Broker**

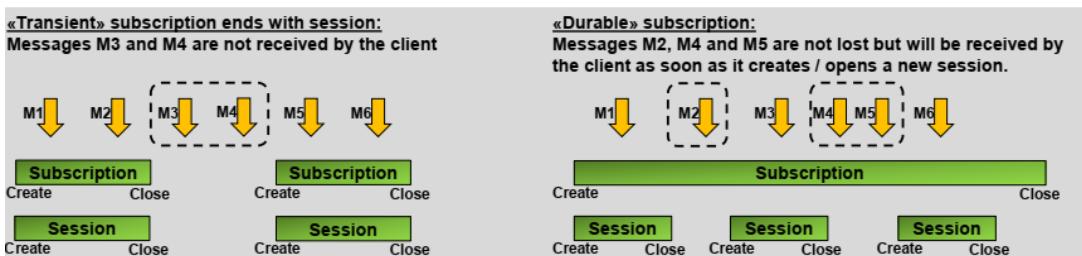
MQTT Broker là một thiết bị trung tâm chịu trách nhiệm xử lý truyền thông giữa các MQTT Client và phân phát các bản tin giữa chúng dựa vào các chủ đề mà chúng theo dõi. Một Broker có khả năng xử lý hàng ngàn thiết bị kết nối tới nó tại cùng một thời điểm. Dựa trên bản tin nhận được, Broker sẽ xác định và tìm kiếm tất cả các thiết bị đã theo dõi tới chủ đề của bản tin đó và sẽ chuyển các bản tin tới các thiết bị đó. Một nhiệm vụ khác của Broker là xác thực và ủy quyền cho các Client. Vì mọi tin nhắn đều đi qua Broker nên nó phải có khả năng mở rộng cao, có thể tích hợp vào các hệ thống Backend, dễ dàng giám sát và có khả năng chống lỗi.

## Phiên MQTT

Một phiên MQTT được định nghĩa là toàn bộ quá trình Client được kết nối đến Server. Tất cả các hoạt động truyền thông giữa Client và Server đều được diễn ra trong phiên MQTT. Có hai loại phiên MQTT được định nghĩa trong giao thức MQTT là thường trực và không thường trực được bởi thiết lập cờ *cleanSession*.

### Khởi đăng kí

Khởi đăng kí trên Broker sẽ làm nhiệm vụ gán Client đến chủ đề tương ứng mà Client đăng kí theo dõi, do đó, khi Client có thể nhận tất cả các bản tin thuộc chủ đề đó. Có hai loại đăng kí được định nghĩa trong giao thức MQTT là đăng kí ngắn hạn (Transient Subscription) và đăng kí dài hạn (Durable Subscription). Như được minh họa ở hình bên dưới, đối với loại đăng kí ngắn hạn, Client sẽ không nhận được các bản tin M3, M4 vì đăng kí của Client sẽ kết thúc khi phiên MQTT kết thúc. Ngược lại, mặc dù các bản tin M2, M4, M5 đến khi phiên MQTT của Client kết thúc, nó vẫn nhận được các bản tin này ngay khi nó tạo các phiên MQTT mới.



Hình 4. 15: Các loại đăng kí trong giao thức MQTT.

### Chủ đề

Về mặt kỹ thuật, chúng ta có thể hiểu chủ đề là một hàng đợi các bản tin có cùng đặc điểm chung nào đó (ví dụ, các bản tin thông báo dữ liệu nhiệt độ được xuất bản trong chủ đề “/nhietdo/”).

Mỗi chủ đề có thể chứa nhiều cấp chủ đề, mỗi cấp chủ đề được phân tách bởi dấu gạch chéo “/”. Lưu ý rằng tên chủ đề phải chứa ít nhất một ký tự, đồng thời nó cũng cho phép tồn tại khoảng trắng giữa các ký tự. Tên chủ đề cũng phân biệt chữ thường và chữ in hoa. Ví dụ, “\_nhathongminh/nhietdo” và “\_Nhathongmin/Nhietdo” là hai chủ đề khác nhau.



Hình 4. 16: Định dạng phân cấp của chủ đề.

MQTT hỗ trợ ký tự đại diện (wildcards) trong cú pháp khai báo chủ đề. Khi một Client theo dõi một chủ đề nào đó, nó có thể đăng ký chính xác một chủ đề mà nó quan tâm hoặc nó có thể sử dụng wildcards để đăng ký nhiều chủ đề cùng lúc. Wildcards chỉ được dùng cho các Subscriber để đăng ký các chủ đề. Có hai loại wildcards là đơn cấp hoặc đa cấp.

Wildcard đơn cấp sử dụng dấu "+" làm đại diện cho một cấp chủ đề. Ví dụ, nếu muốn biết thông tin nhiệt độ của từng phòng thì Subscriber chỉ cần khai báo chủ đề theo cú pháp sau đây:



Hình 4. 17: Cú pháp Wildcard đơn cấp.

Khi đó, Subscriber sẽ đăng ký các chủ đề nhiệt độ ở tất cả các phòng thuộc tầng trệt (groundfloor):

- ✓ myhome / groundfloor / livingroom / temperature
- ✓ myhome / groundfloor / kitchen / temperature
- ✗ myhome / groundfloor / kitchen / brightness
- ✗ myhome / firstfloor / kitchen / temperature
- ✗ myhome / groundfloor / kitchen / fridge / temperature

Hình 4. 18: Các chủ đề được khai báo theo cú pháp trên sử dụng Wildcard đơn cấp.

Wildcard đa cấp sử dụng dấu "#" làm đại diện cho tất cả các chủ đề ở dưới chủ đề hiện tại. Ví dụ, nếu muốn biết tất cả các thông tin của một tầng trệt (groundfloor) thì Subscriber chỉ cần khai báo chủ đề theo cú pháp sau đây:



Hình 4. 19: Cú pháp Wildcard đa cấp.

Khi đó, Subscriber sẽ đăng ký tất cả các chủ đề nằm dưới chủ đề “myhome /groundfloor”.

- ✓ myhome / groundfloor / livingroom / temperature
- ✓ myhome / groundfloor / kitchen / temperature
- ✓ myhome / groundfloor / kitchen / brightness
- ✗ myhome / firstfloor / kitchen / temperature

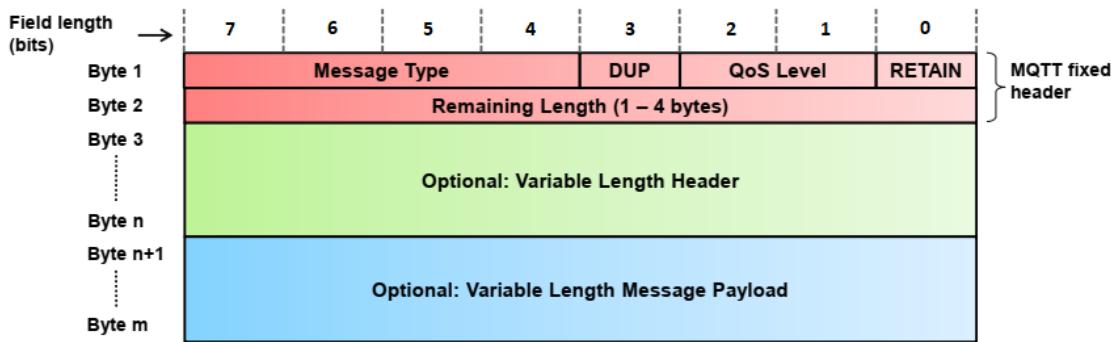
Hình 4. 20: Các chủ đề được khai báo theo cú pháp trên sử dụng Wildcard đa cấp.

## Các gói tin điều khiển trong giao thức MQTT

### Định dạng gói tin

Giao thức MQTT hoạt động thông qua việc trao đổi một loạt các gói tin điều khiển theo quy ước. Định dạng của các gói tin điều khiển được mô tả như sau:

- Một gói tin điều khiển MQTT bao gồm 3 phần: tiêu đề có độ dài cố định (2 bytes), tiêu đề có độ dài thay đổi, và payload. Trong đó, 2 phần sau là tùy chọn, không bắt buộc phải có, và thường làm cho việc xử lý giao thức thêm phức tạp.
- Tuy nhiên, giao thức MQTT được tối ưu cho các mạng không ổn định và có băng thông giới hạn, nên các phần tùy chọn được sử dụng để giảm số lần truyền dữ liệu nhiều nhất có thể.



Hình 4. 21: Định dạng của gói tin điều khiển.

Phần tiêu đề cố định có độ dài 2 bytes bao gồm 3 trường: trường khai báo loại gói tin, trường chứa các cờ điều khiển, và trường chứa thông tin về độ dài còn lại của gói tin (số bytes).

Trường khai báo loại gói tin điều khiển là một số 4 bit không dấu nằm ở byte 1 (từ bit 7 → 4), được mô tả trong bảng dưới đây.

Bảng 4.2. Các loại gói tin điều khiển.

Loại gói tin	Giá trị (thập phân)	Luồng hoạt động	Mô tả
Reserved	0	Chưa dùng	Chưa dùng
CONNECT	1	Client → Broker	Client yêu cầu kết nối tới Broker
CONNACK	2	Broker → Client	Kết nối được chấp nhận
PUBLISH	3	Publisher → Broker Broker → Subscriber	Xuất bản bản tin
PUBACK	4	Publisher → Broker Broker → Subscriber	Việc xuất bản bản tin được chấp nhận
PUBREC	5	Publisher → Broker Broker → Subscriber	Gói tin PUBLISH đã được nhận (quá trình chuyển phát lần 1)

PUBREL	6	Publisher → Broker Broker → Subscriber	Gói tin PUBLISH đã đ
PUBCOMP	7	Publisher → Broker Broker → Subscriber	
SUBSCRIBE	8	Client → Broker	Yêu cầu đăng ký từ Client
SUBACK	9	Broker → Client	Xác nhận đăng ký được chấp nhận
UNSUBSCRIBE	10	Client → Broker	Yêu cầu bỏ đăng ký từ Client
UNSUBACK	11	Broker → Client	Xác nhận việc bỏ đăng ký
PINGREQ	12	Client → Broker	Yêu cầu PING
PINGRESP	13	Broker → Client	Phản hồi yêu cầu PING
DISCONNECT	14	Client → Broker	Client đang mất kết nối
Reserved	15	Chưa dùng	Chưa dùng

Các bit còn lại trong byte 1 chứa các cờ điều khiển (bit 3 → 0):

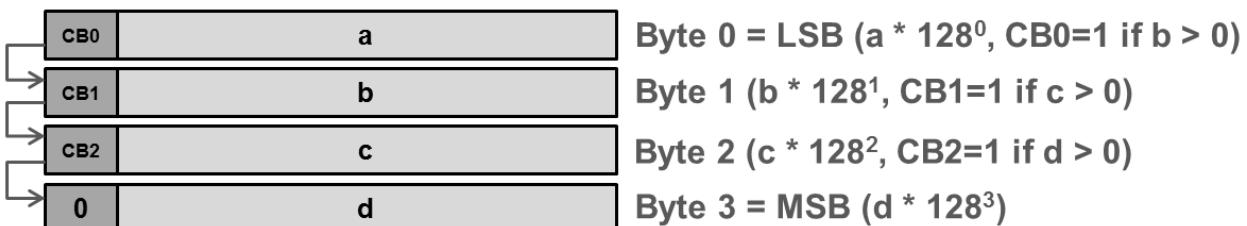
- Cờ DUP (bit 3): Cờ này được bật lên khi Client hoặc Broker đang có gửi lại một gói tin (PUBLISH, PUBREL, SUBSCRIBE, hoặc UNSUBSCRIBE). Nhìn vào giá trị của cờ DUP ta sẽ biết được gói tin đã được nhận trước đó hay chưa? Khi cờ DUP được thiết lập (QoS > 0), phần tiêu đề có độ dài thay đổi sẽ chứa định danh của gói tin.
- Cờ QoS (bit 2 → 1): Cờ này để xác định độ đảm bảo của việc nhận gói tin PUBLISH. Các giá trị của QoS được mô tả trong bảng 1.2.
- Cờ RETAIN (bit 0): Cờ này chỉ được sử dụng cho gói tin PUBLISH. Xem chi tiết tại mục 1.3.2.3.

**Bảng 4.3. Giá trị QoS.**

Giá trị QoS	Bit 2	Bit 1	Mô tả		
0	0	0	Nhiều nhất một lần	Gửi rồi quên ngay	$\geq 1$
1	0	1	Ít nhất một lần	Gửi có xác nhận bằng ACK	$\leq 1$
2	1	0	Chính xác một lần	Nhận có đảm bảo	$= 1$
3	1	1	Chưa dùng		

Trường này bắt đầu từ byte 2, dùng để lưu thông tin về độ dài (byte) còn lại của gói tin, bao gồm phần tiêu đề có độ dài thay đổi và phần dữ liệu (payload) của gói tin. Trường này có độ dài từ 1 → 4 byte, do đó, độ dài còn lại của gói tin tối đa là 268.435.455 bit (256 MB). Độ dài còn lại của gói tin được tính theo cách sau:

- Mỗi byte trong trường này được chia thành 2 phần, trong đó, bit có trọng số cao nhất (MSB) được gọi là bit báo có byte tiếp theo hay không (CB), và bảy bit có trọng số thấp nhất (LSB) còn lại biểu diễn dữ liệu.
- Độ dài còn lại của gói tin được tính theo công thức sau:  $a * 128^0 + b * 128^1 + c * 128^2 + d * 128^3$ .



Hình 4. 22: Cách tính giá trị trường Độ dài còn lại.

**Ví dụ 1:** Độ dài còn lại của gói tin (RL) là 364 bit sẽ được biểu diễn như sau:

$$RL = 108 * 128^0 + 2 * 128^1 = 108 + 256, \text{ trong đó, } a = 108, CB0 = 1, b = 2, CB1 = 0.$$

Do đó, 2 byte của RL là: 0xEC (byte 2), 0x02 (byte 3).

**Ví dụ 2:** Độ dài còn lại của gói tin (RL) là 25897 bit sẽ được biểu diễn như sau:

$$RL = 41 * 128^0 + 74 * 128^1 + 1 * 128^2 = 41 + 9.472 + 16.384 = 25.897,$$

( $a = 41$ ,  $CB0 = 1$ ,  $b = 74$ ,  $CB1 = 1$ ,  $c = 1$ ,  $CB2 = 0$ ).

Do đó, 3 byte của RL là: 0xA9 (byte 1), 0xCA (byte 2), 0x01 (byte 3).

Một số gói tin điều khiển chứa phần tiêu đề có độ dài thay đổi. Nội dung của phần này phụ thuộc vào loại gói tin. Trong đó, trường định danh gói tin có độ dài 2 byte được sử dụng trong một số loại gói tin: PUBLISH (QoS > 0), PUBACK, PUBREC, PUBREL, PUBCOMP, SUBSCRIBE, SUBACK, UNSUBSCRIBE, UNSUBACK.

Các gói tin điều khiển PUBLISH (QoS > 0), SUBSCRIBE, UNSUBSCRIBE phải chứa một định danh khác 0, dài 16 bit. Mỗi khi Client gửi một gói tin điều khiển thuộc các loại kể trên, một mã định danh chưa được sử dụng sẽ được gán cho gói tin đó. Nếu Client gửi lại gói tin đó, thì nó phải dùng lại đúng mã định danh đó để gửi lại trong lần tiếp theo. Mã định danh đó sẽ được giải phóng khi Client đã xử lý xong gói tin xác nhận tương ứng. Ví dụ, trong trường hợp gói tin PUBLISH có QoS = 1, thì PUBACK là gói tin xác nhận tương ứng. Tương tự, PUBCOMP là gói tin xác nhận tương ứng với gói tin PUBLISH có QoS = 2. Đối với gói tin SUBSCRIBE, và UNSUBSCRIBE thì các gói tin xác nhận tương ứng lần lượt là SUBACK và UNSUBACK. Điều kiện trên cũng được áp dụng khi Broker gửi các gói tin PUBLISH có QoS > 0 đến Client.

Các gói tin PUBACK, PUBREC, và PUBREL phải chứa định danh giống như gói PUBLISH được gửi đến. Tương tự, các gói SUBACK và UNSUBACK cũng phải chứa mã định danh được sử dụng trong các gói tin SUBSCRIBE và UNSUBSCRIBE.

Lưu ý, Client và Broker phải sử dụng các định danh gói tin độc lập với nhau. Mỗi cặp Client-Broker trao đổi thông tin với nhau sử dụng chung một định danh gói tin.

Tương tự như phần tiêu đề có độ dài thay đổi, thì phần dữ liệu cũng chỉ được sử dụng bởi một số gói tin điều khiển nhất định như được liệt kê trong bảng sau:

**Bảng 4.4. Các loại gói tin điều khiển.**

Loại gói tin	Dữ liệu	Loại gói tin	Dữ liệu
CONNECT	Bắt buộc	SUBSCRIBE	Bắt buộc
CONNACK	Không có	SUBACK	Bắt buộc
PUBLISH	Không bắt buộc	UNSUBSCRIBE	Bắt buộc
PUBACK	Không có	UNSUBACK	Không có
PUBREC	Không có	PINGREQ	Không có

PUBREL	Không có	PINGRESP	Không có
PUBCOMP	Không có	DISCONNECT	Không có

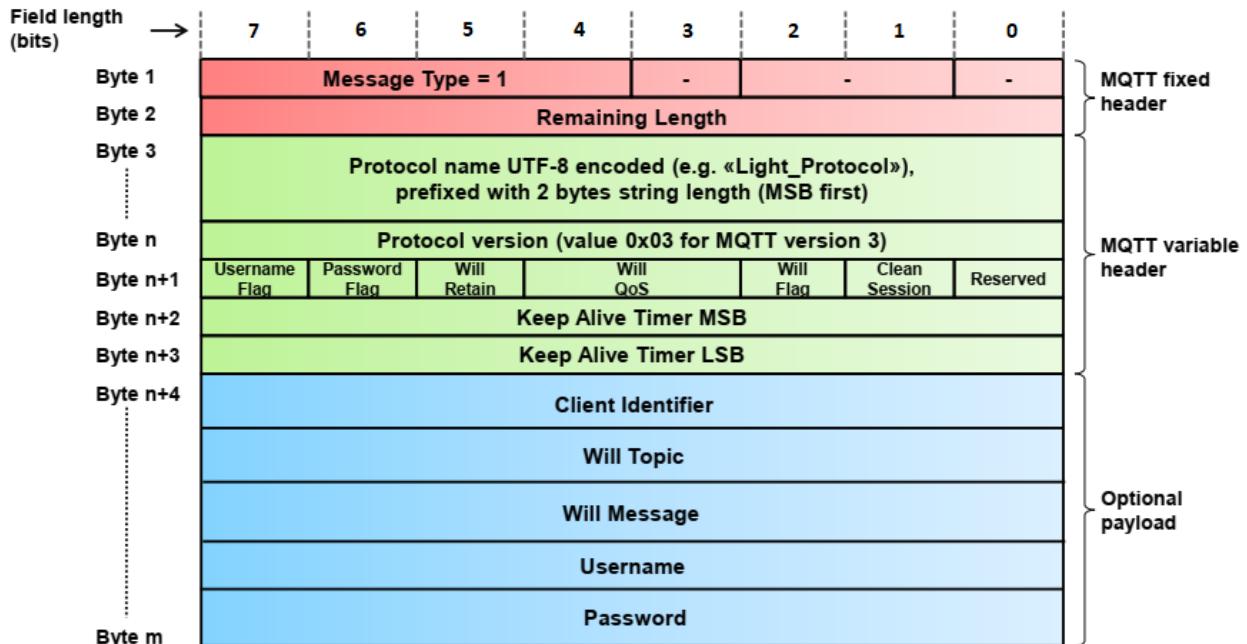
Trong trường hợp gói tin PUBLISH, thì phần dữ liệu của gói tin này chứa bản tin ứng dụng. Chính vì vậy, phần dữ liệu trong gói tin này có thể có hoặc không.

### Các gói tin điều khiển trong giao thức MQTT

Gói tin CONNECT:

Phần tiêu đề cố định:

- Byte 1: 4 bit ( $7 \rightarrow 4$ ) khai báo loại gói tin, 4 bit ( $3 \rightarrow 0$ ) chưa sử dụng.
- Byte 2: Trường khai báo độ dài còn lại được tính theo phần 3.1.1.3.



Hình 4. 23: Gói tin CONNECT.

Phần tiêu đề có độ dài thay đổi: Phần này bao gồm 4 trường theo thứ tự sau: Trường khai báo tên giao thức, trường khai báo phiên bản MQTT sử dụng, các cờ kết nối, và trường khai báo thời gian Keep Alive.

- *Trường khai báo tên giao thức:* Sử dụng mã UTF-8 để mã hoá tên giao thức “MQTT”.
- *Trường khai báo phiên bản:* Sử dụng 8 bit không dấu để biểu diễn phiên bản giao thức sử dụng, ví dụ phiên bản 3.1.1 có giá trị là 4 (0x04).
- *Các cờ kết nối*
  - + Cờ báo User Name (bit 7): Nếu cờ này được thiết lập là 1 thì phần dữ liệu phải khai báo User Name, ngược lại thì không phải khai báo User Name.
  - + Cờ báo Password (bit 6): Nếu cờ này được thiết lập là 1 thì phần dữ liệu phải khai báo Password, ngược lại thì không phải khai báo Password. Nếu cờ User Name được thiết lập là 0 thì cờ này cũng phải thiết lập là 0.
  - + Cờ CleanSession (bit 1): Nếu cờ này được thiết lập là 1, Broker sẽ huỷ tất cả các thông tin trước đó của các Client đang kết nối đến Broker. Nếu cờ này được thiết lập là 0, Broker giữ tất cả các theo dõi của các Client (kể cả Client đang không kết nối tới Broker), đồng thời lưu lại tất cả các bản tin có mức QoS 1 và 2 của các Client này. Broker sẽ xuất bản tất cả các bản tin đó đến các Client này khi chúng kết nối lại tới Broker.
  - + Cờ WILL (bit 2): Nếu cờ này được thiết lập là 1, một bản tin WILL phải được lưu ở Broker. Bản tin này phải được xuất bản trước khi quá trình kết nối kết thúc trừ khi nó bị xoá bởi Broker khi nhận được gói tin DISCONNECT. Các cờ khác như WILL QoS, WILL Retain sẽ được sử dụng bởi Broker, và phần dữ liệu (payload) phải biểu diễn các trường WILL Topic và WILL Message. Ngược lại, nếu cờ WILL được thiết lập là 0, các cờ WILL QoS và WILL Retain được thiết lập là 0 và phần dữ liệu (payload) không chứa các trường WILL Topic và WILL Message. Xem thêm về bản tin WILL ở mục 1.5.4.
  - + Cờ WILL QoS (bit 4 → 3): Cờ này được sử dụng để khai báo mức QoS khi xuất bản bản tin WILL (với điều kiện cờ WILL được thiết lập là 1). Nếu cờ WILL được thiết lập là 0, thì WILL QoS sẽ được thiết lập là 0 (0x00).
  - + Cờ WILL Retain (bit 5): Nếu cờ này được thiết lập là 1, bản tin WILL sẽ được lưu lại ở Broker sau khi nó xuất bản bản tin này.
- *Trường khai báo thời gian KeepAlive*

- + Trường này được sử dụng bởi Broker để phát hiện các kết nối giữa Broker và Client bị ngắt.
- + Keep Alive được tính bằng giây sử dụng 1 word (16 bit). Keep Alive là khoảng thời gian tối đa giữa thời điểm Client truyền xong một gói tin điều khiển và thời điểm mà nó bắt đầu truyền một gói tin điều khiển tiếp theo. Trong trường hợp không có gói tin điều khiển nào thì Client phải gửi một gói tin PINGREQ. Gói tin PINGREQ có thể được gửi bởi Client bất kỳ lúc nào, không bị ảnh hưởng bởi Keep Alive. Gói tin PINGRESP được dùng để xác định mạng và Broker vẫn đang hoạt động.
- + Nếu giá trị của Keep Alive khác không, và Broker không nhận được bất kỳ một gói tin điều khiển nào trong nhiều nhất 1,5 lần thời gian Keep Alive thì nó phải ngắt kết nối mạng đến Client như thể là mạng bị lỗi.
- + Lưu ý, nếu Client không nhận được gói tin PINGRESP sau khi nó giữ gói tin PINGREQ trong một khoảng thời gian hợp lý thì nó cũng phải đóng kết nối với Broker.
- + Nếu giá trị của Keep Alive bằng không thì cơ chế Keep Alive không hoạt động.

**Phần dữ liệu:** Phần dữ liệu của gói tin CONNECT chứa một hoặc nhiều trường có độ dài cố định được xác định bởi phần tiêu đề có độ dài thay đổi ở trên. Các trường này (nếu có) phải được sắp xếp theo thứ tự sau: Định danh Client, WILL topic, WILL Message, User Name, Password.

#### - *Trường định danh Client*

- + Mã định danh Client (ClientID) được dùng để định danh Client kết nối đến Broker. Client và Broker sử dụng ClientID để xác định trạng thái của phiên MQTT hiện tại giữa Client và Broker.
- + ClientID là trường bắt buộc và phải xuất hiện đầu tiên trong phần dữ liệu của gói tin CONNECT. Nó là một chuỗi kí tự sử dụng mã hoá UTF-8 có độ dài ngắn nhất là 1 byte và dài nhất là 23 byte. Lưu ý, nó chỉ sử dụng các kí tự: a → z, A → Z, và 0 → 9.

#### - *Trường WILL Topic*

- + Nếu cờ WILL được thiết lập là 1, thì trường WILL Topic sẽ xuất hiện tiếp theo ClientID trong phần dữ liệu của gói tin CONNECT.
- *Trường WILL Message*
  - + Tương tự như trên, nếu cờ WILL được thiết lập là 1, thì trường WILL Message cũng sẽ xuất hiện sau trường WILL Topic. Trường này chính là bản tin ứng dụng được xuất bản đến WILL Topic. Xem ứng dụng của bản tin WILL ở mục 1.5.4.
- *Trường UserName*
  - + Nếu cờ UserName được thiết lập là 1 thì trường này sẽ là trường tiếp theo trong phần dữ liệu. User Name sử dụng mã hoá UTF-8. Nó được sử dụng cho việc xác nhận và cấp phép cho Client của Broker.
  - + Nếu cờ UserName được thiết lập là 0 thì trường này không xuất hiện trong phần dữ liệu.
- *Trường Password*
  - + Nếu cờ Password được thiết lập là 1, trường này sẽ xuất hiện trong phần dữ liệu sau trường UserName để thực hiện xác thực cho Client.
  - + Trường này có thể chứa tối đa 65535 byte dữ liệu nhị phân để biểu diễn Password. Độ dài dữ liệu nhị phân để biểu diễn Password trong trường Password được xác định bởi 2 byte đầu tiên.

### **Tóm tắt hoạt động của gói tin CONNECT**

Khi một kết nối TCP/IP được thiết lập từ Client đến Broker, thì một phiên ở tầng MQTT cũng được tạo ra bằng việc Client gửi một gói tin CONNECT đến Broker. Broker sẽ gửi gói tin CONNACK để trả lời gói tin CONNECT từ Client.

- Nếu Broker không nhận được gói tin CONNET từ Client trong một khoảng thời gian nào đó sau khi thiết lập kết nối TCP/IP, thì Broker nên đóng kết nối lại. Thời gian chờ để nhận được CONNECT phụ thuộc vào ứng dụng và điều kiện kết nối.
- Nếu Client gửi một gói tin CONNECT không hợp lệ, Broker nên đóng luôn kết nối. Không hợp lệ ở đây bao gồm việc khác nhau về Protocol Name hoặc

Protocol Version Numbers. Trước khi đóng kết nối, Broker cần gửi một gói tin CONACK với mã trả lại có mã 0x80 hoặc cao hơn.

- Nếu Broker đã xử lý gói tin CONNECT rồi mới phát hiện ra có một trường nào đó không hợp lệ, nó sẽ gửi lại gói tin CONNACK với mã phản hồi là 0x80 hoặc lớn hơn trước khi hủy kết nối này.

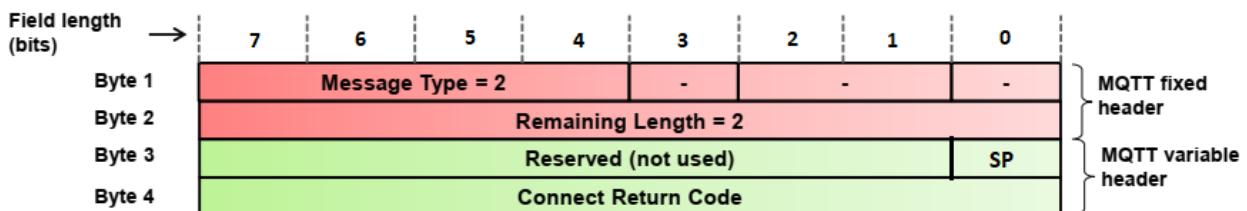
Nếu việc xác thực thành công thì Broker phải thực hiện các bước sau:

- Nếu một Client kết nối bằng một Client ID đang được kết nối với Server rồi, thì Client trước đó phải được ngắt kết nối một cách bắt buộc bởi Broker trước khi thực hiện luồng CONNECT với Client mới.
- Broker thực hiện tính năng CleanSession được mô tả trong cờ *cleanSession*.
- Broker phải phản hồi gói tin CONNACK tương ứng với gói tin CONNECT với mã trả lại có giá trị 0.
- Thực hiện quá trình chuyển phát bản tin và duy trì cơ chế Keep Alive.

### **Gói tin CONNACK**

Gói tin CONNACK được gửi bởi Broker để phản hồi lại gói tin CONNECT từ Client. Đây là gói tin đầu tiên được gửi từ Broker đến Client.

Nếu Client không nhận được một gói tin CONNACK từ Broker trong một khoảng thời gian nhất định, thì Client cũng sẽ đóng kết nối đó lại, và tạo một phiên MQTT bằng một socket mới đến Broker rồi tiếp tục gửi yêu cầu kết nối bằng gói tin CONNECT. Tương tự gói tin CONNECT, thời gian chờ để nhận được CONNACK phụ thuộc vào ứng dụng và điều kiện kết nối.



Hình 4. 24: Gói tin CONNACK.

Phân tiêu đề cố định: Loại gói tin được khai báo trong 4 bit ( $7 \rightarrow 4$ ) của byte 1 là 2, các cờ điều khiển không được khai báo trong gói tin CONNACK.

- Độ dài còn lại của gói tin là 2 byte chính là độ dài của phần tiêu đề có độ dài thay đổi. Lưu ý là gói tin CONNACK không có phần dữ liệu.

Phần tiêu đề có độ dài thay đổi:

- Cờ SessionPresent (SP)
  - + Cờ này nằm ở bit 0 của byte thứ 3 của gói tin (hay byte 1 trong phần tiêu đề có độ dài thay đổi).
  - + Nếu Broker chấp nhận kết nối từ Client với cờ CleanSession là 0 thì Broker phải thiết lập cờ SessionPresent là 0 trong bản tin CONNACK cùng với mã phản hồi là 0.
  - + Nếu Broker trả về gói tin CONNACK với mã phản hồi khác không thì nó phải thiết lập cờ SessionPresent là 0.
- Mã phản hồi trong gói tin CONNACK
  - + Các giá trị của mã phản hồi được liệt kê trong bảng 1.4.
  - + Nếu Broker nhận được một gói tin CONNECT hợp lệ từ Client, nhưng nó không thể xử lý gói tin này vì một số lý do nào đó thì nó phải gửi lại một gói tin phản hồi CONNACK chứa mã phản hồi khác không được liệt kê trong bảng phía trên và sau đó đóng lại kết nối tới Client.

**Bảng 4.5. Mã phản hồi trong gói tin CONNACK**

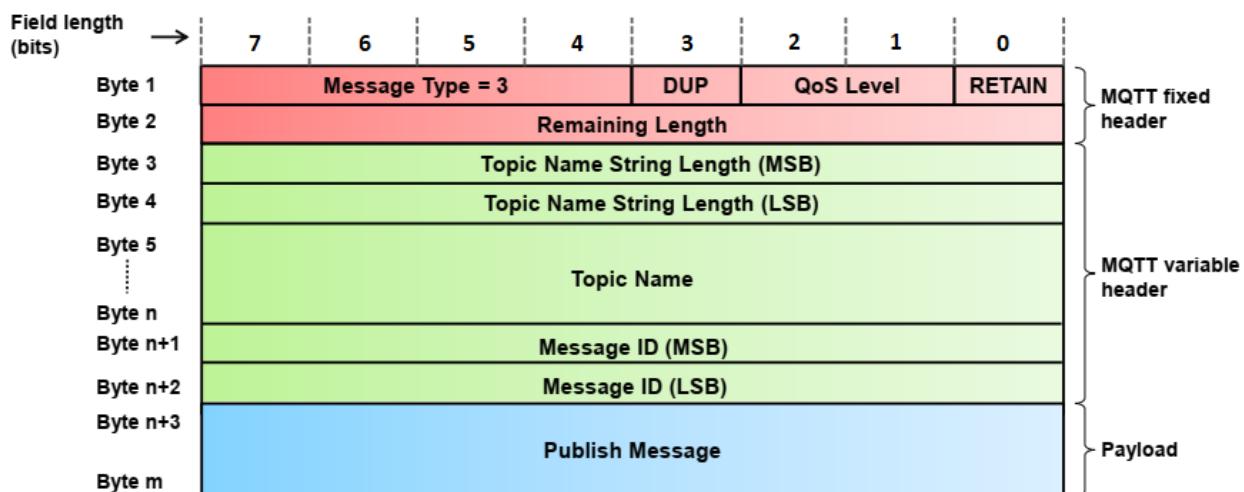
Giá trị	Mã phản hồi	Mô tả
0	0x00: Chấp nhận kết nối	Cho phép kết nối
1	0x01: Từ chối kết nối do phiên bản không phù hợp	Broker không hỗ trợ phiên bản của giao thức mà Client yêu cầu
2	0x02: Từ chối kết nối do định danh sai	Mã định danh được mã hoá theo UTF-8 nhưng không được cho phép bởi Broker
3	0x03: Từ chối kết nối do Broker không sẵn sàng	Kết nối mạng được thực hiện nhưng dịch vụ MQTT không sẵn sàng

4	0x04: Từ chối kết nối do User Name và Password sai	User Name và Password có định dạng không đúng
5	0x05: Từ chối kết nối do chưa được cấp phép	Client không được phép kết nối
6 - 255		Chưa được sử dụng

### Gói tin PUBLISH

Gói tin PUBLISH được sử dụng để chuyển bản tin MQTT từ Client (Publisher) đến Broker hoặc từ Broker đến Client (Subscriber). Bên nhận sẽ phải phản hồi lại gói tin PUBLISH theo các mức QoS khác nhau:

- QoS = 0: Không cần phải phản hồi.
- QoS = 1: Phản hồi với bản tin PUBACK.
- QoS = 2: Phản hồi với bản tin PUBREC.



Hình 4. 25: Gói tin PUBLISH.

Phần tiêu đề có độ dài cố định:

- Cờ DUP
  - + Cờ này nằm ở bit 3 của byte 1 của gói tin PUBLISH.
  - + Nếu cờ này là 0 nghĩa là bên gửi (Publisher hoặc Broker) đang gửi bản tin PUBLISH này lần đầu tiên. Ngược lại, nếu cờ này là 1 nghĩa là bên gửi đang cố gửi lại bản tin này.

- + Tuy nhiên, cờ DUP chỉ được thiết lập là 1 đối với các mức QoS 1 và 2, nó được thiết lập là 0 với mọi gói tin có mức QoS là 0.
- + Trong trường hợp bên nhận là Broker, thì giá trị của cờ DUP trong gói tin PUBLISH đi phải được thiết lập độc lập với giá trị DUP trong gói tin PUBLISH đến.
- Cờ QoS: Cờ này được thiết lập ở các bit 2 và 1 của byte 1 trong gói tin PUBLISH. Có 3 mức QoS được liệt kê trong bảng 1.2.
- Cờ Retain:
  - + Nếu cờ Retain trong gói tin PUBLISH được gửi từ Client đến Broker được thiết lập là 1, thì Broker phải lưu bản tin và giá trị QoS để chuyển phát đến các Subscriber sẽ đăng ký theo dõi chủ đề chứa bản tin đó trong tương lai. Trong trường hợp Broker nhận được một bản tin có QoS bằng 0 thì nó sẽ xoá hết các bản tin Retain được lưu trước đó trong chủ đề chứa bản tin này và lưu bản tin này vào trong chủ đề đó như là một bản tin Retain mới. Tuy nhiên, nó cũng có thể loại bỏ bản tin này bất cứ lúc nào. Khi đó, chủ đề đó sẽ không chứa bất kỳ bản tin Retain nào.
  - + Khi Broker gửi các gói tin PUBLISH tới Client thì Broker phải thiết lập cờ Retain là 1 nếu gói tin PUBLISH được gửi ngay sau khi Client đăng ký theo dõi chủ đề chứa bản tin đó. Cờ Retain được thiết lập lại thành 0 nếu gói tin PUBLISH được gửi tới chủ đề mà Client đã theo dõi bất kể giá trị của cờ này trong gói tin PUBLISH đến Broker.
  - + Nếu gói tin PUBLISH được xử lý bởi Broker và gửi đến Client có cờ Retain là 1 và phần dữ liệu không chứa dữ liệu thì tất cả các bản tin Retain nằm trong chủ đề được khai báo trong gói tin PUBLISH sẽ bị loại bỏ, và các Subscriber mới sẽ không thể nhận được bản tin Retain khi chúng đăng ký theo dõi chủ đề đó. Đồng thời, bản tin trong gói tin PUBLISH đó cũng không được lưu như là một bản tin Retain mới.
  - + Nếu gói tin PUBLISH được gửi từ Client tới Broker với cờ Retain là 0 thì Broker sẽ không được lưu bản tin cũng như không được loại bỏ hoặc thay thế bất kỳ bản tin Retain nào.

Phần tiêu đề có độ dài thay đổi:

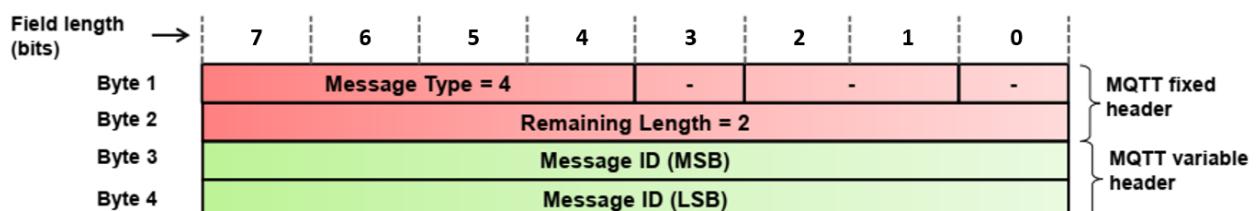
- Trường tên chủ đề:
  - + Trường này xác định chủ đề mà bản tin được xuất bản đến.
  - + Tên chủ đề phải xuất hiện ở đầu tiên trong phần tiêu đề có độ dài thay đổi và được mã hoá UTF-8.
  - + Tên chủ đề trong gói tin PUBLISH không được chứa Wildcards.
  - + Mặc dù, tên chủ đề được khai báo trong bản tin PUBLISH phải trùng với tên chủ đề nằm trong bộ lọc chủ đề được đăng ký theo dõi. Tuy nhiên, Broker được phép ghi đè tên chủ đề, do đó nó có thể thay đổi tên chủ đề trong gói tin PUBLISH ban đầu.
- Trường định danh gói tin: Trường này chỉ được khai báo trong các gói tin PUBLISH có QoS mức 1 và 2.

Phần dữ liệu:

- Phần dữ liệu trong gói tin PUBLISH chứa bản tin của ứng dụng. Nội dung và định dạng của dữ liệu được định nghĩa bởi ứng dụng, ví dụ: nó có thể ở dưới dạng JSON, XML, Text, v.v. Phần dữ liệu có thể không chứa dữ liệu.

### **Gói tin PUBACK**

Gói tin PUBACK được sử dụng bởi bên nhận để phản hồi lại gói tin PUBLISH. Trong đó, trường định danh gói tin (MessageID) là định danh của gói tin PUBLISH tương ứng. Gói tin PUBACK không chứa phần dữ liệu.

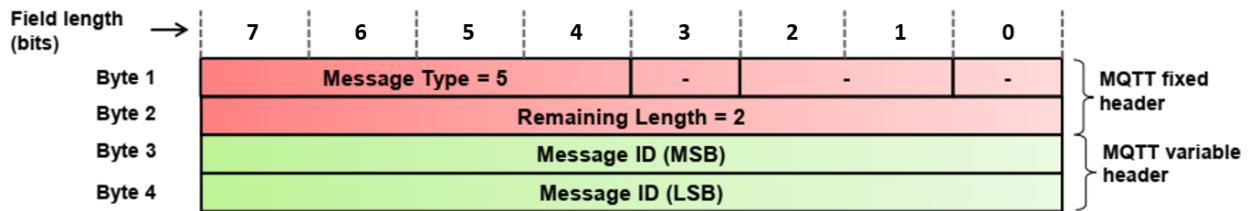


Hình 4. 26: Gói tin PUBACK.

### **Gói tin PUBREC**

Gói tin PUBREC được sử dụng bởi bên nhận để phản hồi lại gói tin PUBLISH có QoS mức 2. Trong đó, trường định danh gói tin (MessageID) là định danh của gói tin

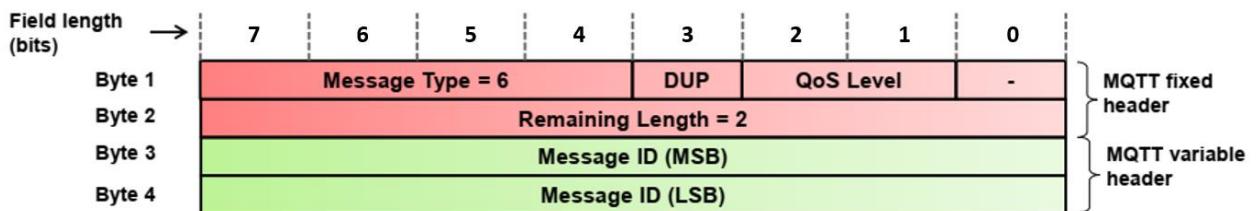
PUBLISH tương ứng. Tương tự gói tin PUBACK, gói tin PUBREC cũng không có phần dữ liệu.



Hình 4. 27: Gói tin PUBREC.

### Gói tin PUBREL

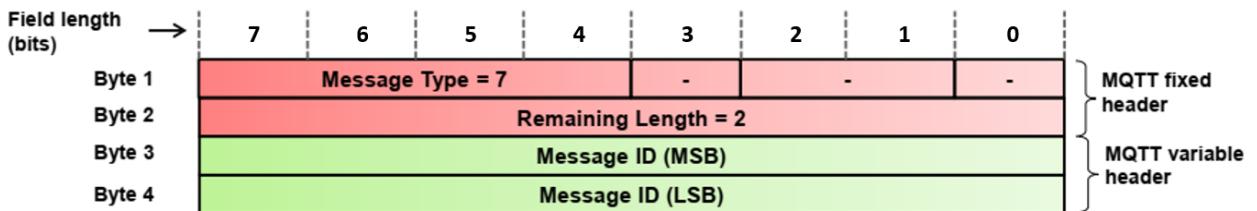
Gói tin PUBREL được sử dụng bởi bên gửi để phản hồi lại gói tin PUBREC. Nó là gói tin thứ 3 trong quy trình xuất bản tin có QoS mức 2.



Hình 4. 28: Gói tin PUBREL.

### Gói tin PUBCOMP

Gói tin PUBCOMP được sử dụng bởi bên nhận để phản hồi gói tin PUBREL. Nó là gói tin thứ 4 trong quá trình xuất bản tin với QoS mức 2.



Hình 4. 29: Gói tin PUBCOMP.

### Gói tin SUBSCRIBE

Gói tin SUBSCRIBE được gửi từ Client đến Broker để tạo một hoặc nhiều đăng ký theo dõi. Mỗi đăng ký khai báo một hoặc nhiều chủ đề mà Client quan tâm. Broker gửi các

gói tin PUBLISH đến Client để chuyển phát các bản tin được xuất bản đến các chủ đề được khai báo trong đăng kí. Gói tin cũng phải khai báo mức QoS tối đa mà Broker có thể gửi các bản tin đến Client.

Phần tiêu đề có độ dài cố định:

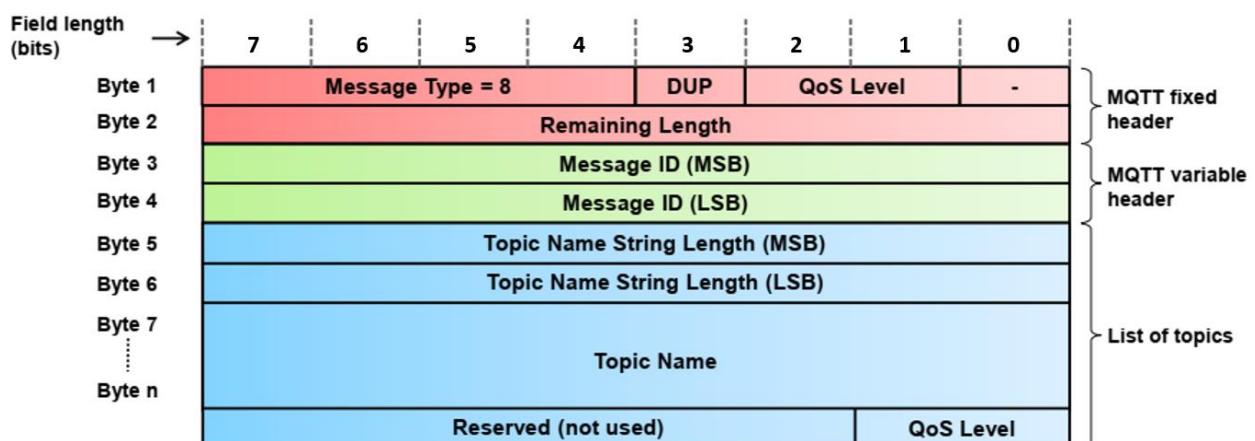
- Trường loại gói tin: được khai báo bởi 4 bit ( $7 \rightarrow 4$ ) trong byte 1 có giá trị 0x08.
- Trường độ dài còn lại: có giá trị bằng tổng độ dài của phần tiêu đề có độ dài thay đổi (2 bytes) và phần dữ liệu.

Phần tiêu đề có độ dài thay đổi:

- Phần này cung cấp thông tin về định danh của gói tin SUBSCRIBE (2 bytes).

Phần dữ liệu:

- Phần này chứa một danh sách các bộ lọc chủ đề cho biết các chủ đề mà Client muốn đăng kí theo dõi. Các bộ lọc chủ đề được mã hoá UTF-8. Broker nên hỗ trợ các bộ lọc chủ đề chứa các ký tự Wildcard nhưng nếu nó chọn không hỗ trợ thì tất cả các đăng kí có bộ lọc chủ đề chứa các ký tự này sẽ bị từ chối. Sau mỗi bộ lọc là một byte khai báo mức QoS cao nhất mà Client yêu cầu cho các bản tin được gửi từ Broker đến Client.
- Phần dữ liệu của gói tin SUBSCRIBE phải chứa ít nhất một cặp Bộ lọc chủ đề/QoS.



Hình 4. 30: Gói tin SUBSCRIBE.

## **Tóm tắt hoạt động của gói tin SUBSCRIBE**

Khi Broker nhận gói tin SUBSCRIBE từ Client, nó phải phản hồi một gói tin SUBACK có định danh gói tin trùng với định danh gói tin của gói tin SUBSCRIBE tương ứng. Broker được phép gửi các gói tin PUBLISH phù hợp với đăng ký đã được xác nhận trước cả khi gửi bản tin SUBACK.

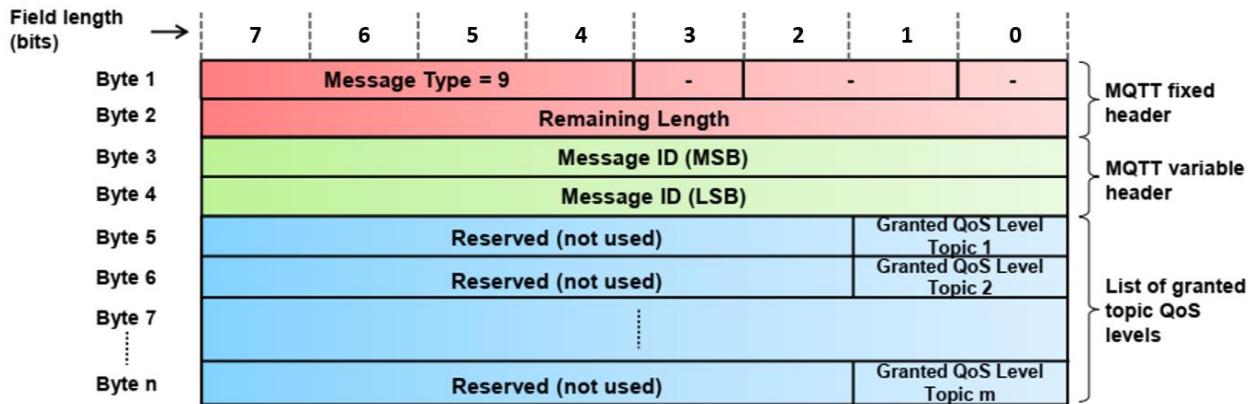
Nếu Broker nhận gói tin SUBSCRIBE chứa một bộ lọc chủ đề giống hệt bộ lọc chủ đề trong một đăng ký đã tồn tại thì nó phải thay thế đăng ký đã tồn tại đó bằng một đăng ký mới. Bộ lọc chủ đề trong đăng ký mới sẽ giống hệt trong đăng ký trước đó, nhưng mức QoS cao nhất có thể khác nhau. Ngược lại, nếu bộ lọc chủ đề không giống với bất kỳ bộ lọc chủ đề nào trong các đăng ký đã tồn tại thì một đăng ký mới được tạo và tất cả các bản tin Retain trong chủ đề đó.

Nếu Broker nhận được một gói tin SUBSCRIBE chứa nhiều bộ lọc chủ đề thì nó phải xử lý gói tin đó như thể nó nhận được một dãy các gói tin SUBSCRIBE. Tuy nhiên, nó chỉ cần phải phản hồi duy nhất một bản tin SUBACK để xác nhận chúng.

Gói tin SUBACK được gửi từ Broker đến Client phải chứa mã phản hồi cho mỗi cặp Bộ lọc chủ đề/QoS. Mã phản hồi phải khai báo mức QoS tối đa được cấp cho đăng ký tương ứng với gói tin SUBSCRIBE hoặc báo việc đăng ký không thành công. Broker được phép cung cấp mức QoS thấp hơn so với mức QoS mà Subscriber yêu cầu. Các bản tin trong gói tin PUBLISH được gửi từ Broker đến Subscriber tương ứng với đăng ký theo dõi mà Subscriber đã thực hiện có mức QoS là giá trị tối thiểu trong 2 giá trị sau: mức QoS của gói tin PUBLISH ban đầu (được gửi bởi Publisher đến Broker) và mức QoS tối đa do Broker cung cấp.

## **Gói tin SUBACK**

Gói tin SUBACK được gửi bởi Broker đến Client để xác nhận việc nhận và xử lý gói tin SUBSCRIBE. Nó phải chứa mã phản hồi để chỉ định mức QoS tối đa được cấp phép cho mỗi đăng ký bởi Broker.



Hình 4. 31: Gói tin SUBACK.

Phần tiêu đề có độ dài cố định:

- Trường loại gói tin: được khai báo bởi 4 bit ( $7 \rightarrow 4$ ) trong byte 1 có giá trị 0x09.
- Trường độ dài còn lại: có giá trị bằng tổng độ dài của phần tiêu đề có độ dài thay đổi (2 bytes) và phần dữ liệu.

Phần tiêu đề có độ dài thay đổi:

- Phần này cung cấp thông tin về định danh của gói tin SUBACK giống với định danh của gói tin SUBSCRIBE tương ứng (2 bytes).

Phần dữ liệu:

- Phần này chứa một danh sách các mã phản hồi, mỗi mã này tương ứng với một bộ lọc chủ đề trong gói tin SUBSCRIBE tương ứng. Thứ tự của các mã phản hồi phải khớp với thứ tự của các bộ lọc chủ đề trong gói tin SUBSCRIBE.

Bảng 4.6. Mã phản hồi trong gói tin SUBACK.

Mã phản hồi	Trạng thái	Mô tả
0x00	Thành công	Mức QoS tối đa = 0
0x01	Thành công	Mức QoS tối đa = 1
0x02	Thành công	Mức QoS tối đa = 2
0x80	Lỗi	

## **Gói tin UNSUBSCRIBE**

Gói tin UNSUBSCRIBE được gửi bởi Client đến Broker để huỷ đăng ký theo dõi tới các chủ đề.

Phần tiêu đề có độ dài cố định:

- Trường loại gói tin: được khai báo bởi 4 bit ( $7 \rightarrow 4$ ) trong byte 1 có giá trị 0x0A.
- Trường độ dài còn lại: có giá trị bằng tổng độ dài của phần tiêu đề có độ dài thay đổi (2 bytes) và phần dữ liệu.

Phần tiêu đề có độ dài thay đổi:

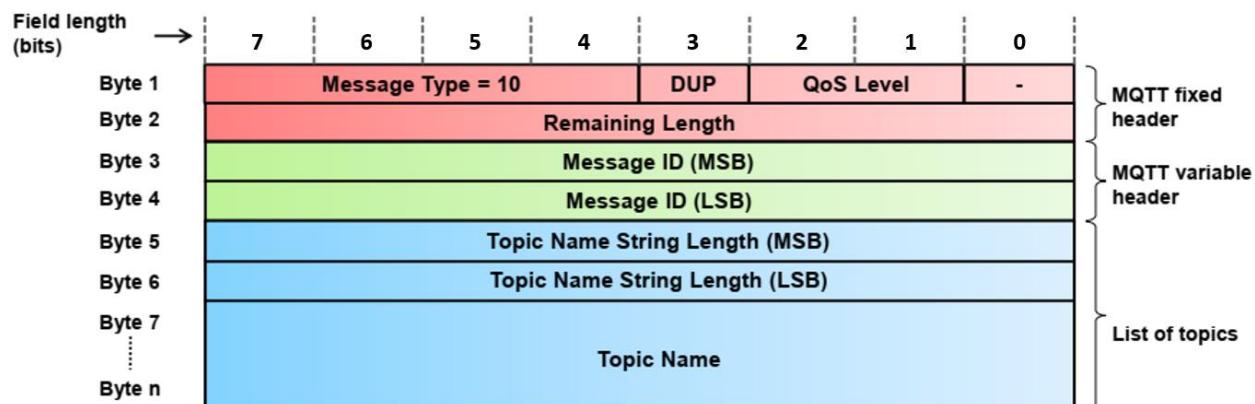
- Phần này cung cấp thông tin về định danh của gói tin (2 bytes).

Phần dữ liệu:

- Phần này chứa một danh sách các bộ lọc chủ đề (chứa ít nhất một chủ đề) mà Client muốn huỷ đăng ký.
- Các bộ lọc chủ đề phải được mã hoá UTF-8.

## **Tóm tắt các hoạt động của gói tin UNSUBSCRIBE**

Các bộ lọc chủ đề (dù chứa kí hiệu Wildcard hay không) trong gói tin UNSUBSCRIBE phải chính xác từng kí tự với các bộ lọc chủ đề được quản lý bởi Broker. Mỗi bộ lọc chủ đề khớp chính xác thì đăng ký tương ứng với nó sẽ bị xoá, ngược lại thì Broker sẽ không làm gì.



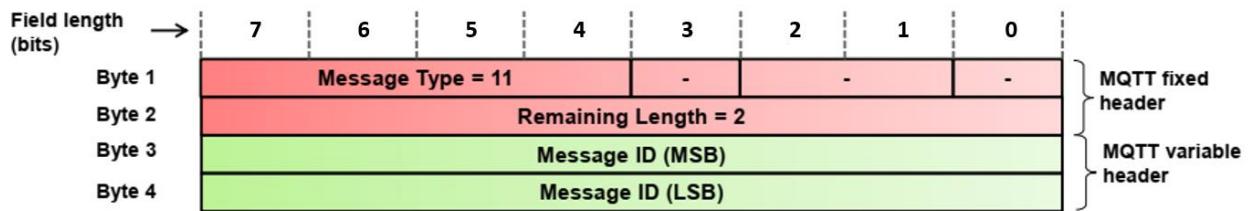
Hình 4. 32: Gói tin UNSUBSCRIBE.

Khi Broker xoá đăng kí:

- Nó phải dừng việc chuyển phát các bản tin đến Client.
- Nó phải hoàn thành việc chuyển phát các bản tin có QoS mức 1 hoặc 2 mà nó đã bắt đầu quá trình chuyển phát từ trước đó.
- Nó có thể tiếp tục gửi các bản tin đang tồn tại trong bộ nhớ đệm đến Client.

### **Gói tin UNSUBACK**

Gói tin UNSUBACK được gửi bởi Broker đến Client để xác nhận việc nhận gói tin UNSUBSCRIBE.



Hình 4. 33: Gói tin UNSUBACK.

### **Gói tin PINGREQ, PINGRESP, DISCONNECT**

Gói tin PINGREQ được gửi bởi Client đến Broker để:

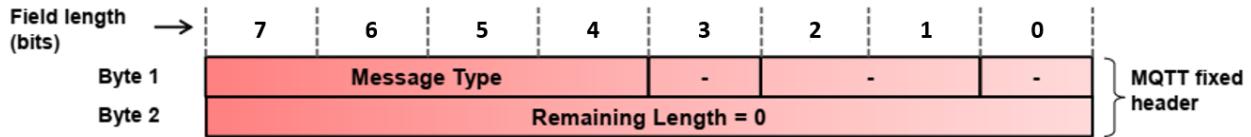
- Cho Broker biết Client vẫn hoạt động trong trường hợp Client không gửi bất kỳ gói tin điều khiển nào khác đến Broker.
- Yêu cầu Broker xác nhận nó vẫn còn hoạt động.
- Hoặc đơn giản chỉ sử dụng mạng để xác nhận mạng vẫn đang hoạt động.

Gói tin PINGRESP được gửi bởi Broker đến Client để:

- Cho Client biết Broker vẫn hoạt động.

Gói tin DISCONNECT được gửi bởi Client đến Broker để:

- Cho Broker biết nó đang ngắt kết nối với Broker.



Hình 4. 34: Gói tin DISCONNECT, PINGREQ, PINGRESP.

## Các quy trình hoạt động của giao thức MQTT

### *Thiết lập kết nối và đăng kí theo dõi*

Giao thức MQTT chạy trên nền TCP/IP. Cả Client và Broker đều cần sử dụng TCP/IP stack. Kết nối MQTT luôn luôn được thực hiện giữa một Client và Broker. Các Client không bao giờ kết nối trực tiếp với nhau. Để khởi tạo kết nối, Client sẽ gửi một bản tin CONNECT đến Broker. Ngược lại, khi nhận được bản tin CONNECT từ Client, Broker sẽ phản hồi một bản tin CONNACK chứa mã phản hồi. Một khi kết nối được thiết lập, Broker sẽ giữ cho kết nối luôn mở cho đến khi Client gửi một gói tin DISCONNECT để ngắt kết nối.

Trong nhiều trường hợp, MQTT Broker thường được đặt trên mạng Internet do đó nó có địa chỉ công cộng, trong khi đó, MQTT Client thường được đặt ở phía sau bộ định tuyến, có sử dụng tính năng NAT để đổi địa chỉ riêng thành địa chỉ công cộng. Đồng thời, Broker luôn giữ kết nối mở để cho phép việc gửi và nhận các bản tin. Do đó, việc các Client được đặt sau bộ định tuyến có thể kết nối tới Broker bình thường mà không có bất kỳ hạn chế nào.

Có 2 quy trình thiết lập phiên MQTT và đăng kí theo dõi:

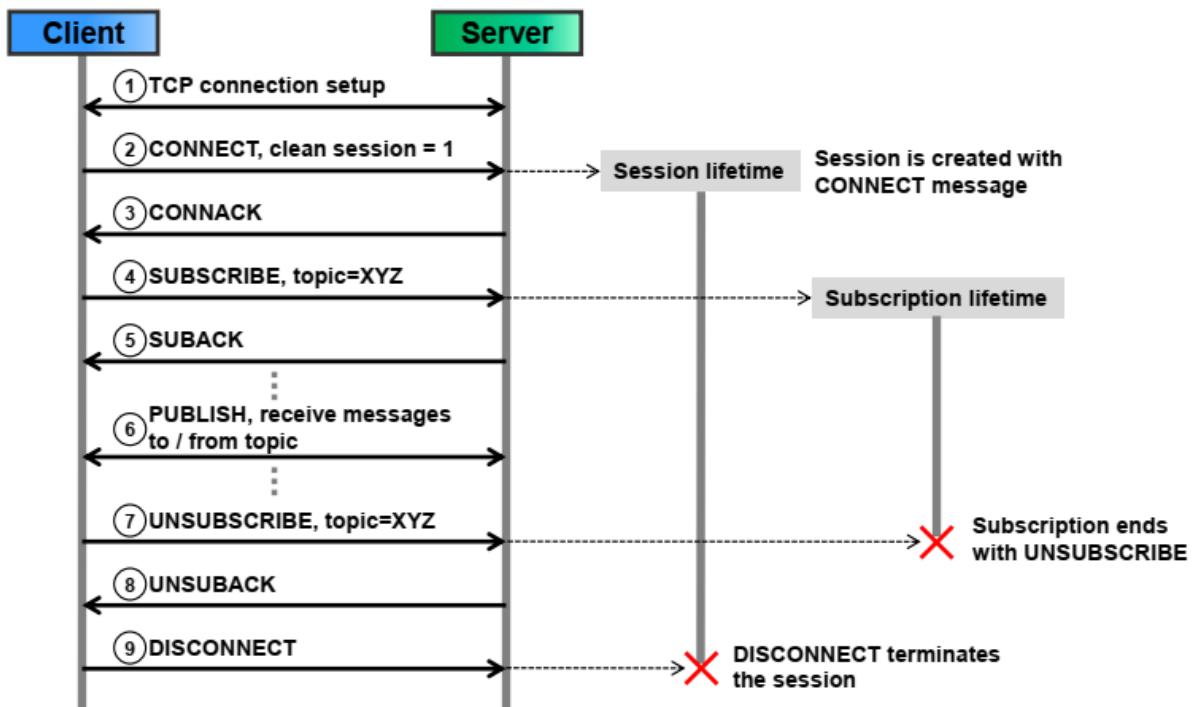
- Đăng kí ngắn hạn, tương ứng với cờ *cleanSession* = 1.
- Đăng kí dài hạn, tương ứng với cờ *cleanSession* = 0.

*Quy trình thiết lập phiên MQTT và đăng kí theo dõi với cờ cleanSession = 1*

Quy trình thiết lập phiên MQTT và đăng kí với cờ *cleanSession* = 1 (đăng kí ngắn hạn) bao gồm các bước sau (Hình 1.26):

- Client và Broker thiết lập kết nối qua giao thức TCP.
- Client gửi gói tin CONNECT yêu cầu kết nối đến Broker với cờ *cleanSession* = 1. Đây là thời điểm đánh dấu phiên MQTT được thiết lập.

- Broker gửi gói tin CONNACK để xác nhận việc thiết lập kết nối thành công với mã phản hồi là 0x00, hoặc thông báo việc kết nối không thành công với mã phản hồi khác không.
- Nếu Client nhận được bản tin CONNACK với mã phản hồi là 0x00, nó sẽ thực hiện việc đăng ký theo dõi một chủ đề nào đó mà nó quan tâm (ví dụ, chủ đề “XYZ”) bằng việc gửi gói tin SUBSCRIBE đến Broker. Đây là thời điểm đánh dấu thời gian bắt đầu của quá trình theo dõi chủ đề “XYZ”.
- Broker gửi gói tin SUBACK để xác nhận quá trình đăng ký thành công.
- Client có thể bắt đầu gửi các bản tin đến chủ đề “XYZ” bằng cách gửi các gói tin PUBLISH, hoặc nó cũng có thể nhận được các bản tin từ chủ đề “XYZ”.
- Sau khi trao đổi thông tin xong với chủ đề “XYZ”, Client sẽ gửi gói tin UNSUBSCRIBE đến Broker để yêu cầu kết thúc quá trình theo dõi chủ đề này.
- Broker sẽ trả về gói tin UNSUBACK tương ứng với gói tin UNSUBSCRIBE.
- Client sẽ gửi gói tin DISCONNECT để kết thúc một phiên MQTT.

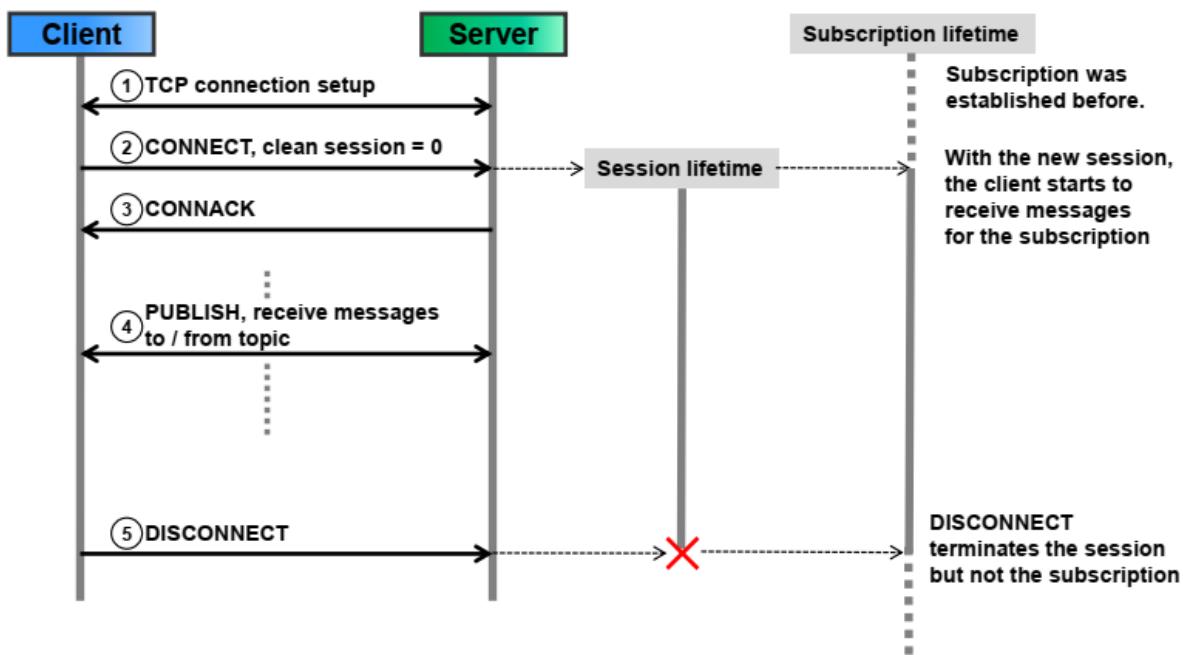


Hình 4. 35: Quy trình thiết lập phiên MQTT và đăng ký theo dõi với cờ `cleanSession = 1`.

Quy trình thiết lập phiên MQTT và đăng ký với cờ `cleanSession = 0`

Quy trình thiết lập phiên MQTT và đăng kí với cờ cleanSession = 0 (đăng kí dài hạn) bao gồm các bước sau (Hình 1.27):

- Quá trình đăng kí theo dõi chủ đề “XYZ” được thiết lập từ trước.
- Client gửi gói tin CONNECT yêu cầu kết nối đến Broker với cờ cleanSession = 0. Đây là thời điểm đánh dấu phiên MQTT được thiết lập.
- Broker gửi gói tin CONNACK để xác nhận việc thiết lập kết nối thành công với mã phản hồi là 0x00, hoặc thông báo việc kết nối không thành công với mã phản hồi khác không.
- Nếu Client nhận được bản tin CONNACK với mã phản hồi là 0x00, nó bắt đầu gửi các bản tin đến chủ đề “XYZ” bằng cách gửi các gói tin PUBLISH, hoặc nó cũng có thể nhận được các bản tin từ chủ đề “XYZ”.
- Client sẽ gửi gói tin DISCONNECT để kết thúc một phiên MQTT.



Hình 4. 36: Quy trình thiết lập phiên MQTT và đăng kí theo dõi với cờ cleanSession = 0.

### Xuất bản bản tin MQTT

Quy trình xuất bản bản tin phụ thuộc vào các mức Chất lượng dịch vụ khác nhau được khai báo trong bản tin PUBLISH. Chất lượng dịch vụ (QoS) trong giao thức MQTT được

định nghĩa là sự thoả thuận giữa phía gửi và phía nhận bản tin về sự đảm bảo trong việc chuyển phát các bản tin. Có ba mức QoS được định nghĩa trong MQTT:

- QoS mức 0: Phía gửi sẽ gửi bản tin nhiều nhất một lần.
- QoS mức 1: Phía gửi sẽ gửi bản tin ít nhất một lần.
- QoS mức 2: Phía nhận sẽ nhận bản tin đúng một lần duy nhất.

Vì mô hình MQTT chuyển phát các bản tin trong hai chặng, do đó, chúng ta sẽ xem xét QoS ở cả hai chặng:

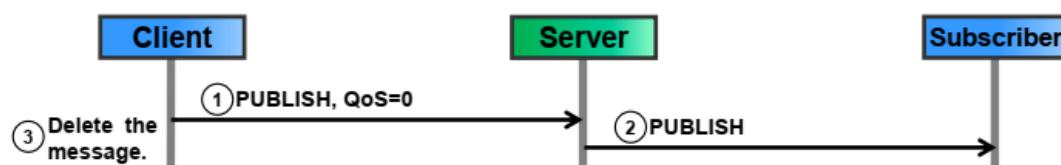
- Các bản tin được chuyển phát từ Client đóng vai trò là Publisher đến Broker.
- Các bản tin được chuyển phát từ Broker đến Client đóng vai trò là Subscriber.

Chúng ta sẽ xem xét việc chuyển phát các bản tin trong hai chặng để cập ở trên một cách riêng biệt. Client sẽ định nghĩa mức QoS cho bản tin mà nó sẽ xuất bản khi nó gửi bản tin đó đến Broker. Ngược lại, Broker sẽ gửi các bản tin đến Client đóng vai trò là Subscriber với mức QoS được định nghĩa bởi Client trong quá trình đăng ký chủ đề của Client. Trong trường hợp Subscriber định nghĩa mức QoS nhỏ hơn mức QoS mà Publisher sử dụng, thì Broker sẽ gửi bản tin đến Subscriber với mức QoS nhỏ hơn.

QoS là một trong những tính năng quan trọng của giao thức MQTT. Nó cho phép Client lựa chọn mức QoS phù hợp với độ ổn định của mạng và ứng dụng sử dụng MQTT. Do giao thức MQTT có thể quản lý việc truyền lại các bản tin bị lỗi và đảm bảo việc chuyển phát bản tin (ngay cả khi tầng vận chuyển bên dưới nó không ổn định), QoS giúp cho việc truyền thông trong các mạng không ổn định một cách dễ dàng hơn.

#### *Quy trình xuất bản với QoS = 0*

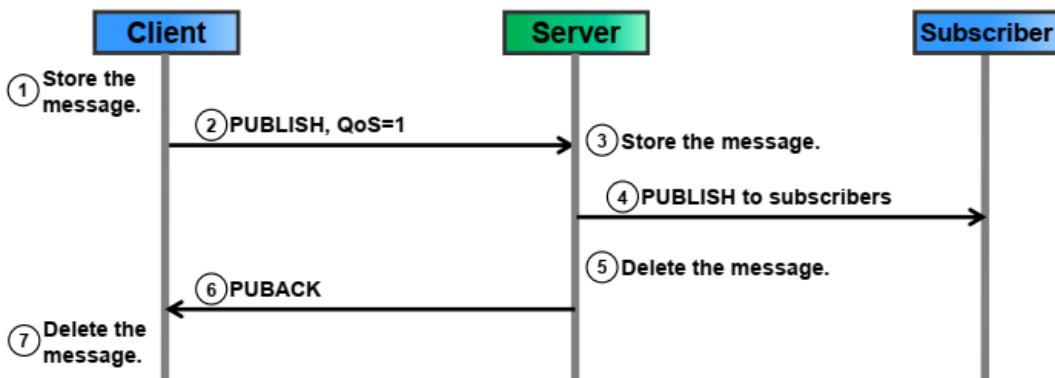
Mức QoS thấp nhất là mức 0. Ở mức QoS này, bên gửi sẽ chuyển phát bản tin trong tình trạng tốt nhất và không có sự đảm bảo việc bản tin đến được bộ nhận thành công. Bên nhận sẽ không xác nhận việc nhận bản tin và bên gửi cũng sẽ không lưu trữ và chuyển phát lại bản tin. Mức QoS này tương tự như mức độ QoS của giao thức TCP ở tầng vận chuyển.



Hình 4. 37: Quy trình xuất bản bản tin với QoS = 0.

### *Quy trình xuất bản với QoS = 1*

Mức QoS này sẽ đảm bảo bản tin được gửi đến bên nhận ít nhất một lần. Bên gửi sẽ lưu trữ bản tin cho đến khi nó nhận được bản tin PUBACK từ bộ nhận xác nhận việc nhận bản tin đó. Điều đó có nghĩa là một bản tin có thể được gửi nhiều lần ở mức QoS này.



Hình 4. 38: Quy trình xuất bản bản tin với QoS mức 1.

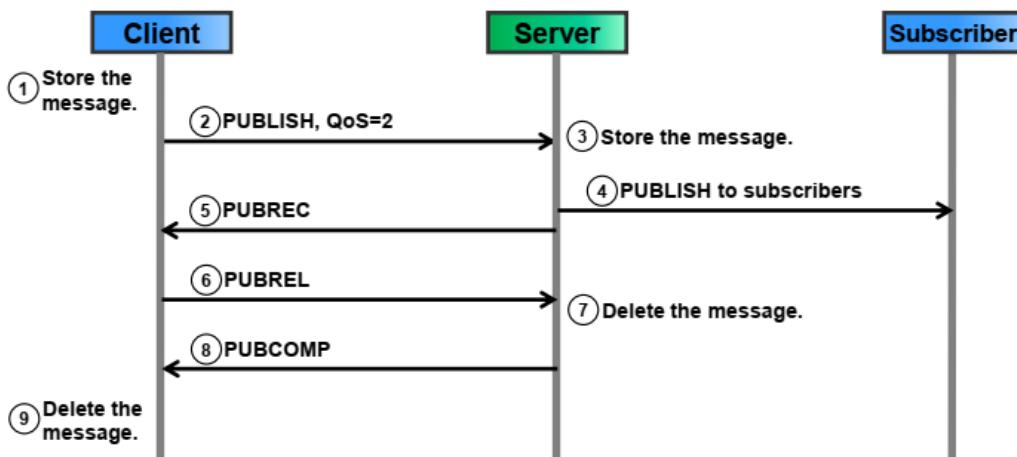
Bên gửi sẽ sử dụng định danh (ClientID) của gói tin để kiểm tra xem gói tin PUBACK có tương ứng với gói tin PUBLISH hay không. Nếu bên nhận không nhận được gói tin PUBACK trong một khoảng thời gian nhất định, thì bên gửi sẽ gửi lại bản tin PUBLISH. Khi bên nhận nhận được một bản tin với QoS mức 1, nó sẽ xử lý ngay tức thì. Ví dụ, nếu bên nhận là Broker, nó sẽ gửi ngay bản tin đến tất cả các Client đã đăng ký chủ đề tương ứng với bản tin đó, và trả lời bên gửi bằng một bản tin PUBACK.

Nếu Client đóng vai trò Publisher phải gửi lại bản tin, nó sẽ thiết lập cờ DUP. Bên nhận sẽ gửi lại gói tin PUBACK mà không quan tâm đến cờ này.

### *Quy trình xuất bản với QoS = 2*

Đây là mức QoS cao nhất trong giao thức MQTT. Mức QoS này đảm bảo mỗi bản tin chỉ được nhận một lần bởi người dùng xác định. Nó là mức QoS có độ an toàn cao nhất nhưng cũng làm cho việc chuyển phát bản tin có tốc độ chậm nhất. Để đảm bảo được chất lượng dịch vụ cao nhất thì bên nhận và bên gửi sẽ phải thực hiện ít nhất hai luồng yêu cầu/phản hồi (quy trình bắt tay 4 bước). Bên gửi và bên nhận sẽ sử dụng định danh của gói tin của bản tin PUBLISH ban đầu để phối hợp gửi tin nhắn. Khi bên nhận nhận được gói tin PUBLISH từ bên nhận, nó sẽ xử lý gói tin PUBLISH và trả lời bên gửi bằng một gói tin PUBREC để xác nhận gói tin PUBLISH. Nếu bên nhận không nhận được gói tin PUBREC

từ bên nhận, thì nó sẽ gửi lại gói tin PUBLISH với cờ DUP cho đến khi nó nhận được xác nhận từ bên nhận.



Hình 4. 39: Quy trình xuất bản bản tin với QoS mức 2.

Khi bên gửi nhận được gói tin PUBREC từ phía nhận, nó sẽ huỷ bỏ gói tin PUBLISH, đồng thời lưu trữ gói tin PUBREC và phản hồi lại bằng gói tin PUBREL.

Sau khi bên nhận nhận được gói tin PUBREL, nó sẽ loại bỏ toàn bộ trạng thái đã được lưu. Đồng thời, nó sẽ lưu thông tin định danh của gói tin PUBLISH ban đầu cho đến khi nó hoàn tất việc xử lý và gửi gói tin PUBCOMP đến bên gửi. Điều này rất quan trọng, nó giúp tránh việc gửi bản tin lần thứ hai. Sau khi bên gửi nhận được gói tin PUBCOMP, thì định danh của gói tin PUBLISH ban đầu có thể được sử dụng lại.

Khi quy trình bắt tay kết thúc, cả hai bên gửi và nhận đều chắc chắn rằng bản tin đã được gửi và bên gửi sẽ có xác nhận về quá trình gửi bản tin. Nếu một gói tin bị mất trên đường đi, bên gửi (cả MQTT Client hoặc Broker) đều phải có trách nhiệm gửi lại bản tin trong một khoảng thời gian nhất định. Bên nhận phải có trách nhiệm phản hồi lại các yêu cầu tương ứng từ bên gửi trong quy trình bắt tay.

#### *Một số điểm cần lưu ý*

Như đề cập ở trên, mức QoS được sử dụng ở hai chặng chuyển phát bản tin (từ Publisher đến Broker và từ Broker đến Subscriber) có thể khác nhau. Ví dụ, nếu A là Publisher, và B là Subscriber, A sẽ gửi bản tin đến Broker với QoS ở mức 2 mà nó đã định nghĩa trong gói tin PUBLISH. Nếu B đăng ký chủ đề chứa bản tin đó tới Broker với mức QoS là 1, thì Broker sẽ gửi bản tin đó tới B với QoS ở mức 1. Do đó, bản tin có thể được gửi đến B nhiều

lần bởi vì mức QoS 1 sẽ đảm bảo việc gửi bản tin đến bộ nhận ít nhất một lần và không quan tâm đến việc bản tin được gửi đến bộ nhận nhiều lần.

Ngoài ra định danh gói tin mà MQTT sử dụng với QoS mức 1 và 2 là duy nhất đối với mỗi cặp Client-Broker nhưng lại không duy nhất giữa tất cả các Client. Khi quy trình bắt tay giữa bên nhận và bên gửi ở QoS mức 1 và 2 kết thúc, định danh này được sử dụng lại nên số lượng định danh này không vượt quá 65535.

Tất cả các bản tin được gửi với mức QoS 1 và 2 sẽ được đưa vào hàng đợi trong trường hợp Client đang không hoạt động và được lưu cho đến khi nó hoạt động trở lại. Tuy nhiên, việc đưa các bản tin vào hàng đợi chỉ xảy ra khi Client có một phiên MQTT thường trực.

#### *Lựa chọn mức QoS phù hợp*

Việc lựa chọn mức QoS phù hợp phụ thuộc vào ứng dụng sử dụng MQTT. Dưới đây là một số gợi ý chọn các mức QoS trong các trường hợp khác nhau.

#### Mức QoS 0:

- Khi kết nối giữa bên gửi và bên nhận được thiết lập và có độ ổn định cao. Mức QoS này thường được dùng để kiểm tra kết nối đến Client hoặc kết nối một ứng dụng front-end đến MQTT Broker thông qua kết nối có dây.
- Việc bị mất một vài bản tin là chấp nhận được nếu như dữ liệu không quá quan trọng hoặc được gửi trong một thời gian ngắn.
- Không cần phải có bộ nhớ đệm để lưu các bản tin khi các Client không hoạt động như ở mức QoS 1 và 2.

#### Mức QoS 1:

- Trường hợp bên nhận cần nhận mọi bản tin và nó có khả năng xử lý các bản tin trùng nhau. Mức QoS này được sử dụng nhiều nhất bởi nó đảm bảo mỗi bản tin được gửi đến bộ nhận ít nhất một lần, chấp nhận việc phải gửi nhiều lần.
- Nếu không muốn nhận quá nhiều gói tin xử lý như ở mức QoS 2. Mức QoS 1 có tốc độ nhanh hơn QoS 2 rất nhiều.

#### Mức QoS 2:

Khi bên nhận cần nhận được mọi bản tin đúng một lần. Trong trường hợp này, việc nhận được nhiều bản tin giống nhau sẽ ảnh hưởng tới hoạt động của bên nhận. Tuy nhiên, việc sử dụng QoS mức 2 sẽ tạo ra nhiều gói tin xử lý và tốc độ cũng chậm hơn so với hai mức QoS còn lại.

#### **4.5.2 Giao thức MQTT-SN**

Gần đây, cùng với sự phát triển nhanh chóng của các ứng dụng IoT, sự quan tâm tới mạng cảm biến không dây (WSN) cũng được gia tăng cả trong lĩnh vực thương mại lẫn kỹ thuật bởi sự đơn giản, chi phí thấp và triển khai dễ dàng mạng WSN. Mạng WSN được sử dụng trong nhiều mục đích khác nhau, từ đo lường và phát hiện đến tự động hóa và kiểm soát quá trình. Một mạng WSN điển hình bao gồm một số lượng lớn các cảm biến và thiết bị chấp hành hoạt động bằng pin (SAs) có khả năng lưu trữ và xử lý hạn chế. Các mạng WSNs có đặc tính động, ví dụ, các liên kết không dây có thể tạm thời bị ngắt bất cứ lúc nào và các nút mạng có thể bị lỗi và được thay thế thường xuyên. Do đó, việc sử dụng địa chỉ để thực hiện truyền thông (ví dụ, wifi) cho một số lượng lớn các nút là không khả thi. Trong hầu hết các trường hợp, họ không cần biết địa chỉ hoặc danh tính của thiết bị truyền dữ liệu, mà họ quan tâm nhiều hơn đến nội dung của dữ liệu. Ví dụ: ứng dụng theo dõi đồ vật quan tâm nhiều hơn đến vị trí hiện tại của một đồ vật so với địa chỉ mạng của thiết bị GPS cung cấp thông tin đó. Ngoài ra, nhiều ứng dụng có thể cùng sử dụng dữ liệu cảm biến giống nhau nhưng cho các mục đích khác nhau. Trong trường hợp này, các nút SA cần quản lý và duy trì việc truyền thông với nhiều ứng dụng song song. Điều này có thể vượt quá khả năng của những thiết bị SA đơn giản có chi phí thấp với khả năng hạn chế.

Một vấn đề khác là sự khác biệt giữa các loại địa chỉ của các mạng trong cùng một hệ thống. Ví dụ: làm thế nào để một ứng dụng sử dụng mạng dựa trên TCP/IP có thể thu thập các thông tin từ các thiết bị SA chạy trên mạng không dây ZigBee?

Các vấn đề nêu trên có thể được giải quyết bằng việc sử dụng một giải pháp truyền thông coi dữ liệu là trọng tâm, trong đó việc truyền dữ liệu không dựa trên địa chỉ mạng mà dựa trên nội dung. Một trong những giải pháp truyền thông coi dữ liệu trọng tâm phổ biến là mô hình “Xuất bản/Theo dõi”. Nó được sử dụng nhiều trong các mạng doanh nghiệp bởi khả năng mở rộng và topo mạng linh hoạt của nó. Giao thức MQTT là một ví dụ điển hình cho mô hình “Xuất bản/Theo dõi”, tuy nhiên, nó phải chạy trên nền một mạng khác, ví dụ: TCP/IP, nhằm cung cấp một kết nối không mất mát (gói tin). Thế nhưng điều này quá phức

tập đối với các thiết bị đơn giản và có chi phí thấp như các thiết bị SA. Do đó, giao thức MQTT-SN sử dụng mô hình “Xuất bản/Theo dõi” được thiết kế dành riêng cho các mạng cảm biến không dây. MQTT-SN có thể được coi là biến thể của giao thức MQTT có thể giải quyết được các đặc tính riêng của các mạng cảm biến không dây vì các liên kết không dây thường có tần số lõi cao hơn các liên kết có dây bởi nhiễu fading và tốc độ truyền thông thấp. Ví dụ, mạng cảm biến không dây sử dụng tiêu chuẩn IEEE 802.15.4 có băng thông tối đa là 250 kbit/s ở dải tần số 2.4 GHz. Ngoài ra, để tránh lỗi truyền tin, các gói tin thường phải ngắn, ví dụ gói tin ở tầng vật lý trong chuẩn IEEE 802.15.4 có độ dài tối đa là 128 byte. Giao thức MQTT-SN cũng được tối ưu cho các thiết bị sử dụng pin, có chi phí thấp với khả năng xử lý và lưu trữ thấp.

Giao thức MQTT-SN được thiết kế để dùng trong tất cả các mạng mà có truyền thông 2 chiều giữa các nút mạng bất kỳ và một nút mạng đặc biệt (Gateway).

### **So sánh MQTT-SN và MQTT**

Giao thức MQTT-SN được thiết kế gần giống với giao thức MQTT nhất có thể, nhưng lại phải đáp ứng được các đặc tính riêng của mạng cảm biến không dây như băng thông thấp, khả năng lỗi liên kết cao, độ dài bản tin ngắn, v.v. MQTT-SN được tối ưu cho các thiết bị chạy bằng pin, có giá thành thấp, và khả năng lưu trữ và xử lý thấp.

Các điểm khác nhau cơ bản của giao thức MQTT-SN so với MQTT như sau:

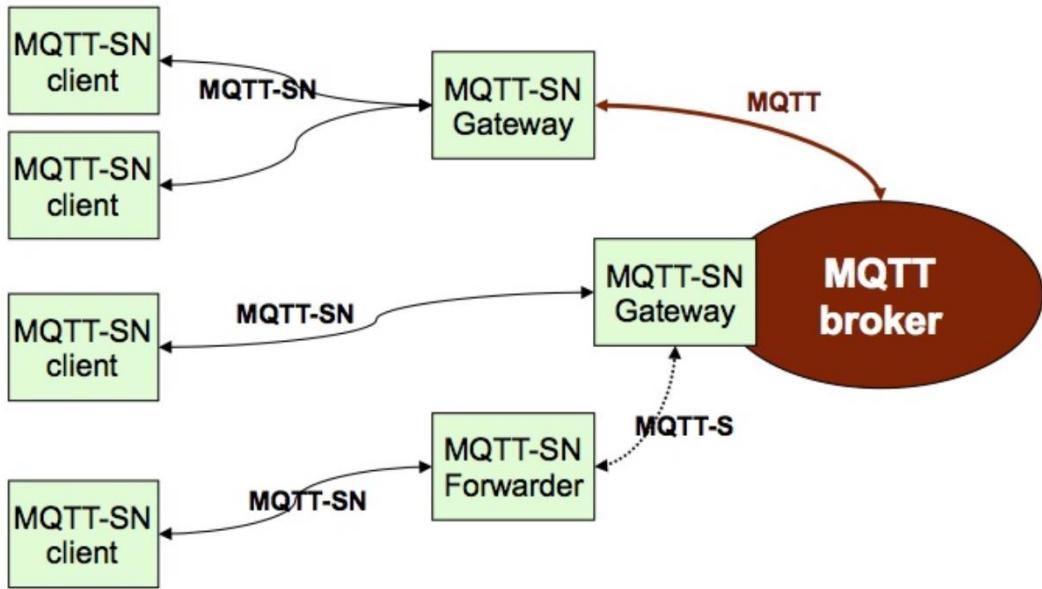
- Gói tin CONNECT được chia thành ba gói tin nhỏ hơn. Hai gói tin thêm là tùy chọn và được sử dụng để truyền chủ đề WILL và bản tin WILL đến server.
- Để giải quyết các vấn đề độ dài bản tin ngắn và băng thông giới hạn của các mạng WSN, tên chủ đề trong bản tin PUBLISH được thay thế bởi một trường “ID” của mỗi chủ đề có độ dài 2 bytes. Quy trình đăng ký cho phép Client đăng ký tên chủ đề với server hoặc gateway và nhận ID của chủ đề tương ứng.
- TopicID định sẵn, tên chủ đề rút gọn được giới thiệu trong giao thức MQTT-SN giúp cho Client không cần thực hiện khai báo tên chủ đề. TopicID định sẵn (có độ dài 2 byte) được dùng để thay thế cho tên chủ đề, và ánh xạ giữa nó và tên chủ đề được cả Client và GW biết trước. Do đó, cả Client và GW đều có thể sử dụng ngay mà không cần tiến trình khai báo như khi sử dụng TopicID thông thường.

- Tiến trình tìm kiếm cho phép các Client không được cấu hình địa chỉ server hoặc gateway có thể tìm kiếm được địa chỉ mạng thực tế của server hoặc gateway. Nhiều gateway có thể được tìm thấy trong cùng một mạng không dây cùng một lúc và có thể hợp tác với nhau trong chế độ chia sẻ tải hoặc chế độ chờ.
- Khái niệm “Clean Session” trong giao thức MQTT-SN được mở rộng cho tính năng Will.
- Để hỗ trợ tính năng “ngủ” của các Client, cơ chế Keep Alive mới được giới thiệu trong MQTT-SN. Với tiến trình này, các thiết bị vận hành bằng pin có thể chuyển sang trạng thái ngủ khi không hoạt động. Sau đó đó, nếu có bất kỳ bản tin được gửi tới chúng được lưu tạm ở server hoặc gateway, và được chuyển phát tới chúng khi chúng chuyển sang trạng thái hoạt động.

### **Kiến trúc giao thức MQTT-SN**

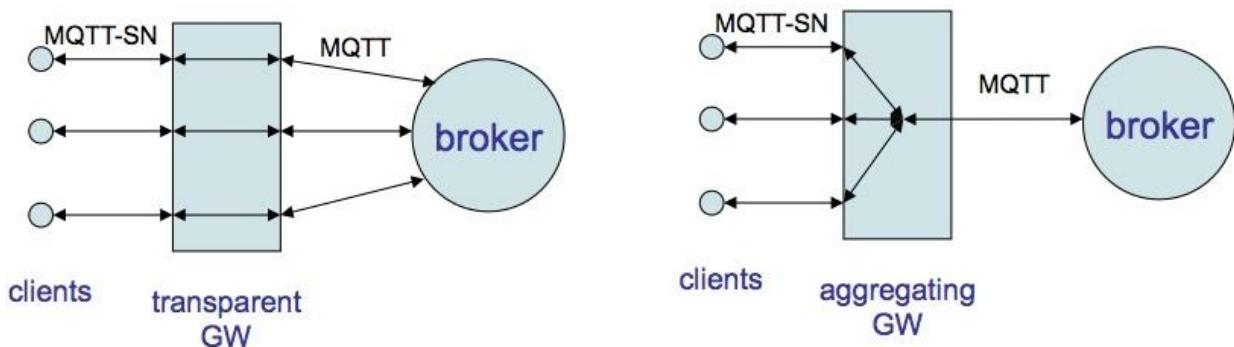
Kiến trúc của giao thức MQTT-SN được mô tả trong hình ?. Giao thức MQTT-SN bao gồm 3 thành phần: MQTT-SN Client, MQTT-SN gateway (GW), và MQTT-SN forwarder. Trong đó, các MQTT-SN Client kết nối đến MQTT server thông qua trung gian là MQTT-SN gateway sử dụng giao thức MQTT-SN. Lưu ý, một MQTT-SN GW có thể được tích hợp vào trong một MQTT server. Trong trường hợp MQTT-SN GW tách biệt, thì giao thức MQTT được sử dụng để giao tiếp giữa MQTT-SN GW và MQTT server.

Các MQTT-SN Client cũng có thể truy cập đến GW thông qua một *forwarder* khi mà GW không nằm trong mạng. *Forwarder* nhận các khung dữ liệu MQTT-SN từ các Client và chuyển tiếp đến GW mà không thay đổi dữ liệu. Ở chiều ngược lại, nó sẽ nhận các khung dữ liệu từ GW và gửi chúng đến Client mà không thay đổi dữ liệu.



Hình 4. 40: Kiến trúc giao thức MQTT-SN.

Dựa vào cách thực hiện chuyển đổi giao thức giữa MQTT-SN và MQTT của GW, chúng ta có hai loại GW khác nhau: transparent GW và aggregating GW, được mô tả sau đây:



Hình 4. 41: Transparent Gateway và Aggregating Gateway.

#### *Transparent Gateway*

Với mỗi kết nối từ MQTT-SN Client đến Transparent GW, nó sẽ thiết lập và duy trì một kết nối MQTT tương ứng đến MQTT server. Số lượng kết nối MQTT từ GW đến server sẽ tương ứng với số lượng kết nối từ Client đến GW. Transparent GW sẽ thực hiện việc biên dịch theo “cú pháp” giữa hai giao thức. Lưu ý, các Client có thể sử dụng tất cả các tính năng và đặc trưng của MQTT server.

Mặc dù việc triển khai Transparent GW đơn giản hơn Aggregating GW nhưng nó yêu cầu MQTT server hỗ trợ các kết nối tách biệt giữa các Client. Một số MQTT server chỉ hỗ trợ một số lượng giới hạn các kết nối MQTT đồng thời.

### *Aggregating Gateway*

Thay vì cung cấp một kết nối MQTT cho mỗi Client, Aggregating GW sẽ chỉ cung cấp một kết nối MQTT duy nhất đến MQTT server. Các bản tin từ gửi từ các Client được tổng hợp tại GW. Sau đó, GW sẽ lựa chọn bản tin nào sẽ được chuyển tới server. Mặc dù, việc thực hiện Aggregating GW phức tạp hơn Transparent GW, nhưng nó lại phù hợp trong trường hợp mạng WSN có số lượng SA rất lớn vì số lượng kết nối MQTT đồng thời tới server được giảm xuống.

### **Định dạng gói tin điều khiển trong MQTT-SN**

Trong giao thức MQTT-SN, một số gói tin mới được sử dụng để thực hiện các tính năng không có trong giao thức MQTT như: Quảng bá và tìm kiếm Gateway, ....

#### *Định dạng gói tin chung*

Một gói tin điều khiển trong MQTT-SN bao gồm 2 phần: tiêu đề có độ dài 2 hoặc 4 octet và phần thay đổi.

Phần tiêu đề gói tin (2 hoặc 4 octet)	Phần thay đổi của gói tin (n octets)
--	---

*Hình 4. 42: Định dạng gói tin trong MQTT-SN.*

Phần tiêu đề gói tin:

- Phần tiêu đề bao gồm 2 trường: trường khai báo độ dài của gói tin (số octets) và trường khai báo loại gói tin.
- Trường khai báo độ dài gói tin: Có độ dài là 1 hoặc 3 octet. Nếu octet đầu tiên là “0x01” thì trường này có độ dài là 3 octet, 2 octet sau chỉ định tổng số octet có trong gói tin (biểu diễn được độ dài tối đa 65535 octet). Ngược lại, trường khai báo độ dài gói tin có độ dài là 1 octet và chỉ định tổng số octet của gói tin bằng chính octet đó (biểu diễn được độ dài tối đa 256 octet).
- Trường khai báo loại gói tin: Có độ dài 1 octet và biểu diễn các loại gói tin như trong bảng sau:

**Bảng 4.7. Các loại gói tin trong giao thức MQTT-SN.**

Loại gói tin	Giá trị	Loại gói tin	Giá trị
ADVERTISE	0x00	SEARCHGW	0x01
GWINFO	0x02	Chưa sử dụng	0x03
CONNECT	0x04	CONNACK	0x05
WILLTOPICREQ	0x06	WILLTOPIC	0x07
WILLMSGREQ	0x08	WILLMSG	0x09
REGISTER	0x0A	REGACK	0x0B
PUBLISH	0x0C	PUBACK	0x0D
PUBCOMP	0x0E	PUBREC	0x0F
PUBREL	0x10	Chưa sử dụng	0x11
SUBSCRIBE	0x12	SUBACK	0x13
UNSUBSCRIBE	0x14	UNSUBACK	0x15
PINGREQ	0x16	PINGRESP	0x17
DISCONNECT	0x18	Chưa sử dụng	0x19
WILLTOPICUPD	0x1A	WILLTOPICRESP	0x1B
WILLMSGUPD	0x1C	WILLMSGRESP	0x1D
Chưa sử dụng	0x1E-0xFD	Khung dữ liệu được đóng ở Forwarder	0xFE
Chưa sử dụng	0xFF		

Phần thay đổi:

- Nội dung trong phần này phụ thuộc vào các gói tin khác nhau.
- Trường ClientID: Là một chuỗi kí tự có độ dài thay đổi từ 1 → 23 được dùng để định danh Client kết nối đến GW.
- Trường Dữ liệu: Trường này tương tự như trường dữ liệu trong gói tin PUBLISH trong giao thức MQTT. Nó có độ dài thay đổi và chứa bản tin ứng dụng.
- Trường Thời gian: có độ dài 2 octet chỉ định khoảng thời gian được tính bằng giây, có độ lớn xấp xỉ 18 giờ.

- Trường Cờ: bao gồm các cờ DUP, cờ QoS, cờ Retain, cờ Will, cờ cleanSession, cờ khai báo loại TopicID. Trong giao thức MQTT-SN có thêm một mức QoS - 1 được khai báo bởi cờ QoS có giá trị 0b11. Ngoài ra, trường khai báo loại TopicID để xác định thay thế cho tên chủ đề: 0b00 (TopicID), 0b01 (TopicID định sẵn), 0b10 (tên chủ đề rút gọn), 0b11 (chưa sử dụng).
- Trường GwAdd: Chứa địa chỉ của một GW, và có độ dài thay đổi phụ thuộc vào mạng sử dụng MQTT-SN. Ví dụ, trong mạng Zigbee, địa chỉ mạng có độ dài 2 octet.
- Trường GwId: Có độ dài 1 octet chứa ID của GW.
- Trường MsgId: Giống trường MessageID trong giao thức MQTT và có độ dài 2 octet để định danh bản tin.
- Trường ProtocolId: Có độ dài 1 octet và chỉ xuất hiện trong gói tin CONNECT để biểu diễn phiên bản của giao thức.
- Trường Bán kính: Có độ dài 1 octet, xác định bán kính của bản tin quảng bá. Giá trị 0x00 ám chỉ rằng bản tin sẽ được quảng bá tới tất cả các nút trong mạng.
- Trường Mã phản hồi: Có độ dài 1 octet.

**Bảng 4.8. Mã phản hồi.**

Mã phản hồi	Mô tả
0x00	Chấp nhận
0x01	Tù chối: Tắc nghẽn
0x02	Tù chối: Sai TopicID
0x03	Tù chối: Không được hỗ trợ
0x04-0xFF	Chưa sử dụng

- Trường TopicID: Chứa giá trị của TopicID, có độ dài 2 octet. Các giá trị 0x0000 và 0xFFFF chưa được sử dụng.
- Trường Tên chủ đề: Chứa tên chủ đề được mã hoá theo mã UTF-8 có độ dài không cố định.
- Trường WillMsg: Chứa bản tin Will và có độ dài không cố định.
- Trường WillTopic: Chứa tên chủ đề Will và có độ dài không cố định

### **Một số tính năng của MQTT-SN:**

*Quảng bá và tìm kiếm Gateway*

Tính năng này mới và không có trong giao thức MQTT.

Một GW có thể thông báo sự hiện diện của nó bằng cách quảng bá gói tin ADVERTISE một cách định kỳ đến tất cả các thiết bị trong mạng. Các GW có thể hoạt động trong cùng một mạng ở cùng thời điểm bằng cách gán các ID khác nhau. Các Client có thể lựa chọn kết nối đến một GW bất kỳ, và chỉ có thể kết nối tới duy nhất một GW tại một thời điểm. Mỗi Client nên duy trì một danh sách các GW đang hoạt động cùng với địa chỉ mạng của chúng. Các thông tin này được lấy từ các gói tin ADVERTISE và GWINFO.

Khoảng thời gian  $T_{ADV}$  là khoảng thời gian giữa hai lần GW gửi gói tin ADVERTISE. Nó được Client dùng để quản lý tính sẵn sàng của GW.

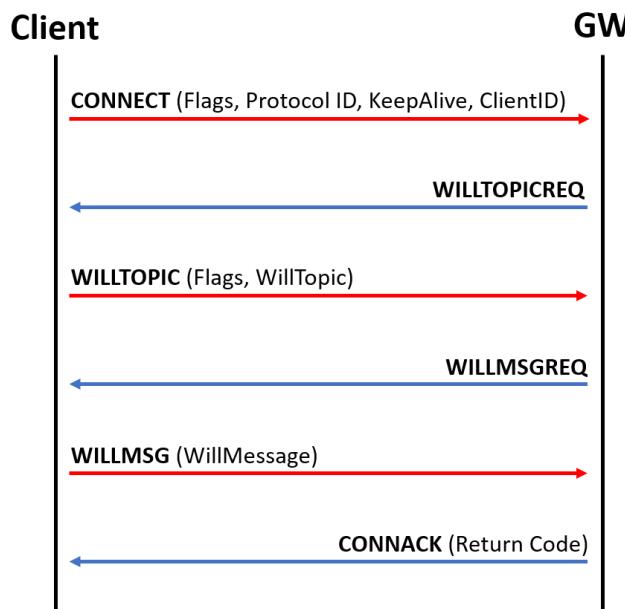
Vì các gói tin ADVERTISE được quảng bá trong toàn mạng nên khoảng thời gian  $T_{ADV}$  phải đủ lớn để tránh làm tắc nghẽn mạng (ví dụ,  $T_{ADV} > 15$  phút). Ngược lại, nếu thời gian này quá lớn thì dẫn tới thời gian chờ quá lâu cho các Client mới trong quá trình tìm kiếm GW. Để giảm thời gian chờ đợi này thì Client có thể quảng bá một gói tin SEARCHGW. Việc này dẫn tới cơn bão các gói tin quảng bá từ nhiều Client muốn tìm kiếm GW trong cùng một thời điểm. Để ngăn ngừa việc này thì việc gửi gói tin SEARCHGW sẽ bị tạm hoãn trong một khoảng thời gian ngẫu nhiên từ 0 đến  $T_{SEARCH}$ . Một Client có thể huỷ bỏ việc truyền gói tin SEARCHGW của nó khi nó nhận được một gói tin SEARCHGW từ một Client khác giống hệt của nó, và nó sẽ coi như gói tin SEARCHGW đó được gửi bởi chính nó. Ngoài ra, bán kính quảng bá  $R_b$  của gói tin SEARCHGW cũng được giới hạn, ví dụ, trong trường hợp có nhiều Client thì bán kính này chỉ là một chặng (hop).

Khi nhận được gói tin SEARCHGW nhận được, GW sẽ trả lời bằng cách gửi một gói tin GWINFO chứa ID của nó. Tương tự, nếu Client nhận được gói tin SEARCHGW, nó cũng sẽ phản hồi bằng một gói tin GWINFO chứa ít nhất một GW đang hoạt động trong danh sách của nó cùng với địa chỉ GW đó. Gói tin GWINFO cũng được quảng bá với bán kính  $R_b$  giống với bán kính quảng bá của gói tin SEARCHGW. Để ưu tiên cho các GW, Client sẽ trì hoãn việc gửi gói tin GWINFO trong một khoảng thời gian ngẫu nhiên  $T_{GWINFO}$ . Nếu nó nhận được một gói tin GWINFO trong khoảng thời gian này thì nó sẽ huỷ bỏ việc gửi gói tin GWINFO của nó. Trong trường hợp không có phản hồi cho gói tin SEARCHGW, thì gói tin này được gửi lại. Khi đó, khoảng thời gian giữa hai lần gửi gói tin SEARCHGW nên tăng theo cấp số nhân.

### *Thiết lập kết nối phía Client*

Cũng giống như giao thức MQTT, một Client trong giao thức MQTT-SN cũng phải thiết lập một kết nối đến một GW trước khi nó bắt đầu trao đổi với GW đó. Tiến trình thiết lập kết nối đến GW được minh họa trong hình 1.35, được bắt đầu bằng việc Client gửi gói tin CONNECT với thiết lập cờ WILL để yêu cầu GW gửi chủ đề WILL và bản tin WILL. Client sẽ gửi các thông tin này cho GW sau khi nhận được các yêu cầu tương ứng từ GW. Tiến trình này được kết thúc bằng gói tin CONNACK được gửi bởi GW. Nếu cờ WILL không được thiết lập thì GW sẽ kết thúc ngay tiến trình kết nối bằng gói tin CONNACK.

Trong trường hợp GW không chấp nhận yêu cầu kết nối từ Client, thì gói tin phản hồi CONNACK sẽ chứa lý do từ chối.



Hình 4. 43: Tiến trình Thiết lập kết nối từ phía Client.

### *Clean session*

Trong giao thức MQTT, khi một Client ngắt kết nối đến Broker, các đăng ký đến các chủ đề của nó không bị xoá và sẵn sàng cho các kết nối mới cho đến khi Client huỷ đăng ký hoặc thiết lập kết nối mới với cờ cleanSession được thiết lập là 1.

Trong giao thức MQTT-SN, khái niệm cleanSession được mở rộng cho cả tính năng Will, không chỉ có đăng ký theo dõi là liên tục mà còn cả chủ đề Will và bản tin Will. Hai cờ cleanSession và Will trong gói tin CONNECT được thiết lập như sau:

- Nếu cờ cleanSession = 1, Will = 1: GW sẽ xoá tất cả các đăng ký và dữ liệu về bản tin Will liên quan đến Client và bắt đầu yêu cầu chủ đề và bản tin Will mới.
- Nếu cờ cleanSession = 1, Will = 0: GW sẽ xoá tất cả các đăng ký và dữ liệu về bản tin Will liên quan đến Client và phản hồi bằng gói tin CONNACK (mà không yêu cầu chủ đề và bản tin Will mới).
- Nếu cờ cleanSession = 0, Will = 1: GW sẽ lưu tất cả dữ liệu của Client, nhưng yêu cầu chủ đề và bản tin Will mới và ghi đè lên chủ đề và bản tin Will hiện tại.
- Nếu cờ cleanSession = 0, Will = 0: GW sẽ lưu tất cả dữ liệu của Client và phản hồi bằng gói tin CONNACK (mà không yêu cầu chủ đề và bản tin Will mới).

Lưu ý rằng, nếu Client chỉ muốn xoá dữ liệu Will của nó trong tiến trình thiết lập kết nối, nó chỉ cần gửi gói tin CONNECT với cờ cleanSession = 0, Will = 1, và gửi một bản tin WILLTOPIC rỗng đến GW. Nó cũng có thể gửi gói tin CONNECT với cờ cleanSession = 0, Will = 0, và sử dụng tiến trình trong mục 1.6.5.4 để xoá hoặc sửa dữ liệu Will.

### **Tiến trình cập nhật dữ liệu bản tin WILL**

Client có thể cập nhật dữ liệu Will được lưu ở GW tại bất kỳ lúc nào bằng cách gửi các gói tin TOPICUPD và WILLMSGUPD. Thông tin chứa trong hai gói tin này sẽ được ghi đè lên các thông tin đã được lưu ở GW. Cả 2 gói tin này độc lập với nhau và đều được xác nhận bởi GW. Lưu ý rằng, gói tin rỗng WILLTOPICUPD có thể xoá cả chủ đề và bản tin Will được lưu ở GW.

### **Tiến trình khai báo tên chủ đề**

Vì các mạng truyền thông có băng thông thấp và các bản tin trao đổi có dữ liệu nhỏ như mạng cảm biến không dây, nên các bản tin không được xuất bản cùng với tên chủ đề chứa các bản tin đó như trong giao thức MQTT. Do đó, một tiến trình khai báo tên chủ đề được giới thiệu trong MQTT-SN cho phép cả GW và Client thông báo cho đối phương về định danh chủ đề (Topic ID) và tên chủ đề tương ứng trước khi bắt đầu gửi gói tin PUBLISH sử dụng định danh chủ đề.

Để khai báo một tên chủ đề, Client phải gửi một gói tin REGISTER đến GW. Nếu khai báo được chấp nhận, GW sẽ gán một TopicID cho tên chủ đề trong gói tin REGISTER và sau đó phản hồi thông tin đó đến Client bằng gói tin REGACK. Ngược lại, nếu khai báo không được chấp nhận, GW cũng sẽ phản hồi bằng một gói tin REGACK với lý do từ chối được mã hoá bằng trường mã phản hồi.

Sau khi nhận gói tin REGACK thông báo việc chấp thuận từ GW, Client sẽ được sử dụng TopicID đó để xuất bản các bản tin thuộc chủ đề tương ứng. Ngược lại, nếu gói tin REGACK thông báo việc khai báo thất bại thì Client có thể gửi lại. Đặc biệt nếu mã phản hồi là “Từ chối: Do tắc nghẽn”, Client nên chờ một khoảng thời gian  $T_{WAIT}$  trước khi gửi lại gói tin REGISTER.

Tại một thời điểm, Client chỉ có thể có thể gửi duy nhất một gói tin REGISTER và nó phải chờ cho tới khi nhận được gói tin REGACK trước khi khai báo một tên chủ đề khác. Ngược lại, khi Client thực hiện kết nối lại đến GW với cờ cleanSession = 0 hoặc khi Client đăng ký theo dõi các chủ đề có tên chứa các kí tự Wildcard (#, +), GW phải gửi gói tin REGISTER đến Client nếu nó muốn thông báo cho Client về tên chủ đề và TopicID mà nó gán cho tên chủ đề đó trước khi xuất bản các bản tin ứng dụng với TopicID đó tới Client.

### **Tiến trình xuất bản của Client**

Sau khi khai báo thành công một tên chủ đề với GW, Client có thể bắt đầu xuất bản các bản tin ứng dụng đến GW bằng cách gửi gói tin PUBLISH với TopicID tương ứng. Cả ba mức QoS và các cơ chế bắt tay khi xuất bản các bản tin đều được hỗ trợ như trong giao thức MQTT. Sự khác biệt duy nhất là trường TopicID được dùng để thay thế cho tên chủ đề ở trong bản tin PUBLISH. Bất kể mức QoS nào, thì Client đều nhận được gói tin PUBACK để phản hồi lại việc từ chối gói tin PUBLISH của nó với mã phản hồi:

- Nếu mã phản hồi là “Từ chối: Sai TopicID”: Trong trường hợp này, Client cần khai báo lại tên chủ đề trước khi nó xuất bản lại bản tin đến tên chủ đề đó, hoặc
- Nếu mã phản hồi là “Từ chối: Tắc nghẽn”: Trong trường hợp này, Client sẽ dừng việc xuất bản bản tin đến GW ít nhất trong khoảng thời gian  $T_{WAIT}$ .

Tại một thời điểm, Client chỉ có thể gửi duy nhất một bản tin có QoS mức 1 hoặc 2. Do đó, nó phải chờ cho đến khi kết thúc tiến trình trao đổi gói tin PUBLISH trước khi nó thực hiện tiến trình trao đổi gói tin PUBLISH có QoS mức 1 hoặc 2 tiếp theo.

### **TopicID định sẵn và tên chủ đề rút gọn**

Như được đề cập trong mục 1.6.5.5, một TopicID (có độ dài 2 byte) được dùng để thay thế cho tên chủ đề (biểu diễn dưới dạng string). Một phương pháp thay thế khác được sử dụng là TopicID định sẵn mà cả Client và GW đều biết trước. Trường này được khai báo bởi các cờ trong gói tin PUBLISH. Khi sử dụng TopicID định sẵn, cả hai bên có thể bắt đầu gửi gói tin PUBLISH ngay lập tức mà không cần tiến trình khai báo tên chủ đề như TopicID thông thường. Nếu nhận được một gói tin PUBLISH với TopicID định sẵn mà không khớp với tên chủ đề, người nhận sẽ phản hồi bằng một bản tin PUBACK với mã phản hồi “Từ chối: Sai TopicID”. Lưu ý là lỗi này không thể giải quyết được bằng cách khai báo lại như trường hợp sử dụng TopicID thông thường. Client (trong vai trò Subscriber) vẫn phải đăng ký theo dõi một TopicID định sẵn nếu nó muốn nhận các gói tin PUBLISH liên quan đến TopicID đó.

Một cách khác để thay thế tên chủ đề thông thường đó là sử dụng một tên chủ đề rút gọn có độ dài cố định 2 byte. Nó được gửi cùng với dữ liệu trong gói tin PUBLISH, do đó không cần tiến trình khai báo như sử dụng TopicID. Ngoài ra, tất cả các quy tắc áp dụng cho một tên chủ đề thông thường đều được áp dụng cho tên chủ đề rút gọn. Lưu ý, không sử dụng các kí tự Wildcard trong tên chủ đề rút gọn.

Để tránh nhầm lẫn giữa TopicID định sẵn và tên chủ đề rút gọn (2 byte), gói tin SUBSCRIBE phải chứa một cờ để chỉ ra rằng nó đang theo dõi một tên chủ đề rút gọn hay TopicID định sẵn.

### **Tiến trình xuất bản bản tin với mức QoS là -1**

Tính năng này cho phép Client xuất bản các bản tin ứng dụng một cách cực kỳ đơn giản mà không cần tiến trình thiết lập kết nối/huỷ kết nối, không cần tiến trình đăng ký theo dõi, không cần tiến trình khai báo. Client chỉ cần gửi gói tin PUBLISH của nó đến GW (có địa chỉ mà Client đã biết) và quên luôn sau khi gửi. Nó không quan tâm liệu địa chỉ GW có chính xác hay không, GW có đang hoạt động hay không, hoặc gói tin có đến được GW hay không. Các thông số sau được thiếp lập cho gói tin PUBLISH có mức QoS là -1:

- Cờ QoS được thiết lập là “0b11”.
- Cờ TopicIdType được thiết lập là “0b01” đối với TopicID định sẵn hoặc “0b10” đối với tên chủ đề rút gọn.

- Trường TopicID khai báo giá trị TopicID định sẵn hoặc giá trị của tên chủ đề rút gọn.
- Trường dữ liệu: bản tin ứng dụng cần xuất bản.

### **Tiến trình đăng ký và huỷ đăng ký theo dõi chủ đề của Client**

Để đăng ký theo dõi một chủ đề, Client gửi gói tin SUBSCRIBE đến GW với tên chủ đề trong gói tin đó. Nếu GW chấp nhận đăng ký này, nó sẽ gán một TopicID cho tên chủ đề nhận được và phản hồi Client bằng gói tin SUBACK. Nếu GW không chấp nhận đăng ký này, nó cũng phản hồi bằng gói tin SUBACK với lý do từ chối được mã hoá trong trường Mã phản hồi. Nếu lý do từ chối là “Từ chối: Tắc nghẽn”, Client nên chờ một khoảng thời gian TWAIT trước khi gửi lại gói tin SUBSCRIBE đến GW.

Nếu Client đăng ký theo dõi chủ đề có tên chứa kí tự Wildcard, GW sẽ phản hồi bằng gói tin SUBACK chứa TopicID = 0x0000. GW sẽ sử dụng tiến trình khai báo để thông báo cho Client về giá trị TopicID được sử dụng khi gửi gói tin PUBLISH đến Client.

Tương tự quy trình xuất bản của Client, các TopicID cũng có thể được định sẵn cho các tên chủ đề nhất định. Tên chủ đề rút gọn cũng có thể được sử dụng. Trong hai trường hợp này, Client vẫn cần đăng ký theo dõi TopicID định sẵn, hoặc tên chủ đề rút gọn.

Để huỷ đăng ký theo dõi một chủ đề, Client gửi một gói tin UNSUBSCRIBE đến GW, và GW phản hồi bằng một gói tin UNSUBACK. Đối với mỗi tiến trình REGISTER, Client chỉ có thể thực hiện một thủ tục đăng ký/huỷ đăng ký theo dõi tại một thời điểm.

### **Tiến trình xuất bản của GW**

Tương tự tiến trình xuất bản của Client được mô tả trong mục 1.6.4.6, GW gửi các gói tin PUBLISH đến Client với giá trị TopicID trong gói tin SUBACK. Trước khi gửi gói tin PUBLISH, GW gửi gói tin REGISTER để thông báo cho Client về tên chủ đề và TopicID được gán tương ứng khi Client kết nối lại mà không sử dụng tính năng clean session hoặc đăng ký theo dõi các chủ đề có tên chứa các kí tự Wildcard. Khi nhận được gói tin REGISTER, Client sẽ phản hồi bằng gói tin REGACK. GW sẽ chờ cho đến khi nhận được gói tin REGACK trước khi nó gửi các gói tin PUBLISH đến Client.

Client có thể từ chối gói tin REGISTER bằng một gói tin REGACK chứa lý do từ chối. Điều này tương ứng với việc huỷ đăng ký theo dõi tên chủ đề được khai báo trong gói tin REGISTER. Chú ý rằng việc huỷ đăng ký theo dõi với một chủ đề có tên chứa các kí tự

Wildcard có thể được thực hiện bằng tiến trình huỷ đăng ký theo dõi được trình bày trong mục 1.6.4.9 hoặc từ chối gói tin REGISTER bởi vì gói tin REGISTER không bao giờ chấp nhận tên chủ đề chứa kí tự Wildcard.

Nếu Client nhận một gói tin PUBLISH với một TopicID mà nó chưa biết, nó sẽ phản hồi với gói tin PUBACK với mã phản hồi “Từ chối: Sai TopicID”. Điều này sẽ làm cho GW thực hiện xoá hoặc chỉnh sửa lại TopicID sai mà nó đã gán. Chú ý nếu tên chủ đề hoặc dữ liệu quá dài và không phù hợp với gói tin REGISTER hoặc PUBLISH thì GW sẽ lặng lẽ huỷ bỏ quy trình xuất bản bản tin (không có cảnh báo nào đến Client).

### **Tiến trình PING và Keep Alive**

Giống như trong giao thức MQTT, Client gửi gói tin PINGREQ để kiểm tra xem GW mà nó đang kết nối đến có đang hoạt động hay không. Khi nhận được gói tin PINGREQ từ Client, GW sẽ phản hồi bằng gói tin PINGRESP để xác nhận rằng nó vẫn đang hoạt động. Nếu Client không nhận được gói tin PINGRESP từ GW mặc dù nó đã gửi nhiều gói tin PINGREQ thì Client nên thử kết nối tới GW khác trước khi thử kết nối lại với GW hiện tại (xem thêm mục 1.6.4.13). Lưu ý rằng, vì các bộ định thời của các Client không đồng bộ với nhau, do đó trong trường hợp GW bị lỗi và bị ngắt kết nối, không có khả năng xảy ra một cơn bão gói tin CONNECT được gửi đồng thời bởi các Client đang kết nối tới GW đó.

### **Tiến trình huỷ kết nối của Client**

Client gửi gói tin DISCONNECT đến GW để cho biết nó sắp đóng kết nối. Sau thời điểm này, Client phải thiết lập một kết nối mới thì mới lại có thể trao đổi dữ liệu với GW đó. Tương tự giao thức MQTT, việc gửi gói tin DISCONNECT đến GW không ảnh hưởng đến các theo dõi hiện có và bản tin WILL nếu cờ cleanSession được thiết lập. Chúng được lưu cho đến khi Client thực hiện huỷ theo dõi, xoá hoặc thay đổi hoặc Client thiết lập một kết nối mới với cờ *cleanSession* được thiết lập.

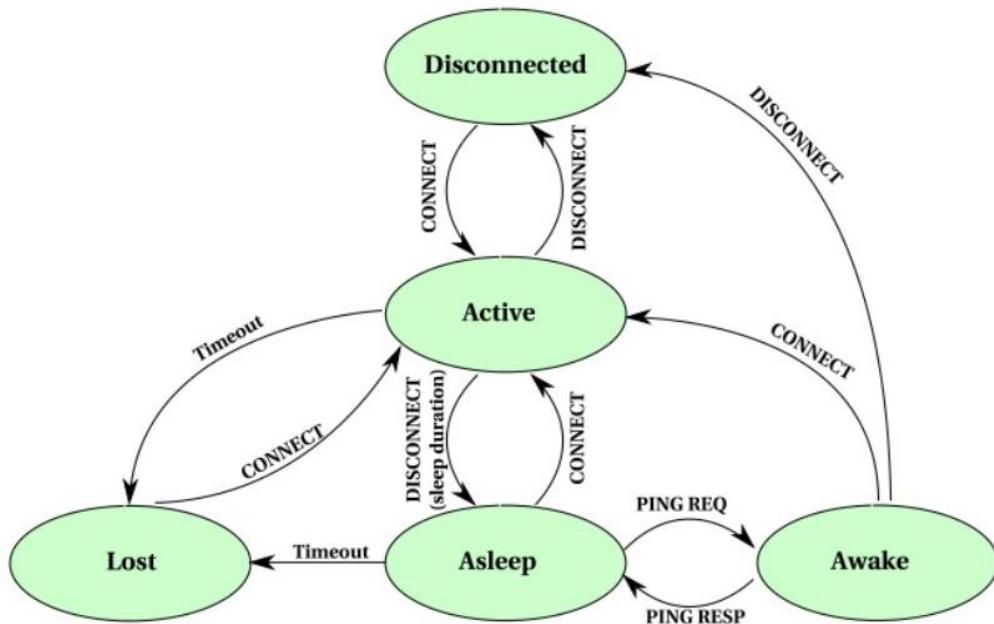
GW xác nhận đã nhận được gói tin DISCONNECT bằng cách phản hồi lại một gói tin DISCONNECT đến Client. Client cũng có thể nhận một gói tin DISCONNECT từ GW khi GW không thể xác định được Client do lỗi. Khi nhận được gói tin DISCONNECT này, Client nên thử thiết lập lại kết nối bằng cách gửi gói tin CONNECT đến GW.

### **Tiến trình truyền lại bản tin của Client**

Khi Client gửi định hướng các gói tin (không phải các gói tin quảng bá) đến một GW (một địa chỉ duy nhất), thì GW sẽ gói tin phản hồi tương ứng để xác nhận việc đã nhận được các gói tin đó. Việc này được giám sát bởi thời gian  $T_{retry}$ . Nó được tính bắt đầu từ khi Client gửi một gói tin và kết thúc khi nhận được phản hồi từ GW. Nếu vượt quá thời gian  $T_{retry}$  và không nhận được phản hồi từ GW thì Client sẽ gửi lại gói tin. Sau  $N_{retry}$  lần gửi lại, Client sẽ huỷ tiến trình này và cho rằng kết nối đến GW bị ngắt. Nó nên thử kết nối đến GW khác khi nó không thể kết nối lại với GW cũ.

### **Tính năng**

Tiết kiệm năng lượng là yêu cầu vô cùng quan trọng đối với các thiết bị truyền thông không dây sử dụng pin (ví dụ như cảm biến trong WSNs). Do đó, giao thức MQTT-SN cho phép Client rơi vào trạng thái ngủ khi chúng không hoạt động và có khả năng thức dậy bất cứ khi nào chúng có dữ liệu phải gửi đi. Broker/GW phải có khả năng nhận biết khi nào Client rơi vào trạng thái ngủ, và nó phải lưu ý các bản tin đến Client và gửi đến Client ngay khi chúng thức dậy.

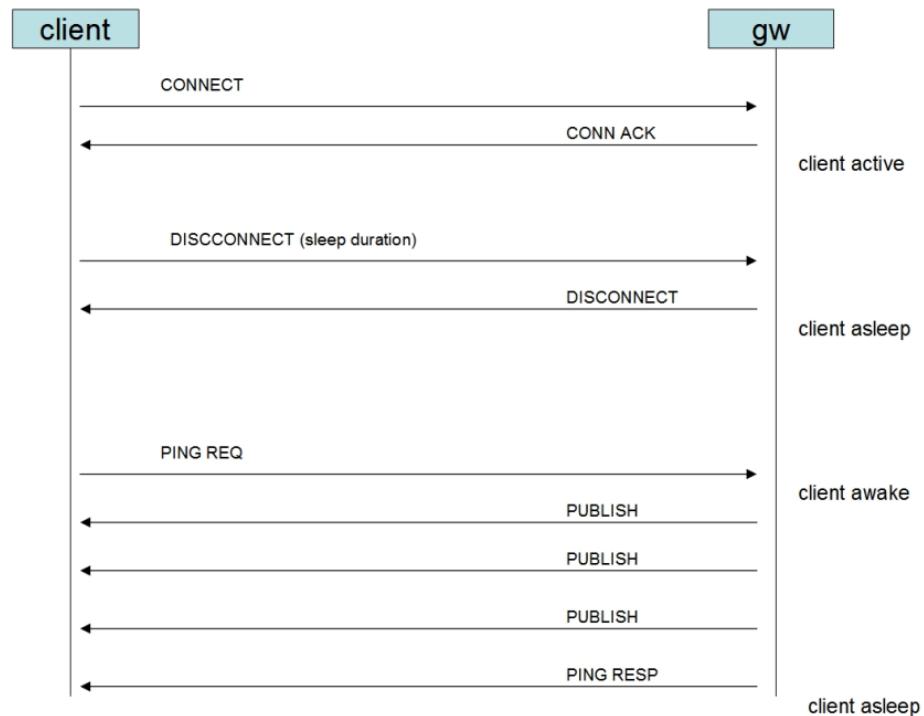


Hình 4.44: Sơ đồ chuyển trạng thái của Client.

Hình 4.44. mô tả sơ đồ chuyển trạng thái của Client, Broker/GW xem xét trạng thái của Client có thể ở một trong 4 trạng thái sau: Hoạt động, ngủ, ngắt kết nối và mất kết nối (bất thường). Một Client ở trạng thái hoạt động nếu như Broker/GW nhận được gói tin

CONNECT từ Client như được mô tả trong mục 1.6.5.2. Trạng thái này được giám sát bởi tính năng Keep Alive như được mô tả trong mục 1.6.5.11. Nếu Broker/GW không nhận được bất kỳ một gói tin nào từ Client trong khoảng thời gian lớn hơn khoảng thời gian Keep Alive thì nó sẽ coi như Client đó bị mất kết nối bất thường và nó sẽ kích hoạt tính năng WILL cho Client đó. Broker/GW coi Client chuyển sang trạng thái ngắt kết nối khi nó nhận được gói tin DISCONNECT từ Client, trạng thái này không bị giám sát bởi các hàm thời gian.

Nếu Client muốn rời vào trạng thái ngủ, nó phải gửi gói tin DISCONNECT có chứa khoảng thời gian mà nó ở trong trạng thái ngủ. Broker/GW sẽ xác nhận bằng việc phản hồi một gói tin DISCONNECT và coi như Client đang ở trong trạng thái ngủ. Trạng thái ngủ của Client được giám sát bởi Broker/GW với thời gian ngủ đã được thông nhất trong bản tin DISCONNECT. Nếu sau khoảng thời gian ngủ mà Broker/GW không nhận được bất kỳ gói tin nào từ Client thì nó sẽ coi như Client chuyển sang trạng thái mất kết nối và nó sẽ kích hoạt tính năng WILL.



Hình 4. 45: Tiến trình chuyển sang trạng thái ngủ của Client.

Bộ định thời sẽ dừng khi Broker/GW nhận được một gói tin PINGREQ chứa ClientID (giống gói tin CONNECT) từ Client. Khi đó, Broker/GW sẽ coi Client đó chuyển sang

trạng thái thức (hoạt động). Nếu có bất kỳ bản tin nào trong bộ nhớ đệm của Broker/GW thì nó sẽ gửi chúng đến Client trong các gói tin PUBLISH. Sau khi Broker/GW kết thúc việc gửi các bản tin, nó sẽ gửi một gói tin PINGRESP đến Client, sau đó khởi động lại bộ định thời và coi như Client lại bắt đầu trạng thái ngủ trong chu kỳ tiếp theo. Nếu Broker/GW không có bất kỳ bản tin nào của Client được lưu trong bộ nhớ đệm thì nó sẽ phản hồi ngay bằng một bản tin PINGRESP và khởi động lại bộ định thời.

Sau khi gửi gói tin PINGREQ đến Broker/GW, Client sử dụng cơ chế truyền lại bản tin tương tự như được mô tả trong mục 1.6.5.13 để giám sát việc gửi các bản tin của Broker/GW. Tức là nó sẽ khởi động bộ định thời  $T_{retry}$  khi nó nhận được bất kỳ gói tin nào khác gói tin PINGRESP. Nó sẽ truyền lại gói tin PINGREQ khi và khởi động lại bộ định thời  $T_{retry}$  khi vượt quá thời gian  $T_{retry}$ . Tuy nhiên, để tránh việc hao pin do phải truyền lại quá nhiều gói tin PINGREQ (ví dụ như khi Client mất kết nối đến GW) thì Client nên đặt một giới hạn số lần gửi lại gói tin PINGREQ. Nó sẽ quay lại trạng thái ngủ nếu như số lần gửi lại bị vượt quá giới hạn mà vẫn không nhận được gói tin phản hồi PINGRESP.

Từ trạng thái ngủ, Client có thể trở lại trạng thái hoạt động bằng cách gửi một gói tin CONNECT hoặc chuyển sang trạng thái ngắn kết nối bằng cách gửi một gói tin DISCONNECT. Client cũng có thể sửa đổi thời gian ngủ của mình bằng cách gửi gói tin DISCONNECT với giá trị mới cho trường khai báo thời gian ngủ.

Lưu ý, Client khi đang ở trạng thái ngủ, chỉ nên chuyển sang trạng thái thức khi mà nó muốn kiểm tra xem có bất kỳ bản tin nào trong bộ nhớ đệm của Broker/GW hay không, và sẽ trả lại ngay trạng thái ngủ nếu không có bất kỳ bản tin nào. Hoặc nếu không nó có thể quay trở lại trạng thái hoạt động bằng cách gửi gói tin CONNECT.

## 4.6 Kết luận chương

Internet vạn vật là một xu hướng mới trong các năm gần đây và đã mở ra hàng loạt các ứng dụng hữu ích cho người sử dụng. Với một số các ứng dụng chính, các giao thức hiện hỗ trợ cho ứng dụng và dịch vụ này được đưa ra trong chương. Với tiếp cận gần nhất với kịch bản thực tế, các giao thức ứng dụng internet cho vạn vật được giới thiệu chi tiết để người đọc có thể tiếp cận và ứng dụng vào trong các kịch bản mạng thực tế.

## KẾT LUẬN

Cuốn bài giảng Internet và giao thức là nhằm cung cấp các nội dung kiến thức cơ bản như sau.

Một cách nhìn tổng quát về mạng toàn cầu và liên mạng (Internet) với các đặc trưng về kiến trúc, cấu trúc và thành phần mạng. Bên cạnh đó là các giải pháp công nghệ sử dụng cho các truy nhập mạng Internet.

Các giao thức lớp ứng dụng của Internet bao gồm các ứng dụng phổ biến liên quan tới người sử dụng và ứng dụng hạ tầng mạng được đưa ra nhằm giúp người đọc phân biệt các nguyên tắc triển khai các ứng dụng trên nền tảng Internet.

Các mô hình cung cấp dịch vụ như máy chủ- máy khách và mạng ngang hàng được trình bày qua các giao thức phổ biến nhất. Các phân tích về hiệu năng kết nối cũng như sự phù hợp của các giao thức đã được đưa ra.

Để làm rõ xu hướng hội tụ mạng, các nguyên tắc báo hiệu và điều khiển cho các dịch vụ đa phương tiện, trong bài giảng đã chi tiết về các giao thức điều khiển phiên và điều khiển chất lượng thông tin qua mạng Internet.

Cuối cùng, hướng tiếp cận gần nhất là internet vạn vật được trình bày cùng với các giao thức chính nhằm mở rộng sự phát triển các ứng dụng trên nền tảng internet cho người học.

Nhóm tác giả rất mong sự góp ý của đồng nghiệp và sinh viên.

## **TÀI LIỆU THAM KHẢO**

1. James F. Kurose and Keith W. Ross. **Computer Networking: A Top-Down Approach Featuring the Internet.** Addison Westley, 2010.
2. Peterson, Larry L., and Bruce S. Davie. Computer networks: a systems approach. Elsevier, 2007.
3. Ahson, Syed A., and Mohammad Ilyas, eds. SIP handbook: services, technologies, and security of Session Initiation Protocol. CRC Press, 2018.
4. Cirani, Simone, Gianluigi Ferrari, Marco Picone, and Luca Veltri. Internet of Things: Architectures, Protocols and Standards. John Wiley & Sons, 2018.