TRƯỜNG ĐẠI HỌC CÔNG NGHỆ GTVT
**KHOA CÔNG NGHỆ THÔNG TIN**


**TS. ĐỖ XUÂN THU**
**ThS. PHAN NHƯ MINH**

**(Khoa Công nghệ thông tin)**

# BÀI GIẢNG
# BỘ GIAO THỨC TCP/IP

DÙNG CHO SINH VIÊN KHOA CÔNG NGHỆ THÔNG TIN

LƯU HÀNH NỘI BỘ
Hà nội 2022

# MỤC LỤC

# DANH MỤC CÁC KÝ HIỆU, CÁC CHỮ VIẾT TẮT

| STT | Từ viết tắt | Ý nghĩa |
|---|---|---|
| 1 | AC | Access Control |
| 2 | ACK | Acknowledgment |
| 3 | ADSL | Asymmetrric Digital Subcriber Line |
| 4 | ARP | Address Resolution Protocol |
| 5 | ARPANet | Advanced Research Projects Agency Net |
| 6 | AS | Infraction routing |
| 7 | ASCII | American Standard Code for Information Interchange |
| 8 | ASE | Application Service Emlement |
| 9 | ATM | Asynchronous Transfer Mode |
| 10 | BGP | Border Gateway Protocol |
| 11 | B-ISDN | Broadband-Integrated Services Digital Network |
| 12 | CIDR | Classless Inter domain routing |
| 13 | CRC | Cyclic Redundancy Check |
| 14 | DHCP | Dianmic Host Configuaration Protocol |
| 15 | DNS | Domain Name Service |
| 16 | ED | End Deliniter |
| 17 | EIGRP | Enhanced Interior Geteway Routing Protocol |
| 18 | FC | Frame Control |
| 19 | FCS | Frame Check Sequence |
| 20 | FIN | Relase the connection |
| 21 | FTP | File Transfer Protocol |
| 22 | GGP | Getway to Getway Protocol |
| 23 | HTML | HyperText Transport Protocol |
| 24 | HTTP | Hypertext Tranfer Protocol |
| 25 | ICANN | Internet Copration for Assigned Names and Number |
| 26 | ICMP | Internet Control Message Protocol |
| 27 | IETF | Internet Euginerring Task Force |

| | | |
|---|---|---|
| 28 | IGRP | Interior Geteway Routing Protocol |
| 29 | IP | Internet Protocol |
| 30 | IPX | Internetwork Packet eXchange |
| 31 | ISDN | Integrated Services Digital Network |
| 32 | ISP | Internet Service provider |
| 33 | ITU | International Telecommunication Union |
| 34 | LAN | Local Area Network |
| 35 | LEN | Length |
| 36 | MAC | Media Access Control |
| 37 | MSS | Maximum Segment Size |
| 38 | MTU | Maximum Tranfer Unit |
| 39 | NIC | Network Inormation Center |
| 40 | NLRI | Network Layer Reachability Information |
| 41 | NSF | National Science Foundation |
| 42 | OSI | Open Systems Interconnection |
| 43 | OSPF | Open shortest Path First |
| 44 | POP | Post Office Protocol |
| 45 | RARP | Reverse Address Resolution Protocol |
| 46 | RED | Random Early Discard |
| 47 | RIP | Routing Information Protocol |
| 48 | ROADS | Running Out of Address Space |
| 49 | ROM | Read Only Memory |
| 50 | RTP | Realtime Transport Protocol |
| 51 | RTT | Round Trip Times |
| 52 | SAO | Single Association Object |
| 53 | SD | Start Deliniter |
| 54 | SFD | Start of Frame Delimiter |
| 55 | SMTP | Simple Mail Transfer Protocol |
| 56 | SPF | Shortest Path First |
| 57 | SPX | Sequence Packet eXchange |

| | | |
|---|---|---|
| 58 | SYN | Synchronize sequence numbers |
| 59 | TCP | Transmisstion Control Protocol |
| 60 | ToS | Type of Service |
| 61 | TTL | Time To Live |
| 62 | UDP | User Datagram Protocol |
| 63 | VLSM | Variable-Length Subnet Mask |
| 64 | WAN | Wide Area Network |
| 65 | WWW | World Wide Web |

# DANH MỤC CÁC BẢNG

# DANH MỤC CÁC HÌNH VẼ

# LỜI NÓI ĐẦU

Ngày nay thế giới đã và đang bước vào kỷ nguyên của sự bùng nổ thông tin, cùng với sự phát triển như vũ bão của các phương tiện truyền thông đại chúng, lĩnh vực truyền thông mạng máy tính đã và đang phát triển không ngừng. Mạng máy tính toàn cầu Internet đã và đang trở thành hạ tầng của hạ tầng cho mọi nền kinh tế, vai trò của nó đã trở thành một thành tố không thể thiếu trong mọi mặt của nhân loại toàn cầu.

Với Internet, một mạng truyền thông toàn cầu dựa trên công nghệ chuyển mạch gói điển hình, với xương sống là các hệ thống định tuyến làm nhiệm vụ kết nối và chuyển phát các gói tin. Giao thức mạng nền tảng của Internet là bộ giao thức TCP/IP.

Giáo trình này sẽ đi trình bày chi tiết về bộ giao thức TCP/IP và cách thức hoạt động của các bộ định tuyến trong việc chuyển phát và dẫn đường cho các gói tin đến đích một cách tối ưu nhất, giáo trình cũng trình bày chi tiết các giao thức định tuyến phổ biến và điển hình hiện nay.

Những kiến thức được trình bày rất cần thiết cho sinh viên các ngành Công nghệ thông tin và đặc biệt là ngành An toàn thông tin, là nền tảng kiến thức về mạng và truyền thông và là cơ sở kiến thức cho các khối kiến thức chuyên sâu về quản trị mạng, tối ưu mạng, giám sát mạng, an toàn và an ninh mạng, an toàn giao thức mạng.

*Hà nội 8/2013*

# CHƯƠNG 1 TỔNG QUAN VỀ GIAO THỨC TCP/IP VÀ CẤU TRÚC LIÊN MẠNG INTERNET

## 1.1. Giới thiệu chung về lịch sử Internet

Internet bắt nguồn từ đề án ARPANET (Advanced Research Project Agency Network) khởi sự trong năm 1969 bởi Bộ Quốc phòng Mỹ (American Department of Defense). Đề án Arpanet với sự tham gia của một số trung tâm nghiên cứu, đại học tại Mỹ (UCLA, Stanford,...) nhằm mục đích thiết kế một mạng WAN (Wide Area Network) có khả năng tự bảo tồn chống lại sự phá hoại một phân mạng bằng chiến tranh nguyên tử. Đề án này dẫn tới sự ra đời của giao thức truyền IP (Internet Protocol). Nguyên lý cơ bản của giao thức này là: thông tin truyền sẽ được đóng thành các gói dữ liệu và truyền trên mạng theo nhiều đường khác nhau từ người gửi tới nơi người nhận. Một hệ thống máy tính (hoặc thiết bị) đóng vài trò làm nhiệm vụ tìm đường đi tối ưu cho các gói dữ liệu và có tên gọi là Router, do trong giao thức này tất cả các máy tính trên mạng đều tham dự vào việc truyền dữ liệu, nhờ vậy nếu một phân mạng bị phá huỷ các Router có thể tìm đường khác để truyền các gói tin người nhận. Mạng Arpanet được phát triển và sử dụng trước hết trong các trường đại học, các cơ quan nhà nước Mỹ, tiếp theo đó, các trung tâm tính toán lớn, các trung tâm truyền vô tuyến điện và vệ tinh được nối vào mạng này ngày càng nhiều,... trên cơ sở này, Arpanet được nối với khắp các vùng trên thế giới.

Tới năm 1983, trước sự thành công của việc triển khai mạng Arpanet, Bộ quốc phòng Mỹ tách một phân mạng giành riêng cho quân đội Mỹ (MILNET). Phần còn lại, gọi là NSFnet, được quản lý bởi NSF (National Science Foundation) NSF dùng 5 siêu máy tính để làm Router cho mạng, và lập một tổ chức không chính phủ để quản lý mạng, chủ yếu dùng cho đại học và nghiên cứu cơ bản trên toàn thế giới. Tới năm 1987, NSFnet mở cửa cho cá nhân và cho các công ty tư nhân (BITnet), tới năm 1988 siêu mạng được mang tên Internet.

Tuy nhiên cho tới năm 1988, việc sử dụng Internet còn hạn chế trong các dịch vụ truyền mạng (FTP: File Transfer Protocol), thư điện tử (Email), truy nhập từ xa (TELNET) không thích ứng với nhu cầu kinh tế và đời sống hàng ngày. Internet chủ yếu được dùng trong môi trường nghiên cứu khoa học và giảng dạy đại học. Trong năm 1988, tại trung tâm nghiên cứu nguyên tử của Pháp CERN(Centre Européen de Recherche Nuclaire) ra đời đề án Mạng nhện thế giới WWW (World Wide Web). Đề án này, nhằm xây dựng một phương thức mới sử dụng Internet, gọi là phương thức Siêu văn bản (Hyper Text). Các tài liệu và hình ảnh được trình bày bằng ngôn ngữ HTML (HyperText Markup Language) và được phát hành trên

Internet qua các máy chủ làm việc với giao thức HTTP (HyperText Transport Protocol). Từ năm 1992, phương thức làm việc này được đưa ra thử nghiêm trên Internet. Rất nhanh chóng, các công ty tư nhân tìm thấy vô số cơ hội kinh doanh và truyền thông sử dụng phương thức này, từ đó Internet được ứng dụng rộng rãi trong kinh tế và đời sống. Các nguồn lực đầu tư cho sự phát triển của Internet được nhân lên hàng chục lần. Từ năm 1994 Internet trở thành siêu mạng kinh doanh, số các công ty sử dụng Internet vào việc kinh doanh và quảng cáo lên gấp hàng nghìn lần kể từ năm 1995. Doanh số giao dịch thương mại qua mạng Internet lên hàng ngàn tỉ USD trong những năm gần đây...

Với phương thức siêu văn bản, người sử dụng, qua một phần mềm truy đọc (thường gọi là Web browse hoặc Navigator), có thể tìm đọc tất cả các dữ liệu đa phương tiện được công bố tại mọi nơi trên thế giới (bao gồm: văn bản, hình ảnh, âm thanh, video…. Với công nghệ WWW, chúng ta bước vào giai đoạn mà mọi thông tin có thể có ngay trên bàn làm việc của mình. Mỗi công ty hoặc người sử dụng, được phân phối một trang chủ gốc (Home Page) trên hệ chủ HTTP. Trang chủ này là siêu văn bản gốc, để từ đó có thể truy xuất tới tất cả các siêu văn bản khác mà người sử dụng muốn phát hành. Địa chỉ của trang chủ được tìm thấy từ khắp mọi nơi trên thế giới. Vì vậy, đối với một doanh nghiệp, trang chủtrở thành một văn phòng đại diện điện tử trên Internet. Từ khắp mọi nơi, khách hàng có thể xem các quảng cáo và liên hệ trực tiếp với xí nghiệp qua các dòng siêu liên kết (Hyper Link) trong siêu văn bản.

Tới năm 1994, một điểm yếu của Internet là không có khả năng lập trình và tương tác dữ liệu một cách linh hoạt, vì các máy nối vào mạng không đồng bộ và không tương thích. Thiếu khả năng này, Internet chỉ được dùng trong việc phát hành và truyền thông tin (tĩnh) chứ không dùng để xử lý thông tin (động) được. Trong năm 1994, hãng máy tính SUN Corporation công bố một ngôn ngữ mới, gọi là JAVA (với lô gô hình ly cafe), cho phép lập trình cục bộ trên Internet, các chương trình JAVA được gọi thẳng từ các siêu văn bản qua các siêu liên (Applet). Vào mùa thu năm 1995, ngôn ngữ JAVA chính thức ra đời, đánh dấu một bước tiến quan trọng trong việc linh hoạt và năng động hóa Internet. Trước hết, một chương trình JAVA, sẽ được chạy trên máy khách (Workstation) chứ không phải trên máy chủ (server). Điều này cho phép sử dụng hiệu suất của tất cả các máy khách vào việc xử lý số liệu. Hàng triệu máy tính (hoặc thiết bị xử lý thông minh như PDA, SmartPhone…) có thể thực hiện cùng một lúc một chương trình được lưu trữ trong các máy chủ.

Việc lập trình trên Internet cho phép truy nhập từ một trang siêu văn bản vào các chương trình xử lý thông tin, đặc biệt là các chương trình điều hành và quản lý

thông tin của một doanh nghiệp. phương thức làm việc này, được gọi là Intranet. Chỉ trong năm 1995-1996, hàng trăm nghìn dịch vụ phần mềm Intranet được phát triển. Nhiều hãng máy tính và phần mềm như Microsoft, SUN, IBM, Oracle, Netscape,... đã phát triển và kinh doanh hàng loạt phần mềm hệ thống và phần mềm cơ bản để phát triển các ứng dụng Internet / Intranet.

Ngày nay Internet đã phát triển rất mạnh và đang ở giai đoạn của thế hệ điện toán đám mây (cloud computing), còn gọi là điện toán máy chủ ảo, là mô hình xử lý và lưu trữ thông tin sử dụng các công nghệ máy tính và phát triển dựa vào mạng Internet. Thuật ngữ "đám mây" ở đây là lối nói ẩn dụ chỉ mạng Internet (dựa vào cách được bố trí của nó trong sơ đồ mạng máy tính) và như một liên tưởng về độ phức tạp của các cơ sở hạ tầng chứa trong nó. Ở mô hình điện toán này, mọi khả năng liên quan đến công nghệ thông tin đều được cung cấp dưới dạng các "dịch vụ", cho phép người sử dụng truy cập các dịch vụ công nghệ từ một nhà cung cấp nào đó "trong đám mây" mà không cần phải có các kiến thức, kinh nghiệm về công nghệ đó, cũng như không cần quan tâm đến các cơ sở hạ tầng phục vụ công nghệ đó. Theo tổ chức Xã hội máy tính IEEE "Nó là hình mẫu trong đó thông tin được lưu trữ thường trực tại các máy chủ trên Internet và chỉ được được lưu trữ tạm thời ở các máy khách, bao gồm máy tính cá nhân, trung tâm giải trí, máy tính trong doanh nghiệp, các phương tiện máy tính cầm tay,...".

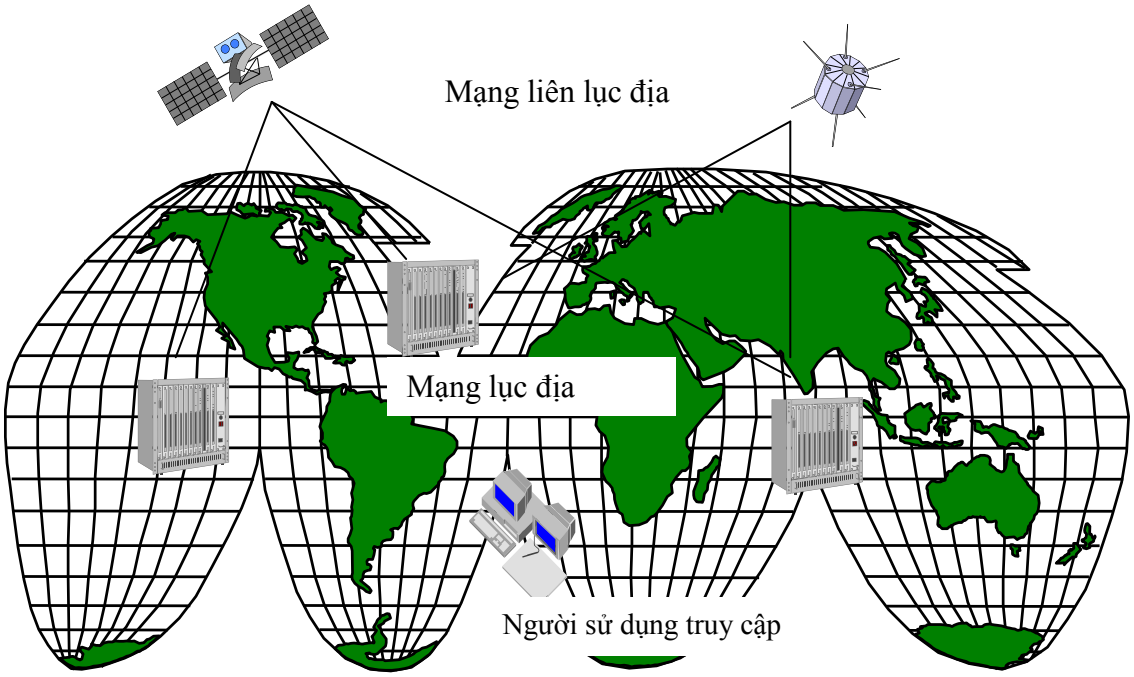
## 1.2. Kiến trúc mạng Internet, tham chiếu giữa mô hình giao thức TCP/IP và OSI.

### 1.2.1. Giới thiệu kiến trúc mạng Internet

Về mặt kiến trúc kết nối Internet là một siêu mạng dựa trên sự liên nối trên nhiều lớp mạng khác nhau:

- Mạng liên lục địa: Sử dụng trục cable qua các đại dương, hoặc sử dụng các vệ tinh. Mục đích là nối thông tin giữa các lục địa. Một số hãng điều tiết chính (Operaters) trên thế giới: MCInet, SPRINTlink, ANSnet AOL, CERFnet, Ebone, Eurpanet,...
- Mạng lục địa: Gồm các hãng điều tiết quốc gia hay liên quốc gia, cung cấp phương tiện truyền tin cho các khách hàng trên một vùng nhất định của một lục địa: VIETPAC (Việt Nam), TRANSPAC (Pháp), AUSPAC (Australia), TELEPAC (Singapore),...
- Mạng truy nhập địa phương: Gồm các hãng bán dịch vụ cổng vào cho khách hàng qua mạng lưới điện thoại hay mạng riêng, và nối vào các mạng lục địa bởi các đường truyền đặc biệt (Specialized links): TRANSPAC France Télécom, FranceNet, World Net, Imaginet,...

- Mạng biệt lập: Các mạng được xây dựng riêng để bán dịch vụ cho khách và có cổng nối với siêu mạng Internet (Computer Serve, IBM, Micronet, Microsoft Network,...)




*Hình 1.1: Kiến trúc các lớp mạng Internet*

Các nhà cung cấp dịch vụ, bao gồm:

- Các hãng điều tiết Internet: Các hãng này, có khả năng cung cấp đường kết nối và liên tục vào siêu mạng (on line services), họ tham gia vào việc quản lý hệ Internet trên phạm vi một địa phương hay một quốc gia: VIETPAC (VDC Việt Nam), AUSPAC, TRANSPAC, FPT, VIETEL, SPT...
- Các hãng cung cấp dịch vụ dial up: cho thuê bao cổng vào qua hệ thống điện thoại. Các dịch vụ này không phải là dịch vụ liên tục (off line services). Tại Việt Nam: Varenet và Netnam Viện Công nghệ Thông tin, Vietnet Bưu điện Khánh Hoà, Trí tuệ Việt Nam Công ty FPT, Phương Nam Trung tâm Hội chợ Triển lãm,...
- Các hãng cung cấp dịch vụ giá trị gia tăng trên mạng Internet: tài liệu trực tuyến, thanh toán điện tử, ngân hàng, lưu trữ dữ liệu, thư viện phim ảnh, trường học online, video conference, Ip phone, tìm kiếm nội dung…
- Các hãng thuê bao cổng vào thường kết hợp với việc làm các dịch vụ Internet như: thuê làm trang chủ (Home Page), thiết kế và xây dựng các

website và cho thuê dung lượng lưu trữ, quản lý các nhóm hội thảo (NEWGROUPS), dịch vụ INTRANET,...

Về mặt thiết bị ba thành phần chính tạo nên Internet là:

- Các trạm máy chủ (Hosts), các trạm làm việc (Workstation), các thiết bị xử lý thông tin thông minh cầm tay,... trên đó chạy các chương trình ứng dụng. Các máy tính có thể thuộc các loại khác nhau, chỉ cần hiểu được TCP/IP và có phần cứng, phần mềm tương ứng để truy cập và sử dụng các dịch vụ Internet.
- Các mạng diện rộng, mạng cục bộ, đường thuê bao điểm điểm (Point to Point), liên kết Dial Up (điện thoại, ISDN, X.25, ADSL) v.v... mang tải thông tin trao đổi giữa các máy tính.
- Các bộ dẫn đường (ROUTER) phục vụ việc kết nối giữa các mạng.

Về công nghệ mạng

Nhiều công nghệ mạng khác nhau được kết hợp nhằm đảm bảo ở khắp mọi nơi dịch vụ chuyển nhận các gói dữ liệu (IP packet đơn vị cấu thành trao đổi thông tin) trên mạng. Vấn đề then chốt là cần có chuẩn truyền thông thống nhất và cơ chế dẫn đường trên các mạng phân tán diện rộng. Về mặt truyền thông thì Internet dựa trên tập hợp các giao thức có tên chung là TCP/IP được xây dựng nhằm cho mục đích trên:

- Mỗi máy tính trên mạng Internet đều có 1 địa chỉ IP duy nhất.
- Cơ chế dẫn đường được thực hiện qua các ROUTER. Tại đó có các bảng thông tin dẫn đường được cập nhật liên tục chỉ cho biết điểm đến tiếp theo trên mạng. Khi có một trạm nào đó bị hỏng thông tin có thể được lái đi qua một số trạm khác để đến địa chỉ cuối cùng.

### 1.2.2. Tổng quan về giao thức TCP/IP

#### *1.2.2.1. Giới thiệu giao thức TCP/IP*

Sự ra đời của họ giao thức TCP/IP gắn liền với sự ra đời của Internet mà tiền thân là mạng **ARPA**net (**A**dvanced **R**esearch **P**rojects **A**gency) do Bộ Quốc phòng Mỹ tạo ra. Đây là bộ giao thức được dùng rộng rãi nhất vì tính mở và độ linh hoạt của nó. Điều đó có nghĩa là bất cứ máy nào dùng bộ giao thức TCP/IP đều có thể nối được vào Internet. Hai giao thức được dùng chủ yếu ở đây là **TCP** (**T**ransmission **C**ontrol **P**rotocol) và **IP** (**Internet P**rotocol). Chúng đã nhanh chóng được đón nhận và phát triển bởi nhiều nhà nghiên cứu và các hãng công nghiệp máy tính với mục đích xây dựng và phát triển một mạng truyền thông mở

rộng khắp thế giới mà ngày nay chúng ta gọi là Internet. Phạm vi phục vụ của Internet không còn dành cho quân sự như **ARPA**net nữa mà nó đã mở rộng lĩnh vực cho mọi loại đối tượng sử dụng, trong đó tỷ lệ quan trọng nhất vẫn thuộc về giới nghiên cứu khoa học và giáo dục.

Và như vậy, bộ giao thức TCP/IP (Transmission Control Protocol/ Internet Protocol) ra đời hình thành một bộ giao thức cùng làm việc với nhau để cung cấp phương tiện truyền thông liên mạng từ những năm 70, và nó là một bộ giao thức gồm nhiều giao thức khác nhau, với hai giao thức xương sống và quan trọng nhất là TCP và IP.

Đến năm 1981, TCP/IP phiên bản 4 mới hoàn tất và được phổ biến rộng rãi cho toàn bộ những máy tính sử dụng hệ điều hành UNIX. Sau này Microsoft cũng đã đưa TCP/IP trở thành một trong những giao thức căn bản của hệ điều hành Windows 200x mà hiện nay đang sử dụng, và hiện nay khi nói đến TCP/IP là người ta hiểu là giao thức TCP/IP phiên bản 4.

Đến năm 1994, một bản thảo của phiên bản IPv6 được hình thành với sự cộng tác của nhiều nhà khoa học thuộc các tổ chức Internet trên thế giới để cải tiến những hạn chế của IPv4.

Khác với mô hình ISO/OSI tầng liên mạng sử dụng giao thức kết nối mạng "không liên kết" (connectionless) IP, tạo thành hạt nhân hoạt động của Internet. Cùng với các thuật toán định tuyến RIP, OSPF, BGP, tầng liên mạng IP cho phép kết nối một cách mềm dẻo và linh hoạt các loại mạng "vật lý" khác nhau như: Ethernet, Token Ring, X.25...

Giao thức trao đổi dữ liệu "có liên kết" (connection oriented) TCP được sử dụng ở tầng vận chuyển để đảm bảo tính chính xác và tin cậy việc trao đổi dữ liệu dựa trên kiến trúc kết nối "không liên kết" ở tầng liên mạng IP.

Các giao thức hỗ trợ ứng dụng phổ biến như truy nhập từ xa (telnet), chuyển tệp (FTP), dịch vụ World Wide Web (HTTP), thư điện tử (SMTP), dịch vụ tên miền (DNS) ngày càng được cài đặt phổ biến như những bộ phận cấu thành của các hệ điều hành thông dụng như UNIX (và các hệ điều hành chuyên dụng cùng họ của các nhà cung cấp thiết bị tính toán như AIX của IBM, SINIX của Siemens, Digital UNIX của DEC), Windows/NT, Novell Netware, Linux...

*1.2.2.2. Mô hình OSI 7 tầng*

Mô hình OSI (Open Systems Interconnection Reference Model, viết ngắn là OSI Model hoặc OSI Reference Model) - tạm dịch là Mô hình tham chiếu kết nối các hệ thống mở - là một thiết kế dựa vào nguyên lý tầng cấp, lý giải một cách trừu

tượng kỹ thuật kết nối truyền thông giữa các máy vi tính và thiết kế giao thức mạng giữa chúng. Mô hình này được phát triển thành một phần trong kế hoạch Kết nối các hệ thống mở (Open Systems Interconnection) do ISO và IUT-T khởi xướng. Nó còn được gọi là Mô hình bảy tầng của OSI.

Mô hình OSI phân chia chức năng của một giao thức ra thành một chuỗi các tầng cấp. Mỗi một tầng cấp có một đặc tính là nó chỉ sử dụng chức năng của tầng dưới nó, đồng thời chỉ cho phép tầng trên sử dụng các chức năng của mình. Một hệ thống cài đặt các giao thức bao gồm một chuỗi các tầng nói trên được gọi là "chồng giao thức" (protocol stack). Chồng giao thức có thể được cài đặt trên phần cứng, hoặc phần mềm, hoặc là tổ hợp của cả hai. Thông thường thì chỉ có những tầng thấp hơn là được cài đặt trong phần cứng, còn những tầng khác được cài đặt trong phần mềm.

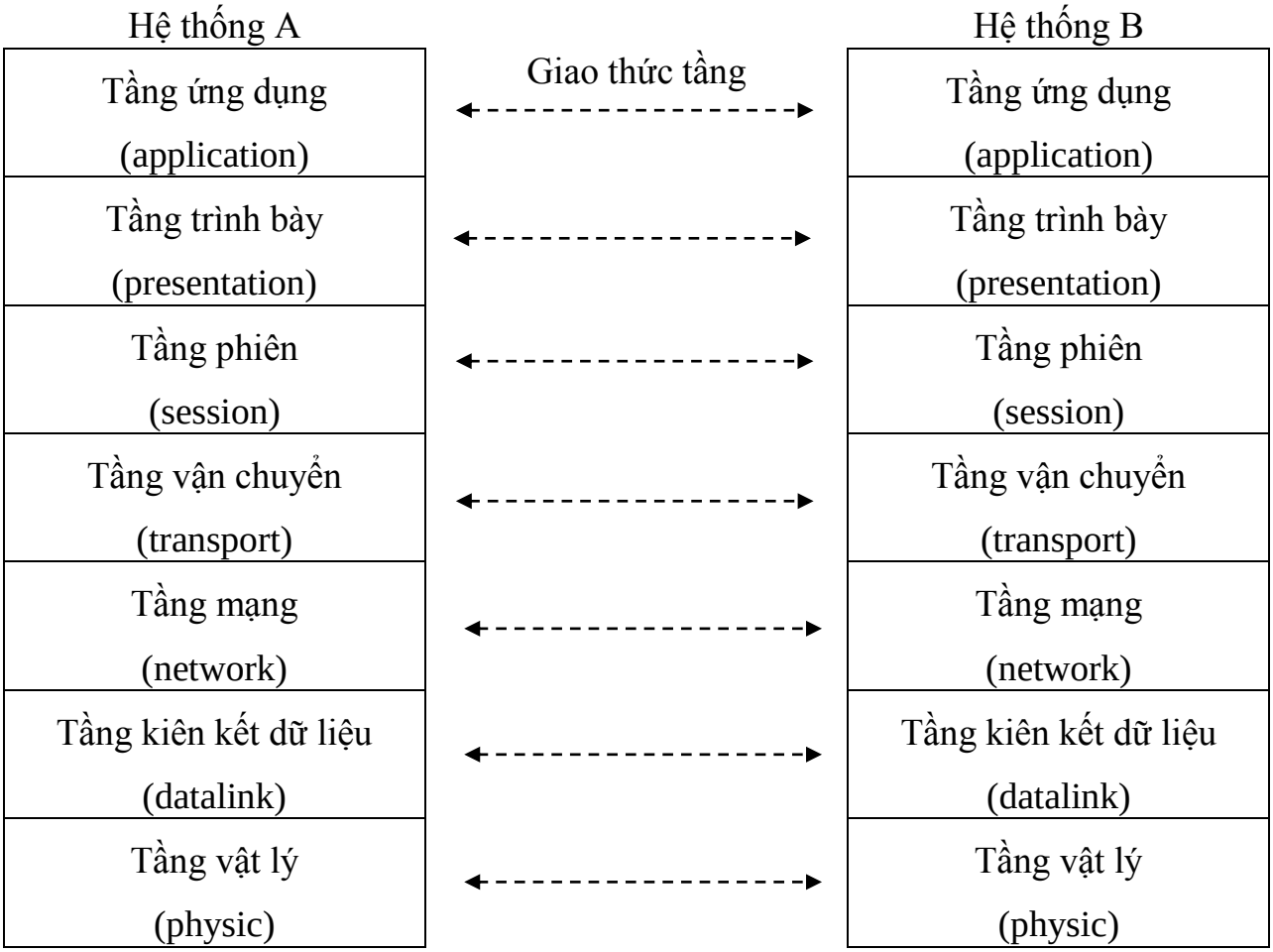
***Tầng 1: Vật lý (Physical)***

Tầng vật lý (Physical layer) là tầng dưới cùng của mô hình OSI. Nó mô tả các đặc trưng vật lý của mạng: Các loại cáp được dùng để nối các thiết bị, các loại đầu nối được dùng, các dây cáp có thể dài bao nhiêu v.v... Mặt khác các tầng vật lý cung cấp các đặc trưng điện của các tín hiệu được dùng để khi chuyển dữ liệu trên cáp từ một máy này đến một máy khác của mạng, kỹ thuật nối mạch điện, tốc độ cáp truyền dẫn.

Tầng vật lý không qui định một ý nghĩa nào cho các tín hiệu đó ngoài các giá trị nhị phân 0 và 1. Ở các tầng cao hơn của mô hình OSI ý nghĩa của các bit được truyền ở tầng vật lý sẽ được xác định.

Ví dụ: Tiêu chuẩn Ethernet cho cáp xoắn đôi được dùng theo chuẩn 10 baseT định rõ các đặc trưng điện của cáp xoắn đôi, kích thước và dạng của các đầu nối, độ dài tối đa của cáp.

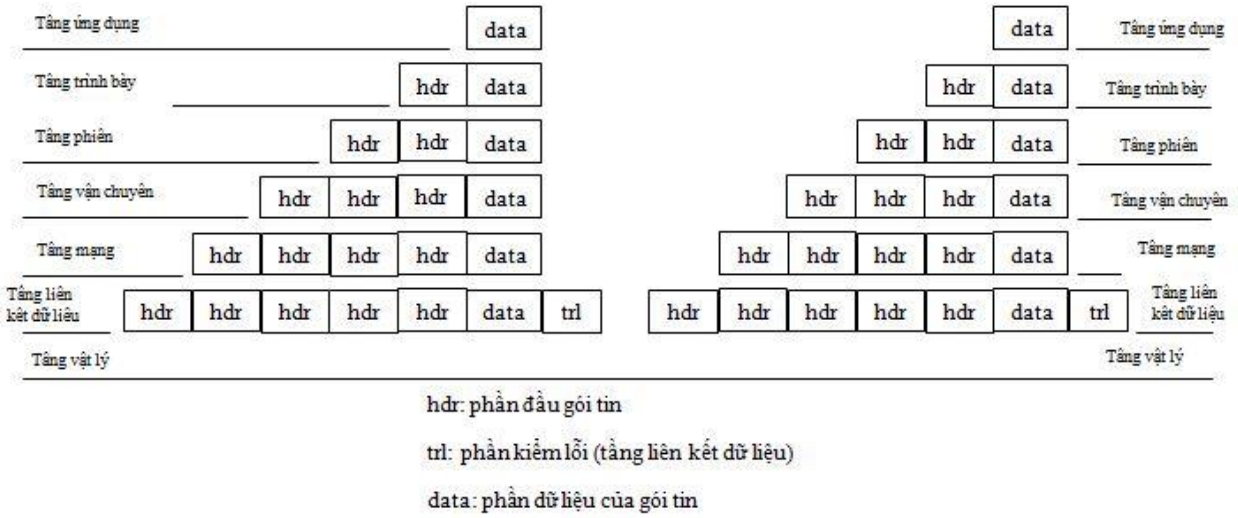
Khác với các tầng khác, tầng vật lý là không có gói tin riêng và do vậy không có phần đầu (header) chứa thông tin điều khiển, dữ liệu được truyền đi theo dòng bit. Một giao thức tầng vật lý tồn tại giữa các tầng vật lý để quy định về phương thức truyền (đồng bộ, phi đồng bộ), tốc độ truyền.




*Hình1.2: Mô hình OSI 7 tầng*

Các giao thức được xây dựng cho tầng vật lý được phân chia thành hai loại giao thức sử dụng phương thức truyền thông dị bộ (asynchronous) và phương thức truyền thông đồng bộ (synchronous).

Phương thức truyền dị bộ: không có một tín hiệu quy định cho sự đồng bộ giữa các bit giữa máy gửi và máy nhận, trong quá trình gửi tín hiệu máy gửi sử dụng các bit đặc biệt START và STOP được dùng để tách các xâu bit biểu diễn các ký tự trong dòng dữ liệu cần truyền đi. Nó cho phép một ký tự được truyền đi bất kỳ lúc nào mà không cần quan tâm đến các tín hiệu đồng bộ trước đó.yêu cầu phục vụ, các thông tin điều khiển và dữ liệu.



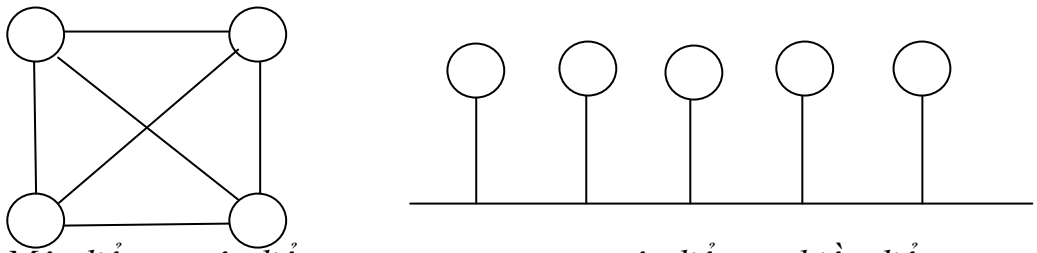

*Hình 1.3: Phương thức xác lập các gói tin trong mô hình OSI*

Phương thức truyền đồng bộ: sử dụng phương thức truyền cần có đồng bộ giữa máy gửi và máy nhận, nó chèn các ký tự đặc biệt như SYN (Synchronization), EOT (End Of Transmission) hay đơn giản hơn, một cái "cờ " (flag) giữa các dữ liệu của máy gửi để báo hiệu cho máy nhận biết được dữ liệu đang đến hoặc đã đến.

***Tầng 2: Liên kết dữ liệu (Data link)***

Tầng liên kết dữ liệu (data link layer) là tầng mà ở đó ý nghĩa được gán cho các bít được truyền trên mạng. Tầng liên kết dữ liệu phải quy định được các dạng thức, kích thước, địa chỉ máy gửi và nhận của mỗi gói tin được gửi đi. Nó phải xác định cơ chế truy nhập thông tin trên mạng và phương tiện gửi mỗi gói tin sao cho nó được đưa đến cho người nhận đã định.

Tầng liên kết dữ liệu có hai phương thức liên kết dựa trên cách kết nối các máy tính, đó là phương thức "một điểm một điểm" và phương thức "một điểm nhiều điểm". Với phương thức "một điểm một điểm" các đường truyền riêng biệt được thiết lập để nối các cặp máy tính lại với nhau. Phương thức "một điểm - nhiều điểm" tất cả các máy phân chia chung một đường truyền vật lý.



*Một điểm – một điểm* *một điểm – nhiều điểm*

*Hình 1.4: Các đường truyền kết nối*

Tầng liên kết dữ liệu cũng cung cấp cách phát hiện và sửa lỗi cơ bản để đảm bảo cho dữ liệu nhận được giống hoàn toàn với dữ liệu gửi đi. Nếu một gói tin có lỗi không sửa được, tầng liên kết dữ liệu phải chỉ ra được cách thông báo cho nơi gửi biết gói tin đó có lỗi để nó gửi lại.

Các giao thức tầng liên kết dữ liệu chia làm 2 loại chính là các giao thức hướng ký tự và các giao thức hướng bit. Các giao thức hướng ký tự được xây dựng dựa trên các ký tự đặc biệt của một bộ mã chuẩn nào đó (như ASCII hay EBCDIC), trong khi đó các giao thức hướng bit lại dùng các cấu trúc nhị phân (xâu bit) để xây dựng các phần tử của giao thức (đơn vị dữ liệu, các thủ tục.) và khi nhận, dữ liệu sẽ được tiếp nhận lần lượt từng bit một.

***Tầng 3: Mạng (Network)***

Tầng mạng (network layer) nhắm đến việc kết nối các mạng với nhau bằng cách tìm đường (routing) cho các gói tin từ một mạng này đến một mạng khác. Nó xác định việc chuyển hướng, vạch đường các gói tin trong mạng, các gói này có thể phải đi qua nhiều chặng trước khi đến được đích cuối cùng. Nó luôn tìm các tuyến truyền thông không tắc nghẽn để đưa các gói tin đến đích.

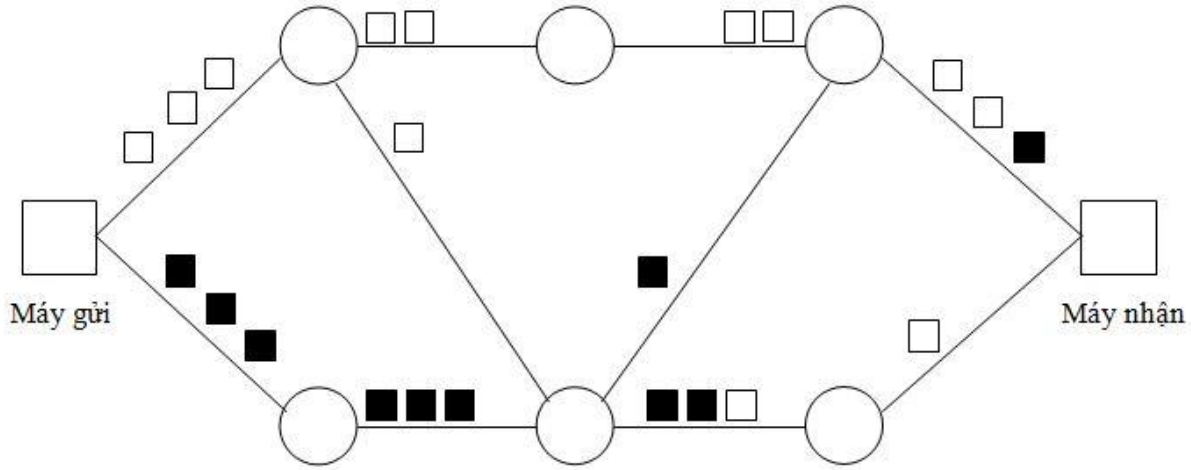
Tầng mạng cung các các phương tiện để truyền các gói tin qua mạng, thậm chí qua một mạng của mạng (network of network). Bởi vậy nó cần phải đáp ứng với nhiều kiểu mạng và nhiều kiểu dịch vụ cung cấp bởi các mạng khác nhau. hai chức năng chủ yếu của tầng mạng là chọn đường (routing) và chuyển tiếp (relaying). Tầng mạng là quan trọng nhất khi liên kết hai loại mạng khác nhau như mạng Ethernet với mạng Token Ring khi đó phải dùng một bộ tìm đường (quy định bởi tầng mạng) để chuyển các gói tin từ mạng này sang mạng khác và ngược lại.

Đối với một mạng chuyển mạch gói (packet switched network) gồm tập hợp các nút chuyển mạch gói nối với nhau bởi các liên kết dữ liệu. Các gói dữ liệu được truyền từ một hệ thống mở tới một hệ thống mở khác trên mạng phải được chuyển qua một chuỗi các nút. Mỗi nút nhận gói dữ liệu từ một đường vào (incoming link) rồi chuyển tiếp nó tới một đường ra (outgoing link) hướng đến đích của dữ liệu. Như vậy ở mỗi nút trung gian nó phải thực hiện các chức năng chọn đường và chuyển tiếp.

Việc chọn đường là sự lựa chọn một con đường để truyền một đơn vị dữ liệu (một gói tin chẳng hạn) từ trạm nguồn tới trạm đích của nó. Một kỹ thuật chọn đường phải thực hiện hai chức năng chính sau đây:

Quyết định chọn đường tối ưu dựa trên các thông tin đã có về mạng tại thời điểm đó thông qua những tiêu chuẩn tối ưu nhất định.

Cập nhật các thông tin về mạng, tức là thông tin dùng cho việc chọn đường, trên mạng luôn có sự thay đổi thường xuyên nên việc cập nhật là việc cần thiết.




*Hình 1.5: Mô hình chuyển vận các gói tin trong mạng chuyển mạch gói*

Có hai phương thức đáp ứng cho việc chọn đường là phương thức xử lý tập trung và xử lý tại chỗ.

Phương thức chọn đường xử lý tập trung được đặc trưng bởi sự tồn tại của một (hoặc vài) trung tâm điều khiển mạng, chúng thực hiện việc lập ra các bảng chọn đường đi tại từng thời điểm cho các nút và sau đó gửi các bảng chọn đường đi tới từng nút dọc theo con đường đã được chọn đó. Thông tin tổng thể của mạng cần dùng cho việc chọn đường chỉ cần cập nhập và được cất giữ tại trung tâm điều khiển mạng.

Phương thức chọn đường xử lý tại chỗ được đặc trưng bởi việc chọn đường được thực hiện tại mỗi nút của mạng. Trong từng thời điểm, mỗi nút phải duy trì các thông tin của mạng và tự xây dựng bảng chọn đường cho mình. Như vậy các thông tin tổng thể của mạng cần dùng cho việc chọn đường cần cập nhập và được cất giữ tại mỗi nút.

Thông thường các thông tin được đo lường và sử dụng cho việc chọn đường bao gồm:

- Trạng thái của đường truyền.

- Thời gian trễ khi truyền trên mỗi đường dẫn.

- Mức độ lưu thông trên mỗi đường.

- Các tài nguyên khả dụng của mạng.

Khi có sự thay đổi trên mạng (ví dụ thay đổi về cấu trúc của mạng do sự cố tại một vài nút, phục hồi của một nút mạng, nối thêm một nút mới... hoặc thay đổi về

mức độ lưu thông) các thông tin trên cần được cập nhật vào các cơ sở dữ liệu về trạng thái của mạng.

Hiện nay khi nhu cầu truyền thông đa phương tiện (tích hợp dữ liệu văn bản, đồ hoạ, hình ảnh, âm thanh) ngày càng phát triển đòi hỏi các công nghệ truyền dẫn tốc độ cao nên việc phát triển các hệ thống chọn đường tốc độ cao đang rất được quan tâm.

***Tầng 4: Vận chuyển (Transport)***

Tầng vận chuyển cung cấp các chức năng cần thiết giữa tầng mạng và các tầng trên. Nó là tầng cao nhất có liên quan đến các giao thức trao đổi dữ liệu giữa các hệ thống mở. Nó cùng các tầng dưới cung cấp cho người sử dụng các phục vụ vận chuyển.

Tầng vận chuyển (transport layer) là tầng cơ sở mà ở đó một máy tính của mạng chia sẻ thông tin với một máy khác. Tầng vận chuyển đồng nhất mỗi trạm bằng một địa chỉ duy nhất và quản lý sự kết nối giữa các trạm. Tầng vận chuyển cũng chia các gói tin lớn thành các gói tin nhỏ hơn trước khi gửi đi. Thông thường tầng vận chuyển đánh số các gói tin và đảm bảo chúng chuyển theo đúng thứ tự.

Tầng vận chuyển là tầng cuối cùng chịu trách nhiệm về mức độ an toàn trong truyền dữ liệu nên giao thức tầng vận chuyển phụ thuộc rất nhiều vào bản chất của tầng mạng. Người ta chia giao thức tầng mạng thành các loại sau:

Mạng loại A: Có tỷ suất lỗi và sự cố có báo hiệu chấp nhận được (tức là chất lượng chấp nhận được). Các gói tin được giả thiết là không bị mất. Tầng vận chuyển không cần cung cấp các dịch vụ phục hồi hoặc sắp xếp thứ tự lại.

Mạng loại B: Có tỷ suất lỗi chấp nhận được nhưng tỷ suất sự cố có báo hiệu lại không chấp nhận được. Tầng vận chuyển phải có khả năng phục hồi lại khi xẩy ra sự cố.

Mạng loại C: Có tỷ suất lỗi không chấp nhận được (không tin cậy) hay là giao thức không liên kết. Tầng vận chuyển phải có khả năng phục hồi lại khi xảy ra lỗi và sắp xếp lại thứ tự các gói tin.

Trên cơ sở loại giao thức tầng mạng chúng ta có 5 lớp giao thức tầng vận chuyển đó là:

Giao thức lớp 0 (Simple Class lớp đơn giản): cung cấp các khả năng rất đơn giản để thiết lập liên kết, truyền dữ liệu và hủy bỏ liên kết trên mạng "có liên kết" loại A. Nó có khả năng phát hiện và báo hiệu các lỗi nhưng không có khả năng phục hồi.

Giao thức lớp 1 (Basic Error Recovery Class: Lớp phục hồi lỗi cơ bản) dùng với các loại mạng B, ở đây các gói tin (TPDU) được đánh số. Ngoài ra giao thức

còn có khả năng báo nhận cho nơi gửi và truyền dữ liệu khẩn. So với giao thức lớp 0 giao thức lớp 1 có thêm khả năng phục hồi lỗi.

Giao thức lớp 2 (Multiplexing Class: lớp dồn kênh) là một cải tiến của lớp 0 cho phép dồn một số liên kết vận chuyển vào một liên kết mạng duy nhất, đồng thời có thể kiểm soát luồng dữ liệu để tránh tắc nghẽn. Giao thức lớp 2 không có khả năng phát hiện và phục hồi lỗi. Do vậy nó cần đặt trên một tầng mạng loại A.

Giao thức lớp 3 (Error Recovery and Multiplexing Class: Lớp phục hồi lỗi cơ bản và dồn kênh) là sự mở rộng giao thức lớp 2 với khả năng phát hiện và phục hồi lỗi, nó cần đặt trên một tầng mạng loại B.

Giao thức lớp 4 (Error Detection and Recovery Class: Lớp phát hiện và phục hồi lỗi) là lớp có hầu hết các chức năng của các lớp trước và còn bổ sung thêm một số khả năng khác để kiểm soát việc truyền dữ liệu.

***Tầng 5: Phiên(Session)***

Tầng phiên (session layer) thiết lập "các giao dịch" giữa các trạm trên mạng, nó đặt tên nhất quán cho mọi thành phần muốn đối thoại với nhau và lập ánh xa giữa các tên với địa chỉ của chúng. Một giao dịch phải được thiết lập trước khi dữ liệu được truyền trên mạng, tầng phiên đảm bảo cho các giao dịch được thiết lập và duy trì theo đúng qui định.

Tầng phiên còn cung cấp cho người sử dụng các chức năng cần thiết để quản trị các giao dịnh ứng dụng của họ, cụ thể là:

Điều phối việc trao đổi dữ liệu giữa các ứng dụng bằng cách thiết lập và giải phóng (một cách lôgic) các phiên (hay còn gọi là các hội thoại dialogues)

Cung cấp các điểm đồng bộ để kiểm soát việc trao đổi dữ liệu.

Áp đặt các qui tắc cho các tương tác giữa các ứng dụng của người sử dụng.

Cung cấp cơ chế "lấy lượt" (nắm quyền) trong quá trình trao đổi dữ liệu.

Trong trường hợp mạng là hai chiều luân phiên thì nẩy sinh vấn đề: hai người sử dụng luân phiên phải "lấy lượt" để truyền dữ liệu. Tầng phiên duy trì tương tác luân phiên bằng cách báo cho mỗi người sử dụng khi đến lượt họ được truyền dữ liệu. Vấn đề đồng bộ hóa trong tầng phiên cũng được thực hiện như cơ chế kiểm tra/phục hồi, dịch vụ này cho phép người sử dụng xác định các điểm đồng bộ hóa trong dòng dữ liệu đang chuyển vận và khi cần thiết có thể khôi phục việc hội thoại bắt đầu từ một trong các điểm đó

Ở một thời điểm chỉ có một người sử dụng đó quyền đặc biệt được gọi các dịch vụ nhất định của tầng phiên, việc phân bổ các quyền này thông qua trao đổi thẻ bài (token). Ví dụ: Ai có được token sẽ có quyền truyền dữ liệu, và khi người

giữ token trao token cho người khác thì cũng có nghĩa trao quyền truyền dữ liệu cho người đó.

Tầng phiên có các hàm cơ bản sau:

Give Token: cho phép người sử dụng chuyển một token cho một người sử dụng khác của một liên kết phiên.

Please Token: cho phép một người sử dụng chưa có token có thể yêu cầu token đó.

Give Control: dùng để chuyển tất cả các token từ một người sử dụng sang một người sử dụng khác.

***Tầng 6: Trình bày (Presentation)***

Trong giao tiếp giữa các ứng dụng thông qua mạng với cùng một dữ liệu có thể có nhiều cách biểu diễn khác nhau. Thông thường dạng biểu diễn dùng bởi ứng dụng nguồn và dạng biểu diễn dùng bởi ứng dụng đích có thể khác nhau do các ứng dụng được chạy trên các hệ thống hoàn toàn khác nhau (như hệ máy Intel và hệ máy Motorola). Tầng trình bày (Presentation layer) phải chịu trách nhiệm chuyển đổi dữ liệu gửi đi trên mạng từ một loại biểu diễn này sang một loại khác. Để đạt được điều đó nó cung cấp một dạng biểu diễn chung dùng để truyền thông và cho phép chuyển đổi từ dạng biểu diễn cục bộ sang biểu diễn chung và ngược lại.

Tầng trình bày cũng có thể được dùng kĩ thuật mã hóa để xáo trộn các dữ liệu trước khi được truyền đi và giải mã ở đầu đến để bảo mật. Ngoài ra tầng trình bày cũng có thể dùng các kĩ thuật nén sao cho chỉ cần một ít byte dữ liệu để thể hiện thông tin khi nó được truyền ở trên mạng, ở đầu nhận, tầng trình bày giải nén để được dữ liệu ban đầu.

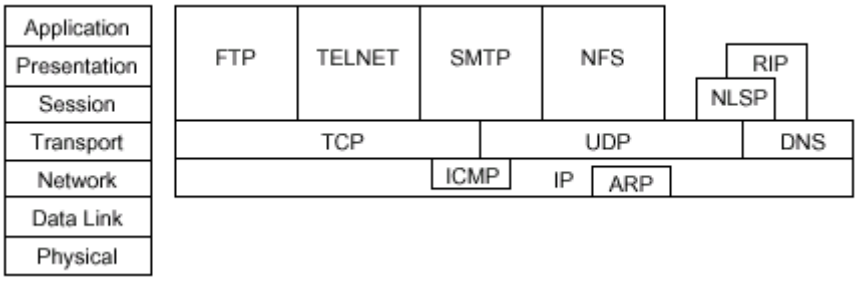
***Tầng 7: Ứng dụng (Application)***

Tầng ứng dụng (Application layer) là tầng cao nhất của mô hình OSI, nó xác định giao diện giữa người sử dụng và môi trường OSI và giải quyết các kỹ thuật mà các chương trình ứng dụng dùng để giao tiếp với mạng.

Để cung cấp phương tiện truy nhập môi trường OSI cho các tiến trình ứng dụng, Người ta thiết lập các thực thể ứng dụng (AE), các thực thể ứng dụng sẽ gọi đến các phần tử dịch vụ ứng dụng (Application Service Element viết tắt là ASE) của chúng. Mỗi thực thể ứng dụng có thể gồm một hoặc nhiều các phần tử dịch vụ ứng dụng. Các phần tử dịch vụ ứng dụng được phối hợp trong môi trường của thực thể ứng dụng thông qua các liên kết (association) gọi là đối tượng liên kết đơn (Single Association Object viết tắt là SAO). SAO điều khiển việc truyền thông

trong suốt vòng đời của liên kết đó cho phép tuần tự hóa các sự kiện đến từ các ASE thành tố của nó.

*1.2.2.3. Tham chiếu giữa TCP/IP và mô hình OSI 7 tầng*




*Hình 1.6: Mô hình TCP/IP và các tầng tương đương trong OSI 7 tầng*

Như vậy, theo mô hình tham chiếu với OSI 7 tầng thì TCP tương ứng với tầng 4 cộng thêm một số chức năng của tầng 5 trong giao thức OSI. Còn IP tương ứng với tầng 3 của mô hình OSI.

Trong cấu trúc bốn tầng của TCP/IP, khi dữ liệu truyền từ tầng ứng dụng cho đến tầng vật lý, mỗi tầng đều cộng thêm vào phần điều khiển của mình để đảm bảo cho việc truyền dữ liệu được chính xác. Mỗi thông tin điều khiển này được gọi là một header và được đặt ở trước phần dữ liệu được truyền. Mỗi tầng xem tất cả các thông tin mà nó nhận được từ tầng trên là dữ liệu, và đặt phần thông tin điều khiển header của nó vào trước phần thông tin này. Việc cộng thêm vào các header ở mỗi tầng trong quá trình truyền tin được gọi là encapsulation. Quá trình nhận dữ liệu diễn ra theo chiều ngược lại: mỗi tầng sẽ tách ra phần header trước khi truyền dữ liệu lên tầng trên.

Mỗi tầng có một cấu trúc dữ liệu riêng, độc lập với cấu trúc dữ liệu được dùng ở tầng trên hay tầng dưới của nó. Sau đây là giải thích một số khái niệm thường gặp.

- Stream là dòng số liệu được truyền trên cơ sở đơn vị số liệu là Byte.
- Số liệu được trao đổi giữa các ứng dụng dùng TCP được gọi là stream, trong khi dùng UDP, chúng được gọi là message.
- Mỗi gói số liệu TCP được gọi là segment còn UDP định nghĩa cấu trúc dữ liệu của nó là packet.
- Tầng Internet xem tất cả các dữ liệu như là các khối và gọi là datagram. Bộ giao thức TCP/IP có thể dùng nhiều kiểu khác nhau của tầng mạng dưới cùng, mỗi loại có thể có một thuật ngữ khác nhau để truyền dữ liệu.
- Phần lớn các mạng kết cấu phần dữ liệu truyền đi dưới dạng các packets hay là các frames.

| | |
|---|---|
| Ứng dụng (Application) | Stream |
| Vận chuyển (Transport) | Segment/datagram |
| Liên mạng (Internet) | Datagram |
| Truy nhập mạng (Network Access) | Frame |

*Hình 1.7: Cấu trúc dữ liệu tại các tầng của TCP/IP*

***Tầng truy nhập mạng (Network Access)***

Network Access là tầng thấp nhất trong cấu trúc phân bậc của TCP/IP. Những giao thức ở tầng này cung cấp cho hệ thống phương thức để truyền dữ liệu trên các tầng vật lý khác nhau của mạng. Nó định nghĩa cách thức truyền các khối dữ liệu (datagram) IP. Các giao thức ở tầng này phải biết chi tiết các phần cấu trúc vật lý mạng ở dưới nó (bao gồm cấu trúc gói số liệu, cấu trúc địa chỉ...) để định dạng được chính xác các gói dữ liệu sẽ được truyền trong từng loại mạng cụ thể.

So sánh với cấu trúc OSI/OSI, tầng này của TCP/IP tương đương với hai tầng Datalink, và Physical.

Chức năng định dạng dữ liệu sẽ được truyền ở tầng này bao gồm việc nhúng các gói dữ liệu IP vào các frame sẽ được truyền trên mạng và việc ánh xạ các địa chỉ IP vào địa chỉ vật lý được dùng cho mạng.

***Tầng liên mạng (Internet)***

Internet là tầng ở ngay trên tầng Network Access trong cấu trúc phân tầng của TCP/IP. Internet Protocol là giao thức trung tâm của TCP/IP và là phần quan trọng nhất của tầng Internet, cụ thể các chức năng của nó như sau

- Định nghĩa cấu trúc các gói dữ liệu là đơn vị cơ sở cho việc truyền dữ liệu trên Internet.
- Định nghĩa phương thức đánh địa chỉ IP.
- Truyền dữ liệu giữa tầng vận chuyển và tầng mạng.
- Định tuyến để chuyển các gói dữ liệu trong mạng.
- Thực hiện việc phân mảnh và hợp nhất (fragmentation reassembly) các gói dữ liệu và nhúng/tách chúng trong các gói dữ liệu ở tầng liên kết.

***Tầng vận chuyển***

TCP cung cấp khả năng truyền dữ liệu một cách an toàn giữa các máy trạm trong hệ thống các mạng. Nó cung cấp thêm các chức năng nhằm kiểm tra tính chính xác của dữ liệu khi đến và bao gồm cả việc gửi lại dữ liệu khi có lỗi xảy ra. TCP cung cấp các chức năng chính sau:

- Thiết lập, duy trì, kết thúc liên kết giữa hai quá trình.
- Phân phát gói tin một cách tin cậy.
- Đánh thứ tự các gói dữ liệu nhằm truyền dữ liệu một cách tin cậy.
- Cho phép điều khiển lỗi.
- Cung cấp khả năng đa kết nối với các quá trình khác nhau giữa trạm nguồn và trạm đích nhất định thông qua việc sử dụng các cổng.
- Truyền dữ liệu sử dụng cơ chế song công (full duplex).

**Tầng ứng dụng**

Bao gồm các ứng dụng chạy trên nền giao thức TCP/IP, các giao thức ứng dụng phổ biến là:

http: dịch vụ web

smtp, pop: dịch vụ email

ftp: dịch vụ truyền tệp

telnet: dịch vụ truy cập từ xa

rtp: dịch vụ truyền voice và video qua mạng Internet

...và cùng với sự phát triển rất mạnh mẽ của Internet, các giao thức ứng dụng mới liên tục ra đời

*1.2.2.4. Các thành phần của khuôn dữ liệu (frame) tầng vật lý*

Như ta đã thấy ở phần trên, dữ liệu khi truyền ngang qua mạng được phân tách thành những khối nhỏ, có kích thước phụ thuộc vào hình trạng lôgíc của mạng đó. Như đối với mạng Ethernet không thể sử dụng các khối dữ liệu lớn hơn 1500 byte. Các khối dữ liệu nhỏ này được gọi là các frame (khung hoặc khuôn dạng).

Có hai loại frame cơ bản truyền trong các mạng cụ bộ là: Ethernet và Token Ring – tương ứng với tên hai loại mạng được sử dụng thông thường nhất.

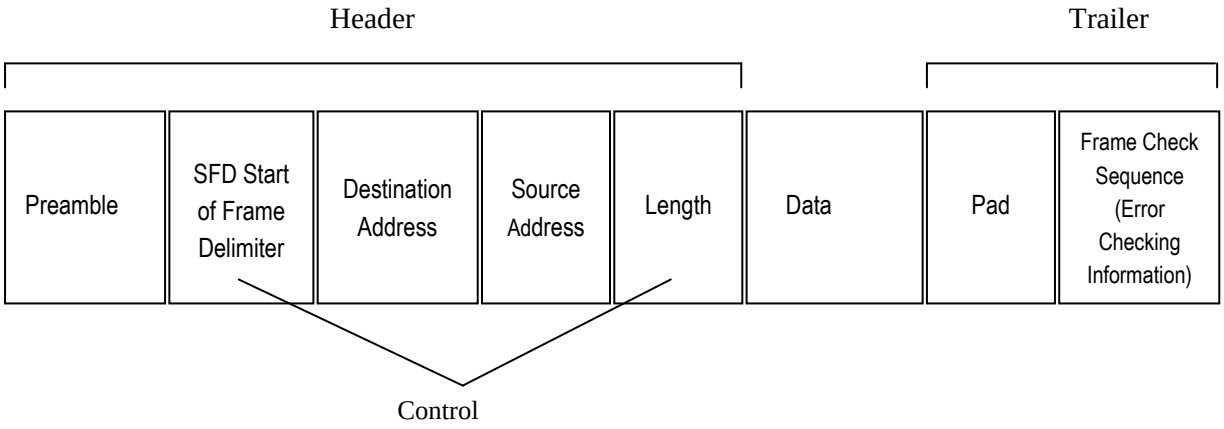
Lịch sử của mạng Ethernet bắt đầu từ khi công ty Xerox Corporation phát triển một tiêu chuẩn đơn giản cho Ethernet vào năm 1970, sau đó do sự liên kết và hợp chuẩn giữa Xerox Corp. với DEC và Intel, Ethernet đã được cải tiến và trở thành một chuẩn phổ biến nhất cho các hệ thống mạng máy tinh, hiện nay có đến 5 công nghệ Ethernet chủ yếu đang được sử dụng: 10Base2, 10Base5, 10BaseT, 100BaseT và 1000BaseT.

Token Ring đã được phát triển bởi IBM vào năm 1980 và dựa trên liên kết giữa các nút với công nghệ vòng (ring): một thẻ bài (token) được truyền quanh các nút. Một nút chỉ có thể truyền dữ liệu trên mạng sau khi nó nhận được thẻ bài. Cáp

nối mạng hình thành một vòng (ring hoặc circle) và các tín hiệu dữ liệu được truyền chỉ theo một hướng quanh vòng.

Mặc dù về lý thuyết có thể truyền cả hai frame Ethernet và Token Ring trên cùng một mạng, nhưng điều này không thực hiện trong thực tế. Giao tiếp Ethernet không thể phiên dịch các frame Token Ring và trái lại. Một mạng luôn chỉ là Ethernet hoặc Token Ring chứ không thể đồng thời cả hai. Tuy nhiên có thể kết hợp các giao thức trên cùng trên một mạng. Chẳng hạn, có thể sử dụng cả hai bộ giao thức TCP/IP và IPX/SPX trên mạng mạng Ethernet, vì cả hai giao thức này cùng sử dụng một kiểu frame dữ liệu.

*a. Khuôn dữ liệu Etherne*



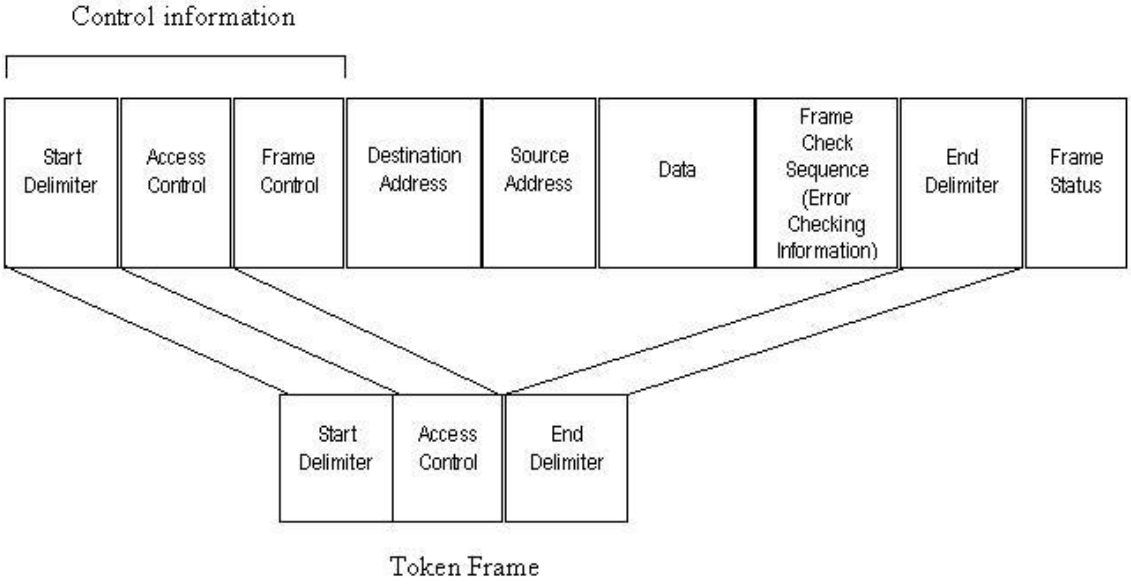

*Hình 1.8: Cấu trúc Frame dữ liệu Ethernet*

Các thành phần của frame Ethernet 802.3 bao gồm:

- Preamble (Phần mở đầu)– Đánh dấu bắt đầu của toàn bộ frame, là tín hiệu thông báo tới mạng rằng dữ liệu đang truyền. (Vì trường này là một phần của quá trình giao tiếp, nên nó không được tính vào kích thước của frame)
- Start of Frame Delimiter (SFD) – Chứa thônng tin khởi đầu của việc định địa chỉ frame.
- Destination Address – Chứa địa chỉ của nút đích.
- Source Address – Chứa địa chỉ của nút nguồn.
- Length (LEN) – Chứa chiều dài của gói.
- Data – Chứa dữ liệu được truyền từ nút nguồn.
- Pad – Được sử dụng để tăng kích thước của frame tới kích thước yêu cầu nhỏ nhất là 46 byte.

- Frame Check Sequence (FCS) – Cung cấp một giải thuật để xác định xem dữ liệu nhận được có chính xác hay không. Giải thuật được sử dụng thông thường nhất là bộ mã sửa sai Cyclic Redundancy Check (CRC).

*b. Khuôn dữ liệu Token Ring*




*Hình 1.9: Cấu trúc Frame dữ liệu Token Ring*

Các thành phần của frame Token ring 802.5 bao gồm:

Start Delimiter (SD) – Báo hiệu bắt đầu gói. Nó là một trong ba trường tạo thành khuôn dạng Token Ring.

Access Control (AC) – Chứa thông tin về độ ưu tiên của frame. Nó là trường thứ hai tạo thành khuôn dạng Token Ring.

Frame Control (FC) – Định nghĩa kiểu của frame, được dùng trong Frame Check Sequence.

Destination Address – Chứa địa chỉ của nút đích.

Source Address – Chứa địa chỉ của nút nguồn.

Data – Chứa dữ liệu được truyền từ nút nguồn, cũng có thể chứa thông tin quản lý và tìm đường.

Frame Check Sequence (FCS) – Được sử dụng để kiểm tra tính toàn vẹn của frame.

End Delimiter (ED) – Báo hiệu kết thúc frame. Nó là trường thứ ba của khuôn dạng Token Ring.

Frame Status (FS) – Báo hiệu nút đích nhận dạng và sao chép đúng frame hay không.

**Câu hỏi và Bài tập**

1.1. Cấu trúc cơ bản của liên mạng Internet?

1.2. Lịch sử ra đời của giao thức TCP/IP, vai trò của giao thức này?

1.3. Tính năng của các Router trong liên mạng Internet?

1.4. Vai trò của các Router trên mạng Internet?

1.5. Tham chiếu giữa mô hình TCP/IP và mô hình OSI?

1.6. Chức năng từng tầng trong mô hình OSI?

1.7. Cấu trúc các Frame dữ liệu tầng vật lý của mạng Ethernet và Token Ring?

1.8. Sử dụng lệnh TRACERT để dò vết các gói dữ liệu gửi từ máy tính của sinh viên đến một máy chủ bất kỳ trên Internet?

1.9. Hay trình bày mô hình cơ bản của các dịch vụ phổ biến trên mạng Internet: WWW và Email?

# CHƯƠNG 2 CẤU TRÚC IP DATAGRAM

## 2.1. Cấu trúc IP DATAGRAM

### 2.1.1. Khái niệm chuyển phát phi kết nối (Connectionless)

Chúng ta đã biết, khái quát một cách cơ bản nhất thì mạng Internet thực chất là một hệ thống khổng lồ làm nhiệm vụ chuyển phát các gói dữ liệu. Về mặt kỹ thuật, dịch vụ này được định nghĩa như là dịch vụ không có độ tin cậy (unreliable), hệ chuyển phát Connectionless tương tự như dịch vụ cung cấp bởi phần cứng mạng mà hoạt động trên mô hình nỗ lực tối đa (best effort).

- Dịch vụ này được gọi là không có độ tin cậy vì việc chuyển phát không được bảo đảm gói dữ liệu chắc chắn tới đích 100%. Gói dữ liệu có thể bị thất lạc, bị trùng lập, bị chuyển chậm, hoặc chuyển đi không theo đúng thứ tự, những dịch vụ này không nhận ra được những sự việc này, và cũng không thông báo nơi gửi hoặc nơi nhận.

- Dịch vụ này được gọi là connectionless vì mỗi gói dữ liệu được xử lý độc lập với gói khác, gần như không có sự liên kết giữa các gói dữ liệu với nhau. Một chuỗi các gói dữ liệu gửi từ một máy tới máy khác có thể di chuyển theo những con đường khác nhau, hoặc một số bị mất trong khi một số khác vẫn đến đích được.

- Dịch vụ này còn được được gọi là chuyển phát nỗ lực tối đa là vì phần mềm của hệ thống Internet thực hiện một cố gắng lớn nhất để chuyển phát các gói. Nghĩa là, Internet không bỏ sót/làm mất các gói dữ liệu một cách bất thường; "không có độ tin cậy" ở đây chỉ xuất hiện khi các dữ liệu truyền quá tải hoặc cơ sở hạ tầng mạng có vấn đề.

### 2.1.2. Mục đích của giao thức IP (Internet Protocol)

Giao thức mà xác định cơ chế chuyển phát Connectionless, không có độ tin cậy, được gọi là Internet Protocol và thường được gọi tắt là IP.

Nhiệm vụ chính của giao thức IP là cung cấp khả năng kết nối các mạng con thành liên kết mạng để truyền dữ liệu, vai trò của IP là vai trò của giao thức tầng mạng trong mô hình OSI. Giao thức IP là một giao thức kiểu không liên kết (connectionless) có nghĩa là không cần có giai đoạn thiết lập liên kết trước khi truyền dữ liệu.

Giao thức IP có 3 nguyên lý quan trọng là:

- Trước hết, giao thức IP định nghĩa đơn vị cơ sở của việc truyền dữ liệu được sử dụng thông qua một TCP/IP Internet. Như thế, nó xác định định dạng chính xác của tất cả dữ liệu khi nó đi qua Internet.

- Thứ hai phần mềm IP thực hiện chức năng định tuyến (routing), chọn một con đường mà dữ liệu sẽ được gửi đi.

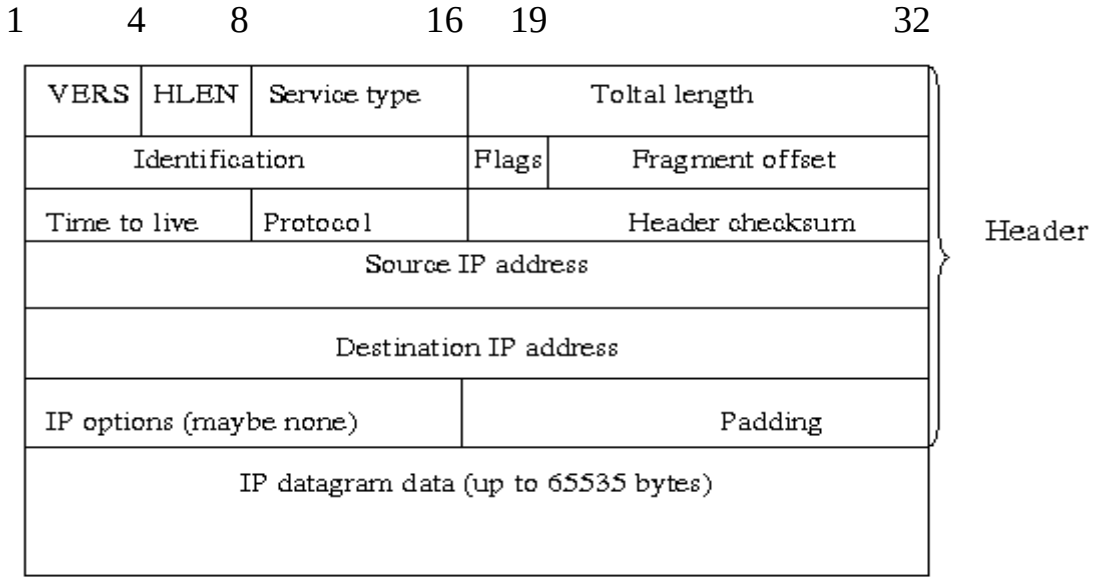
- Thứ ba, các quy tắc về kiểm tra độ chính xác, đặc tả chính thức của định dạng dữ liệu và việc định tuyến, ngoài ra IP còn bao gồm một tập hợp các quy tắc mà thể hiện ý tưởng của hệ chuyển phát dữ liệu không có độ tin cậy.

Các quy tắc này đặc trưng cho cách mà máy tính và bộ định tuyến xử lý truyền các gói dữ liệu, xử lý khi gặp lỗi làm thế nào và khi nào các thông điệp lỗi được phát sinh, cũng như định nghĩa dưới những điều kiện nào thì các gói dữ liệu được huỷ bỏ.

IP là một phần rất cơ bản và quan trọng nhất của thiết kế đến nỗi TCP/IP Internet đôi khi được gọi là một kỹ thuật dựa trên IP.

### 2.1.3. Cấu trúc gói dữ liệu IP Datagram (Internet datagram)

Có sự tương đồng lớn giữa một mạng vật lý và một mạng TCP/IP Internet. Trên một mạng vật lý, đơn vị truyền dữ liệu là một frame bao gồm phần đầu (header) và phần dữ liệu, với phần đầu cung cấp các thông tin như địa chỉ nguồn và đỉa chỉ đích (vật lý). Internet gọi đơn vị truyền dữ liệu của nó là một Internet datagram, và thường được gọi là IP Datagram, hoặc đơn giản là datagram. Cũng giống như một frame trong mạng vật lý, một datagram bao gồm hai phần, phần đầu và phần dữ liệu. Phần đầu của datagram bao gồm địa chỉ nguồn và đích và một vùng kiểu để xác định nội dung của datagram bao gồm các địa chỉ IP trong khi phần đầu của frame bao gồm các địa chỉ vật lý. Dạng tổng quát của datagram được trình bày trong hình 2.1




*Hình 2.1: Cấu trúc IP Datagram*

Ý nghĩa của thông số như sau:

VER (4 bits): chỉ version hiện hành của giao thức IP hiện được cài đặt, việc có chỉ số version cho phép có các trao đổi giữa các hệ thống sử dụng version cũ và hệ thống sử dụng version mới, version phổ biến hiện nay là version 4.

HLEN (4 bits): chỉ độ dài phần đầu (Internet header Length) của gói tin datagram, tính theo đơn vị từ (32 bits). Trường này bắt buột phải có vì phần đầu IP có thể có độ dài thay đổi tùy ý. Độ dài tối thiểu là 5 từ (20 bytes), độ dài tối đa là 15 từ hay là 60 bytes.

Type of service (8 bits): đặc tả các tham số về dịch vụ nhằm thông báo cho mạng biết dịch vụ nào mà gói tin muốn được sử dụng, chẳng hạn ưu tiên, thời hạn chậm trễ, năng suất truyền và độ tin cậy. Hình sau cho biết ý nghĩ của trường 8 bits này.

| | | | | |
|---|---|---|---|---|
| Precedence | D | T | R | Unused |

Trong đó:

Precedence (3 bits)*:* chỉ thị về quyền ưu tiên gửi datagram, cụ thể là:

| | | | |
|---|---|---|---|
| 111 | Network Control (cao nhất) | 011 | Flash |
| 110 | Internetwork Control | 010 | Immediate |
| 101 | CRITIC/ECP | 001 | Priority |
| 100 | Flas Override | 000 | Routine (thấp nhất) |

D (delay) (1 bit): chỉ độ trễ yêu cầu

D=0 độ trễ bình thường

D=1 độ trễ thấp

T (Throughput) (1 bit): chỉ số thông lượng yêu cầu

T=1 thông lượng bình thường

T=1 thông lượng cao

R (Reliability): (1 bit): chỉ độ tin cậy yêu cầu

R=0 độ tin cậy bình thường

R=1 độ tin cậy cao

Total Length (16 bits): chỉ độ dài toàn bộ gói tin, kể cả phần đầu tính theo đơn vị byte với chiều dài tối đa là 65535 bytes. Hiện nay giới hạn trên là rất lớn nhưng trong tương lai với những mạng Gigabit thì các gói tin có kích thước lớn là cần thiết.

Identification (16 bits): cùng với các tham số khác (như Source Address và Destination Address) tham số này dùng để định danh duy nhất cho một datagram trong khoảng thời gian nó vẫn còn trên liên mạng.

Flags (3 bits): liên quan đến sự phân đoạn (fragment) các datagram, Các gói tin khi đi trên đường đi có thể bị phân thành nhiều gói tin nhỏ, trong trường hợp bị phân đoạn thì trường Flags được dùng điều khiển phân đoạn và tái lắp ghép bó dữ liệu. Tùy theo giá trị của Flags sẽ có ý nghĩa là gói tin sẽ không phân đoạn, có thể phân đoạn hay là gói tin phân đoạn cuối cùng. Trường Fragment Offset cho biết vị trí dữ liệu thuộc phân đoạn tương ứng với đoạn bắt đầu của gói dữ liệu gốc. Ý nghĩa cụ thể của trường Flags là:

| | | |
|---|---|---|
| O | DF | MF |

bit 0: reserved chưa sử dụng, luôn lấy giá trị 0.

bit 1: (DF) = 0 (May Fragment) = 1 (Don't Fragment)

bit 2: (MF) = 0 (Last Fragment) = 1 (More Fragments)

Fragment Offset (13 bits): chỉ vị trí của đoạn (fragment) ở trong datagram tính theo đơn vị 8 bytes, có nghĩa là phần dữ liệu mỗi gói tin (trừ gói tin cuối cùng) phải chứa một vùng dữ liệu có độ dài là bội số của 8 bytes. Điều này có ý nghĩa là phải nhân giá trị của Fragment offset với 8 để tính ra độ lệch bytes.

Time to Live (8 bits): qui định thời gian tồn tại (tính bằng số lần qua các router trung gian trên đường đi) của gói tin trong mạng để tránh tình trạng một gói tin bị quẩn trên mạng. Thời gian này được cho bởi trạm gửi và được giảm đi (thường qui ước là 1 đơn vị) khi datagram đi qua mỗi router của liên mạng. Thời lượng này giảm xuống tại mỗi router với mục đích giới hạn thời gian tồn tại của các gói tin và kết thúc những lần lặp lại vô hạn trên mạng. Sau đây là 1 số điều cần lưu ý về trường Time To Live:

Router nào nhận được gói tin có giá trị trường này bằng 0 thì nó sẽ drop gói tin và báo cho trạm gửi biết.

Một giao thức có thể ấn định Time To Live để thực hiện cuộc ra tìm tài nguyên trên mạng trong phạm vi mở rộng.

Protocol (8 bits): chỉ giao thức tầng kế tiếp sẽ nhận vùng dữ liệu ở trạm đích (hiện tại thường là TCP hoặc UDP được cài đặt trên IP).

Header checksum (16 bits): mã kiểm soát lỗi sử dụng phương pháp CRC (Cyclic Redundancy Check) dùng để đảm bảo thông tin về gói dữ liệu được truyền đi một cách chính xác (mặc dù dữ liệu có thể bị lỗi). Nếu như việc kiểm tra này thất bại, gói dữ liệu sẽ bị huỷ bỏ tại nơi xác định được lỗi. Cần chú ý là IP không

cung cấp một phương tiện truyền tin cậy bởi nó không cung cấp cho ta một cơ chế để xác nhận dữ liệu truyền tại điểm nhận hoặc tại những điểm trung gian. Giao thức IP không có cơ chế Error Control cho dữ liệu truyền đi, không có cơ chế kiểm soát luồng dữ liệu (flow control).

Source Address (32 bits): địa chỉ của trạm nguồn.

Destination Address (32 bits): địa chỉ của trạm đích.

Option (có độ dài thay đổi): sử dụng trong một số trường hợp, nhưng thực tế chúng rất ít dùng. Trường option sử dụng trong một số chức năng định tuyến đặc biệt.

Padding (độ dài thay đổi): vùng đệm, được dùng để đảm bảo cho phần header luôn kết thúc ở một mốc 32 bits

Data (độ dài thay đổi): vùng dữ liệu có độ dài là bội của 8 bits, tối đa là 65535 bytes.

### 2.1.4. Thời gian sống (Time to Live – TTL)

Về nguyên lý, vùng Time to live xác định khoảng thời gian mà datagram được phép lưu lại trên hệ thống mạng. Ý tưởng này vừa đơn giản vừa quan trọng: bất cứ khi nào một máy tính đưa một datagram vào Internet, nó thiết lập thời gian tối đa mà datagram đó được tồn tại. Bộ định tuyến sẽ xử lý datagram này giảm dần giá trị vùng Time to live khi thời gian trôi qua và huỷ bỏ nó khỏi Internet khi đã hết hạn.

Việc đánh giá chính xác thời gian là khó vì bộ định tuyến thường không biết thời gian truyền trong các mạng vật lý. Một vài quy tắt đã làm đơn giản tiến trình và dễ dàng hơn trong việc xử lý datagram mà không cần sự đồng bộ các đồng hồ. Trước hết, mỗi định tuyến dọc theo con đường từ nguồn đến đích được yêu cầu giảm giá trị vùng Time to live bớt đi 1 khi nó xử lý phần đầu datagram. Hơn nữa, để xử lý trường hợp các bộ định tuyến bị quá tải, gây ra tình trạng trì hoãn, mỗi bộ định tuyến ghi nhận thời điểm (theo đồng hồ của nó) khi datagram, đến và giảm giá trị của Time to live theo số giây mà datagram còn nằm trong bộ định tuyến (trong thực tế, những bộ định tuyến hiện đại chỉ giữ datagram tối đa trong vài giây).

Bất cứ khi nào vùng đạt đến giá trị 0, bộ định tuyến sẽ huỷ bỏ gói dữ liệu (datagram) và gửi thông báo lỗi trở về nơi xuất phát. Ý tưởng về bộ đếm thời gian cho mỗi datagram là rất quan trọng bởi vì nó bảo đảm rằng datagram không thể di chuyển trên Internet mãi mãi, ngay cả trong trường hợp bảng dữ liệu trong bộ định tuyến bị hỏng cũng như trường hợp chúng tạo thành vòng lặp vô hạn.

Mặc dù vậy, ý niệm về việc bộ định tuyến trì hoãn datagram một khoảng thời gian nhiều giây bây giờ đã lạc hậu – những bộ định tuyến và mạng hiện tại được thiết kế để chuyển datagram đi trong một khoảng thời gian hợp lý. Nếu việc trì hoãn vượt quá một thời hạn, bộ định tuyến sẽ huỷ bỏ datagram. Như vậy, trong thực tế, Time to live có vai trò như "bộ đếm ngược" chứ không phải tính thời gian trì hoãn. Một bộ định tuyến chỉ có quyền giảm giá trị đi 1 đơn vị khi datagram đi qua nó.

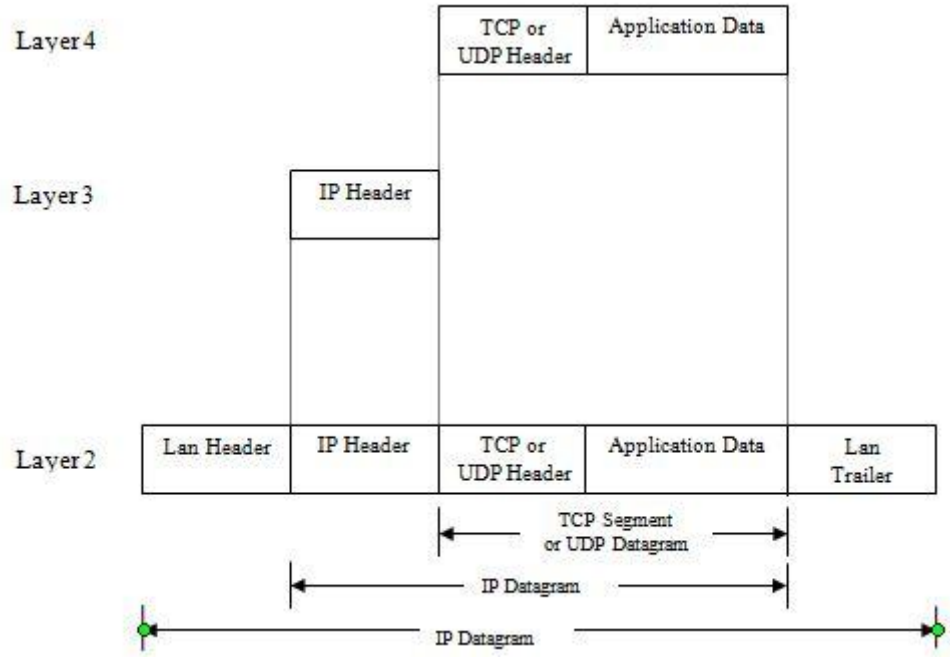
### 2.1.5. Đóng gói datagram

Một điều quan trọng trước khi chúng ta có thể hiểu các vùng kế tiếp trong một datagram, là xem xét làm cách nào các datagram liên hệ với các frame mạng vật lý. Chúng ta bắt đầu với câu hỏi: "độ lớn tối đa của một datagram là bao nhiêu?" Không như các frame mạng vật lý mà phải được nhận diện bởi phần cứng, datagram được xử lý bằng phần mềm. Chúng có thể có độ dài bất kỳ từ nguồn gửi miễn là không vượt quá số bytes tối đa cho phép.

Định dạng datagram IP v4 dành ra 16 bits cho vùng độ dài do vậy giới hạn độ dài của datagram là 65,535 bytes.

Trong thực tế, xuất hiện nhiều giới hạn cơ bản hơn về kích thước datagram. Chúng ta biết rằng khi datagram di chuyển từ máy này sang máy khác, chúng phải luôn luôn được chuyển đi bởi mạng vật lý cơ sở. Để cho việc chuyển phát trên Internet được hiệu quả, cần bảo đảm rằng mỗi datagram được chuyển tải trong một frame vật lý riêng biệt.

Ý tưởng về việc chuyển tải một datagram trong một frame mạng được gọi là sự đóng gói (encapsulation). Đối với mạng vật lý cơ sở, một datagram giống như bất kỳ một thông điệp khác gửi từ một máy tới máy khác. Phần cứng không nhận biết định dạng datagram, cũng như không cần biết đến địa chỉ IP của cả nguồn gửi và nguồn nhận datagram đó, toàn bộ datagram di chuyển trong phần dữ liệu của frame mạng (một vùng trong phần đầu frame thường xác định dữ liệu được chuyển tải; Ethernet sử dụng kiểu dữ liệu 0800 (hệ hexa), để xác định rằng vùng dữ liệu chứa một IP Datagram được đóng gói).




*Hình 2.2: Đóng gói IP Datagram vào trong Frame vật lý*

## 2.2. Kích thước và sự phân mảnh IP DATAGRAM

2.2.1. Kích thước datagram.

Trong trường hợp lý tưởng, toàn bộ IP Datagram nằm vừa vặn trong một frame vật lý, làm cho việc truyền trên mạng vật lý được hiệu quả. Để đạt được hiệu quả này, nguồn gửi có thể chọn một kích thước tối đa của datagram sao cho một datagram luôn luôn nằm vừa trong một frame. Nhưng kích thước nào của frame sẽ được chọn? Hơn nữa, một datagram có thể phải di chuyển qua nhiều loại mạng vật lý khi nó di chuyển trên Internet để tới đích, mà thông tin về các mạng vật lý trung gian thì nguồn gửi không thể có được đầy đủ.

Một yếu tố quan trọng về phần cứng mạng đó là kỹ thuật chuyển gói đặt một mốc chặn trên cố định đối với tổng số dữ liệu có thể được truyền trong một frame vật lý. Ví dụ, với mạng Ethernet giới hạn này là 1500 bytes dữ liệu, trong khi mạng cáp quang FDDI cho phép khoảng 4470 bytes dữ liệu mỗi frame (giới hạn 1500 là từ đặc tính của Ethernet; khi sử dụng với phần đầu SNA chuẩn IEEE 802.3 giới hạn là 1492 bytes. Cũng có một số nhà sản xuất cho giới hạn lớn hơn một chút).

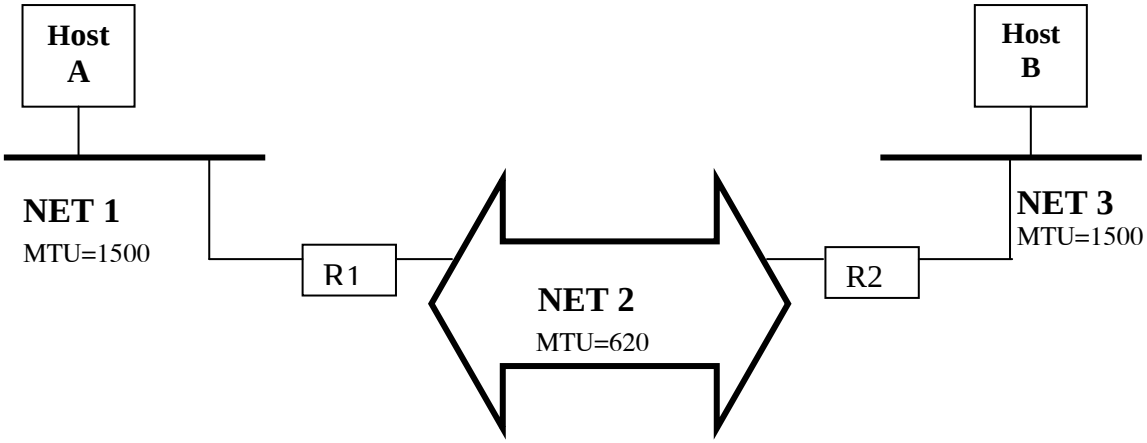
Người ta định nghĩa giới hạn này bằng tên gọi *đơn vị truyền tối đa của mạng* (Maximum Transfer Unit – MTU). Kích thước MTU cũng có thể rất nhỏ: một vài mạng viễn thông tốc độ thấp chỉ giới hạn MTU xuống 128 bytes, thậm chí còn ít hơn nữa. Việt giới hạn datagram để vừa MTU nhỏ nhất trong Internet làm cho việc truyền không được hiệu quả khi datagram đi qua một mạng có thể chuyển tải frame có kích thước lớn hơn. Tuy nhiên, việc có những datagram có kích thước lớn hơn

MTU tối thiểu của mạng vật lý trung gian trong Internet là điều không phải hiếm gặp, có nghĩa là một datagram có thể không luôn luôn đóng gó vừa vặn trong một frame mạng.

Hiển nhiên sự chọn lựa sẽ là: tiêu chí của thiết kế Internet là để che dấu kỹ thuật mạng cơ sở và làm cho việc thông tin liên lạc được tiện lợi cho người sử dụng. Như thế, thay vì thiết kế các datagram tuân theo các ràng buộc của mạng vật lý, phần mềm TCP/IP chọn một kích thước khởi đầu và kèm theo giải pháp chia các datagram lớn thành những phần nhỏ hơn khi datagram cần đi qua một mạng mà có MTU nhỏ. Những phần nhỏ của một datagram được gọi là fragments, và tiến trình chia một datagram thành các fragments được gọi là quá trình phân đoạn (fragmentation). Điều hiển nhiên là có quá trình phân đoạn thì cũng phải có quá trình hợp nhất lại gói tin nguyên thủy ban đầu.

### 2.2.2. Phân mảnh IP Datagram

Việc phân đoạn thường xảy ra tại bộ định tuyến trên đường đi từ nguồn đến đích cuối cùng. Bộ định tuyến nhận một datagram từ một mạng có MTU lớn và phải gửi nó trên một mạng có MTU nhỏ hơn kích thước datagram.




*Hình 2.3: Các mạng có MTU khác nhau*

Trong hình này, cả hai máy tính nối trực tiếp vào Ethernet có MTU 1500 bytes. Như thế, cả hai máy đều có thể gửi các datagram dài đến 1500 bytes. Tuy nhiên, con đường nối giữa chúng, là một mạng có MTU là 620. Nếu máy A gửi cho máy B một datagram lớn hơn 620 bytes, bộ định tuyến R1 sẽ phân đoạn datagram đó. Tương tự, nếu máy B gửi một datagram lớn cho máy, bộ định tuyến R2 sẽ phân đoạn datagram này.

Kích thước fragmet được chọn sao cho mọi fragment có thể được gửi qua mạng cơ sở trong một frame. Hơn nữa, vì giao thức IP xác định vị trí tương đối của phần dữ liệu trong datagram là bội của tám bytes, nên kích thước fragment phải được chọn là bội số của tám. Dĩ nhiên, việc chọn bội số của tám mà gần nhất với MTU mạng không phải luôn luôn chia datagram thành những phần bằng nhau;

thông thường mảnh cuối sẽ nhỏ hơn. Các fragment phải được kết hợp lại để có được đúng bản sao cả datagram ban đầu trước khi nó được xử lý tại đích cuối cùng.

Giao thức IP không có giới hạn nhỏ nhất cho datagram, và cũng không bảo đảm rằng các datagram lớn sẽ được chuyển mà không bị phân đoạn. Máy nguồn có thể chọn kích thước datagram bất kỳ mà nó nghĩ là thích hợp; việc phân đoạn và kết hợp lại sẽ tự động xảy ra, mà không có tác động đặc biệt gì từ máy nguồn. Đặc tả của IP phát biểu rằng bộ định tuyến phải chấp nhận datagram có kích thước bất kỳ và bộ định tuyến sẽ phải xử lý cho phù hợp với MTU của các mạng mà nó được nối vào. Việc phân đoạn một datagram có nghĩa là chia nó thành những phần nhỏ. Và mỗi phần này về cơ bản có cùng định dạng như datagram ban đầu, đa phần các thông số của datagram ban đầu được giữ nguyên.

**Original IP Datagram**

| Sequence | Identifier | Total Length | DF May / Don't | MF Last / More | Fragment Offset |
|---|---|---|---|---|---|
| 0 | 345 | 5140 | 0 | 0 | 0 |

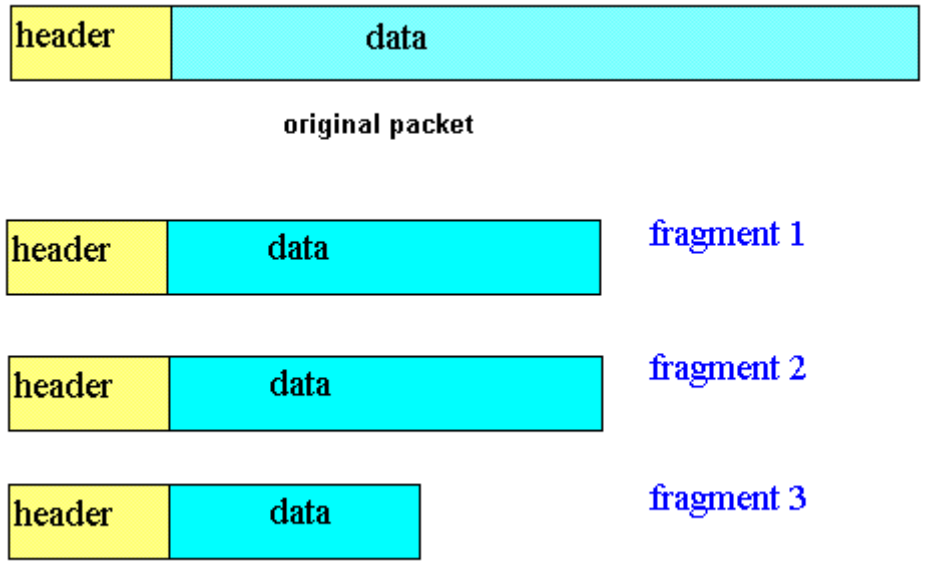
**IP Fragments (Ethernet)**

| Sequence | Identifier | Total Length | DF May / Don't | MF Last / More | Fragment Offset |
|---|---|---|---|---|---|
| 0-0 | 345 | 1500 | 0 | 1 | 0 |
| 0-1 | 345 | 1500 | 0 | 1 | 185 |
| 0-2 | 345 | 1500 | 0 | 1 | 370 |
| 0-3 | 345 | 700 | 0 | 0 | 555 |

*Hình 2.4: Ví dụ việc phân mảnh*

IP dùng cờ MF (3 bit thấp của trường Flags trong phần đầu của gói IP) và trường Flagment offset của gói IP (đã bị phân đoạn) để định danh gói IP đó là một phân đoạn và vị trí của phân đoạn này trong gói IP gốc. Các gói cùng trong chuỗi phân mảnh đều có trường này giống nhau. Cờ MF bằng 1 nếu là gói đầu của chuỗi phân mảnh và 0 nếu là gói cuối của gói đã được phân mảnh.

Mỗi fragment ban đầu datagram giống hầu hết với phần đầu của datagram ban đầu (ngoại trừ bit trong vùng FLAGS để chỉ rằng nó là một fragment), kế tiếp là phần dữ liệu có thể chuyển tải trong một fragment mà vẫn duy trì tổng độ dài nhỏ hơn MTU của mạng mà nó di chuyển trên đó.




*Hình 2.5: Ví dụ về việc phân mảnh 1:3*

2.2.3. Kết hợp các Fragment

Một câu hỏi là: Các fragment nên được kết nối hợp lại sau khi đi qua một mạng, hay các fragment nên được chuyển đến máy cuối cùng trước khi kết hợp lại?

Trong mạng Internet, khi một datagram được phân đoạn, các fragment sẽ di chuyển như những datagram riêng biệt cho đến đích cuối cùng nơi mà chúng phải được kết hợp lại. Giải pháp giữ nguyên cả fragment suốt cả đoạn đường cho đến đích cuối cùng có hai khuyết điểm nhỏ như sau:

- Trước hết, vì các datagram không được tập hợp lại ngay sau khi đi qua một mạng có MTU nhỏ, các fragment nhỏ phải được chuyển tải từ lúc phân đoạn cho tới đích cuối cùng. Việc kết hợp lại datagram tại đích cuối cùng có thể không hiệu quả: kể từ sau nơi phân đoạn, các fragment nhỏ có thể sẽ đi qua những mạng có khả năng MTU lớn.

- Thứ hai, nếu có một fragment bị thất lạc, datagram không thể kết hợp lại được. Máy nhận sẽ bắt đầu khởi động bộ đếm kết hợp khi nó nhận fragment khởi đầu. Nếu thời hạn đến đã hết trước khi tất cả các fragment đến được, máy nhận sẽ huỷ bỏ tất cả các fragment (đã nhận) mà không xử lý gì datagram. Như thế, xác suất mất datagram tăng lên khi việc phân đoạn xảy ra bởi vì việc mất một fragment sẽ làm mất toàn bộ datagram.

Tuy vậy, mặc cho những nhược điểm nhỏ này, thực hiện việc kết hợp lại tại đích cuối cùng vẫn làm việc tốt. Nó cho phép mỗi fragment được chuyển tải một

cách độc lập, và tiết kiệm thời gian và hiệu năng xử lý của các bộ định tuyến trung gian trong phần việc lấy lại và kết hợp các fragment.

### 2.2.4. Điều khiển việc phân đoạn

Ba vùng trong phần đầu datagram, Identification, Flags, và Fragment Offset dùng để điều khiển việc phần đoạn và kết hợp lại của datagram.

- Vùng Identification chứa một số nguyên duy nhất để xác định datagram. Chúng ta nhớ lại rằng khi một bộ định tuyến phân đoạn một datagram, nó sao chép hầu hết các trường dữ liệu của datagram nguyên thủy ban đầu, đặc biệt là trường Identification. Mục đích chính yếu của nó là để máy đích biết các fragment đến thuộc về datagram nào. Khi fragment đến, máy đích sử dụng vùng Identification cùng với địa chỉ nguồn datagram để xác định datagram. Máy tính gửi datagram phải gán một giá trị duy nhất cho vùng Identification cho mỗi datagram (Trong lý thuyết, việc chuyển lại một gói dữ liệu có thể lấy lại cùng vùng Identification như ban đầu; trong thực tế, giao thức cấp cao thực hiện việc chuyển lại nên datagram mới có riêng Identification của nó). Một kỹ thuật được sử dụng là phần mềm điều khiển IP sẽ lưu trữ một bộ đếm trong bộ nhớ, tăng nó lên một mỗi khi có một datagram mới được tạo ra, và gán kết quả cho vùng Identification của datagram.

- Chúng ta nhớ lại rằng mỗi fragment có chính xác cùng định dạng như datagram đầu đủ. Đối với một fragment, vùng Fragment Offset xác định vị trí tương đối trong datagram ban đầu của dữ liệu được chuyển tải trong fragment, được tính theo đơn vị 8 bytes (để giảm kích thước trong phần đầu, nó được xác định là bội số của 8 bytes) bắt đầu từ 0. Để kết hợp lại datagram, máy đích phải có được tất cả các fragment, bắt đầu từ fragment có offset là 0 cho đến fragment không nhất thiết phải đến theo thứ tự, và không có thông tin liên lạc gì giữa bộ định tuyến đã phân đoạn datagram với máy đích sẽ kết hợp chúng lại.

- Hai bit thứ tự thấp trong số 3 bits của vùng FLAGS điều khiển việc phân đoạn. Thông thường, phần mềm ứng dụng sử dụng TCP/IP không quan tâm về việc phân đoạn vì cả hai việc phân đoạn và kết hợp lại là những thủ tục tự động xảy ra tại cấp thấp nhất trong hệ điều hành, mà người sử dụng không thấy được. Tuy nhiên để kiểm tra phần mềm Internet hay bắt lỗi (debug) các vấn đề hoạt động, cũng là điều quan trọng khi kiểm tra kích thước của datagram mà việc phân đoạn xảy ra.

Bit điều khiển đầu tiên hỗ trợ việc kiểm tra đó bằng cách xác định rằng datagram có thể phân đoạn không. Nó được gọi là bit *không phân đoạn* vì thiết lập nó lên 1 để chỉ rằng không nên phân đoạn datagram này. Một ứng dụng có thể

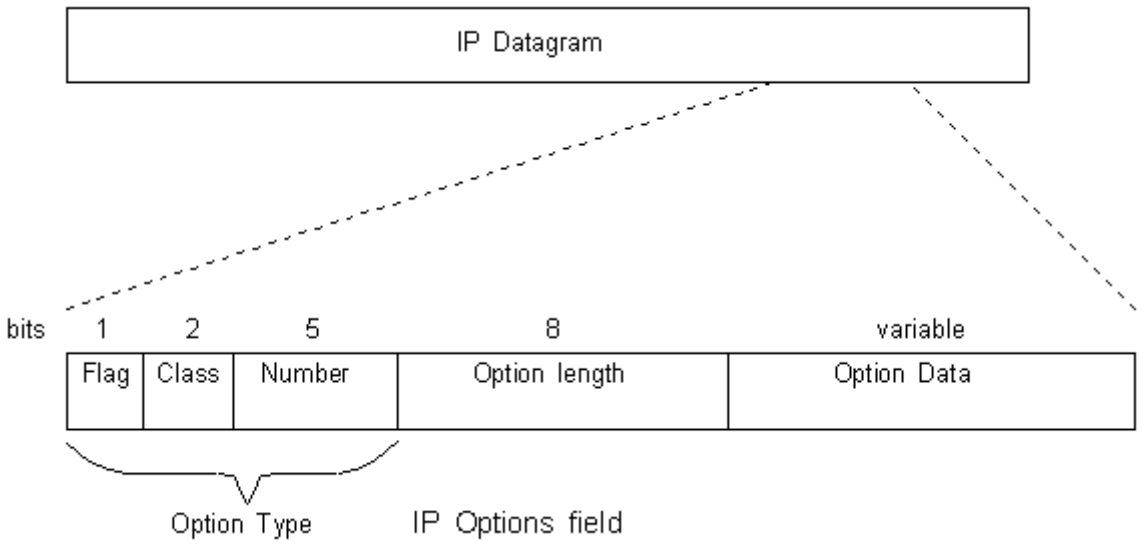
chọn giải pháp cấm việc phân đoạn khi mà việc truyền dữ liệu chỉ có ý nghĩa nếu bảo toàn datagram trong suốt chặng. Ví dụ, xét một quy trình bootstrap trong đó một hệ thống nhúng nhỏ thực hiện một chương trình trong ROM để gửi một yêu cầu qua Internet mà qua đó một máy khác sẽ đáp lời bằng cách gửi lại một hình ảnh bộ nhớ. Nếu hệ thống nhúng đó được thiết kế sao cho nó cần toàn bộ hình ảnh bộ nhớ hoặc không có gì hết, thì không nên thiết lập bit không phân đoạn trong datagram. Bất cứ khi nào một bộ định tuyến cần phân đoạn một datagram có bit không phân đoạn được lập, bộ định tuyến sẽ huỷ bỏ datagram và gửi một thông báo lỗi trở về nơi xuất phát.

Bit thứ tự thấp trong vòng FLAGS xác định rằng fragment chứa dữ liệu từ giữa của datagram gốc hay từ cuối. Nói được gọi là bit vẫn còn phân đoạn. Để hiểu được vì sao chúng ta cần đến bit này, hay xét phần mềm IP tại đích cuối cùng đang cố gắng kết hợp lại một datagram. Nó sẽ nhận các fragment của một datagram. Khi một fragment đến, vùng TOTAL LENGTHT trong phần đầu để chỉ kích thước của fragment đó chứ không phải kích thước của datagram ban đầu, nên máy đích không thể sử dụng vùng TOTAL LENGTHT để biết nó đã nhận đủ các fragment chưa. Bit vẫn còn phân đoạn giải quyết vấn đề này dễ dàng. một khi máy đích nhận một fragment với bit vẫn còn phân đoạn được tắt; nó biết rằng fragment này chuyển tải dữ liệu thuộc phần đuôi của datagram. Từ các vùng FRAGMENT OFFSET và TOTAL LENGTHT, nó có thể tính độ dài của datagram gốc. Bằng cách kiểm tra FRAGMENT OFFSET và TOTAL LENGTH của tất cả các fragment đến, máy nhận có thể biết các fragment nhận được đủ kết hợp lại datagram ban đầu.

### 2.3. Các IP DATAGRAM đặc biệt

Vùng IP OPTIONS ngay sau địa chỉ IP đích trong IP Datagram không bắt buộc có trong mỗi datagram. các chọn lựa đưa thêm vào chủ yếu cho việc kiểm tra và bắt lỗi trên mạng. Tuy nhiên, việc xử lý các chọn lựa lại là một phần không thể thiếu của giao thức IP, nên tất cả các cài đặt chuẩn đều có nó.

Độ dài của vùng IP OPTIONS thay đổi tuỳ theo các chọn lựa được lấy. Một vài chọn lựa có độ dài một byte, chúng bao gồm mã chọn lựa (option code) một byte. Các chọn lựa khác có độ dài khác nhau. Khi các chọn lựa có mặt trong một datagram, chúng nằm kế tiếp nhau, và không có ký hiệu phân cách đặc biệt gì. Mỗi chọn lựa bao gồm một mã chọn lựa một byte, có thể theo sau là một byte độ dài và các byte dữ liệu cho chọn lựa đó. Mã chọn lựa (một byte) được chia thành ba vùng như trong hình 2.6.




*Hình 2.6: Các trường của phần IP Option*

Các vùng của OPION CODE bao gồm 1 bit cờ COPY, 2 bit OPTION CLASS, và 5 bit OPTION NUMBER.

* Cờ COPY kiểm soát cách mà bộ định tuyến làm việc với các chọn lựa trong quá trình phân đoạn. Khi cờ COPY được lập (1), nó xác định rằng chọn lựa phải được sao chép chuyển tất cả các fragment đầu tiên mà thôi.

0 = không được sao chép

1 = được sao chép

* OPTIONS CLASS xác định lớp tổng quát của chọn lựa

0 = điều khiển

1 = dành cho người dùng tiếp

2 = gỡ rỗi và đo lường

3 = dành cho người dùng tiếp

* Các bit OPTION CLASS kết hợp OPTION NUMBER xác định lớp tổng quát của chọn lựa và một chọn lựa đặc biệt trong lớp đó, danh sách trên đây trình bày cách gán các lớp chọn lựa.

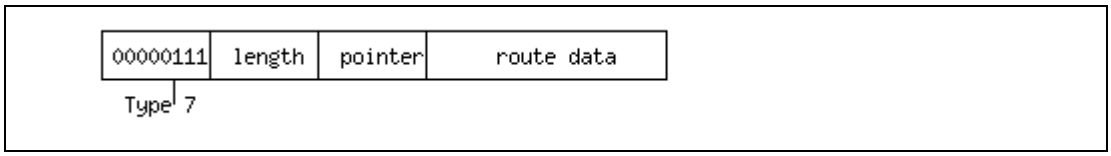
Bảng sau liệt kê các ví dụ của các chọn lựa cho một IP Datagram và trình bày các giá trị OPTION CLASS và OPTION NUMBER của chúng. Qua bảng này ta thấy, hầu hết các chọn lựa được sử dụng cho mục đích điều khiển.

| Mô tả độ dài của Class và Number | | |
|---|---|---|
| **Class** | **Number** | **Mô tả** |
| 0 | 0 | End of Option list. This option occupies only 1 octet; it has no length octet. |
| 0 | 1 | No Operation. This option occupies only 1 octet; it has no length octet |
| 0 | 2 | 11 Security. Used to carry Security, Compartmentation, User Group (TCC), and Handling Restriction Codes compatible with DOD requirements. |
| 0 | 3 | var. Strict Source Routing. Used to route the Internet datagram based on information supplied by the source. |
| 0 | 9 | var. Strict Source Routing. Used to route the Internet datagram based on information supplied by the source. |
| 0 | 7 | var. Record Route. Used to trace the route an Internet datagram takes. |
| 0 | 8 | 4 Stream ID. Used to carry the stream identifier. |
| 2 | 4 | var. Internet Timestamp. |

*Bảng 2.1: Ví dụ lựa chọn cho một IP diagram*

2.3.1. IP Datagram dạng bản ghi định tuyến (Record Route)

Đây là một option khá hữu ích, bởi vì nó cung cấp một cách kiểm tra hoặc điều khiển mà các bộ định tuyến Internet chuyển datagram. Chọn lựa record route cho phép nơi gửi tạo ra một danh sách trắng các địa chỉ IP và sắp xếp để mỗi bộ định tuyến mà xử lý datagram sẽ tự động thêm địa chỉ IP của nó vào danh sách, khi đó ta được một IP Datagram mà nội dung của nó là toàn bộ các địa chỉ IP của các router trung gian mà chính nó đi qua. Hình 2.7 trình bày định dạng của chọn lựa bản ghi định tuyến.




*Hình 2.7: Định dạng bản ghi định tuyến*

Như đã mô tả ở trên, vùng CODE chứa lớp chọn lựa và số chọn lựa (0 và 7) dành cho bản ghi định tuyến.

Vùng LENGTH xác định tổng độ dài của phần dữ liệu chọn lựa khi nó xuất hiện trong IP Datagram, bao gồm cả ba bytes đầu tiên.

ROUTE DATA: các vùng bắt đầu với nhãn FIRST IP ADDRESS, SECOND IP ADDRESS... bao gồm những vùng dành riêng cho việc lưu trữ địa các địa chỉ IP của các Router.

Vùng POINTER xác định vị trí tương đối trong chọn lựa của vùng trống kế tiếp.

Cách hoạt động của Router (máy tính) khi gặp bản ghi này như sau: bất cứ khi nào một máy (host) xử lý một datagram mà có chọn lựa bản ghi định tuyến được lập, máy sẽ thêm địa chỉ của nó vào danh sách bản ghi định tuyến (nơi gửi phải cấp đủ vùng trống trong chọn lựa để có thể lưu trữ được tất cả các địa chỉ mà nó sẽ cần đến). Để thêm bản thân nó vào danh sách, trước hết Router so sánh các vùng POINTER và LENGTH. Nếu POINTER lớn hơn LENGTH, tức là danh sách nối tiếp này đã đầy, Router sẽ chuyển datagram đi mà không chèn địa chỉ của nó vào danh sách này. Còn nếu danh dách chưa đầy, Router sẽ chèn 4 bytes địa chỉ IP của nó vào danh sách ở vị trí xác định bởi POINTER, và tăng POINTER thêm 4 đơn vị.

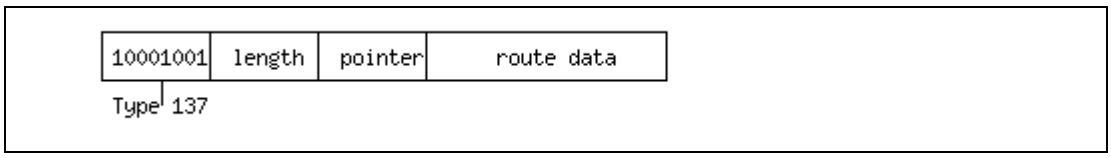
Khi datagram đến, máy đích có thể trích và xử lý lý danh sách các địa chỉ IP này và gửi trả cho trạm nguồn. Thông thường, sử dụng chọn lựa bản ghi định tuyến chỉ khi nào hai máy gửi và nhận cùng đồng ý hợp tác; .

### 2.3.2. IP Datagram dạng bản ghi nguồn định tuyến xác định (Source Route)

Một ý tưởng khác mà những người xây dựng cho rằng đáng quan tâm, là chọn lựa nguồn định tuyến (source route). Ý tưởng nằm trong chọn lựa này là nó cung cấp một cách cho máy gửi để tìm ra đường đi trên Internet. Ví dụ, để kiểm tra tốc độ truyền trên một mạng vật lý cụ thể N, người quản trị hệ thống có thể sử dụng source route để buộc các IP Datagram di chuyển trên mạng N ngay cả nếu bộ định tuyến thường chọn một con đường mà không bao gồm mạng N. Khả năng thực hiện các kiểm tra này là đặc biệt quan trọng trong môi trường đang được khai thác (khác với môi trường đang thử nghiệm), bởi vì người quản trị mang được tự do chuyển datagram của người sử dụng trên các mạng mà họ biết rằng hoạt động chính xác đồng thời có thể kiểm tra những mạng khác. Dĩ nhiên, source route chỉ hữu dụng đối với những người hiểu cấu trúc kết nối mạng;.

IP hỗ trợ hai dạng source route:

* Một dạng, được gọi là strict source route, xác định một con đường định tuyến bằng cách chỉ ra môt chuỗi các địa chỉ IP trong chọn lựa, như trong hình sau.




*Hình 2.8: Bản ghi nguồn định tuyến xác đinh*

Sở dĩ có tên là "strict" là vì các địa chỉ xác định chính xác con đường mà datagram phải đi qua để đến đích cuối cùng. Con đường giữa hai địa chỉ kế tiếp nhau trong danh sách phải bao gồm một mạng vật lý; nếu bộ định tuyến không thể tuân theo strict source route nó sẽ gửi thông báo lỗi.

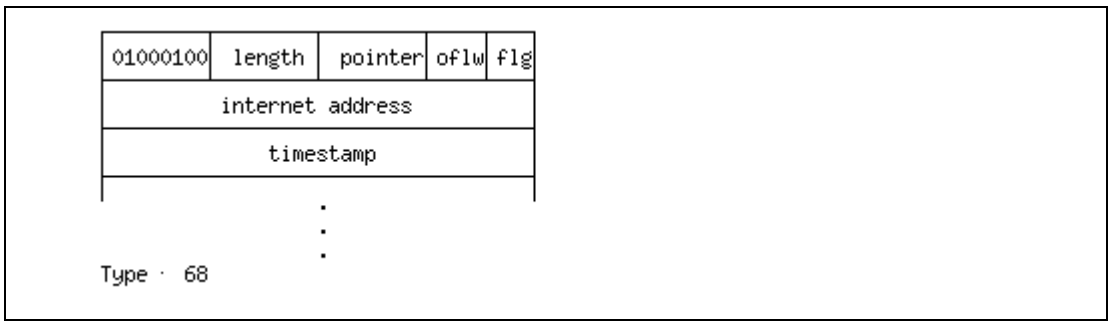
* Một dạng khác, được gọi là loose source route, cũng bao gồm một chuỗi các địa chỉ IP. Nó xác định rằng datagram phải đi theo chuỗi các địa chỉ IP này, nhưng cho phép có nhiều con đường mạng giữa hai địa chỉ kế tiếp nhau trong danh sách.

Cả hai chọn lựa source route đều đòi hỏi các bộ định tuyến dọc theo con đường phải ghi chồng lên các mục trong danh sách địa chỉ bằng các địa chỉ mạng cục bộ của nó. Như thế, khi datagram đến đích của nó, nó xác như là danh sách liên tiếp được tạo ra bởi chọn lựa source route.

Định dạng của chọn lựa source route cũng tương tự như của chọn lựa bản ghi định tuyến ở trên. Mỗi bộ định tuyến kiểm tra các vùng POINTER và LENGTH để xem danh sách đã bị đầy chưa, vùng ROUTE DATA: là các vùng bắt đầu với nhãn FIRST IP ADDRESS, SECOND IP ADDRESS... bao gồm những vùng dành riêng cho việc ghi nhận các địa chỉ Internet. Nếu POITNER lớn hơn LENGTH, danh sách đã đầy, thì bộ định tuyến sẽ chuyển datagram đến đích của nó như thường lệ. Nếu danh sách chưa đầy, bộ định tuyến đi theo POINTER, lấy ra địa chỉ IP, thay thế nó bởi địa chỉ IP của bộ định tuyến (một định tuyến có một địa chỉ IP cho mỗi bộ giao tiếp; nó ghi nhận địa chỉ mà tương ứng với mạng mà nó sẽ chuyển datagram), và chuyển datagram đi thông qua địa chỉ lấy ra từ danh sách địa chỉ.

### 2.3.3. IP Datagram dạng bản ghi ghi nhận thời điểm (Timestamp)

Chọn lựa timestamp làm việc giống như chọn lựa bản ghi định tuyến. Khởi đầu nó bao gồm một danh sách trống, và mỗi bộ định tuyến dọc theo con đường từ nguồn đến đích sẽ điền một mục vào danh sách. Mỗi mục trong danh sách bao gồm hai phần 32 bit: phần địa chỉ IP của bộ định tuyến và vùng ghi giá trị tem thời gian mà gói tin đến bộ định tuyến này timestamp.




*Hình 2.9: Bản ghi ghi nhận thời điểm*

Hình này, các vùng LENGTH và POINTER được sử dụng để xác định độ dài của vùng trống được dành riêng cho chọn lựa và vị trí của mục kế tiếp chưa sử dụng (chính xác như chọn lựa bản ghi định tuyến). Vùng 4 bit OFLOW chứa một số nguyên để đếm số bộ định tuyến mà không thể cung cấp timestamp bởi vì chọn lựa quá nhỏ.

Giá trị trong vùng 4 bit FLAGS điều khiển định dạng chính xác của chọn lựa và cho biết mà độ định tuyến phải cung cấp timestamp. Các giá trị đó là:

0 – timestamp chỉ được lưu trữ trong 32bit words liên tục.

1 – mỗi timestamp được đứng trước với địa chỉ Internet của thực thể đang đăng ký.

3 – địa chỉ các trường Internet được xác định trước. Một module IP chỉ đăng ký dấu thời gian (timestamp) của nó nếu nó tính toán địa chỉ của nó với địa chỉ Internet được quy định tiếp theo.

Timestamp ghi nhận ngày và thời gian mà bộ định tuyến xử lý datagram, được tính đến đơn vị millisecond với mốc giờ kể từ 0h lúc nửa đêm, theo giờ GMT. Nếu thể hiện chuẩn của thời gian không có, bộ định tuyến có thể sử dụng thể hiện bất kỳ của thời gian địa phương miễn sao nó bật lên bit thứ tự cao trong vùng timestamp. Dĩ nhiên, timestamp được ghi nhận từ những router độc lập và thường không luôn luôn thống nhất với nhau về định dạng trình bày theo giờ GMT; mỗi Router thể hiện thời gian theo đồng hồ riêng của nó, và các đồng hồ có thể khác nhau. Như thế, các mục timestamp luôn luôn được xem là "ước lượng", độc lập với cách thể hiện.

Việc ghi nhận địa chỉ IP cùng với timestamp trừ loại trừ sự nhầm lẫn cho người kiểm tra. nó hữu ích bởi vì nó cho phép nơi nhận biết chính xác con đường và tốc độ mà datagram đã di qua.

### 2.3.4. Xử lý các option trong quá trình phân đoạn

Bây giờ, chúng ta sẽ phân tích tác dụng của bit COPY trong vùng CODE. Khi phân đoạn một datagram, bộ định tuyến sao chép một số chọn lựa IP vào tất cả các fragment trong khi lại sao chép những chỗ khác vào chỉ một fragment. Ví dụ, xét chọn lựa dùng để để ghi lại danh sách địa chỉ IP các Router trung gian trên đường đi của datagram. Chúng ta đã biết rằng mỗi fragment sẽ được xử lý như một datagram độc lập, vì thế không có gì để bảo đảm rằng tất cả các fragment sẽ đi theo cùng một con đường đến đích. Nếu những fragment của một gói tin kiểu record route đi theo nhiều con đường khác nhau, đích đến có thể nhận một danh sách khác nhau của các bộ định tuyến từ mỗi fragment. Nó không thể tạo ra duy nhất một danh sách của các bộ định tuyến cho datagram (đã kết hợp lại). Hơn nữa, chuẩn IP mô tả rằng chọn lựa record route chỉ được sao chép vào một fragment.

Không phải tất cả chọn lựa IP có thể được giới hạn vào một fragment. Ví dụ, xét chọn lựa record route, xác định cách mà một datagram phải di chuyển qua Internet. Thông tin về record route sẽ không tuân theo con đường đã xác định. Vì thế, vùng mã của record route xác định rằng chọn lựa phải được sao chép vào tất cả các fragment.

**Câu hỏi và bài tập**

2.1. Khái niệm chuyển phát phi kết nối?

2.1. Vai trò của giao thức IP?

2.3. Cấu trúc gói dữ liệu IP Datagram?

2.4. Nguyên lý phân mảnh và hợp nhất IP Datagram?

2.5. Khái niệm MTU, cho biết MTU của các mạng phổ biến?

2.6. Tìm hiểu lệnh IP CONFIG?

2.7. Các IP Datagram đặc biệt?

2.8. Thiết kế nguyên tắc hoạt động của một bộ lọc gói IP Datagram ứng dụng vào Firewall lọc gói ở tầng IP?

2.9. Bài tập: phân mảnh và hợp nhất một IP Datagram có kích thược vượt quá MTU của mạng mà nó đi qua?

2.10. Cách đóng gói IP Datagram?

2.11. Bài tập: Dùng các phầm mềm lọc và bắt gói tin phổ biến như Ethereal, Wireshack…để lọc và phân tích các các tham số của gói tin IP gửi từ máy sinh viên ra mạng Internet.

# CHƯƠNG 3 ÁNH XẠ CÁC ĐỊA CHỈ IP LÊN ĐỊA CHỈ VẬT LÝ (ARP & RARP)

## 3.1. Giao thức phân giải địa chỉ (ADDRESS RESOLUTION PROTOCOL)

### 3.1.1. Khái niệm ánh xạ địa chỉ

Trong mô hình của bộ giao thức TCP/IP, việc gán địa chỉ cho mỗi máy tính (host) một địa chỉ logic có độ dài 32 bit, và mô hình hóa liêm mạng Internet như một mạng ảo. Trong việc truyền dữ liệu ở mạng vật lý, hai máy trên cùng một mạng vật lý chỉ có thể liên lạc nếu biết được địa chỉ vật lý của nhau. Nhưng làm thế nào một máy tính hay một bộ định tuyến có thể ánh xạ một địa chỉ IP thành chính xác địa chỉ vật lý khi gửi các gói dữ liệu qua một mạng (vật lý), chương này sẽ đi sâu vào vấn đề này.

Xét hai máy A và B được nối vào cùng một mạng vật lý. Mỗi máy được gán một địa chỉ IP là IA và IB và địa chỉ vật lý PA và PB. Mục đích là để tìm ra phần mềm cấp thấp che dấu địa chỉ phần cứng và cho phép những chương trình mức cao hơn chỉ phải làm việc với địa chỉ Internet. Tuy nhiên, cuối cùng thì việc liên lạc phải được thực hiện bởi các mạng vật lý với bất kỳ mô hình địa chỉ vật lý nào do phần cứng mạng cung cấp. Giả sử máy A muốn gửi dữ liệu cho máy này nối vào, nhưng A chỉ có địa chỉ Internet IB của B. Câu hỏi đặt ra là: làm sao A ánh xạ địa chỉ đó vào địa chỉ vật lý PB của B?

Việc ánh xạ địa chỉ phải được thực hiện tại mỗi bước dọc theo con đường từ nơi xuất phát đầu tiên tới đích cuối cùng. Cụ thể, có hai trường hợp:

+ Tại bước cuối cùng của việc chuyển phát dữ liệu, gói dữ liệu phải được gửi qua một mạng vật lý để tới đích cuối cùng của nó. Máy tính gửi dữ liệu (trung gian), phải ánh xạ địa chỉ IP của đích cuối cùng thành địa chỉ vật lý của nó.

+ Tại điểm bất kỳ dọc theo con đường từ nguồn đến đích tuyến trung gian. Như thế, nơi gửi phải ánh xạ địa chỉ IP của bộ định tuyến trung gian thành địa chỉ vật lý (của bộ định tuyến).

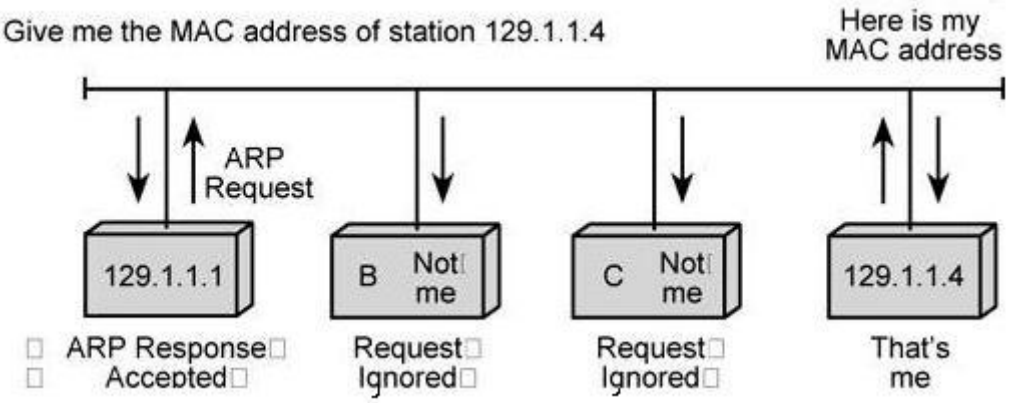
### 3.1.2. Nguyên lý hoạt động của giao thức ARP

#### *3.1.2.1. Khái niệm địa chỉ vật lý (MAC Address)*

Mỗi bộ giao tiếp Ethernet được gán một địa chỉ vật lý 48 bit khi sản xuất. Hệ quả là, khi phần cứng bị hỏng và phải thay một bộ giao tiếp Ethenet khác, thì địa chỉ vật lý của máy thay đổi. Hơn nữa, vì địa chỉ Ethenet dài 48 bit, không thể có

phương pháp để mã hoá nó thành địa chỉ IP 32 bit (bởi vì ánh xạ trực tiếp là tiện lợi và hiệu quả hơn liên kết động nên thế hệ kế tiếp của IP được thiết kế để mà địa chỉ phần cứng 48 bit có thể được mã hoá thành địa chỉ IP).

Những người thiết kế giao thức TCP/IP đã tìm ra một lời giải tốt và sáng tạo cho bài toán giải địa chỉ cho những mạng như Ethenet mà có cả khả năng quảng bá. Giải pháp này cho phép ta thêm vào mạng những máy mới và bộ định tuyến mới mà không phải thay đổi địa chỉ của nguyên cả mạng, và cũng không đòi hỏi việc bảo trì cơ sở dữ liệu trung tâm. Để tránh việc bảo trì bảng ánh xạ, những nhà thiết kế đã chọn giải pháp sử dụng một giao thức cấp thấp để kết hợp các địa chỉ. Được gọi là giao thức phân giải địa chỉ (Address Resoluion Protocol – ARP), giao thức này có một cơ chế vừa dễ bảo trì vừa có độ hiệu quả trong tính toán, xử lý.




*Hình 3.1: Gửi quảng bá trên mạng để yêu cầu tìm địa chỉ MAC*

*3.1.2.2. Nguyên tắc tìm địa chỉ MAC khi biết địa chỉ IP*

Như trình bày trong hình 3.1, ý tưởng về phần giải địa chỉ và ánh xạ giữa địa chỉ vật lý (MAC Address) và địa chỉ logic (IP Address) của ARP đơn giản như sau:

+ Khi máy A (129.1.1.1) muốn tìm địa chỉ vật lý của máy tính có địa chỉ IP (129.1.1.4) là IB, nó phát đi đến mỗi máy khác một gói dữ liệu đặc biệt để hỏi xem máy nào có địa chỉ IP là IB, thì trả lời bằng địa chỉ vật lý của máy B (PB).

+ Tất cả các máy, bao gồm cả B (129.1.1.4) đều nhận được yêu cầu này, nhưng chỉ có máy B nhận ra địa chỉ của nó và gửi lại lời đáp có bao gồm địa chỉ vật lý của nó.

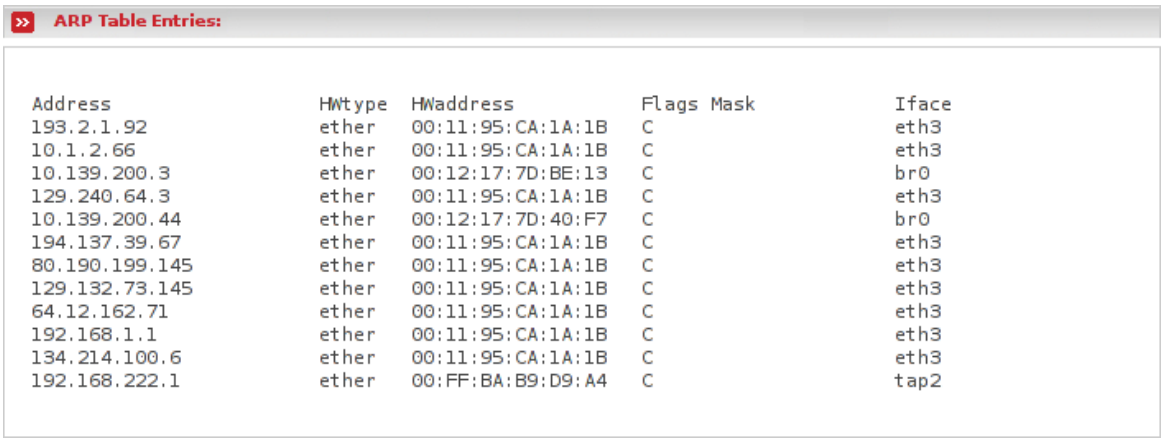
+ Khi A nhận được lời đáp, nó lưu trữ thông tin về địa chỉ vật lý của B vào bảng lưu trữ của nó, đồng thời A sử dụng PB để gửi dữ liệu trực tiếp tới B.

+ Song song quá trình đó là những máy nào chưa có bộ thông tin về (địa chỉ vật lý của A, địa chỉ IP của A) sẽ lưu lại thông tin này vào bảng lưu trữ địa chỉ của nó.

*3.1.2.3. Bảng lưu trữ ARP Table*

Việc quảng bá dữ liệu rất tốn kém tài nguyên, không thể luôn luôn dùng đến khi một máy cần gửi dữ liệu đi, vì mọi máy trên mạng đều phải nhận và xử lý gửi dữ liệu quảng bá.

Để giảm chi phí truyền tin, các máy tính sử dụng ARP có duy trì một kho chứa (cache) những yêu cầu kết hợp địa chỉ IP thành địa chỉ vật lý, mới nhất. Có nghĩa là, bất cứ khi nào một máy tính gửi một yêu cầu ARP và nhận một lời đáp ARP, nó cất thông tin về địa chỉ IP và địa chỉ phần cứng tương ứng vào kho chứa, để lần sau lấy ra dùng lại. Khi truyền một gói dữ liệu, máy tính luôn luôn tìm thông tin địa chỉ trong kho của nó trước khi gửi một yêu cầu ARP. Nếu nó tìm thấy địa chỉ kết hợp như mong muốn trong kho chứa ARP của nó, máy tính sẽ không cần quảng bá trên mạng. Như thế, khi hai máy tính trên một mạng muốn liên lạc với nhau, chúng sẽ bắt đầu với một lời yêu cầu ARP từ máy gửi và lời đáp ARP từ máy nhận, và sau đó chỉ việc gửi dữ liệu mà không cần đến ARP nữa, vì hầu hết các trao đổi thông tin trên mạng đều có việc gửi dữ liệu, nên chỉ cần một bộ nhớ chứa nhỏ cũng đủ.


ARP Table Entries:

| Address | HWtype | HWaddress | Flags Mask | Iface |
|---|---|---|---|---|
| 193.2.1.92 | ether | 00:11:95:CA:1A:1B | C | eth3 |
| 10.1.2.66 | ether | 00:11:95:CA:1A:1B | C | eth3 |
| 10.139.200.3 | ether | 00:12:17:7D:BE:13 | C | br0 |
| 129.240.64.3 | ether | 00:11:95:CA:1A:1B | C | eth3 |
| 10.139.200.44 | ether | 00:12:17:7D:40:F7 | C | br0 |
| 194.137.39.67 | ether | 00:11:95:CA:1A:1B | C | eth3 |
| 80.190.199.145 | ether | 00:11:95:CA:1A:1B | C | eth3 |
| 129.132.73.145 | ether | 00:11:95:CA:1A:1B | C | eth3 |
| 64.12.162.71 | ether | 00:11:95:CA:1A:1B | C | eth3 |
| 192.168.1.1 | ether | 00:11:95:CA:1A:1B | C | eth3 |
| 134.214.100.6 | ether | 00:11:95:CA:1A:1B | C | eth3 |
| 192.168.222.1 | ether | 00:FF:BA:B9:D9:A4 | C | tap2 |

*Hình 3.2: Ví dụ một bảng ARP Cache Table*

Kho chứa ARP cho ta một ví dụ về trạng thái mềm (soft state), một kỹ thuật thường được sử dụng trong các giao thức mạng. Tên của nó mô tả cho một trạng thái mà kho đó thông tin có thể trở nên "lạc hậu" mà lại không thông báo trước. Trong trường hợp của ARP, ta hãy xét hai máy tính A và B được nối vào mạng Ethernet. Giả sử máy A gửi một yêu cầu ARP, và máy B đáp lời. Giả sử rằng sau đó máy B bị hỏng. Máy A sẽ không nhận được thông tin gì về việc máy B đã bị hỏng. Hơn nữa, vì nó sẽ không nhận được thông tin về địa chỉ của máy B trong

kho chứa ARP, máy A sẽ tiếp tục gửi dữ liệu cho B. Phần cứng Ethernet cũng không cung cấp bất kỳ dấu hiệu gì về việc máy B không còn trên mạng vì kỹ thuật Ethernet không có sự chuyển phát được bảo đảm. Vì thế, A không có cách nào biết được có sự chắc chắn máy nhận sẽ nhận được dữ liệu từ nó gửi đi. Vì thế, A không có cách nào biết được khi nào thì thông tin trong kho chứa ARP của nó hết giá trị.

Để khắc phục tình trạng trên, trách nhiệm cho tính chính xác thuộc về người quản lý thông tin. Thông thường, các giao thức mà cài đặt trạng thái mềm sử dụng các bộ đếm thời gian, và thông tin trạng thái bị xoá bỏ sau khi hết thời hạn. Ví dụ, bất cứ khi nào thông tin về địa chỉ liên kết được đặt vào kho chứa ARP, giao thức sẽ yêu cầu bộ đếm thời gian bắt đầu đếm, thông thường là 20 phút. Khi hết hạn (sau 20 phút), thông tin phải được xoá bỏ. Sẽ có hai khả năng xảy ra sau khi đã xoá bỏ, nếu không còn dữ liệu phải được gửi tới máy đích này, không có gì xảy ra nữa. Nếu không còn dữ liệu phải được gửi tới máy đích này và không còn thông tin về địa chỉ này trong kho chứa, máy đích sẽ phải lặp lại thủ tục thông thường là quảng bá một yêu cầu ARP và lấy lại thông tin địa chỉ. Nếu máy đích vẫn còn đó, thông tin địa chỉ lại được đặt vào kho chứa ARP. Nếu không, nơi gửi sẽ phát hiện được rằng máy đích không còn nối mạng nữa.

Hiển nhiên việc sử dụng kỹ thuật trạng thái mềm trong ARP vừa có ưu điểm vừa có nhược điểm.

Ưu điểm lớn nhất là tính tự động, trước hết, một máy tính có thể xác định khi nào thông tin trong kho chứa của nó phải được làm mới (refresh) trở lại độc lập với các máy nhận hay nơi thứ ba để xác định rằng thông tin về liên kết địa chỉ không còn giá trị sử dụng nữa: nếu máy đích không đáp lại một yêu cầu ARP, máy gửi sẽ xem như máy đích không còn nữa. Thứ ba, mô hình này không phụ thuộc vào phần cứng mạng để xác định sự tin cậy của việ truyền dữ liệu

Nhược điểm lớn nhất của trạng thái mềm là tiến trình bị chậm, nếu thời gian đếm là N giây, máy gửi sẽ chỉ nhận biết được rằng máy nhận đã bị hỏng sau N giây.

*3.1.2.4. Các tính năng nâng cao của ARP*

Khi A sắp sử dụng ARP để tìm địa chỉ vật lý của B vì nó cần gửi dữ liệu đến B, thì sẽ có xác xuất khá cao là máy B sẽ gửi dữ liệu cho A trong tương lai gần. Để lợi dụng điều này và tránh bớt giao dịch trên mạng, A khi gửi cho B một yêu cầu tìm địa chỉ IP của B, nó sẽ gửi kèm cả cặp thông tin về địa chỉ vật lý địa chỉ IP của A. B sẽ trích thông tin liên kết của A ra và lưu vào kho chứa ARP của nó, và sau đó gửi lời đáp đến A.

Khi A gửi quảng bá yêu cầu tìm địa chỉ vật lý của B tới mọi máy trên mạng (yêu cầu này mang cặp thông tin PA IA), máy tính nào chưa có cặp thông tin (PA, IA) thì nó sẽ tranh thủ lưu trữ ngay thông tin này vào ARP Cache Table của nó.

Khi A khởi động lại trong mạng, nó luôn thông báo cho các máy tính khác thông tin về (PA, IA) quảng bá cho mọi máy tính khác biết về thông tin của nó, điều này giúp xử lý việc cập nhật thông tin thay đổi khi có sự cố về phần cứng máy tính (thay thế phần cứng mới, thay thế card mạng).

Mối liên hệ của ARP với những giao thức khác

ARP cung cấp một cơ chế khả dĩ để ánh xạ từ địa chỉ IP thành địa chỉ vật lý; chúng ta cũng thấy từng một số kỹ thuật mạng không cần đến nó. Vấn đề là, ARP sẽ là hoàn toàn không cần thiết nếu chúng ta có thể làm cho mỗi phần cứng mạng nhận biết được địa chỉ IP. Như thế ARP chỉ đơn giản là áp đặt một mô hình địa chỉ mới lên trên bất kỳ mô hình địa chỉ cấp thấp nào mà phần cứng sử dụng. ý tưởng này có thể được tóm tắt như sau:

ARP là một giao thức cấp – thấp che dấu địa chỉ vật lý của mạng cơ sở, cho phép ta gán địa chỉ IP bất kỳ cho mỗi máy. Chúng ta xem ARP như một phần của hệ thống mạng vật lý, chứ không phải là một phần của các giao thức là Internet.

*3.1.2.5. Cài đặt ARP*

Về tính năng, phần mềm ARP được chia thành hai phần xử lý như sau.

**Đầu tiên**: Ánh xạ một địa chỉ IP ra một địa chỉ vật lý khi gửi gói dữ liệu, và phần hai đáp lại những yêu cầu từ máy khác. Việc giải địa chỉ cho dữ liệu được gửi được dường như là chuyện không khó khăn, tuy nhiên, có những chi tiết nhỏ làm phức tạp việc cài đặt. Khi được cung cấp thông tin về một địa chỉ IP của máy đích, phần mềm sẽ kiểm tra trong kho có chứa ARP của nó để xem có một ánh xạ từ địa chỉ IP ra địa chỉ vật lý. Nếu có, phần mềm sẽ trích ra phần địa chỉ vật lý, đưa dữ liệu và địa chỉ này vào một Frame, và gửi Frame đó đi. Nếu nó không biết có anh xạ này, phần mềm phải quảng bá một yêu cầu ARP một lời đáp lại.

Việc quảng bá một yêu cầu ARP để tìm ánh xạ của một địa chỉ có thể là công việc phức tạp. Máy đích có thể đã bị tắt hoặc quá bận nên không đáp lại lời yêu cầu được. Nếu vậy, máy gửi sẽ không nhận được lời đáp hoặc nhận được nhưng chậm hơn nhiều. Bởi vì tính chất của Ethernet, lời yêu cầu của quảng bá ARP ban đầu có thể bị mất đi (trong trường hợp đó, máy gửi phải đưa ra yêu cầu, ít nhất một lần nữa). Trong lúc đó, máy tính này phải lưu trữ gói dữ liệu sắp gửi để mà gửi nó đi một khi địa chỉ được giải xong. Nếu quá trình này làm chậm một khoảng thời gian đáng kể, thì máy sẽ quyết định huỷ bỏ các gói dữ liệu sắp được

gửi. Thực ra, máy tính phải quyết định có cho phép các chương trình khác được thực hiện trong khi nó xử lý một yêu cầu ARP. Nếu vậy, phần mềm phải xử lý trường hợp khi một chương trình đưa ra thêm một yêu cầu ARP cho cùng một địa chỉ, không cần quảng bá thêm cho địa chỉ đích đó.

Cuối cùng xét trường hợp khi máy A đã lấy được thông tin địa chỉ của máy B, nhưng khi phần cứng của B bị hỏng và đã được thay thế. Mặc dù địa chỉ của B đã bị thay đổi nhưng thông tin trong kho chứa của A vẫn chưa cập nhật, vì vậy A sử dụng địa chỉ phần cứng không đúng cho B do đó không thể gửi dữ liệu đến đích được. Trường hợp này cho thấy tầm quan trọng của việc phần mềm ARP xem bảng thông tin địa chỉ như một kho chứa và loại bỏ thông tin trong bảng sau một thời gian định kỳ nhất định. Dĩ nhiên, bộ đếm thời gian cho một mục (bản ghi) trong kho chứa phải được khởi động lại bấ t cứ khi nào máy nhận được một quảng bá ARP mà có chứa thông tin địa chỉ của mục đó.

**Thứ hai:** ARP xử lí các gói dữ liệu ARP đến từ mạng. Khi một gói dữ liệu ARP đến, trước tiên phần mềm sẽ trích ra một cặp địa chỉ phần cứng và địa chỉ IP của máy gửi, rồi kiểm tra kho chứa của mình xem đã có mục tương ứng với máy này không. Nếu có một mục (bản ghi) tương ứng với địa chỉ IP này, phần mềm sẽ cập nhật thông tin của mục này bằng cách ghi địa chỉ phần cứng mới (vừa được tách ra từ gói ARP) chồng lên địa chỉ phần cứng hiện tại. Sau đó, máy này xử lý phần còn lại của gói dữ liệu ARP.

Máy nhận sẽ phải xử lý hai loại dữ liệu ARP. Khi một yêu cầu ARP đến, máy nhận phải kiểm tra xem nó có phải là đích đến của yêu cầu này (nghĩa là một máy nào đó đã quảng bá một yêu cầu về địa chỉ vật lý của máy nhận). Nếu đúng, phần mềm ARP sẽ đáp lại bằng việc cung cấp địa chỉ phần cứng của nó, và gửi lời đáp trực tiếp về nơi yêu cầu. Máy nhận cũng thêm địa chỉ của máy chưa tồn tại. Nếu địa chỉ IP trong yêu cầu ARP không trùng với địa chỉ IP của máy nhận, nghĩa là nó đang tìm một ánh xạ của máy nào khác đó trên mạng, nên có thể bỏ qua.

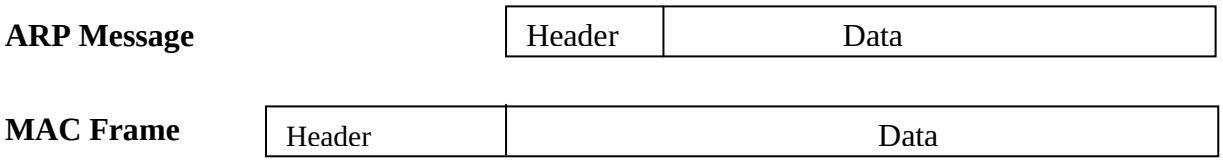
Một trường hợp khác là khi một lời đáp ARP đến. Tuỳ vào việc cài đặt, phần mềm xử lý có thể cần tạo ra một mục trong kho chứa, hoặc mục này có thể được tạo khi phát sinh yêu cầu ARP. Cho dù là trường hợp nào, một khi kho chứa đã được cập nhập, máy nhận sẽ thử so sánh sự giống nhau giữa lời đáp với lời yêu cần đã đưa ra trước đó. Thông thường, lời đáp đến đáp ứng lời yêu cầu, yêu cầu này đã được đưa ra vì máy tính cần gửi gói dữ liệu. Giữa thời gian một máy quảng bá yêu cầu ARP của nó và nhận được lời đáp, các chương trình ứng dụng và những giao thức cấp cao có thể đưa ra thêm những yêu cầu (với cùng một địa chỉ); phần mềm phải nhớ rằng nó đã gửi đi một yêu cầu và không gửi nữa. Thông thường, phần mềm ARP nhúng gói dữ liệu kế tiếp trong một hàng đợi. Một khi nhận được lời

đáp và biết thông tin địa chỉ, phần mềm ARP lần lượt lấy các gói dữ liệu ra khỏi hàng đợi, đặt mỗi gói vào một frame, và sử dụng thông tin địa chỉ có được để cho vào vùng địa chỉ đích (vật lý). Nếu trước đây nó không có lời yêu cầu về địa chỉ IP chứa trong lời đáp, máy tính sẽ cập nhật mục này trong kho chứa của nó. rồi chỉ đơn giản ngưng xử lý gói dữ liệu.

*3.1.2.6. Đóng gói và định dạng thông điệp ARP*

*a. Đóng gói thông điệp ARP*

Khi một thông điệp ARP di chuyển từ máy này tới máy kia, nó được chuyển tải trong các frame, (vật lý). Hình 3.3 trình bày thông điệp ARP được chuyển tải trong phần dữ liệu của frame.




*Hình 3.3: Đóng gói thông điệp ARP*

Để xác định frame khi nó chuyển tải thông điệp ARP, nơi gửi sẽ gán một giá trị đặc biệt cho vùng kiểu trong phần đầu (header) của frame, và đặt thông điệp ARP vào vùng dữ liệu của frame. Khi một frame đến được một máy tính, phần mềm mạng sử dụng kiểu frame để xác định nội dung của nó. Trong hầu hết các kỹ thuật, chỉ một kiểu giá trị được sử dụng cho tất cả frame chuyển tải thông điệp ARP – phần mềm mạng của máy nhận cũng phải kiểm tra các thông điệp ARP để phân biệt giữa yêu cầu ARP và lời đáp ARP. Ví dụ, trong mạng Ethernet, các frame chuyển tải thông điệp ARP có vùng kiểu là $0806_{16}$. Đây là một giá trị chuẩn được gán bởi chuẩn Ethernet; và dĩ nhiên những kỹ thuật phần cứng mạng khác sẽ sử dụng những giá trị khác.

*b. Định dạng thông điệp ARP*

| 0 | 8 | 16 | 31 |
|---|---|---|---|
| Hardware Type | | Protocol Type | |
| Hardware Length | Protocol Length | Operation | |
| Sender Hardware Address (0 - 3) | | | |
| Sender Hardware Address (4-5) | | Sender IP Address (0 -1) | |
| Sender IP Address (2 - 3) | | Target Hardware Address (0 -1) | |
| Target Hardware Address (2 -5) | | | |
| Target IP Address | | | |

*Hình 3.4: Định dạng thông điệp ARP*

Không giống như hầu hết các giao thức, dữ liệu trong các gói ARP không có phần đầu (header) được định dạng cố định. Thay vì thế, để cho ARP có ích đối với các kỹ thuật mạng khác nhau, độ dài của các vùng địa chỉ tuỳ thuộc vào kiểu mạng. Tuy nhiên, để có thể hiểu được một thông điệp ARP bất kỳ, trong phần đầu có những vùng cố định nằm gần nơi xác định độ dài của địa chỉ trong các vùng tiếp theo sau. Thực ra dạng của thông điệp ARP rất là tổng quát sao cho nó có thể được sử dụng với những địa chỉ vật lý bất kỳ và các giao thức địa chỉ bất kỳ.

Ví dụ trong hình 3.4 là một dạng thông điệp ARP 28 bytes được sử dụng trong phần cứng Ethernet (trong đó địa chỉ vật lý dài 48 bit hay 6 bytes), khi giải các địa chỉ IP (chỉ dài 4 bytes).

Hinh 3.4 trình bày một thông điệp ARP trên các mạng Ethernet, tổ chức thành 4 bytes trên mỗi hàng, và đây là dạng chuẩn được dùng trong toàn bộ giáo trình này. Không như hầu hết các giao thức khác, các vùng có độ dài thay đổi trong các gói ARP không phải lúc nào cũng là 32 bit, làm cho công việc thêm khó khăn. Ví dụ, địa chỉ phần cứng của máy gửi, có nhãn là SENDER Ha, chiếm 6 bytes liên tục, vì thế nó được trình bày trên hai dòng.

Các vùng (trường) có ý nghĩa như sau:

+ HARDWARE TYPE xác định kiểu của bộ giao tiếp phần cứng mà máy gửi đang cần biết; với giá trị 1 dành cho Ethernet.

+ PROTOCOL TYPE xác định kiểu của giao thức địa chỉ cấp–cao mà máy gửi cung cấp, nó có giá trị $0800_{16}$ dành chi địa chỉ IP.

+ OPERATION xác định kiểu của thông điệp, là một trong các loại sau:

Thông điệp này là một yêu cầu ARP

Thông điệp này là một lời đáp ARP

Thông điệp này là một yêu cầu RARP (là một giao thức khác sử dụng cùng dạng thông điệp, sẽ được trình bày tiếp sau)

Thông điệp này là một lời đáp RARP (4).

+ HLEN và PLEN cho phép được sử dụng với các mạng bất kỳ vì chúng xác định độ dài của địa chỉ phần cứng (vật lý) và độ dài của địa chỉ logic (IP) của nó,

+ SENDER HA: Địa chỉ vật lý của trạm gửi thông điệp ARP (MAC Address)

+ SENDER IP: Địa chỉ logic (IP Address) của trạm gửi.

+ TARGET HA: Địa chỉ vật lý của trạm nhận thông điệp ARP

+ TARGET: Địa chỉ logic (IP Address) của trạm nhận.

Khi thực hiện một yêu cầu, nơi gửi cũng cung cấp địa chỉ phần cứng của máy đích (RARP) hay địa chỉ IP của máy đích (ARP), thông qua các vùng TARGET HA hay TARGET IP. Trước khi máy đích đáp lời, nó sẽ điền vào các địa chỉ còn thiếu, hoán đổi cặp gửi và nhận, và thay đổi thao tác trở thành lời đáp. Như thế, một lời đáp sẽ chuyển tải địa chỉ IP và địa chỉ phần cứng của nơi yêu cầu ban đầu, cũng như địa chỉ IP và địa chỉ phần cứng máy đáp lời.

### 3.2. Giao thức giải địa chỉ ngược (RARP: REVERSE ADDRESS RESOLUTION PROTOCOL)

Thông thường, địa chỉ IP của một máy được lưu trên bộ trữ (đĩa cứng), mà hệ điều hành có thể lấy được khi khởi động. Câu hỏi được đặt ra là, "nếu một máy không có gán đĩa cứng, thì làm thế nào nó có thể xác định được địa chỉ IP của nó?"

Nếu phần mềm hệ điều hành được gán (lập trình) một địa chỉ IP nhất định, thì không thể sử dụng chúng cho nhiều máy tính, vì thế người thiết kế thường tránh việc này. Cụ thể là phần mềm bootstrap, hay được ghi vào ROM, thường được lập trình sao cho chúng có thể chạy được trên nhiều máy. Khi các chương trình đó bắt đầu chạy, nó sử dụng mạng để liên lạc với máy chủ và nhận được địa chỉ IP từ máy chủ cấp cho nó.

Thoạt tiên, thủ tục bootstrap có vẻ ngược đời: một máy tính liên lạc với máy chủ từ xa để lấy một địa chỉ cần cho việc thông tin liên lạc. Tuy nhiên không phải thế, vì máy tính biết cách để liên lạc. Nó có thể sử dụng địa chỉ phần cứng của nó để liên trên một mạng đơn. Vì vậy, máy tính phải tạm thời viện đến địa chỉ mạng

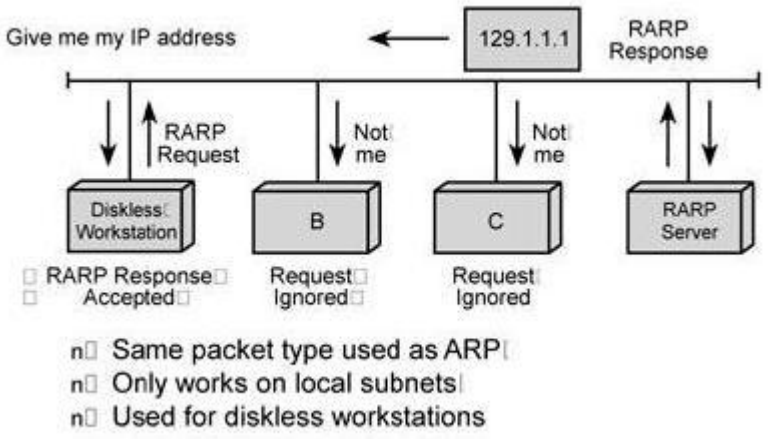
vật lý để liên hệ với máy chủ để nhận được địa chỉ IP. Một khi máy tính biết địa chỉ IP của nó, nó có thể liên lạc qua toàn bộ mạng Internet.

Ý tưởng nằm sau việc tìm một địa chỉ IP thật đơn giản: một máy khi cần biết địa chỉ của nó sẽ gửi một yêu cầu đến một "bộ phận phục vụ cấp địa chỉ IP" trên một máy khác, tạm gọi là máy chủ, và đợi lời đáp. Chúng ta giả sử rằng máy chủ có một đĩa để lưu trữ cơ sở dữ liệu về địa chỉ IP. Trong lời yêu cầu, máy tính mà đang cần cấp địa chỉ IP phải phải được xác định duy nhất, để máy chủ gửi lại lời đáp chính xác, cả máy yêu cầu và máy chủ cùng sử dụng địa chỉ mạng vật lý trong quá trình trao đổi của chúng.

Các nhà thiết kế giao thức TCP/IP nhận ra rằng có sẵn một loại thông tin định danh duy nhất, đó là địa chỉ mạng vật lý của máy. Có hai ưu điểm khi sử dụng địa chỉ vật lý là định danh duy nhất. Vì một máy lấy địa chỉ vật lý của nó từ bộ giao tiếp mạng (phần cứng), các địa chỉ đó luôn luôn có sẵn và không phải đưa địa chỉ đó vào chương trình bootstrap. Vì thông tin định danh tuỳ thuộc vào mạng chứ không phải bộ vi xử lý, tất cả các máy trên cùng một mạng sẽ cung cấp các định danh duy nhất và cùng một dạng. Như thế, đây là bài toán ngược của việc giải địa chỉ: cho một địa chỉ mạng vật lý, tìm ra một mô hình sẽ cho phép máy chủ ánh xạ nó vào một địa chỉ Internet.

Giao thức TCP/IP mà cho phép một máy tính tìm ra địa chỉ IP của nó từ một máy chủ được gọi là Reverse Address Resolution Protocol (RARP). RARP được chuyển thể từ giao thức ARP trong chương trước và sử dụng cùng dạng thông điệp như trong hình 3.4. Trong thực tế, thông điệp RARP gửi đi để yêu cầu một địa chỉ IP là tổng quát hơn những gì chúng ta vừa trình bày: nó cho phép một máy gửi yêu cầu tìm địa chỉ IP của một đơn vị thứ ba. Nó cũng cho phép nhiều loại mạng vật lý.

Giống như giao thức ARP, một thông điệp RARP được gửi từ máy này đến máy khác được gửi vào trong phần dữ liệu của một frame. Ví dụ, một frame Ethernet chuyển tải một yêu cầu RARP sẽ có phần mở đầu như thông thường, gồm địa chỉ nguồn và đích Ethernet, và vùng kiểu dữ liệu. Kiểu frame có giá trị 8035 (hệ16) để chỉ rằng đây là một thông điệp RARP. Phần dữ liệu của frame chứa thông điệp RARP dài 28 bytes.




*Hình 3.5: Máy chủ RARP trả lời yêu cầu được cấp địa chỉ IP của máy trạm*

Hình 3.5 cho ta thấy cách mà một máy sử dụng RARP. Máy gửi quảng bá một yêu cầu RARP mà xác định nó vừa là máy gửi vừa là máy đích, và cung cấp địa chỉ mạng vật lý của nó trong vùng địa chỉ phần cứng của máy đích. Tất cả máy trên mạng đều nhận được lời yêu cầu này, nhưng chỉ máy có thẩm quyền cung cấp dịch vụ RARP sẽ xử lý lời yêu cầu và gửi lời đáp; các máy này được gọi là (một cách không chính thức) máy chủ RARP. Để làm được điều này, trên mạng phải có ít nhất một máy chủ RARP.

Máy chủ trả lời yêu cầu bằng cách điền vào vùng địa chỉ đích, thay đổi kiểu thông điệp từ yêu cầu trở thành lời đáp, và gửi lời đáp trực tiếp trở lại máy nào đã yêu cầu. Máy yêu cầu ban đầu sẽ nhận lời đáp từ tất cả máy chủ RARP, mặc dù chỉ cần một lời đáp đầu tiên.

Cũng nên lưu ý rằng, mọi thông tin liên lạc chỉ trên mạng vật lý. Hơn nữa, giao thức cho phép một máy tính gửi lời yêu cầu tới một đích bất kỳ. Vì thế, nơi gửi sẽ cung cấp địa chỉ phần cứng của nó tách biệt với địa chỉ phần cứng của nơi gửi. Trên Ethernet, việc có một cùng dành cho địa chỉ phần cứng của máy đích, và máy chủ sẽ cẩn thận gửi lời đáp trở về địa chỉ phần cứng của máy dường như thừa vì thông tin này cũng có trong phần đầu frame Ethernet. Tuy nhiên, không phải tất cả phần cứng Ethernet đều cho phép hệ điều hành truy xuất phần đầu frame.

## CÂU HỎI VÀ BÀI TẬP

3.1. Khái niệm địa chỉ vật lý MAC Address?

3.2. ARP thường bị chỉ trích là có tính bảo mật kém. Tại sao?

3.3. Bất kỳ cài đặt nào của ARP mà sử dụng vùng cache có kích thước cố định đều có thể không làm việc khi được sử dụng trên một mạng mà có nhiều máy và lượng giao thông lớn. Tại sao?

3.4. Giả sử máy C nhận được một yêu cầu ARP gửi đi từ A để tìm kiếm đích B, và giả sử C có sự phối hợp địa chỉ từ IB và PB trong cache của nó. Liệu rằng C có đáp lời không? Tại sao?

3.5. Hãy giải thích tại sao việc gửi các packet IP đi đến những địa chỉ không tồn tại trên một Ethernet ở xa có thể làm phát sinh lượng lớn giao thông trên mạng?

3.6. Tìm hiểu nguyên lý hoạt động của lệnh ARP trên các hệ điều hành Windows và Linux?

3.7. Cách hoạt động của RARP Server?

3.8. Khuôn dạng của thông điệp ARP?

3.9. Khi muốn tìm một địa chỉ vật lý, máy tính sẽ gửi cho mọi máy trên mạng thông điệp hỏi địa chỉ, làm thế nào để thông điệp hỏi này đến được mọi máy tính khi mà máy gửi chưa biết địa chỉ vật lý của các máy nhận?

3.10. Bài tập: dùng phần mềm chặn và bắt gói tin phổ biến để đọc và phân tích các gói tin ARP trong mạng LAN.

# CHƯƠNG 4 PHÂN LỚP CÁC ĐỊA CHỈ MẠNG, KỸ THUẬT CHIA MẠNG

## 4.1. Phân lớp địa chỉ IP (Internet)

### 4.1.1. Khái niệm địa chỉ IP (Internet)

Như chúng ta đã biết Internet là một mạng máy tính toàn cầu, do hàng triệu mạng máy tính từ khắp mọi nơi nối lại tạo nên. Khác với cách tổ chức theo các cấp: nội hạt, liên tỉnh, quốc tế của một mạng viễn thông như mạng thoại chẳng hạn, mạng Internet tổ chức chỉ có một cấp, các mạng máy tính dù nhỏ, dù to khi nối vào Internet đều bình đẳng với nhau. Do cách tổ chức như vậy nên trên Internet có cấu trúc địa chỉ, cách đánh địa chỉ đặc biệt, trong khi cách đánh địa chỉ đối với mạng viễn thông lại đơn giản hơn nhiều.

Đối với mạng viễn thông như mạng thoại chẳng hạn, khách hàng ở các vùng khác nhau hoàn toàn có thể có cùng số điện thoại, phân biệt với nhau bằng mã vùng, mã tỉnh hay mã quốc tế. Đối với mạng Internet, do cách tổ chức chỉ có một cấp nên mỗi một khách hàng hay một host (Host) hoặc Router đều có một địa chỉ Internet duy nhất mà không được phép trùng với bất kỳ ai. Do vậy mà địa chỉ trên Internet thực sự là một tài nguyên và là tài nguyên có hạn.

Hàng chục triệu host trên hàng trăm nghìn mạng. Để địa chỉ không được trùng nhau cần phải có cấu trúc địa chỉ đặc biệt quản lý thống nhất và một Tổ chức của Internet gọi là Trung tâm thông tin mạng Internet Network Information Center (NIC) chủ trì phân phối, NIC chỉ phân địa chỉ mạng (Net ID) còn địa chỉ host trên mạng đó (Host ID) do các Tổ chức quản lý Internet của từng quốc gia một tự phân phối. (Trong thực tế để có thể định tuyến (routing) trên mạng Internet ngoài địa chỉ IP còn cần đến tên riêng của các host (Host) Domain Name). Các phần tiếp theo chúng ta hãy nghiên cứu cấu trúc đặc biệt của địa chỉ Internet.

### 4.1.2. Khuôn dạng địa chỉ IP

#### *4.1.2.1.Cấu trúc và khuôn dạng dạng của địa chỉ IP*

Địa chỉ IP đang được sử dụng hiện tại (IPv4) có 32 bit chia thành 4 Byte (mỗi Byte có 8 bit, tương đương 1 byte) cách đếm đều từ trái qua phải bít 1 cho đến bít 32, các Byte tách biệt nhau bằng dấu chấm (.), bao gồm có 3 thành phần chính.

| | | |
|---|---|---|
| Class bit | Net ID | Host ID |

Bit 1 ……..32

Bit nhận dạng lớp (Class bit, là các bít đầu tiên của byte1)

Địa chỉ của mạng (Net ID)

Địa chỉ của host (Host ID).

***Ghi chú:*** Tên là Địa chỉ host nhưng thực tế không chỉ có host mà tất cả các máy con (Workstation), các cổng truy nhập v.v..đều cần có địa chỉ.

Bit nhận dạng lớp (Class bit) để phân biệt địa chỉ ở lớp nào.

i) Địa chỉ Internet biểu hiện ở dạng bit nhị phân:

**xxxxxxxx.xxxxxxxx.xxxxxxxx.xxxxxxxx**

x = 0 hoặc 1.

Ví dụ:

| **0** | **0 1 0 1 1 0 0.** | **0 1 1 1 1 0 1 1.** | **0 1 1 0 1 1 1 0.** | **1 1 1 0 0 0 0 0** |
|---|---|---|---|---|
| bit nhận dạng | Byte 1 | Byte 2 | Byte 3 | Byte 4 |

ii) Địa chỉ Internet biểu hiện ở dạng thập phân:

**xxx.xxx.xxx.xxx**

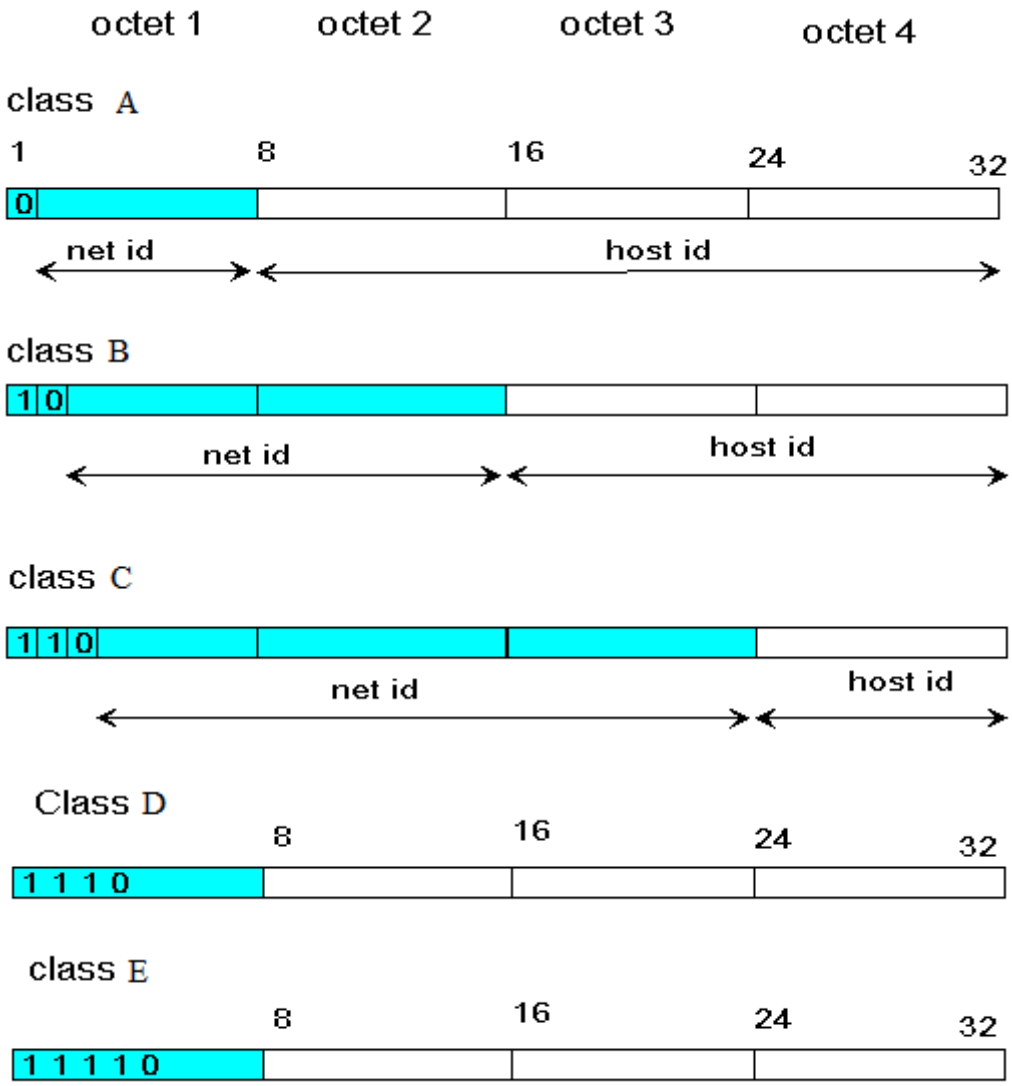
x là số thập phân từ 0 đến 9

Ví dụ: 146. 123. 110. 224

Dạng viết đầy đủ của địa chỉ IP là 3 con số (nhỏ hơn giá trị 256) trong từng Byte. Ví dụ: địa chỉ IP thường thấy trên thực tế có thể là 53.143.10.2 nhưng dạng đầy đủ là 053.143.010.002.

*4.1.2.2. Các lớp địa chỉ IP*

Do quy mô và lực lượng gán địa chỉ IP của từng mạng là nhiều ít khác nhau, do vậy những người thiết kết địa chỉ IP chia ra thành 5 lớp địa chỉ là A,B,C, D, E. Hiện tại đã dùng hết lớp A,B và gần hết lớp C, còn lớp D và E Tổ chức Internet đang để dành cho mục đích khác không phân, nên chúng ta chỉ nghiên cứu 3 lớp đầu.




*Hình 4.1: Các lớp địa chỉ IP*

Qua cấu trúc các lớp địa chỉ IP chúng ta có nhận xét sau:

Bit nhận dạng là những bit đầu tiên của lớp A là 0, của lớp B là 10, của lớp C là 110.

Lớp D có 4 bit đầu tiên để nhận dạng là 1110, còn lớp E có 5 bít đầu tiên để nhận dạng là 11110.

Địa chỉ lớp A: Địa chỉ mạng ít và địa chỉ host trên từng mạng nhiều.

Địa chỉ lớp B: Địa chỉ mạng vừa phải và địa chỉ host trên từng mạng vừa phải.

Địa chỉ lớp C: Địa chỉ mạng nhiều, địa chỉ host trên từng mạng ít.

Ví dụ một số địa chỉ IP

192.1.1.1 địa chỉ lớp C có địa chỉ mạng 192.1.1.0, địa chỉ host là 1

200.6.5.4 địa chỉ lớp C có địa chỉ mạng 200.6.5.0, địa chỉ host là 4

150.150.5.6 địa chỉ lớp B có địa chỉ mạng 150.150.0.0, địa chỉ host là 5.6

9.6.7.8 địa chỉ lớp A có địa chỉ mạng 9.0.0.0, địa chỉ host là 6.7.8

128.1.0.1 địa chỉ lớp B có địa chỉ mạng 128.1.0.0, địa chỉ host là 0.1

i) Bảng thống kê số mạng tối đa và số máy tối đa trong mỗi mạng của mỗi lớp:

| *Địa chỉ lớp* | *Vùng địa chỉ lý thuyết* | *Số mạng tối đa sử dụng* | *Số máy tối đa trên từng mạng* |
|---|---|---|---|
| A | Từ 0.0.0.0 đến 127.0.0.0 | 126 | 16777214 |
| B | Từ 128.0.0.0 đến 191.255.0.0 | 16382 | 65534 |
| C | Từ 192.0.0.0 đến 223.255.255.0 | 2097150 | 254 |
| D | Từ 224.0.0.0 đến 240.0.0.0 | Không phân | |
| E | Từ 241.0.0.0 đến 255.0.0.0 | Không phân | |

*Bảng 4.1. Bảng thống kê số mạng và số máy tối đa*

ii) Bảng thống kê các bít nhận dạng ban đầu:

| *Địa chỉ lớp* | *Vùng địa chỉ sử dụng* | *Các bít đầu tiên của byte đầu tiên (b)it nhận dạng* | *Số bit dùng đểphân cho mạng* |
|---|---|---|---|
| A | Từ 1 đến 127 | 0xxxxxxx | 7 |
| B | Từ 128.1 đến 191.254 | 10xxxxxx | 14 |
| C | Từ 192.0.1 đến 223.255.254 | 110xxxxx | 21 |
| D | | 1110xxxx | |
| E | | 11110xxx | |

*Hình 4.2. Bảng thống kê các bit nhận dạng*

Như vậy nếu chúng ta thấy 1 địa chỉ IP có 4 nhóm số cách nhau bằng dấu chấm, nếu thấy nhóm số thứ nhất nhỏ hơn 126 biết địa chỉ này ở lớp A, nằm trong

khoảng 128 đến 191 biết địa chỉ này ở lớp B và từ 192 đến 223 biết địa chỉ này ở lớp C.

***Ghi chú:*** Địa chỉ thực tế không phân trong trường hợp tất cả các bit trong một hay nhiều Byte sử dụng cho địa chỉ mạng hay địa chỉ host đều bằng 0 hay đều bằng 1. Điều này đúng cho tất cả các lớp địa chỉ.

*4.1.2.3. Khảo sát chi tiết lực lượng địa chỉ của các lớp*

Chúng ta cùng nhau khảo sát lớp A, các lớp B, C hoàn toàn tương tự

Tổng quát chung:

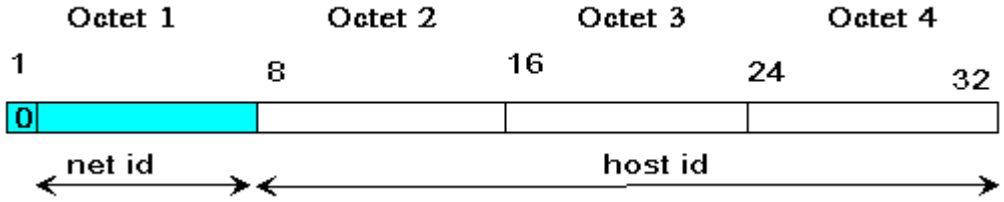
Bit thứ nhất là bit nhận dạng lớp A = 0.

7 bit còn lại trong Byte thứ nhất dành cho địa chỉ mạng.

3 Byte còn lại có 24 bit dành cho địa chỉ của host.

Cứ 8bit hợp với nhau từ trái qua phải gọi là Octet.

**Class A: (0 126)**




*Hình 4.2: Cấu trúc địa chỉ IP lớp A*

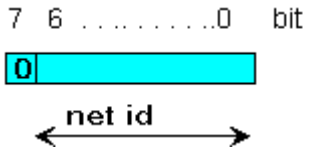
**net id: 126 mạng**
**host id:16.777.214 host trên một mạng**

i) Địa chỉ mạng (Net ID) của lớp A

* Khả năng phân địa chỉ

Khi đếm số bit chúng ta đếm từ trái qua phải, nhưng khi tính giá trị thập phân $2^n$ của bit lại tính từ phải qua trái, bắt đầu từ bit 0. Byte thứ nhất dành cho địa chỉ mạng, bit 7 = 0 là bit nhận dạng lớp A. 7 bit còn lại từ bit 0 đến bit 6 dành cho địa chỉ mạng ($2^{7)}$ = 128. Nhưng trên thực tế địa chỉ khi tất cả các bit bằng 0 hoặc bằng 1 đều không phân cho mạng. Khi giá trị các bit đều bằng 0, giá trị thập phân 0 là không có nghĩa, còn địa chỉ là 127 khi các bit đều bằng 1 dùng để thông báo nội bộ, nên trên thực tế còn lại 126 mạng.

**Byte 1**




Cách tính địa chỉ mạng lớp A.

Số thứ tự Bit (n) tính từ phải qua trái: 6 5 4 3 2 1 0

Giá trị nhị phân (0 hay 1) của Bit: x x x x x x x

Giá trị thập phân tương ứng khi giá trị bit = 1 sẽ là $2^n$

Giá trị thập phân tương ứng khi giá trị bit = 0 không tính.

Giá trị thập phân lớn nhất khi giá trị của 7 bit đều bằng 1 là 127.

Xin xem bảng tính trọn vẹn giá trị của tất cả các Bit

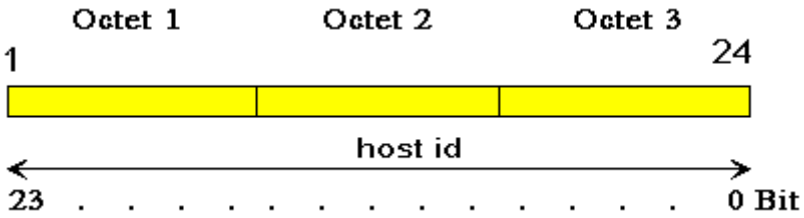
**Như vậy khả năng phân địa chỉ của lớp A cho 126 mạng**

* Biểu hiệu địa chỉ trên thực tế: Từ 001 đến 126

ii) Địa chỉ của các máy trên một mạng của lớp A

* Khả năng phân địa chỉ

Ba Byte sau gồm 24 bit được tính từ bit 0 đến bit 23 dành cho địa chỉ host trên từng mạng.




Với cách tính như trên, để được tổng số host trên một mạng ta có.

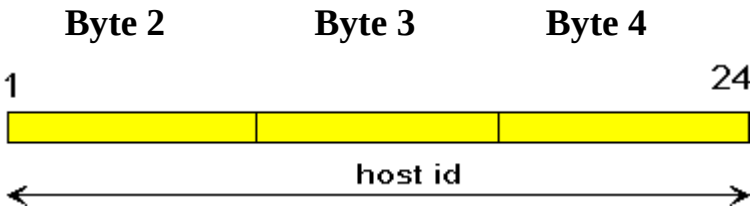
| *Gía trị tương ứng với Bit n*<br>23.22.21.20.19.18.16.\|15.14.13.12.11.10.9.8.\|7.6.5.4.3.2.1.0 | *Giá trị 2n* | *Địa chỉ* |
|---|---|---|
| ..0...0...0...0...0...0...0.\|.0...0...0...0...0...0..0.0.\|0.0.0.0.0.0.0.0 | | 000 |
| ..0...0...0...0...0...0...0.\|.0...0...0...0...0...0..0.0.\|0.0.0.0.0.0.0.0 | $2^0$ | 001 |
| ..0...0...0...0...0...0...0.\|.0...0...0...0...0...0..0.0.\|0.0.0.0.0.0.0.0 | $2^1$ | 002 |
| ..... | ..... | ..... |
| ..... | ..... | ..... |
| ..1...1...1...1...1...1...1.\|.1...1...1...1...1...1..1.1.\|1.1.1.1.1.1.1.0 | $2^{23}$+...+$2^1$ | 16777214 |
| ..1...1...1...1...1...1...1.\|.1...1...1...1...1...1..1.1.\|1.1.1.1.1.1.1.1 | | 16777215 |
| < Byte2 > < Byte3 > < Byte4 > | | |

*Bảng 4.3. Số Host một mạng lớp A*

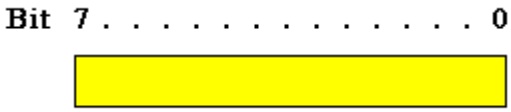
Địa chỉ khi các bit đều bằng 0 hay bằng 1 bỏ ra không dùng đến theo quy ước. Trên thực tế còn lại 224 2 = 16 777 214

Như vậy khả năng phân địa chỉ cho 16 777 214 host.

* Biểu hiện địa chỉ trên thực tế




**Byte 2**

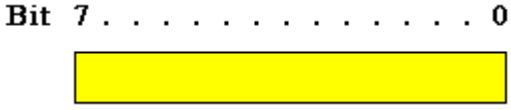



| *Gía trị tương ứng với thứ tự bit (n) 76543210* | *Giá trị 2n* | *Địa chỉ host* |
|---|---|---|
| *00000000* | | *000* |
| *00000001* | *20* | *001* |
| *00000010* | *21* | *002* |
| *00000011* | *21+20* | *003* |
| ..... | ..... | ..... |
| ..... | ..... | ..... |
| *11111111* | *27+26+25+24+23+22+21+20* | *255* |

*Bảng 4.4: Địa chỉ của Byte 2*

**Như vậy giá trị thập phân ở Byte 2 tính từ 000 tới 255.**

**Byte 3**



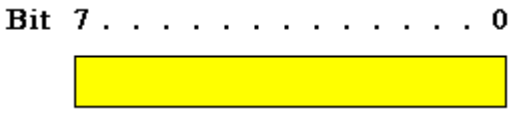

| *Gía trị tương ứng với thứ tự bit (n) 76543210* | *Giá trị 2n* | *Địa chỉ host* |
|---|---|---|
| *00000000* | | *000* |
| *00000001* | *20* | *001* |
| *00000010* | *21* | *002* |
| *00000011* | *21+20* | *003* |

| | | |
|---|---|---|
| ..... | ..... | ..... |
| ..... | ..... | ..... |
| *11111111* | *27+26+25+24+23+22+21+20* | *255* |

*Bảng 4.5: Địa chỉ của Byte 3*

**Như vậy giá trị thập phân ở Byte 3 tính từ 000 tới 255.**

**Byte 4**




| *Gía trị tương ứng với thứ tự bit (n)* 76543210 | *Giá trị 2n* | *Địa chỉ host* |
|---|---|---|
| 00000000 | | 000 Không phân |
| 00000001 | $2^0$ | 001 |
| 00000010 | $2^1$ | 002 |
| 00000011 | $2^1+2^0$ | 003 |
| ..... | ..... | ..... |
| ..... | ..... | ..... |
| 11111110 | $2^7+2^6+2^5+2^4+2^3+2^2+2^1$ | 254 |
| 11111111 | $2^7+2^6+2^5+2^4+2^3+2^2+2^1+2^0$ | 255 Không phân |

*Bảng 4.6: Địa chỉ của Byte 4*

Như vậy giá trị thập phân ở Byte 4 tính từ 001 tới 254.

* Tổng quát lại tại địa chỉ của một mạng, khi lần lượt thay đổi các giá trị của các Byte 2, 3, 4.ta sẽ có 16 777 216 khả năng thay đổi mà các con số không trùng lặp nhau (Combinations) có nghiã là 16 777 216 địa chỉ của host trên mạng, nhưng thực tế phân chỉ là

(256 x 256 x 256) 2 =16 777 214

Biểu hiện trên thực tế là ba số thập phân trong 3 Byte cách nhau dấu.

Từ 000. 000. 0001 đến 255. 255. 254

Kết luận: Địa chỉ lớp A có thể phân cho 126 mạng và mỗi một mạng có 16 777 214 host. Nói cách khác địa chỉ thực tế sẽ từ 001.000.000.001 đến 126.255.255.254

Ví dụ:

Một địa chỉ đầy đủ của lớp A: 124. 234. 200. 254. Trong đó:

Địa chỉ mạng: 124

Địa chỉ host: 234.200.254

## 4.2. Kỹ thuật chia mạng con (IP SUBNETTING)

Như đã nêu trên địa chỉ trên Internet thực sự là một tài nguyên, một mạng khi gia nhập Internet được Trung tâm thông tin mạng Internet (NIC) phân cho một số lượng địa chỉ vừa đủ dùng với yêu cầu lúc đó, sau này nếu mạng phát triển thêm lại phải xin NIC thêm, đó là điều không thuận tiện cho các nhà khai thác mạng.

Mô hình địa chỉ IP phân lớp ban đầu dường như xử lý được tất cả mọi tình huống, nhưng nó có một nhược điểm rất lớn mà hiện tại chúng ta phải giải quyết, đó là: sự phát triển quá nhanh của mạng Internet. Bởi vì ban đầu họ (những người thiết kế lên mạng Internet) làm việc trong một thế giới các máy tính lớn (mainframe) đắt tiền, những người thiết chỉ thấy được một Internet với hàng trăm mạng và hàng ngàn máy tính lớn. Họ đã không tiên liệu được cả chục ngàn mạng nhỏ với các máy tính cá nhân bỗng dưng xuất hiện trong vòng chỉ một thập niên sau khi TCP/IP được thiết kế.

Internet đã và đang phát triển rất nhanh, cứ khoảng chín tới mười lăm tháng thì độ lớn (số máy, mạng) của nó tăng lên gấp đôi. một số lượng lớn các mạng với kích thước trung bình đã làm việc cấp phát địa chỉ IP trở thành rất căng thẳng vì:

- Chỉ đơn thuần các yêu cầu quả lý các địa chỉ mạng cũng là công việc "hành chánh" khổng lồ,

- Các lớp địa chỉ mau chóng bị can kiệt (mặc dù trước đây đã có nhiều sự đoán rằng không gian địa chỉ IPv4 sẽ bị cạn kiệt trước năm 2000, tuy nhiên với sự cấp phát cẩn thận và các kỹ thuật được mô tả trong chương này, các địa chỉ IPv4 sẽ vẫn đủ cho đến khoảng năm 2019). Điều quan trọng là vấn đề thứ hai bởi vì nó có nghĩa rằng khi các bộ định tuyến trao đổi thông tin từ các bộ định tuyến của chúng, lượng giao dịch trên Internet rất cao, cũng như các yêu cầu tính toán của các bộ định tuyến tham gia.

- Vấn đề thứ ba là cốt tử bởi vì mô hình địa chỉ ban đầu không thể dung nạp được số lượng mạng hiện tại trên Internet. Ví dụ, lớp địa chỉ B hiện tại không đủ dung nạp tất cả các mạng kích thước trung bình. Vì thế câu hỏi đặt ra là: "Làm sao chúng ta có thể giảm thiểu số lượng địa chỉ mạng được gán, đặc biệt là lớp B, mà không từ bỏ mô hình địa chỉ 32 bit?"

Hơn nữa các lớp địa chỉ của Internet không phải hoàn toàn phù hợp với yêu cầu thực tế, địa chỉ lớp B chẳng hạn, mỗi một địa chỉ mạng có thể cấp cho 65534 host, Thực tế có mạng nhỏ chỉ có vài chục host thì sẽ lãng phí rất nhiều địa chỉ còn

lại mà không ai dùng được. Để khắc phục vấn đề này và tận dụng tối đa địa chỉ được NIC phân, bắt đầu từ năm 1985 người ta nghĩ đến một biện pháp phân chia một mạng thành các mạng con độc lập với nhau.

Như vậy phân địa chỉ mạng con là mở rộng địa chỉ cho nhiều mạng trên cơ sở **một địa chỉ mạng** mà NIC phân cho, phù hợp với số lượng thực tế host có trên từng mạng.

4.2.1. Phương pháp phân chia subnet

Trước khi nghiên cứu phần này chúng ta cần phải hiểu qua một số khái niệm liên quan tới việc phân địa chỉ các mạng con.

Default Mask: (Giá trị trần địa chỉ mạng) được định nghĩa trước cho từng lớp địa chỉ A,B,C. Thực chất là giá trị thập phân cao nhất (khi tất cả 8 bit đều bằng 1) trong các Byte dành cho địa chỉ mạng Net ID.

Default Mask:

Lớp A 255.0.0.0

Lớp B 255.255.0.0

Lớp C 255.255.255.0

Subnet Mask: Phân định Net ID và HostID

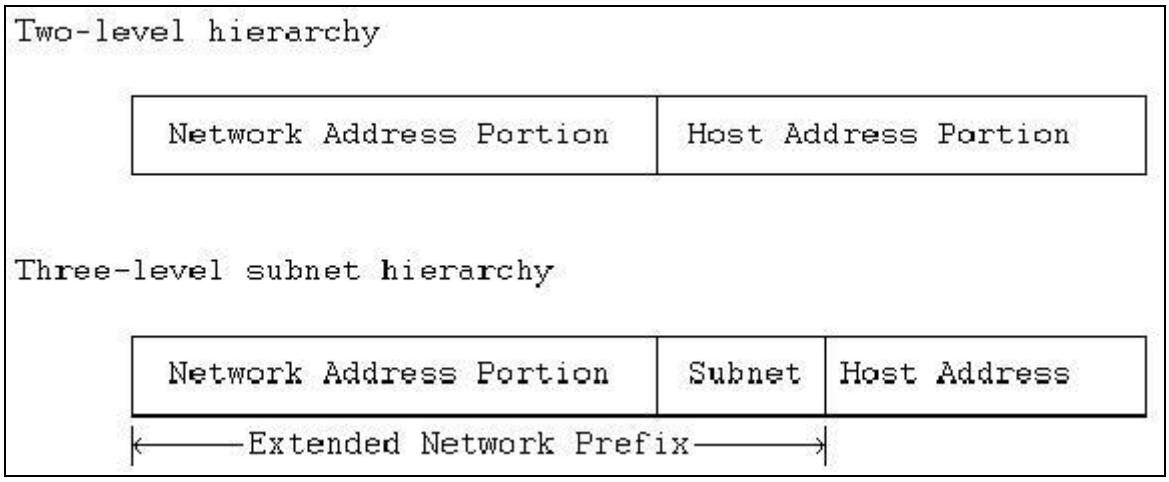
Subnet Mask là kết hợp của Default Mask với giá trị thập phân cao nhất của các bit lấy từ các Byte của địa chỉ host sang phần địa chỉ mạng để tạo địa chỉ mạng con. Về mặt cài đặt thì subnet mask là một chuỗi 0/1 có độ dài 4 bytes, với những bit có giá trị bằng 1 ở bên subnet mask thì tương ứng với nó, những bit ở phần địa chỉ IP là chỉ địa chỉ mạng (net_id), những bit có giá trị bằng 0 ở bên subnet mask thì tương ứng với nó, những bit ở phần địa chỉ IP là chỉ địa chỉ máy thuộc mạng (host_id). Như vậy về mặt cài đặt trên máy tính, một địa chỉ IP đầy đủ phải bao gồm cặp giá trị (địa chỉ IP, subnet mask của địa chỉ IP).

Ngoài việc giúp cho máy tính dễ phân biệt trong cài đặt, subnet mask bao giờ cũng đi kèm với địa chỉ mạng tiêu chuẩn còn nhằm báo cho người quản trị mạng biết địa chỉ mạng tiêu chuẩn này dùng cả cho 254 host hay chia ra thành các mạng con. Mặt khác nó còn giúp Router trong việc định tuyến để truyền dữ liệu.

**Nguyên tắc chung của kỹ thuật chia subnet:**

Lấy bớt một số bit của phần địa chỉ host để tạo địa chỉ mạng con.

Lấy đi bao nhiêu bit phụ thuộc vào số mạng con cần thiết (subnet mask) mà nhà khai thác mạng quyết định sẽ tạo ra.




*Hình 4.3: Nguyên tắc chia subnet*

***Ví dụ***

Để hiểu chi tiết phương pháp này chúng ta cùng nhau khảo sát chi tiết một ví dụ về việc phân chia mạng con của một mạng lớp C, cụ thể như sau:

Mạng lớp C: 205.131.175.0 / 255.255.255.0

Lấy 3 bít cao nhất của phần HostID làm subnet, như vậy số mạng con tối đa phân giải ra thành: 8 mạng

Base Network: 11001101.10000011.10101111.00000000 = 205.131.175.0

Subnet #0: 11001101.10000011.10101111.00000000 = 205.131.175.0

Subnet #1: 11001101.10000011.10101111.00100000 = 205.131.175.32

Subnet #2: 11001101.10000011.10101111.01000000 = 205.131.175.64

Subnet #3: 11001101.10000011.10101111.01100000 = 205.131.175.96

Subnet #4: 11001101.10000011.10101111.10000000 = 205.131.175.128

Subnet #5: 11001101.10000011.10101111.10100000 = 205.131.175.160

Subnet #6: 11001101.10000011.10101111.11000000 = 205.131.175.192

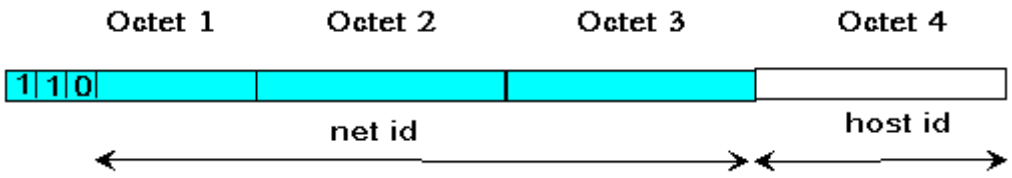
Subnet #7: 11001101.10000011.10101111.11100000 = 205.131.175.224

Trong những năm đầu của sự phát triển mạng INTERRNET thì không dùng hết cả 8 subnet này, bởi vì quy ước không sử dụng phần địa chỉ khi tất cả các bít cùng bằng 1 hoặc cùng bằng 0, (tuy nhiên, về sau do lực lượng địa chỉ IP ngày càng cạn kiệt, cho nên người ta sử dụng tất cả các phân lớp địa chỉ được chia – do vậy một số tài liệu chính tắc thì nói không sử dụng, một số tài liệu về mặt sử dụng

vẫn ghi sử dụng các lớp con này), sau đây chúng ta đi vào cụ thể việc phân giải subnet ở từng lớp địa chỉ.

*4.2.1.1. Địa chỉ mạng con của địa chỉ lớp C*

Class c:




*Hình 4.4: Địa chỉ lớp C*

Địa chỉ lớp C có 3 byte cho địa chỉ mạng và 1 byte cuối cho địa chỉ host vì vậy chỉ có 8 bit lý thuyết để tạo mạng con, thực tế nếu dùng 1 bit để mở mạng con và 7 bit cho địa chỉ host thì vẫn chỉ là một mạng và ngược lại 7 bit để cho mạng và 1 bit cho địa chỉ host thì một mạng chỉ được một máy (do không sử dụng tất cả các bit=0 hoặc tất cả các bit=1), như vậy không logic, ít nhất phải dùng 2 bit để mở rộng địa chỉ và 2 bit cho địa chỉ host trên từng mạng. Do vậy trên thực tế chỉ dùng như bảng sau.

**Default Mask của lớp C: 255.255.255.0**

*Địa chỉ host*

< >

(255.255.255).**1 1** 0 *0 0 0 0 0*; 192 (2 bit đ/ chỉ mạng con 6 bit đ/chỉ host)

(255.255.255).**1 1 1** *0 0 0 0 0*; 224 (3 bit đ/chỉ mạng con 5 bit đ/chỉ host)

(255.255.255).**1 1 1 1** *0 0 0 0*; 240 (4 bit đ/chỉ mạng con 4 bit đ/chỉ host)

(255.255.255).**1 1 1 1 1** *0 0 0*; 248 (5 bit đ/chỉ mạng con 3 bit đ/chỉ host)

(255.255.255).**1 1 1 1 1 1** *0 0*; 252 (6 bit đ/chỉ mạng con 2 bit đ/chỉ host)

< > < >

Default Mask **Địa chỉ**

**mạng con**

Thống kê các khả năng chia mạng con của lớp C

*Trường Subnetmask Số lượng Số host trên*

*hợp mạng con từng mạng*

1 255.255.255.192 2 62

2 255.255.255.224 6 30

3 255.255.255.240 14 14

4 255.255.255.248 30 6

5 255.255.255.252 62 2

Như vậy một địa chỉ mạng ở lớp C chỉ có 5 trường hợp lựa chọn trên (Hay 5 Subnet Mask khác nhau), tuỳ từng trường hợp cụ thể để quyết định số mạng con.

Phân chia mạng con rất quan trọng, giúp tiết kiệm địa chỉ IP phải cung cấp cho các tổ chức. Một tổ chức có thể cần nhiều cụm máy chủ làm việc độc lập, thay vì ta cấp địa chỉ mạng cho từng cụm máy chủ trong khi mỗi cụm máy chỉ có sốlượng máy ít thì ta chỉ cần cung cấp 1 địa chỉ rồi từ đó sử dụng kỹ thuật chia subnet, mỗi subnet tương ứng với 1 cụm máy chủ.

**i) Trường hợp 1: hai mạng con**

Subnet Mask 255.255.255.192.

Từ một địa chỉ tiêu chuẩn tạo được địa chỉ cho hai mạng con, mỗi một mạng có 62 host.

Sử dụng hai bit (bit 7 và 6) của phần địa chỉ host để tạo mạng con. Như vậy còn lại 6 bit để phân cho host.

*Tính địa chỉ mạng*

| **3 Byte đầu** | **Byte 4** | Địa chỉ mạng các mạng con |
|---|---|---|
| | Bít 7 6 5 4 3 2 1 0 | |
| xxx.xxx.xxx. | *0 0 0 0 0 0 0 0* | *= xxx.xxx.xxx.0* |
| xxx.xxx.xxx. | 0 1 0 0 0 0 0 0 | = xxx.xxx.xxx.64 |
| xxx.xxx.xxx. | 1 0 0 0 0 0 0 0 | = xxx.xxx.xxx.128 |
| xxx.xxx.xxx. | 1 1 0 0 0 0 0 0 | = xxx.xxx.xxx.192 |

Ghi chú: xxx.xxx.xxx là địa chỉ mạng tiêu chuẩn của lớp C.

Địa chỉ của mạng là giá trị của bit 7 và 6 lần lượt bằng 0 và 1. Trong trường hợp chính tắc thì chia địa chỉ mạng con không bao giờ được dùng địa chỉ khi các bit đều bằng 0 hay bằng 1. Do vậy trường hợp 2 mạng con nói trên, địa chỉ mạng con sẽ là:

Mạng con 1: Địa chỉ mạng xxx.xxx.xxx.64

Mạng con 2: Địa chỉ mạng xxx.xxx.xxx.128

Tính địa chỉ cho host cho mạng con 1

Chúng ta chỉ còn 6 bit cho địa chỉ host trên từng mạng.

**Byte 4**

| Bit 7 6 5 4 3 2 1 0 | |
|---|---|
| xxx.xxx.xxx. 0 1 0 0 0 0 0 0 | = xxx.xxx.xxx.64 Địa chỉ mạng |
| xxx.xxx.xxx. 0 1 0 0 0 0 0 1 | = xxx.xxx.xxx.65 |
| xxx.xxx.xxx. 0 1 0 0 0 0 1 0 | = xxx.xxx.xxx.66 |
| ..... ..... | ..... |
| xxx.xxx.xxx. 0 1 1 1 1 1 1 0 | = xxx.xxx.xxx.126 |
| xxx.xxx.xxx. 0 1 1 1 1 1 1 1 | =xxx.xxx.xxx.127 Không phân |

Còn lại 62 địa chỉ cho host.

Mạng 1: Từ xxx.xxx.xxx. 065 đến xxx.xxx.xxx.126

Tính địa chỉ cho host cho mạng con 2

Tương tự như cách tính trên ta có

Byte 4

| Bit 7 6 5 4 3 2 1 0 | |
|---|---|
| xxx.xxx.xxx. 1 0 0 0 0 0 0 0 | = xxx.xxx.xxx.128 Địa chỉ mạng |
| xxx.xxx.xxx. 1 0 0 0 0 0 0 1 | = xxx.xxx.xxx.129 |
| xxx.xxx.xxx. 1 0 0 0 0 0 1 0 | = xxx.xxx.xxx.130 |
| ..... ..... | ..... |
| xxx.xxx.xxx. 1 0 1 1 1 1 1 0 | = xxx.xxx.xxx.190 |
| xxx.xxx.xxx. 1 0 1 1 1 1 1 1 | = xxx.xxx.xxx.191 Không phân |

Từ xxx.xxx.xxx.129 đến xxx.xxx.xxx.190.

***Ví dụ:*** Địa chỉ tiêu chuẩn lớp C là 196. 200. 123

Subnetmask 255.255.255.192

Từ địa chỉ này ta có 2 mạng con là:

* Mạng 1: Địa chỉ mạng 196.200.123.064

Địa chỉ Host trên mạng này.

Từ 196.200.123.065 đến 196. 200. 123. 126.

* Mạng 2: Địa chỉ mạng 196.200.123.128

Địa chỉ host trên mạng này.

Từ 196.200.123.129 đến 196.200.123. 190

**ii) Trường hợp 2: sáu mạng con**

- Subnetmask: 255.255.255.224.
- Tạo được 6 mạng con, mỗi mạng con có 30 host

Tính địa chỉ Mạng con

Trường hợp này sử dụng 3 bit (bit 7,6,5) của địa chỉ host (Byte 4) bổ sung cho địa chỉ mạng tiêu chuẩn để tạo mạng con.

| | **Byte 4** | **Địa chỉ các mạng con** |
|---|---|---|
| | Bit 7 6 5 4 3 2 1 0 | |
| *xxx.xxx.xxx.* | *0 0 0 0 0 0 0 0* | *= xxx.xxx.xxx.0* |
| xxx.xxx.xxx. | 0 0 1 0 0 0 0 0 | = xxx.xxx.xxx.32 |
| xxx.xxx.xxx. | 0 1 0 0 0 0 0 0 | = xxx.xxx.xxx.64 |
| xxx.xxx.xxx. | 0 1 1 0 0 0 0 0 | = xxx.xxx.xxx.96 |
| xxx.xxx.xxx. | 1 0 0 0 0 0 0 0 | = xxx.xxx.xxx.128 |
| xxx.xxx.xxx. | 1 0 1 0 0 0 0 0 | = xxx.xxx.xxx.160 |
| xxx.xxx.xxx. | 1 1 0 0 0 0 0 0 | = xxx.xxx.xxx.192 |
| *xxx.xxx.xxx.* | *1 1 1 0 0 0 0 0* | *= xxx.xxx.xxx.224* |

Bỏ trường hợp các bit đều bằng 0 hay 1, chúng ta còn lại địa chỉ của 6 mạng con sau.

xxx.xxx.xxx.32; Mạng con 1

xxx.xxx.xxx.64; Mạng con 2

xxx.xxx.xxx.96; Mạng con 3

xxx.xxx.xxx.128; Mạng con 4

xxx.xxx.xxx.160; Mạng con 5

xxx.xxx.xxx.192; Mạng con 6

Tính địa chỉ host cho mạng con 1

| | Byte 4 | Địa chỉ các mạng con |
|---|---|---|
| Bit | 7 6 5 4 3 2 1 0 | |
| xxx.xxx.xxx. | 0 0 1 0 0 0 0 0 | = xxx.xxx.xxx. 32 Địa chỉ mạng |
| xxx.xxx.xxx. | 0 0 1 0 0 0 1 1 | = xxx.xxx.xxx.33 |
| xxx.xxx.xxx. | 0 0 1 0 0 0 0 0 | = xxx.xxx.xxx.34 |
| xxx.xxx.xxx. | 0 0 1 0 0 0 1 1 | = xxx.xxx.xxx.35 |
| xxx.xxx.xxx. | 0 0 1 0 0 1 0 0 | = xxx.xxx.xxx.36 |
| ..... | .... .... | ..... |
| xxx.xxx.xxx. | 0 0 1 1 1 1 1 0 | = xxx.xxx.xxx.62 |
| xxx.xxx.xxx. | 0 0 1 1 1 1 1 1 | = xxx.xxx.xxx.63 Không phân |

Như vậy địa chỉ host của mạng con 1 sẽ từ 33 đến 62.

Tương tự như cách tính đã nêu trên chúng ta có thể tính được cho tất cả các trường hợp còn lại.

Tổng hợp lại các trường hợp như sau:

**1/ Trường hợp 1: Subnetmask 255.255.255.192**

- 2 mạng con.
- 62 host mỗi mạng.

**2/ Trường hợp 2: Subnetmask 255.255.255.224**

- 6 mạng con.
- 30 host mỗi mạng.

**3/ Trường hợp 3: Subnetmask 255.255.255.240**

- 14 mạng con.
- 14 hostmỗi mạng

**4/ Trường hợp 4: Subnetmask 255.255.255.248**

- 30 mạng con.
- 6 host mỗi mạng.

**5/ Trường hợp 5: Subnetmask 255.255.255.252.**

- 62 mạng con.

- 2 host mỗi mạng.

Xem bảng tính địa chỉ cho các trường hợp trên

**Ví dụ:** Địa chỉ mạng lớp C mà NIC phân cho VDC là 203.162.4.0. Trên địa chỉ này phân ra 2 mạng con thì địa chỉ sẽ là.

Mạng 1: Địa chỉ mạng 203.162.4.64.

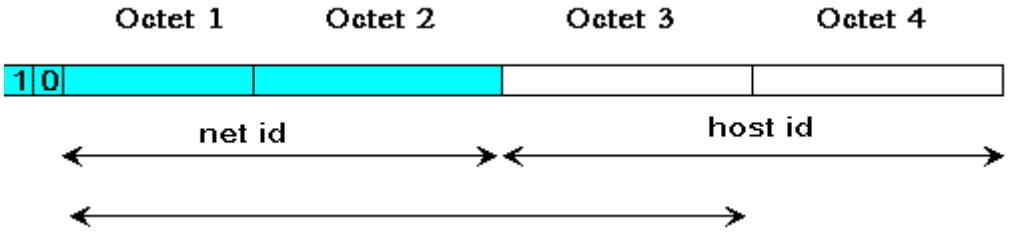
Địa chỉ host trên mạng đó từ 203.162.4.65 đến 203.162.4.126

Mạng 2: Địa chỉ mạng 203.162.4.128.

Địa chỉ host trên mạng đó từ 203.162.4.129 đến 203.162.4.190

*4.2.1.2. Địa chỉ mạng con của địa chỉ lớp B*

**Class b:**




*Hình 4.5: Địa chỉ lớp B*

Default Mask của lớp B là 255.255.0.0

Net ID Khi phân địa chỉ mạng con sử dụng Byte 3

Địa chỉ lớp B có 2 Byte thứ 3 và thứ 4 dành cho địa chỉ host nên về nguyên lý có thể lấy được cả 16 bit để tạo địa chỉ mạng. Nếu từ một địa chỉ mạng được NIC phân chúng ta định mở rộng lên 254 mạng và mỗi mạng sẽ có 254 host. Trường hợp này sẽ lấy hết 8 bit của byte thứ 3 bổ sung vào địa chỉ mạng và chỉ còn lại 8 bit thực tế cho địa chỉ host, theo cách tính số thập phân $2^n$ giá trị của 8 bit như đã nêu ở phần lớp C, chúng ta sẽ có:

**Bảng phân chia địa chỉ mạng con ở lớp B**

| Class B Subnetting (Default Subnet mask) 255.255.0.0 | Subnet Mask | #of subnets Số mạng con | #of hosts per subnet Số host trên mỗi mạng con |
|---|---|---|---|
| Sử dụng Byte 3 để mở rộng mạng con | 255.255.192.0 | 2 | 16382 |
| | 255.255.224.0 | 6 | 8190 |
| | 255.255.240.0 | 14 | 4094 |

| | | | |
|---|---|---|---|
| | 255.255.248.0 | 30 | 2460 |
| | 255.255.252.0 | 62 | 1022 |
| | 255.255.254.0 | 126 | 510 |
| | 255.255..255.0 | 254 | 254 |
| Sử dụng cả Byte 4 để mở rộng mạng con | 255.255.255.128 | 510 | 126 |
| | 255.255.255.192 | 1022 | 62 |
| | 255.255.255.224 | 2046 | 30 |
| | 255.255.255.240 | 4094 | 14 |
| | 255.255.255.248 | 8190 | 6 |
| | 255.255.255.252 | 16382 | 2 |

*Hình 4.7. Bảng phân chia địa chỉ mạng con lớp B*

Địa chỉ lớp B về lý thuyết có 2 byte đầu cho địa chỉ mạng, khi chia mạng con theo phương pháp sử dụng tất cả 8 bit trong 3 byte cho địa chỉ mạng, trên thực tương ứng với lớp C, như vậy về địa chỉ NIC phân là lớp B nhưng cách tổ chức địa chỉ lại ở lớp C

Ví dụ: Trường hợp Subnetmask 255.255.240.0.

Chia được 14 mạng con, mỗi mạng con có 4094 host, khoảng cách địa chỉ giữa hai mạng con là 16.0 có nghĩa là.

- Mạng con 1 có địa chỉ là xxx.xxx.16.0; Mạng con 2 sẽ có địa chỉ là xxx.xxx.16.0 + 16.0 = xxx.xxx.32.0 cứ tiếp tục như vậy ta sẽ tính được địa chỉ của từng mạng con và mạng con 14 là xxx.xxx. 224.0.
- Địa chỉ host đầu tiên trên mạng con 1 là xxx.xxx.16.1; địa chỉ host đầu tiên trên mạng con 2 sẽ là xxx.xxx.16.1 + 16.0 = xxx.xxx.32.1. Tiếp tục như vậy ta sẽ tính địa chỉ được host đầu tiên của mạng con 14 là xxx.xxx.224.1 v.v..
- Tương tự chúng ta biết được địa chỉ cuối cùng của các host trên một mạng con.

Theo hướng dẫn này chúng ta sẽ tìm được các trường hợp khác.

Tóm lại chia địa chỉ mạng con cũng phải theo một quy luật nhất định ngoài ý muốn của chúng ta, khi chia mạng con cũng bị mất khá nhiều địa chỉ, mất ít hay nhiều tuỳ thuộc vào các trường hợp cụ thể.

### 4.2.2. Mặt nạ mạng con

#### *4.2.2.1. Cài đặt mạng con với mặt nạ (mask)*

Kỹ thuật mạng con làm cho việc cấu hình dễ dàng hơn, đối với cả hai trường hợp độ dài cố định biến đổi. Một mặt nạ 32 bit được dùng để xác định sự phân chia. Vì thế một đơn vị sử dụng mạng con phải chọn một mặt nạ mạng con 32 bit cho mỗi mạng. Các bít trong mặt nạ mạng con có giá trị bằng 1 nếu các máy trên mảng xem bít tương ứng trong địa chỉ IP như một phần của tiền tố mạng con, có giá trị bằng 0 nếu chúng xem bít này như một phần của địa chỉ máy.

Ví dụ, xét mặt nạ mạng con 32 bít:

11111111 11111111 11111111 00000000

Mô tả rằng ba byte đầu tiên xác định mạng và bytes thứ tư xác định máy tính trên mạng đó. Một mặt nạ mạng con phải có tất cả các bit là một trong phần tương ứng với phần mạng của địa chỉ (ví dụ, mặt nạ mạng con cho một mạng lớp B sẽ có các bit là một trong 2 bytes đầu tiên cùng với 1 hay nhiều bit trong hai bytes cuối cùng).

Một điểm quan trọng trong việc địa chỉ mạng con là chuẩn không ràng buộc mặt nạ mạng con phải là các bit liên tục nhau trong địa chỉ. Ví dụ, có thẻ gán mặt nạ sau đây cho mạng:

11111111 11111111 00011000 01000000

Gồm hai byte đầu tiên, hai bit trong byte thứ ba và một bít trong byte cuối cùng. Mặc dù sự uyển chuyển trên cho phép ta sắp xếp các phép gán địa chỉ đặc biệt cho mát, nhưng nó làm cho việc gán địa chỉ và hiểu bảng định tuyến trở thành công việc chi li lắt léo.Vì vậy, các đơn vị được khuyên nên sử dụng mặt nạ mạng con liên tục và nên sử dụng cùng mặt nạ trên toàn bộ các mạng vật lý có cùng chung một địa chỉ IP.

#### *4.2.2.2. Cách thể hiện mặt nạ mạng con*

Việc mô tả mặt nạ mạng con dưới dạng nhị phân vừa khó diễn tả và dễ bi lỗi. Vì vậy, hầu hết phần mềm đều có cách thể hiện khác. Hầu hết phần mềm IP sử dụng cách thể hiện dấu chấm thập phân cho mặt nạ mạng con; nó thích hợp nhất cho những đơn vị chọn mạng con theo từng bytes. Ví tụ, nhiều tổ chức chọn lập mạng con các địa chỉ lớp B bằng cách sử dụng bytes thứ ba để xác định mạng vật lý và bytes thứ tư để xác định máy tính như được trình bày trong phàn trước. Trong trường hợp đó, mạng con có cách thể hiện dấu chấm thập phân 255.255.255.0, giúp dễ hiểu và dễ viết.

Cũng có một cách thể hiện khác cho địa chỉ mạng con và mặt nạ mạng con, ở dạng nhóm 3 trong ngoặc nhọn.

{<số của mạng>, <số mạng con>, số của máy>}

theo cách thể hiện này, giá trị 1 có nghĩa là "tất cả đều là 1". Ví dụ, nếu mặt nạ mạng con củamột mạng lớp B là 255.255.255.0, thì ta có thể viết lại là { 1, 1, 0}.

Nhược điểm chính của cách thể hiện nhóm 3 là nó không mô tả chính xác bao nhiêu bit được dùng cho mỗi phần của địa chỉ; ưu điểm là nó tách bỏ hoá đi những chi tiết về các bit và nhấn mạnh giá trị của ba phần của địa chỉ. Để thấy được tại sao đôi khi các giá trị địa chỉ quan trọng hơn các vùng bit, xét nhóm 3 sau đây:

{128.10, 1. 0}

Để mô tả địa chỉ với số của mạng là 128.10, vùng mạng con gồm tất cả là bít 1, và vùng máy tính gồn tất cả các bít 0. nếu thể hiện địa chỉ này bằng cách khác sẽ đòi hỏi một mặt nạ mạng con 32 bit cùng với một địa chỉ 32 bit, và người đọc phải giải mã các vùng bit để có được giá trị của từng vùng. Hơn nữa, cách thể hiện tập hợp các địa chỉ hoặc các ý tưởng trừu tượng. Ví dụ, nhóm 3:

{<số của mạng>, 1, 1}

để chỉ " các địa chỉ với một số của mạng hợp lệ, một vùng mạng con gồm tất cả các bit 1, và một vùng máy tính gồm tất cả các bit 1".

## 4.3. Một số vấn đề liên quan đến địa chỉ IP

### 4.3.1. Địa chỉ IP và liên kết mạng

Để đơn giản việc tìm hiểu, chúng ta đã nói rằng một địa chỉ IP xác định một máy, nhưng điều này không hoàn toàn chính xác. Xét một bộ định tuyến được nối với hai mạng vật lý. Làm sao có thể gán chỉ một địa chỉ IP duy nhất nếu địa chỉ bao gồm một phần xác định mạng và một phần xác định máy? Thực ra, điều đó là không thể được. Khi những máy có hai hay nhiều liên kết vật lý, thì chúng được gọi là các máy multi homed. Những máy Multi homed và các bộ định tuyến có nhiều địa chỉ IP. Mỗi địa chỉ tương ứng với một liên kết mạng của máy. Các máy Multi homed dẫn chúng ta đến một ý tưởng quan trọng.

Vì các địa chỉ IP bao gồm cả hai phần, phần mạng và phần máy trên mạng đó chúng không xác định một máy đơn. mà xác định một liên kết với một mạng.

Như vậy, một bộ định tuyến liên kết n mạng sẽ có n địa chỉ IP khác nhau, mỗi địa chỉ cho một liên kết mạng.

### 4.3.2. Mạng và địa chỉ quảng bá

Chúng ta đã trình bày những ưu điểm chính của việc đưa thông tin về mạng vào địa chỉ Internet. nó giúp chi việc định tuyến được hiệu quả. Một ưu điểm khác là các địa chỉ Internet có thể dùng để chỉ ra mạng cũng như để chỉ ra các địa chỉ Internet có thể dùng để chỉ ta mạng cũng như để chỉ ra các máy. Theo quy ước, hostid 0 sẽ không bao giờ được gán cho một máy nào. Một địa chỉ IP với phần hostid bằng zero được dùng chỉ bản thân mạng. Tóm lại:

Các địa chỉ IP có thể được dùng để chỉ các mạng cũng như từng máy trong mạng. Theo quy ước, một địa chỉ có tất cả các bít trong phần hostid bằng 0 được dùng để chỉ bản thân mạng.

Một ưu điểm đáng kể khác của mô hình địa chỉ Internet là nó bao gồm một địa chỉ quảng bá để chỉ tất cả các máy trong mạng. Theo chuẩn, một địa chỉ có tất cả các bít trong phần hostid bằng 1 được dành riêng cho việc quảng bá. Khi một gói được gửi tới một địa như thế, chỉ có một phiên bản được chuyển gói trên mạng Internet (từ nơi gửi). Các bộ định tuyến trong Internet sẽ sử dụng phần netid của địa chỉ khi chọn con đường; chúng không xem xét đến phần hostid. Một khi gói dữ liệu đến được bộ định tuyến mà nối với mạng đích đến, bộ định tuyến này sẽ kiểm tra phần hostid của địa chỉ để xác định cách gửi dữ liệu đi. Nếu thấy tất cả các bit là 1, bộ định tuyến sẽ gửi (quảng bá) gói dữ liệu đến tất cả các máy trong mạng.

Trong nhiều kỹ thuật mạng (ví dụ Ethernet), việc quảng bá cũng hiệu quả như việc chuyển đến một máy đơn; cũng có những hệ, việc quảng bá được hỗ trợ bởi phần mềm mạng, nhưng bị chậm hơn nhiều so với gửi đến một máy đơn. Cũng có một vài phần cứng mạng không hỗ trợ việc quảng bá. Như thế, việc có một địa chỉ quảng bá IP không bảo đảm được tính khả thi hoặc hiệu quả của chuyển phát quảng bá. Tóm lại:

Các địa chỉ IP có thể được dùng để xác định quảng bá, mà qua đó dữ liệu được gửi tới tất cả các máy trên một mạng; những địa chỉ đó ánh xạ vào quảng bá phần cứng, nếu hiện hữu. Theo quy ước, một địa chỉ quảng bá sẽ gồm một phần netid và phần hostid với tất vả các bit bằng 1.

### 4.3.3. Quảng bá giới hạn

Địa chỉ quảng bá mà chúng ta vừa mô tả được gọi là trực tiếp (directed) vì nó bao gồm cả phần ID mạng và phần hostid quảng bá. Một địa chỉ quảng bá trực tiếp có thể được hiểu một cách không mơ hồ trên mọi nơi trong Internet và nó xác định một mạng duynhất cùng với việc quảng bá trên mạng đó. Địa chỉ quảng bá trực tiếp cho ta một cơ chế hữu hiệu (và đôi khi hơi nguy hiểm) cho phép một hệ từ xa có thể gửi chỉ một gói dữ liệu nhưng đến được mọi máy của một mạng.

Dưới quan điểm về địa chỉ, bất lợi chính của quảng bá trực tiếp là nó đòi hỏi kiến thức về địa chỉ mạng. Một dạng khác của địa chỉ quảng bá được gọi là địa chỉ quảng bá giới hạn hay địa chỉ quảng bá cục bộ, cung cấp một địa chỉ quảng bá cho mạng cục bộ độc lập với địa chỉ IP. Địa chỉ quảng bá cục bộ bao gồm 32 bit 1 (nên đôi khi được gọi là địa chỉ quảng bá "toàn là 1"). Một máy có thể sử dụng địa chỉ quảng bá cục bộ trong phần khởi động trước khi biết được địa chỉ IP của nó hay phần tiền tố (preix) địa chỉ IP của mạng cục bộ. Tuy nhiên, một khi máy tính biết được chính xác địa chỉ IP của mạng cục bộ, nó sẽ sử dụng quảng bá trực tiếp.

Một nguyên tắc chung, giao thức TCP/IP giới hạn việc quảng bá trong nhóm có ít máy. Chúng đã xem xét quy tắc này ảnh hưởng như thế nào đến nhiều mạng có chung địa chỉ trong các phần trước về địa chỉ mạng con.

### 4.3.4. Quy ước tổng quan về ý nghĩa bít và địa chỉ

Chúng ta đã thấy rằng một vùng bao gồm tất cả các bit là 1 được hiểu là "tất cả", như là "tất cả các máy" trên một mạng.

Nói chung, phần mềm Internet diễn dịch các vùng gồm các bit 0 có nghĩa là "tại đây". Cách diễn dịch này sẽ xuyên suốt toàn bộ giáo trình. Như thế, một địa chỉ IP có phần hostid là 0 có nghĩa là mạng "này".

Dĩ nhiên, diễn dịch này chỉ có nghĩa trong những bối cảnh nào không thể có sự nhầm lần. Ví dụ, nếu một máy nhận một gói mà trong đó phần netid của địa chỉ đích là 0 và phần hostid của địa chỉ đích đúng với địa chỉ của nó, thì máy này sẽ hiểu vùng netid có nghĩa là máy "này" (nghĩa là mạng mà gói này đã đến).

Sử dụng netid bằng 0 đặc biệt quan trọng trong những trường hợp khi mà một máy muốn liên lạc trên mạng nhưng chưa biết địa chỉ IP của mạng. Máy này sẽ tạm thời sử dụng mạng ID là 0, và những máy khác trên mạng sẽ hiểu rằng địa chỉ này có nghĩa là mạng "này".

### 4.3.5. Địa chỉ IP multicast (truyền đồng thời nhiều hướng)

Cùng với việc chuyển phát duy nhất (unicast), trong đó một gói được chuyển phát đến chỉ một máy tính, và chuyển phát quảng bá, trong đó một gói được chuyển phát tới tất cả các máy tính trong một mạng cụ thể, mô hình địa chỉ IP hỗ trợ một dạng đặc biệt của việc chuyển phát nhiều điểm tên là multicasting đặc biệt hữu dụng cho những mạng mà kỹ thuật phần cứng hỗ trợ việc chuyển phát truyền cùng một lúc đi nhiều hướng. Địa chỉ lớp D được dành riêng cho việc truyền cùng một lúc đi nhiều hướng.

### 4.3.6. Nhược điểm của cách đánh địa chỉ IP

Việc kết hợp thông tin mạng vào địa chỉ Internet có một vài bất lợi. Bất lợi hiển nhiên nhất là các địa chỉ tham chiếu đến các liên kết mạng, Chứ không phải đến máy tính:

Nếu một máy tính chuyển từ mạng này sang mạng khác, địa chỉ IP của nó phải thay đổi.

Để hiểu hậu quả của nó, giả sử một người đi du lịch muốn mang máy tính cá nhân không thể được gán một địa chỉ IP vĩnh viễn vì một địa chỉ IP xác định luôn cả mạng mà máy này được nối vào.

Điểm yếu khác của mô hình địa chỉ phân lớp xuất hiện khi một mạng thuộc lớp C phát triển lên với nhiều hơn 255 máy, lúc đó nó phải đổi các địa chỉ của nó thành địa chỉ lớp B. Thoạt nhìn, điều này dường như chỉ là vấn đề nhỏ, việc thay đổi các địa chỉ mạng là một công việc vô cùng mất thời gian và kho truy tìm lỗi. Vì hầu hết phần mềm không được thiết kế để hỗ trợ cho việc một mạng vật lý có nhiều địa chỉ mạng, thay đổi địa chỉ của tất cả các máy, và rồi cho mạng hoạt động lại với địa chỉ mạng mới.

Nhược điểm quan trọng nhất của mô hình địa chỉ Internet vẫn chưa lộ diện cho đến khi chúng ta xem xét việc định tuyến. Vì tầm quan trọng của nó, chúng ta sẽ trình bày thật tóm tắt ngay bây giờ. Chúng ta đã nói rằng việc định tuyến sẽ đưa vào địa chỉ Internet, với phần netid của địa chỉ được dùng để ra các quyết định về việc định tuyến. Hãy thử xét một máy có hai liên kết Internet. Chúng ta biết rằng một máy như vậy phải có nhiều hơn một địa chỉ IP. Do đó ta suy ra một điều.

Vì việc định tuyến sử dụng phần mạng của địa chỉ IP, con đường mà một gói đi qua để đến một máy có nhiều địa chỉ IP tuỳ thuộc vào địa chỉ được sử dụng.

### 4.3.7. Địa chỉ lặp

Tiền tố mạng 127.0.0.0, một giá trị của lớp A, được dành riêng cho địa chỉ lặp (loopback), và thực ra được sử dụng trong việc kiểm tra TCP/IP và cho các tiến trình liên lạc bên trong của một máy tính. Khi một chương trình có địa chỉ đích là địa chỉ lặp, phần mềm. Hơn thế nữa, một máy tính hay bộ định tuyến sẽ không gửi nó qua mạng. Ngoài ra, một máy tính hay bộ định tuyến sẽ không bao giờ nhân bản thông tin về việc định tuyến cho mạng số 127; vì nó không phải là địa chỉ một mang.

4.3.8. Thứ tự các byte trong địa chỉ IP

Để xây dựng một Internet độc lập với kiến trúc máy và phần cứng mạng của một công ty bất kỳ, phần mềm phải định nghĩa cách thể hiện chuẩn cho dữ liệu. Ví dụ, hãy thử xem chuyện gì xảy ra khi phần mềm trên một máy tính gửi dữ liệu 32 bit nhị phân cho máy khác. Thiết bị nhập xuất (phần cứng) chuyển tuần tự từng bit một từ máy xuất phát đến máy thứ hai mà không thấy đổi thứ tự của chúng. Tuy nhiên, không phải mọi kiến trúc (phần cứng) đều có cách lưu trữ số nguyên 32 bit như nhau. Trên một số kiến trúc (endian nhỏ), bộ nhớ địa chỉ thấp nhất chứa byte thứ tự thấp của số nguyên. Trên một số kiến trúc khác (endian lớn), bộ nhớ địa chỉ thấp nhất chứa byte thứ tự cao của số nguyên. Cũng có những hệ khác lưu trữ số nguyên trong những nhóm gồm các từ 16 bit, với địa chỉ thấp nhất lưu trữ từ thứ tự thấp, nhưng các bytes từ máy này sang máy kia có thể làm thay đổi giá trị của số đó.

Việc chuẩn hoá thứ tự byte của số nguyên là đặc biệt quan trọng trong một Internet vì dữ liệu Internet chuyển tải các số nguyên nhị phân để mô tả những thông tin như địa chỉ đích đến và độ dài của gói dữ liệu. Những đại lượng như thế phải được hiểu giống nhau đối với cả hai nơi gửi và nơi nhận. Các giao thức TCP/IP giải quyết vấn đề thứ tự byte bằng việc định nghĩa một thứ tự byte chuẩn mạng mà mỗi máy phải sử dụng cho các vùng số nhị phân trong những gói dữ liệu Internet. Mỗi máy tính hoặc bộ định tuyến sẽ chuyển đổi các thành phần nhị phân từ cách thể hiện địa phương thành thứ tự byte chuẩn mạng trước khi gửi gói dữ liệu đi, và chuyển đổi từ thứ tự chuẩn mạng thành thứ tự đặc thù riêng máy tính khi nhận gói dữ liệu. Hiển nhiên, vùng dữ liệu cho người sử dụng trong gói dữ liệu được miễn trừ đối với chuẩn này vì giao thức TCP/IP không biết được dữ liệu gì đang được chuyển tải trong đó người lập trình ứng dụng được tự do định dạng cách thể hiện và chuyển đổi dữ liệu. Khi gửi các giá trị số nguyên, nhiều người lập trình ứng dụng đã chọn lựa là tuân theo các chuẩn TCP/IP về thứ tự byte. Dĩ nhiên, đối với người sử dụng, chỉ đơn giản là gửi các chương trình ứng dụng mà không cần biết chi tiết về thứ tự byte.

Chuẩn Internet về thứ tự byte mô tả rằng các số nguyên được gửi với byte thứ tự cao trước (theo kiểu endian lớn). Nếu ta xem xét các byte trong một gói dữ liệu khi nó di chuyển từ một máy này sang máy khác, một số nguyên nhị phân trong gói dữ liệu đó sẽ có byte cuối cùng nằm gần ở phần đầu của gói dữ liệu và byte đầu tiên sẽ nắm gần ở phấn cuối của gói dữ liệu. Đã có nhiều lập luận được đưa ra về cách biểu diễn dữ liệu nào nên được sử dụng. Cụ thể, những người ủng hộ việc thay đổi lập luận rằng mặc dù hầu hết các máy tính đã là “endian lớn” khi

chuẩn này được xác lập, nhưng hầu hết những máy tính bây giờ lại là "endian nhỏ". Tuy nhiên, mỗi người đồng ý rằng việc có được một chuẩn là điều cốt yếu, còn dạng chính xác của chuẩn lại ít quan trọng hơn.

### 4.3.9. Quảng bá đến mạng con

Việc quảng bá gặp nhiều khó khăn hơn trong kiến trúc mạng con. Chúng ta nhớ lại trong mô hình địa chỉ IP ban đầu, một địa chỉ với phần máy tính gồm tất cả các bit 1 để chỉ việc quảng bá đến tất cả các máy thường. Như vậy, một địa chỉ {mạng, 1, 1} có nghĩa là "chuyển phát một bản sao đến tất cả các náy mà có mạng là địa chỉ mạng của nó, ngay cả nếu chúng thuộc về các mạng vật lý riêng biệt" Về mặt hoạt động, việc quảng bá đến một địa chỉ như thế chỉ hợp lý nếu các bộ định tuyến nối các mạng con cùng đồng ý để nhân bản quảng bá đến tất cả các mạng vật lý. Dĩ nhiên, phải rất cẩn thận để tránh định tuyến bị lặp. Cụ thể, một bộ định tuyến không thể nào chỉ đơn giản nhân bản một dữ liệu (quảng bá) từ một bộ giao tiếp đến tất cả những bộ giao tiếp khác có cùng tiền tố mạng con. Để ngăn chặn các vòng lặp này, bộ định tuyến sử dụng cách chuyển theo con đường ngược. Bộ định tuyến trích ra nguồn của datagram (được quảng bá), và tìm kiếm nguồn bày trong bảng định tuyến của nó. Sau đó bộ định tuyến huỷ bỏ datagram trừ khi nó đã đến bộ giao tiếp được dùng để chuyển đến nguồn (nghĩa là, đã đến từ con đường ngắn nhất).

Bên trong các mạng có lập mạng con. Thì có thể quảng bá đến một mạng con cụ thể (nghĩa là, quảng bá đến tất cả các máy trên một mạng vật lý mà đã được gán một địa chỉ mạng con). Chuẩn địa chỉ mạng con sử dụng một vùng máy tính với tất cả các bit 1 để chỉ việc quảng bá mạng con. Như vậy, một địa chỉ quảng bá mạng con có dạng:

{ mạng, mạng con, 1}

Xem xét các địa chỉ quảng bá mạng con ta hiểu rõ hơn lời khuyến cáo nên sử dụng một mặt nạ mạng con thống nhất trên bộ các mạng có cùng một địa chỉ IP mạng con. Một khi mạng con và các vùng máy tính giống nhau, thì các địa chỉ quảng bá là rõ ràng không bị mơ hồ. Các phép gán địa chỉ mạng con phức tạp hơn có thể cho phép hoặc không cho phép việc quảng bá đến một nhóm mạng con được chọn của những mạng vật lý có mạng con.

### 4.3.10. Địa chỉ không phân lớp (siêu mạng)

Địa chỉ mạng con được phát minh đầu những năm 1980 để giúp bản vệ không gian địa chỉ Ip; và sau đó là kỹ thuật mạng không đánh số. Đến năm 1993, người ta nhận thấy rằng chỉ những kỹ thuật này thôi thì không đủ ngăn chặn việc cạn kiệt không gian địa chỉ khi Internet phát triển. Những người thiết kế đã làm

việc lại để xác định một phiên bản IP hoàn toàn mới với các địa chỉ lớn hơn. tuy nhiên, để đáp ứng được với sự phát triển cho tới khi phiên bản của IP được chuẩn hoá và lưu hành, người ta đã đưa ra một giải pháp tạm thời.

Giải pháp đó được gọi là địa chỉ không phân lớp (Classless addressing), địa chỉ diêu mạng. hoặc siêu mạng (supernetting). Mô hình này chọn một cách tiếp cận là phần bổ sung của địa chỉ mạng con. Thay vì sử dụng phần đầu Ip mạng đơn cho nhiều mạng vật lý của một tổ chức, siêu mạng cho phép các địa chỉ được gán cho một tổ chức có thể trải ra trên nhiều tiền tố đã được phân lớp.

Để hiểu được tại sao người ta đã đưa ra mô hình địa chỉ không phân lớp, chúng ta cần biết ba sự việc. Trước hết, mô hình phân lớp đã không phân chia các địa chỉ mạng thành những lớp bằng nhau. Mặc dù có ít hơn mười bảy ngàn địa chỉ cho lớp B, nhưng lại có hơn hai triệu địa chỉ cho lớp C. thứ hai, mức độ yêu cầu cấp phát địa chỉ cho lớp C chậm hơn; chỉ một phần nhỏ địa chỉ thuộc lớp C đã được cấp phát. Thứ ba, thống kê cho thấy rằng với mức độ đã cấp phát địa chỉ cho lớp B, thì tiền tố lớp B sẽ mau chóng cạn kiệt. Tình huống này được biết dưới tên cạn kiệt không gian địa chỉ (Running Out of Address Space – ROADS).

Để hiểu được siêu mạng làm việc như thế nào, hãy xét một tổ chức có tầm cỡ (của mạng) trung bình tham gia vào Internet. Một tổ chức như thế sẽ muốn sử dụng một địa chỉ lớp B vì hay lý do: một địa chỉ lớp B không dung nạp được nhiều hơn 254 máy tính và một địa chỉ lớp B có dư số bit để thiết lập mạng con dễ dàng và tiện lợi. Để giữ gìn địa chỉ lớp B, mô hình siêu mạng cấp phát cho một tổ chức một nhóm các địa chỉ lớp C thay vì chỉ một địa chỉ lớp B. Nhóm này phải đủ lớn để gán cho tất cả các mạng mà tổ chức này sẽ nối vào Internet. Ví dụ, giả sử một tổ chức yêu cầu một địa chỉ lớp B và dự định lập mạng con lấp bytes thứ ba làm vùng mạng con. Thay cho một số duy nhất thuộc lớ B, siêu mạng cấp phát cho tổ chức này một nhóm 256 địa chỉ liên tục nhau thuộc lớp C, tổ chức này có thể sử dụng để sau đó gán cho các mạng vật lý.

Mặc dù siêu mạng là dễ hiểu khi được xem như một cách để thoả mãn cho một tổ chức, những người thiết kế có ý định sẽ dùng nó trong một bối cảnh rộng hơn. Họ đã tiên liệu một Internet phân cấp mà trong đó nhà cung cấp dịch vụ Internet thương mại (Internet Service Provides – ISPS) cung cấp mối liên kết với Internet. Để nối các mạng của nó vào Internet, một tổ chức hợp đồng với một ISP; nhà cung cấp dịch vụ sẽ lo liệu các chi tiết về việc gán các địa chỉ IP cho tổ chức này cũng như việc thiết lập các liên kết vật lý. Những người thiết kế siêu mạng đề nghị rằng một nhà cung cấp dịch vụ Internet được cấp phát cho một lượng lớn không gian địa chỉ (nghĩa là, một tập hợp các địa chỉ mà trải ra trên nhiều mạng lớp C). Sau đó, ISP có thể cấp phát một hoặc nhiều địa chỉ từ tập hợp này cho những khách hàng của nó.

### 4.3.11. Ảnh hưởng của siêu mạng đối với việc định tuyến

Việc cấp phát nhiều địa chỉ lớp C thay cho một địa chỉ lớp B giữ gìn được các số thuộc lớp B và giải quyết được vấn đề trước mặt về việc cạn kiệt không gian địa chỉ. Tuy nhiên, nó gây ra vấn đề khác: thông tin mà các bộ định tuyến lưu trữ và trao đổi tăng lên đáng kể. ví dụ, việc gán cho một tổ chức 256 địa chỉ lớp C thay vì chỉ một địa chỉ lớp B, đòi hỏi đến 256 con đường định tuyến thay vì một.

Một kỹ thuật có tên là Đinh tuyến Vùng Trong Không Phân Lớp (Classless Inter Domain Routing – CIDR, tên của nó không được chính xác lắm bởi vì mô hình này xác định địa chỉ cũng như việc định tuyến) giải quyết được vấn đề này. Về ý tưởng, CIDR, tập hợp một nhóm các địa chỉ liên tục nhau thuộc lớp C một dòng (trong bảng) được thể hiện bởi một cặp:

{địa chỉ mạng, bộ đếm}

với địa chỉ mạng là địa chỉ nhỏ nhất trong nhóm này, và bộ đếm xác định số lượng của các địa chỉ mạng trong nhóm. Ví dụ, một cặp:

{192.5.48.0,3}

được dùng để xác định ba địa chỉ mạng 192.5.48.0, 192.5.49.0, 192.5.50.0.

Nếu một vài nhà cung cấp dịch vụ Internet hình thành nên những nhà cung cấp chính của Internet và mỗi ISP chiếm giữ một nhóm lớn các số IP mạng liên tục, thì ta thấy rất rõ lợi ích của siêu mạng: các bảng định tuyến sẽ nhỏ hơn rất nhiều. Hãy xét bảng định tuyến trong bộ định tuyến thuộc về nhà cung cấp dịch vụ P. Bảng này phải có một con đường định tuyến chính xác cho mỗi khách hàng của P, những bảng này không cần lưu trữ con đường định tuyến cho khách hàng của những nhà cung cấp dịch vụ khác, với mỗi dòng xác định nhóm các địa chỉ thuộc về nhà cung cấp dịch vụ.

### 4.3.12. Những nhóm địa chỉ được để dành cho những mạng riêng

IETF đã chỉ định một tập các tiền tố được dành riêng để sử dụng với các mạng riêng. Như là một sự bảo vệ, các tiền tố dành riêng sẽ không bao giờ được gán cho các mạng trong Internet toàn cầu. Các tiền tố dành riêng thường được biết dưới tên các địa chỉ riêng hay các địa chỉ không định tuyến được (nonroutable address). Sở dĩ có tên địa chỉ không định tuyến được là bởi vì các bộ định tuyến trong Internet toàn cầu hiểu rằng các địa chỉ này được dành riêng; nếu một datagram (có địa chỉ là thuộc nhóm những địa chỉ riêng) tình cờ bị chuyển lên Internet toàn cầu, bộ định tuyến trong Internet sẽ có thể nhận biết nó và không chuyển tiếp nó lên mạng Internet toàn cầu.

Các nhóm địa chỉ dành riêng bao gồm:

10.0.0.0 -> 10.255.255.255

172.16.0.0 ->172.31.255.255

192.168.0.0 - >192.168.255.255

169.254.0.0 - >169.254.255.255

Trong những địa chỉ này, địa chỉ cuối cùng 169.254/16 là không bình thường bởi vì nó được sử dụng bởi các hệ thống tự cấu hình địa chỉ IP.

4.3.13. Cơ quan quản lý địa chỉ Internet

Mỗi địa chỉ tiền tố mạng được sử dụng TCP/IP Internet phải là duy nhất. Một tổ chức sử dụng kỹ thuật TCP/IP để xây dựng riêng một mạng Internet (nghĩa là, không được nối vào mạng Internet toàn cầu) có thể gán địa chỉ bất kỳ mà không lưu ý đến những tiền tố của những tổ chức khác. Tuy nhiên, nếu được nối vào mạng toàn cầu, thì nó không được sử dụng tiền tố đã được gán cho tổ chức khác. Để bảo đảm rằng phần mạng của một địa chỉ là duy nhất trong toàn bộ mạng Internet toàn cầu cần có mộ tổ chức lo việc kiểm soát và cấp phát. Cuối năm 1998, một tổ chức mới đã được thành lập để điều hành việc này. Được đặt tên là Internet Copration For Assigned Names and Numbers (ICANN), tổ chức này thiết lập chính sách và gán giá trị cho các tên và những tham số khác được sử dụng trong những giao thức cũng như các địa chỉ.

Trong mô hình phân lớp ban đầu, cơ quan điều hành Internet chọn địa chỉ thích hợp với kích thước của mạng. Một số thuộc lớp C được gán cho mạng có ít máy tính (ít hơn 255); số thuộc lớp B được dành riêng cho những mạng lớn hơn. Và cuối cùng, nếu mạng có thể có nhiều hơn 65535 máy thì sẽ được dành riêng cho mạng lớp A. Không gian địa chỉ bị lệch, mất cân đối vì hầu hết là những mạng nhỏ, một số nhỏ là mạng cỡ trung, và rất ít làm mạng cỡ lớn.

Hầu hết các tổ chức không bao giờ liên lạc trực tiếp với cơ quan quản lý, Để nối mạng của họ vào Internet toàn cầu, một tổ chức thường hợp đồng với đơn vị cung cấp dịch vụ Internet (Internet Service Provider – ISP) địa phương. Cùng với việc cung cấp mối liên kết giữa một tổ chức và phần còn lại của Internet, một ISP cũng nhận được một địa chỉ tiền tố hợp lệ cho mỗi mạng của khách hàng. Thực ra, nhiều ISP địa phương lại là khách hàng của những ISP lớn hơn. Khi khách hàng yêu cầu một địa chỉ tiền tố, ISP địa phương chỉ đơn thuần lấy một tiền tố từ ISP lớn hơn. Như vậy, chỉ những ISP lớn cần liên hệ trực tiếp với ICANN.

Lưu ý rằng cơ quan quản lý trung tâm chỉ cấp pháp phần mạng của một địa chỉ; một khi tổ chức có được tiền tố cho mạng, tổ chức này có thể tuỳ chọn việc

gán duy nhất cho mỗi máy trên mạng của nó. Hơn nữa, cần nhớ rằng cơ quan quản lý trung tâm chỉ có nhiệm vụ cấp phát địa chỉ IP cho những mạng nào nối vào Internet toàn cầu.

## CÂU HỎI VÀ BÀI TẬP

4.1. Một cách chính xác, thì có bao nhiêu mạng thuộc từng lớp A,B và C có thể tồn tại? Và một cách chính xác thì có bao nhiêu máy thuộc mỗi lớp có thể tồn tại trong một mạng?

4.2. Nhược điểm của cách đánh địa chỉ IP?

4.3. Kỹ thuật subnet và cách tính mạng con ở từng lớp mạng cụ thể?

4.4. Các địa chỉ dành riêng cho mạng cục bộ?

4.5. Khái niệm địa chỉ quảng bá và địa chỉ Loopback, địa chỉ multicast?

4.5. Học viện Kỹ thuật Mật Mã bao gồm 5 phòng ban, mỗi phòng ban cần số lượng máy chủ như sau:

- Phòng A: 20 máy
- Phòng B: 15 máy
- Phòng C: 12 máy
- Phòng D: 10 máy
- Phòng E: 3 máy.

Học viện được cấp địa chỉ mạng 203.16.8.0. Hãy phân chia mạng cho học viện?

# CHƯƠNG 5 GIAO THỨC ICMP (INTERNET CONTROL MESSAGE PROTOCOL)

## 5.1. ICMP và thông điệp kiểm soát báo lỗi

### 5.1.1. Giới thiệu về ICMP và thông điệp kiểm soát

Các chương trước đã trình bày cách mà phần mềm IP cung cấp địa chỉ "unreliable", chuyển dữ liệu connectionless bằng cách dàn xếp cho mỗi bộ định tuyến chuyển phát dữ liệu. Mỗi datagram sẽ di chuyển từ bộ định tuyến này đến bộ định tuyến khác cho tới khi đến bộ định tuyến cuối cùng làm nhiệm vụ chuyển datagram trực tiếp đến đích cuối cùng (máy nhận). Nếu một bộ định tuyến không thể gửi đi một datagram, hay nếu nó phát hiện một dấu hiệu không bình thường có ảnh hưởng đến việc chuyển dữ liệu (như nghẽn mạng), bộ định tuyến cần phải thông báo cho nơi xuất phát của datagram, để tránh lỗi hoặc khắc phục lỗi. Chương này sẽ trình bày một cơ chế mà các máy tính và bộ máy định tuyến Internet sử dụng để thông báo cho nhau những thông tin về lỗi và việc kiểm soát. Chúng ta sẽ thấy rằng các bộ định tuyến sử dụng cơ chế này để thông báo các vấn đề và các máy tính truyền hoặc nhận (host) sử dụng chúng để kiểm tra xem dữ liệu liệu thể đến đích hay không.

Trong hệ thống kết nối connectionless mà chúng ta đã mô tả trước đây, mỗi bộ định tuyến hoạt động một cách tự chủ, chuyển tuyến hoặc gửi datagram đi đến nơi mà không cần sự phối hợp với nơi gửi ban đầu.

Hệ này làm việc tốt nếu tất cả các máy hoạt động một cách chính xác và thống nhất với nhau về việc định tuyến. Nhưng không một hệ thông tin liên lạc tầm cỡ lớn nào luôn luôn làm việc một cách chính xác. Bên cạnh sự hỏng hóc của định tuyến và bộ xử lý, còn có trường hợp mạng IP không chuyển phát datagram được khi máy đích bị tách ra khỏi mạng (tạm thời hoặc vĩnh viễn) vì trường "time to live" của datagram trở về zero, hay khi bộ định tuyến trung gian bị nghẽn mạch (quá tải) nên không thể xử lý các datagram đến. Sự khác biệt quan trọng, giữa việc có một mạng đơn được cài đặt với phần cứng nhất định và một Internet được cài đặt với phần mềm, chính là ở chỗ đối với trường hợp đầu, người thiết kế có thể thêm phần cứng đặc biệt để thông báo cho các máy nối mạng khi có lỗi. Trong một Internet, không có cơ chế phần cứng này nơi gửi không thể biết việc chuyển phát không được là do các máy địa phương hay các máy ở xa. Việc bắt lỗi trở thành công việc vô cùng khó khăn. Bản thân giao thức IP (gói tin IP) không chứa thông tin gì để giúp nơi gửi kiểm tra mối liên kết hoặc biết về các lỗi này.

Để cho phép các bộ định tuyến trong một Internet thông báo lỗi hoặc cung cấp thông tin về người tình huống không mong đợi, những nhà thiết kế đã thêm một cơ chế thông báo có mục đích đặc biệt nằm trong bộ giao thức TCP/IP. Cơ chế này (giao thức), được biết dưới tên Internet Control Message Protocol (ICMP), nó là một phần bắt buộc của bộ giao thức TCP/IP và phải có trong mọi cài đặt TCP/IP.

Giống như các giao dịch khác, các thông điệp ICMP di chuyển qua Internet và được đóng gói trong phần dữ liệu của IP Datagram. Tuy nhiên, đích đến cuối cùng của một thông điệp ICMP không phải là một chương trình ứng dụng hay người sử dụng trong máy đích, mà lại là phần mềm Internet protocol trên máy đó. Nghĩa là, khi một thông báo lỗi ICMP đến, module phần mềm ICMP sẽ xử lý nó. Dĩ nhiên, nếu ICMP xác định rằng cụ thể một giao thức cấp cao hơn hay một chương trình ứng dụng đã gây ra lỗi này, nó sẽ thông báo module tương ứng.

Internet Control Message Protocol cho phép bộ định tuyến gửi thông báo lỗi và thông báo điều khiển đến các bộ định tuyến khác hoặc các máy khác; ICMP cung cấp phương tiện thông tin liên lạc giữa phần mềm Internet Protocol trên một máy và phần mềm Internet Protocol trên máy khác.

Ban đầu, ICMP được thiết kế để cho phép các bộ định tuyến thông báo đến các máy tính nguyên nhân của các lỗi phát chuyển, ICMP không chỉ giới hạn cài đặt dành cho các bộ định tuyến. Mặc dù các hướng dẫn giới hạn việc sử dụng của một vài thông điệp ICMP, một máy bất kỳ có thể gửi thông điệp ICMP tới bất kỳ máy khác. Như thế, một máy tính có thể sử dụng ICMP để trao đổi thông báo với bộ định tuyến hoặc máy khác. Ưu điểm chính của việc cho phép máy tính sử dụng ICMP là nó cung cấp chỉ một cơ chế được dùng cho tất cả các thông báo thông tin và điều khiển.

### 5.1.2. Thông báo lỗi và sửa lỗi

Về mặt kỹ thuật, ICMP là một cơ chế thông báo lỗi. Nó cung cấp cho bộ định tuyến một phương pháp để khi gặp lỗi thì sẽ thông báo lỗi cho nguồn đầu tiên. Mặc dù đặc tả giao thức chỉ ra mục đích sử dụng của ICMP và đề nghị các thao tác cần thiết để đáp lại các thông báo lỗi, ICMP không xác định một cách đầy đủ thao tác phải thực hiện cho mỗi lỗi. Tóm lại:

Khi một datagram gây ra một lỗi. ICMP chỉ có thể thông báo điều kiện lỗi trở về nguồn ban đầu của datagram; nguồn này phải liên hệ lỗi này với chương trình ứng dụng, hoặc thực hiện thao tác khác để sửa lỗi, ICMP không có khả năng sửa lỗi.

Hầu hết các lỗi có nguyên nhân từ nguồn ban đầu. Tuy nhiên, bởi vì ICMP thông báo các vấn đề cho nguồn ban đầu, nó không thể thông báo các vấn đề này cho bộ định tuyến trung gian. Ví dụ, giả sử một datagram đi theo con đường qua một dãy các bộ định tuyến, R1, R2,… Rk. Nếu Rk có thông tin định tuyến không chính xác nên chuyển sai datagram đến bộ định tuyến Re, thì Re không thể sử dụng ICMP để thông báo lỗi ngược trở về bộ định tuyến Rk; ICMP chỉ có thể gửi báo cáo trở về nguồn ban đầu. Tiếc thay, nguồn ban đầu không có trách nhiệm cho vấn đề này và cũng không có quyền đối với bộ định tuyến này (Re). thực tế, nguồn ban đầu này không có khả năng xác định được bộ định tuyến nào đã gây ra lỗi.

Tại sao lại giới hạn ICMP chỉ liên lạc với nguồn ban đầu? Dựa vào những tìm hiểu của chúng ta trong các chương trước về định dạng datagram và việc định tuyến, chúng ta có được câu trả lời. Một datagram chỉ gồm những vùng xác định nguồn ban đầu và đích cuối cùng; nó không chứa một hồ sơ đầy đủ về đường đi của nó qua Internet (ngoại trừ hợp đặc biệt khi chọn lựa record ruote được sử dụng). Hơn nữa, bởi vì các bộ định tuyến có thể thiết lập và thay đổi bảng định tuyến của chúng, nên không có định tuyến toàn bộ. Nếu bộ định tuyến nhận ra một vấn đề, nó không thể biết các máy trung gian đã xử lý datagram, nên không thể thông báo chúng về vấn đề này. Thay vì huỷ bỏ datagram một cách im lặng, bộ định tuyến sử dụng ICMP để thông báo nguồn ban đầu rằng vấn đề đã xảy ra, và tin tưởng rằng các quản trị viên máy sẽ hợp tác với quản trị viên mạng để xác định và sửa lỗi.

Như vậy ICMP (Internet Control Message Protocol) là một giao thức điều khiển của mức IP, được dùng để trao đổi các thông tin điều khiển dòng số liệu, thông báo lỗi và các thông tin trạng thái khác của bộ giao thức TCP/IP, một số chức năng tiêu biểu của nó là:

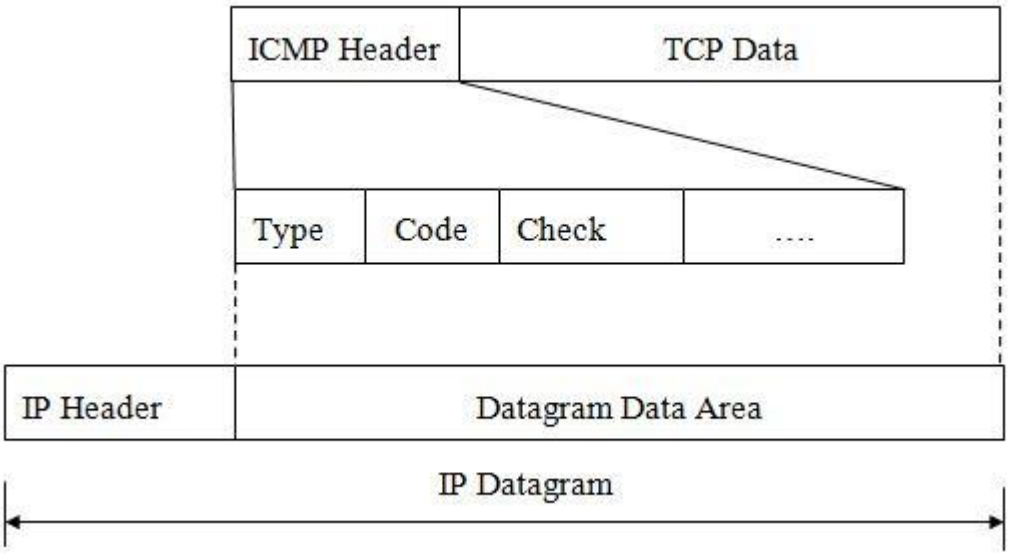
- Điều khiển lưu lượng dữ liệu (Flow control): khi các gói dữ liệu đến quá nhanh, thiết bị đích hoặc thiết bị định tuyến ở giữa sẽ gửi một thông điệp ICMP trở lại thiết bị gửi, yêu cầu thiết bị gửi tạm thời ngừng việc gửi dữ liệu.
- Thông báo lỗi: trong trường hợp địa chỉ đích không tới được thì hệ thống sẽ gửi một thông báo lỗi "Destination Unreachable".
- Định hướng lại các tuyến đường: một thiết bị định tuyến sẽ gửi một thông điệp ICMP "định tuyến lại" (Redirect Router) để thông báo với một trạm là nên dùng thiết bị định tuyến khác để tới thiết bị đích. Thông điệp này có thể chỉ được dùng khi trạm nguồn ở trên cùng một mạng với cả hai thiết bị định tuyến.

- Kiểm tra các trạm ở xa: một trạm có thể gửi một thông điệp ICMP "Echo" để kiểm tra xem một trạm có hoạt động hay không.

## 5.2. Nguyên lý hoạt động của giao thức ICMP

### 5.2.1. Chuyển phát thông điệp ICMP bằng IP Datagram

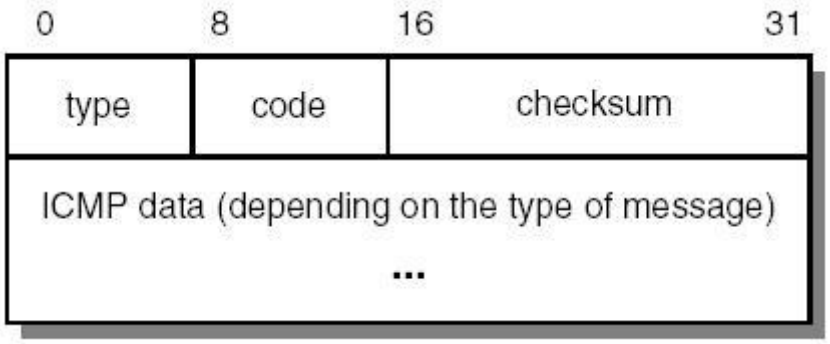
Các thông điệp cần có hai mức đóng gói như trong hình 5.1. Mỗi thông điệp ICMP di chuyển qua Internet trong phần dữ liệu của một IP Datagram, IP Datagram lại di chuyển qua mỗi mạng vật lý trong phần dữ liệu của một frame. Những datagram chuyển tải thông điệp ICMP được chuyển đi y hệt như những datagram chuyển tải dữ liệu bình thường cho người sử dụng; không hề có thêm độ tin cậy hay quyền ưu tiên. Như thế bản thân các thông báo lỗi cũng có thể bị thất lạc hoặc bị huỷ bỏ. Hơn nữa, trong một mạng vốn đã bị giao thông quá tải, thông báo lỗi có thể làm tình hình xấu hơn. Có một ngoại lệ đối với thủ tục xử lý lỗi nếu một một IP Datagram chuyển tải một thông điệp ICMP gây nên lỗi. Ngoại lệ này, được thiết lập để tránh vấn đề tạo ra thông báo lỗi về thông báo lỗi khác, xác định rằng thông điệp ICMP không được tạo ra cho những lỗi gây ra từ những datagram chuyển tải thông báo lỗi ICMP.




*Hình 5.1: Thông điệp ICMP được đóng gói trong IP Datagram*

Chúng ta cần lưu ý một điều quan trọng rằng mặc dù các thông điệp ICMP được gói và gửi đi bằng IP Datagram, ICMP không được xem là một giao thức cấp cao hơn, nó là một phần bắt buộc của IP. Lý do sử dụng IP để chuyển phát thông điệp ICMP là vì chúng có thể phải di chuyển qua một số mạng vật lý để đến đích cuối cùng của nó. Vì thế, chúng không thể được chuyển phát chỉ bằng phương tiện vật lý.

5.2.2. Khuôn dạng thông thông điệp ICMP




*Hình 5.2: Khuông dạng thông điệp ICMP*

Mặc dù mỗi thông điệp ICMP có dạng riêng của nó, chúng đều bắt đầu với ba vùng;

- Một vùng số nguyên 8 bit TYPE xác định kiểu thông điệp ICMP.
- Một vùng 8 bit CODE cung cấp thêm thông tin về kiểu thông điệp.
- Một vùng CHECKSUM 16 bit (ICMP sử dụng cùng thuật giải checksum như IP, nhưng ICMP checksum chỉ tính đến thông điệp ICMP). Hơn nữa, các thông điệp ICMP thông báo lỗi luôn luôn bao gồm phần đầu và 64 bít dữ liệu đầu tiên của datagram gây nên lỗi.

Lý do có thêm phần đầu này cùng với phần đầu datagram là để cho phép nơi nhận xác định chính xác hơn những giao thức nào và chương trình ứng dụng nào có trách nhiệm đối với datagram. Chúng ta sẽ thấy ở các chương sau này về việc các giao thức cấp cao hơn trong bộ TCP/IP được thiết kế sao cho các thông tin cốt yếu được mã hoá trong 64 bit đầu tiên.

Vùng TYPE của ICMP xác định ý nghĩa của thông điệp cũng như định dạng của nó. Các kiểu bao gồm:

| Kiểu | Tên |
|---|---|
| 0 | Echo Reply |
| 1 | Unassigned |
| 2 | Unassigned |
| 3 | Destination Unreachable |
| 4 | Source Quench |
| 5 | Redirect |
| 6 | Alternate Host Address |

| | |
|---|---|
| 7 | Unassigned |
| 8 | Echo Request |
| 9 | Router Advertisement |
| 10 | Router Selection |
| 11 | Time Exeeded |
| 12 | Parameter Problem |
| 13 | Timestamp |
| 14 | Timestamp Reply |
| 15 | Information Request |
| 16 | Information Reply |
| 17 | Address Mask Request |
| 18 | Address Mask Reply |

*Bảng 5.1. ý nghĩa vùng TYPE*

Các phần tiếp theo sẽ mô tả từng loại thông điệp này, trình bày chi tiết về dạng thông điệp và ý nghĩa của chúng

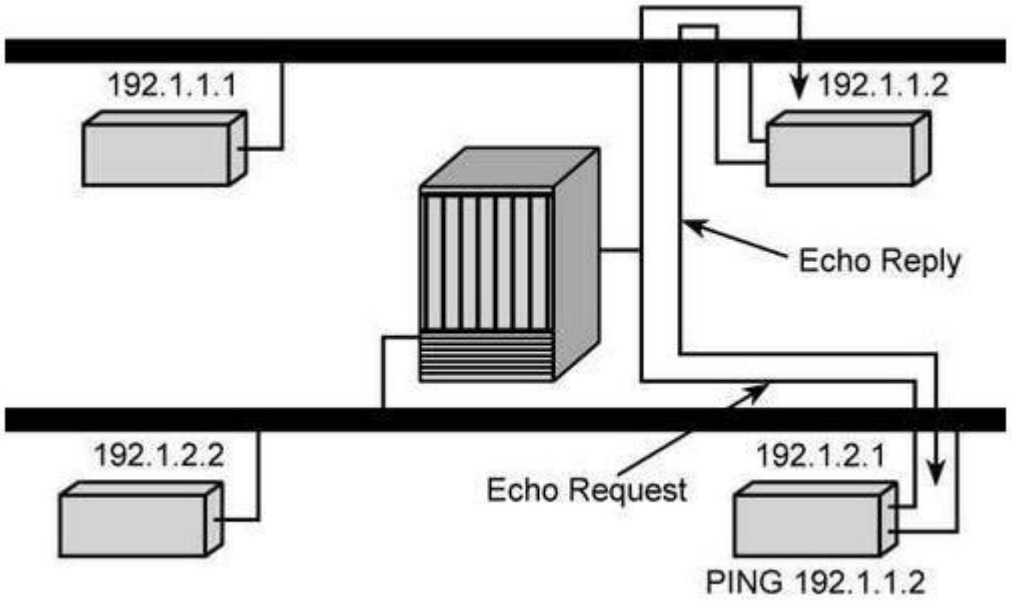
### 5.2.3. Các thông điệp ICMP quan trọng

#### *5.2.3.1. Thông điệp ICMP kiểm tra khả năng đến đích và các trạng thái của đích (Ping ICMP)*

Giao thức TCP/IP cung cấp các phương tiện giúp đỡ người quản lý mạng hay người sử dụng xác định các vấn đề của mạng. Một trong những công cụ tìm lỗi thường được sử dụng nhất liên quan đến các thông điệp "echo request" và "echo reply" của ICMP. Một máy tính hoặc bộ định tuyến gửi một thông điệp ICMP "echo request" tới một đích cụ thể. Máy nào nhận được một "echo request" sẽ tạo ra một"echo reply" và trả nó về nơi gửi ban đầu. Lời yêu cầu (echo request) có bao gồm một vùng dữ liệu, tuy chọn; lời đáp (echo reply) bao gồm một phiên bản của dữ liệu được gửi trong lời yêu cầu. "Echo request" và "echo reply" tương ứng có thể được dùng để kiểm tra xem một máy đích là có thể đến được hay không và có đáp lời không. Vì cả hai, lời yêu cầu và lời đáp, đều di chuyển trong các IP Datagram, việc nhận được đầy đủ lời đáp chứng minh rằng những phần chính của mạng chuyển dữ liệu đang làm việc tốt trên các phương diện: Trước hết, phần mềm IP trên máy nguồn phải chuyển datagram đi. Thứ hai các bộ định tuyến trung gian

giữa nguồn và đích phải đang hoạt động và phải định tuyến datagram một cách chính xác. thứ ba, máy đích phải đang chạy, và cả hai phần mềm IP và ICMP phải đang làm việc. Cuối cùng, tất cả các bộ định tuyến dọc theo con đường trở về phải có thông tin định tuyến chính xác.

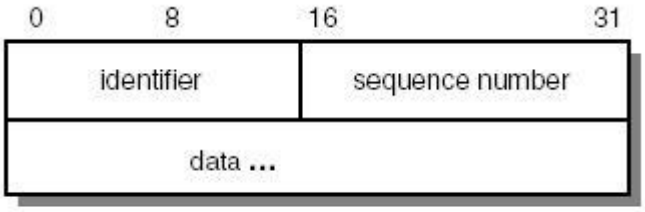
Trên nhiều hệ thống, lệnh thực hiện việc gửi thông điệp ICMP. "echo request" có tên là ping (một nhà khoa học máy tính là Dave Mills đã từng đề nghị rằng PING là chữ viết tắt của Packet Internet Groper). Các phiên bản phức tạp của ping gửi một loạt các "echo request" ghi nhận các "echo reply", và cho ta thống kê về các datagram bị mất, thậm chí cả thời gian đáp ứng. Chúng ta cho phép người sử dụng xác định độ dài của dữ liệu được gửi và khoảng thời gian giữa các lần gửi. Các phiên bản đơn giản hơn chỉ đơn thuần gửi đi một "echo request" và đợi "echo reply".




*Hình 5.3: Hoạt động của lệnh PING*

Khuôn dạng của thông điệp ICMP Echo Request và Echo Reply bao gồm các phần Header chuẩn ban đầu và cộng thêm vùng có tên OPTIONAL DATA là một vùng có độ dài thay đổi để chứa dữ liệu sẽ được trả về cho nơi gửi. Một "echo reply" luôn luôn trả về một cách chính xác cùng một dữ liệu như nó nhận được từ "echo requerst".

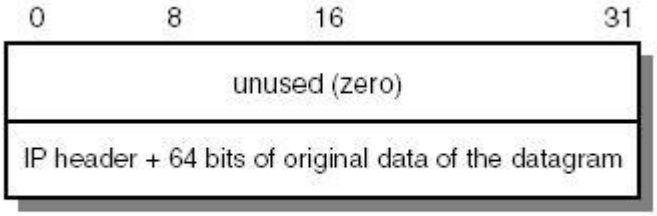
Các vùng IDENTIFIER và SEQUENCE NUMBER được nơi gửi sử dụng để so sánh giữa lời yêu cầu và lời đáp. Giá trị của vùng gửi TYPE để xác định thông điệp là một yêu cầu (8) hay lời đáp (0).




*Hình 5.4: Thông điệp kiểm tra khả năng và trạng thái đến đích*

*5.2.3.2. Thông điệp ICMP báo lỗi các đích không đến được*

Khi một bộ định tuyến không thể truyền hay chuyển phát một IP Datagram, nó gửi một thông báo "*đích không thể đến"* ngược trở về nguồn ban đầu, thông qua định dạng của phần Data Option như hình sau:




*Hình 5.5: Thông điệp ICMP báo lỗi các đính không đến được*

Vùng CODE trong một thông điệp *"đích không thể đến"* chưa một số nguyên để mô tả thêm về vấn đề. Các giá trị đó là:

| | |
|---|---|
| 0 | Network unreachable |
| 1 | Host unreachable |
| 2 | Protocol unreachable |
| 3 | Port unreachable |
| 4 | Fragmentation needed but the Do Not Fragment bit was set |
| 5 | Source route failed |
| 6 | Destination network unknown |
| 7 | Destination host unknown |
| 8 | Source host isolated (obsolete) |
| 9 | Destination network administratively prohibited |
| 10 | Destination host administratively prohibited |
| 11 | Network unreachable for this type of service |

| | |
|---|---|
| 12 | Host unreachable for this type of service |
| 13 | Communication administratively prohibited by filtering |
| 14 | Host precedence violation |
| 15 | Procedence cutoff effect |

*Bảng 5.2. Bảng giá trị mô tả lỗi vùng CODE*

Mặc dù IP là cơ chế chuyển phát nỗ lực tối da (best effort), không được xem nhẹ việc huỷ bỏ datagram. Bất cứ khi nào một lỗi ngăn cản bộ định tuyến làm việc định tuyến hoặc chuyển phát datagram, bộ định tuyến sẽ gửi một thông điệp "đích có thể đến" ngược trở về nguồn và sau đó huỷ bỏ datagram. Các lỗi "máy không thể đến" thường là vì việc phát định tuyến bị hỏng (một ngoại lệ đối với bộ định tuyến sử dụng mô hình địa chỉ mạng con. chúng thông báo lỗi "định tuyến mạng con" bị hỏng với một thông điệp ICMP "máy không thể đến "). Bởi vì các thông báo lỗi ICMP chứa đựng một tiền tố ngắn của datagram đã gây nên vấn đề, máy nguồn sẽư biết chính xác địa chỉ nào là không thể đến.

Các đích có thể là "không đến được" bởi vì phần cứng không hoạt động (ví dụ tạm ngưng để bảo trì), bởi vì máy gửi xác định một địa chỉ đích không tồn tại, hay bởi vì bộ định tuyến không có thông tin về đường đi đến mạng đích (trường hợp này hiếm khi xảy ra). Lưu ý rằng, mặc dù bộ định tuyến báo cáo hỏng hóc chúng gặp phải, chúng có thể không biết về tất cả các vấn đề chuyển phát (bị hỏng). Ví dụ, nếu máy đích nối vào một mạng Ethernet, phần cứng mạng không cung cấp lời đáp. Vì thế, bộ định tuyến có thể tiếp tục giữ dữ liệu đến một đích sau khi đích đó đã tắt mà không nhận được bất kỳ dấu hiệu gì.

Tóm lại:

Mặc dù một bộ định tuyến gửi một thông báo lỗi: "đích không thể đến" khi nó gặp một datagram mà không thể truyền hoặc phát chuyển, bộ định tuyến không thể nhận biết tất cả các lỗi đó.

Ý nghĩa của giao thức và các thông báo "đích không thể đến" sẽ được thấy rõ hơn khi chúng ta tìm hiểu cách mà giao thức cấp cao hơn sử dụng các điểm đích trừu tượng gọi là cổng (port). Hầu hết các thông điệp còn lại tự nói lên ý nghĩa của nó. Nếu datagram bao gồm chọn lựa "source route" với một thông tin định tuyến không chính xác, nó có thể tạo nên một thông điệp  route bị lỗi. Nếu bộ định tuyến cần phân đoạn một datagram nhưng bit "đừng phân đoạn" được lập, bộ định tuyến sẽ gửi một thông báo lỗi "cần phân đoạn" trở về nguồn.

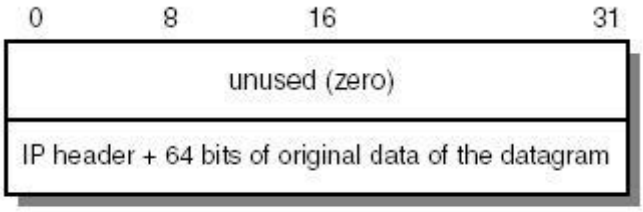
*5.2.3.3. Thông điệp ICMP làm nguội nguồn phát (Source Quench) khi có sự cố nghẽn mạng*

Bởi vì IP là connectionless, một bộ định tuyến không thể dành riêng sẵn bộ nhớ hay tài nguyên thông tin liên lạc cho việc nhận các datagram. Kết quả là, bộ định tuyến có thể bị quá tải, một trạng thái gọi là nghẽn mạch. Cần phải hiểu một điều quan trọng rằng việc nghẽn mạch có thể xuất hiện bởi hai lý do hoàn toàn khác nhau. Trước hết, các máy tính tốc độ cao có thể tạo ra các giao dịch nhanh hơn tốc độ một mạng chuyển dữ liệu đi. Ví dụ, thử hình dung một siêu máy tính gửi dữ liệu lên Internet. Các datagram có thể phải đi qua một mạng diện rộng (WAN) tốc độ thấp mặc dù siêu máy tính được nối với một mạng cục bộ (LAN) tốc độ cao. Việc nghẽn mạch sẽ xảy ra tại bộ định tuyến nối WAN với LAN bởi vì các datagram đến nhanh hơn lúc chuyển chúng đi. Thứ hai, nếu nhiều máy tính đồng thời cần gửi datagram qua một bộ định tuyến, sự nghẽn mạch có thể xảy ra.

Khi datagram đến quá nhanh mà máy tính hoặc bộ định tuyến không xử lý kịp, chúng sẽ được sắp vào hàng đợi (bộ nhớ tạm thời). Nếu các datagram này là một phần một đoạn dữ liệu nhỏ, vùng đệm này có thể giải quyết được vấn đề. Nhưng nếu dữ liệu vẫn được gửi liên tục, sẽ làm cạn kiệt vùn nhớ đệm, máy tính hoặc bộ định tuyến sẽ phải huỷ bỏ những datagram đến sau. Một máy sử dụng thông điệp ICMP “source quench” để thông báo sự nghẽn mạch cho nguồn ban đầu. Một thông điệp “source quench” là một yêu cầu đối với nguồn ban đầu để giảm bớt cường độ truyền datagram. Thông thường, khi bị nghẽn mạch, bộ định tuyến sẽ gửi một thông điệp “source quench” cho mỗi datagram bị chúng huỷ. Bộ định tuyến cũng có thể sử dụng các kỹ thuật kiểm soát nghẽn mạch phức tạp hơn. Một số kỹ thuật kiểm soát dữ liệu đến và “làm nguội” những nguồn nào có cường độ truyền datagram cao nhất. Cũng có những kỹ thuật khác cố gắng tránh tình trạng nghẽn mạch bằng cách gửi các yêu cầu “làm nguội: khi hàng đợi của nó bắt đầu dài ra, trước khi nó thật sự đầy.

Không có thông điệp ICMP để đảo ngược tác dụng của một “source quench”. Thay vì vậy, khi một máy nhận được thông điệp “source quench” cho máy đích D, nó sẽ giảm dần cường độ gửi datagram đến máy D cho tới khi nó hết nhận được thông điệp “source quench”, sau đó nó sẽ lại tăng dần cường độ gửi miễn sao không còn nhận được các yêu cầu “source quench”.

Ngoài các vùng thông thường như TYPE, CODE, CHECKSUM, và một vùng 32 bit không sử dụng, các thông điệp “source quench” còn có một vùng để chứa tiền tố cuả datagram. Định dạng phần Data Option của thông điệp ICMP “source quench” được trình bày trong hình sau




*Hình 5.6: Thông điệp khi có sự cố nghẽn mạng*

*5.2.3.4. Thông điệp ICMP yêu cầu thay đổi đường đi từ bộ định tuyến*

Các bảng định tuyến Internet thường ổn định trong một thời gian dài. Các máy tính (router) khởi động chúng từ một tập tin cấu hình khi khởi động hệ thống, và người quản trị hệ thống hiếm khi thực hiện các thay đổi về việc định tuyến trong suốt quá trình hoạt động. Nếu cấu hình mạng thay đổi, các bảng dịnh tuyến trong máy tính và bộ định tuyến có thể không còn chính xác. Một thay đổi về các tuyến đường có thể là tạm thời (ví dụ khi sửa chữa phần cứng) hoặc vĩnh viễn (ví dụ khi có một mạng mới thêm vào Internet). Chúng ta sẽ thấy trong các chương sau, các bộ định tuyến trao đổi thông tin định tuyến theo định kỳ để cập nhật những thay đổi trên mạng và giữ cho thông tin định tuyến của chúng được cập nhật liên tục, nguyên tắc tổng quát là:

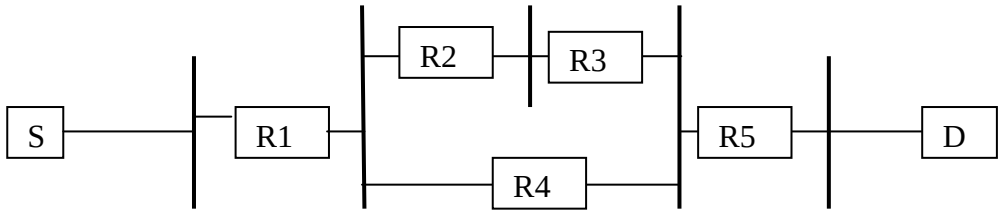
Các bộ định tuyến được giả định là biết chính xác các kênh; các máy tính bắt đầu với thông tin định tuyến tối thiểu và biết các kênh mới từ các bộ định tuyến.

Để tuân theo đúng quy tắt này và để tránh sự trùng lặp thông tin định tuyến trong các tập tin cấu hình trên mỗi máy, cấu hình định tuyến ban đầu của máy tính xác định thông tin tối thiểu có thể dùng để liên lạc (ví dụ địa chỉ của một bộ định tuyến mặc định). Vì thế, các máy tính bắt đầu với thông tin tối thiểu và đưa vào các bộ định tuyến để cập nhật bằng định tuyến của nó. Trong một trường hợp đặc biệt, khi bộ định tuyến nhận thấy một máy tính đang sử dụng con đường không tối ưu, nó gửi cho máy tính thay đổi đi đến đích cuối cùng của nó.

Ưu điểm của mô hình đổi hướng ICMP là tính đơn giản của nó: nó cho phép máy tính khi khởi động chỉ cần biết địa chỉ của một bộ định tuyến trên mạng cục bộ. Bộ định tuyến sẽ gửi về các thông điệp đổi hướng ICMP bất cứ khi nào máy tính gửi một datagram đi theo một con đường nhưng lại tồn tại một con đường tốt hơn. bảng định tuyến của máy tính được giữ có kích thước nhỏ nhưng vẫn chưa đựng các con đường tối ưu cho tất cả các đích đang sử dụng.

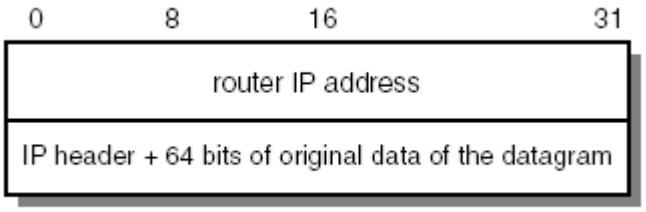
Tuy nhiên, các thông điệp đổi hướng không giải quyết vấn đề nhân bản các kênh bởi vì chúng bị giới hạn trong các tương tác giữa bộ định tuyến và máy tính

trên một mạng mà chúng được nối trực tiếp vào; như được minh hoạ trong hình 5.4 Trong hình này, giả sử nguồn S gửi một datagram tới đích D. Giả sử rằng bộ định tuyến R1 chuyển datagram theo kênh đi qua bộ định tuyến R2 thay vì qua bộ định tuyến R4 (nghĩa là, R1 đã chọn sai một đường dài hơn). Khi bộ định tuyến R5 nhận datagram, nó không thể gửi một thông điệp ICMP đổi hướng tới R1 được bởi vì nó không biết địa chỉ của R1. Trong các chương sau chúng ta sẽ tìm hiểu bài toán; làm thế nào để nhân bản thông tin định tuyến qua nhiều mạng.




*Hình 5.7: Định tuyến bằng tuyến đường tốt hơn*

Cùng với các vùng bắt buộc TYPE, CODE, và CHECKSUM. Mỗi thông điệp đổi hướng còn có một vùng 32 bit ROUTER Internet ADDRESS và một vùng Internet HEADER, phần Data Option của nó như trong hình sau.




*Hình 5.8: Thông điệp yêu cầu thay đổi đường đi từ bộ định tuyến*

Vùng ROUTER IP ADDRESS chứa địa chỉ của một bộ định tuyến mà thấy máy tính sẽ dùng để đến được máy đích đã được ghi trong phần đầu datagram. Vùng IP HEADER chứa phần đầu IP và 64 bit đầu tiên của datagram mà tạo ra thông điệp. Như thế, khi nhận một thông điệp ICMP đổi hướng, máy tính sẽ kiểm tra tiền tố của datagram để xác địa chỉ đích của datagram. Vùng CODE của một thông điệp ICMP đổi hướng sẽ xác định thêm cách diễn dịch địa chỉ đích, dựa vào các giá trị được gán như sau:

| Mã | Mô tả |
|---|---|
| 0 | Redirect cho mạng |
| 1 | Redirect cho host |
| 2 | Redirect cho loại dịch vụ (TOS) và |

| | |
|---|---|
| | mạng |
| 3 | Redirect cho loại dịch vụ và host |

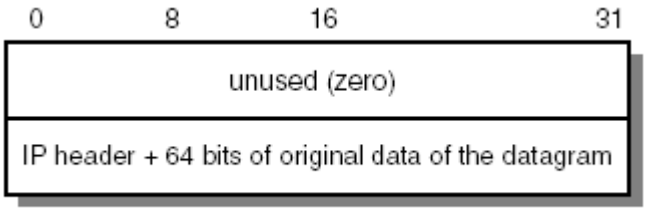
*Bảng 5.3: Mô tả vùng Mã của thông điệp ICMP đổi hướng*

Một quy tắt tổng quát là, bộ định tuyến chỉ gửi yêu cầu ICMP đổi hướng đến máy tính chứ không gửi đến bộ định tuyến. Chúng ta sẽ thấy trong các chương sau rằng bộ định tuyến sử dụng các giao thức khác để trao đổi thông tin định tuyến.

*5.2.3.5. Thông điệp ICMP nhận biết vòng kín hoặc định tuyến quá dài*

Bởi vì bộ định tuyến Internet tính trạm tiếp thông qua các bảng cục bộ, các lỗi trong bảng định tuyến có thể tạo nên một vòng kín định tuyến cho một đích D nào đó. Một vòng kín định tuyến có thể bao gồm hai bộ định tuyến, mỗi cái chuyển datagram cho đích D đến cái kia, hoặc có thể bao gồm nhiều bộ định tuyến. Khi các bộ định tuyến hình thành nên vòng kín, mỗi cái chuyển datagram cho đích D đến bộ định tuyến kế tiếp trong vòn kín. Nếu một datagram rơi vào một vòng kín định tuyến, nó sẽ di chuyển quanh vòng mãi mãi. Tuy nhiên, chúng ta đã biết trước đây, để ngăn chặn tình huống khi các datagram di chuyển mãi trên TCP/IP Internet, trong mỗi IP Datagram có một bộ đếm thời gian sống, đôi khi còn gọi là đếm số trạm. Mỗi bộ định tuyến sẽ giảm bớt một giá trị của thời gian sống bất cứ khi nào nó xử lý datagram và huỷ bỏ datagram đi khi giá trị này zero.

Bất cứ khi nào bộ định tuyến huỷ bỏ một datagram vì bộ đếm thời gian sống của nó đã về zero hoặc bởi vì đã hết thời gian đợi các fragment của một datagram, nó sẽ gửi một thông điệp ICMP "quá thời hạn" (time exceeded) ngược trở về nguồn của datagram, sử dụng định danh như phần Data Option trong hình sau:




*Hình 5.9: Thông điệp nhập biết vòng kín hoặc định tuyến quá dài*

ICMP sử dụng vùng CODE trong mỗi thông điệp "quá thời hạn" (có giá trị 0 hoặc 1) để giải thích lý do huỷ bỏ datagram.

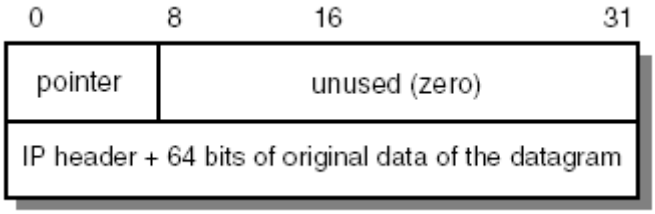
| *Mã* | *Ý nghĩa* |
|---|---|
| 0 | Bộ đếm thời gian sống bằng zero |
| 1 | Quá thời hạn đợi kết hợp các fragment |

*Hình 5.4: Giá trị vùng mã trong thông điệp quá thời hạn*

Kết hợp fragment để chỉ công việc tập hợp tất cả các fragment của một datagram hoàn chỉnh được phân mảnh khi truyền. Khi fragment đầu tiên của một datagram đến máy tính sẽ khởi động một bộ đếm thời gian và xem như là có lỗi nếu đã quá thời hạn mà chưa nhận đủ mọi fragment của một datagram, giá trị 1 của trường CODE được dùng để báo cáo lỗi này cho nguồn gửi.

*5.2.3.6. Thông điệp ICMP báo lỗi có vấn đề tham số của Datagram*

Khi bộ định tuyến hoặc nhận thấy có vấn đề với datagram, nhưng khác với những loại lỗi đã trình bày (ví dụ, phần đầu datagram không chính xác), nó gửi một thông báo "vấn đề tham số" về nguồn ban đầu. Một nguyên nhân có thể có cho những vấn đề đó, xảy ra khi các tham số của một chọn lựa không chính xác. Thông báo này, phần Data Option được định dạng như trong hình sau, chỉ được gửi khi vấn đề quá nghiêm trọng đến nỗi phải huỷ bỏ datagram.



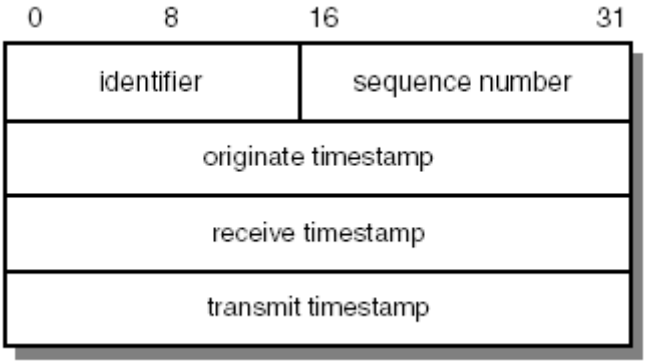

*Hình 5.10: Thông điệp báo lỗi khi có vấn đề tham số*

Để cho thông điệp không bị mơ hồ khó hiểu, nơi gửi sử dụng vùng POINTER trong phần đầu thông điệp để xác định byte trong datagram đã gây ra lỗi. Giá trị trị 1 được dùng để cho biết rằng còn thiếu một chọn lựa.

***5.2.3.7. Thông điệp ICMP đồng bộ đồng hồ và ước lượng thời gian truyền***

Mặc dù máy trên một Internet có thể thông tin liên lạc, chúng thường hoạt động một cách độc lập, và mỗi máy duy trì về thời gian hiện hành của riêng nó. Nếu các đồng hồ quá khác nhau có thể làm bối rối người sử dụng những hệ phần mềm được phân bố. Bộ giao thức TCP/IP bao gồm một số giao thức được sử dụng để đồng bộ các đồng hồ. Một trong những kỹ thuật đơn giản nhất sử dụng một thông điệp ICMP để lấy thời gian từ một máy khác. Một máy có yêu cầu, sẽ gửi một thông điệp ICMP yêu cầu ghi nhận thời gian (timestamp) đến một máy khác, để yêu cầu máy thứ hai này. Để đáp lại, máy thứ hai này sẽ gửi một thông điệp

ICMP lời đáp ghi nhận thời gian cho máy thứ nhất. Hình sau trình bày định dạng phần Data Option của các thông điệp yêu cầu và lời đáp ghi nhận thời gian.




*Hình 5.11: Thông điệp đồng bộ và ước lượng thời gian truyền*

Vùng TYPE để xác định thông điệp là yêu cầu (13) hay lời đáp (14); các vùng IDENTIFIER và SEQUENCE NUMBER được dùng bởi máy nguồn để phối hợp lời đáp với yêu cầu. Các vùng còn lại xác định thời gian, ở dạng millisecond kể từ nửa đêm, giờ quốc tế GMT. Vùng ORIGINATE TIESTAMP được điền vào bởi máy gửi ban đầu, ngày trước khi dữ liệu được truyền đi; vùng RECEIVE TIMESTAMP được điền vào tức khắc ngay khi nhận được yêu cầu, và vùng TRANSMIT TIMESTAMP được điền vào tức khắc trước khi lời đáp được chuyển đi.

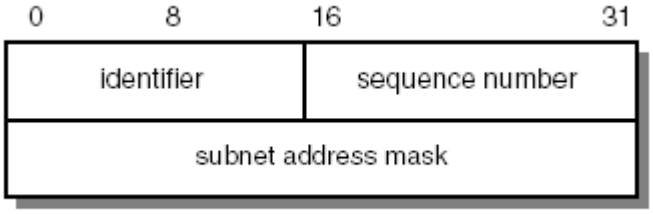
Các máy tính sử dụng ba vùng dấu thời gian để tính ước lượng thời gian trì hoãn giữa chúng và để đồng bộ các đồng hồ của chúng. Bởi vì trong lời đáp cũng bao gồm vùng ORIGINATE TIMESTAMP, một máy có thể tính tổng thời gian cần cho mọt yêu cầu di chuyển đến một đích, được chuyển dạng thành lới đáp, và trở về. Bởi vì lời đáp mang theo cả thời điểm mà lời yêu cầu đến được máy thứ hai, cũng như thời điểm mà đáp được chuyển đi, máy tính có thể tính thời gian truyền trên mạng, và từ đó ước lượng sự khác biệt giữa hai đồng hồ.

Trong thực tế, ước lượng chính xác thời gian trì hoãn hai chiều có thể là công việc khó khăn. Dĩ nhiên, để có được một ước lượng chính xác thời gian trì hoãn hai chiều, phải tính nhiều thứ và lấy trung bình. Tuy nhiên, thời gian trì hoãn hai chiều giữa một cặp máy tính nối vào một Internet lớn có thể thay đổi rất lớn, ngay cả trong một khoảng thời gian ngắn. Hơn nữa chúng ta nhớ rằng bởi vì IP là kỹ thuật nỗ lực tối đa, các datagram có thể bị mất, bị hoãn, hoặc chuyển phát không theo thứ tự.

*5.2.3.8. Thông điệp ICMP tìm mặt nạ mạng con*

Trong chương trước chúng ta sẽ tìm hiểu động lực cho địa chỉ mạng con cũng như các chi tiết về cách hoạt động của các mạng con. Tạm thời lúc này, điều quan trọng chỉ là hiểu được khi nào thì náy tính sử dụng địa chỉ mạng con, một số bit trong phần hostid của địa chỉ IP của chúng xác định một mạng vật lý. Để tham gia vào địa chỉ mạng con, một máy tính cần phải biết những bit nào của địa chỉ mạng Internet 32 bit tương ứng với mạng vật lý và những bit nào tương ứng với định danh máy. Thông tin cần để diễn dịch địa chỉ được thể hiện trong một đại lượng 32 bit, được gọi là mặt nạ mạng con (subnet mask).

Để biết subnet mask được sử dụng cho mạng cục bộ, một máy có thể gửi một thông điệp yêu cầu địa chỉ mặt nạ đến một bộ định tuyến và nhận một lời đáp địa chỉ mặt nạ. Máy tính thực hiện yêu cầu có thể gửi thông điệp một cách trực tiếp, nếu nói biết địa chỉ của bộ định tuyến, hoặc quảng bá thông điệp nếu không biết. Hình sau trình bày định dạng phần Data Option của thông điệp tìm địa chỉ mặt nạ.




*Hình 5.12: Thông điệp tìm mặt nạ mạng con*

Vùng TYPE trong một thông điệp địa chỉ mặt nạ để xác định thông điệp là yêu cầu (17) hay lời đáp (18). Một lời đáp chứa đựng địa chỉ mặt nạ mạng con của mạng trong vùng ADDRESS MASK. Kết quả là, các vùng INDENTIFIER Và SEQUENCE NUMBER cho phép một máy phối hợp lời đáp với yêu cầu.

*5.2.3.9. Thông điệp ICMP tìm ra bộ định tuyến*

Sau khi máy tính khởi động, nó phải biết địa chỉ của ít nhất một bộ định tuyến trên mạng cục bộ trước khi có thể gửi datagram tới các đích trên mạng khác. ICMP cung cấp một mô hình tìm ra bộ định tuyến mà cho phép một máy tìm ra một địa chỉ bộ định tuyến

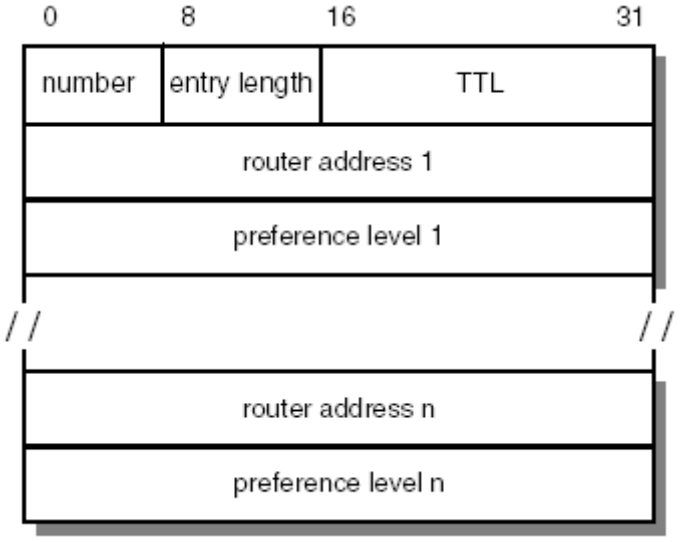
Tìm ra bộ định tuyến ICMP không phỉa là cơ chế duy nhất mà máy tính có thể sử dụng để tìm một địa chỉ bộ định tuyến. Các giao thức BOOTP và DHCP – mỗi giao thức này cung cấp một cách để một máy lấy được địa chỉ của bộ định tuyến mặc cùng với các thông tin bootstrap khác. Tuy nhiên, BOOT và DHCP có

một nhược điểm nghiêm trọng: thông tin mà chúng trả về đến từ một cơ sở dữ liệu mà người quản trị mạng thiết lập một cách thủ công. Vì thế, thông tin không thể thay đổi nhanh chóng.

Dĩ nhiên, việc cấu hình bộ định tuyến làm việc tốt trong một số tình huống. Ví dụ, xét một mạng mà chỉ có mọt bộ định tuyến nối nó vào Internet. Sẽ không cần thiết để các máy tính trong mạng đó đi tìm bộ định tuyến hoặc thay đổi con đường một cách tự động. Tuy nhiên, nếu một mạng có nhiều bộ định tuyến nối nó vào Internet, một máy tính lấy thông tin định tuyến mặc định (ví dụ từ bọ định tuyến mặc định) khi khởi động, có thể sẽ bị mất liên lạc nếu bộ định tuyến đó bọ hỏng. Quan trọng hơn nữa, máy tính không thể nhận ra việc hỏng hóc này.

Mô hình ICMP tìm ra bộ định tuyến giúp đó theo hai cách. Thay vì cung cấp địa chỉ bộ định tuyến được cấu hình thông qua giao thức bootstrap, mô hìnhnày cho phép máy tính lấy thông tin trực tiếp từ chính bộ định tuyến. Thứ hai, cơ chế này sử dụng kỹ thuật trạng thái mềm với bộ đếm thời gian để tránh tình trạng máy tính giữ lại một con đường sau khi bộ định tuyến bị hỏng – bộ định tuyến quảng bá thông tin của nó một cách định kỳ, và máy tính huỷ bỏ một con đường nếu bộ đếm thời gian cho đường đó đã hết hạn.

Hình sau trình bày định dạng phần Data Option của thông điệp quảng bá mà bộ định tuyến gửi đi.




*Hình 5.13: Thông điệp tìm ra bộ định tuyến*

Bên cạnh các vùng TYPE, COĐE, và CHECKSUM, thông điệp còn bao gồm một vùng có tên NUMBER để xác định số lượng các địa chỉ để sử dụng (thường là 1), một vùng ENTRY LENGTH để xác định kích thước của một vùng TTL để xác định thời gian tính bằng giây mà một máy tính có thể sử dụng các địa chỉ được quảng bá. giá trị mặc định của TTL là 30 phút, và giá trị mặc định cho

mỗi đình kỳ truyền lại là 10 phút, có nghĩa là máy tính sẽ không huỷ bỏ một con đường nếu máy tính mất một thông điệp quảng bá đơn.

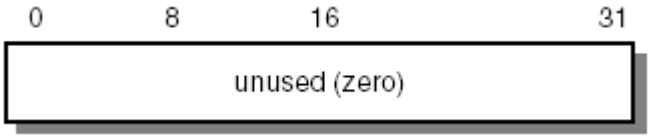
Phần còn lại của thông điệp bao gồm các cặp thông tin, với mỗi cặp cho một con đường, bao gồm ROUTER ADDRESS và một số nguyên PRECEDENCE LEVEL. Giá trị ưu tiên là một số nguyên phần bù của 2; máy tính chọn con đường có độ ưu tiên cao nhất.

Nếu bộ định tuyến và mạng hỗ trợ việc truyền cùng một lúc đi nhiều hướng (multicasting, bộ định tuyến sẽ cùng một lúc truyền đi nhiều hướng các thông điệp ICMP quản bá bộ định tuyến đến tất cả các hệ thống địa chỉ truyền nhiều hướng (nghĩa là 224.0.0.1). nếu không, bộ định tuyến gửi các thông điệp đến địa chỉ quảng bá giới hạn (nghĩa là tất cả địa chỉ 1). Dĩ nhiên, máy tính không bao giờ được gửi một thông điệp quảng bá bộ định tuyến.

*5.2.3.10. Thông điệp ICMP khẩn khoản bộ định tuyến*

Mặc dù những người thiết kế đã cung cấp một khoảng các giá trị được sử dụng làm thời gian chờ giữa hai lần quảng bán kế tiếp, họ đã chọn giá trị mặc định là 10 phút. Giá trị được chọn như một sự thoả hiệp giữa việc nhận biết lỗi nhanh và tổng chi phí thấp.một giá trị nhỏ hơn sẽ cho phép nhận biết nhanh hơn về việc bộ định tuyến hỏng, nhưng là gia tăng lượng giao thông trên mạng; một giá trị lớn hơn sẽ giảm bớt giao thông, nhưng làm chậm việc nhận biết hỏn hóc. Một trong những vấn đề những người thiết kế đã xem xét là làm cách nào để hoà hợp một số lượng nhiều bộ định tuyến trên cùng một mạng.

Từ quan điểm của một máy tính. Thời gian chờ mặc định có nhược điểm lớn: một máy tính không thể chờ nhiều phút cho một quảng bá khi nó khởi động lần đầu. Để tránh các trì hoãn này,những nhà thiết kế đã thêm một ICMP *thông điệp khẩn khoản bộ định tuyến*, cho phép máy tính yêu cầu một sự quảng bá tức thì từ phí router cung cấp thông tin cho nó. Hình sau trình bày định dạng phần Data Option của thông điệp này




*Hình 5.14: Thông điệp khẩn khoản bộ định tuyến*

Nếu một máy tính hỗ trợ việc truyền cùng một lúc đi nhiều hướng, máy tính này sẽ gửi lời khẩn khoản tới tất cả cac địa chỉ multicast(nghĩa là 224.0.0.2), nếu không thì máy tính gửi lời khẩn khoản tới địa chỉ quảng bá giới hạn (nghĩa là tất cả địa chỉ 1). Khi nhận được một thông điệp khẩn khoản bộ định tuyến sẽ gửi một

quảng bá bộ định tuyến. Như hình vẽ khuôn dạng chỉ ra lời khẩn khoản không cần chuyển tải nhiều thông tin hơn là TYPE,CODE,và CHECKSUM.

**Câu hỏi và bài tập**

5.1. Hãy thực nghiệm gửi thật nhiều các packet đi qua bộ định tuyế để kích hoạt động một thông điệp làm nguội nguồn ICMP.

5.2. Nguyên lý hoạt động của ICMP.

5.3. Khuôn dạng của các ICMP quan trọng

5.4. Giả sử rằng tất cả các bộ định tuyến đều gửi đi thông điệp ICMP tìm exceeded, và phần mềm TCP/IP cục bộ của ta sẽ trả về các thông điệp đó cho chương trình ứng dụng. Hãy thử sử dụng khả năng này để xây dựng một lệnh traceroute để thông báo danh sách các bộ định tuyến giữa nguồn và một đỉnh cụ thể.

5.5. Ping tới máy 203.162.0.11? (một máy chủ rất quan trọng tại VN).

5.6. Trình bày nguyên lý hoạt động và cú pháp của các lệnh PING, TRACERT (TraceRoute), hãy liệt kê đường đi từ máy tính của bạn tới máy chủ 203.162.131.1

# CHƯƠNG 6 GIAO THỨC UDP (USER DATAGRAM PROTOCOL)

## 6.1. Giới thiệu giao thức UDP

### 6.1.1. Giới thiệu

Các chương trước đã mô tả khả năng của Internet trong việc gửi datagram giữa các máy tính, trong đó mỗi datagram được gửi qua Internet dựa vào địa chỉ IP của đích đến. Tại lớp Internet Protocol, một địa chỉ đích xác định một máy tính; không cần một phân biệt nào nữa cho dù người sử dụng nào hoặc chương trình ứng dụng nào sẽ nhận datagram này. Chương này sẽ mở rộng bộ Giao thức TCP/IP bằng việc thêm vào một cơ chế để phân biệt các đích đến trong một máy tính (host), cho phép nhiều chương trình ứng dụng trên một máy có thể cùng một lúc gửi và nhận datagram một cách độc lập.

Giao thức UDP (User Datagram Protocol) là giao thức ở tầng vận chuyển, không kết nối (connectionless). Do đó UDP là giao thức vận chuyển không tin cậy: UDP không có cơ chế kiểm tra số tuần tự phát, số tuần tự thu và kiểm tra lỗi. Ưu điểm của UDP là thực hiện đơn giản. Một phần dịch vụ thư điện tử và dịch vụ tên miền sử dụng giao thức UDP trong việc trao đổi dữ liệu của mình. Mặc dù không tin cậy nhưng theo thống kê vận hành khai thác mạng, 99% các gói dữ liệu UDP vẫn được chuyển đúng.

### 6.1.2. Cơ chế xác định đích đến cuối cùng trong chuyển phát

Các hệ điều hành trong hầu hết các máy tính đều hỗ trợ việc lập trình song song, nghĩa là chúng cho phép nhiều chương trình ứng dụng được thực hiện cùng một lúc. Sử dụng thuật ngữ hệ điều hành muốn ám chỉ một phiên bản đang chạy của một chương trình là một tiến trình (process), công việc (task), chương trình ứng dụng (application program), hay một xử lý ở mức người sử dụng (user level process); các hệ thống này được gọi là hệ thống đa nhiệm (multitasking). Vì thế một tiến trình là đích cuối cùng của một thông điệp. Tuy nhiên, việc xác định rằng một tiến trình cụ thể trên một máy cụ thể là đích cuối cùng cho một datagram vẫn có những điểm sai lệch vì:

- Trước hết, bởi vì các tiến trình được tạo ra và huỷ đi một cách tự động, máy gửi hiếm khi biết đủ thông tin để xác định một tiến trình trên máy khác.
- Thứ hai, chúng ta mong muốn có thể thay đổi tiến trình nhận datagram mà không phải thông báo tất cả các máy gửi (ví dụ, khởi động lại một máy có

thể thay đổi tất cả các tiến trình, nhưng các máy gửi không được yêu cầu biết về các tiến trình mới).

- Thứ ba, chúng ta cần xác định các đích đến từ các chức năng chúng cài đặt mà không biết chương trình mà cài đặt chức năng này (ví dụ, cho phép máy gửi liên lạc với máy chủ mà không biết tiến trình nào trên máy đích cài đặt chức năng máy chủ). Quan trọng hơn nữa, trong những hệ cho phép một tiến trình xử lý hai hay nhiều chức năng, một điều cốt lõi là chúng ta phải bố trí một cách để cho một tiến trình quyết định một cách chính xác chức năng nào máy gửi cần có.

Thay vì suy nghĩ về một tiến trình như là đích cuối cùng, chúng ta sẽ hình dung rằng mỗi máy bao gồm một tập hợp các điểm đích trừu tượng gọi là cổng ứng dụng (port). Mỗi cổng ứng dụng được xác định bằng một số nguyên dương. Thực chất khái niệm cổng ứng dụng là của hệ điều hành máy tính, hệ điều hành sử dụng các vùng nhớ đệm có đánh số thứ tự và cung cấp một cơ chế giao tiếp mà các tiến trình dùng để truy xuất dữ liệu trong quá trình làm việc.

Hầu hết các hệ điều hành đều cung cấp sự truy xuất đồng bộ đến các cổng. Từ quan điểm của một tiến trình cụ thể, sự truy xuất đồng bộ có nghĩa là sự tính toán sẽ dừng lại trong quá trình một cổng truy xuất các hoạt động. Lấy ví dụ, nếu một tiến trình muốn trích dữ liệu ra từ một cổng trước khi một dữ liệu bất kỳ nào đến, hệ điều hành sẽ tạm thời ngưng (khoá lại) tiến trình cho đến khi dữ liệu đến. Một khi dữ liệu đến rồi, hệ điều hành sẽ chuyển dữ liệu đến tiến trình và khởi động nó trở lại. Một cách tổng quát, cổng là vùng đệm, sao cho nếu dữ liệu đến trước khi một tiến trình sẵn sàng nhận nó, dữ liệu sẽ không bị mất. Để thực hiện điều này trên các vùng nhớ đệm (ram, cache), phần mềm giao thức bên trong hệ điều hành sẽ đặt các gói dữ liệu gửi đến cho một giao thức cụ thể vào một hàng đợi (hữu hạn) cho đến khi tiến trình trích nó ra.

Để liên lạc với cổng bên ngoài, máy gửi cần phải biết cả hai thông số, đó là địa chỉ IP của máy đích và giá trị cổng ứng dụng của tiến trình đích đến trong máy đó. Mỗi thông điệp phải mang theo giá trị của cổng đích trên máy mà thông điệp được gửi đến, cũng như là giá trị của cổng nguồn trên máy nguồn, nơi mà lời đáp sẽ được gửi đến. Điều này cho phép tiến trình bất kỳ khi nhận thông điệp có thể đáp lại chính xác cho tiến trình nơi gửi.

### 6.1.3. Chức năng của giao thức User Datagram Protocol

Trong bộ giao thức TCP/IP, giao thức *User Datagram Protocol* (UDP) cung cấp cơ chế chính yếu mà các chương trình ứng dụng sử dụng để gửi datagram đến các chương trình ứng dụng khác. UDP cung cấp các cổng ứng dụng được dùng để phân biệt các chương trình đang thực hiện trên một máy đơn. Nghĩa là, cùng với

dữ liệu gửi đi, mỗi thông điệp UDP còn bao gồm một giá trị cổng đích và giá trị cổng nguồn, giúp cho phần mềm UDP tại đích có thể chuyển phát thông điệp tới đúng nơi nhận và cho phép nơi nhận gửi lại lời hồi đáp cũng chính xác tới cổng nguồn của máy gửi.

UDP dựa trên cơ sở Internet Protocol để chuyển cho thông điệp từ một máy đến một máy khác, nên cũng cung cấp dịch vụ chuyển phát datagram không định hướng, không có độ tin cậy như IP. Nó không sử dụng cơ chế acknowledgement (phản hồi xác nhận) để bảo đảm rằng thông điệp đi đến đích, nó không sắp thứ tự các thông điệp gửi đến, và nó không cung cấp thông tin phản hồi để kiểm soát mức độ truyền thông tin giữa các máy. Như thế, các thông điệp UDP có thể bị mất, bị trùng lặp. Hoặc đến đích không theo đúng thứ tự. Hơn thế nữa, các gói dữ liệu có thể đến nhanh hơn khả năng xử lý của máy nhận. Chúng ta có thể tóm tắt như sau:

User Datagram Protocol (UDP) cũng cung cấp dịch vụ chuyển phát datagram không định hướng, không có độ tin cậy, sử dụng IP để chuyển thông điệp giữa các máy. Nó sử dụng IP để chuyển tải thông điệp, nhưng thêm vào khả năng phân biệt nhiều đích đến bên trong một máy tính.

Một chương trình ứng dụng mà sử dụng UDP chấp nhận hoàn toàn trách nhiệm cho việc xử lý các vấn đề về độ tin cậy, bao gồm việc các thông điệp bị mất, bị trùng lặp, bị trì hoãn, đến đích không theo đúng thứ tự, và bị mất liên lạc. Tiếc thay, người lập trình ứng dụng thường bỏ qua các vấn đề này khi thiết kế phần mềm. Hơn nữa, bởi vì người lập trình thường kiểm tra phần mềm mạng qua việc sử dụng các mạng cục bộ có độ tin cậy cao, độ trì hoãn ít, và các dữ liệu kiểm tra có thể không khai thác các lỗi tiềm ẩn. Vì thế, nhiều chương trình ứng dụng mà dựa vào UDP hoạt động tốt trong môi trường mạng LAN nhưng lại có hiệu năng và độ tin cậy thấp khi được khai thác trong môi trường mạng WAN TCP/IP lớn.

## 6.2. Nguyên lý hoạt động của UDP

### 6.2.1. Định dạng thông điệp UDP

Mỗi thông điệp UDP được gọi là user datagram. Về mặt khái niệm, một user datagram bao gồm hai phần: một phần đầu UDP và một vùng dữ liệu UDP. Như trình bày trong hình 6.1, phần đầu được chia thành bốn vùng 16 bit để xác định cổng ứng dụng mà thông điệp được gửi đi từ đó, cổng ứng dụng của máy đích mà thông điệp được dự kiến sẽ gửi đến, độ dài thông điệp, và các mã kiểm tra UDP checksum.

| Source Port | Destination Port |
|---|---|
| Length | Checksum |
| Data…… | |

*Hình 6.1: Cấu trúc thông điệp UDP*

Các vùng SOURCE PORT và DESTINATION PORT chứa các giá trị 16 bit cho cổng ứng dụng UDP được dùng để (phân kênh) các datagram trong các tiến trình đang đợi để nhận chúng. SOURCE PORT là vùng tuỳ chọn. Khi được dùng, nó xác định cổng mà lời đáp sẽ được gửi đến; nếu không được dùng, nó sẽ có giá trị zero.

Vùng LENGTH chứa độ dài của UDP datagram tính theo Byte, bao gồm cả phần đầu UDP và dữ liệu của người sử dụng. Như thế giá trị tối thiểu của LENGTH là 8, chính là độ dài của phần đầu UDP.

UDP checksum là vùng tuỳ chọn và chẳng cần dùng đến; một giá trị zero trong vùng CHECKSUM có nghĩa là checksum chưa được tính. Lý do người thiết kế để vùng CHECKSUM là tuỳ chọn để cho phép việc cài đặt được thực hiện với ít bước tính toán khi phải sử dụng UDP trên một mạng cục bộ có độ tin cậy cao. Tuy nhiên, chúng ta nhớ lại rằng IP không có tính checksum cho phần dữ liệu của một IP Datagram. Như thế, UDP checksum cung cấp cách duy nhất để bảo đảm rằng dữ liệu nhận được nguyên vẹn và nên được sử dụng.

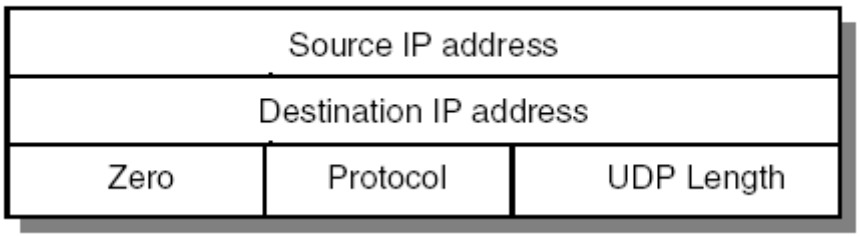
Có điều gì sẽ xảy ra cho các thông điệp UDP khi mà giá trị vùng *CHECKSUM* là zero. Giá trị tính cho checksum hoàn toàn có thể là zero bởi vì UDP sử dụng cùng một thuật giải checksum như IP: nó chia dữ liệu thành các đại lượng 16 bit và tính phần bù của một của tổng các phần bù của một của chúng. Tuy nhiên, zero không phải là vấn đề bởi vì phép tính số học phần bù của một sẽ có hai thể hiện cho zero: tất cả các bit được lập là 1 hoặc tất cả các bit được lập là zero. Khi checksum tính được là zero, UDP sử dụng cách thể hiện có tất cả các bít là 1.

### *Phần đầu giả UDP*

Checksum UDP bao hàm nhiều thông tin hơn là chỉ có mặt trong UDP datagram. Để tính checksum, UDP gán một phần đầu giả vào UDP datagram, thêm vào một byte các giá trị zero vào datagram để có được đúng bội số của 16 bit, và tính checksum cho toàn bộ. Byte đó được dùng để nối vào phần đầu giả sẽ không được truyền đi với UDP datagram và chúng cũng không được kể trong phần độ dài. Để tính checksum, trước tiên phần mềm ghi giá trị zero vào vùng *CHECKSUM*, rồi tích luỹ tổng 16 bit phần bù của toàn bộ đối tượng, bao gồm phần đầu giả, phần đầu UDP, và dữ liệu của người sử dụng.

Mục đích của việc sử dụng một phần đầu giả là để kiểm chứng rằng UDP datagram đã đến được đích chính xác của nó. Điểm mấu chốt để hiểu phần đầu giả

chính là việc nhận thức rằng đích chính xác bao gồm một máy cụ thể và một cổng ứng dụng xác định trong máy đó. Bản thân phần đầu UDP chỉ xác định giá trị cổng ứng dụng. Như thế, để kiểm chứng đích đến, UDP trên máy gửi sẽ tính một checksum mà bao gồm cả địa chỉ IP đích cũng như là UDP datagram. Tại đích đến cuối cùng, phần mềm UDP kiểm chứng checksum bằng cách sử dụng địa chỉ IP đích có được từ phần đầu của IP Datagram mà đã chuyển tải thông điệp UDP. Nếu checksum trùng khớp, có nghĩa là datagram đã đến được đích cuối cùng của nó và cũng đến được đúng cổng ứng dụng trong máy đó.

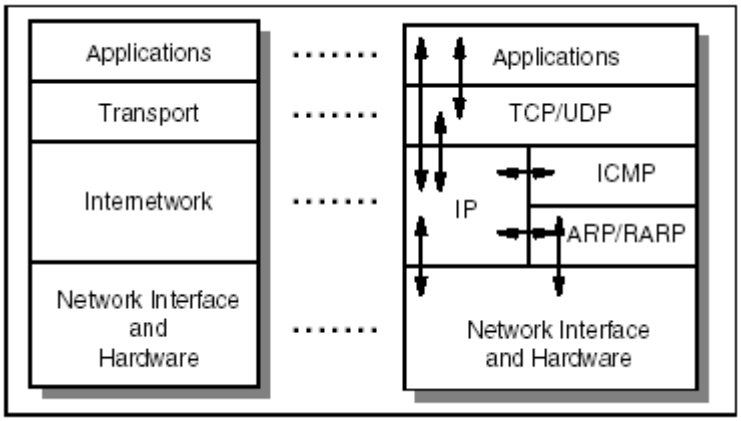



*Hình 6.2: Phần đầu giả của thông điệp UDP*

Phần đầu giả được dùng trong việc tính UDP checksum bao gồm 12 byte dữ liệu được bố trí như trong hình 6.2. Các vùng của phần đầu giả được đánh nhãn SOURCE IP ADDRESS và DESTINATION IP ADDRESS chứa địa chỉ IP nguồn và địa chỉ IP đích mà sẽ được dùng khi gửi thông điệp UDP. Vùng PROTOCOL chứa mã kiểu của giao thức IP (17 cho UDP), và vùng UDP LENGTH chứa độ dài của UDP datagram (không bao gồm phần đầu giả). Để kiểm chứng checksum, nơi nhận phải trích ra các vùng này từ phần đầu IP, lắp ráp chúng thành hình dạng phần đầu giả, và tính lại checksum.

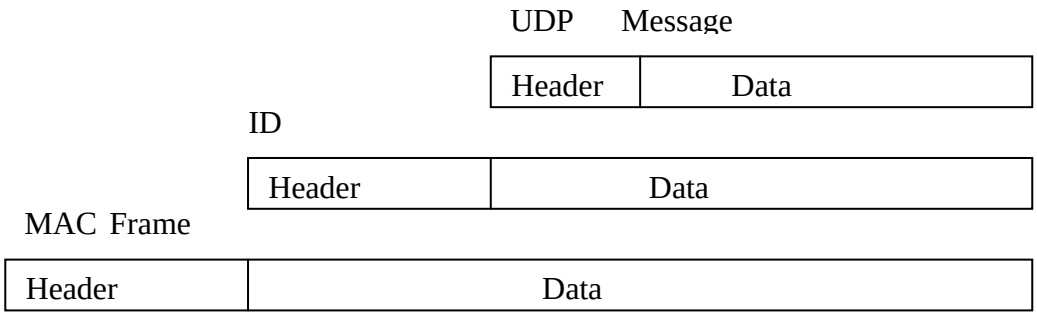
### 6.2.2. Đóng gói UDP và việc phân lớp Protocol

UDP cung cấp cho ta ví dụ đầu tiên về giao thức vận chuyển dữ liệu. Trong mô hình phân lớp dưới, UDP thuộc về lớp ở trên lớp Internet Protocol. Về mặt khái niệm, các chương trình ứng dụng sẽ truy xuất UDP, và UDP lại sử dụng IP để gửi và nhận datagram.




*Hình 6.3:Vị trí của UDP trong giao thức TCP*

Việc phân lớp UDP ở trên IP có nghĩa là một thông điệp UDP hoàn chỉnh, bao gồm phần đầu UDP và dữ liệu, dược đóng gói trong một IP Datagram khi nó di chuyển qua Internet như trong hình:




*Hình 6.4:Đóng gói thông điệp UDP*

Đối với những giao thức chúng ta đã xem xét, việc đóng gói có ý nghĩa là UDP thêm một phần đầu vào phần dữ liệu mà người sử dụng gửi và chuyển chúng đến IP. Lớp IP lại thêm một phần đầu vào gói dữ liệu có nhận được từ UDP. Cuối cùng, lớp giao tiếp mạng nhúng datagram vào một frame trước khi gửi nó từ máy này đến máy khác. Định dạng của frame tuỳ thuộc vào kỹ thuật của mạng cơ sở. Thông thường, frame mạng bao gồm thêm một phần đầu.

Về phía nhận dữ liệu, gói dữ liệu đến tại lớp thấp nhất của phần mềm mạng và bắt đầu con đường đi lên tuần tự qua các lớp cao hơn. Mỗi lớp sẽ loại bỏ một phần đầu trước khi chuyển thông điệp đến lớp trên, sao cho đến khi lớp cao nhất chuyển dữ liệu đến tiến trình nhận, tất cả phần đầu đã được loại bỏ. Như thế, phần đầu nằm ngoài cùng nhất tương ứng với lớp thấp nhất của giao thức, trong khi phần đầu nằm trong cùng nhất tương ứng với lớp cao nhất của giao thức. Khi xem xét cách mà các phần đầu đưa thêm vào và loại ra, điều quan trọng cần lưu ý là nguyên lý phân lớp. Cụ thể, chúng ta nhận thấy rằng nguyên lý phân lớp áp dụng cho UDP, vì vậy UDP datagram nhận được từ IP trên máy đích giống chính xác với datagram mà UDP đã chuyển đến IP trên máy nguồn. Tương tự dữ liệu mà UDP chuyển phát

tới một tiến trình của người sử dụng trên máy nhận cũng sẽ giống chính xác dữ liệu mà một tiến trình của người sử dụng đã chuyển đến UDP trên máy ghi.

Việc phân chia nhiệm vụ trong số các lớp giao thức là nghiêm ngặt và rõ ràng:

Lớp IP chỉ có trách nhiệm cho việc truyền dữ liệu giữa một cấp máy trên Internet, trong khi lớp UDP chỉ có trách nhiệm trong việc phân biệt giữa các nguồn hay các đích bên trong một máy.

Như thế, chỉ có phần đầu IP xác định các máy nguồn và máy đích; chỉ có lớp UDP xác định các cổng nguồn hay cổng đích bên trong một máy.

### 6.2.3. Sự phân lớp và tính UDP checksum

Nếu ta quan sát kỹ thì sẽ nhận thấy rằng dường như có một sự mâu thuẫn giữa quy tắc phân lớp và việc tính UDP checksum. Chúng ta cũng nhớ lại rằng UDP checksum bao gồm một phần đầu giả, trong đó có các vùng địa chỉ IP nguồn và địa chỉ IP đích. Có thể lập luận rằng người sử dụng phải biết địa chỉ IP đích khi gửi đi một UDP datagram, và người sử dụng phải chuyển nó đến lớp UDP. Như thế, lớp UDP có thể lấy được địa chỉ IP đích mà không cần liên lạc với lớp IP. Tuy nhiên, địa chỉ IP nguồn tuỳ thuộc vào định tuyến IP đã chọn cho datagram, bởi vì địa chỉ nguồn IP xác định giao tiếp mạng mà datagram được truyền ttrên đó. Như thế, UDP không thể biết địa chỉ IP nguồn trừ khi nó giao tiếp với lớp IP.

Chúng ta giả định rằng phần mềm UDP yêu cầu lớp IP tính ra địa chỉ IP nguồn và (có thể) địa chỉ IP đích, sử dụng chúng để xây dựng phần đầu giả, tính checksum, loại bỏ phần đầu giả, và rồi chuyển UDP datagram đến IP để truyền đi. Cũng có một cách tiếp cận khác, tạo ra một cách bố trí có hiệu quả lớn hơn để cho lớp UDP đóng gói UDP datagram vào trong IP Datagram, lấy được địa chỉ nguồn từ IP, lưu trữ địa chỉ nguồn và địa chỉ đích vào các vùng thích hợp trong phần đầu datagram, tính UDP checksum, và rồi chuyển IP Datagram đến lớp IP, và nó chỉ cần điền vào các vùng còn lại trong phần đầu IP.

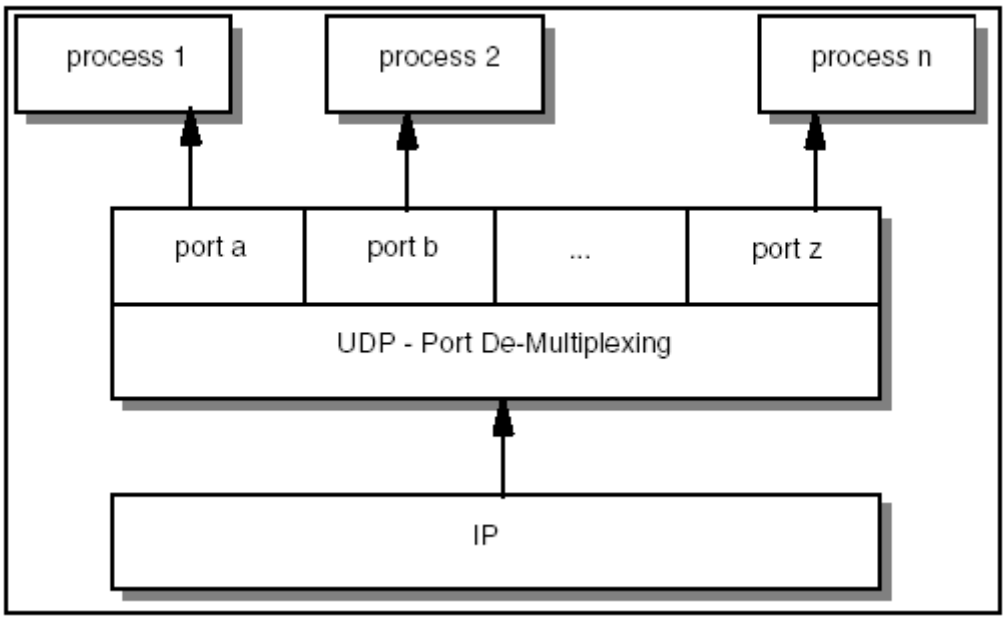
Liệu rằng qua nhiều tương tác giữa UDP và IP có vi phạm tiền đề cơ bản của nguyên lý phân cấp: sự phân lớp thể hiện sự tách biệt của chức năng? Câu trả lời là, đúng vậy. UDP đã được phối hợp chặt chẽ với Giao thức IP. Đây rõ rằng là dung hoà với việc tách rời thuần tuý, và được thực hiện hoàn toàn vì lý do thực tế. Chúng ta sẵn lòng bỏ qua việc vi phạm nguyên lý phân lớp này bởi vì chúng ta không thể nào chỉ ra một cách đầy đủ chương trình ứng dụng đích mà không xác định máy đích, và chúng ta muốn thực hiện một cách hiệu quả phép ánh xạ giữa các địa chỉ được sử dụng bởi UDP và các địa chỉ được sử dụng bởi IP. Một trong những bài tập để kiểm tra vấn đề này theo một quan điểm khác, yêu cầu chúng ta xét xem liệu UDP có nên được tách rời khỏi IP hay không.

6.2.4. UDP Multiplexing, Demultiplexing, và các cổng

Chúng ta đã biết rằng phần mềm đi qua các lớp của một giao thức phân cấp phải multiplex hay demultiplex trong số các đối tượng từ lớp kế tiếp. Phần mềm UDP cung cấp một ví dụ khác về việc multiplexing và demultiplexing. Nó chấp nhận UDP datagram từ nhiều chương trình ứng dụng và chuyển chúng đến IP để truyền đi, và nó chấp nhận các UDP datagram đến từ IP và chuyển mọi datagram đến chương trình ứng dụng thích hợp.

Về mặt khái niệm, tất cả việc multiplexing và demultiplexing giữa phần mềm UDP và chương trình ứng dụng xảy ra thông qua cơ chế cổng. Trong thực tế, mọi chương trình ứng dụng phải thương thảo với hệ điều hành để có được một cổng ứng dụng và một giá trị cổng tương ứng trước khi nó có thể gửi UDP datagram đi. Một khi cổng đã được gắn, bất kỳ datagram nào mà chương trình ứng dụng gửi qua cổng sẽ lưu giá trị cổng đó vào vùng *UDP SOURCE PORT* của nó.

Trong khi xử lý dữ liệu nhập vào, UDP chấp nhận các datagram đến từ phần mềm IP và demultiplex đưa vào cổng đích UDP, như trình bày như hình sau:




*Hình 6.5:Cổng TCP*

Một cách dễ hiểu nhất là hình dung cổng UDP như một hàng đợi. Trong hầu hết các cài đặt, khi chương trình ứng dụng thương thảo với hệ điều hành để sử dụng một cổng nào đó, hệ điều hành sẽ tạo ra một hàng đợi bên trong để có thể giữ lại các thông điệp gửi đến. Thông thường, chương trình ứng dụng có thể xác định hoặc thay đổi kích thước của hàng đợi. Khi UDP nhận datagram, nó kiểm tra xem giá trị cổng đích có phù hợp với một trong những cung hiện đang sử dụng không. Nếu không, nó gửi một thông điệp lỗi ICMP port unreachable và huỷ bỏ datagram. Nếu có cổng phù hợp, UDP đặt datagram mới vào hàng đợi (cổng) nơi mà chương

trình ứng dụng có thể truy xuất nó. Dĩ nhiên, lỗi có thể xảy ra khi cổng bị đầy, UDP sẽ huỷ bỏ những datagram gửi đến.

### 6.2.5. Các giá trị cổng hợp lệ và dành riêng

Các giá trị cổng ứng dụng sẽ được gắn như thế nào? Đây là vấn đề quan trọng bởi vì hai máy tính cần phải thống nhất với nhau về các giá trị tổng trước khi chúng có thể làm việc với nhau. Lấy ví dụ, khi máy tính A muốn lấy một tập tin từ máy B, nó cần biết cổng mà chương trình truyền tập tin trên máy B sử dụng. Có hai cách tiếp cận cơ bản cho việc gắn cổng. Cách tiếp cận đầu tiên sử dụng đơn vị điều hành trung tâm. Mọi người đều đồng ý cho phép đơn vị điều hành trung tâm gán các giá cổng khi cần và phổ biến danh sách của tất cả các giá trị gán. Sau đó tất cả phần mềm được xây dựng theo danh sách này. Đôi khi, cách tiếp cận này được gọi là phép gán toàn cầu, và việc gán các cổng được xác định bởi đơn vị điều hành được gọi là phép gán cổng well know n (rõ ràng).

Cách tiếp cận thứ hai cho việc gán cổng sử dụng liên kết động. Trong cách tiếp cận liên kết động, các cổng không được biết một cách toàn cục. Thay vì vậy, bất cứ khi nào một chương trình cần một cổng, phần mềm mạng sẽ gán cho một cổng. Để biết về việc gán cổng hiện tại trên máy tính khác, thì phải gửi một yêu cầu để hỏi máy tính đó về việc gán cổng hiện tại (ví dụ, dịch vụ truyền tập tin đang sử dụng cổng gì). Máy đích sẽ đáp lại bằng cách cho biết chính xác giá trị cổng.

Người thiết kế TCP/IP đã đưa ra một cách tiếp cận pha trộn. Đó là gán cho một vài giá trị cổng giá trị cho trước, và để nhiều giá trị khác có thể sử dụng cho đơn vị địa phương hoặc chương trình ứng dụng. Các giá trị cổng được gán bắt đầu từ giá trị thấp rồi mở rộng lên, dành ra các giá trị số nguyên lớn cho việc cấp phát động.

**Câu hỏi và bài tập**

6.1 Tại sao UDP checksum tách biệt với IP checksum?

6.2 Liệu rằng ý niệm về nhiều đích đến được xác định bởi các cổng ứng dụng có nên được xây dựng trong IP? tại sao? Tại sao không?

6.3 Đâu là ưu điểm chính của việc sử dụng các giá trị cổng UDP được gán trước? và đâu là khuyết điểm?

6.4 Đâu là ưu điểm của việc sử dụng các cổng ứng dụng thay vì xử lý các định danh để xác định đích ở trong một máy?

6.5 Gửi UDP datagram qua mạng diện rộng bằng một chương trình tiện ích nào đó và đo xác xuất bị mất. Kết quả này có phụ thuộc vào thời điểm? lượng giao dịch trên mạng?

6.6. Cấu trúc gói tin UDP và ý nghĩa cụ thể của các trường?

# CHƯƠNG 7 GIAO THỨC TCP

## 7.1. Dịch vụ vận chuyển dữ liệu có độ tin cậy

### 7.1.1. Giới thiệu dịch vụ vận chuyển có độ tin cậy

Các chương trước đã trình bày dịch vụ chuyển phát dữ liệu theo kiểu connectionless, không đáng tin cậy đã hình thành nên cơ sở cho mỗi thông tin liên lạc trên Internet và giao thức IP xác định nó. Chương này trình bày dịch vụ quan trọng thứ hai nổi tiếng ở mức mạng, dịch vụ chuyển phát có độ tin cậy, và Giao thức điều khiển việc truyền dữ liệu Trasmission Control Protocol (TCP) để làm việc đó. Chúng ta sẽ thấy rằng TCP thêm vào tính năng đáng kể cho giao thức mà chúng ta đang tìm hiểu, nhưng việc cài đặt nó cũng rất phức tạp.

Mặc dù TCP được trình bày ở đây như một phần của bộ giao thức TCP/IP, thực ra nó là một giao thức độc lập, một giao thức tổng quát mà có thể được điều chỉnh để sử dụng với các hệ phát triển khác. Lấy ví dụ, bởi vì TCP có rất ít giả định về mạng cơ sở, nên có thể sử dụng nó trên một mạng đơn giản như Ethernet, cũng như trên một Internet phức tạp. Thực ra, TCP hết sức phổ biến đến mức một trong những giao thức hệ thống mở của tổ chức chuẩn hoá quốc tế đã được tạo ra từ đó.

### 7.1.2. Sự cần thiết của việc chuyển phát dữ liệu theo dòng

Tại mức thấp nhất, các mạng truyền thông máy tính cung cấp dịch vụ chuyển phát không tin cậy. Các gói dữ liệu có thể bị mất hay bị hỏng khi các lỗi đường truyền tác động lên dữ liệu, hoặc khi phần cứng mạng bị hỏng, hay khi mạng bị quá tải bởi vì lượng giao dịch vượt quá khả năng của mạng. Những mạng nào chuyển gói dữ liệu tự động cũng có thể phát triển chúng không theo đúng thứ tự, chuyển phát chúng với đọ trì hoãn lớn, hay chuyển phát bị trùng lặp. Hơn thế nữa, kỹ thuật mạng cơ sở có thể đưa ra một kích thươc gói dữ liệu tối ưu hay áp đạt những ràng buộc khác cần có để đạt được mức độ truyền hiệu quả.

Tại mức cao nhất, các chương trình ứng dụng thường cần gửi một khối lượng lớn dữ liệu từ máy này đến máy khác. Sẽ trở nên rất phiền phức và mệt mỏi khi sử dụng hệ chuyển phát connectionless không tin cậy để truyền một khối lượng lớn dữ liệu, và nó đòi hỏi người lập trình phải xây dựng thủ tục nhận biết và phục hồi lỗi trong mỗi chương trình ứng dụng. Bởi vì rất khó để thiết kế, để hiểu, và sửa đổi phần mềm cung cấp độ tin cậy một cách chính xác, rất ít người lập trình ứng dụng có được nền tảng kỹ thuật cần thiết. Hệ quả là, một trong những mục đích của việc nghiên cứu giao thức mạng là để tìm ra các lời giải chung cho vấn đề cung cấp dịch vụ chuyển phát stream đáng tin cậy, giúp cho các chuyên gia có thể xây dựng

duy nhất một phiên bản có phần mềm giao thức stream mà tất cả các chương trình ứng dụng có thể sử dụng được. Việc có được duy nhất một giao thức chung cũng giúp tách biệt các chương trình ứng dụng khỏi các chi tiết của mạng, và ta cũng có thể định nghĩa một giao thức thống nhất cho dịch vụ truyền stream.

### 7.1.3. Các tính chất của dịch vụ chuyển phát tin cậy

Sự giao tiếp giữa các chương trình ứng dụng và dịch vụ chuyển phát tin cậy TCP/IP có thể đặc trưng hoá bởi 5 khía cạnh:

* Định hướng stream: Khi hai chương trình ứng dụng (các tiến trình của người sử dụng) truyền những khối lượng lớn dữ liệu, chúng ta xem dữ liệu như một chuỗi các bit, được chia thành các byte 8 bit, mà chúng ta thường gọi là byte. Dịch vụ chuyển phát stream trên máy đích chuyển đến nơi nhận một cách chính xác cùng một chuỗi các byte mà máy gửi chuyển nó đi.

* Kết nối mạch ảo: Thực hiện việc truyền stream cũng tương tự như thực hiện một cuộc gọi điện thoại. Trước khi việc truyền có thể bắt đầu, cả hai chương trình ứng dụng gửi và chương trình ứng dụng nhận tương tác với các hệ điều hành của chúng, thông báo chúng về mong muốn có được việc truyền stream. Về mặt khái niệm, một chương trình ứng dụng sẽ thực hiện một "cuộc gọi" mà phải được đầu kia chấp nhận. Các module phần mềm giao thức trong hai hệ điều hành thông tin liên lạc với nhau bằng cách gửi các thông điệp qua Internet, kiểm tra xem việc truyền đã được cho phép chưa. Một khi tất cả mọi chi tiết đã được thiết lập, các module giao thức thông báo cho các chương trình ứng dụng rằng kết nối đã được thiết lập và có thể bắt đầu việc truyền. Trong suốt quá trình truyền, phần mềm giao thức trên hai máy liên tục liên lạc với nhau để kiểm tra rằng dữ liệu nhận được một cách chính xác. Nếu việc thông tin liên lạc bị hỏng vì bất cứ lý do gì (ví dụ, phần cứng mạng trên con đường nối hai máy bị hỏng), cả hai máy đều nhận ra lỗi và thông báo cho chương trình ứng dụng tương ứng. Chúng ta sử dụng thuật ngữ mạch ảo để mô tả các kết nối này bởi vì mặc dù các chương trình ứng dụng xem kết nối này như mạch phần cứng được dành riêng, tính tin cậy là một sự ảo giác được cung cấp bởi dịch vụ chuyển phát stream.

* Việc truyền có vùng đệm: Các chương trình ứng dụng gửi một dòng dữ liệu qua mạch ảo bằng cách lập lại việc chuyển các byte dữ liệu đến phần mềm giao thức. Khi truyền dữ liệu, mỗi chương trình ứng dụng sử dụng bất kỳ kích thước đơn vị truyền nào nó thấy thuận tiện, mà có thể chỉ bằng một byte. Tại đầu nhận, phần mềm giao thức chuyển phát tự động dữ liệu theo đúng chính xác thứ tự mà chúng được gửi đi, làm cho chúng sẵn sàng được sử dụng đối với chương trình ứng dụng nhận, ngay sau khi chúng được nhận và kiểm tra. Phần mềm giao thức được tự do phân chia dòng dữ liệu thành những gói dữ liệu độc lập với đơn vị mà chương trình ứng dụng truyền đi. Để làm cho việc truyền hiệu quả hơn và để tối

thiểu giao thông trên mạng, các cài đặt thường tập hợp cho đủ dữ liệu từ dòng dữ liệu để đặt vào datagram có độ lớn thích hợp trước khi truyền nó qua Internet. Như thế, ngay cả khi chương trình ứng dụng phát sinh dòng dữ liệu gồm một byte mỗi lần, việc truyền qua Internet vẫn có thể hoàn toàn hiệu quả. Tương tự như vậy, nếu chương trình ứng dụng quyết định chuyển phát những khối dữ liệu vô cùng lớn, phần mềm giao thức có thể quyết định chia khối này thành những mảnh nhỏ hơn trước khi truyền đi.

Đối với những chương trình ứng dụng mà dữ liệu phải được chuyển phát ngay cả khi nó không đầy một vùng đệm, dịch vụ stream cung cấp một cơ chế đẩy (push) mà các chương trình ứng dụng sử dụng để bắt buộc truyền đi. Tại nơi gửi lệnh, lệnh đẩy bắt buộc TCP làm cho dữ liệu được sẵn sàng đối với chương trình ứng dụng mà không trì hoãn gì cả. Tuy nhiên, chúng ta cần lưu ý rằng, chức năng đẩy chỉ đảm bảo rằng tất cả dữ liệu sẽ được truyền đi; nó không hề cung cấp thông tin về đường biên của dữ liệu. Như thế, ngay cả khi việc chuyển phát bị bắt buộc, phần mềm giao thức có thể quyết định chia dòng dữ liệu.

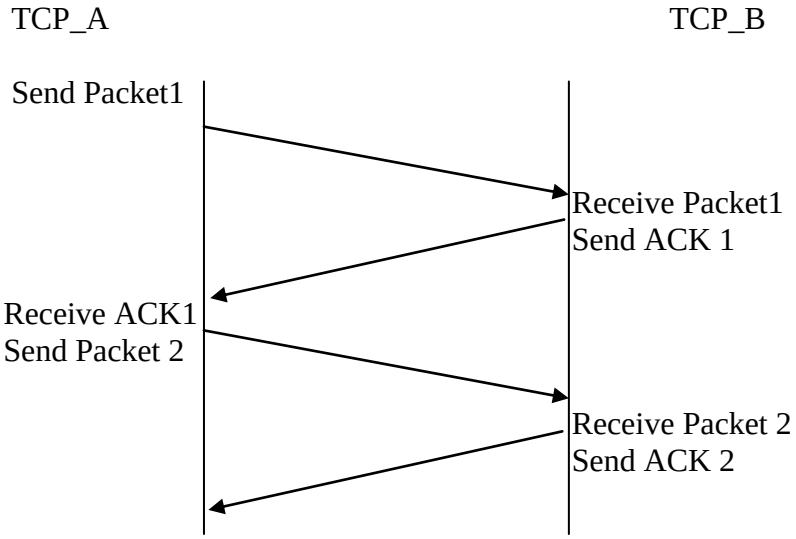
* Stream không có cấu trúc: Một điều quan trọng cần hiểu là dịch vụ TCP/IP stream không xác định các dòng dữ liệu có cấu trúc. Lấy ví dụ, chương trình trả lương nhân viên, không có cách nào để mà dịch vụ stream đánh dấu biên giới giữa các bản ghi nhân viên, hay để xác định nơi dừng của dòng dữ liệu là dữ liệu nhân viên. Các chương trình ứng dụng sử dụng dịch vụ stream phải hiểu nội dung stream và thông nhất với nhau về định dạng stream trước khi khởi động việc kết nối.

* Kết nối hai chiều: Các kết nối được cung cấp bởi dịch vụ TCP/IP stream cho phép truyền dữ liệu đồng thời từ cả hai chiều. Cách kết nối này được gọi là full-duplex (song công). Từ quan điểm của một tiến trình ứng dụng, một kết nối hai chiều bao gồm hai dòng dữ liệu độc lập "chạy" theo hai chiều ngược nhau, và không có tương tác hay va chạm. Dịch vụ stream cho phép một tiến trình ứng dụng chấm dứt "dòng chảy" theo một chiều trong khi dữ liệu vẫn tiếp tục chạy theo chiều kia, làm cho kết nối trở thành một chiều half duplex (bán song công). Ưu điểm chính của kết nối hai chiều là phần mềm giao thức cơ sở có thể gửi thông tin điều khiển cho một stream ngược trở về nguồn trong những datagram đang truyền tải dữ liệu theo chiều ngược lại. Điều này giúp giảm bớt giao thông trên mạng.

### 7.1.4. Tính tin cậy của dịch vụ chuyển phát tin cậy

Chúng ta đã nói rằng dịch vụ chuyển phát stream đáng tin cậy bảo đảm chuyển phát một dòng dữ liệu được gửi từ một máy đến máy khác mà không bị trùng lặp hay mất dữ liệu. Câu hỏi đặt ra là: "làm thế nào phần mềm giao thức có thể cung cấp việc truyền đáng tin cậy khi mà hệ thống thông tin liên lạc cơ sở cung cấp việc chuyển phát dữ liệu không đáng tin cậy?" câu trả lời thật là phức tạp,

nhưng hầu hết các giao thức đáng tin cậy đều sử dụng một kỹ thuật cơ bản có tên là đáp lời tích cực và truyền lại (positive acknowledgement with retransmission). Kỹ thuật này đòi hỏi nơi nhận phải liên lạc với nguồn, gửi ngược trở lại thông điệp acknowledgement (ACK) khi nó nhận được dữ liệu. Nơi gửi lưu lại thông tin (bản ghi) của mỗi gói dữ liệu nó gửi đi và đợi thông điệp acknowledgement trước khi gửi gói dữ liệu kế tiếp. Nơi gửi cũng khởi động một bộ đếm để đo thời gian nhận thông điệp acknowledgement.




*Hình 7.1: Trình bày cách đơn giản nhất mà giao thức đáp lời tích cực truyền dữ liệu.*

Trong hình này, những sự kiện xảy ra tại nơi gửi và nơi nhận được thể hiện lần lượt tại bên trái và bên phải. Mỗi đường chéo đi từ bên này qua bên kia thể hiện việc truyền một thông điệp qua mạng.

Khi một gói dữ liệu bị mất hoặc bị hỏng. Nơi gửi khởi động một bộ đếm thời gian ngay sau khi truyền một gói dữ liệu. Khi bộ đếm đã hết hạn, nơi gửi giả định rằng gói dữ liệu đã bị mất và phải truyền lại.

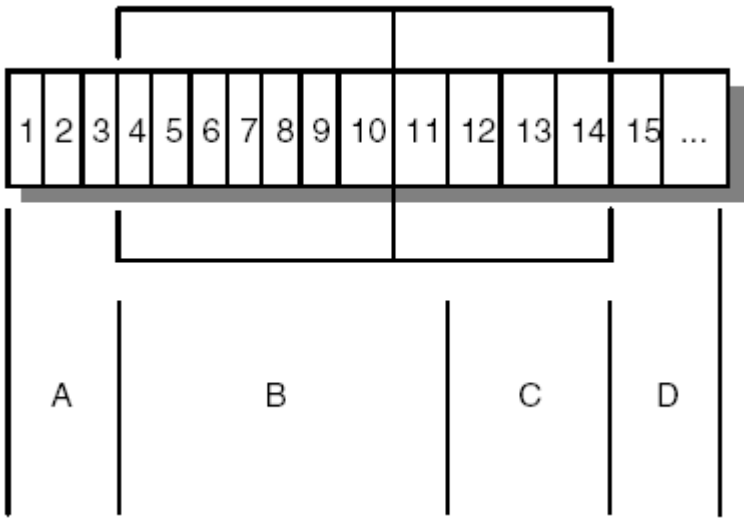
Vấn đề cuối cùng về tính tin cậy xuất hiện khi hệ chuyển phát dữ liệu cơ sở làm trùng lặp dữ liệu. Sự trùng lặp cũng có thể xuất hiện khi xảy ra sự trì hoãn lâu trên mạng, mà gây ra việc truyền lại sớm. Giải quyết vấn đề trùng lặp đòi hỏi phải thật cẩn thận bởi vì cả gói dữ liệu và acknowledgement đều có thể bị trùng lặp. Thông thường, các giao thức đáng tin cậy nhận biết sự trùng lặp dữ liệu bằng cách gán cho mỗi gói dữ liệu một số thứ tự và yêu cầu nơi nhận ghi nhớ lại những số thứ tự nào nó nhận được. Để tránh sự nhầm lẫn gây ra bởi sự trì hoãn hay trùng lặp các acknowledgement, các giao thức đáp lời tích cực cũng gửi số thứ tự ngược trở lại trong mỗi acknowledgement, để cho nơi nhận có thể phối hợp một cách chính xác các acknowledgement với các gói dữ liệu.

7.1.5. Ý tưởng kỹ thuật cửa sổ trượt

Trước khi xem xét dịch vụ stream TCP, chúng ta cần tìm hiểu thêm một khái niệm tạo nên cơ sở của việc truyền stream. Khái niệm này, có tên là cửa sổ trượt (sliding window), làm cho việc truyền stream được hiệu quả. Để hiểu được động lực thúc đẩy sự ra đời của khái niệm cửa sổ trượt, chúng ta tham khảo lại chuỗi sự kiện xảy ra ở trong hình 7.1. Để đạt được tính tin cậy, nơi gửi truyền một gói dữ liệu và sau đó đợi acknowledgement trước khi truyền gói dữ liệu khác. Như trình bày trong hình 7.1, tại mỗi thời điểm dữ liệu chỉ lưu chuyển giữa các máy theo 1 chiều, ngay cả nếu mạng có khả năng thông tin đồng thời theo hai chiều. Mạng sẽ hoàn toàn ở trạng thái nhàn rỗi trong khoảng thời gian máy tính trì hoãn việc đáp lời (ví du, trong khi máy tính phải tính checksum hay chọn định tuyến). Nếu chúng ta hình dung một mạng có độ trì hoãn truyền rất lớn, vấn đề trở nên rõ ràng hơn:

Một giao thức đáp lời tích cực đơn giản sẽ bỏ phí một lượng đáng kể băng thông của mạng bởi vì nó phải trì hoãn việc gửi gói dữ liệu mới cho đến khi nó nhận được acknowledgement của gói dữ liệu trước đó.

Kỹ thuật cửa sổ trượt là một dạng phức tạp hơn của đáp lời tích cực và truyền lại so với phương pháp đơn giản được trình bày. Các giao thức cửa sổ trượt sử dụng băng thông của mạng tốt hơn bởi vì chúng cho phép nơi gửi truyền nhiều gói dữ liệu trước khi qua trạng thái đợi acknowledgement. Cách dễ dàng nhất để tưởng tượng hoạt động của cửa sổ trượt là xét một dãy các gói dữ liệu sắp được truyền như trong hình 7.2. Giao thức này đặt một cửa sổ nhỏ có kích thước cố định lên dãy này và truyền đi tất cả những gói dữ liệu nào nằm trong cửa sổ.

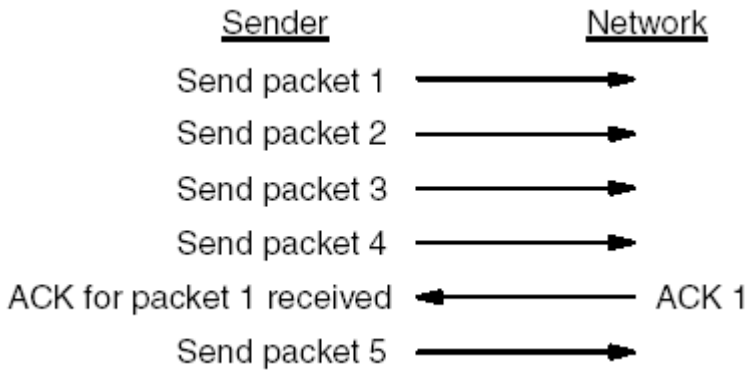



*Hình 7.2: Cửa sổ trượt*

Chúng ta nhớ rằng một gói dữ liệu không được đáp lời nếu nó đã được truyền đi nhưng nơi gửi không nhận được acknowledgement. Về mặt kỹ thuật, số lượng của gói dữ liệu mà có thể không được đáp lời tại một thời điểm bất kỳ bị ràng buộc

bởi kích thước của cửa sổ và bị giới hạn là một giá trị nhỏ và cố định. Lấy ví dụ, trong một giao thức cửa sổ trước có kích thước cửa sổ là 8, nơi gửi được phép truyền đi 8 gói dữ liệu trước khi nó nhận một acknowledgement.

Như trình bày trong hình 7.2, một khi nơi gửi nhận một acknowledgement của gói dữ liệu đầu tiên bên trong cửa sổ, nó "trượt" cửa sổ qua bên phải và gửi gói dữ liệu kế tiếp. Cửa sổ tiếp tục trượt khi nơi gửi vẫn còn nhận được acknowledgement. Hiệu suất của giao thức cửa sổ trượt tuỳ thuộc vào kích thước cửa sổ và tốc độ nhận dữ liệu của mạng. Hình 7.3 trình bày một ví dụ về cách hoạt động của giao thức cửa sổ trượt khi gửi ba gói dữ liệu. Lưu ý rằng nơi gửi truyền tất cả ba gói dữ liệu trước khi nhận được bất kỳ acknowledgement nào.




*Hình 7.3: Gửi 1 lần nhiều gói trong khi chờ nhận ACK*

Khi kích thước của cửa sổ là 1, giao thức cửa sổ trượt sẽ trở thành đơn giản giống y hệt giao thức đáp lời tích cực. Bằng việc tăng kích thước cửa sổ, hệ thống có thể loại bỏ hoàn toàn thời gian nhàn rỗi của mạng. Nghĩa là, trong trạng thái ổn định, nơi gửi có thể truyền gói dữ liệu đi với tốc độ nhanh như tốc độ truyền của mạng. Điểm chính yếu là:

Bởi vì một giao thức cửa sổ trượt được điều chỉnh tốt (theo kích thước cửa sổ và tốc độ mạng) làm cho mạng luôn luôn ở trạng thái bận rộn (truyền dữ liệu), kết quả là một mạng có hiệu suất cao hơn nhiều so với giao thức chỉ đơn giản là đáp lời tích cực).

Về mặt khái niệm, giao thức cửa sổ trượt luôn luôn ghi nhớ những gói dữ liệu nào đã được đáp lời (nhận được acknowledgement) và duy trì một bộ đếm thời gian riêng cho mỗi gói dữ liệu không được đáp lời. Nếu một gói dữ liệu bị mất, bộ đếm thời gian bị hết hạn và nơi gửi phải truyền lại gói dữ liệu đó. Khi nơi gửi trượt cửa sổ của nó tới, nó đi qua tất cả những gói dữ liệu đã được đáp lời. Tại nơi nhận, phần mềm giao thức duy trì một cửa sổ tương tự, chấp nhận các gói dữ liệu gửi đến và gửi trả về đáp lời (acknowledgement). Như thế cửa sổ phân chia dãy các gói dữ liệu thành ba tập hợp:

- Bên trái của cửa sổ là những gói dữ liệu đã được truyền đi thành công, đầu kia đã nhận được, đầu này đã nhận được lời đáp;
- Bên phải của cửa sổ là những gói dữ liệu chưa được truyền đi;
- Bên trong cửa sổ là những gói dữ liệu đang được truyền đi. Gói dữ liệu được đánh số thấp nhất trong cửa sổ là gói dữ liệu đầu tiên trong dãy này mà chưa nhận được lời đáp.

## 7.2. Nguyên lý hoạt động của giao thức TCP

### 7.2.1. Giao thức điều khiển truyền

Phần trước đã trình bày về nguyên lý của cửa sổ trượt, trong phần này, chúng ta xem xét dịch vụ stream đáng tin cậy được cung cấp bởi bộ giao thức TCP/IP Internet. Dịch vụ này được xác định bởi giao thức điều khiển việc truyền (Transmission Control Protocol TCP). Dịch vụ stream đáng tin cậy quan trọng đến nỗi mà tổng thể bộ giao thức được gọi là TCP/IP. Điều quan trọng cần phải hiểu là:

TCP là một giao thức thông tin liên lạc, không phải là thành phần của phần mềm.

Sự khác biệt giữa một giao thức và phần mềm cài đặt nó thì cũng tương tự như sự khác biệt giữa định nghĩa của một ngôn ngữ lập trình và một phần mềm viết bằng ngôn ngữ đó. Bởi vì trong thế giới ngôn ngữ lập trình, đôi khi sự phân biệt giữa định nghĩa và cài đặt là không rõ ràng. Chúng ta thường gặp và làm việc với phần mềm TCP nhiều hơn là với các định nghĩa đặc tả giao thức, vì thế cũng là "tự nhiên" khi người ta xem một cài đặt cụ thể như một chuẩn. Vì vậy, chúng ta nên cố gắng phân biệt sự khác nhau giữa chúng.

Giao thức TCP đặc tả định dạng của dữ liệu và lời đáp (acknowledgement) mà hai máy tính trao đổi để đạt được việc truyền đáng tin cậy, cũng như là những thủ tục mà các máy tính sử dụng để đảm bảo rằng dữ liệu đến được một cách chính xác. Nó đặc tả cách mà phần mềm TCP phân biệt trong số các đích trên một máy cụ thể, và cách này mà các máy thông tin liên lạc với nhau để phục hồi các lỗi như mất gói dữ liệu hay dữ liệu bị trùng lặp. Giao thức này cũng xác định phương thức hai máy tính khởi động qua trình truyền stream TCP và cách mà chúng thống nhất với nhau khi nào thì hoàn tất việc truyền.

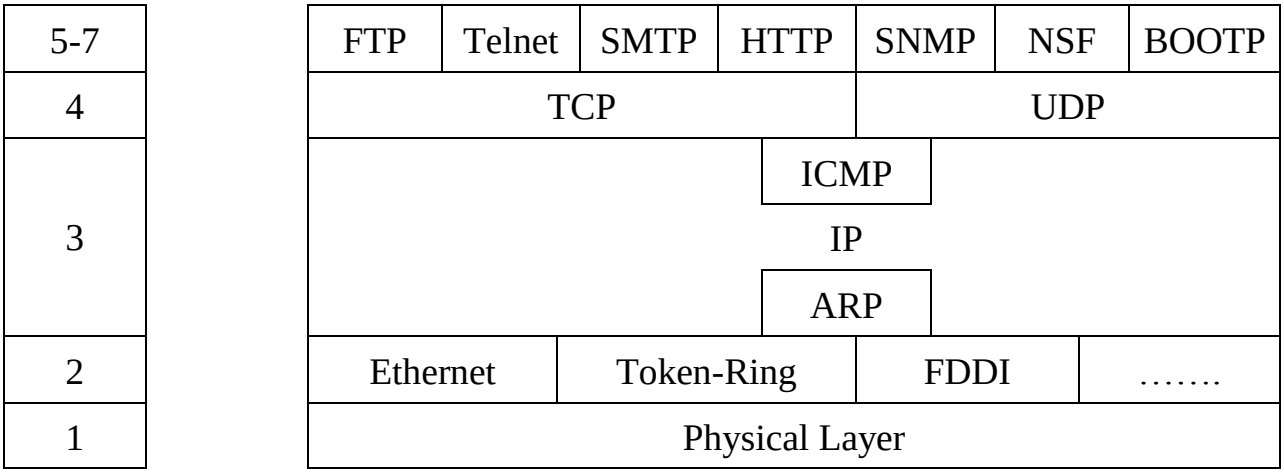
Mặc dù đặc tả TCP có mô tả một cách tổng quát về cách mà các chương trình ứng dụng sử dụng TCP, nhưng nó không chỉ ra chi tiết về sự giao tiếp giữa một chương trình ứng dụng và TCP. Nghĩa là giao thức chỉ trình bày những hoạt động mà TCP cung cấp; nó không đặc tả chính xác thủ tục mà chương trình ứng dụng gửi đến để truy xuất các hoạt động này. Một lý do mà TCP không đặc tả chi tiết về giao tiếp với chương trình ứng dụng là vì tính uyển chuyển trong cài đặt nêu trên,

TCP có thể sử dụng để xây dựng phần mềm cho nhiều loại máy khác nhau, vì thế nó gần như độc lập với hạ tầng phần cứng và hệ điều hành.

Bởi vì TCP có rất ít giả định về hệ thống thông tin liên lạc cơ sở, TCP có thể được sử dụng với nhiều hệ chuyển phát dữ liệu khác nhau, bao gồm cả dịch vụ chuyển phát IP Datagram. Lấy ví dụ, TCP có thể được cài đặt để sử dụng với đường điện thoại dial-up, với mạng cục bộ, với mạng cáp quang tốc độ cao, hay mạng đường dài tốc độ chậm, hoặc mạng không dây cũng như mạng di động thế hệ mới. Việc TCP hoạt động tốt trên nhiều hệ thống chuyển phát khác nhau là một trong những điểm mạnh của giao thức này.

### 7.2.2. Cổng, kết nối, và điểm cuối

Giống như User Datagram Protocol (UDP) đã trình bày trong chương trước, TCP nằm trên IP trong mô hình phân lớp giao thức. Cửa sổ trượt trình bày tổ chức khái niệm của mô hình phân cấp. TCP cho phép nhiều chương trình ứng dụng trên một máy được phép thông tin liên lạc đồng thời, và demultiplex những dữ liệu TCP gửi đến trong số các chương trình ứng dụng. Cũng giống như UDP, TCP sử dụng giá trị cổng ứng dụng để xác định đích cuối cùng trong một máy. Mỗi cổng được gán cho một giá trị số nguyên nhỏ, và đây là định danh của nó (mặc dù cả hai TCP và UDP sử dụng các định danh cổng là số nguyên bắt đầu từ 1 để xác định cổng, không có sự nhầm lẫn giữa chúng bởi vì một IP Datagram gửi đến sẽ xác định cả giá trị cổng cũng như giao thức được sử dụng).




ARP: Address Resolution Protocol

BOOTP: Bootstrap Protocol

FTP: File Transfer Protocol

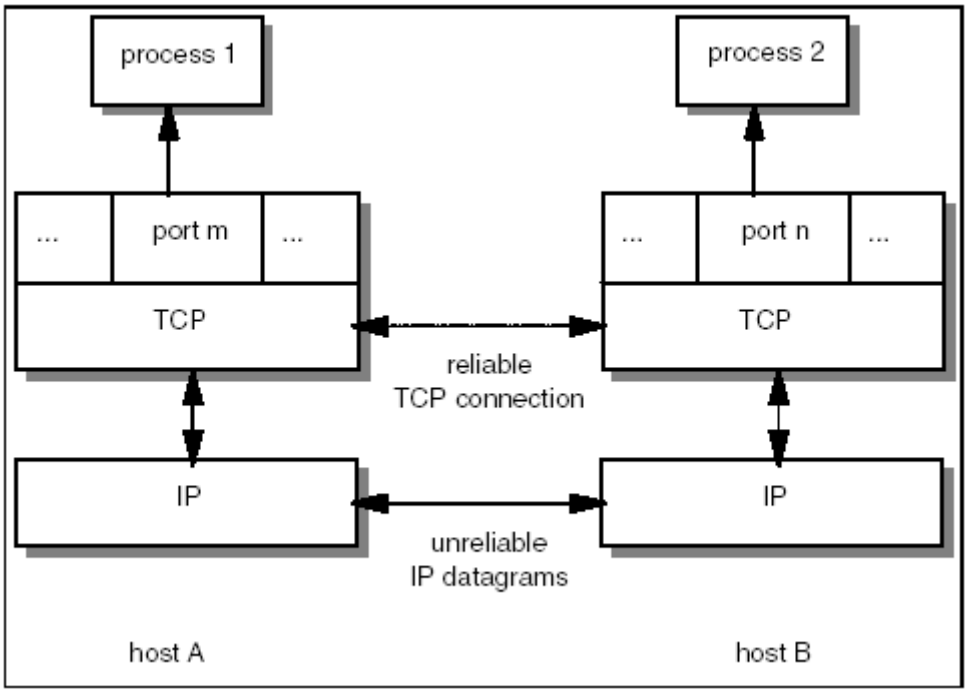
HTTP: HyperText Transmission Protocol

SNMP: Simple Network Management Protocol

ICMP: Internet Control Message Protocol

*Hình 7.4: Vị trí TCP trong mô hình TCP/IP*

Khi chúng ta tìm hiểu về cổng ứng dụng trong phần về giao thức UDP, chúng ta biết mỗi cổng được xem như một hàng đợi mà trong đó phần mềm giao thức sẽ lưu trữ các datagram gửi đến. Các cổng TCP thì phức tạp hơn rất nhiều bởi vì một giá trị cổng cho trước không tương ứng với một đối tượng đơn. Thay vì vậy, TCP được xây dựng trên kết nối trừu tượng, mà trong đó các đối tượng được xác định, là những liên kết mạch ảo, không phải từng cổng. Điều cốt yếu là phải hiểu được rằng TCP sử dụng ý niệm về các kết nối bởi vì nó giúp chúng ta giải thích ý nghĩa và việc sử dụng các giá trị cổng TCP:




*Hình 7.5: Ý nghĩa sử dụng giá trị cổng IP*

TCP sử dụng kết nối, không phải cổng ứng dụng, là sự trừu tượng cơ sở của nó; các kết nối được xác định bởi một cặp các điểm cuối (socket).

Một cách chính xác thì các "điểm cuối" của một kết nối là gì? Chúng ta đã nói rằng một kết nối bao gồm một mạch ảo giữa hai chương trình ứng dụng, vì vậy có vẻ tự nhiên khi giả định rằng chương trình ứng dụng phục vụ làm "điểm cuối" của kết nối. Nhưng không phải thế. Thay vì vậy, TCP định nghĩa một điểm cuối là một cặp thông số (địa chỉ IP của máy, cổng TCP trên máy đó). Ví dụ, socket (128.10.2.3, 25) xác định cổng TCP 25 trên máy có địa chỉ 128.10.2.3.

Bây giờ, khi chúng ta đã định nghĩa điểm cuối, thì sẽ dễ dàng hơn để hiểu được kết nối. Như thế, ví dụ nếu có một kết nối từ máy (18.26.0.36) tại viện đại

học MIT đến máy (128.10.2.3) tại viện đại học Purdue, nó có thể được xác định bởi các điểm cuối:

(18.26.0.36, 1069) và (128.10.2.3, 25)

Trong khi đó, có thể đang có một kết nối khác từ máy (128.9.0.32) tại Viện Khoa học thông tin (CMU) tới cùng máy tính trên ở Viện đại học Purdue, được xác định bởi các điểm cuối của nó:

(128.9.0.32, 1184) và (128.10.2.3, 53)

Cho tới bây giờ, các ví dụ của chúng ta về kết nối đơn giản và dễ hiểu bởi vì các cổng được sử dụng tại tất cả các điểm cuối là duy nhất. Tuy nhiên, kết nối trừu tượng cho phép nhiều kết nối cùng chia xẻ một điểm cuối. Lấy ví dụ, chúng ta có thể thêm kết nối khác vào hai cổng ở trên từ máy (128.2.254.139) tại CMU tới máy tại Viện đại học Purdue:

(128.2.254.139, 1184) và (128.10.2.3, 53)

Thoạt nhìn, có vẻ hơi lạ khi hai kết nối có thể đồng thời sử dụng cổng TCP số 53 trên máy 128.10.2.3, nhưng không có gì nhầm lẫn cả. Bởi vì TCP phối hợp các thông điệp gửi đến với một kết nối thay vì với cổng ứng dụng, nó sử dụng cả hai điểm cuối để xác định kết nối thích hợp. Ý tưởng quan trọng cần nhớ là:

TCP xác định kết nối bằng một cặp các điểm cuối, một cổng TCP cho trước có thể được dùng chung bởi nhiều kết nối trên cùng một máy.

Dưới quan điểm của người lập trình, sự kết nối trừu tượng rất có ý nghĩa. Nó có nghĩa là người lập trình có thể tạo ra một chương trình cung cấp dịch vụ song song cho kết nối đồng thời mà không cần một cổng (cục bộ) duy nhất cho mỗi kết nối. Lấy ví dụ, hầu hết các hệ thống cung cấp việc truy xuất song song vào hệ thống thư điện tử của chúng, cho phép nhiều máy tính gửi thư điện tử cho chúng cùng một lúc. Bởi vì chương trình nhận thư điện tử sử dụng TCP để thông tin liên lạc, nó chỉ cần sử dụng một cổng TCP cục bộ mặc dù nó cho phép nhiều kết nối xử lý đồng thời.

### 7.2.3. Cơ chế mở chủ động và mở thụ động

Không giống như UDP, TCP là giao thức kết nối có định hướng (connecttion oriented), đòi hỏi cả hai điểm cuối cùng đồng ý tham gia. Nghĩa là, trước khi một giao dịch TCP có thể chuyển qua Internet, các chương trình ứng dụng ở hai đầu của kết nối phải cùng đồng ý rằng chúng mong muốn có kết nối. Để làm việc đó, chương trình ứng dụng ở bên này thực hiện một chức năng mở thụ động (passive open) bằng cách liên hệ với hệ điều hành của nó và chỉ ra rằng nó sẽ chấp nhận một yêu cầu kết nối (đến từ đầu kia). Vào lúc đó, hệ điều hành sẽ gán một giá trị cổng TCP cho kết nối tại đầu của nó. Chương trình ứng dụng ở bên kia phải liên hệ

với hệ điều hành của nó sử dụng một yêu cầu mở chủ động (active open) để thiết lập kết nối. Hai module phần mềm TCP sẽ thông tin liên lạc với nhau để thiết lập và kiểm tra kết nối. Một khi kết nối đã được tạo xong, các chương trình ứng dụng có thể truyền dữ liệu; các module phần mềm TCP tại mỗi đầu trao đổi thông điệp với nhau để bảo đảm việc chuyển phát đáng tin cậy. Chúng ta sẽ quay trở lại trong phần sau để tìm hiểu chi tiết việc thiết lập các kết nối sau khi xem định dạng thông điệp TCP.

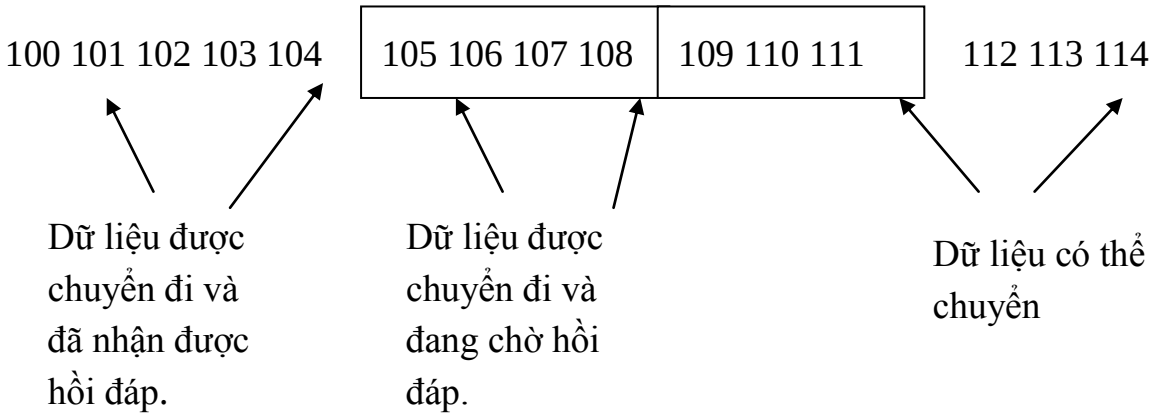
### 7.2.4. Cơ chế truyền dữ liệu trong cửa số trượt

TCP xem một dòng dữ liệu như một dãy các byte được chia thành những đoạn (segment) để truyền đi. Thông thường, mỗi segment di chuyển qua Internet trong một IP Datagram.

TCP sử dụng một cơ chế cửa sổ trượt đặc biệt để giải quyết hai vấn đề quan trọng: tăng hiệu quả việc truyền và điều khiển tốc độ dòng dữ liệu. Giống như giao thức cửa sổ trượt đã mô tả trước đây, cơ chế cửa sổ trượt cho TCP cho phép nó gửi đi nhiều segment trước khi nhận được một lời đáp (acknowledgement). Làm như vậy sẽ tăng được toàn bộ hiệu suất và giảm thời gian "nhàn rỗi" của mạng. Dạng TCP của giao thức cửa sổ trượt cũng giải quyết vấn đề điều khiển tốc độ dòng chuyển end to end, bằng cách cho phép nơi nhận giới hạn lại việc truyền cho đến khi nó có đủ không gian vùng đệm để chấp nhận thêm dữ liệu.

Cơ chế cửa sổ trượt TCP hoạt động theo byte, không phải theo segment hay theo gói dữ liệu. Các byte của dòng dữ liệu được đánh số tuần tự, và nơi gửi duy trì ba con trỏ phối hợp với mỗi kết nối. Các con trỏ này định nghĩa cửa sổ trượt như minh hoạ trong hình 7.6. Con trỏ đầu tiên đánh dấu biên bên trái cửa sổ trượt, tách biệt những byte đã được gửi và đã nhận được lời đáp ra khỏi những byte còn chưa được đáp lời. Con trỏ thứ hai đánh dấu biên bên phải cửa sổ trượt và xác định byte cao nhất trong dãy này mà có thể được gửi đi trước khi nhận được thêm lời đáp. Con trỏ thứ ba đánh dấu biến bên trong cửa sổ để tách biệt những byte đã được gửi đi và những byte chưa được gửi đi. Phần mềm giao thức gửi đi tất cả các byte trong cửa sổ mà không hề trì hoãn, vì vậy đường biên bên trong cửa sổ trượt luôn luôn di chuyển nhanh chóng từ trái sang phải.

Cửa sổ dựa vào các thông báo khi máy đích nhận được gói tin




*Hình 7.6:Hoạt động của cửa sổ trượt*

Chúng ta đã mô tả cách mà cửa sổ TCP của máy gửi trượt đi và cũng đã lưu ý rằng máy nhận cũng phải duy trì một cửa sổ tương tự để ráp dữ liệu lại. Tuy nhiên, vì các kết nối TCP là hai chiều, hai quá trình truyền xảy ra đồng thời trên mỗi kết nối, mỗi quá trình theo một chiều. Chúng ta xem việc truyền là hoàn toàn độc lập bởi vì tại thời điểm bất kỳ dữ liệu có thể di chuyển qua kết nối theo một chiều, hay theo cả hai chiều. Như thế, phần mềm TCP tại mỗi đầu duy trì hai cửa sổ cho mỗi kết nối (tổng cộng là bốn), một cửa sổ trượt theo dòng dữ liệu được gửi đi, còn cửa sổ kia trượt theo dữ liệu được nhận vào.

### 7.2.5. Cửa sổ với kích thước và việc điều khiển tốc độ truyền

Có một khác biệt lớn giữa giao thức cửa sổ trượt của TCP và giao thức cửa sổ trượt được đơn giản hoá đã trình bày trước đây, đó là TCP cho phép kích thước cửa sổ có thể thay đổi qua tuỳ thời điểm. Với mỗi lời đáp xác định có bao nhiêu byte đã được nhận, có chứa một thông cáo cửa sổ để xác định có thêm bao nhiêu byte (dữ liệu) mà máy nhận được chuẩn bị để nhận. Chúng ta có thể xem thông cáo cửa sổ như một cách xác định kích vùng đệm hiện tại của máy nhận. Để đáp lại việc gia tăng kích thước thông cáo cửa sổ, máy gửi sẽ tăng kích thước cửa sổ trượt của nó và tiến hành gửi các byte của nó còn chưa được đáp lời. Và để đáp lại việc giảm bớt kích thước thông cáo cửa sổ, máy gửi sẽ giảm kích thước cửa sổ trượt của nó và thôi gửi các byte vượt qua biên cửa sổ. Phần mềm TCP không được phủ nhận các giá trị thông cáo trước bằng cách thu hẹp cửa sổ lại vượt qua các vị trí chấp nhận được trước đó trong chuỗi các bytes.

Ưu điểm của việc sử dụng cửa sổ có kích thước thay đổi là nó hỗ trợ việc điều khiển tốc độ truyền dữ liệu cũng như là việc truyền đáng tin cậy. Để tránh việc nhận nhiều dữ liệu hơn khả năng lưu trữ, nơi nhận sẽ gửi đi thông cáo cửa sổ có kích thước là zero để ngưng tất cả việc truyền. Sau đó, khi không gian vùng

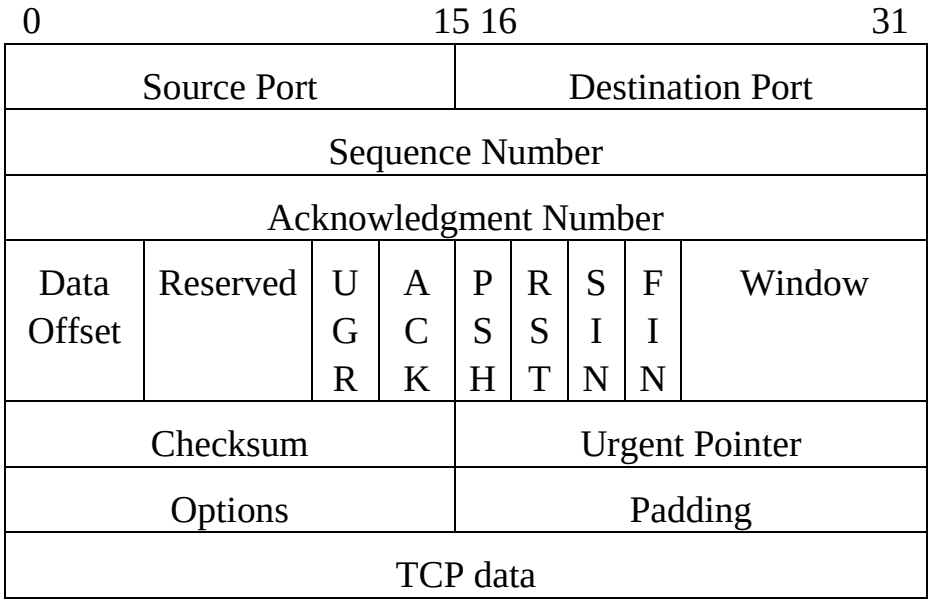
đệm đã được giải phóng bớt, nơi nhận lại gửi đi thông cáo cửa sổ có kích thước khác zero để kích hoạt trở lại việc truyền dữ liệu (có hai ngoại lệ trong việc truyền khi kích thước cửa sổ là zero. Trước hết, nơi gửi được phép truyền một segment với bit URGENT lập để thông báo nơi nhận rằng có dữ liệu quan trọng. Thứ hai, để tránh nguy cơ nghẽn mạch có thể xuất hiện trong tình huống khi một thông cáo cửa sổ có kích thước khác zero bị thất lạc sau khi nó đã có kích thước zero, nơi nơi gửi thăm dò theo định kỳ cửa sổ có kích thước zero).

Việc có một cơ chế kiểm soát tốc độ truyền dữ liệu là cốt tử trong môi trường Internet, trong đó có nhiều máy tính có dung lượng bộ nhớ đệm & tốc độ truyền khác nhau khi liên lạc với nhau qua các mạng và bộ định tuyến. Có hai vấn đề độc lập nhau về tốc độ truyền. Trước hết, các giao thức Internet cần có sự kiểm soát tốc độ truyền end to end giữa nguồn và đích cuối cùng. Lấy ví dụ, khi một máy tính cỡ trung liên lạc với một máy mainframe lớn, máy tính cỡ trung cần phải điều chỉnh lượng dữ liệu, nếu không phần mềm giao thức sẽ nhanh chóng bị quá tải. Như thế TCP phải cài đặt việc kiểm soát tốc độ truyền end to end để bảo đảm việc chuyển phát đáng tin cậy. Thứ hai, các giao thức Internet cần phải có một cơ chế kiểm soát tốc độ truyền để cho phép các hệ thống trung gian (ví dụ, bộ định tuyến) kiểm soát một nguồn mà gửi đi nhiều dữ liệu hơn khả năng nhận của một máy.

Khi các máy trung gian bị quá tải, thì tình huống này truyền thông gọi là nghẽn mạch (congestion), và cơ chế để giải quyết vấn đề này được gọi là cơ chế kiểm soát sự nghẽn mạch (congestion control). TCP sử dụng mô hình cửa sổ trượt của nó để giải quyết vấn đề kiểm soát tốc độ truyền end to end. Tuy nhiên, chúng ta sẽ thấy sau này, một cài đặt nghẽn mạch dù một mô hình truyền lại đã được chọn cẩn thận có thể giúp tránh khỏi sự nghẽn mạch, có một mô hình lựa chọn tồi có thể làm nó trầm trọng hơn.

### 7.2.6. Định dạng của TCP segment

Đơn vị truyền giữa phần mềm TCP trên hai máy được gọi segment. Các segment được trao đổi để thiết lập các kết nối, để truyền dữ liệu, để gửi acknowledgement, để thông báo kích thước cửa sổ, và để đóng kết nối. Trong TCP, một acknowledgement chuyển từ máy A đến máy B có thể di chuyển trong cùng một segment như là dữ liệu truyền từ máy A đến máy B, mặc dù acknowledgement để chỉ dữ liệu đã gửi từ B đến A (trong thực tế, việc này không thường xuyên xảy ra bởi vì hầu hết các ứng dụng không đồng thời gửi dữ liệu theo cả hai chiều). Hình 7.7 Trình bày định dạng của segment.




*Hình 7.7: TCP Segment*

Mỗi segment được chia thành hai phần, phần đầu và dữ liệu. Phần đầu, có phần đầu TCP, chuyển tải thông tin điều khiển và các định danh cần thiết khác.

- Các vùng SOURCE PORT và DESTINATION PORT chứa các giá trị cổng TCP để xác định các chương trình ứng dụng tại hai đầu của kết nối.
- Vùng SEQUENCE NUMBER xác định vị tí trong chuỗi các byte dữ liệu trong segment của nơi gửi.
- Vùng ACKNOWLEDGEMENT NUMBER xác định số lượng byte mà nguồn đang đợi để nhận kế tiếp. Lưu ý rằng SEQUENCE NUMBER để chỉ đến lượng dữ liệu theo cùng chiều với segment, trong khi giá trị ACKNOWLEDGEMENT NUMBER để chỉ đến lượng dữ liệu theo chiều ngược lại với segment.
- Vùng HLEN chứa một số nguyên để xác định độ dài của phầng đầu segment, điện tử tính theo bội số của 32 bit (đặt tả giao thức mô tả vùng HLEN là vị trí tương đối của ùng dữ liệu bên trong segment). Cần có giá trị của HLEN bởi vì vùng OPTIONS có độ dài thay đổi, tuỳ thuộc vào những chọn thay đổi tuy vào các chọn lựa đã được lấy.
- Vùng RESERVED được dành riêng sử dụng trong tương lai.

Có những segment chỉ chuyển tải acknowledgement, còn những segment khác chuyển tải dữ liệu. Cũng có những segment chuyển tải những yêu cầu để thiết lập hoặc đóng lại một kết nối. Phần mềm TCP sử dụng vùng 16 bit có tên CODE BITS để xác định mục đích và nội dung của segment. Sau bit nàycũng cho biết

cách diễn dịch các vùng khác trong phần đầu, dựa vào nội dung của bảng trong hình 7.8

| URG | ACK | PSH | RST | SYN | FIN |
|---|---|---|---|---|---|

| | |
|---|---|
| URG | Urgent Pointer field significant |
| ACK | Acknowledement field signficant |
| PSK | Push function |
| RST | Reset the connection |
| SYN | Synchronize sequence numbers |
| FIN | Release the connection |

*Hình 7.8: Các bít xác định dịch vụ*

Phần mềm TCP cũng thông báo cho biết bao nhiêu dữ liệu nó sẵn sàng nhận mỗi khi nó gửi một segment cách xác định kích thước vùng đệm của nó trong vùng WINDOW. Vùng này chứa một số nguyên không dấu theo thứ tự byte mạng chuẩn.

### 7.2.7. Dữ liệu ngoài dòng (out of band)

Mặc dù TCP là giao thức định hướng dòng dữ liệu, đôi khi cũng cần thiết để cho một chương trình ở một đầu của kết nối gửi dữ liệu ngoài dòng (out of band), mà không phải đến khi chương trình ở đầu kia của kết nối "xử lý" hết các byte đã ở trong dòng dữ liệu. Ví dụ, khi TCP được sử dụng cho việc truy xuất từ xa (remote login), người sử dụng có thể quyết định gửi một loạt các tín hiệu từ bàn phím để ngắt hay huỷ bỏ một chương trình tại đầu kia. Thông thường, các tín hiệu như thế rất cần thiết khi mà chương trình trên máy ở xa không hoạt động chính xác. Các tín hiệu này phải được gửi đi mà không cần đợi đến khi chương trình (ở đầu kia) đọc hết các byte trong dòng dữ liệu TCP. (Nếu không thì không thể huỷ bỏ điện tử chương trình mà đang đọc dữ liệu).

Để dung nạp điện tử tín hiệu out of band, TCP cho phép nơi gửi xác định loại dữ liệu nào là khẩn cấp, có nghĩa là chương trình nhận phải được thông báo tức thì ngay khi dữ liệu đến, cho dù nó có nằm đâu trong dòng dữ liệu. Giao thức này mô tả rằng khi thấy có dữ liệu khẩn cấp, TCP (nơi nhận) phải thông báo cho bất cứ chương trình nào đang phối hợp với kết nối này để chuyển sang "trạng thái khẩn cấp". Sau khi đã xử lý tất cả dữ liệu khẩn cấp, TCP thông báo cho chương trình ứng dụng trở về trạng thái thông thường.

Dĩ nhiên chi tiết cụ thể về cách mà TCP thông báo cho chương trình ứng dụng về dữ liệu khẩn cấp tuỳ thuộc vào hệ điều hành của máy tính. Cơ chế được sử

dụng để đánh dấu loại dữ liệu khẩn cấp khi truyền nó trong một segment bao gồm mã bit URG trong vùng CODE BITS và vùng URGENT POINTER. Khi bit URG được lập, vùng URGENT POINTER xác định vị trí cuối cùng của dữ liệu khẩn trong segment.

### 7.2.8. Kích thước tối đa của segment

Không phải tất cả các segment gửi qua một kết nối sẽ có cùng một kích thước. Tuy nhiên, cả hai đầu phải thống nhất với nhau về kích thước tối đa của segment mà chúng sẽ truyền. Phần mềm TCP tại đầu này sử dụng vùng OPTIONS để thương thảo với phần mềm tại đầu kia của kết nối; một trong những lựa chọn này cho phép phần mềm TCP xác định kích thước tối đa của segment (maximum segment size: MSS), mà hai bên sẵn lòng nhận. Lấy ví dụ, một hệ thống nhúng với chỉ vài trăm byte cùng đệm nối vào một siêu máy tính lớn, chúng có thể thương thảo với nhau để có một MSS sao cho các segment không vượt quá độ lớn của vùng đệm. Một điều đặc biệt quan trọng cho các máy tính được nối với mạng cục bộ tốc độ cao là việc chọn kích thước tối đa của segment để chứa gói dữ liệu. Như thế nếu hai đầu cùng thuộc một mạng vật lý, TCP thường tính kích thước tối đa của segment sao cho có được IP Datagram mà vừa khớp với MTU mạng. Nếu hai đầu không cùng nằm trên một mạng vật lý, chúng có thể tìm cách biết để được MTU nhỏ nhất của mạng nằm trên con đường giữa hai điểm, hoặc chọn kích thước tối đa của segment là 536 (đó là kích thước mặc định của một IP Datagram, 576, trừ đi kích thước chuẩn của phần đầu IP và phần đầu TCP).

Trong môi trường Internet tổng quát, việc chọn kích thước tối đa tốt cho segment là một việc khó bởi vì hiệu suất có thể rất kém đối với những kích thước segment rất nhỏ hoặc là những kích thước segment rất lớn. Trong trường hợp đầu, khi segment có kích thước nhỏ, mức độ sử dụng mạng sẽ rất thấp. Để thấy được lý do, chúng ta nhớ lại rằng các segment TCP khi di chuyển được đóng gói trong các IP Datagram, mà lại được đóng gói trong các frame mạng vật lý. Như thế, ngoài phần dữ liệu ra mỗi segment có thêm ít nhất 40 byte phần đầu TCP và phần đầu IP. Vì vậy, những datagram chỉ chuyển tải một byte dữ liệu sử dụng nhiều nhất 1/41 của băng thông mạng cơ sở dành cho dữ liệu của người sử dụng.

Trong trường hợp thứ hai, những kích thước segment quá lớn cũng có thể gây nên hiệu quả kém. Những segment lớn sẽ tạo ra những IP Datagram lớn. Khi các datagram đó di chuyển qua mạng có MTU nhỏ, IP phải phân đoạn chúng ra. Không giống như TCP segment, một phân đoạn (fragment) không thể được đáp lời (acknowledge) cũng không thể được truyền lại một cách độc lập; tất cả các phân đoạn phải đến được nếu không thì toàn bộ datagram phải được truyền lại. Bởi vì xác suất của việc thất lạc một fragment là khác zero, nên khi gia tăng kích thước segment lớn hơn mức phân đoạn sẽ làm giảm hiệu suất của mạng.

Trên lý thuyết, kích thước tối ưu của segment MSS rất cần thiết khi các IP Datagram chuyển tải những segment đủ lớn mà vẫn chưa phải phân đoạn trong suốt lộ trình từ nguồn tới đích cuối cùng. Trong thực tế, tìm ra MSS là công việc đầy khó khăn vì một số lý do. Trước hết, hầu hết các cài đặt của TCP không bao gồm một cơ chế để làm việc này. Thứ hai, bởi vì các bộ định tuyến trong Internet có thể thay đổi định tuyến bất cứ lúc nào, con đường mà datagram đi qua giữa một cặp máy tính đang thông tin liên lạc với nhau cũng có thể bị thay đổi bất cứ lúc nào và như thế làm cho datagram có thể bị phân đoạn (do đi qua mạng có MTU nhỏ). Thứ ba, kích thước tối ưu phụ thuộc vào phần đầu của giao thức cấp thấp (ví dụ, kích thước segment phải bị giảm bớt để dung nạp được các lựa chọn IP). Nên việc nghiên cứu để tìm ra kích thước segment tối ưu vẫn còn tiếp tục.

### 7.2.9. Tính TCP Checksum

Vùng CHECKSUM trong phần đầu TCP bao gồm một checksum số nguyên 16 bit được sử dụng để kiểm chứng tính toàn vẹncủa dữ liệu cũng như là phần đầu TCP. Để tính checksum, phần mềm TCP trên máy gửi tuân theo một thủ tục như đã mô tả trong chương 6 cho UDP. Nó gán vào segment một phần đầu giả, thêm vào một số bit zero sao cho độ lớn của segment là bội số của 16 bit, và tính checksum 16 bit trên toàn bộ segment (mới). TCP không đếm phần đầu giả, cũng không thêm vào độ dáim, và cũng không truyền nó đi. Tương tự, nó giả định bản thân vùng checksum là zero cho mục đích tính checksum. Cũng như với những checksum khác, TCP sử dụng phép tính số học 16 bit và lấy phần bù của một của tổng các phần bù của một. Tại nơi nhận, phần mềm TCP cũng thực hiện tính toán tương tự để kiểm chứng rằng segment đến được nguyên vẹn.

Mục đích của việc sử dụng phần đầu giả cũng y hệt như đối với UDP. Nó cho phép nơi nhận kiểm chứng rằng segment đã đến đúng chính xác đích của nó, bao gồm cả địa chỉ IP của máy cũng như giá trị cổng ứng dụng. Cả hai địa chỉ Ip nguồn và đích đều quan trọng đối với TCP bởi vì nó phải sử dụng chúng để xác định kết nối mà segment được truyền trên đó. Như thế, bất cứ khi nào một datagram đến mạng theo một segment TCP, thì IP phải chuyển tới TCP địa chỉ IP nguồn, địa chỉ IP đích lấy được từ datagram, và bản thân segment. Hình 7.9 trình bày định dạng của phần đầu giả được sử dụng trong việc tính checksum.

<table>
<tr><td colspan="3">Source IP address</td></tr>
<tr><td colspan="3">Destination IP address</td></tr>
<tr><td>Zero</td><td>Protocol</td><td>TCP Length</td></tr>
</table>

*Hình 7.9: Phần đầu giả của TCP Segment*

Trong TCP gửi đi, vùng PROTOCOL sẽ được gán giá trị mà hệ thống chuyển phát cơ sở sẽ sử dụng trong vùng kiểu giao thức của nó. Đối với IP Datagram đang chuyển tải TCP, giá trị này là 6. Vùng TCP LENGTH xác định tổng độ dài của segment TCP bao gồm cả phần đầu TCP. Tại nơi nhận, thông tin được sử dụng trong phần đầu giả được trich ra từ IP Datagram mà đã chuyển tải segment và được đưa vào trong việc tính checksum để kiểm chứng rằng segment đến đúng chính xác đích của nó một cách nguyên vẹn.

### 7.2.10. Đáp lời và việc truyền lại

Bởi vì TCP gửi dữ liệu đi trong những segment có độ dài thay đổi và bởi vì các segment được truyền lại có thể bao gồm nhiều hơn dữ liệu gốc, sự xác nhận (acknowledgement) không thể dễ dàng tham chiếu tới datagram hay segment. Thay vì vậy, chúng chỉ tới vị trí ở trong dòng dữ liệu, được đanh theo số thứ tự. Nơi nhận tập hợp các byte dữ liệu từ những segment gửi đến và xây dựng lại một phiên bản giống y như dòng dữ liệu gửi đi. Bởi vì các segment di chuyển trong IP Datagram, chúng có thể bị mất hoặc chuyển phát không đúng thứ tự; nơi nhận sử dụng số thứ tự này để sắp lại thứ tự các segment. Tại thời điểm bất kỳ, nơi nhận sẽ xây dựng lại zero hoặc nhiều byte liên tục nhau từ đầu của dòng dữ liệu, nhưng có thể thêm một số dữ liệu vừa đến từ các datagram không theo đúng thứ tự. Nơi nhận luôn luôn đáp lời cho tiền tố dài nhất liên tục nhau của dòng dữ liệu mà nó đã nhận được một cách chính xác. Mỗi lời đáp xác định một số thứ tự có giá trị lớn hơn một so với vị trí byte cao nhất trong tiền tố liên tục nhau mà nó đã nhận. Như thế, nơi gửi sẽ nhận được thông tin phản hồi liên tục từ nơi nhận trong quá trình xử lý dòng dữ liệu. Chúng ta có thể tóm tắt ý tưởng quan trọng này như sau:

Acknowledgement  TCP xác định số thứ tự của byte kế tiếp mà nơi nhận chờ đợi để nhận.

Mô hình đáp lời TCP được gọi là tích luỹ bởi vì nó cho biết bao nhiêu dữ liệu của dòng dữ liệu đã được tích luỹ. Có cả ưu điểm và nhược điểm trong mô hình đáp lời tích luỹ. Một ưu điểm đó là việc thất lạc đáp lời không nhất thiết phải truyền lại dữ liệu. Nhược điểm chính là nơi gửi không nhận được thông tin về tất cả cuộc truyền thành công, mà chỉ là một vị trí trong dòng dữ liệu mà nơi nhận đã được nhận.

Để hiểu tại sao việc thiếu thông tin về tất cả các cuộc truyền thành công làm cho mô hình đáp lời tích luỹ kém hiệu quả, chúng ta hãy hình dung một cửa sổ trải ra trên 5000 byte bắt đầu tại vị trí 101 trong dòng dữ liệu, và giả sử nơi gửi đã truyền đi tất cả dữ liệu trong cửa sổ bằng cách gửi đi năm segment. Giả sử thêm rằng segment đầu tiên bị mất, nhưng các segment khác đến được nguyên vẹn. Khi mỗi segment đến được, nơi nhẩng lại một đáp lời, nhưng mỗi đáp lời mô tả byte thứ 101, byte liên tục cao nhất kế tiếp mà nó chờ đợi để nhận. Không có cách nào

để nơi nhận thông báo cho nơi gửi rằng hầu hết dữ liệu của cửa sổ hiện hành đã đến được.

Khi bộ đếm thời gian tại nơi gửi đã hết hạn, nơi gửi phải quyết định giữa hai mô hình mà đều có thể không hiệu quả. Nó có thể quyết định truyền lại một segment hoặc tất cả năm segment. Trong trường hợp này việc truyền lại tất cả năm segment sẽ không hiệu quả. Khi segment đầu tiên đến, nơi nhận sẽ có được tất cả dữ liệu trong cửa sổ, và sẽ đáp lời là 5101. Nếu nơi gửi thực hiện theo chuẩn được chấp nhận và chỉ truyền lại segment đầu tiên không được đáp lời, nó phải đợi lời đáp trước khi có thể biết dữ liệu gì và bao nhiêu phải gửi đi. Như thế, nó lại trở về giao thức đáp lời tích cực và đánh mất ưu điểm của việc có cửa sổ lớn.

### 7.2.11. Hết hạn (Timeout) và việc truyền lại

Một trong chức năng quan trọng và phức tạp nhất trong TCP thể hiện rõ qua cách nó xử lý việc chờ nhận hồi đáp hết hạn và truyền lại. Cũng giống những giao thức đáng tin cậy khác, TCP mong đợi máy đích gửi lại lời đáp bất cứ khi nào nó nhận thành công những byte mới từ dòng dữ liệu. Mỗi khi nó gửi một segment, TCP khởi động một bộ đếm thời gian và đợi lời đáp gửi về. Nếu bộ đếm thời gian hết hạn trước khi dữ liệu trong segment được đáp lời, TCP giả định rằng segment đã bị mất hay bị hỏng, nên truyền lại. Để hiểu được tại sao thuật giải truyền lại của TCP khác với thuật giải được sử dụng trong nhiều giao thức mạng, chúng ta cần nhớ lại rằng TCP nhằm mục đích để truyền dữ liệu trong môi trường rất đa dạng về truyền thông là Internet, một segment di chuyển giữa một cặp máy có thể đi qua một mạng đơn có độ trễ thấp (ví dụ, mạng cục bộ tốc độ cao), hay nó có thể di chuyển qua nhiều mạng trung gian thông qua các bộ định tuyến. Như thế, không thể nào biết được một tiêu chuẩn về thời gian (mức độ nhanh) mà các lời đáp được gửi về nguồn. Hơn nữa, độ trì hoãn tại mỗi bộ định tuyến còn tuỳ thuộc vào mức độ giao thông, vì vậy toàn bộ thời gian cần thiết để một segment di chuyển tới đích và để một đáp lời đi trở về nguồn vô cùng khác biệt tuỳ vào từng thời điểm.

Dung nạp độ trì hoãn khác nhau trên Internet bằng cách sử dụng một thuật giải truyền lại có tính năng hiệu chỉnh. Về bản chất, TCP kiểm tra hiệu suất của mỗi kết nối và điều chỉnh giá trị của bộ đếm thời gian một cách hợp lý. Khi hiệu suất của một kết nối thay đổi, TCP sẽ xem xét lại giá trị bộ đếm thời gian của nó (nghĩa là, nó cập nhật, TCP ghi nhận thời điểm khi mỗi segment được gửi đi và thời điểm khi nhận được lời đáp gửi trở về đối với dữ liệu trong segment đó. Dựa vào hai thời điểm này, TCP tính thời gian cách biệt được biết dưới tên mẫu thời gian đi trọn một vòng. Bất cứ khi nào nó có được một mẫu thời gian đi trọn một vòng mới, TCP điều chỉnh lại ý niệm của nó về thời gian trung bình đi trọn một vòng cho kết nối này. Thông thường, phần mềm TCP lưu trữ một ước lượng thời gian đi trọn một vòng, Round trip times (RTT), sử dụng làm giá trị trọng số trung

bình và lấy các mẫu thời gian đi trọn một vòng mới để từ từ thay đổi giá trị trọng số trung bình. Lấy ví dụ, khi tính giá trị trọng số trung bình mới, một kỹ thuật tính giá trị trung bình trước đây đã sử dụng một hằng số $\alpha$ làm hệ số, với $0 \leq \alpha \leq 1$, để tính giá trị trọng số trung bình cũ đối với mẫu thời gian đi trọn một vòng mới:

RTT mới = ($\alpha$ * RTT cũ) + ((1-$\alpha$) * RTT đi trọn 1 vòng mới nhất)

Việc chọn giá trị cho $\alpha$ gần với 1 làm cho giá trị trọng số trung bình không bị ảnh hưởng bởi những thay đổi trong một thời gian ngắn (ví dụ, khi chỉ có một segment bị trì hoãn lâu). Ngược lại, việc chọn giá trị cho $\alpha$ gần với 0 làm cho giá trị trung bình thay đổi tức thì theo những thay đổi của sự trì hoãn.

Khi nó gửi một gói dữ liệu, TCP tính giá trị cho bộ đếm thời gian theo một hàm của ước lượng thời gian đi trọn một vòng hiện tại. Các cài đặt trước đây của TCP sử dụng một hằng số $\beta$ ($\beta > 1$) làm hệ số, gán cho bộ đếm thời gian giá trị lớn hơn ước lượng thời gian đi trọn một vòng hiện tại:

Time = $\beta$ * RTT

Chọn giá trị cho $\beta$ là một công việc khó khăn. Một mặt, để nhanh chóng nhận biết việc mất gói dữ liệu, giá trị của bộ đếm thời gian phải gần với thời gian đi trọn một vòng hiện tại (nghĩa là, $\beta$ phải gần bằng 1). Việc nhận biết nhanh chóng gói dữ liệu bị mất sẽ hoàn thiện hiệu suất mạng vì TCP sẽ không phải đợi một thời gian dài một cách không cần thiết trước khi truyền lại dữ liệu mất. Mặt khác, nếu $\beta = 1$, TCP lại trở nên năng nổ quá mức chỉ một trì hoãn nhỏ cũng gây ra sự truyền lại không cần thiết, và như thế phí phạm băng thông của mạng. Đặc tả ban đầu đã đề nghị gán cho $\beta = 2$. Trong phần sau, chúng ta sẽ trình bày những nghiên cứu gần đây, đã đưa ra những kỹ thuật tốt hơn cho việc điều chỉnhbộ đếm thời gian.

Chúng ta có thể tóm tắt những ý tưởng vừa được trình bày:

Để dung nạp được những độ trì hoãn khác nhau gặp phải trong môi trường Internet, TCP sử dụng thuật giải truyền lại có tính năng hiệu chỉnh để theo dõi những độ trì hoãn trên mỗi kết nối và điều chỉnh giá trị bộ đếm thời gian của nó một cách tương ứng.

### 7.2.12.Xử lý khi gặp nghẽn mạng

Dường như rằng phần mềm TCP có thể được thiết kế để xem xét sự tương tác giữa hai đầu của một kết nối và sự trì hoãn thông tin liên lạc giữa hai đầu này. Tuy nhiên, trong thực tế TCP cũng phải đáp ứng với sự nghẽn mạch trong Internet. Sự nghẽn mạch là một trạng thái mà sự trì hoãn là rất cao gây ra bởi sự quá tải các datagram tại một hay nhiều điểm (ví dụ, tại các bộ định tuyến). Khi sự nghẽn mạch xảy ra, độ trì hoãn gia tăng và bộ định tuyến băt đầu xếp hàng các datagram cho đến khi nó có thể chuyển chúng đi. Chúng ta nhớ lại rằng máy có khả năng lưu trữ

giới hạn (bộ nhớ giới hạn) và các datagram phải cạnh tranh để vào đó. Trong trường hợp xấu nhất, tổng số datagram gửi đến bộ định tuyến (bị nghẽn mạch) tăng lên cho đến khi bộ định tuyến đạt đến khả năng lưu trữ tối đa (tràn bộ đệm) và bắt đầu huỷ bỏ những datagram đến sau.

Các điểm đầu cuối thường không biết các chi tiết về nơi đã xảy ra sự nghẽn mạch hoặc tại sao nó xảy ra. Đối với chúng, sự nghẽn mạch chỉ đơn giản có nghĩa là độ trì hoãn bị gia tăng. Tiếc thay, hầu hết các giao thức chuyên chở sử dụng bộ đếm thời gian và sự truyền lại, vì vậy chúng phản ứng lại với sự tăng độ trì hoãn bằng việc truyền lại các datagram. Việc truyền còn làm tồi tệ thêm sự nghẽn mạch chứ không giải quyết được vấn đề. Nếu không được kiểm tra, sự gia tăng giao thông sẽ gây ra sự tăng độ trì hoãn, mà như thế lại làm gia tăng giao thông, v.v.. cho đến khi mạng trở nên vô dụng. Tình huống này được gọi là sự sụp đổ do nghẽn mạch. Phần sau sẽ trình bày những kỹ thuật để tránh tối đa tình huống này.

*7.2.12.1.Kỹ thuật giảm thật nhanh*

Để tránh sự sụp đổ do nghẽn mạch, TCP phải giảm mật độ truyền khi xảy ra nghẽn mạch. Các bộ định tuyến theo dõi độ dài hàng đợi và sử dụng những kỹ thuật giống như làm nguội nguồn ICMP để thông báo các máy tính rằng đã xảy ra sự nghẽn mạch, nhưng các giao thức chuyên chở như TCP có thể giúp tránh khỏi sự nghẽn mạch bằng cách giảm cường độ truyền một cách tự động bất cứ khi nào xảy ra sự trì hoãn. Dĩ nhiên, các thuật giải để tránh sự nghẽn mạch phải được xây dựng cẩn thận bởi vì ngay cả trong những điều khiển hoạt động bình thường, một Internet cũng sẽ gặp những biến đổi lớn trong sự trì hoãn của thời gian đi trọn một vòng.

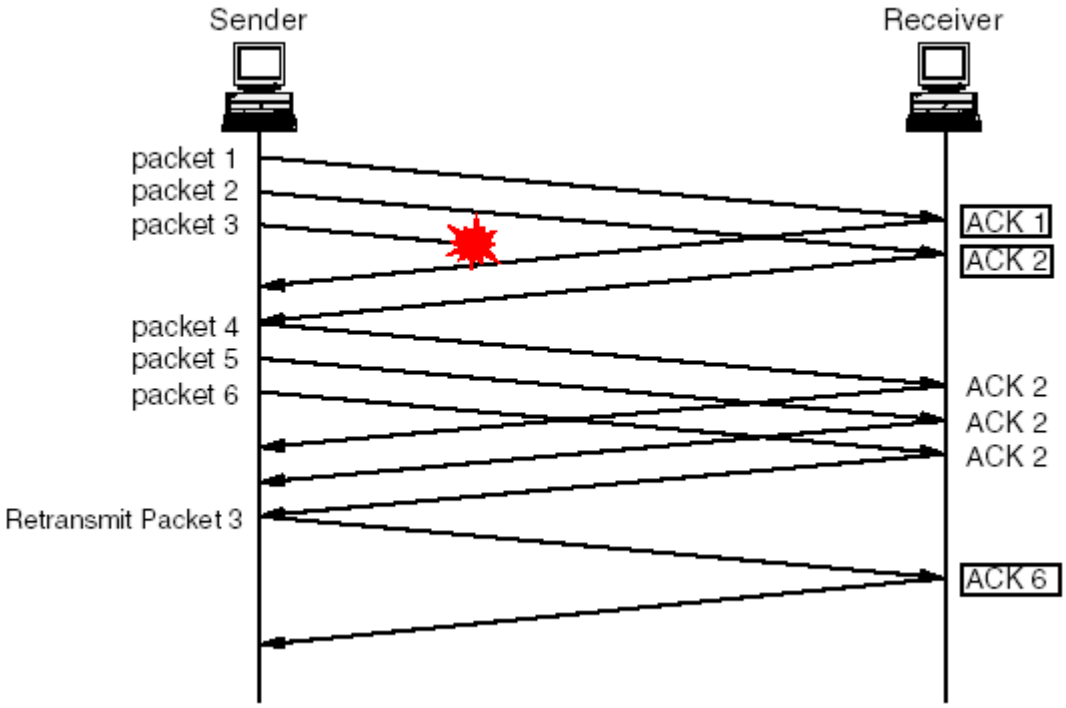
Để tránh sự sụp đổ do nghẽn mạch, chuẩn TCP hiện tại đề nghị sử dụng hai kỹ thuật: khởi đầu chậm và giảm thật nhanh theo cấp số nhân. Chúng có liên hệ với nhau và có thể được cài đặt một cách dễ dàng. Chúng ta đã nói rằng với mỗi kết nối, TCP phải nhớ kích thước của cửa sổ của nơi nhận (nghĩa là, kích thước vùng đệm được thông báo trong lời đáp). Để kiểm soát việc nghẽn mạch TCP duy trì một giá trị giới hạn thứ hai, được gọi là giới hạn cửa sổ nghẽn mạch hay đơn giản là cửa sổ nghẽn mạch, được sử dụng để giới hạn lượng dữ liệu ở mức ít hơn kích thước vùng đệm của nơi nhận khi xảy ra sự nghẽn mạch. Có nghĩa là, tại thời điểm bất kỳ, TCP duy trì một cửa sổ có kích thước:

Kích thước được phép = min (kích thước thông báo, kích thước cửa sổ nghẽn mạch)

Trong trạng thái ổn định của một kết nối không bị nghẽn mạch, cửa sổ nghẽn mạch có cùng kích thước với cửa sổ của nơi nhận. Việc giảm bớt kích thước cửa sổ nghẽn mạch sẽ làm giảm bớt giao thông mà TCP sẽ thực hiện trên kết nối. Để ước

lượng kích thước cửa sổ nghẽn mạch, TCP giả định rằng hầu hết các datagram bị mất là do sự nghẽn mạch gây nên và sử dụng chiến lược sau đây:

Tránh nghẽn mạch bằng cách giảm theo cấp số nhân: khi bị mất một segment, giảm kích thước cửa sổ nghẽn mạch đi một nửa (cho tới khi chỉ còn kích thước của một segment). Với những segment vẫn còn nằm trong cửa sổ được phép, nhượng bộ bằng cách gia tăng bộ đếm thời gian truyền lại theo hàm mũ.




*Hình 7.10: Kỹ thuật giảm thật nhanh của TCP*

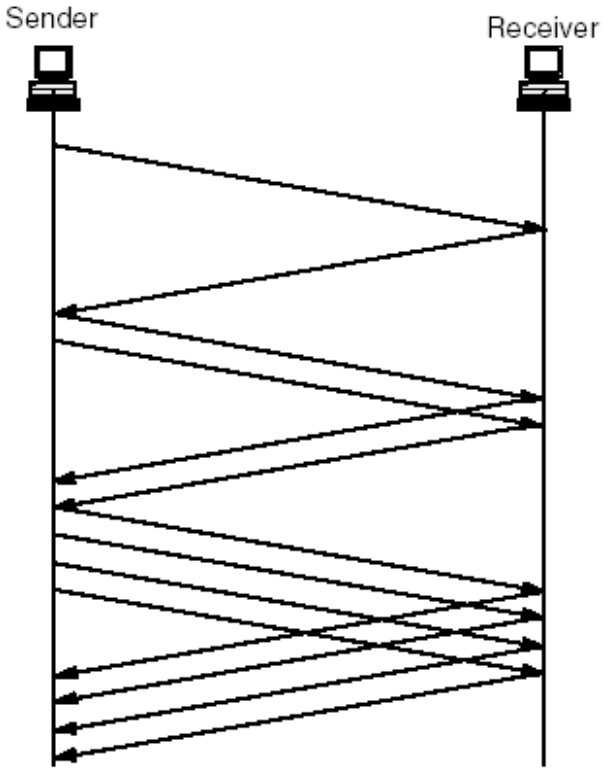
Bởi vì TCP giảm bớt kích thước cửa sổ nghẽn mạch đi một nửa cho mỗi lần bị mất, nó giảm kích thước cửa sổ đi theo một hàm mũ nếu còn tiếp tục mất. Nói một cách khác, nếu có hiện tượng nghẽn mạch, TCP sẽ giảm lượng giao thông theo hàm mũ và cả tốc độ truyền lại cũng theo hàm mũ. Nếu vẫn tiếp tục bị mất, thì cuối cùng TCP giới hạn dung lượng truyền xuống chỉ còn là một datagram và tiếp tục nhân đôi giá trị bộ đếm thời gian trước khi truyền lại. Ý tưởng của việc này nhanh chóng giảm bớt thật đáng kể lượng giao thông để cho phép bộ đếm thời gian có đủ thời gian để xử lý các datagram đã nằm trong hàng đợi.

*7.2.12.2. Kỹ thuật khởi đầu chậm*

Làm thế nào mà TCP có thể phục hồi lại sau khi không còn nghẽn mạch? Ta có thể nghĩ rằng TCP sẽ phục hồi việc giảm theo cấp số nhân và nhân đôi kích thước cửa sổ nghẽn mạch khi bắt đầu truyền dữ liệu. Tuy nhiên, làm như vậy sẽ tạo ra một hệ thống không ổn định, bị giao động rất lớn giữa trạng thái nghẽn mạch và không có giao thông. Thay vì thế, TCP sử dụng một kỹ thuật được gọi là khởi động

chậm (thuật ngữ khởi động chậm là do Jonh Nagle đưa ra; ban đầu kỹ thuật này được gọi là khởi động mềm) để gia tăng từ từ việc truyền dữ liệu:

Phục hồi theo cách khởi động chậm (thêm vào từ từ): bất cứ khi nào khởi động giao dịch (truyền dữ liệu) trên một kết nối mới hoặc gia tăng lượng giao dịch sau một giai đoạn bị nghẽn mạch, hoặc bắt đầu với kích thước cửa sổ nghẽn mạch bằng một segment và gia tăng kích thước cửa sổ nghẽn mạch thêm một segment sau mỗi lần nhận được một lời hồi đáp.




*Hình 7.11: Kỹ thuật bắt đầu chậm của TCP*

Việc khởi động chậm sẽ tránh cho Internet bị tràn bởi lượng giao dịch thêm vào đột ngột sau khi hết nghẽn mạch hay khi một kết nối mới đột nhiên khởi động.

Thuật ngữ khởi động chậm có thể không tuyệt đối chính xác bởi vì dưới những điều kiện lý tưởng, việc khởi động không phải là chậm. TCP bắt đầu với kích thước cửa sổ nghẽn mạch là 1, gửi một segment đầu tiên và đợi. Khi nhận được hiệu trả lời đầu tiên, nó tăng kích thước cửa sổ nghẽn mạch thành 2, nó gửi 2 segment và đợi. Khi nhận được hai tín hiệu trả lời kế tiếp, mỗi cái tăng kích thước cửa sổ nghẽn mạch thêm 1 nên TCP có thể gửi 4 segment. Những tín hiệu trả lời cho các segment này sẽ tăng kích thước cửa sổ nghẽn mạch thành 8. Chỉ trong 4 vòng TCP có thể gửi đi 16 (N) segment, và đây là kích thước giới hạn của nơi nhận. Ngay cả đối với những cửa sổ thật lớn chỉ cần sau $\log_2 N$ vòng là TCP có thể gửi đi N segment. Để tránh việc tăng kích thước cửa sổ quá nhanh và gây thêm nghẽn mạch, TCP bổ sung thêm một ràng buộc. Khi cửa sổ nghẽn mạch đã đạt đến một nửa kích thước ban đầu của nó so với trước khi xảy ra nghẽn mạch, TCP

chuyển sang giai đoạn tránh nghẽn mạch và làm chậm lại mức độ tăng. Trong suốt giai đoạn tránh sự nghẽn mạch, nó chỉ tăng kích thước cửa sổ nghẽn mạch thêm 1 sau khi tất cả các segment trong cửa sổ đã nhận được tín hiệu trả lời.

Kết hợp các kỹ thuật này lại với nhau: tăng chậm khi khởi động, giảm theo cấp số nhân khi nghẽn mạng, theo dõi sự biến đổi của thời gian nhận hồi đáp và tăng bộ đếm thời gian theo hàm mũ, sẽ tăng hiệu suất của TCP thật đáng kể mà không phải bổ sung thêm những phép tính toán phức tạp vào phần mềm giao thức. Các phiên bản của TCP sử dụng các kỹ thuật này đã tăng hiệu suất từ 2 đến 10 lần so với các phiên bản trước.

*7.2.12.3. Kỹ thuật cắt bớt phần dưới khi nghẽn mạng*

Như chúng ta đã biết, các giao thức thông tin liên lạc thường được chia thành các mức để giúp cho những người thiết kế tại mỗi thời điểm chỉ tập trung vào một vấn đề. Việc tách rời các chức năng tương đối độc lập là cần thiết và rất hữu dụng, vì một mức có thể bị thay đổi mà không ảnh hưởng đến những mức khác. Điều đó có nghĩa là các mức hoạt động một cách riêng biệt. Lấy ví dụ, vì TCP hoạt động theo nguyên tắc end to end, TCP vẫn giữ nguyên không đổi khi đường đi giữa hai đầu thay đổi (ví dụ, định tuyến thay đổi hoặc có thêm các bộ định tuyến). Tuy nhiên, việc tách biệt các mức cũng giới hạn sự trao đổi thông tin liên lạc giữa các lớp. Cụ thể là, mặc dù TCP ở nguồn ban đầu tương tác với TCP tại đích cuối cùng, nó không thể tương tác với các thành phần ở mức thấp hơn dọc theo con đường giữa hai đầu. Như thế, cả TCP nơi nhận lẫn TCP nơi gửi đều không nhận được các báo cáo về những trạng thái và điều kiện trên mạng trung gian, cũng như là cả hai đầu đều không thông báo được thông tin về trạng thái của chúng cho các mức thấp hơn dọc theo con đường trước khi truyền dữ liệu.

Những nhà nghiên cứu đã quan sát được rằng việc thiếu thông tin liên lạc giữa các mức có nghĩa là những quyết định về chính sách và cài đặt tại một mức có thể có ảnh hưởng rất lớn đến hiệu quả của những mức cao hơn. Trong trường hợp của TCP, những chiến lược mà các bộ định tuyến sử dụng để xử lý IP datagram có thể ảnh hưởng đáng kể đến cả hiệu quả của một kết nối TCP cũng như lan truyền ra tất cả các kết nối. Nếu bộ định tuyến trì hoãn một vài IP datagram lâu hơn những cái khác, TCP sẽ gia tăng giá trị của bộ đếm thời gian cho lần truyền lại. Nếu độ trì hoãn vượt quá thời gian hết hạn của lần truyền lại, TCP sẽ giả định rằng đã xảy ra sự nghẽn mạch. Như thế, mặc dù mỗi lớp được định nghĩa một cách độc lập, những nhà nghiên cứu cố gắng để có được những cơ chế và sự cài đặt để làm việc tốt với những giao thức trong các mức khác.

Sự giao tiếp quan trọng nhất giữa các chính sách cài đặt IP và TCP xuất hiện khi bộ định tuyến bị quá tải và loại bỏ bớt những datagram phía sau. Bởi vì bộ định tuyến lưu trữ các datagram gửi đến trong một hàng đợi của bộ đếm nhớ cho đến

khi nó có thể được xử lý, chính sách này phải tập trung vào việc quản lý hàng đợi. Khi các datagram được gửi đến nhanh hơn là chúng được chuyển đi, thì hàng đợi sẽ dài ra; khi các datagram được gửi đến chậm hơn là chúng được chuyển đi, hàng đợi được thu ngắn lại. Tuy nhiên, bởi vì bộ nhớ là hữu hạn, hàng đợi không thể dài ra vô hạn được. Phần mềm của bộ định tuyến trước đây đã sử dụng chính sách "cắt bớt phần đuôi" để quản lý việc hàng đợi bị tràn:

Chính sách cắt bớt phần đuôi của các bộ định tuyến: nếu hàng đợi của dữ liệu đến bị đầy, bộ định tuyến sẽ huỷ bỏ những datagram được gửi đến tiếp theo sau.

Tên của nó, "cắt bớt phần đuôi", phản ánh ảnh hưởng chính sách này đối với các datagram gửi tiếp theo sau. Một khi hàng đợi bị đầy, bộ định tuyến dẽ huỷ bỏ tất cả những datagram sau đó. Nghĩa là, bộ định tuyến huỷ bỏ "phần đuôi" của chuỗi dữ liệu.

Việc cắt bớt phần đuôi có ảnh hưởng đáng kể với TCP, trong trường hợp đơn giản khi các datagram di chuyển qua bộ định tuyến mang theo các segment của chỉ một kết nối TCP, việc mất này làm cho TCP đi vào trạng thái khởi động chậm, nghĩa là giảm bớt tốc độ truyền cho tới khi TCP bắt đầu nhận được lời đáp và gia tăng kích thước cửa sổ nghẽn mạch. Tuy nhiên, một vấn đề nghiêm trọng hơn có thể xảy ra, khi các datagram di chuyển qua bộ định tuyến mạng theo các segment của nhiều kết nối TCP việc "cắt bớt phần đuôi" có thể ảnh hưởng đến toàn bộ Internet. Để thấy được tại sao, cần lưu ý rằng các datagram thường được mutiplex, trong đó các datagram kế tiếp nhau có thể đến từ nhiều nguồn khác nhau. Như thế, chính sách "cắt bớt phần đuôi" có thể gây ra khả năng bộ định tuyến sẽ huỷ bỏ một segment từ nhiều kết nối hơn là nhiều segment của một kết nối. Sự mất datagram cùng một lúc này sẽ làm cho nhiều liên kết TCP cùng chuyển sang trạng thái khởi động chậm, khi đó nó sẽ ảnh hưởng đến toàn bộ mạng trung gian.

*7.2.12.4. Kỹ thuật huỷ bỏ sớm ngẫu nhiên (RED)*

Làm thế nào để bộ định tuyến tránh gây ảnh hưởng đến toàn bộ mạng trung gian (Internet). Câu trả lời chính là một mô hình thông minh hơn trong việc hủy bỏ các segment để tránh việc "cắt bớt phần đuôi" bất cứ khi nào có thể được. Kỹ thuật này được đặt tên là huỷ bỏ sớm ngẫu nhiên (Random Early Discard), mô hình này thường được gọi tắt là RED. Bộ định tuyến cài đặt RED sử dụng hai giá trị chặn trên và dưới để đánh dấu các vị trí trong hàng đợi: Tmin và Tmax. Một cách tổng quát, hoạt động của RED có thể được mô tả bởi ba quy tắc:

* Nếu hiện tại, hàng đợi chứa ít hơn Tmin datagram, thêm datagram mới vào hàng đợi.

* Nếu hàng đợi chứa nhiều hơn Tmax datagram, huỷ bỏ những datagram mới.

* Nếu hàng đợi chứa trong khoảng Tmin và Tmax datagram, huỷ bỏ datagram một cách ngẫu nhiên tuỳ theo 1 hàm xác suất P.

Tính ngẫu nhiên của RED có nghĩa là thay vì đợi đến khi hàng đợi bị đầy và rồi buộc nhiều kết nối TCP phải chuyển qua trạng thái khởi động chậm, bộ định tuyến huỷ bỏ các datagram một cách ngẫu nhiên và theo sự gia tăng của sự nghẽn mạch. Chng ta có thể tóm tắt:

Chính sách của RED trong các bộ định tuyến: nếu hàng đợi của dữ liệu đến bị đầy khi có một datagram gửi đến thì sẽ huỷ bỏ datagram; nếu hàng đợi của dữ liệu đến chưa đầy nhưng đã vượt qua kích thước giới hạn trên, để tránh làm ảnh hưởng đến toàn bộ Internet thì phải huỷ bỏ datagram một cách ngẫu nhiên theo một hàm xác suất P.

Điểm mấu chốt để cho RED hoạt động tốt chính là việc chọn các giá trị chặn Tmin và Tmax, và hàm xác suất P. Tmin phải đủ lớn để đảm bảo rằng đường liên kết để gửi dữ liệu đi được sử dụng với hiệu suất cao. Hơn nữa, bởi vì RED hoạt động giống như "cắt bớt phần đuôi" khi kích thước vượt quá Tmax, nên Tmax phải lớn hơn Tmin một lượng cỡ vào khoảng sự gia tăng kích thước hàng đợi trong suốt quá trình đi trọn một vòng (ví dụ, Tmax phải ít nhất lớn gấp đôi Tmin). Nếu không thì RED cũng gây ra cùng ảnh hưởng như là "cắt bớt phần đuôi".

Việc tính hàm xác suất P là khía cạnh phức tạp nhất của RED. Thay vì sử dụng một hằng số, một giá trị mới của P được tính cho mỗi datagram; giá trị này phụ thuộc vào mối quan hệ giữa kích thước hàng đợi hiện tại và các giá trị giới hạn trên và giới hạn dưới. Để hiểu được mô hình này, hãy lưu ý rằng tất cả tiến trình RED có thể nhìn dưới góc độ xác suất. Khi kích thước hàng đợi nhỏ hơn Tmin, RED không huỷ bỏ bất kỳ datagram nào, thì cho xác suất huỷ bỏ là 0. Tương tự, khi kích thước hàng đợi nhỏ hơn Tmax, RED huỷ bỏ tất cả các datagram đến sau, thì cho xác suất huỷ bỏ là 1. Đối với những giá trị trung gian khác của kích thước hàng đợi (nghĩa là những giá trị giữa Tmin và Tmax), xác suất có thể thay đổi từ 0 đến 1 một cách tuyến tính.

Mặc dù mô hình tuyến tính hình thành nên cơ sở của phép tính xác suất cho RED, nhưng cần phải có những hiệu chỉnh để tránh tình trạng phản ứng "quá vội". Sở dĩ cần có những thay đổi này là vì giao thông trên mạng là theo từng "đợt", và gây ra những thay đổi quá nhanh của hàng tuyến tính đơn giản, những datagram đến sau trong mỗi đợt sẽ có xác suất cao bị loại bỏ (bởi vì chúng đến khi hàng đợi đã có nhiều datagram). Tuy nhiên, bộ định tuyến không nên huỷ bỏ những datagram này một cách không cần thiết, bởi vì làm như thế sẽ tác động xấu đến hiệu suất của TCP. Nếu gặp một đợt (các datagram) ngắn, ít có khả năng bị loại bớt

datagram bởi vì hàng đợi chưa bị đầy. Dĩ nhiên, RED không thể tạm hoãn việc huỷ bỏ vô thời hạn bởi vì một đợt dài sẽ làm đầy hàng đợi, kết quả chẳng khác nào chiến lược "cắt bớt phần đuôi' và sẽ gây ra ảnh hưởng đến toàn bộ Internet.

Làm thế nào RED có thể gán một xác suất huỷ bỏ cao hơn khi hàng đợi bị đầy mà không phải huỷ bỏ datagram của mỗi đợt? Câu trả lời thuộc về kỹ thuật được vay mượn từ TCP: thay vì sử dụng kích thước thật của hàng đợi tại thời điểm đó, RED tính kích thước hàng đợi trung bình có trọng số avg và sử dụng kích thước trung bình này để xác định xác suất. Giá trị của avg mới được cập nhật mỗi khi có datagram gửi đến, theo phương trình sau:

$$Avg\_mới = (1 - \chi) * Avg\_cũ + \chi * kích\_thước\_hàng\_đợi\_hiện\_tại$$

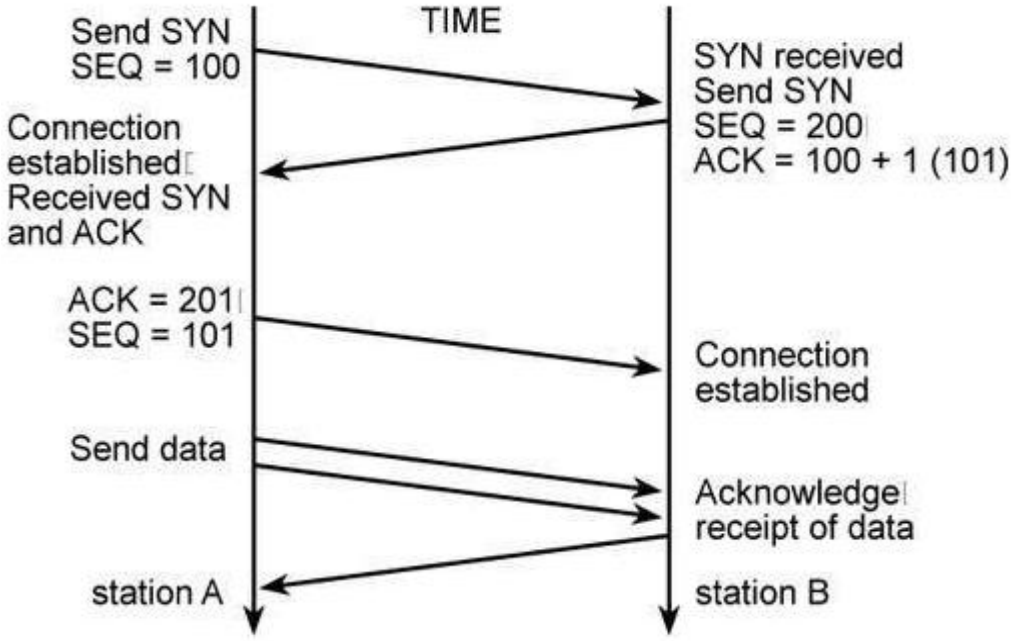
Với $\chi$ có giá trị trong khoảng từ 0 đến 1. Nếu giá trị của $\chi$ đủ nhỏ, thì giá trị trung bình sẽ có khuynh hướng ít thay dổi, và sẽ ít bị ảnh hưởng đối với những đợt ngắn (ví dụ, giá trị là 0,002).

Cùng với những phương trình để xác định $\chi$, RED còn bao gồm những chi tiết khác mà chúng ta đã phớt lờ đi. Lấy ví dụ, các phép tính có thể thực hiện vô cùng hiệu quả với việc chọn các hằng số là luỹ thừa của hai và sử dụng các phép tính số học trên số nguyên. Một chi tiết quan trọng khác liên quan đến việc đo độ dài của hàng đợi, mà có ảnh hưởng đến cả việc tính RED cũng như tác động của chúng đến TCP. Cụ thể, bởi vì thời gian cần để chuyển datagram đi là tỉ lệ thuận với kích thước của nó, nên sẽ hợp lý hơn khi đo hàng đợi theo byte thay vì theo datagram; làm như thế sẽ chỉ đòi hỏi những thay đổi nhỏ giá trị $\chi$. Việc đo kích thước hàng đợi theo byte ảnh hưởng đến kiểu dữ liệu bị loại bỏ bởi vì nó tính xác suất huỷ bỏ tỷ lệ thuận với tổng số dữ liệu mà có chuyển đi thay vì số lượng của segment. Những datagram nhỏ (ví dụ, những datagram chuyển tải dữ liệu cho việc kết nối từ xa) sẽ có xác suất bị loại bỏ thấp hơn những datagram lớn (ví dụ, những datagram chuyển tải dữ liệu của dịch vụ truyền tập tin). Một kết quả tích cực của việc sử dụng kích thước là khi những lời đáp di chuyển trên một con đường bị nghẽn mạch, chúng sẽ có xác suất bị loại bỏ thấp hơn. Và kết quả là, nếu một segment dữ liệu (lớn) đến được, thì TCP nơi gửi sẽ nhận được lời đáp và sẽ tránh được việc truyền lại không cần thiết. Cả việc phân tích lẫn sự nghẽn mạch, tránh được ảnh hưởng đến toàn bộ Internet gây ra bởi "cắt bớt phần đuôi", và cho phép gửi đi những đợt dữ liệu ngắn mà không bị bỏ những datagram. Hiện tại IETF đã khuyến khích cài đặt RED trong các bộ định tuyến.

## 7.3. Thiết lập, hủy bỏ, khởi tạo lại kết nối TCP

### 7.3.1. Thiết lập một kết nối TCP

Để thiết lập một kết nối, TCP sử dụng mô hình bắt tay ba bước. Hình 7.12 trình bày tiến trình bắt tay trong trường hợp đơn giản nhất.




*Hình 7.12: Quá trình bắt tay 3 bước kết nối TCP*

Segment đầu tiên của tiến trình bắt tay có thể được xác định bởi bit SYN của nó được lập (SYN là chữ viết tắt của Synchronization) trong vùng "CODE". Trong tín hiệu thứ hai, cả hai bit SYN và ACK đều được lập, để chỉ ra rằng có dữ liệu segment cần truyền, SYN đầu tiên và tiếp tục việc bắt tay. Tín hiệu bắt tay cuối cùng chỉ là một lời đáp và đơn giản dùng để thông báo cho đích rằng cả hai bên cùng đồng ý kết nối.

Thông thường, phần mềm TCP trên một máy đợi một cách thụ động cho việc bắt tay còn phần mềm TCP trên máy truyền thì khởi tạo nó. Tuy nhiên, việc bắt tay được thiết kế một cách cẩn thận để có thể làm việc ngay trong trường hợp cả hai máy đều thực hiện kết nối cùng một lúc. Như thế, một kết nối có thể được thiết lập từ mỗi bên hoặc đồng thời từ hai bên. Một khi kết nối được thiết lập, dữ liệu có thể di chuyển theo hai chiều. Không phân biệt chính hay phụ.

Việc bắt tay ba bước là điều kiện cần và đủ để có sự đồng bộ chính xác giữa hai đầu kết nối. Để hiểu được tại sao, chúng ta hãy nhớ lại rằng TCP xây dựng trên dịch vụ chuyển phát không tin cậy, vì thế các thông điệp có thể bị mất. Vấn đề khó khăn xuất hiện nếu các yêu cầu ban đầu và yêu cầu được truyền lại đến trong khi kết nối đang được thiết lập, hay các yêu cầu được truyền lại bị trì hoãn cho đến sau khi một kết nối đã được thiết lập, đang sử dụng và đã kết thúc. Việc bắt tay ba bước sẽ giải quyết các vấn đề này (cùngvới quy tắc TCP bỏ qua những yêu cầu tiếp theo cho một kết nối sau khi kết nối đã được thiết lập).

Khởi tạo số thứ tự

Việc bắt tay ba bước đạt được hai chức năng quan trọng, nó bảo đảm rằng cả hai phía đều sẵn sàng để truyền dữ liệu (và chúng ta cũng biết cả hai phía đều sẵn sàng), nó cho phép cả hai phía đồng ý với nhau về việc khởi xướng số thứ tự. Các số thứ tự này được gửi đi và lưu lại trong quá trình bắt tay. Mỗi máy phải chọn ngẫu nhiên số thứ tự ban đầu mà sẽ sử dụng để xác định các bytes trong dòng dữ liệu nó đang gửi đi. các số thứ tự có thể không luôn bắt đầu với cùng một giá trị. Cụ thể, TCP không thể đơn giản chỉ chọn số thứ tự 1 mỗi khi nó tạo một kết nối. Dĩ nhiên, điều quan trọng là cả hai phía đều đồng ý với con số khởi đầu, để cho số lượng byte được sử dụng trong tín hiệu trả lời giống với số được sử dụng trong segment dữ liệu.

Để thấy được các máy tính có thể thống nhất với nhau về các số thứ tự cho hai dòng dữ liệu chỉ với ba thông điệp (sau ba bước), chúng ta hãy nhớ lại rằng mỗi segment bao gồm cả vùng số thứ tự và vùng lời đáp. Máy tính khởi xướng việc bắt tay, gọi là máy A, gửi đi số thứ tự khởi đầu của nó, x, trong vùng số thứ tự của segment SYN đầu tiên trong quá trình bắt tay ba bước. Máy tính thứ hai, B, nhận SYN, ghi nhận số thứ tự, và trả lời bằng cách gửi đi số thứ tự khởi đầu của nó trong vùng số thứ tự cùng lời đáp để xác định rằng B chờ đợi byte thứ x+1. Trong thông điệp cuối cùng của việc bắt tay, A đáp lại việc nhận từ B tất cả các byte kể từ y. Trong mọi trường hợp, những lời đáp đều tuân theo quy ước về việc sử dụng con số của byte kế tiếp.

Chúng ta đã mô tả cách mà TCP thường thực hiện việc bắt tay ba bước bằng việc trao đổi các segment chứa đựng lượng thông tin tối thiểu. Bởi vì cách thiết kế giao thức, cũng có thể gửi dữ liệu đi cùng với các số thứ tự khởi đầu trong các segment bắt tay. Trong những trường hợp đó, phần mềm TCP phải giữ lại dữ liệu cho đến khi hoàn tất việc bắt tay. Một khi kết nối đã được thiết lập, phần mềm TCP có thể giải phóng dữ liệu đã được giữ trước đây và nhanh chóng chuyển phát nó tới chương trình ứng dụng đang đợi.

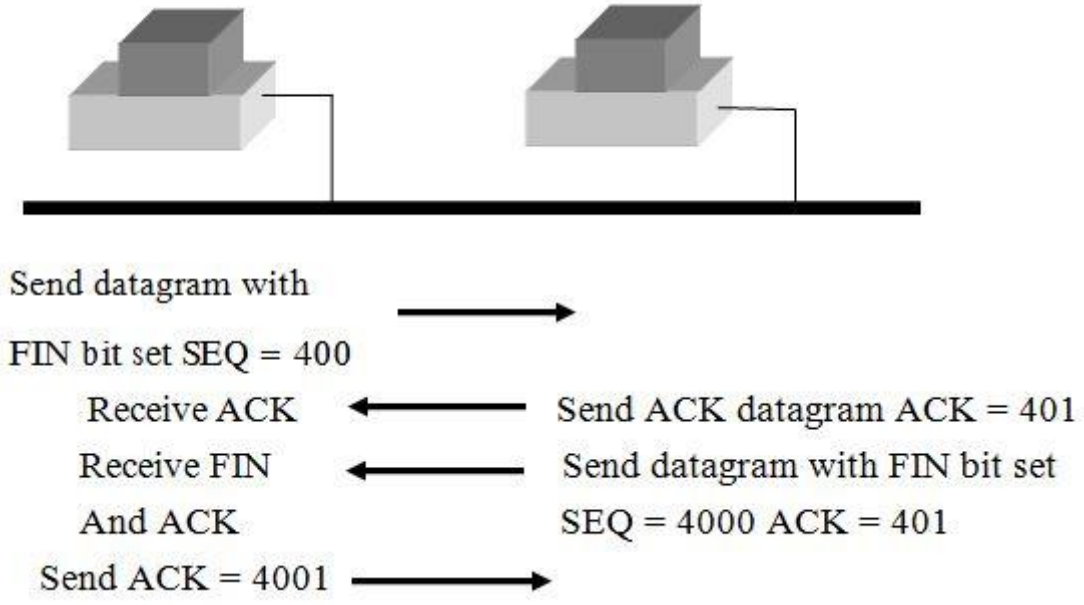
### 7.3.2. Đóng lại một kết nối TCP

Hai chương trình sử dụng TCP để trình bày liên lạc có thể kết thúc cuộc trao đổi một cách êm đẹp bằng cách sử dụng thao tác "đóng". Về kỹ thuật, TCP lại sử dụng kiểu bắt tay ba bước (có chút sửa đổi) để đóng các kết nối. Chúng ta hãy nhớ lại rằng các kết nối TCP là hai chiều (dữ liệu có thể lưu chuyển đồng thời theo hai chiều) và chúng ta xem như chúng chứa hai dòng truyền dữ liệu độc lập, mỗi dòng theo một chiều. Khi một chương trình ứng dụng thông báo cho TCP rằng nó không còn dữ liệu gửi đi nữa, TCP sẽ đóng kết nối theo một chiều. Để đóng một nửa của kết nối, và rồi gửi đi một segment có bit FIN được lập. TCP nơi nhận dữ liệu segment FIN này và thông báo cho chương trình ứng dụng tại phía nó rằng không còn dữ liệu nào nữa (ví dụ, sử dụng cơ chế end of file của hệ điều hành).

Một khi kết nối đã được đóng lại theo một chiều nào đó, TCP sẽ từ chối nhận thêm dữ liệu cho chiều đó. Trong lúc đó, dữ liệu vẫn có thể tiếp tục di chuyển theo chiều ngược lại cho đến khi nơi gửi đóng nó lại. Dĩ nhiên, những lời đáp vẫn tiếp tục di chuyển trở lại nơi gửi ngay cả sau khi kết nối đã đóng lại. Khi cả hai chiều đã được đóng lại, phần mềm TCP tại mỗi bên sẽ xoá bỏ những ghi nhận của nó về kết nối này.

Những chi tiết về việc đóng một kết nối có nhiều điểm tinh tế hơn nhiều so với những gì vừa trình bày bởi vì TCP sử dụng một kiểu bắt tay ba bước đã được sửa đổi để đóng lại một kết nối. Hình 7.13 trình bày các thủ tục này.

Sự khác biệt giữa hai bắt tay ba bước được sử dụng để thiết lập kết nối và đóng kết nối xuất hiện sau khi một máy nhận được segment khởi động FIN. Thay vì phát sinh ngay tức khắc segment FIN thứ hai, TCP gửi đi một lời đáp và sau đó thông báo cho chương trình ứng dụng (của lời yêu cầu) để chấm dứt. Việc thông báo cho chương trình ứng dụng và lấy được sự đáp lại có thể mất một khoảng thời gian đáng kể (ví dụ, nó có thể liên quan đến sự giao tiếp với con người). Lời đáp sẽ ngăn chặn việc truyền lại segment FIN khởi động trong suốt quá trình chờ đợi. Cuối cùng, khi chương trình ứng dụng chỉ thị cho TCP để chấm dứt hoàn toàn kết nối, TCP gửi đi segment FIN thứ hai và phía bên kia đáp lại bằng thông điệp thứ ba, chính là một lời đáp ACK.




*Hình 7.13: Kết thúc kết nối TCP*

### 7.3.3. Hủy kết nối TCP

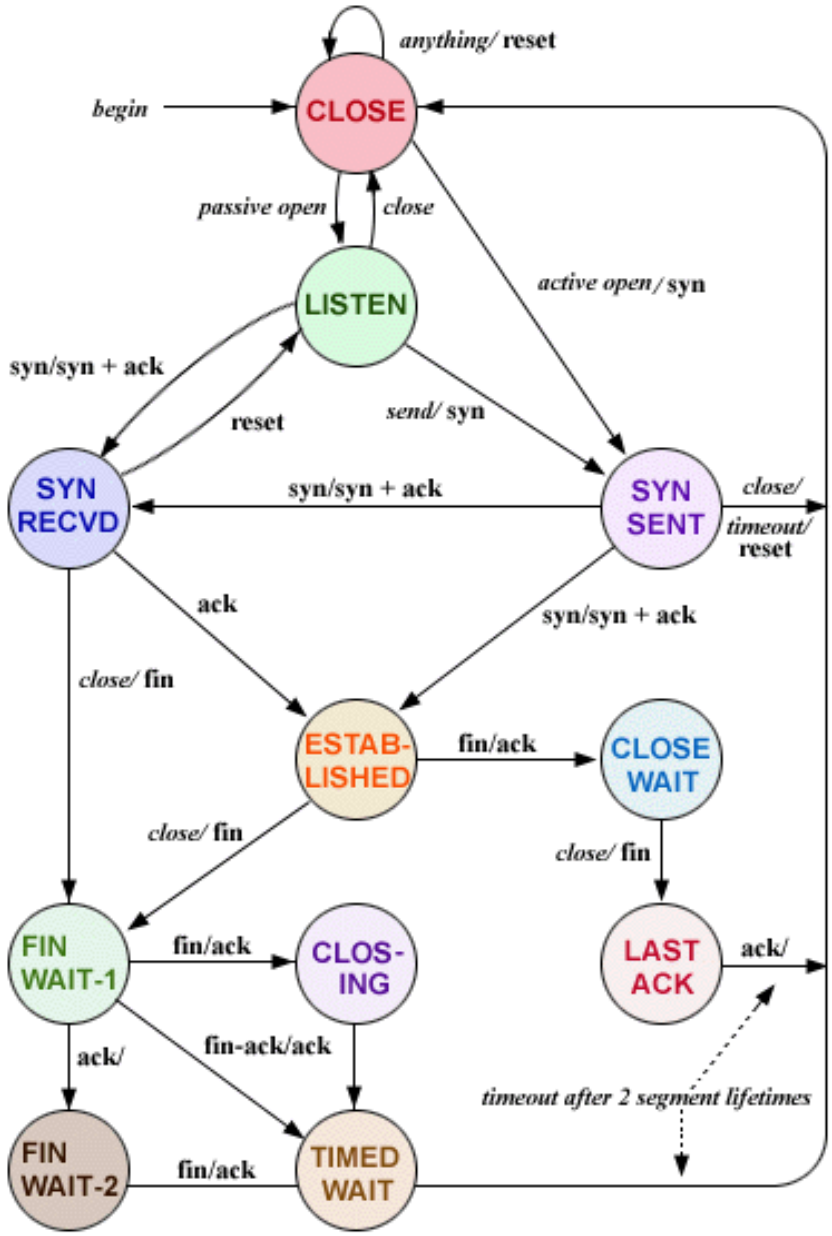
Thông thường, một chương trình ứng dụng sử dụng thao tác đóng để kết thúc một kết nối khi nó đã sử dụng chúng xong. Như thế, việc đóng các kết nối được xem như một phần thông thường khi sử dụng, tương tự như việc đóng tập tin. Đôi

khi, xuất hiện điều kiện bất thường làm cho một chương trình hay phần mềm mạng phá vỡ đi một kết nối. Khi đó TCP phải cung cấp một chức năng reset đối với những trường hợp ngắt kết nối không bình thường này.

Để reset một kết nối, một bên sẽ khởi xướng việc kết thúc bằng cách gửi đi một segment với bit RST trong vùng CODE được lập. Bên kia lập tức đáp lại với segment reset bằng việc huỷ bỏ kết nối. TCP cũng thông báo cho chương trình ứng dụng rằng đã xảy ra việc reset. Reset là thao tác huỷ bỏ ngay tức thời, có nghĩa là việc truyền theo cả hai chiều phải kết thúc ngay lập tức, và các tài nguyên như vùng đệm được giải phóng.

### 7.3.4. Máy trạng thái TCP

Giống như hầu hết các giao thức, hoạt động của TCP có thể được giải thích tốt nhất thông qua một mô hình lý thuyết được gọi là máy trạng thái hữu hạn (finite state machine). Hình 7.13 trình bày máy trạng thái hữu hạn TCP, với vòng tròn thể hiện trạng thái và mũi tên thể hiện việc chuyển đổi từ trạng thái này sang trạng thái khác. Nhãn ghi trên mỗi trạng thái cho thấy những gì TCP nhận được để gây nên việc chuyển trạng thái và những gì nó gửi đi để đáp lại. Lấy ví dụ, phần mềm TCP tại mỗi đầu được bắt đầu ở trạng thái CLOSE. Các chương trình ứng dụng phải đưa ra lệnh "Passive open" (để đợi một kết nối từ máy khác), hoặc là lệnh "active open" (để khởi xướng một kết nối). Lệnh "active open" làm cho một trạng thái chuyển từ CLOSE sang SYN SENT. Khi TCP tuân theo việc chuyển trạng thái, nó gửi ra một segment SYN. Khi đầu kia trả về một segment có chứa SYN cùng với ACK, TCP chuyển sang trạng thái ESTABLISHED và bắt đầu truyền dữ liệu đi.




*Hình 7.14: Máy trạng thái TCP*

Trạng thái TIME WAIT cho ta biết cách mà TCP xử lý một số vấn đề xảy ra với việc chuyển phát không có độ tin cậy. TCP duy trì một khái niệm maximum segment lifetime (MSL), thời gian tối đa một segment cũ có thể còn sống trong một Internet. Để tránh tình trạng có nhiều segment từ các kết nối trước ảnh hưởng đến kết nối hiện tại, TCP chuyển sang trạng thái TIME WAIT sau khi đóng lại một kết nối. Nó vẫn giữ nguyên trạng thái đó trong khoảng thời gian gấp đôi thời gian sống tối đa của một segment trước khi xoá bỏ những ghi nhận của nó về kết nối này. Nếu có bất kỳ segment trùng lặp đến được kết nối này trong thời hạn này. TCP sẽ từ chối xử lý chúng. Tuy nhiên, để xử lý các trường hợp khi mà lời đáp cuối cùng bị mất, TCP sẽ đáp lời cho những segment hợp lệ và khởi động lại bộ đếm thời gian. Bởi vì bộ đếm thời gian cho phép TCP phân biệt những kết nối cũ

với kết nối mới, nó ngăn ngừa việc TCP đáp lại lệnh RST nếu đầu kia truyền lại một yêu cầu FIN.

### 7.3.5. Bắt buộc truyền dữ liệu

Chúng ta đã nói rằng TCP được tự do phân chia dòng dữ liệu thành các segment để truyền đi mà không xét đến kích thước của đơn vị truyền mà chương trình ứng dụng sử dụng. Ưu điểm lớn nhất khi cho phép TCP phân chia đó là sự hiệu quả. Nó có thể tích lũy đủ lượng byte trong vùng đệm để hình thành nên những segment có độ dài hợp lý, làm giảm bớt việc truyền những dữ liệu "quản lý" trong phần đầu khi segment chỉ chứa có vài byte dữ liệu.

Mặc dù sử dụng vùng đệm hoàn thiện được hiệu suất của mạng, nó có thể ảnh hưởng đến một số ứng dụng. Chúng ta hãy thử xem xét việc sử dụng kết nối TCP để truyền những ký tự (character) từ một trạm làm việc tới một máy ở xa. Người sử dụng mong muốn có lời đáp tức thời cho mọi ký tự nhập vào. Nếu TCP tại nơi gửi tạm thời lưu dữ liệu trong vùng đệm, lời đáp có thể bị trì hoãn, có thể lên đến cả trăm ký tự. Tương tự, bởi vì TCP tại nơi nhận có thể tạm thời lưu dữ liệu trong vùng đệm trước khi gửi chúng đến cho chương trình ứng dụng, việc bắt buộc nơi gửi truyền dữ liệu đi có thể chưa đủ để bảo đảm cho việc phát chuyển.

Để đáp ứng được với những ứng dụng tương tác như trên, TCP cung cấp một thao tác push mà chương trình ứng dụng có thể sử dụng để ép buộc chuyển phát các byte hiện đang có trong dòng dữ liệu mà không phải đợi đến lúc đầy vùng đệm. Thao tác push thực hiện nhiều việc hơn là buộc TCP gửi segment đi. Nó cũng yêu cầu TCP thiết lập bit PSH trong vùng code của segment này, để cho dữ liệu sẽ được chuyển đến chương trình ứng dụng ở nơi nhận. Như thế, khi gửi dữ liệu từ một trạm làm việc, chương trình ứng dụng sử dụng hàm push sau mỗi lần nhập từ bàn phím. Tương tự, các chương trình ứng dụng có thể buộc dữ liệu xuất được gửi đi và thể hiện tức thời trên trạm làm việc bằng cách gọi hàm push sau khi có dữ liệu xuất.

### 7.3.6. Các cổng TCP được dành riêng

Giống như UDP, TCP kết hợp việc liên kết cổng tĩnh và cổng động, sử dụng một tập hợp các phép gán cổng well known cho các chương trình thông dụng (ví dụ, thư điện tử), nhưng cũng dành ra rất nhiều cổng cho hệ điều hành để cấp phát khi chương trình cần đến. Mặc dù chuẩn ban đầu đã dành riêng các cổng nhỏ hơn 256 để làm các cổng well known, bây giờ thì các giá trị lớn hơn 1024 đã được gán. Phụ lục cuối giáo trình này liệt kê một số cổng TCP hiện tại đã được gán. Chúng ta cũng cần lưu ý rằng mặc dù các giá trị cổng TCP và UDP là độc lập, những người thiết kế đã quyết định sử dụng cùng một giá trị cổng cho bất kỳ dịch vụ tên miền (domain name) có thể được truy xuất với TCP hoặc UDP. Trong các giao thức này,

giá trị cổng 53 đã được dành riêng. Danh sách các cổng TCP phổ biến, có rất nhiều cổng mà cả TCP và UDP đều sử dụng dành cho cùng một dịch vụ mạng (tài liệu tham khảo số 14).

### 7.3.7. Vấn đề kích thước gói tin

#### *7.3.7.1. Silly Window Syndrome và những gói dữ liệu nhỏ*

Những nhà nghiên cứu tham gia trong việc phát triển TCP đã nhận thấy một vấn đề nghiêm trọng về hiệu quả mà ta có thể gặp phải khi các chương trình gửi và nhận hoạt động ở những tốc độ khác nhau. Để hiểu được vấn đề, chúng ta hãy nhớ lại rằng TCP lưu trữ dữ liệu gửi đến trong vùng đệm, và xem xét chuyện gì có thể xảy ra nếu chương trình nhận quyết định phải đọc dữ liệu mỗi lần một byte. Khi một kết nối được thiết lập lần đầu tiên, TCP nơi nhận cấp phát một vùng đệm K bytes, và sử dụng vùng Window trong những segment đáp lời để thông báo cho nơi gửi biết kích thước hiện tại vùng đệm. Nếu chương trình ứng dụng nơi gửi phát sinh dữ liệu quá nhanh, TCP nơi gửi sẽ truyền các segment với dữ liệu cho toàn bộ cửa sổ. Cuối cùng thì, nơi gửi sẽ nhận được thông báo rằng toàn bộ cửa sổ đã chứa đầy dữ liệu, không có vùng trống nào còn lại trong vùng đệm.

Khi chương trình ứng dụng nơi nhận đọc một byte dữ liệu từ vùng đệm đầy, thì sẽ có được một vùng vùng trống (một byte). Chúng ta đã nói rằng khi có được vùng trống ở trong vùng đệm của nó, TCP nơi máy nhận sẽ phát sinh một lời đáp có sử dụng vùng Window để thông báo cho nơi gửi. Trong ví dụ này, nơi nhận sẽ thông báo một cửa sổ có giá trị 1 byte. Khi biết được rằng đã có vùng trống, TCP nơi gửi sẽ đáp lại bằng cách truyền đi một segment có chứa một byte dữ liệu.

Mặc dù việc thông báo cửa sổ một byte hoạt động thật chính xác để giữ cho vùng đệm của nơi nhận luôn luôn đầy, nhưng kết quả là tạo ra nhiều segment nhỏ. TCP nơi gửi phải hình thành một segment có chứa một byte dữ liệu, đặt segment này vào một IP Datagram rồi truyền nó đi. Khi chương trình ứng dụng ở nơi nhận đọc thêm byte khác, TCP phát sinh lời đáp khác, mà làm cho nơi gửi sẽ truyền đi một segment khác chứa một byte dữ liệu.

Việc truyền các segment có kích thước nhỏ chiếm dụng băng thông của mạng một cách không cần thiết và cũng tạo thêm những xử lý tính toán không cần thiết. Vệc truyền các segment có kích thước nhỏ chiếm dụng băng thông của mạng bởi vì mỗi datagram không chỉ chuyển tải các byte dữ liệu; mà còn có thêm phần đầu và nó chiếm một lượng bytes cũng khá lớn. Các tính toán cũng gia tăng bởi vì TCP tại máy tính gửi và máy tính nhận đều phải xử lý mỗi segment. Phần mềm TCP tại nơi gửi phải cấp phát khoảng trống làm vùng đệm, hình thành phần đầu của segment, và tính checksum cho segment này. Tương tự như vậy, phần mềm IP trên máy gửi phải đóng gói segment này vào một datagram, tính checksum phần đầu, và truyền

nó đến bộ giao tiếp mạng thích hợp. Tại máy nhận, IP phải kiểm tra checksum của segment, xem xét số thứ tự, trích ra phần dữ liệu, và đặt nó vào vùng đệm. Mặc dù chúng ta đã mô tả kết quả gây ra những segment nhỏ khi nơi nhận thông báo kích thước cửa sổ nhỏ, nơi gửi cũng có thể làm cho ghi bên trong về cửa sổ hiện hành, những sự trì hoãn việc thông báo việc gia tăng kích thước cửa sổ cho nơi gửi, cho đến khi cửa sổ có thể gia tăng một kích thước đáng kể. Việc xác định bao nhiêu là đáng kể còn tuỳ thuộc vào kích thước vùng đệm của nơi nhận và kích thước tối đa của segment. TCP định nghĩa nó là giá trị tối thiểu của một cửa của kích thước vùng đệm nơi máy nhận hay số lượng các byte dữ liệu trong một segment có kích thước tối đa.

Cơ chế tránh vấn đề gặp phải như trên (vấn đề Silly Window Syndrome) của nơi nhận ngăn chặn việc thông báo cửa sổ có kích thước nhỏ trong trường hợp mà chương trình ứng dụng nơi nhận trích các byte dữ liệu ra một cách chậm chạp. Lấy ví dụ, khi vùng đệm của nơi nhận hoàn toàn đầy, nó sẽ gửi đi một lời đáp trong đó thông báo một cửa sổ có kích thước zero. Khi chương trình ứng dụng nơi nhận trích các byte ra từ vùng đệm, TCP nơi nhận sẽ tính số lượng khoảng trống trong vùng đệm. Tuy nhiên, thay vì gửi đi tức thì một thông báo về kích thước cửa sổ, nơi nhận sẽ đợi cho đến khi lượng khoảng trống đạt đến một nửa của kích thước vùng đệm hay là kích thước tối đa của segment. Như thế, nơi gửi luôn nhận được một lượng gia tăng lớn trong cửa sổ hiện hành; điều này cho phép nó truyền đi những segment lớn. Chúng ta có thể tóm tắt cơ chế này như sau:

Để tránh vấn đề Silly Window Syndrome tại nơi nhận: trước khi gửi hồi đáp, nơi nhận phải được thông báo về kích thước đã cập nhật của cửa sổ sau khi đã thông báo cửa sổ với kích thước zero, nó sẽ đợi đến khi có đủ vùng trống đạt giá trị ít nhất 50% của kích thước vùng đệm hoặc bằng với kích thước tối đa của segment.

*7.3.7.2. Các lời đáp được trì hoãn*

Có hai cách tiếp cận đã được chọn để cài đặt kỹ thuật tránh vấn đề Silly Window Syndrome tại nơi nhận. Trong cách tiếp cận đầu tiên, TCP đáp lời cho mỗi segment gửi đến, nhưng không thông báo việc gia tăng kích thước cửa sổ của nó cho đến khi kích thước này đạt đến giới hạn được xác định bởi cơ chế tránh vấn đề Silly Window Syndrome. Trong cách tiếp cận thứ hai, TCP trì hoãn việc gửi lời đáp khi mà kích thước của cửa sổ chưa đủ lớn như xác định bởi cơ chế tránh vấn đề Silly Window Syndrome. Các chuẩn thường khuyến khích sử dụng cách tiếp cận thứ hai.

Việc trì hoãn gửi lời đáp có cả ưu điểm và nhược điểm. Ưu điểm lớn nhất là: bởi vì việc trì hoãn gửi lời đáp có thể làm giảm lưu lượng và như vậy tăng hiệu suất. Lấy ví dụ, nếu có thêm dữ liệu đến trong suốt thời gian trì hoãn, chỉ cần một lời đáp lại cho tất cả dữ liệu đã gửi đến. Nếu chương trình ứng dụng tại nơi nhận

trả lời ngay tức khắc sau khi nhận được dữ liệu(ví dụ, ký tự thể hiện trên màn hình của máy truy xuất từ xa), việc trì hoãn một thời gian ngắn cũng tránh vấn đề Silly Window Syndrome. Hơn thế nữa, TCP không thể dịch chuyển cửa sổ nó cho đến khi chương trình ứng dụng nơi nhận trích dữ liệu ra khỏi vùng đệm. Trong những trường hợp mà chương trình ứng dụng nơi nhận đọc dữ liệu ra ngay khi nó được gửi đến, một thời gian trì hoãn ngắn sẽ cho phép TCP gửi đi chỉ một segment để đáp lời và đer thông báo về kích thước cửa sổ mới cập nhật. Nếu không trì hoãn lời đáp, TCP sẽ gửi lời đáp tức thì ngay khi dữ liệu đến, và sau đó lại gửi đi một lời đáp nữa để thông báo kích thước cửa sổ mới cập nhật.

Nhược điểmạng của việc trì hoãn lời đáp cũng thật rõ ràng. Quan trọng nhất là: nếu nơi nhận trì hoãn việc gửi lời đáp quá lâu, TCP nơi gửi sẽ truyền lại segment này. Việc truyền lại không cần thiết này sẽ làm giảm hiệu suất bởi vì đã sử dụng phung phí băng thông của mạng. Đồng thời, việc truyền lại cũng đòi hỏi những bước tính toán tạo máy gửi và máy nhận. Hơn nữa TCP sử dụng thời điểm đến của lời đáp để ước lượng thời gian đi trọn một vòng; việc trì hoãn lời đáp có thể làm sai lệch giá trị ước lượng và làm cho việc truyền lại kéo ra quá dài.

Để tránh những vấn đề này, chuẩn TCP đưa ra một giới hạn về thời gian TCP trì hoãn gửi lời đáp. Các cài đặt không thể trì hoãn một lời đáp lâu hơn 500 miliseconds. Hơn thế nữa, để bảo đảm rằng TCP nhận đủ số lượng của các ước lượng thời gian đi trọn một vòng, chuẩn đề nghị rằng nơi nhận không được bỏ qua việc đáp lời cho hai segment dữ liệu liền nhau.

*7.3.7.3. Tránh vấn đề Silly Window Syndrome tại nơi gửi*

Cơ chế mà TCP nơi gửi sử dụng để tránh vấn đề Silly Window Syndrome khá hiệu quả. Chúng ta nhớ lại rằng mục đích chính của kỹ thuật này là tránh việc gửi đi những segment nhỏ. Lưu ý rằng chương trình ứng dụng nơi gửi có thể phát sinh dữ liệu theo những khối có độ lớn bất kỳ (ví dụ, một lần byte). Như thế, để đạt được mục đích này, TCP nơi gửi phải cho phép chương trình ứng dụng nơi gửi thực hiện nhiều cuộc gọi để ghi nhớ lại, và phải tập hợp dữ liệu được truyền đi trong mỗi lần gửi trước khi truyền nó đi trong một segment lớn. Nghĩa là, TCP nơi gửi phải trì hoãn việc gửi một segment cho đến khi nó có thể tích luỹ đủ một lượng dữ liệu đáng kể. Kỹ thuật này được gọi là tạo nhóm.

Câu hỏi đặt ra là, "TCP nên đợi trong bao lâu trước khi truyền dữ liệu đi?" Trong trường hợp, nếu TCP đợi quá lâu, chương trình ứng dụng sẽ bị trì hoãn lâu. Quan trọng hơn nữa, TCP không thể biết có nên đợi hay không bởi vì nó không thể biết chương trình ứng dụng sẽ còn phát sinh thêm dữ liệu trong tương lai gần hay không. Nhưng ngược lại, nếu TCP không đợi đủ lâu, sẽ tạo ra những segment nhỏ và sẽ tạo ra hiệu suất thấp.

Các giao thức được thiết kế ưu tiên cho TCP để giải quyết vấn đề này và sử dụng các kỹ thuật để tạo nhóm dữ liệu thành những packet lớn. Lấy ví dụ, để đạt được việc truyền hiệu quả qua mạng, các giao thức tại trạm ở xa trước đây đã trì hoãn việc truyền mỗi ký tự bàn phím vài trăm milisecond để xác định xem người sử dụng có còn tiếp tục gõ phím nữa không. Tuy nhiên, bởi vì TCP được thiết kế tổng quát, nó có thể được sử dụng bởi rất nhiều ứng dụng khác nhau. Các ký tự có thể di chuyển qua một kết nối TCP bởi vì người sử dụng đang gõ vào bàn phím hay bởi vì một chương trình đang truyền một tập tin. Việc cố định thời hạn trì hoãn là không tối ưu cho mỗi chương trình ứng dụng.

Giống như thuật giải TCP sử dụng cho việc truyền lại và thuật giải khởi động chậm được sử dụng để tránh việc nghẽn mạch, kỹ thuật mà TCP nơi gửi sử dụng để tránh những gói dữ liệu nhỏ có khả năng điều chỉnh độ trì hoãn tuỳ thuộc vào hiệu suất hiện tại của Internet. Giống như khởi động chậm, việc tránh vấn đề Silly Window Syndrome tại nơi gửi được gọi là tự tính thời gian bởi vì nó không tính độ trì hoãn. Thay vì thế, TCP sử dụng sự gửi trở về của một lời đáp để kích hoạt việc truyền thêm những gói dữ liệu. Cơ chế này có thể tóm tắt như sau:

Để tránh vấn đề Silly Window Syndrome tại nơi gửi: khi chương trình ứng dụng tại nơi gửi phát sinh thêm dữ liệu để gửi qua một kết nối mà dữ liệu trước đã được truyền nhưng chưa nhận được lời đáp, nó sẽ đặt dữ liệu mới vào vùng đệm dành cho dữ liệu chuyển đi như thông thường, nhưng không gửi thêm các segment cho đến khi có đủ dữ liệu để điền vào segment có độ lớn tối đa. nếu nhận lời đáp đến trong khi đang đợi, thì gửi đi tất cả dữ liệu đã được tích luỹ trong vùng đệm. Áp dụng quy tắc này ngay cả khi người sử dụng yêu cầu thao tác push.

Nếu một chương trình ứng dụng phát sinh mỗi lần một byte dữ liệu, TCP sẽ tức thời gửi đi byte đầu tiên. Tuy nhiên, cho đến khi nhận được lời đáp, TCP sẽ tích luỹ thêm các byte vào vùng đệm của nó. Như thế, chương trình ứng dụng đủ nhanh so với mạng (nghĩa là, việc truyền tập tin), mỗi segment tiếp theo sẽ chứa nhiều byte. Nếu chương trình ứng dụng là tương đối chậm so với mạng (ví dụ, người sử dụng đưa vào từ bàn phím), những segment nhỏ sẽ được gửi đi mà không bị trì hoãn lâu.

Giải pháp này được biết dưới tên thuật giải Nagle, kỹ thuật này tương đối hiệu quả, bởi vì nó đòi hỏi ít bước tính toán. Một máy tính không cần phải duy trì riêng một bộ đếm thời gian cho mỗi kết nối, máy tính cũng không cần kiểm tra đồng hồ khi một chương trình ứng dụng phát sinh dữ liệu. Quan trọng hơn, mặc dù kỹ thuật này chấp nhận những kết hợp bất kỳ của việc trì hoãn trên mạng, kích thước tối đa của segment, tốc độ của chương trình ứng dụng, nó vẫn không làm giảm hiệu suất mạng trong những trường hợp thông thường.

Để hiểu được tại sao hiệu suất vẫn được duy trì ở mức độ cao trong những trường hợp bình thường, chúng ta cần để ý rằng các chương trình ứng dụng được tối ưu hoá để có hiệu suất cao sẽ không phát sinh mỗi lần một byte dữ liệu (như thế cũng sẽ làm gia tăng thêm những bước tính toán của hệ điều hành). Thay vì vậy, nó tạo ra mỗi lần một khối dữ liệu lớn. Như thế, vùng đệm dành cho dữ liệu xuất của TCP sẽ bắt đầu với đầy đủ dữ liệu cho ít nhất một segment có kích thước tối đa. Hơn nữa, bởi vì chương trình ứng dụng tạo ra dữ liệu nhanh hơn khả năng truyền dữ liệu của TCP, vùng đệm dành cho dữ liệu xuất thường xuyên đầy, và TCP sẽ không trì hoãn việc truyền đi. Kết quả là, TCP tiếp tục gửi đi các segment theo bất kỳ tốc độ nào mà Internet có thể chấp nhận, trong khi đó chương trình ứng dụng vẫn tiếp tục đưa dữ liệu vào vùng đệm. Chúng ta có thể tóm tắt:

Giao thức TCP đòi hỏi nơi gửi và nơi nhận cần cài đặt những cơ chế để tránh vấn đề Silly Window Syndrome. Nơi nhận cần tránh thông báo một cửa sổ nhỏ, và nơi gửi sử dụng một mô hình có thể hiệu chỉnh để trì hoãn việc truyền sao cho nó có thể nhóm dữ liệu lại trong những segment lớn.

**Câu hỏi và bài tập**

7.1. TCP sử dụng một vùng hạn để chứa các số thứ tự segment. Hãy tìm hiểu đặc tả giao thức để biết cách mà nó cho phép một segment có độ dài bất kỳ chuyển từ máy này sang máy kia.

7.2. Dưới những tình huống nào của sự trì hoãn, băng thông, tải và việc mất packet làm cho TCP truyền lại một lượng dữ liệu đáng kể một cách không cần thiết.

7.3. Đâu là những lập luận cho việc đóng lại những kết nối ở trạng thái không tải? Đâu là những lập luận chống đối?

7.4. Cấu trúc cụ thể của TCP segment?

7.5. Chi tiết hoạt động của giao thức cửa sổ trượt?

7.6. Quy trình bắt tay ba bước của TCP

7.7. Xử lý khi nghẽn mạng của TCP?

7.8. Bài tập: Sử dụng các phần mềm lọc gói phổ biến, chặn và phân tích các gói tin TCP gửi đi từ máy sinh viên ra mạng

# CHƯƠNG 8 ĐỊNH TUYẾN IP

## 8.1. Khái niệm định tuyến IP

### 8.1.1. Khái niệm định tuyến trong Internet

Trong một hệ chuyển mạch gói, việc định tuyến (routing) chọn lựa đường đi tối ưu để gửi gói dữ liệu qua là việc quan trọng nhất, và bộ định tuyến (router) cùng các máy tính sẽ thực hiện điều này. Việc định tuyến xảy ra một số mức. Ví dụ, trong một mạng diện rộng có nhiều liên kết vật lý giữa các bộ chuyển gói (packet switches), bản thân mạng có trách nhiệm cho việc định tuyến các gói dữ liệu từ khi chúng đến cho đến khi chúng đi qua mạng. Nhưng việc định tuyến nội bộ thì chỉ là hoàn toàn tự bao hàm trong một mạng (cục bộ). Các máy tính bên ngoài không thể tham gia vào những quyết định; chúng đơn thuần xem mạng như một thực thể chuyển phát gói dữ liệu.

Mục đích của giao thức IP là để tạo ra một mạng ảo bao gồm nhiều mạng vật lý và cung cấp dịch vụ chuyển phát datagram theo kiểu nối kết không trực tiếp. Vì thế, chúng ta sẽ tập trung vào việc chuyển phát các IP Datagram, còn được gọi là định tuyến Internet (Internet routing) hay định tuyến IP. Thông tin được sử dụng để thực hiện các quyết định định tuyến được biết dưới tên thông tin định tuyến IP (IP routing information). Giống như việc định tuyến trong một mạng vật lý, định tuyến IP chọn một con đường mà datagram phải được gửi qua đó. Không giống như việc định tuyến trong một mạng đơn, thuật giải định tuyến IP phải chọn lựa cách để gửi một datagram đi qua đi qua nhiều mạng vật lý trung gian khác nhau.

Việc định tuyến trong Internet có thể là một việc khó khăn, đặc biệt là giữa các máy tính có nhiều liên kết mạng vật lý. Một cách sơ bộ, phần mềm định tuyến sẽ căn cứ mức độ giao thông trên mạng, độ dài của datagram, hay kiểu của dịch vụ được xác định trong phần đầu datagram khi chọn con đường tối ưu (“ngắn”) nhất để gửi gói tin đi.

Để hiểu đầy đủ về việc định tuyến IP, chúng ta phải xem lại kiến trúc của TCP/IP Internet. Trước hết chúng ta nhớ lại rằng một Internet bao gồm nhiều mạng vật lý được nối trong (Internet) bởi các máy tính gọi là bộ định tuyến (ruoters). Mỗi bộ định tuyến có những liên kết trực tiếp với hai hay nhiều mạng. Ngược lại, mỗi máy tính thường nối trực tiếp tới một mạng vật lý. Tuy nhiên, chúng ta biết rằng cũng có khả năng một máy được nối trực tiếp vào nhiều mạng (gọi là multi homed).

Cả máy tính lẫn bộ định tuyến đều tham gia vào việc định tuyến một IP Datagram tới đích của nó. Khi một chương trình ứng dụng trên máy tính thực hiện việc liên lạc, thì các giao thức TCP/IP phát sinh một hay nhiều IP Datagram. Máy tính phải thực hiện một quyết định định tuyến khởi phát đầu tiên khi nó chọn nơi để gửi datagram đi.

Vấn đề định tuyến trên Internet được thực hiện dựa trên các bảng định tuyến (routing table), được lưu tại các trạm (host) hay trên các thiết bị định tuyến (router). Thông tin trong các bảng định tuyến được cập nhật tự động hoặc do người dùng cập nhật.

Các khái niệm quan trọng dùng trong định tuyến là:

–Tính có thể tìm được (reachability) dùng cho các giao thức định tuyến ngoại mạng EGP (Exterior Gateway Protocol).

–Vectơ khoảng cách (Vector Distance), thông số đo độ tối ưu đường đi giữa nguồn và đích dùng cho giao thức RIP (Routing Informaiton Protocol).

–Trạng thái kết nối (link state), nhưngx thông tin chi tiết về tình trạng của kết nối (dải thông, tải, độ tin cậy, dự phòng), dùng trong giao thức OSPF (Open Shortest Path First).

Không có giao thức định tuyến nào là toàn diện, tuỳ vào đặc tính, kích thước của mạng để chọn lựa giao thức phù hợp. Mạng nhỏ, đồng nhất nên dùng RIP, đối với các mạng lớn có topo phức tạp nên dùng OSPF...

### 8.1.2. Định tuyến IP

#### *8.1.2.1. Nguyên tắc hoạt động*

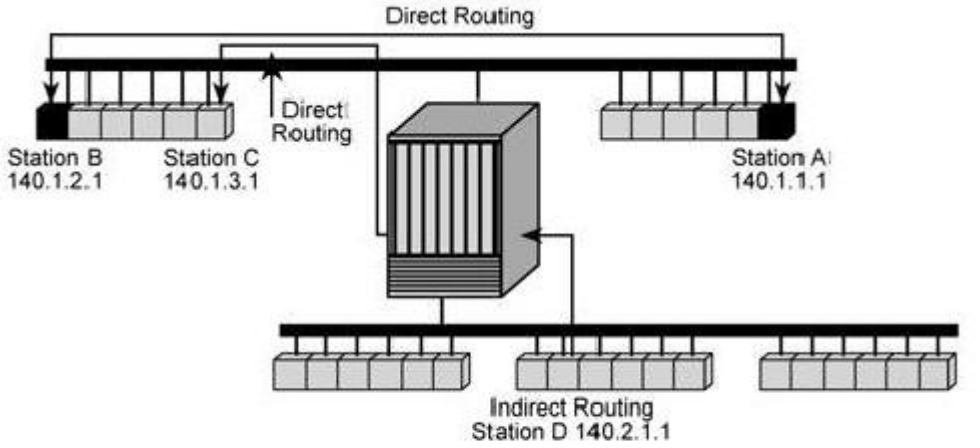
Trong vấn đề định tuyến, người ta phân biệt hai loại, đó là định tuyến trực tiếp và định tuyến gián tiếp.

Việc truyền giữa hai máy được gọi là trực tiếp khi hai máy được nối với nhau vào cùng một mạng vật lý (cùng địa chỉ mạng). Cón định tuyến không trực tiếp xảy ra khi cả máy nguồn và máy đích không cùng nối vào mạng vật lý (khác nhau về địa chỉ mạng), vì vậy việc truyền tín hiệu giữa chúng phải được thực hiện thông qua các trạm trung gian Gateway.

Để kiểm tra xem máy đích có nằm trên cùng một mạng vật lý với máy nguồn hay không thì người gửi phải tách lấy phần địa chỉ mạng của máy đích trong phần địa chỉ đích của gói dữ liệu (datagram), và so sánh với phần địa chỉ mạng trong phần địa chỉ IP của nó. Nếu hai địa chỉ này là giống nhau thì datagram sẽ được truyền trực tiếp, nếu không thì trạm gửi phải xác định một gateway để truyền các

datagram thông qua gateway. Gateway này sẽ hướng về mạng đích của gói dữ liệu đó

Ví dụ: khi có một mạng lớn bao gồm nhiều mạng cục bộ nối với nhau bởi các gateway, chỉ có hai trạm ở cách xa nhau về hai phía. Khi một trạm nguồn muốn gửi các gói dữ liệu đến một trạm khác thì nó phải đóng gói dữ liệu vào một khung (Frame) và gửi nó đến gateway gần nhất. Khi một frame đến một gateway, phần mềm giao thức IP ở gateway đó sẽ căn cứ vào địa chỉ đích để xác định Gateway tiếp theo mà gói đi qua, sau đó sẽ gửi gói tin đến gateway đó,...cho đến khi gói tin đến đích.




- Network numbers must match for direct routing.
- Different network numbers for indirect routing.
- Remote nodes may use a combination of both direct and indirect routing.

*Hình 8.1: Định tuyến trực tiếp và gián tiếp*

*8.1.2.2. Định tuyến trực tiếp*

Chúng ta biết rằng một máy trên mạng vật lý có thể gửi một frame vật lý trực tiếp tới máy khác trên cùng mạng. Để truyền một IP Datagram, nơi gửi sẽ đóng gói datagram trong frame vật lý, ánh xạ địa chỉ IP đích vào địa chỉ vật lý (địa chỉ MAC), và sử dụng phần cứng mạng để truyền nó đi. Trong chương 5 chúng ta đã trình bày hai cơ chế cho việc phân giải địa chỉ, sử dụng giao thức ARP và RARP để ánh xạ giữa địa chỉ IP và địa chỉ MAC. Trong chương 7 chúng ta đã tìm hiểu việc đóng gói datagram. Như thế, chúng ta đã xem qua tất cả các phần cần thiết để hiểu việc chuyển phát trực tiếp.

Việc truyền một IP Datagram giữa hai máy trên cùng một mạng vật lý sẽ không có sự tham dự của bộ định tuyến. Máy gửi sẽ đóng gói datagram trong frame vật lý, kết hợp địa chỉ IP với địa chỉ phần cứng, và gửi frame kết quả trực tiếp đến máy đích.

Làm thế nào mà máy gửi biết máy đích có được nối trực tiếp với mạng hay không? Không khó khăn lắm để kiểm tra việc này. Chúng ta biết rằng các địa chỉ IP được chia thành hai phần, phần đầu xác định địa chỉ mạng và phần sau xác định địa chỉ máy tính thuộc mạng. Để xem máy đích có nối trực tiếp vào cùng mạng không, máy gửi sẽ trích ra phần mạng của địa chỉ IP đích và so sánh nó với phần mạng của địa chỉ IP của máy gửi. Nếu chúng giống nhau, nghĩa là có thể gửi datagram đi trực tiếp. Cho tới đây, chúng ta thấy một trong những ưu điểm của mô hình địa chỉ Internet:

Vì các địa chỉ Internet của tất cả các máy trên cùng một mạng có phần tiền tố mạng giống nhau và việc trích ra phần tiền tố đó chỉ đòi hỏi vài lệnh máy, nên nó rất hiệu quả trong việc kiểm tra xem một máy có được nối trực tiếp vào mạng không.

Từ góc độ Internet, có thể xem chuyển phát trực tiếp là bước cuối cùng trong việc truyền một datagram bất kỳ, ngay cả khi datagram di chuyển qu nhiều mạng và các bộ định tuyến trung gian. Bộ định tuyến cuối cùng của datagram sẽ nối trực tiếp vào cùng mạng vật lý như máy đích. Như thế, bộ định tuyến cuối cùng sẽ chuyển phát datagram theo cách chuyển phát trực tiếp. Chúng ta có thể xem việc chuyển phát trực tiếp giữa nguồn và đích như một trường hợp đặc biệt của việc định tuyến – trong đó định tuyến trực tiếp datagram không phải đi qua bất kỳ một bộ định tuyến trung gian nào.

*8.1.2.3. Chuyển phát gián tiếp*

Chuyển phát gián tiếp sẽ khó khăn hơn chuyển phát trực tiếp bởi vì máy gửi phải xác định tuyến này phải truyền datagram này đi đến mạng cuối cùng của nó.

Để hình dung xem việc định tuyến gián tiếp làm việc như thế nào, hãy tưởng tượng một Internet lớn có nhiều mạng nối với nhau thông qua các bộ định tuyến nhưng chỉ có hai máy ở hai đầu. Khi một máy muốn gửi cho máy khác, nó đóng gói datagram và gửi đến bộ định tuyến gần nhất. Chúng ta biết rằng datagram có thể đến được tối thiểu một bộ định tuyến vì tất cả các mạng vật lý được nối (interconnetct) với nhau thông qua các bộ định tuyến. Giao thức IP sẽ chọn bộ định tuyến kế tiếp dọc theo con đường hướng tới đích, datagram lại được đặt vào trong frame và gửi qua mạng vật lý tiếp theo, cứ lặp lại như vậy; quá trình tiếp tục, cho đến khi nó có thể được bộ định tuyến cuối cùng chuyển phát trực tiếp. Chúng ta có thể tóm tắt ý tưởng này như sau:

Các bộ định tuyến trong một TCP/IP Internet hình thành nên một cấu trúc cùng hợp tác và liên kết (Internet). Các datagram đi từ bộ định tuyến này đến bộ định tuyến liền kề mà có thể chuyển phát datagram một cách trực tiếp.

Làm sao một bộ định tuyến biết nơi nào để gửi một datagram đến? Làm thế nào một máy tính biết được bộ định tuyến nào được sử dụng cho một đích đến cho trước? hai câu hỏi này có liên hệ với nhau bởi vì chúng liên quan đến việc định tuyến IP. Chúgg ta sẽ trả lời những câu hỏi này trong hai bước, tìm hiểu thuật giải định tuyến dựa trên bảng routing table trước khi tìm hiểu về cách mà bộ định tuyến học (cập nhật) các tuyến đường mới.

*8.1.2.4. Bảng định tuyến*

Bảng định tuyến (hay còn gọi là bảng thông tin chọn đường) là nơi lưu thông tin về các đích có thể tới được và cách thức để tới đích đó. Khi phần mềm định tuyến IP tại một trạm hay một cổng truyền nhận được yêu cầu truyền một gói dữ liệu, trước hết nó phải tìm trong bảng định tuyến của chính nó, để quyết định xem sẽ phải gửi datagram đến đâu. Tuy nhiên, không phải bảng thông tin chọn đường của mỗi trạm (hay cổng) đều chứa tất cả các thông tin về các tuyến đường có thể tới được.

Một bảng thông tin chọn đường bao gồm các bản ghi chưa bộ thông tin (N,G, M). Trong đó:

- N: địa chỉ của IP mạng đích
- G: địa chỉ cổng tiếp theo dọc trên đường truyền tới mạng N.
- V: độ tối ưu của đường đi nếu gói tin gửi qua cổng đó.

Như vậy, mỗi cổng truyền không biết được đường truyền đầy đủ để đi đến đích. Ngoài ra, trong bảng định tuyến còn có những thông tin về các cổng có thể tới nhưng không nằm trên cùng mạng vật lý. Phần thông tin này được che khuất đi và được gọi là mặc định (default). Khi không tìm thấy các thông tin về địa chỉ đích cần tìm, các gói dữ liệu sẽ được gửi tới cổng truyền mặc định

PC02 - IP Routing Table

| Destination | Network mask | Gateway | Interface | Metric | Protocol |
|---|---|---|---|---|---|
| 0.0.0.0 | 0.0.0.0 | 192.168.0.250 | LAN | 20 | Network ma.. |
| 10.0.0.0 | 255.0.0.0 | 10.0.0.2 | Cross | 20 | Local |
| 10.0.0.2 | 255.255.255.255 | 127.0.0.1 | Loopback | 20 | Local |
| 10.255.255.255 | 255.255.255.255 | 10.0.0.2 | Cross | 20 | Local |
| 127.0.0.0 | 255.0.0.0 | 127.0.0.1 | Loopback | 1 | Local |
| 127.0.0.1 | 255.255.255.255 | 127.0.0.1 | Loopback | 1 | Local |
| 172.16.0.0 | 255.255.0.0 | 192.168.0.3 | LAN | 3 | RIP |
| 192.168.0.0 | 255.255.255.0 | 192.168.0.2 | LAN | 20 | Local |
| 192.168.0.2 | 255.255.255.255 | 127.0.0.1 | Loopback | 20 | Local |
| 192.168.0.255 | 255.255.255.255 | 192.168.0.2 | LAN | 20 | Local |
| 224.0.0.0 | 240.0.0.0 | 192.168.0.2 | LAN | 20 | Local |
| 224.0.0.0 | 240.0.0.0 | 10.0.0.2 | Cross | 20 | Local |
| 255.255.255.255 | 255.255.255.255 | 192.168.0.2 | LAN | 1 | Local |

*Hình 8.2: Ví dụ về bảng định tuyến trên trạm làm việc Windows*

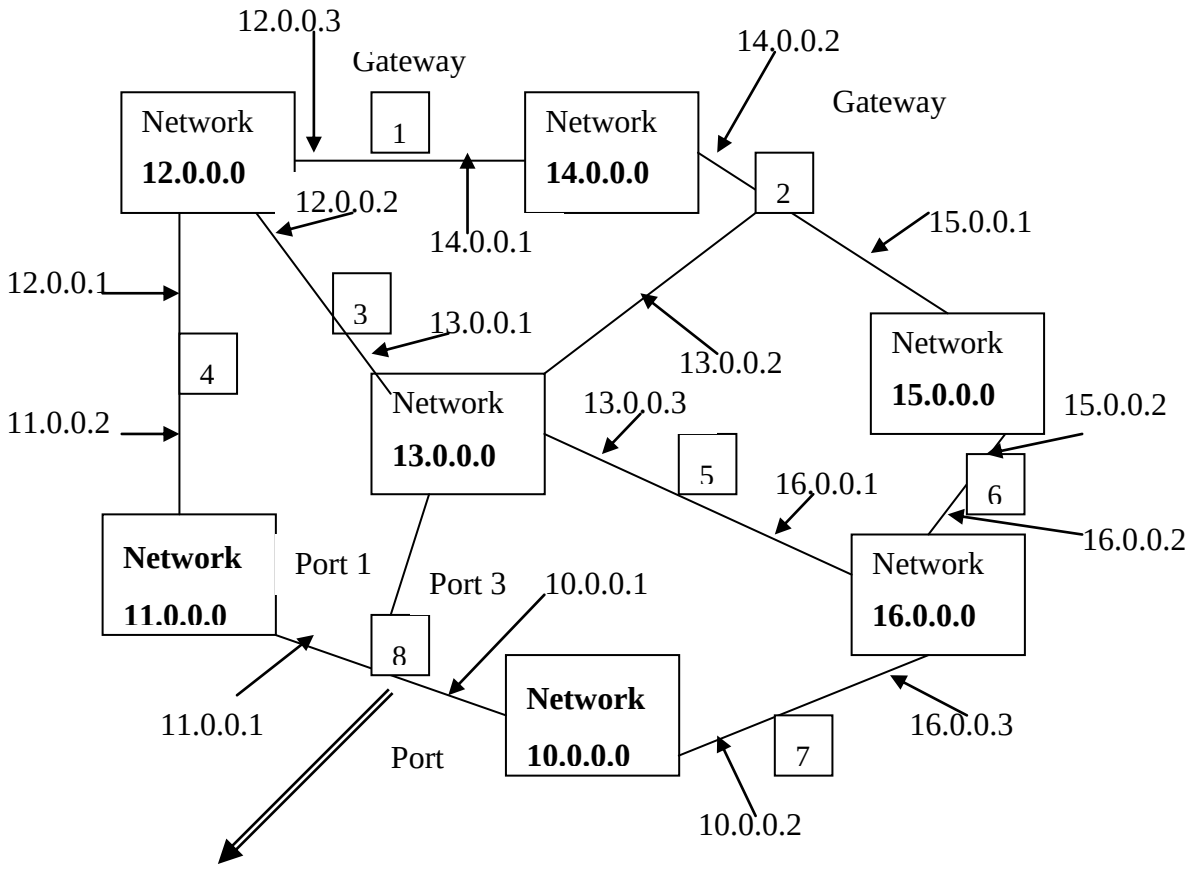
Ý nghĩa của các trường:

Destination: địa chỉ đích của mạng hay trạm.

Gateway: cổng dùng để tới đích đã được chỉ định.

Interface: tên giao diện mạng được dùng của tuyến đường này.

Hình 8.3 là một ví dụ về bảng định tuyến đích trực tiếp và đích gián tiếp.




*Hình 8.3: Sơ đồ mạng và định tuyến*

**Bảng định tuyến gateway 8**

| To host on network | Route the datagram | Across this physical port |
|---|---|---|
| **10.0.0.0** | **Direct** | **2** |
| **11.0.0.0** | **Direct** | **1** |
| **12.0.0.0** | **11.0.0.2** | **1** |
| **13.0.0.0** | **Direct** | **3** |
| **14.0.0.0** | **13.0.0.2** | **3** |

| | | |
|---|---|---|
| **15.0.0.0** | **10.0.0.2** | **2** |
| **16.0.0.0** | **10.0.0.2** | **2** |

*Bảng 8.1. Bảng định tuyến gateway*

*8.1.2.5. Định tuyến đến trạm kế tiếp*

Sử dụng phần địa chỉ mạng của địa chỉ đích thay vì toàn bộ địa chỉ của máy đích làm cho việc định tuyến hiệu quả đồng thời làm cho bảng định tuyến có kích thước nhỏ. Quan trọng hơn nữa, nó giúp cho việc che giấu thông tin, giữ cho các chi tiết thông tin cụ thể của máy chỉ được biết trong môi trường cục bộ mà các máy này hoạt động trong đó. Về cơ bản, một bảng định tuyến bao gồm các cặp (N, G), với N là địa chỉ IP của mạng đích, và G là địa chỉ IP của bộ định tuyến "kế tiếp" trên con đường tới mạng N. Bộ định tuyến G được gọi là trạm kế, và ý tưởng của việc sử dụng một bảng định tuyến để lưu trữ chặng kế cho mỗi đích được gọi là định tuyến trạm kế. Như thế, bảng định tuyến trong một bộ định tuyến G chỉ xác định một bước kế tiếp trên con đường từ G đến một mạng đích, bộ định tuyến không biết toàn bộ con đường đến đích.

Điều quan trọng cần hiểu là mỗi dòng trong một bảng định tuyến trỏ tới một bộ định tuyến mà có thể đi đến được thông qua một mạng đơn. Nghĩa là, tất cả các bộ định tuyến được liệt kê trong bảng định tuyến của máy M phải nằm trong mạng (có thể nhiều hơn một mạng) mà máy M được nối trực tiếp vào. Khi một datagram đã sẵn sàng đi ra khỏi máy M, phần mềm IP xác định địa chỉ IP đích và trích ra phần mạng M, phần mềm IP xác định địa chỉ IP đích và trích ra phần mạng. Sau đó M sử dụng phần mạng để thực hiện quyết định định tuyến, chọn ra một bộ định tuyến mà có thể đến được trực tiếp.

Trong thực tế, chúng ta cũng áp dụng nguyên lý che dấu thông tin đối với các máy tính. Chúng ta lập luận rằng mặc dù máy tính có bảng định tuyến IP, chúng cũng chỉ giữ thông tin tối thiểu trong bảng. Ý tưởng là để buộc máy tính dựa vào bộ định tuyến cho hầu hết việc định tuyến.

Việc lựa chọn định tuyến chỉ dựa trên ID mạng đích cũng có một vài hệ quả. Trước hết, trong hầu hết các cài đặt, nó có nghĩa rằng tất cả các giao dịch đi đến một mạng cho trước, sẽ đi theo cùng một con đường. Kết quả là, ngay cả khi tồn tại nhiều con đường, chúng có thể không được sử dụng cùng lúc. Hơn nữa, tất cả các loại giao dịch đều tuân theo cùng một con đường mà không cần để ý đến độ trì hoãn, hay tốc độ của các mạng vật lý. Thứ hai, bởi vì chỉ có bộ định tuyến cuối cùng trên con đường sẽ liên lạc với máy đích, chỉ nó có thể xác định rằng máy tính này tại hay hoạt động không. Vì thế, chúng ta cần sắp xếp một cách để bộ định

tuyến đó gửi báo cáo về các vấn đề chuyển phát ngược trở về nguồn ban đầu (giao thức ICMP). Thứ ba, bởi vì mỗi bộ định tuyến thực hiện (chuyển) giao dịch một cách độc lập, datagram di chuyển từ máy A đến máy B có thể đi theo một con đường hoàn toàn khác với datagram di chuyển từ máy B đến máy A. Chúng ta cần đảm bảo rằng các bộ định tuyến cùng hợp tác để bảo đảm luôn luôn có được liên lạc hai chiều.

*8.1.2.6. Định tuyến mặc định*

Một kỹ thuật khác được dùng để che dấu thông tin và giữ cho bảng định tuyến có kích thước nhỏ hợp nhất nhiều thông tin (động) và một trường hợp mặc định. Ý tưởng này nhằm mục đích để cho phần mềm định tuyến IP trước hết tìm kiếm mạng đích trong bảng định tuyến. Nếu không tìm ra, thủ tục định tuyến sẽ gửi datagram tới bộ định tuyến mặc định.

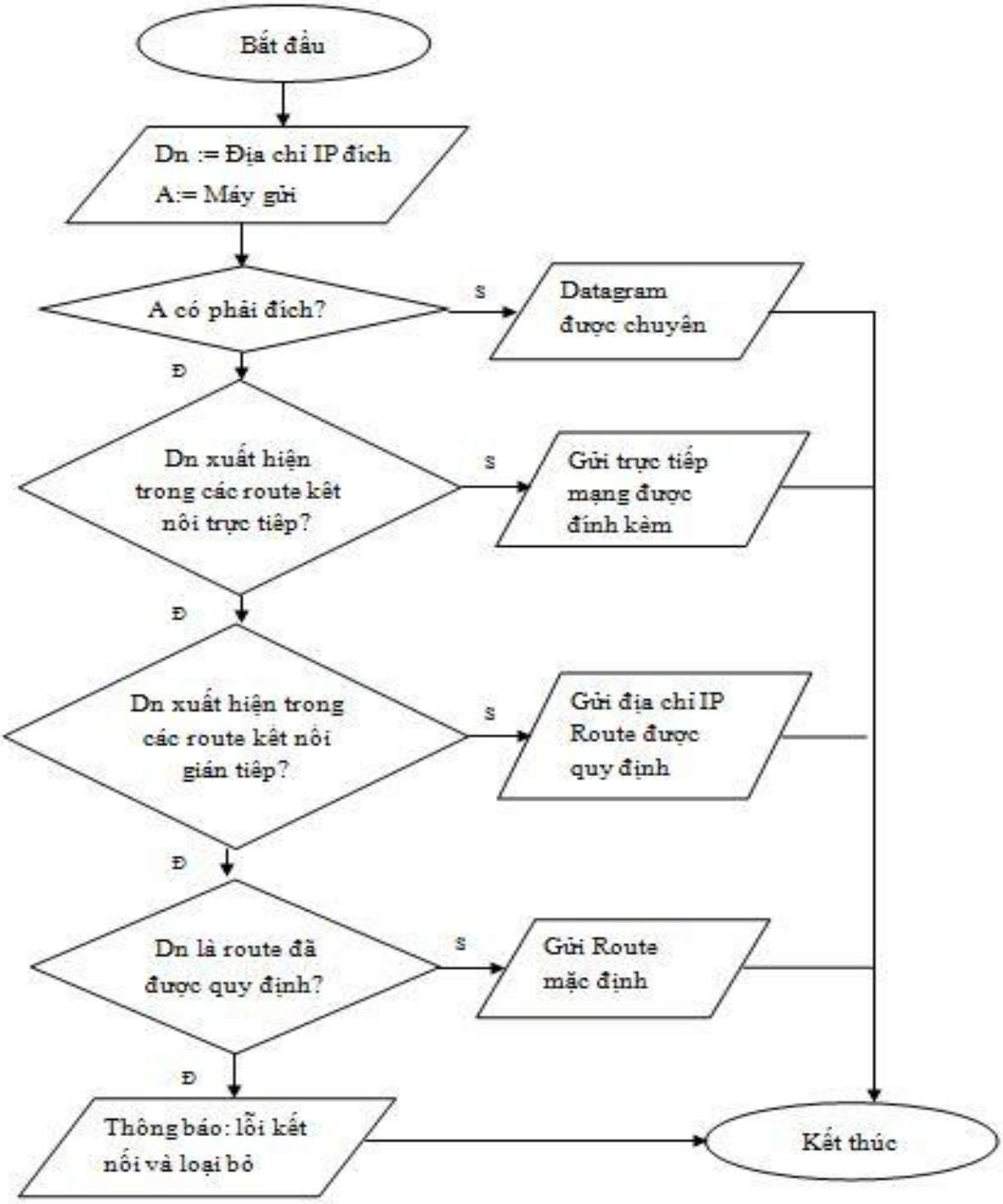
Định tuyến mặc định đặc biệt hữu dụng khi một nơi chỉ có một lượng nhỏ các địa chỉ cục bộ và chỉ có một cái nối vào phần còn lại của Internet (đa phần các mạng cục bộ vừa và nhỏ hiện nay đều theo mô hình này). Ví dụ, định tuyến mặc định làm việc rất tốt trong các máy tính được nối vào mạng vật lý đơn và chỉ có một bộ định tuyến dẫn tới phần còn lại của Internet. Quyết định định tuyến bao gồm hai bước kiểm tra: một cho mạng cục bộ và một mặc định trỏ tới chỉ có một bộ định tuyến. Ngay cả khi nơi này có một vài mạng cục bộ, việc chuyển kênh cũng đơn giản bởi vì nó bao gồm chỉ vài bước kiểm tra cho các mạng cục bộ cùng với một mặc định cho các đích khác.

*8.1.2.7. Định tuyến xác định máy*

Mặc dù chúng ta nói rằng mọi việc định tuyến dựa trên mạng mà không dựa trên những máy đơn, hầu hết phần mềm định tuyến IP cho phép việc định tuyến theo máy, được xác định như là một trường hợp đặc biệt. Với việc định tuyến theo máy, người quản trị mạng cục bộ có quyền điều khiển việc sử dụng mạng, được phép kiểm tra, và cũng có thể được dùng để kiểm soát việc truy xuất cho các mục đích an toàn và bảo mật. Khi bắt lỗi các liên kết mạng cũng như bảng định tuyến, khả xác định một đường đi cụ thể tới từng máy là đặc biệt hữu dụng.

*8.1.2.8. Thuật toán định tuyến IP và mặt nạ mạng con*

Thuật toán định tuyến được mô tả như sau:

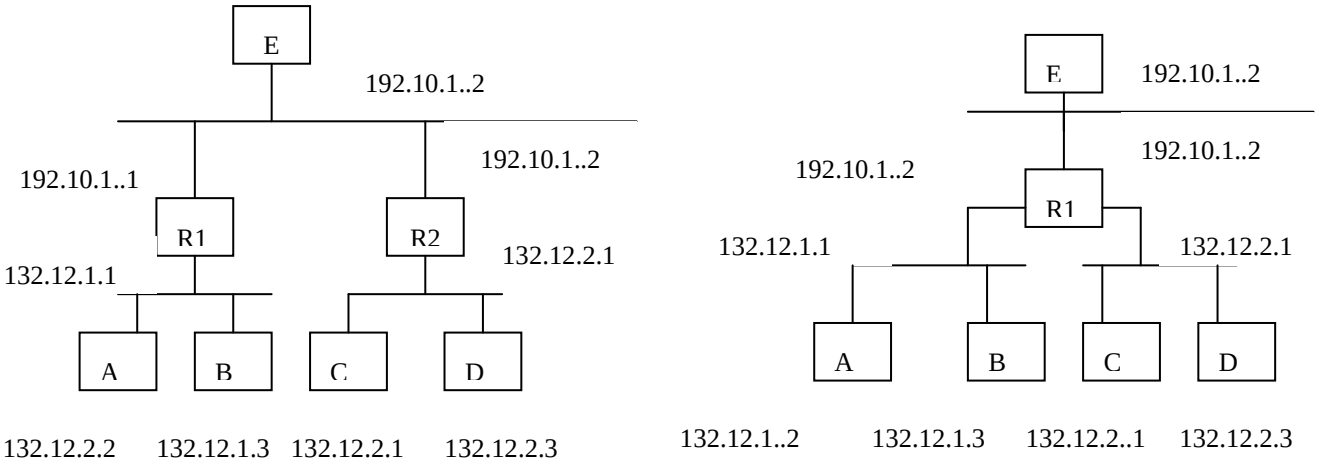



*Hình 8.4: Thuật toán định tuyến IP*

Thông thường, khi kết nối các mạng cục bộ LAN ở mức liên mạng IP, sử dụng các bộ định tuyến IP, người ta gán một địa chỉ (phần) mạng IP thuộc một lớp địa chỉ IP nào đó cho một mạng LAN, tuỳ theo độ lớn của mạng LAN đó. Như vậy, số địa chỉ mạng IP sẽ tăng nhanh chóng, phụ thuộc vào nhu cầu kết nối các mạng cục bộ, làm cạn dần nguồn tài nguyên địa chỉ mạng.

Để giải quyết vấn đề này, người ta sử dụng một phần địa chỉ mạng IP đã có. Cấu trúc mạng IP trên cơ sở phân mạng IP còn được gọi là cấu trúc phân mạng IP (subnet topology). Địa chỉ phân mạng IP sẽ được gán cho các mạng cục bộ cần được kết nối. Đối với "bên ngoài" ta vẫn chỉ có một mạng IP. Thực chất, các mạng cục bộ "bên trong" được kết nối trên cơ sở phân mạng IP, và chúng độc lập với nhau như trong trường hợp khác nhau về mạng vật lý, muốn truyền thông được, chúng phải sử dụng các bộ định tuyến cục bộ.

Để xác định địa chỉ phân mạng, có thể dùng mặt nạ phân mạng (subnet mask), tương tự như mặt nạ mạng. Mặt nạ phân mạng không nhất thiết phải bắt đầu và kết thúc ở danh giới byte. Thông thường, người ta chọn phần địa chỉ thiết bị cuối làm địa chỉ phân mạng sao cho địa chỉ mạng và địa chỉ phân mạng tạo thành địa chỉ liên tục. Ví dụ: nếu địa chỉ phân mạng thuộc lớp B thì byte cao của phần địa chỉ thiết bị cuối được chọn làm địa chỉ phân mạng. Mặt nạ mạng và mặt nạ phân mạng, vì vậy, cũng tạo thành một mặt nạ liên tục. Tất cả các trạm làm việc trong cùng một phân mạng IP sẽ có cùng giá trị mặt nạ phân mạng.




a) cấu hình phân mạng sai b) cấu hình phân mạng đúng

*Hình 8.5: Ví dụ về cấu hình phân mạng*

Khi kết nối các mạng cục bộ trên cơ sở phân mạng IP, các phân mạng IP này phải được kết nối toàn phần (fully interconnected) để đảm bảo định tuyến hoạt động đúng. Trong hình a) trên, E muốn chuyển dữ liệu cho C mà không biết mạng 132.12.0.0 được cấu hình thành các phân mạng 132.12.1.0 và 132.12.2.0. Đây cũng chính là đặc trưng cấu trúc phân mạng IP: đối với các mạng IP "bên ngoài" thì chỉ tồn tại mạng 132.12.0.0. E sẽ chuyển gói dữ liệu cho C qua R1 dựa trên nguyên tắc định tuyến phân mạng IP (tương tự như định tuyến IP) sử dụng mặt nạ phân mạng IP.

*8.1.2.9. Định tuyến với các địa chỉ IP*

Ngoại trừ việc giảm bớt thời gian sống và tính lại checksum, định tuyến IP không làm thay đổi datagram ban đầu. Cụ thể, các địa chỉ datagram nguồn và đích sẽ không bị thay đổi, chúng luôn luôn mô tả địa chỉ IP của nguồn ban đầu và địa chỉ IP của đích cuối cùng (chỉ một ngoại lệ xảy ra khi datagram chứa một chọn lựa record route). Khi IP thực hiện thuật giải định tuyến, nó chọn một địa chỉ IP mới, địa chỉ IP của máy mà datagram sẽ được gửi tiếp. Tuy nhiên, nếu datagram có thể được chuyển phát trực tiếp, địa chỉ mới sẽ giống như địa chỉ của đích cuối cùng.

Chúng ta đã nói rằng, gói tin IP được chọn gửi đên địa chỉ router kế tiếp bởi thuật giải định tuyến. Vậy giao thức IP lưu trữ địa chỉ trạm kế tiếp ở đâu? Không phải trong datagram; không có chỗ dành riêng cho nó. Thực ra, IP không hề "lưu trữ" địa chỉ trạm kế tiếp. Sau khi thực hiện thuật giải định tuyến, IP truyền datagram và địa chỉ trạm kế tiếp đến phần mềm giao tiếp mạng đảm trách cho mạng phần cứng mà datagram phải được gửi trên đó. Phần mềm giao tiếp mạng liên kết địa chỉ trạm kế tiếp với địa chỉ vật lý, tạo nên một frame với địa chỉ vật lý đó, đặt datagram vào phần dữ liệu của frame, và gửi kết quả đi. Sau khi sử dụng địa chỉ trạm kế tiếp để tìm ra địa chỉ vật lý, phần mềm giao tiếp mang sẽ huỷ bỏ địa chỉ trạm kế tiếp.

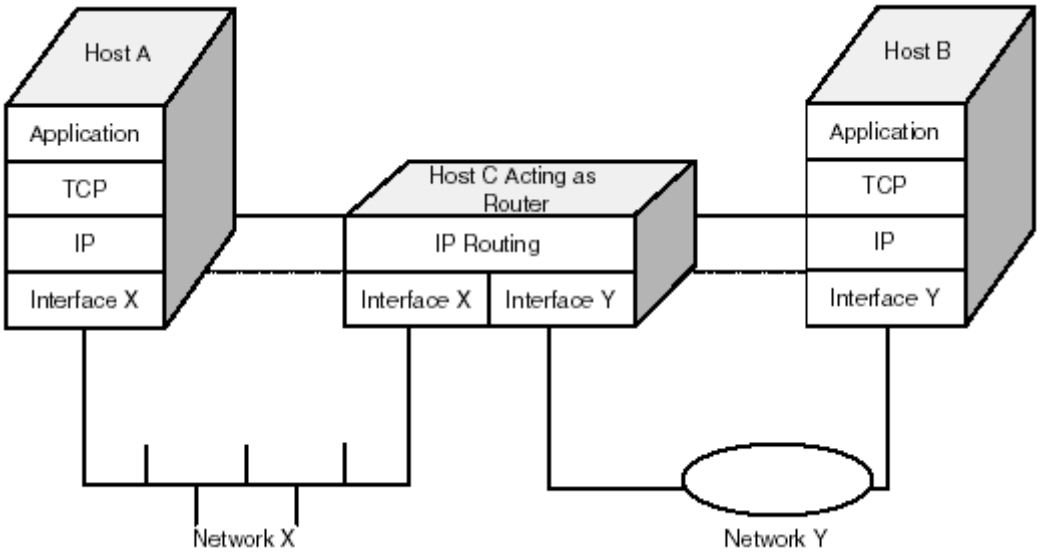
Thoạt nhìn có vẻ hơi kỳ quặc khi các bảng định tuyến lại lưu trữ địa chỉ IP của trạm kế tiếp cho mỗi mạng đích trong khi các địa chỉ đó phải được diễn dịch thành địa chỉ vật lý tương ứng trước khi datagram được gửi đi. nếu chúng ta hình dung khi một máy tính gửi một chuỗi các datagram tới cùng một địa chỉ đích, việc sử dụng các địa chỉ IP sẽ vô cùng kém hiệu quả. Địa chỉ IP đã làm tròn bổn phận là từ nó trích ra được địa chỉ mạng đích trong mỗi datagram và sử dụng bảng định tuyến để có được địa chỉ trạm kế tiếp. Sau đó nó truyền datagram và địa chỉ trạm kế tiếp tới bộ giao tiếp mạng, mà nó sẽ tính lại liên kết địa chỉ với địa chỉ vật lý. Nếu bảng định tuyến đã sử dụng địa chỉ vật lý, việc liên kết giữa địa chỉ IP của trạm kế tiếp và địa chỉ vật lý có thể được thực hiện chỉ một lần, giảm được thời gian tính toán không cần thiết.

Nhưng tại sao phần mềm IP tránh sử dụng các địa chỉ vật lý khi lưu trữ và thuật toán định tuyến? Có hai lý do quan trọng.

- Trước hết, bảng định tuyến cung cấp một giao tiếp đặc biệt rõ ràng giữa phần mềm IP chuyển phát các datagram và phần mềm cấp cao sử dụng việc định tuyến. Để truy tìm lỗi cho các vấn đề định tuyến, người quản trị mạng thường cần kiểm tra các bảng định tuyến. Chỉ sử dụng địa chỉ IP trong bảng định tuyến

làm cho người quản trị dễ hiểu hơn và dễ hơn trong việc xác định phần mềm có cập nhật thông tin định tuyến một cách chính xác không.

- Thứ hai, mục đích của Internet Protocol là để xây dựng một sự trừu tượng nhằm che dấu các chi tiết của phần cứng mạng cơ sở hạ tầng.




*Hình 8.7: Định tuyến IP*

*8.1.2.10. Xử lý các datagram gửi đến*

Trong những phần trước, chúng ta đã tìm hiểu việc định tuyến IP bằng việc mô tả cách thực hiện các quyết định định tuyến đối với các gói dữ liệu được gửi đi. Tuy nhiên, chúng ta biết rằng phần mềm IP cũng phải xử lý các datagram được gửi đến.

Khi một datagram đến một máy, phần mềm giao tiếp mạng chuyển nó tới module IP để xử lý. Nếu địa chỉ đích của datagram giống với nó và chuyển nó đến phần mềm giao thức cấp cao hơn để chuyển tiếp. Nếu địa chỉ IP không giống, máy tính phải huỷ bỏ datagram (nghĩa là máy tính không được phép chuyển đi những datagram mà nó "tình cờ" nhận được).

Không giống như máy tính, bộ định tuyến thực hiện nhiệm vụ định tuyến. Khi một IP Datagram đến một bộ định tuyến, nó được chuyển đến phần mềm IP. Và lại có hai trường hợp: datagram có thể đã đến được đích cuối cùng của nó, hoặc nó có thể phải được chuyển đến trạm kế tiếp. Cũng như với máy tính, nếu địa chỉ IP của datagram giống với địa chỉ IP của bộ định tuyến, phần mềm IP chuyển datagram đến phần mềm giao thức cấp cao hơn để xử lý (thông thường, những datagram được xác định gửi đến cho bộ định tuyến là những cái được dùng để kiểm tra đường truyền hoặc những datagram chuyển tải các lệnh quản trị bộ định

tuyến, những bộ định tuyến cũng phải cũng phải giữ chưa đến đích cuối cùng tin trong bảng định tuyến của nó).

Việc xác định xem một IP Datagram đã đến được đích cuối cùng của nó hay chưa, không hoàn toàn là một việc đơn giản. Chúng ta nhớ lại rằng ngay cả một máy tính cũng có thể có nhiều liên kết mạng (vật lý), mỗi cái có địa chỉ IP riêng. Khi một IP Datagram đến, máy tính phải so sánh địa chỉ Internet đích với địa chỉ IP của mỗi liên kết mạng của nó. Nếu có trường hợp giống, nó sẽ giữ datagram và xử lý. Máy tính cũng phải nhận những datagram được quảng bá trên mạng vật lý nếu địa chỉ IP đích của chúng là địa chỉ IP quảng bá trên hoặc địa chỉ IP quảng bá trực tiếp cho mạng đó, các địa chỉ không phân lớp, mạng con, và truyền nhiều hướng cùng lúc (multicast) làm cho việc nhận biết địa chỉ phức tạp hơn nữa. Trong trường nào đi nữa, nếu địa chỉ không giống bất kỳ địa chỉ nào của máy, IP sẽ giảm bớt vùng thời gian sống (TTL) trong phần đầu datagram, huỷ bỏ datagram nếu giá trị vùng này là zero, hay tính checksum mới và chuyển datagram đi nếu giá trị còn lớn hơn zero.

Liệu rằng mỗi máy sẽ chuyển đi những IP Datagram mà nó nhận? Dĩ nhiên, bộ định tuyến phải chuyển các datagram đi vì đó là chức năng chính của nó. Chúng ta nói rằng một số máy tính multi homed hoạt động như bộ định tuyến mặc dù chỉ là những hệ máy tính thông thường. Thông thường, sử dụng máy tính làm bộ định tuyến không phải ý tưởng hay, nhưng nếu đã chọn như thế, máy tính phải được cấu hình để chuyển datagram như là một bộ định tuyến thực thụ. Còn đối với những máy khác thì sao, những máy mà không sử dụng cho mục đích định tuyến? Câu trả lời là những máy tính không được thiết kế làm bộ định tuyến không được chuyển đi các datagram mà nó nhận; chúng phải huỷ bỏ các datagram đó.

Có bốn lý do tại sao một khi máy tính không được thiết kế làm bộ định tuyến thì không nên thực hiện các chức năng định tuyến. Trước hết, khi những máy này nhận được datagram dù định gửi cho máy khác, đã có nhiều điều gì sai trong việc định địa chỉ Internet, việc định tuyến, hoặc phát chuyển. Sai sót này có thể không được phát hiện nếu máy tính này lại chuyển datagram đi. Thứ hai, việc chuyển đi sẽ tạo nên việc hoạt động không cần thiết trên mạng. Thứ ba, những lỗi đơn giản cũng có thể gây nên sự khủng hoảng. Giả sử mỗi máy tính đều hoạt động như bộ định tuyến, ta hãy thử hình dung chuyện gì sẽ xảy ra nếu một máy tính có quảng bá một datagram mà dự định cho máy tính H. Vì nó đã được quảng bá, mỗi máy tính trên mạng đều nhận được datagram này. Tất cả các máy lại chuyển datagram này tới H, làm cho máy H này "chết" vì nhận được quá nhiều datagram. Những chi tiết hơn sẽ được trình bày trong chương sau, bộ định tuyến làm nhiều

việc hơn là đơn thuần chuyển datagram đi. Bộ định tuyến sử dụng một giao thức đặc biệt để thông báo cùng một lỗi, còn máy tính thì không (một lần nữa, để tránh việc nhiều thông báo cùng một lỗi đổ dồn lên một máy). Bộ định tuyến cũng nhân bản thông tin định tuyến và cập nhật đồng bộ thường xuyên để bảo đảm rằng các bảng định tuyến của chúng được nhất quán.

*8.1.2.11. Thiết lập bảng định tuyến*

Chúng ta tìm hiểu cách IP chuyển các datagram dựa vào nội dung của các bảng định tuyến, mà không bàn đến cách mà hệ thống khởi động các bảng định tuyến của chúng hay cập nhật chúng khi mạng thay đổi. Các chương sau được dành cho vấn đề này và tìm hiểu các giao thức mà cho phép các bộ định tuyến có được thông tin định tuyến nhất quán. Vào lúc này, điều quan trọng đối với chúng ta là hiểu rằng phần mềm IP sử dụng bảng định tuyến bất cứ khi nào cần tìm đường chuyển datagram đi, vì thế việc thay đổi trong bảng định tuyến sẽ làm thay đổi con đường mà datagram sẽ đi qua.

## 8.2. Kiến trúc chính Internet

8.2.1. Giới thiệu chung về các giao thức định tuyến

Mặc dù chúng ta thấy cơ sở của việc chuyển datagram, chúng ta chưa đề cập đến cách mà các máy tính và bộ định tuyến lấy được thông tin cho các bảng định tuyến của chúng. Có hai khía cạnh trong vấn đề này: những thông tin gì lưu trong bảng, và làm thế nào mà bộ định tuyến lấy tự động các giá trị đó. Cả hai chọn lựa đều tuỳ thuộc độ phức tạp của kiến trúc mạng và kích thước của các mạng cũng như các chính sách quản trị.

Nói chung, việc thiết lập các định tuyến bao gồm việc khởi động và cập nhật. Mỗi bộ định tuyến phải thiết lập một tập hợp các tuyến đường ban đầu khi khởi động (bật máy), và nó phải cập nhật bảng này khi tuyến đường thay đổi (ví dụ, khi bộ giao tiếp mạng bị hỏng). Việc khởi động tuỳ thuộc vào hệ điều hành. Trong một số hệ điều hành, bộ định tuyến đọc bảng định tuyến ban đầu từ đĩa cứng, ROM khi khởi động, và duy trì nó trong bộ nhớ chính (RAM). Với một số hệ khác, hệ điều hành bắt đầu với một bảng trống, sau đó tự động điền vào các tuyến đường bằng cách thực hiện các lệnh cụ thể (ví dụ, các lệnh có trong tập tin lệnh khởi động). Cuối cùng cũng có những hệ điều hành bắt đầu bởi một tập hợp các tuyến đường có được từ địa chỉ của các mạng cục bộ mà nối vào bộ định tuyến rồi sau đó liên lạc với các bộ định tuyến lân cận để có thêm và cập nhật tuyến đường.

Một khi đã xây dựng xong bảng định tuyến ban đầu, bộ định tuyến phải cập nhật những thay đổi trong những tuyến đường. Trong những mạng nhỏ, thay đổi chậm, người quản lý có thể thiết lập và sửa đổi tuyến đường một cách thủ công.

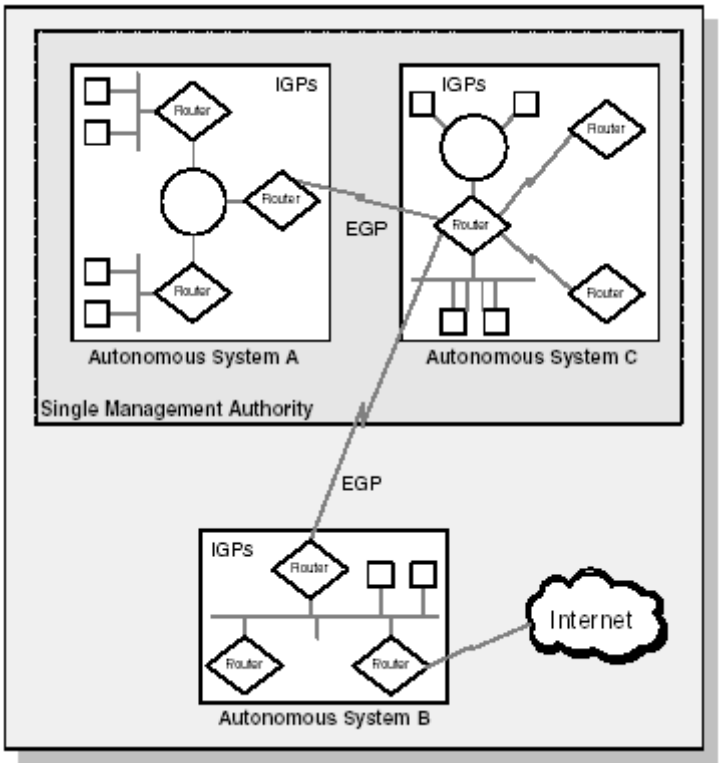
Tuy nhiên, trong mỗi thông tin lớn và thay đổi nhanh, không thể nào cập nhật một cách thủ công được và dễ bị sai sót do con người. Cần có phương pháp tự động cập nhật thông tin giữa các Router với nhau, những giao thức định tuyến có khả năng tự động cập nhật thông tin đó, gọi là các giao thức định tuyến động.

Các giao thức định tuyến động được các Router dùng để liên lạc với nhau như các giao thức RIP (Routing Information Protocol), OSPF (Open Shortest Path First) và BGP (Border Gateway Protocol).

Router dùng phương pháp định tuyến động để thông báo cho các Router láng giềng là hiện thời nó đang liên kết tới mạng nào. Tiến trình Router đang chạy một giao thức định tuyến liên lạc với các Router láng giềng gọi là Routing Daemon. Routing Daemon cập nhật bảng định tuyến của nhân (kernel) hệ thống với thông tin nó nhận được từ các Router láng giềng. Định tuyến động không làm thay đổi cách hệ thống thực hiện định tuyến tại tầng IP, gọi là cơ chế định tuyến (routing mechanism). Kernel vẫn tìm kiếm bảng định tuyến của nó theo cách tương tự, trước hết nó tìm các tuyến cho host, các tuyến cho mạng và cuối cùng nếu không có địa chỉ mạng nào tương ứng thì nó sẽ dùng các tuyến mặc định (default route).

Các tuyến trong bảng định tuyến được thêm bớt một cách tự động bằng một tiến trình Routing Daemon khi các tuyến thay đổi theo thời gian. Một tiến trình Routing Daemon sẽ thêm một chính sách định tuyến (routing policy) vào hệ thống, chọn các tuyến nào để đặt vào bảng định tuyến của Kernel. Nếu một Daemon tìm được nhiều tuyến tới một đích, nó sẽ chọn tuyến nào tốt nhất để chèn vào bảng định tuyến. Nếu Daemon tìm thấy một tuyến hỏng (có thể do router hỏng hoặc do đường thoại hỏng), nó có thể xoá các tuyến bị ảnh hưởng hoặc thêm vào các tuyến thay thế để sửa lỗi.

Trong một hệ thống như Internet, nhiều giao thức định tuyến khác được dùng. Mạng xương sống NSFNET của Internet thiết lập một hệ thống khi tất cả các router trong backbone nằm dưới một sự điều khiển quản trị thống nhất. Mỗi hệ thống có thể chọn giao thức định tuyến riêng cho nó để liên lạc giữa các router trong hệ thống đó, được gọi là IGP (Interior Gateway Protocol) hoặc giao thức định tuyến liên miền (Interdomain Routing Protocol). IGP thông dụng nhất là RIP. Hiện nay dùng IGP mới hơn là OSPF (Open Shortest Path First).




*Hình 8.8: Hai họ giao thức định tuyến IGP & EGP*

Các giao thức định tuyến ngoài gọi là EGP (Exterior Gateway Protocol) hoặc là giao thức định tuyến liên miền (Interdomain Routing Protocol) được dùng giữa các router trong các hệ thống khác nhau. Một EGP mới là BGP được dùng giữa các mạng xương sống NSF NET.

8.2.2. Kiến trúc chính trong Internet, hệ tự quản

*8.2.2.1.Định tuyến với thông tin bán phần*

Sự khác biệt về nguyên lý giữa bộ định tuyến và các máy tính thông thường là ở điểm các máy tính thường biết rất ít về cấu trúc của Internet mà chúng kết nối, các máy tính không có đầy đủ thông tin về tất cả các địa chỉ đích có thể có, thậm chí không biết có đầy đủ thông tin về tất cả các địa chỉ mạng có thể có. Thực ra, nhiều máy tính chỉ có hai tuyến đường trong bảng định tuyến của chúng: một con đường cho mạng cục bộ và một con đường mặc định cho bộ định tuyến gần đó. Máy tính gửi tất cả những datagram không dành cho máy cục bộ đến bộ định tuyến cục bộ để chuyển đi. Như vậy điểm mấu chốt là:

Một máy tính có thể chuyển datagram một thành công ngay cả khi nó chỉ có một phần thông tin định tuyến bởi vì nó có thể dựa vào bộ định tuyến.

Liệu bộ định tuyến cũng có thể chuyển datagram đi khi chỉ có một phần thông tin. Câu trả lời có thể, những chỉ trong những tình huống nhất định. Để hiểu các tình huống này, hình dung một Internet là hệ thống đường xá ở mức ngoài với

những con đường có dấu hiệu chỉ đường đặt tại giao lộ. Hãy tưởng tượng rằng ta không có bản đồ, cũng không thể hỏi người hướng dẫn bởi vì ta không thể nói tiếng địa phương, ta cũng không biết gì về các địa danh có trong dấu hiệu chỉ đường, nhưng ta cần đi tới ngôi làng có tên Sussex. Ta tiếp tục cuộc hành tình của mình, đi theo con đường duy nhất ra khỏi thành phố và bắt đầu xem các hiệu chỉ dẫn. Dấu hiệu đầu tiên là:

Norflok rẽ trái; hammod phải; còn lại thì đi thẳng (giả sử rằng dấu hiệu chỉ dẫn được quốc tế hoá, nên ta hiểu được).

Bởi vì đích đến của ta không có trong hướng dẫn cụ thể, nên ta tiếp tục đi thẳng. theo thuật ngữ định tuyến, chúng ta nói rằng ta đã đi theo tuyến đường định tuyến mặc định. Sau khi đi qua một số dấu hiệu chỉ dẫn, cuối cùng ta thấy một dấu hiệu là:

Essese rẽ trái; Sussex phải; còn lại thì đi thẳng.

Ta sẽ phải, đi theo một số chỉ dẫn nữa, và tìm thấy con đường dẫm đến Sussex.

Chuyến đi tưởng tượng của chúng ta cũng tương tự như một datagram di chuyển trên Internet, các dấu hiệu chỉ đường thì tương tự như bảng định tuyến trong các bộ định tuyến dọc theo đường đi. Khi không có bản đồ hay những hướng dẫn khác (hướng dẫn viên chẳng hạn) thì việc di chuyển hoàn toàn phụ thuộc vào dấu hiệu chỉ đường, cũng như việc định tuyến datagram trong Internet phụ thuộc hoàn toàn vào bảng định tuyến. Rõ ràng, hoàn toàn có thể đi đến đích mặc dù mỗi dấu hiệu chỉ đường chỉ chứa một phần thông tin.

Trung tâm của vấn đề là tính chính xác. lấy ví dụ người du lịch, ta có thể tự hỏi. "làm sao có thể được bảo đảm rằng các dấu hiệu chỉ đường sẽ dẫn đến đích cuối cùng của tôi?" Ta cũng có thể hỏi "làm sao tôi có thể bảo đảm rằng các dấu hiệu chỉ đường sẽ dẫn đến đích cuối cùng của tôi theo con đường ngắn nhất?" những câu hỏi này sẽ trở trở nên mối lo lắng nếu ta đã qua nhiều dấu hiệu chỉ đường nhưng vẫn chưa thấy dấu hiệu đến đích của mình. Dĩ nhiên, câu trả lời tuỳ thuộc vào mô hình của hệ thống đường xá và nội dung của các dấu hiệu chỉ đường, nhưng ý tưởng nền tảng là khi xét trong bối cảnh tổng thể, thông tin trên các dấu hiệu chỉ đường phải đầy đủ và nhất quán. Hãy xem xét việc này theo cách khác, chúng ra sẽ thấy rằng không cần thiết để mà mỗi giao lộ có dấu hiệu cho mọi đích đến. Các dấu hiệu có thể chỉ ra những con đường mặc định miễn sao tất cả những dấu hiệu chỉ đường cụ thể chỉ ra con đường ngắn nhất, và rẽ nhánh cho con đường ngắn nhất tới các đích điện tử đánh dấu. chỉ cần một vài ví dụ sẽ giải thích cho ta những cách để đạt được sự nhất quán.

Trong trường hợp cực đoan nhất, hãy xét mô hình các con đường có hình dạng sao mà trong đó chỉ có duy nhất một con đường dẫn đến mỗi làng, và tất cả các con đường này gặp nhau tại một điểm. Để bảo đảm sự nhất quán, dấu hiệu tại giao lộ trung tâm phải chứa thông tin về tất cả các đích có thể có. Trong trường hợp cực đoan thứ hai, hãy hình dung một nhóm bất kỳ các con đường có các dấu hiệu chỉ đường tại tất cả các giao lộ liệt kê tất cả các đích có thể có. Để bảo đảm sự nhất quán, thì tại giao lộ bất kỳ nếu dấu hiệu cho đích D chỉ đến con đường R, thì không thể con đường nào khác R lại là con đường ngắn nhất đi đến D.

Cả hai kiến trúc cực đoan này đều không làm việc tốt đối với hệ định tuyến Internet. Đối với trường hợp hợp đầu, cách tiếp cận giao lộ trung tâm không phù hợp bởi vì không có máy chủ nào đủ mạnh và nhanh để làm bộ định tuyến trung tâm cho mọi giao thông đi quan nó. Trường hợp thứ hai, trong thực tế, không thể có bản sao thông tin về tất cả các đích có thể có trong tất cả các bộ định tuyến bởi vì nó đòi hỏi phải nhân bản những khối lượng cực lớn thông tin và thường xuyên phải cập nhật lượng thông tin này bất cứ khi nào có thay đổi xảy ra hay bất cứ khi nào người quản trị cần kiểm tra sự nhất quán. Vì thế, chúng ta cần tìm ra một giải pháp mà cho phép các nhóm quản lý bộ định tuyến địa phương một cách tự quản, có thể thêm vào mạng mới và bộ định tuyến mới mà không phải thay đổi những bộ định tuyến ở xa.

Để hỗ trợ cho việc giải thích một số kiến trúc mô tả sau này, hãy xét một mô hình thứ ba trong đó một nửa số các thành phố nằm về phía đông của trục lộ thành phố và nửa còn lại thuộc về phía tây. Giả sử có một cây cầu duy nhất bắc ngang con sông chia cách đông và tây. Giả sử rằng những người sống ở phía đông không thích những người ở phía tây, vì vậy họ vui lòng cho phép các dấu hiệu chỉ đường liệt kê các đích đến ở phía đông nhưng không hề có các đích đến ở phía tây. Giả sử rằng những người ở phía tây cũng không thích người phía đông (và cũng không có dấu hiệu cho đường phía đông). Việc chỉ đường đi sẽ là nhất quán nếu mọi dấu hiệu chỉ đường ở phía đông liệt kê ra tất cả các đích phía đông một cách tường minh và chỉ ra con đường mặc định đi đến cầu, trong khi đó mọi dấu hiệu chỉ đường ở phía tây liệt kê ra mọi đích đến ở phía tây và chỉ ra con đường mặc định đi đến cầu.

#### *8.2.2.2. Kiến trúc nguyên thuỷ của Internet*

Hầu hết sự phát triển của việc định tuyến và các giao thức nhân bản tuyến đường đều có được từ khi mạng Internet toàn cầu phát triển. Trước đây, khi TCP/IP được phát triển lần đầu, các đơn vị tham gia nghiên cứu điện tử nối vào mạng ARPANET, đây là backbone của Internet. Trong quá trình thử nghiệm ban đầu, mỗi đơn vị quản lý các bảng định tuyến và thiết lập các tuyến đường tới các

đích khác một cách thủ công. Khi Internet bắt đầu phát triển rộng lớn hơn, thì rõ ràng là việc duy trì các tuyến đường một cách thủ công là không thực tế; cần có những cơ chế tự động.

Những người thiết kế đã chọn một kiến trúc cho bộ định tuyến bao gồm một tập hợp nhỏ các bộ định tuyến trung tâm để lưu trữ đầy đủ thông tin về tất cả các đích có thể có, và một tập hợp lớn hơn gồm các bộ định tuyến bên ngoài để lưu trữ thông tin bán phần. So sánh tương tự với ví dụ vừa rồi, điều này giống như dành riêng một tập hợp nhỏ các giao lộ trung tâm, nơi đây các dấu hiệu chỉ đường liệt kê ra tất cả các đích trung tâm mạng, và cho phép các giao lộ bên ngoài chỉ liệt kê các đích địa phương. Miễn sao con đường mặc định tại mỗi giao lộ bên ngoài chỉ tới một trong những giao lộ trung tâm, người du lịch cũng sẽ có thể đến đích của họ. Ưu điểm của việc sử dụng thông tin bán phần trong các bộ định tuyến bên ngoài là nó cho phép người quản trị địa phương có thể quản lý các thay đổi cấu trúc ở địa phương mà không ảnh hưởng đến những phần khác của Internet. Khuyết điểm chính của nó là có thể dẫn đến sự không nhất quán (trong các bảng định tuyến). Trong thông tin hợp xấu nhất, một lỗi trong bộ định tuyến bên ngoài có thể dẫn đến việc không đi đến đích được.

Chúng ta có thể tóm tắt các ý tưởng này như sau:

Bảng định tuyến trong một bộ định tuyến chứa một phần thông tin các đích đến. Việc định tuyến sử dụng thông tin bán phần cho phép các mạng tự chủ trong việc thực hiện các thay đổi định tuyến địa phương, nhưng cũng có thể gây nên không nhất quán mà có thể làm cho một số đích không còn con đường đi đến.

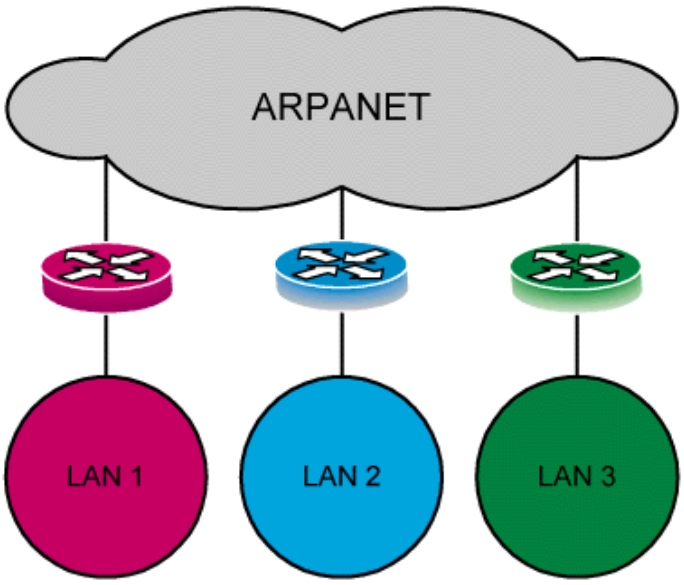
Sự không nhất quán trong số các bảng định tuyến thường xuyên xuất hiện từ các lỗi trong những thuật giải để tính toán xây dựng lên các bảng định tuyến, từ các dữ liệu không chính xác cho những thuật giải này, hay từ các lỗi xảy ra trong khi truyền kết quả đến các bộ định tuyến khác. Người thiết kế giao thức đang tìm kiếm các cách để giới hạn ảnh hưởng của các lỗi, mà chủ đích là duy trì sao cho tất cả các tuyến định tuyến đều thống nhất trong mọi lúc mọi nơi. Nếu vì lý do nào đó, các tuyến đường trở nên không nhất quán, giao thức cũng phải đủ mạnh để nhận ra và sửa chữa các lỗi này một cách nhanh chóng. Quan trọng nhất là, các giao thức phải được thiết kế để giới hạn ảnh hưởng của các lỗi.

*8.2.2.3. Bộ định tuyến chủ chốt*

Nói một cách đơn giản, các bộ định tuyến Internet ban đầu có thể được phân thành hai nhóm, một tập hợp nhỏ các bộ định tuyến chủ chốt được điều khiển bởi trung tâm điều hành mạng Internet (Internet Network Operation Center: INOC) và một tập hợp lớn hơn gồm các bộ định tuyến không chủ chốt được điều hành bởi

các mạng (đơn vị) riêng lẻ. Hệ thống chủ chốt được thiết kế để cung cấp sự tin cậy, nhất quán, các tuyến đường định tuyến hợp thức cho tất cả các đích có thể; nó là chất kết dính để giữ lại với nhau và mà cho sự kết nối toàn cầu trở thành điều khiển thực. Mỗi đơn vị được gán cho một địa chỉ mạng phải sắp xếp để thông báo địa chỉ đó cho hệ thống chủ chốt. Các bộ định tuyến chủ chốt thông tin liên lạc lẫn nhau, vì vậy chúng ta thể bảo đảm rằng thông tin chúng chia xẻ với nhau là nhất quán. Bởi vì luôn có trung tâm giám sát và điều khiển các bộ định tuyến chủ chốt, tạo ra sự rất đáng tin cậy của chúng.

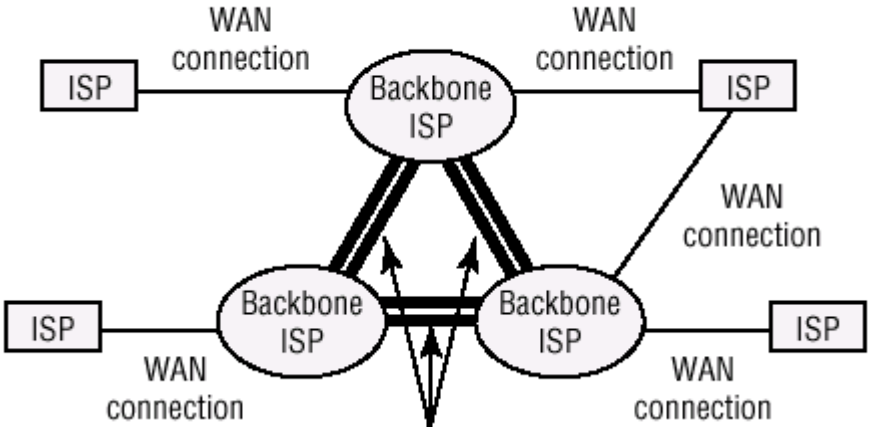
Để hiểu được đầy đủ hệ thống các bộ định tuyến chủ chốt, chúng ta cần nhớ là rằng Internet đã phát triển cùng với mạng diện rộng, khia mà ARPANET đã tồn tại rồi. Khi đã có một số kinh nghiệm về Internet, những người thiết kế đã lấy ARPANET là backbone chính để xây dựng. Như thế một phần lớn động lực cho sự ra đời của hệ các bộ định tuyến chủ chốt đã đến từ mong muốn kết nối mạng cục bộ với mạng trung tâm ARPANET. Hình 8.9. Minh hoạ ý tưởng này.




*Hình 8.9: Các router chủ chốt*

Để hiểu được tại sao một kiến trúc như thế không đưa bản thân nó tới sự định tuyến với bán phần thông tin, hãy giả sử rằng một Internet lớn bao gồm hoàn toàn những mạng cục bộ, mỗi mạng được nối vào một mạng backbone thông qua bộ định tuyến. Chúng ta cũng hãy hình dung rằng một số bộ định tuyến dựa vào tuyến đường mặc định. Bây giờ chúng ta hãy xét con đường mà datagram đi qua. Tại đơn vị nguồn, bộ định tuyến địa phương kiểm tra xem có chứa con đường đi cụ thể đến đích không; nếu không có, nó gửi datagram đi theo con đường mặc định. Đối với tất cả các datagram mà bộ định tuyến không có con đường đi cụ thể cho nó, chúng đều được chuyển tới một định tuyến mặc định bất kể đích đến cuối cùng của chúng. Tương tự như vậy đối với bộ định tuyến kế tiếp trên đường đi của datagram, nó chuyển datagram đến đích hoặc lại gửi theo tuyến đường mặc định.

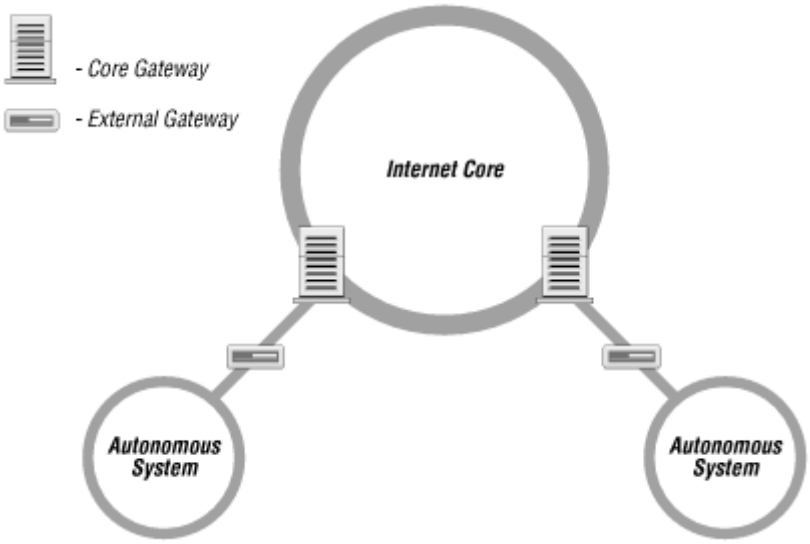
Để bảo đảm sự nhất quán toàn cầu, dây chuyền của các tuyến đường mặc định phải đi qua mọi bộ định tuyến theo một chu trình, như trong hình 8.10. Như thế, kiến trúc này đòi hỏi tất cả các đơn vị địa phương cùng phối hợp tuyến đường mặc định của chúng. Hơn thế nữa, sự phụ thuộc vào các định tuyến mặc ược có thể không hiệu quả ngay cả khi chúng nhất quán. Như trình bày trong hình 8.10, trong trường hợp xấu nhất datagram sẽ đi qua tất cả các bộ định tuyến khi nó di chuyển từ nguồn đến đích thay vì đi trực tiếp qua backbone.



Các tuyến đường mặc định

*Hình 8.10: Các tuyến đường mặc định*

Để tránh sự kém hiệu quả do các tuyến đường mặc định gây ra, những người thiết kế Internet đã bố trí để tất cả các bộ định tuyến chủ chốt trao đổi với nhau, thông tin định tuyến để cho mỗi cái sẽ có đầy đủ về các định tuyến tối ưu đi đến tất cả các đích. Bởi vì mỗi bộ định tuyến chủ chốt biết đường đi đến tất cả các đích, nó không cần đến tuyến đường mặc định. Nếu địa chỉ đích trên datagram không có trong bảng định tuyến của bộ định tuyến chủ chốt, thì bộ định tuyến sẽ phát sinh một thông điệp "ICMP không thể đến đích" và huỷ bỏ datagram. Như vậy, cách thiết kế này bộ định tuyến chủ chốt đã tránh sự kém hiệu quả bằng cách loại bỏ tuyến đường mặc định.




*Hình 8.11: Các hệ tự quản nối vào hạt nhân của Internet*

Hình 8.11 trình bày khái niệm cơ sở của kiến trúc định tuyến chủ chốt. Hình này cho thấy hệ thống chủ chốt tập trung bao gồm một hay nhiều bộ định tuyến, và một tập hợp các bộ định tuyến bên ngoài tại mỗi đơn vị. Các bộ định tuyến bên ngoài lưu trữ thông tin về các đích địa phương và sử dụng tuyến đường mặc định để gửi các datagram, dành cho các đơn vị khác, đến bộ định tuyến chủ chốt.

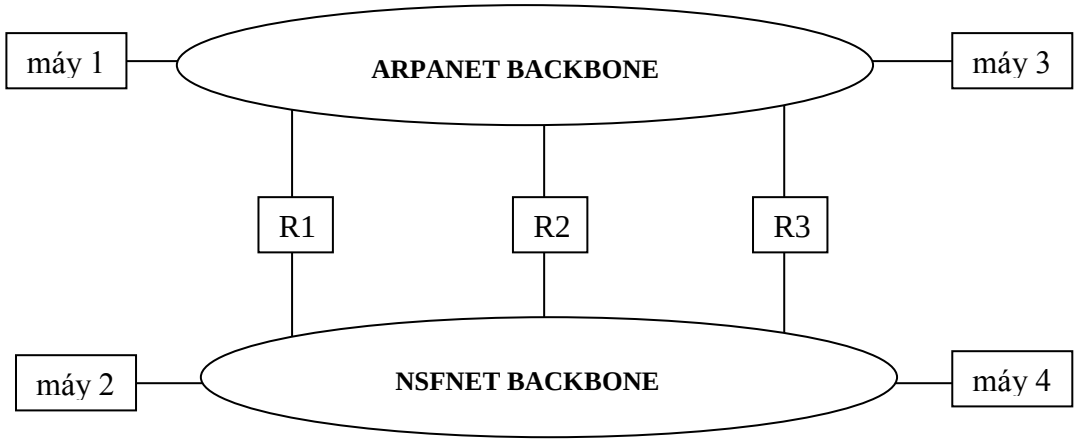
Mặc dù kiến trúc được đơn giản hoá minh hoạ trong hình 8.11 là dễ hiểu, nhưng lại không thực tế vì ba lý do. Trước hết, Internet đã vượt ra khỏi một backbone duy nhất, được quản lý tập trung. Cấu hình của nó đã trở nên phức tạp và các giao thức cần để duy trì tính nhất quán giữa các bộ định tuyến chủ chốt đã không còn bình thường nữa. Thứ hai, không phải mỗi đơn vị đều có thể có một bộ định tuyến chủ chốt kết nối với backbone, vì vậy cần có thêm những giao thức và cấu trúc định tuyến. Thứ ba, bởi vì các bộ định tuyến chủ chốt đều giao tiếp lẫn nhau để bảo đảm sự nhất quán của thông tin định tuyến, kiến trúc chủ chốt không thể đáp ứng nổi kích thước mạng Internet ngày càng rộng lớn đến vô cùng.

*8.2.2.4. Kiến trúc chủ chốt và các backbone đồng đẳng*

Sự ra đời của backbone NSFNET trong Internet đã tạo thêm sự phức tạp đối với cấu trúc định tuyến. Từ quan điểm của hệ định tuyến chủ chốt, sự kết nối vào NSFNET không khác biệt gì hơn kết nối vào những đơn vị khác. NSFNET nối vào backbone ARPANET thông qua một bộ định tuyến duy nhất tại trung tâm Pittsburgh. Các bộ định tuyến trong NSFNET biết về các đích cục bộ và đã sử dụng tuyến đường mặc định để gửi đi tất cả các dữ liệu không phải NSFNET đến hệ thống định tuyến chủ chốt thông qua bộ định tuyến ở Pittsburgh.

Khi NSFNET phát triển và đã trở thành phần chính yếu của Internet, thì rõ ràng rằng kiến trúc định tuyến chủ chốt không còn đủ đáp ứng. Sự thay đổi khái

niệm quan trọng nhất đã xuất hiện khi nhiều kết nối đã được thêm vào giữa các backbone ARPANET và NSFNET. Chúng ta nói rằng cả hai đã trở nên các mạng backbone đồng đẳng, hay gọi tắt là đồng đẳng. Hình 8.12. minh hoạ kết quả của cấu hình đồng đẳng.




*Hình 8.12: Các backbone đồng đẳng*

Để hiểu những khó khăn của việc định tuyến IP giữa các backbone đồng đẳng, chúng ta hãy xét tuyến đường từ máy 3 đến máy 2 trong hình 8.12. hãy giả sử về mặt địa lý, máy 3 ở miền tây đã nối vào backbone NSFNET, còn máy 2 ở miền đông đã nối vào backbone ARPANET. Khi thiết lập các tuyến đường giữa máy 3 và máy 2, những nhà quản lý phải quyết định hoặc là (a) chuyển dữ liệu từ máy 3 thông qua bộ định tuyến miền tây. R1, và rồi đi qua backbone ARPANET, hoặc là (b) chuyển dữ liệu từ máy 3 đi qua backbone NSFNET, thông qua bộ định tuyến miền trung tây, R2, và rồi đi qua backbone ARPANET đến máy 2, hoặc là (c) chuyển dữ liệu đi qua backbone NSFNET, thông qua bộ định tuyến miền tây, R3, và rồi đến máy 2. Cũng có con đường khác dài hơn, dữ liệu có thể di chuyển từ máy 3 đi qua bộ định tuyến miên tây, đi qua backbone ARPANET đến máy 2. Tuyến đường như thế có thể điện tử và cũng có thể không điện tử khuyến khích sử dụng, tuỳ vào các chính sách sử dụng mạng và khả năng của một số bộ định tuyến.

Đối với hầu hết các cấu hình backbone đồng đẳng, đường đi giữa một cặp máy tính gần nhau về mặt địa lý sẽ đi theo tuyến đường ngắn nhất. Độc lập với những tuyến đường đã được chọn cho các giao dịch xuyên quốc gia. Lấy ví dụ, giao dịch từ máy 3 đến máy 1 sẽ đi qua bộ định tuyến miền tây bởi vì nó giảm thiểu khoảng cách trên cả hai backbone.

Lý thuyết dễ hiểu, nhưng lại rất phức tạp khi cài đặt bởi vì hai lý do. Trước hết, mặc dù thuật giải định tuyến IP chuẩn sử dụng phân mạng của một địa chỉ IP để chọn con đường, khi việc định tuyến tối ưu trong một kiến trúc backbone đồng đẳng đòi hỏi những tuyến đường riêng cho từng máy. Với ví dụ của chúng ta ở trên

bảng định tuyến trong máy 3 cần có những con đường khác nhau đối với máy 1 và máy 2, mặc dù cả hai máy 1 và 2 đều nối vào backbone ARPANET. Thứ hai, người quản lý của hai backbone phải thống nhất với nhau để duy trì những tuyến đường nhất quán trong tất cả các bộ stream hoặc xảy ra khi các tuyến đường trong một nhóm các bộ định tuyến tạo thành một vòng tròn.

Chúng ta có thể tóm tắt như sau:

Kiến trúc định tuyến chủ chốt quy định một tập hợp chủ chốt bộ định tuyến trung tâm phục vụ như kho lưu trữ thông tin về tất cả chủ chốt đích trong Internet. Hệ chủ chốt hoạt động tốt nhất trong những Internet chỉ có một backbone được quản lý tập trung. Việc mở rộng cấu hình ra nhiều backbone làm phức tạp thêm việc định tuyến; việc phân chia kiến trúc chủ chốt sao cho tất cả bộ định chủ chốt tuyến sử dụng tuyến đường mặc định có thể dẫn đến chủ chốt vòng lặp định tuyến.

*8.2.2.5. Tự động nhân bản định tuyến*

Chúng ta đã nói rằng các hệ Internet chủ chốt ban đầu đã tránh được các tuyến đường mặc định bởi vì nó nhân bản thông tin đầy đủ về tất cả các đích đến mọi bộ định tuyến chủ chốt. Nhiều cơ quan bây giờ cũng sử dụng mô hình tương tự các bộ định tuyến trong cơ quan chạy những chương trình để trao đổi với nhau thông tin định tuyến. Các đoạn tiếp theo sẽ tìm hiểu hai loại thuật giải cơ bản để tính và nhân bản thông tin định tuyến, và sử dụng giao thức định tuyến chủ chốt ban đầu để minh hoạ một trong những thuật giải này. Phần tiếp theo mô tả một giao thức sử dụng kiểu khác của thuật giải.

Thoạt nhìn, dường như các cơ chế tự động nhân bản định tuyến là không cần thiết, đặc biệt đối với các Internet nhỏ. Tuy nhiên, Internet không phải luôn luôn cố định. Các kết nối có thể bị hỏng và được thay thế sau đó. Các mạng có thể bị quá tải tại thời điểm nào đó và có thể sau đó lại ít được sử dụng. Mục đích của các cơ chế nhân bản tuyến đường không chỉ đơn thuần là để tìm một tập hợp các tuyến đường, nhưng để liên tục cập nhật thông tin. Con người không thể nhanh chóng đáp ứng với những thay đổi; phải sử dụng các chương trình máy tính. Như thế, khi chúng ta nghĩ về việc nhân bản tuyến đường, điều quan trọng cần lưu ý là cách đáp ứng năng động của các giao thức và các thuật giải.

## 8.3. Các giải thuật định tuyến cơ bản

### 8.3.1. Định tuyến theo Vector khoảng cách (Pellman Ford)

#### *8.3.1.1. Khái niệm định tuyến theo vector khoảng cách*

Thuật giải vector khoảng cách (còn được gọi là khoảng cách vector, Ford Fulkeson, bellman Ford, hay Bellman, với các tên sau cùng chính là tên của những nhà nhiên cứu đã tìm ra thuật giải) để chỉ một các thuật giải mà bộ định tuyến sử dụng để nhân bản thông tin định tuyến. Ý tưởng trong thuật giải vector khoảng cách rất là đơn giản. Bộ định tuyến duy trì một danh sách của tất cả các tuyến đường đã biết trong một bảng. Khi bắt đầu hoạt động, bộ định tuyến khởi động bằng định tuyến của nó trong đó mỗi dòng dành cho một mạng được kết nối trực tiếp. Mỗi dòng trong bảng xác định một mạng đích và thông tin về khoảng cách đến mạng đó, thường được theo số trạm (sẽ được định nghĩa một cách chính xác sau này)

Một cách đình kỳ, mỗi bộ định tuyến gửi đi một bản sao của bảng định tuyến của nó đến bộ định tuyến bất kỳ nào khác mà nó có thể đến được trực tiếp.

Thuật ngữ vector khoảng cách có được từ thông tin được gửi trong các thông diệp theo định kỳ. Một thông điệp chứa danh sách các cặp (V, D), với V xác định một đích (được gọi là vector), và D là khoảng cách đi đến đích đó. Lưu ý rằng thuật giải vector khoảng cách cho biết các tuyến đường của "người đầu tiền" (nghĩa là, chúng ta xem như là bộ định tuyến ra thông báo tôi có thể đi đến đích V với khoảng cách là D"). Trong cách thiết kế đó, tất cả các bộ định tuyến phải tham gia trong việc trao đổi vector khoảng cách để có được các tuyến đường được nhất quán và hiệu quả.

Mặc dù thuật giải vector khoảng cách dễ cài đặt, chúng cũng có những nhược điểm lớn. Trong môi trường hoàn toàn tĩnh, ít thay đổi, thuật giải vector khoảng cách nhân bản các tuyến đường chuyển đến tất cả các đích. Tuy nhiên một khi các tuyến đường thay đổi nhanh chóng các phép tính toán có thể không ổn định. Khi một tuyến đường thay đổi (nghĩa là, một kết nối mới đã xuất hiện hay một cái cũ bị hỏng), thông tin sẽ nhân bản từ bộ định tuyến này đến bộ định tuyến khác một cách chậm chạp. trong khoảng thời gian đó, có những bộ định tuyến chứa đựng thông tin định tuyến không chính xác.

#### *8.3.1.2. Giao thức Gateway to Gateway (GGP)*

Các bộ định tuyến chủ chốt ban đầu đã sử dụng giao thức vector khoảng cách có tên Gateway Protocol (GGP lưu ý rằng, mặc dù những nhà sản xuất đã đưa ra

thuật ngữ bộ định tuyến IP, các nhà khoa học trước đây đã sử dụng thuật ngữ IP gateway). Mặc dù GGP chỉ xử lý các tuyến đường phần lớp và không còn là một phần của chuẩn TCP/IP, nó cung cấp cho ta một ví dụ cụ thể về việc định tuyến theo vector khoảng cách. GGP được thiết kế để di chuyển thông điệp GGP có một phần đầu được định dạng cố định để xác định kiểu thông điệp và định dạng của các vùng còn lại. Bởi vì, chỉ có các bộ định tuyến chủ chốt tham gia trongGGP, và bởi vì các bộ định tuyến cung cấp được điều hành bởi đơn vị chỉ huy trung ương, những bộ định tuyến khác không thể can thiệp vào sự thay đổi.

Hệ chủ chốt ban đầu được bố trí để cho phép những bộ định tuyến mới được thêm vào mà không phải điều chỉnh các bộ định tuyến có sẵn. Khi một bộ định tuyến mới, được thêm vào hệ chủ chốt, nó được gán một hay nhiều ”người láng giềng” chủ chốt, mà nó sẽ thông tin liên lạc. Những "người láng giềng" chủ chốt, đã nhân bản thông tin định tuyến lẫn nhau. Vì thế, bộ định tuyến mới chỉ cần thông báo cho "người láng giềng" vì thế, bộ định tuyến mới chỉ cần thông báo cho "người láng giềng" về các mạng mà nó có thể đến được; chúng ta sẽ cập nhật bảng định tuyến của chúng và nhân bản thông tin mới này đi đến các bộ định tuyến khác.

GGP đúng là một giao thức vector khoảng cách. Thông tin mà các bộ định tuyến trao đổi với GGP bao gồm một tập hợp các cặp (N,D), với N là một địa chỉ mạng IP, và D là khoảng cách được tính theo số trạm. Chúng ta nói rằng bộ định tuyến sử dụng GGP để quảng bá những mạng mà nó có thể đến được và chi phối để đi đến đó.

GGP đo khoảng cách theo số trạm các bộ định tuyến, trong đó một bộ định tuyến điện tử định nghĩa là zero trạm từ những mạng nối trực tiếp, một trạm từ các mạng có thể được thông qua một bộ định tuyến khác, … như thế, số lượng trạm dọc theo con đường từ một nguồn tới một đích chính là số lượng bộ định tuyến mà một datagram đi qua trên con đường đó. Một sự thật hiển nhiên là việc sử dụng số lượng trạm để tính con đường ngắn nhất không phải lúc nào cũng cho ra kết quả như mong muốn. Lấy ví dụ, một con đường có ba trạm đi qua ba LAN có thể nhanh hơn nhiều so với con đường có hai trạm mà đi qua hai định tuyến tuần tự tốc độ chậm. Nhiều bộ định tuyến sử dụng số lượng trạm giả tạo khá lớn để dành cho những mạng tốc độ chậm

*8.3.1.3. Thừa số khoảng cách*

Giống như hầu hết các giao thức định tuyến. GGP sử dụng nhiều kiểu thông điệp, mỗi cái có định dạng và mục đích riêng biệt. Một vùng trong phần đầu thông điệp chứa một mã để xác định kiểu thông điệp cụ thể nơi nhận mã này để quyết định cách xử lý thông điệp. Lấy ví dụ, trước khi hai bộ định tuyến có thể trao đổi thông tin định tuyến, chúng phải thiết lập việc thông tin liên lạc, và một vài kiểu

thông điệp được sử dụng cho mục đích này. Kiểu thông điệp cơ bản nhất trong GGP cũng là cơ bản đối với bất kỳ giao thức vector khoảng cách nào là: sự cập nhật việc định tuyến mà được sử dụng để trao đổi thông tin định tuyến.

Về mặt khái niệm, cập nhật sự định tuyến bao gồm danh sách các cặp, với mỗi cặp chứa một địa chỉ mạng IP và khoảng cách đến mạng đó. Tuy nhiên, trong thực tế, nhiều giao thức định tuyến sắp xếp lại thông tin để cho các thông điệp có kích thước nhỏ. Cụ thể, có một vài kiến trúc bao gồm việc bố trí tuyến tính các mạng và bộ định tuyến. Thay vì thế, hầu hết lại có cấu trúc, các giá trị khoảng cách trong một lần cập nhật là những số nhỏ và cùng một agv thường có khuynh hướng được lặp lại. Để giảm bớt kích thước thông điệp, các giao thức định tuyến thường sử dụng kỹ thuật này tránh việc gửi các bản sao của cùng một giá trị khoảng cách, mỗi giá trị khoảng cách các cặp được sắp xếp theo thứ tự khoảng cách. Thay vì thế, danh sách các cặp được sắp xếp theo thứ tự khoảng cách, mỗi giá trị khoảng cách chỉ được thể hiện một lần, và theo sau là các mạng có để đi đến được với khoảng cách đó.

### 8.3.2. Định tuyến theo trạng thái liên kết (SPF)

Khuyết điểm chính của thuật giải vector khoảng cách là nó không chọn lọc được thông tin. Bên cạnh vấn đề đáp ứng chậm đối với những thay đổi đã đề cập trước đây, thuật giải này còn đòi hỏi phải trao đổi thông điệp lớn. Bởi vì mỗi cặp nhật cho việc định tuyến chứa một dòng cho mọi mạng, nên kích thước của thông điệp tỷ lệ thuận với tổng số mạng trong Internet. Hơn nữa, bởi vì giao thức vector khoảng cách cách yêu cầu mọi bộ định tuyến đều tham dự, lượng thông tin trao đổi có thể rất lớn.

Một giải pháp khác đối với thuật giải vector khoảng cách là một lớp các thuật giải có tên là trạng thái liên kết, hay con đường ngắn nhất trước tiên (Shortesst Path First – SPF). Cách gọi tên này không được đúng lắm bởi vì tất cả mọi thuật giải định tuyến đều đi tìm con đường ngắn nhất). Thuật giải SPF đòi hỏi mỗi bộ định tuyến tham dự phải có đầy đủ thông tin về cấu hình. Cách dễ nhất để xem xét thông tin về cấu hình là hãy tưởng tượng rằng mọi bộ định tuyến có một bản đồ trình bày tất cả những bộ định tuyến và những mạng mà chúng kết nối vào. Nói một cách hình tượng, các bộ định tuyến tương ứng với các nút (node) trong đồ thị và các mạng nối các bộ định tuyến tương ứng với các cạnh. Có một cạnh (liên kết) giữa hai nút nếu và chỉ nếu các bộ định tuyến tương ứng có thể trực tiếp thông tin liên lạc.

Thay vì gửi đi các thông điệp có chứa danh sách các đích đến, một bộ định tuyến tham gia trong thuật giải SPF thực hiện hai công việc lân cận. Nói theo thuật

ngữ đồ thị, hai bộ định tuyến được gọi là lân cận nếu chúng cùng chia sẻ một cạnh chung. Thứ hai, nó định kỳ nhân bản thông tin trạng thái của liên kết đến tất cả các bộ định tuyến khác.

Để kiểm tra trạng thái của bộ định tuyến lân cận (có kết nối trực tiếp), một bộ định tuyến sẽ trao đổi định kỳ các thông điệp ngắn để hỏi xem bộ định tuyến lân cận có còn hoạt động và có thể đi đến được không. Nếu các nơi lân cận đáp lời, nghĩa là đường nối giữa chúng vẫn còn hoạt động. Nếu không, đường nối được xem như "bị ngắt". Để thông báo cho tất cả các bộ định tuyến khác, mỗi bộ định tuyến sẽ định kỳ phát đi một thông điệp để liệt kê ra trạng thái của mỗi đường nối của nó. Một thông điệp trạng thái không xác định tuyến đi – nó chỉ đơn giản cho biết xem có thể thông tin liên lạc giữa các cặp bộ định tuyến hay không. Phần mềm giao thức trong các bộ định tuyến sẽ bố trí để chuyển phát một bản sao của mỗi thông điệp về trạng thái liên kết đến tất cả các bộ định tuyến tham gia.

Bất cứ khi nào có thông điệp về trạng thái liên kết gửi đến, bộ định tuyến sử dụng thông tin này để cập nhật bản đồ của nó về tin, bằng cách đánh dấu những liên kết nào còn hoạt động hay hết hoạt động. Bất cứ khi nào trạng thái liên kết thay đổi, bộ định tuyến tính lại các tuyến đi bằng cách áp thuật giải nổi tiếng về đường đi ngắn nhất của Dijkstra để có được hình ảnh mới của đồ thị. Thuật giải của Dijkstra tính các đường đi ngắn nhất đến tất cả các đích kể từ một điểm xuất phát.

Một trong những ưu điểm chính của các thuật giải SPF là mỗi bộ định tuyến tính các tuyến đường một cách độc lập, cùng sử dụng dữ liệu trạng thái ban đầu; chúng không phụ thuộc vào việc tính toán của các máy trung gian. Bởi vì các thông điệp trạng thái liên kết được giữ nguyên khi nhân bản, người ta dễ dàng tìm ra các vấn đề lỗi. Cuối cùng, bởi vì các thông điệp về trạng thái liên kết chỉ chuyển tải thông tin về các kết nối trực tiếp từ một bộ định tuyến, kích thước của nó không phụ thuộc vào số lượng của mạng trong Internet. Như thế, các thuật giải SPF tốt hơn các thuật giải vector khoảng cách.

### 8.3.3. Đảm bảo tính tin cậy cho các giao thức định tuyến

Hầu hết các giao thức sử dụng việc chuyển tải theo kiểu connectionles. Lấu ví dụ, GGP đòng gói vào UDP (cũng có những, ngoại lệ). Cả hai IP và UDP đều hỗ trợ cùng một loại dịch vụ: các thông điệp có thể bị mất, trì hoãn trùng lặp, hư hỏng, và chuyển phát không đúng thứ tự. Như thế, một giao thức định tuyến mà sử dụng chúng phải có cách bù đắp các sai sót này.

Các giao thức định tuyến sử dụng một số kỹ thuật để xử lý các vấn đề phát chuyển. Checksum được sử dụng để chất lượng việc dữ liệu bị hư hỏng. Việc mất

dữ liệu được xử lý bởi trạng thái mềm (chúng ta nhớ lại rằng trạng thái mềm dựa vào bộ đếm thời gian để loại bỏ thông tin cũ thay vì đợi thông điệp từ nguồn) hay thông qua kỹ thuật lời đáp và truyền lại. Lấy ví dụ, trong GGP sử dụng một mô hình lời đáp mở rộng, trong đó nơi nhận có thể gửi trả lại một lời đáp tích cực hay tiêu cực.

Để xử lý việc chuyển phát không theo thứ tự và sự tương ứng, xảy ra khi một thông điệp cũ đến đích, những giao thức định tuyến thường sử dụng các số thứ tự khởi đầu khi bắt đầu việc thông tin liên lạc. Sau đó, đầu kia phải đáp lời theo số thứ tự này. Sau khi khởi động việc trao đổi, mỗi trạm đích sẽ chứa số thứ tự kế tiếp, để cho phép nơi nhận biết các thông điệp sẽ chứa số thớ tự này. Sau khi khởi động việc trao đổi. Mỗi thông điệp sẽ chứa số thứ tự này. Sau khi khởi động việc biết các thông điệp sẽ chứa số thứ tự kế tiếp, để cho phép nơi nhận biết các thông điệp có được gửi đến theo thứ tự không. Trong chương sau, chúng ta sẽ xét một ví dụ về giao thức định tuyến có sử dụng thông tin trạng thái mềm.

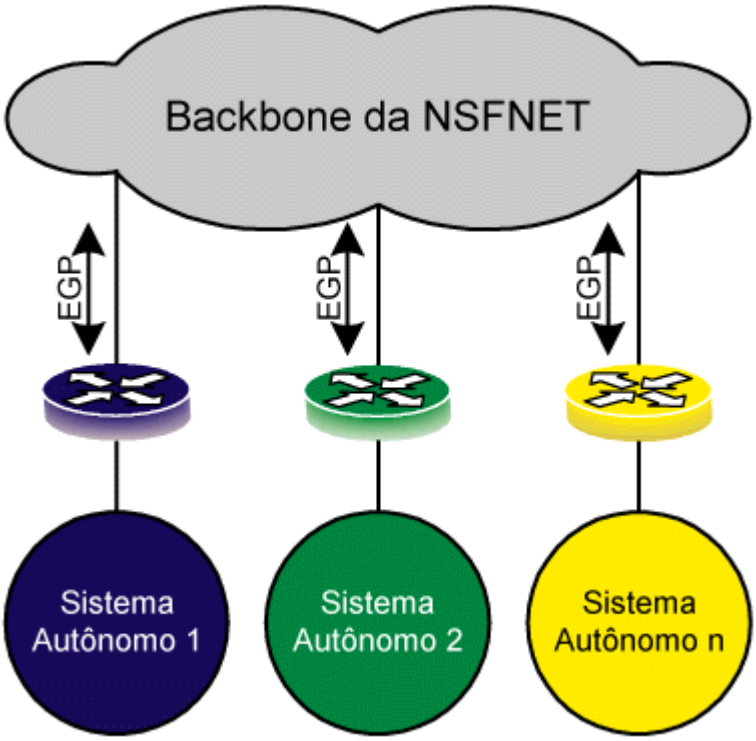
## 8.4. Định tuyến giữa các hệ tự quản và giao thức BGP

### 8.4.1. Khái niệm hệ tự quản

Với các mục đích về định tuyến, một nhóm các mạng và các bộ định tuyến được kiểm soát bởi một đơn vị quản trị được gọi là một hệ tự quản. các bộ định tuyến bên trong một hệ tự quản được tự do chọn trước cơ chế của riêng nó trong việc phát hiện, nhân bản, kiểm định, và kiểm tra tính nhất quán của các tuyến đường. Lưu ý rằng, theo định nghĩa này, các bộ định tuyến chủ chốt ban đầu của Internet hình thành nên một hệ tự quản. Mỗi thay đổi của các giao thức định tuyến bên trong hệ tự quản chủ chốt được thực hiện mà không gây ảnh hướng đến các bộ định tuyến trong các hệ tự quản khác. Trong chương trước, chúng ta đã nói rằng hệ Internet chủ chốt ban đầu đã sử dụng GGP để thông tin liên lạc, và thế hệ sau này đã sử dụng SPREAD. Cuối cùng thì, các ISP đã phát triển mạng backbone của riêng chúng và sử dụng các giao thức mới hơn.

### 8.4.2. Từ hệ chủ chốt đến hệ tự quản độc lập

Về mặt khái niệm, ý tưởng về hệ tự quản là đơn giản và là sự tổng quát hoá tự nhiên của kiến trúc Internet ban đầu, như trình bày trong hình 8.13, với các hệ tự quản thay cho các mạng cục bộ. Hình 8.13. Minh hoạ ý tưởng này.




*Hình 8.13: Các hệ tự quản nối vào hạt nhân của Internet*

Để có thể đi đến được các mạng bị che khuất bên trong những hệ tự quản thông qua Internet, mỗi hệ tự quản phải thông báo cho các hệ tự quản khác về những mạng của mình. Thông báo này có thể được gửi đến một hệ tự quản bất kỳ. Tuy nhiên, trong một kiến trúc tập trung (hệ chủ chốt), điều cốt yếu chính là mỗi hệ tự quản truyền thông tin đến một trong những bộ định tuyến trong hệ tự quản chủ chốt.

Dường như định nghĩa của chúng ta về một hệ tự quản còn mơ hồ, nhưng trong thực tế biên giới giữa các hệ tự quản phải được xác định chính xác để cho phép các thuật giải tự động hoá có thể thực hiện các quyết định trong việc định tuyến. Lấy ví dụ, một hệ tự quản được quản lý bởi một công ty khác mặc dù chúng ta có thể có kết nối trực tiếp. Để cho các thuật giải định tuyến tự động có thể có kết nối trực tiếp. Để cho các thuật giải định tuyến tự động có thể phân biệt được các hệ thống tự quản, từ đơn vị quản trị trung tâm. Khi các bộ định tuyến trong hai hệ tự quản trao đổi thông tin định tuyến, các giao thức sẽ bố trí để cho thông điệp chuyển tải mã số hệ tự quản của hệ thống mà mỗi bộ định tuyến đại diện.

Chúng ta có thể tóm tắt ýtưởng này như sau:

Một hệ Internet TCP/IP lớn có thêm cấu trúc để hỗ trợ việc quản lý những biên giới (giữa các mạng): mỗi tuyển tập các mạng và các bộ định tuyến được quản lý bởi một đơn vị quản trị được xem như là môt hệ tự quản; đơn vị này được tự do chọn riêng một kiến trúc định tuyến nội bộ cũng như là các giao thức.

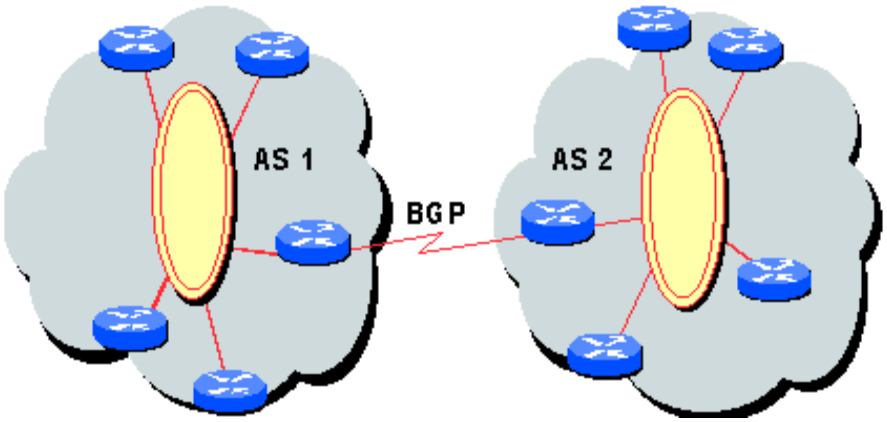
Chúng ta đã nói rằng một hệ tự quản cần tập hợp thông tinvề tất cả các mạng của nó và chỉ định ra một hay nhiều bộ định tuyến để chuyển thông tin đến các hệ tự quản khác. Các phần tiếp theo sẽ trình bày chi tiết của các giao thức mà những bộ định tuyến sử dụng để thông báo về trạng thái thông tin liên lạc của mạng. Những phần sau đó sẽ quay về với những câu hỏi về kiến trúc để tìm hiểu một giới hạn quan trọng mà kiến trúc hệ tự quản áp đặt lên việc định tuyến.

### 8.4.3. Giao thức cổng ngoại (Exterior Gateway Protocol)

Các nhà khoa học máy tính sử dụng thuật ngữ Exterior Gateway Protocol (EGP) để chỉ bất kỳ giao thức nào được sử dụng để chuyển đi thong tin định tuyến giữa hai hệ tự quản (ban đầu, thuật ngữ EGP để chỉ một giao thức đặc biệt mà được sử dụng để thông tin liên lạc với hệ Internet chủ chốt). Hiện tại, chỉ có một giao thức ngoại sử dụng trong hầu hết các Internet TCP/IP. Được biết dưới tên Border Gateway Protocol (BGP), nó đã phát triển quan bốn phiên bản (rất khác nhau). Mỗi phiên bản được đánh số, và phiên bản hiện tại có tên BGP 4. Trong giáo trình này chúng ta sẽ sử dụng thuật ngữ BGP để chỉ BGP 4.

Khi một cặp hai hệ tự quản đồng ý trao đổi thông tin định tuyến, thì mỗi hệ phải chỉ định ra một bộ định tuyến đại diện cho nó mà sẽ cùng sử dụng BGP; hai bộ định tuyến này được gọi là trở thành đồng đẳng BGP với cái kia (mặc dù giao thức này cho phép sử dụng trên máy tính bất kỳ, hầu hết các hệ tự quản sử dụng BGP trên bộ định tuyến).

Bởi vì một bộ định tuyến đang sử dụng BGP phải thông tin liên lạc với bộ định tuyến đồng đẳng trong một hệ tự quản khác, sẽ hợp lý khi chọn một máy mà nằm “cạnh” của hệ tự quản. Do đó, thuật ngữ BGP gọi máy này là cổng biên giới (border gateway) hay bộ định tuyến ở biên (border router). Hình 8.14. minh hoạ ý tưởng này.




*Hình 8.14: BGP trao đổi thông tin giữa các hệ tự quản*

Trong hình này, bộ định tuyến R1 thu thập thông tin về các mạng trong hệ tự quản 1 và thông báo thông tin này cho bộ định tuyến R2 bằng cách sử dụng BGP,

trong khi định tuyến R2 thông báo thông tin bằng cách sử dụng BGP, trong khi định tuyến R2 thông báo thông tin từ hệ tự quản 2.

### 8.4.4. Giao thức BGP

#### *8.4.4.1. Giới thiệu giao thức BGP*

BGP là một giao thức kiểu EGP dùng để liên lạc giữa các hệ thống khác nhau. Một hệ thống dùng BGP trao đổi thông tin trên mạng với các hệ thống dùng BGP khác. Thông tin trao đổi này bao gồm đường dẫn đầy đủ của các hệ thống để chuyển dữ liệu tới các hệ thống khác nhau trên mạng.

Trước hết ta phân loại một gói dữ liệu IP trong một hệ thống như luồng dữ liệu cục bộ. Luồng dữ liệu có thể bắt đầu và kết thúc ngay trong hệ thống đó. Nghĩa là địa chỉ IP nguồn hoặc địa chỉ IP đích xác định một host trong hệ thống đó. Mục đích chính của giao thức BGP trong Internet là để giảm luồng dữ liệu được truyền. Một hệ thống có thể được phân loại theo cách sau:

- Hệ thống chỉ có một liên kết tới một hệ thống khác gọi là Stub AS.
- Một hệ thống Multihomed AS có nhiều liên kết tới các hệ thống khác, nhưng không mang luồng dữ liệu cục bộ và dữ liệu "Transit Traffic".
- Một hệ thống “Transit AS” có nhiều liên kết tới các hệ thống khác dùng để truyền dữ liệu cục bộ và dữ liệu “Transit Tranffic”.

Giao thức định tuyến BGP cho phép định tuyến dựa vào chính sách định tuyến (policy based routing), các chính sách định tuyến được quyết định bởi nhà quản trị và được xác định trong các tệp cấu hình.

Giao thức định tuyến BGP khác với các giao thức định tuyến RIP và OSPF là giao thức BGP dùng TCP như là giao thức vận chuyển của nó. Hai hệ thống dùng BGP thiết lập một liên kết TCP và sau đó trao đổi toàn bộ bảng định tuyến BGP. Từ đó trở đi thì các cập nhật định tuyến được gửi mỗi khi bảng định tuyến thay đổi.

#### *8.4.4.2. Tính năng của BGP*

*Tính chất của giao thức BGP*

BGP có một số tính chất không bình thường. Quan trọng nhất, BGP không phải thuần tuý là một giao thức vector khoảng cách, cũng không phải thuần tuý là một giao thức trạng thái liên lạc. Nó có thể được đặc trưng bởi các tính chất sau đây:

Thông tin liên lạc với hệ tự quản. bởi vì BGP được thiết kế như là một giao thức cổng ngoại, vai trò chính của nó là để cho phép một hệ tự quản thông tin liên lạc với hệ tự khoản khác.

Phối hợp giữa nhiều máy sử dụng BGP. Nếu một hệ tự quản có nhiều bộ định tuyến mỗi thông tin liên lạc với máy đồng đẳng trong một hệ tự quản bên ngoài, BGP có thể được sử dụng để phối hợp giữa các bộ định tuyến để bảo đảm rằng tất cả chúng đều nhân bản thông tin nhất quán.

Nhân bản thông tin về tính liên kết. BGP cho phép một hệ tự quản thông báo các đích mà có thể đi đến được hoặc là trong nó hoặc là đi qua nó, và biết được những thông tin như thế từ hệ tự quản khác.

Mô hình trạm kế. Tương tự như những giao thức định tuyến theo vector khoảng cách, BGP cung cấp thông tin về trạm kế cho mỗi đích đến.

Chính sách hỗ trợ không giống như hầu hết các giao thức vector khoảng cách khi thông báo một cách chính xác các tuyến đường trong bảng định tuyến cục bộ, BGP có thể cài đặt các chính sách mà người địa phương chọn. cụ thể là một bộ định tuyến sử dụng BGP có thể được cấu hình để phân biệt giữa tập hợp các đích có thể đi đến từ các máy tính có thể bên trong hệ tự quản của nó và tập hợp các đích đã thông báo cho những hệ tự quản khác.

Chuyển tải đáng tin cậy. BGP không giống như những giao thức khác khi truyền thông tin định tuyến bởi vì nó giả định rằng việc chuyển tải là đáng tin cậy.Như thế, BGP sử dụng TCP trong mọi việc thông tin liên lạc.

Thông tin về con đường. Cùng với việc xác định các đích có thể đi đến được và mỗi trạm kế, mỗi thông báo BGP còn bao gồm thông tin về con đường; điều này cho phép nơi nhận biết được một các hệ tự quản khác dọc theo con đường đi đến đích.

Hỗ trợ địa chỉ không phân lớp. BGP bảo vệ băng thông của mạng bằng cách cho phép nơi gửi tích luỹ thông tin về tuyến đường và gửi đi chỉ một lần nhưng thể hiện cho nhiều đích đến.

Kiểm định, Xác minh. BGP cho phép nơi nhận quyền xác minh thông điệp, kiểm chứng tên của nơi gửi.

BGP là một giao thức vector khoảng cách, nhưng không giống như RIP (RIP thông báo các host tới một đích), BGP lại liệt kê tuyến đường tới mỗi đích (tuần tự của các số AS tới đích). Một hệ thống AS được định danh bởi một số 16 bit.

BGP phát hiện lỗi liên kết hoặc host tại đầu kia của liên kết TCP bằng cách gửi đi một thông điệp Keepalive tới đầu kia của liên kết một cách đều đặn, thường là khoảng 30 giây thì nó gửi một thông điệp Keepalive. Thông điệp Keepalive ở mức ứng dụng là độc lập với tuỳ chọn TCP Keepalive.

BGP trao đổi thông tin định tuyến dưới dạng các cập nhật. Một tuyến cập nhật bao gồm một địa chỉ mạng, một danh sách các hệ thống AS, thông tin định tuyến đã qua và các thuộc tính đường (path attributes).

Một router dùng BGP gửi và nhận các thông điệp BGP, tạo lập một mối quan hệ láng giềng với các router dùng BGP khác. BGP dùng trong các mạng LAN và WAN như: Ethernet, Token Ring, Sync, Wellflect, Frame Relay, SMDS, X25 (DNN, PDN, PPP), ATM PVC, FDDI, TI, E1, HSSI, PPP... có sử dụng giao thức IP. BGP được đề cập đến trong RFC 1163, 1267 và 1654 tương ứng với các phiên bản BGP2, 3 và 4.

*Các tính năng chính:*

- Hỗ trợ TCP: các router dùng BGP láng giềng kết nối với nhau qua tầng liên kết tin cậy TCP nên không cần thực hiện việc truyền các thông tin cập nhật, truyền lại các gói tin bị mất, các tín hiệu ACK... cần thiết với BGP.
- Router dùng BGP chỉ quảng bá các tuyến nó thực sự dùng. Vì vậy khi một Border Router nhận được nhiều tuyến đến một đích thì nó sẽ chọn lấy một tuyến tốt nhất để thông báo vào trong hệ AS của nó cũng như ra ngoài các Router BGP khác nối vào nó.
- Thuộc tính đường AS: mỗi tuyến BGP chứa thông tin về các hệ AS đã qua. Nếu một router BGP nhìn thấy AS của chính nó trong danh sách, nghĩa là bị lặp tuyến đường thì nó sẽ bỏ qua tuyến đó.
- Chiến lược định tuyến (routing policy): ưu tiên hoặc bỏ qua các cập nhật từ các AS tương ứng.
- BGP phiên bản 2 và 3 chỉ coi các mạng thuộc lớp (A, B, C). Ngược lại, BGP phiên bản 4 không quan tâm đến các lớp địa chỉ, mỗi mạng trong phần thông tin về tầng mạng có thể gửi tới NLRI (Network Layer Reachability Information) của gói cập nhật chứa một số chỉ độ dài netmask. Siêu mạng (Supernet) hỗ trợ khả năng định tuyến liên vùng không phân lớp (CIDR Classless InterDomain Routing), cho phép giảm thiểu kích thước bảng định tuyến bằng cách gộp các tuyến đến các mạng nhỏ (subnet) thành một tuyến cho siêu mạng (supernet).
- Tuyến có trọng số vô cùng sẽ bị bỏ. Giá trị trọng số cho một lớp phải bằng nhau trong mọi router chạy BGP trong một hệ thống AS.
- Giao thức định tuyến trong hệ thống AS (IBGP với định tuyến trong hệ thống AS intra AS routing): Router dùng BGP chỉ chạy tương thích tốt với giao thức định tuyến IGP là OSPF. Khi hệ thống AS không dùng OSPF, Bay Network đưa ra giao thức định tuyến trong hệ thống AS (IBGP intra AS). Với khả năng này router dùng BGP không quảng bá tuyến vào trong hệ thống AS mà các

router trong hệ thống AS sẽ dùng giao thức IBGP. Thông tin IBGP dùng kết hợp với IGP để xác định router dùng BGP cho các mạng bên ngoài.

*8.4.4.3. Thuật toán chọn đường của BGP*

- Nếu hệ thống AS tiếp theo (next host AS) không với tới được thì bỏ qua, ưu tiên các hệ thống AS dùng BGP có tính chất quản trị BGP administrative weights) cao hơn.
- Nếu các Border Router có cùng giá trị trọng số (weight) thì ưu tiên các tuyến có tính năng địa phương (local preference) cao hơn.
- Nếu tính năng địa phương bằng nhau thì ưu tiên các tuyến có router định sẵn (originated router).
- Nếu không có tuyến định trước, chọn đường AS ngắn hơn.
- Nếu đồng bộ giao thức định tuyến trong IGP (IGP Sync) bị bỏ qua và chỉ có các tuyến bên trong, chọn tuyến đến láng giềng gần nhất (closed neighbor).
- Ưu tiên tuyến có giá trị IP thấp nhất cho BGP router ID.

RFC 1163 cùng RFC 1164 định nghĩa một tiêu chuẩn cho giao thức định tuyến liên AS của Internet. Chức năng chính của hệ thống thông báo BGP là để trao đổi thông tin về các mạng có thể tới được với các hệ thống BGP khác. Thông tin này bao gồm đường dẫn đầy đủ của các AS. BGP chạy trên giao thức vận chuyển tin cậy. Bất kỳ cơ chế xác thực nào được dùng cho giao thức vận chuyển đều có thể được dùng cho các cơ chế xác thực riêng của BGP. Cơ chế thông báo lỗi (notification) được dùng trong BGP để yêu cầu đóng liên kết. BGP dùng TCP như giao thức vận chuyển của nó. BGP dùng cổng TCP 179 để thiết lập các liên kết.

*8.4.4.4. Hoạt động của BGP*

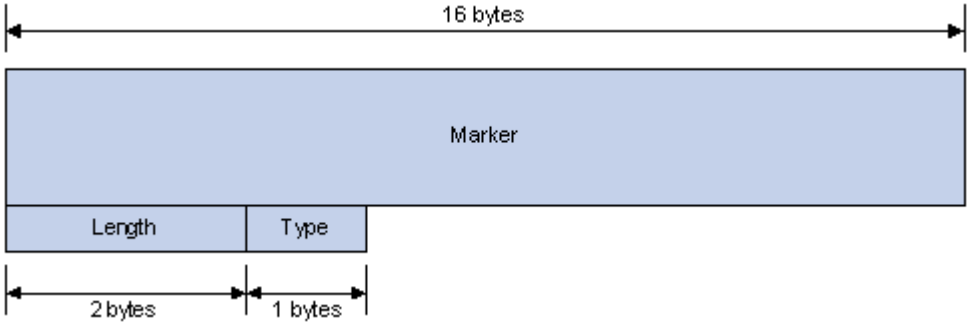
- Hai hệ thống tạo mối liên kết TCP với nhau qua cổng 179. Chúng trao đổi các thông điệp để xác nhận và mở các tham số liên kết. Luồng dữ liệu khởi động liên kết là toàn bộ bảng định tuyến
- BGP không đòi hỏi làm tươi lại theo chu kỳ toàn bộ bảng định tuyến BGP. Bởi vậy một thông báo BGP phải nhớ phiên bản hiện thời của toàn bộ bảng định tuyến của tất cả các đích (peer) cho toàn bộ quá trình liên kết. Các thông điệp thông báo (notification) được gửi trong trả lời khi có lỗi, khi đó liên kết sẽ bị ngắt
- Các host đang thực hiện BGP không cần phải là các router. Một host đang không định tuyến có thể trao đổi thông tin định tuyến với các router qua giao thức định tuyến ngoài EGP hoặc một giao thức định tuyến trong RIP. Host

không đang định tuyến đó có thể dùng BGP để trao đổi thông tin định tuyến với Border Router trong một hệ thống AS khác.

*8.4.4.5. Các dạng thông điệp được BGP dùng*

Các thông điệp được gửi trên một liên kết vận chuyển tin cậy. Một thông điệp được xử lý ngay sau khi nó được nhận. Kích thước lớn nhất của thông điệp là 4096 bytes, nhỏ nhất 19 bytes chỉ gồm các BGP header, không có dữ liệu.

Dạng header của thông điệp: mỗi thông điệp có một header thích hợp. Thông điệp có thể có hoặc không có phần dữ liệu sau header, phụ thuộc vào loại thông điệp. Sơ đồ các trường như sau:




*Hình 8.15: Phần Header chuẩn của BGP Message*

- Phần đánh dấu (marker): trường này dài 16 bytes. Nếu loại thông điệp là OPEN, hoặc mã xác thực được dùng trong thông điệp OPEN của liên kết là 0 thì phần đánh dấu (maker) phải toàn bộ là 1. Phần đánh dấu (maker) có thể được dùng để phát hiện sự mất đồng bộ giữa các cặp đích BGP, và để xác thực các thông điệp BGP đến.
- Chiều dài (length) là số nguyên không dấu dài 2 bytes, chỉ chiều dài của thông điệp, gồm cả header. BGP yêu cầu mỗi bản tin có kích thước lớn nhất là 4096 bytes và nhỏ nhất là 19 bytes.
- Trường loại (type) dài 1 bytes, là số nguyên không dấu, nó chỉ ra loại mã của thông điệp. Các loại mã được định nghĩa như sau:

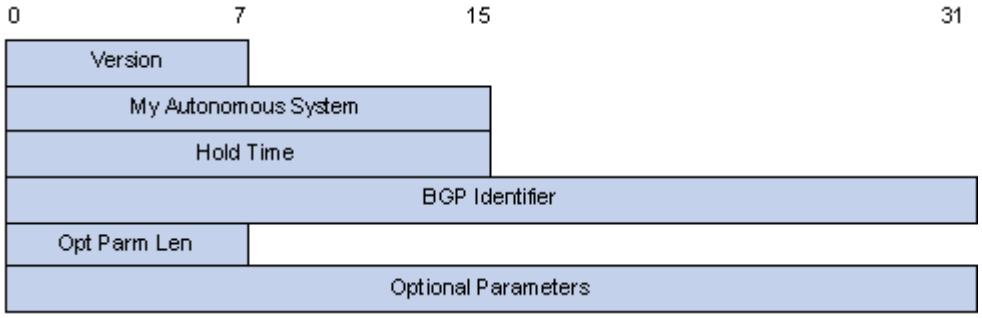
    1. OPEN
    2. UPDATE
    3. NOTIFICATION
    4. KEEPALIVE

Dạng thông điệp OPEN

Sau khi một liên kết TCP được thiết lập, thông điệp đầu tiên được mỗi bên gửi là một thông điệp OPEN.

Nếu thông điệp OPEN được chấp nhận, một thông điệp KEEPALIVE xác nhận OPEN đó được gửi trở lại.

Khi OPEN được xác nhận thì các thông điệp UPDATE, KEEPALIVE và NOTIFICATION được trao đổi. Thông điệp OPEN bao gồm các trường sau:




*Hình 8.16: Dạng thông điệp BGP OPEN*

- Version: là số nguyên không dấu dài 1 bytes, chỉ ra số loại giao thức của thông điệp. Version hiện thời là 2.
- Hệ thống địa phương (My AS): là số nguyên không dấu dài 2 bytes, nó chỉ ra số giây lớn nhất còn lại giữa việc nhận thành công các thông điệp KEEPALIVE và/hoặc UPDATE và/hoặc NOTIFICATION
- Mã xác thực (Authentication code): là số nguyên không dấu dài 1 bytes chỉ ra cơ chế xác thực đang được dùng. Bất cứ khi nào một cơ chế xác thực được xác định để dùng trong hệ thống BGP, phải có ba thành phần cơ bản sau:

  1. Giá trị của mã xác thực chỉ ra tác dụng của cơ chế
  2. Dạng và ý nghĩa của dữ liệu xác thực (Authentication data)
  3. Thuật toán để tính toán các giá trị của trường đánh dấu (maker)

  Chỉ một cơ chế xác thực được xác định như sau:

- Mã xác thực là 0
- Dữ liệu xác thực phải rỗng
- Các trường đánh dấu của tất cả các thông điệp phải toàn là 1

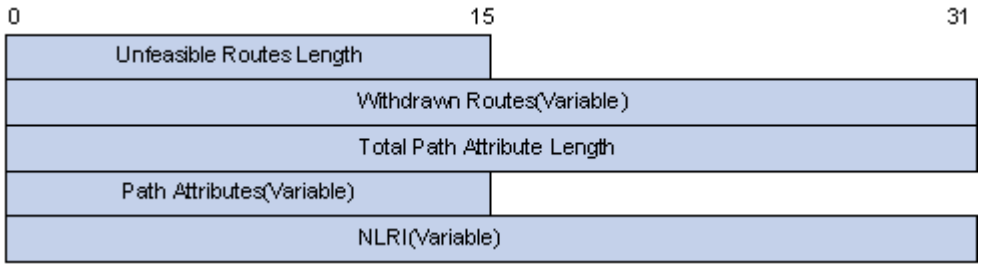
Chú ý: Một cơ chế xác thực riêng có thể được dùng trong việc thiết lập liên kết ở tầng vận chuyển.

Dữ liệu xác thực: trường này có chiều dài thay đổi phụ thuộc vào mã xác thực. Nếu giá trị của trường mã xác thực là 0, thì trường dữ liệu xác thực phải có chiều dài là 0.

Chú ý: Chiều dài của trường dữ liệu xác thực có thể được xác định từ trường *Length* của thông điệp theo dạng sau:

Chiều dài thông điệp = 25 + chiều dài dữ liệu xác thực

Dạng thông điệp UPDATE: các thông điệp UPDATE được dùng để chuyển thông tin định tuyến giữa các hệ thống BGP với nhau. Thông tin trong gói dữ liệu UPDATE có thể được dùng để xây dựng lên một sơ đồ mô tả các quan hệ của các hệ thống. Các thông tin về định tuyến lặp có thể được phát hiện và bị xoá từ định tuyến liên các hệ thống. Thông điệp UPDATE gồm các trường sau:




*Hình 8.17: Dạng thông điệp BGP UPDATE*

- Chiều dài: Tổng chiều dài thuộc tính đường dẫn: là số nguyên không dấu 2 bytes.
- Các thuộc tính đường dẫn: Mỗi thuộc tính đường dẫn là một tập (gồm loại thuộc tính, chiều dài thuộc tính, giá trị thuộc tính) có chiều dài thay đổi
- Loại thuộc tính là trường 2 bytes bao gồm 1 byte các cờ thuộc tính theo sau là 1 byte mã thuộc tính như sau:

| Các cờ thuộc tính | Mã loại thuộc tính |
|---|---|

- Mỗi số mạng Internet 4 byte (network number) chỉ ra một mạng được mô tả bởi các thuộc tính đường dẫn, còn các subnet và các host không được phép. Chiều dài tối thiểu của thông điệp UPDATE là 37 byte (gồm cả header của thông điệp).

Có bốn loại thuộc tính đường dẫn của thông điệp UPDATE như sau:

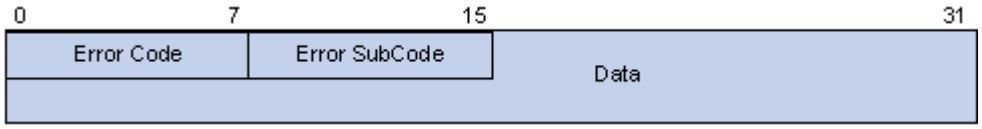
- Well known mandatory
- Well known discretionary
- Optional transitive
- Optional non transitive

Dạng thông điệp KEEPALIVE

BGP không dùng bất kỳ cơ chế keepalive dựa vào giao thức vận chuyển nào để quyết định nếu các đích (peer) là tới được. Thay vào đó, các thông điệp keepalive được trao đổi giữa các đích. Thời gian tối đa hợp lý giữa các thông điệp keepalive là một phần ba của thời gian chờ (hold time interval). Thông điệp keepalive bao gồm header của thông điệp và có chiều dài 19 byte

Dạng thông điệp NONTIFICATION được gửi đi mỗi khi một lỗi được phát hiện. Liên kết BGP bị đóng ngay sau khi gửi thông điệp đi

Thông điệp NONTIFICATION gồm các trường sau:




*Hình 8.18: Dạng thông điệp BGP NOTIFICATION*

Dạng thông điệp NONTIFICATION

Mã lỗi (error code): là số nguyên không dấu 1 byte chỉ ra loại NONTIFICATION. Các mã lỗi sau được định nghĩa

| ***Mã lỗi*** | ***Tên biểu tượng*** |
|---|---|
| 1 | Lỗi header của thông điệp |
| 2 | Lỗi thông điệp OPEN |
| 3 | Lỗi thông điệp UPDATE |
| 4 | Hold Timer Expired |
| 5 | Lỗi Finite State Machine |
| 6 | Cease |

*Bảng 8.2. Mã lỗi thông điệp Nontification*

- Mã lỗi phụ (error subcode): là số nguyên không dấu 1 byte cung cấp thêm thông tin về lỗi thông báo. Mỗi mã lỗi có thể có một hoặc nhiều mã lỗi phụ (error subcode) đi kèm với nó.

  Các mã lỗi phụ của header thông điệp:

  1. Liên kết không được đồng bộ
  2. Chiều dài thông điệp tồi
  3. Loại thông điệp tồi

  Các mã lỗi phụ của thông điệp OPEN

  1. Số hiệu Version không được hỗ trợ
  2. Hệ thống đích tồi
  3. Mã xác thực không được hỗ trợ
  4. Xác thực thất bại

- Dữ liệu: trường này có độ dài thay đổi được dùng để chỉ ra lý do của thông điệp NONTIFICATION. Nội dung của trường dữ liệu phụ thuộc vào mã lỗi phụ.

Chiều dài tối thiểu của thông điệp NONTIFICATION là 21 byte (gồm cả header của thông điệp)

Cấu hình BGP trong Cisco router:

- Kích hoạt định tuyến BGP

router bgp autonomous system; AS của chính router

network network number mask network mask

- Cấu hình Router BGP láng giềng

neighbor ip address remote as number

- Có thể dùng một mẫu chứa danh sách các láng giềng ta dùng lệnh sau:

neighbor template name neighbor list access list number

neighbor template name configure neighbors

- Khởi tạo lại các kết nối BGP:

clear ip bgp address

clear ip bgp *

- Tự động khởi tạo lại router BGP của một láng giềng nếu kết nối bị đứt

bgp fast external fallover

*8.4.4.6. Các mặt nạ địa chỉ nén lại*

Cả hai vùng Withdraw Destinations và Destinations Networks chứa một danh sách các địa chỉ IP mạng. Để dung nạp được địa chỉ không phân lớp, BGP phải gửi đi một mặt nạ địa chỉ với mỗi địa chỉ IP. Tuy nhiên, thay vì gửi đi một địa chỉ và một mặt nạ như những đại lượng 32 bit riêng biệt, BGP sử dụng một cách thể hiện cô đọng (nén lại) để làm giảm bớt kích thước của thông điệp.

Thay vì vậy, nó mã hoá thông tin về mặt nạ vào trong một byte, nằm ở đầu mỗi địa chỉ. Byte mặt nạ (các bit mặt nạ được giả định là liên tục). Phần địa chỉ theo sau mặt nạ. Như thế, sẽ chỉ có một byte địa chỉ tiếp theo sau mặt nạ có giá trị 8 hoặc nhỏ hơn, sẽ có hai byte tiếp theo sau mặt nạ có giá trị từ 8 cho đến 16, sẽ có ba tiếp theo sau mặt nạ có giá trị từ 17 đến 24, và sẽ có bốn tiếp theo mặt nạ có giá trị từ 15 đến 32. Điều lý thú là, chuẩn cũng cho phép một byte mặt nạ chứa zero (trong trường hợp này, sẽ không có byte địa chỉ tiếp theo sau). Giá trị độ dài bằng zero cũng hữu dụng bởi vì nó tương ứng với việc định tuyến mặc định.

*8.4.4.7. Các thuộc tính con đường của BGP*

Chúng ta đã nói rằng BGP không phải là một giao thức vector khoảng cách thuần tuý bởi vì nó thông báo nhiều hơn là chỉ có một trạm kế. Thông tin phụ thêm

được chứa trong vùng Path Attibutes của một thông điệp cập nhật. Nơi gửi có thể sử dụng vùng Path Attibutes để xác định: trạm kế đối với các đích đến đã được thông báo, hoặc là thông tin về con đường được biết từ những hệ tự quan khác hay là được suy ra từ bản thân hệ tự quản của nơi gửi.

Chúng ta cần lưu ý một điều quan trọng rằng vùng Path Attributes được tạo ra để làm giảm bớt kích thước của thông điệp UPDATE; điều này có nghĩa rằng các thuộc tính này áp dụng đối với tất cả các đích được báo trong thông điệp. Như thế, nếu có những thuộc tính khác nhau áp dụng cho một số đích đến, thì chúng phỉ được thông báo trong một thông điệp Update riêng.

Vùng Path Attributes là khá quan trọng trong BGP vì ba lý do. Trước hết, Thông tin con đường cho phép nơi nhận kiểm tra vấn đề các con đường tạo nên vòng lặp. Nợi gửi có thể xác định chính xác con đường đi qua tất cả những hệ tự quản để đến đích. Nếu một hệ tự quản bất kỳ xuất hiện nhiều hơn một lần trong danh sách, thì chắc chắn đã có vòng lặp trong việc định tuyến. Thứ hai, thông tin con đường cho phép nơi nhận cài đặt những ràng buộc về chính sách. Lấy ví dụ, nơi nhận có thể kiểm tra các con đường để biết được rằng chúng không đi quan những hệ tự quản không đáng tin cậy (ví dụ, hệ tự quản của đối thủ cạnh tranh). Thứ ba, thông tin chia nhánh đường cho phép nơi nhận biết nguồn gốc của tất cả các tuyến đường. Cùng với việc cho phép nơi giử xác định rằng thông tin đến từ bên rong hệ tự quản của nó hay từ những hệ khác, vùng Path Attributes cho phép, nơi gửi khai báo rằng thông tin được thu thập với một giao thức cổng ngoại như là BGP hay là một giao thức cổng nội. Như thế, mỗi nơi nhận có thể quyết định xem chấp nhận hay từ chối các tuyến đường có nguồn gốc trong những hệ tự quản nằm ngoài tầm (cùng đẳng cấp với nó).

*8.4.4.8. Hạn chế chính của các giao thức cổng ngoại*

Chúng ta thấy rằng bởi vì các giao thức Cổng Ngoại tuân theo những hạn chế về chính sách, các mạng mà chúng tôi thông báo có thể chỉ là một tập hợp con của các mạng mà chúng có thể đi đến. Tuy nhiên, đã có một giới hạn cơ bản hơn áp đặt lên việc định tuyến ngoại:

Một giao thức cổng ngoại không thông tin liên lạc hay diễn dịch các giá trị về khoảng cách, ngày cả khi các giá trị này tồn tại.

Các giao thức như BGP cho phép máy sử dụng (bộ định tuyến) khai báo rằng một đích không còn có thể đi đến được nữa hay cho một danh sách các hệ tự quản trên con đường đi đến đích, nhưng chúng không thể truyền hay so sánh “chi phí” của hai tuyến đường trừ khi cả hai tuyến đi từ trong cùng một hệ tự quản. Theo nghĩa này thì, BGP chỉ có thể xác định xem có tồn tại một con đường đi đến một đích nào đó hay không; nó không thể truyền hay tính con đường nào ngắn hơn.

Bây giờ chúng ta có thể được lý do tại sao BGP cẩn thận ghi nhận nguồn gốc của thông tin nó gửi đi. Điều cốt yếu chúng ta quan sát được là: khi bộ định tuyến nhận được các thông báo về một đích nào đó từ các đơn vị đồng đẳng trong hai hệ tự quản khác nhau, nó không thể so sánh "chi phí". Như thế, việc thông báo về khả năng đi đến mạng của BGP có ý nghĩa như sau: "Hệ tự quản của tôi cung cấp một con đường đi đến mạng này" không có cách nào bộ định tuyến có thể nói "Hệ tự quản của chúng tôi cung cấp một con đường đi đến mạng này tốt hơn là hệ tự quản khác".

Xem xét cách diễn giải về khoảng cách cho phép chúng ta nhận ra rằng BGP không thể được sử dụng làm thuật giải định tuyến. Cụ thể, ngày cả khi một bộ định tuyến biết được về hai con đường đi đến cùng một mạng, nó không thể biết được con đường nào ngắn hơn bởi vì nó không thể biết "chi phí" của các tuyến đường đi ngang qua các hệ tự quản trung gian. Lấy ví dụ, xét một bộ định tuyến mà sử dụng BGP để thông tin liên lạc với hai đơn vị đồng đẳng trong hệ tự quản P thông báo một con đường đi đến một đích nào đó thông qua các hệ tự quản như p, q, r, và máy đồng đẳng trong hệ tự quản f thông báo một con đường cùng đi đến đích đó thông qua các hệ tự quản như f, g, thì nơi nhận không có cách nào so sánh độ dài của hai con đường. Con đường đi qua ba hệ tự quản có thể chỉ bao gồm một mạng cục bộ trong mỗi hệ, trong khi con đường đi qua hai hệ tự quản một mạng cục bộ trong mỗi hệ, trong khi con đường đi qua hai hệ tự được đầy đủ thông tin định tuyến, nó không thẻ so sánh.

Bởi vì nó không bao gồm giá trị về khoảng cách, một hệ tự quản phải cẩn thận thông báo chỉ những tuyến đường mà số lượng nên đi qua. Về mặt kỹ thuật, chúng ta nói rằng giao thức cổng ngoại là một giao thức về khả năng kết nối chứ không phải giao thức về việc định tuyến chúng ta có thể tóm tắt như sau:

Bởi vì một giao thức cổng ngoại như BGP chỉ có nhân bản thông tin về khả năng kết nối, nơi nhận có thể cài đặt các chính sách để hạn chế, nhưng không thể chọn được tuyến đường ít tốn kén nhất. Nơi gửi chỉ phải thông báo các con đường mà dữ liệu nên đi qua.

Điểm chính yếu ở đây là, một Internet bất kỳ mà sử dụng BGP để cung cấp thông tin định tuyến ngoại, phải dựa vào các chính sách hoặc giả định rằng mỗi hệ tự quản đi qua có chi phí như nhua. Mặc dù trông có vẻ vô hại, giới hạn này có một vài hệ quả thật ngạc nhiên:

1. Mặc dù BGP có thể thông báo nhiều con đường đi đến một mạng nào đó, nó không cung cấp để sử dụng đồng thời nhiều con đường. Điều này có nghĩa là, tại một thời điểm cho trước, tất cả các giao dịch được chuyển từ một máy tính trong một hệ tự quản đến một mạng trong một hệ khác sẽ di chuyển trên một con đường, mặc dù rằng tồn tại nhiều kết nối (vật tư). Chúng ta cũng lưu ý rằng một hệ

thống này phân chia dữ liệu gửi ra trên hai hay nhiều con đường. Kết quả là, độ trì hoãn và hiệu suất giữa một cặp máy có thể bị mất cân đối, làm cho việc kiểm soát và bắt lỗi gặp khó khăn.

2. BGP không hỗ trợ việc chia sẻ giao dịch ra đều trên các bộ định tuyến giữa các hệ tự quản bất kỳ. Nếu hai hệ tự quản có nhiều bộ định tuyến kết nối chung, chúng ta thường muốn cần bằng giao dịch cho đều trong số các bộ định tuyến. BGP cho phép các hệ tự quản phân chia giao dịch theo mạng (ví dụ, phân chia chúng ra nhiều tập con và có nhiều bộ định tuyến cho các tập hợp con), nhưng lại không hỗ trợ việc chia sẽ cân bằng.

3. Trong một trường hợp đặc biệt của điểm #2, bản thân BGP là không cân xứng trong việc tối ưu việc định tuyến trong một kiến trúc là có hai hay nhiều mạng diện rộng được nối với nhai tại nhiều điểm. Thay vì thế, người quản trị phải cấu hình một cách thủ công những mạng nào được thông báo bởi mỗi bộ định tuyến ngoại.

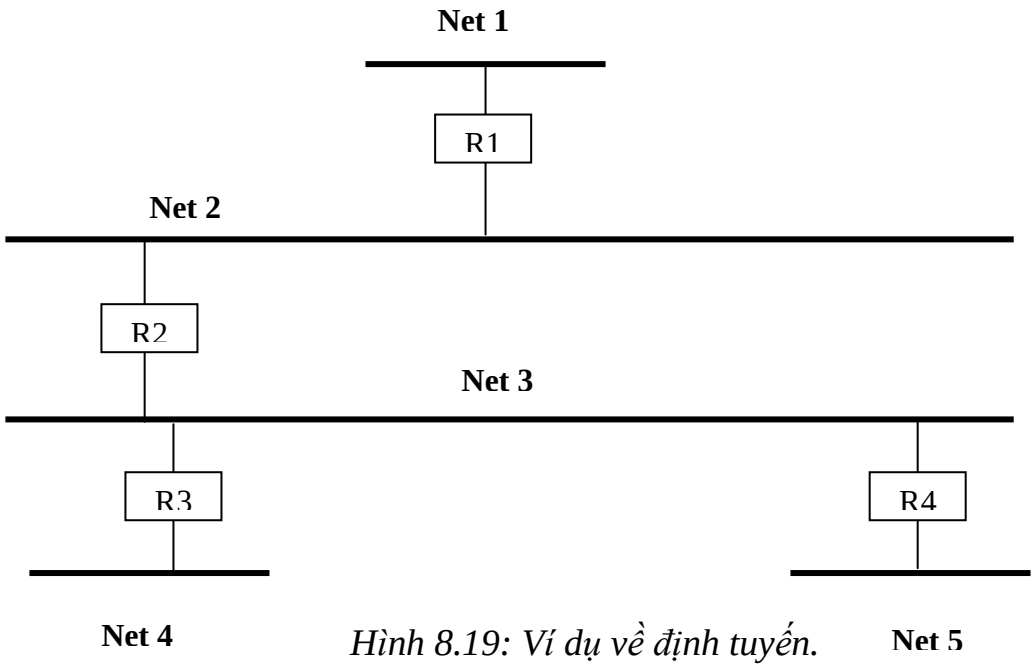
4. Để có được sự định tuyến hợp lý hoá, tất cả các hệ tự quản trong một Internet phải thống nhất với nhau trên một mô hình cho việc thông báo về tính kết nối. Nghĩa là, bản thân BGP sẽ không bảo đảm sự thống nhất toàn cục.

## 8.5. Định tuyến trong một hệ tự quản

### 8.5.1. Giao thức cổng nội IGP

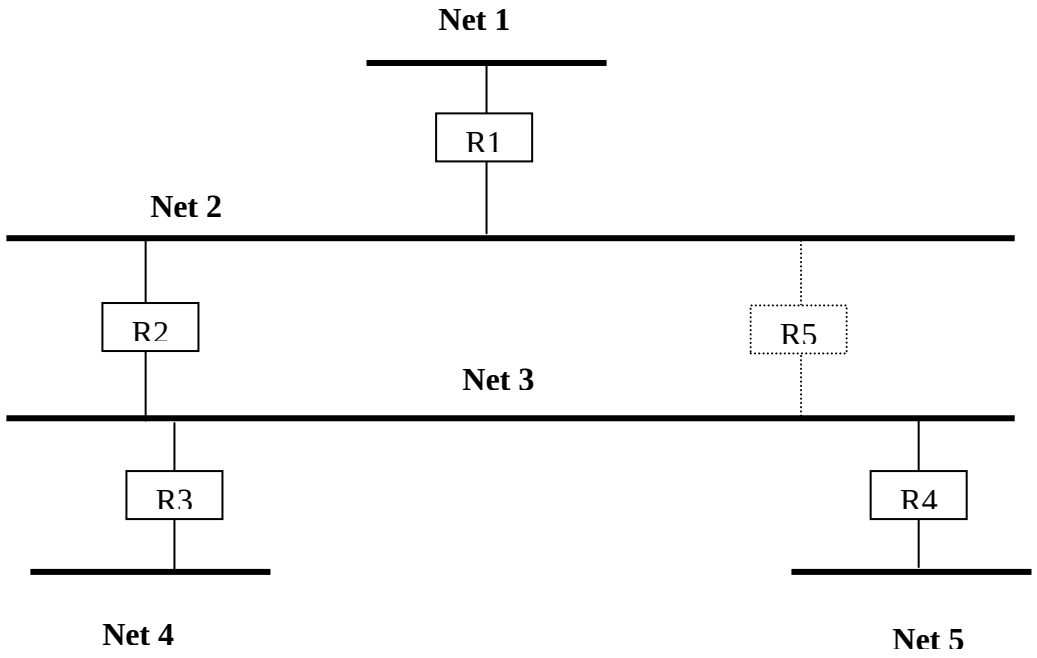
Hai bộ định tuyến bên trong một hệ tự quản được gọi là nội bộ (interior) đối với cái kia. Lấy ví dụ, hai bộ định tuyến của một trường đại học được xem là nội bộ đối với cái kia khi mà các máy này được tập hợp vào một hệ tự quản.

Làm thế nào một bộ định tuyến trong một hệ tự quản biết được về các mạng bên trong hệ tự quản này? Trong một Internet nhỏ, thay đổi chậm, người quản trị có thể thiết lập và hiệu chỉnh các tuyến đường một cách thủ công. Người quản lý lưu trữ một bảng các mạng và cập nhật bảng này bất cứ khi nào một mạng mới được thêm vào, hay một mạng bị loại bỏ khỏi hệ tự quản. lấy ví dụ, hãy xét Internet nhỏ của một công ty như trình bày trong hình 8.19.




*Hình 8.19: Ví dụ về định tuyến.*

Việc định tuyến cho Internet trong hình này thật đơn giản bởi vì chỉ tồn tại có một con đường giữa hai điểm bất kỳ. Người quản lý có thể cấu hình một cách thủ công các tuyến đường trong tất cả các máy tính và bộ định tuyến. Nếu Internet thay đổi (ví dụ, có mạng mới được thêm vào), người quản lý phải cấu hình lại các tuyến đường trong tất cả các máy.




*Hình 8.20: Ví dụ về định tuyến.*

Hiển nhiên, các hệ thống thủ công có nhiều nhược điểm: các hệ thủ công không thể theo kịp sự phát triển nhanh chóng hay sự thay đổi quá nhanh. Trong phạm vi lớn, những thay đổi nhanh chóng của môi trường như là Internet toàn cầu,

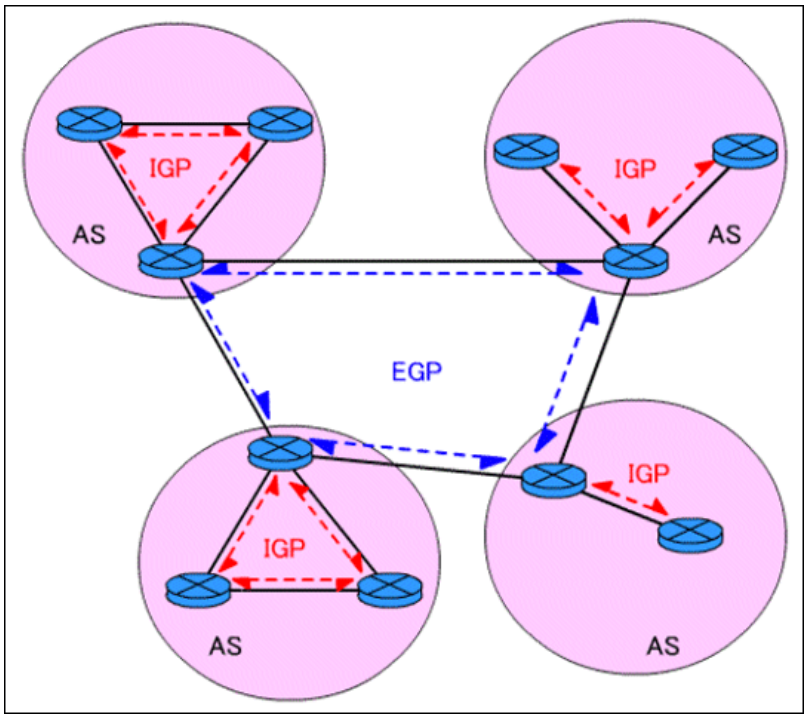
con người thể đáp ứng nhanh đối với những thay đổi để xẻ lý các vấn đê; cần phải sử dụng các phương pháp tự động hoá. Các phương pháp tự động hoá cũng có thể giúp đỡ trong việc hoàn thiện độ tin cậy và đáp ứng đối với những sai sót trong những Internet nhỏ mà có những tuyến đường thứ hai. Để thấy được việc này, chúng ta hãy thử xem điều gì xảy ra nếu chúng ta thêm một bộ định tuyến vào Internet trong hình 8.19. Để tạo ra Internet như trong hình 8.20.

Trong những kiến trúc Internet có nhiều đường vật lý, người quản lý thường chọn một con đường để làm con đường chính. Nếu các bộ định tuyến dọc theo con đường chính bị hỏng, các tuyến đi phải được thay đổi để gửi dữ liệu theo con đường thứ hai. Việc thay đổi các tuyến đường thường mất nhiều thời gian và dễ bị lỗi. Như thế, ngay cả trong những Internet nhỏ, cũng cần phải sử dụng một hệ tự động load để việc thay đổi các tuyến đường được nhanh chóng và có độ tin cậy cao.

Để tự động hoá các công việc duy trì tính chính xác của thông tin về thông tin liên lạc đến một mạng, các bộ định tuyến nội thường thông tin liên lạc với những cái khác, trao đổi với nhau dữ liệu về khả năng đi đến một mạng hoặc thông tin định tuyến của mạng mà từ đó có thể suy ra được khả năng đi đến một mạng. một khi thông tin về khả năng đi đến một mạng của toàn bộ hệ tự quản đã được tập hợp, một trong các bộ định tuyến trong hệ thống có thể thông báo cho các hệ tự quản khác, sử dụng giao thức cổng ngoại (Exterior Gateway Protocol).

Bởi vì không có một chuẩn nhất định, chúng ta sử dụng thuật ngữ giao thức cổng nội (Internet Gateway Protocol – IGP) như là một mô tả tổng quát để chỉ đến bất kỳ thuật giải nào mà bộ định tuyến nội sử dụng khi chúng trao đổi thông tin định tuyến và thông tin về khả năng đi đến một mạng. Lấy ví dụ, thế hệ cuối cùng của các bộ định tuyến chủ chốt đã sử dụng một giao thức có tên SPREAD để làm giao thức cổng nội của nó. Có một vài hệ tự quản sử dụng BGP để làm IGP, mặc dù điều này hiếm khi là một chọn lựa tối ưu đối với những hệ tự quản nhỏ gồm các mạng cục bộ với khả năng quảng bá.

Hình 8.21. Minh hoạ bốn hệ tự quản, mỗi hệ sử dụng một IGP để nhân bản thông tin định tuyến giữa các bộ định tuyến nội của nó.




*Hình 8.21: IGP và EGP*

Một bộ định tuyến có thể đồng thời sử dụng hai giao thức định tuyến khác nhau, một giao thức dành cho việc thông tin liên lạc bên ngoài hệ tự quản của nó và giao thức kia để cho việc thông tin liên lạc bên trong hệ tự quản của nó.

Cụ thể là, các bộ định tuyến mà sử dụng BGP để thông báo về khả năng đi đến một mạng cũng thường cần sử dụng một IGP để lấy được thông tin từ bên trong hệ tự quản của chúng.

### 8.5.2. Giao thức định tuyến RIP

#### *8.5.2.1. Lịch sử của RIP*

Một trong những giao thức IGP được sử dụng rộng rãi nhất là giao thức thông tin định tuyến (RIP – Routing Information Protocol), cũng được biết dưới tên của chương trình đã cài đặt nó, routed (chữ "d" ở cuối là do quy ước đặt tên trong UNIX dành cho các daemon). Phần mềm routed ban đầu được thiết kế tại viện đại học Caliornia ở Berkeley để cung cấp việc định tuyến nhất quán và thông tin về khả năng đi đến các máy trên các mạng của chúng. Nó dựa vào việc quảng bá (phần cứng) để thực hiện việc trao đổi thông thông tin định tuyến một cách nhanh chóng. Nó không được thiết kế để sử dụng trên các mạng lớn cũng như mạng diện rộng (mặc dù vậy, hiện nay một số công ty cũng có bán các phiên bản của RIP được điều chỉnh để sử dụng với WAN).

Dựa vào các nghiên cứu trước đây về Internetwork đã thực hiện tại trung tâm nghiên cứu của công ty Xerox tại Palo Alto, routed, cài đặt một giao thức suy ra từ NS RIP của Xerox, nhưng tổng quát hoá nó lên để bao gồm nhiều họ mạng.

Mặc dù có những cải tiến nhỏ đối với các phiên bản trước, tính phổ biến của RIP như là một IGP không phải có được từ những kỹ thuật của nó. Thay vì thế, nó là kết quả của Berkeley về việc phân phát phần mềm routed cùng với hệ điều hành UNIX 4BSD nổi tiếng của họ. Vì thế, nhiều phiên bản TCP/IP đã chấp nhận và các giới hạn của nó. Một khi đã cài đặt và chạy, nó trở thành cơ sở của việc định tuyến cục bộ, và các nhóm nghiên cứu đã đưa nó vào các mạng lớn hơn.

Có lẽ sự kiện kỳ lạ nhất về RIP là nó đã được xây dựng và chấp nhận rộng rãi trước khi một chuẩn chính thức được xuất bản. Hầu hết các cài đặt đều có nguồn gốc từ những chương trình của Berkeley, với sự tác động qua lại trong các chương trình bị hạn chế bởi sự hiểu biế của người lập trình về các chi tiết chưa được công bố. Khi các phiên bản mới ra đời, rất nhiều vấn đề nảy sinh. Chuẩn RFFC ra đời năm 1988, và giúp cho những nhà sản xuất tránh được các lỗi.

*8.5.2.2. Hoạt động của RIP (version 1.0)*

Giao thức RIP cơ sở là một cài đặt trực tiếp của việc định tuyến vector khoảng cách dành cho mạng cục bộ. Nó chia các thành phần tham dự ra hai phần máy chủ động và máy thụ động (nghĩa là, yên lặng). Các máy chủ động báo các tuyến đường của nó cho những máy khác; các thành phần thụ động lắng nghe các thông điệp RIP và sử dụng chúng để cập nhật bảng định tuyến của chúng, nhưng không gửi thông báo. Chỉ có bộ định tuyến có thể chạy RIP ở chế độ chủ động; một máy tính phải sử dụng chế độ thụ động.

Bộ định tuyến chạy RIP ở chế độ chủ động sẽ quảng bá một thông điệp cập nhật việc định tuyến trong mỗi 30 giây. Việc cập nhật chứa thông tin được lấy từ cơ sở dữ liệu định tuyến hiện tại của bộ định tuyến. Một cập nhật chứa một tập hợp các cặp, trong đó mỗi cặp chứa một địa chỉ mạng IP và một số nguyên là khoảng cách đến mạng đó. RIP sử dụng giá trị số trạm để đo khoảng. Trong cách tính của RIP, một bộ định tuyến được tính là một trạm kể từ mạng được kết nối trực tiếp (các giao thức định tuyến khác tính một kết nối trực tiếp là zero), tính là hai trạm kể từ mạng mà có thể đi đến được khi đi qua một trạm khác,… Như thế, số trạm dọc theo con đường từ một nguồn đến đích để chỉ số lượng bộ định tuyến mà datagram đi qua trên suốt con đường. Dĩ nhiên là, việc sử dụng số trạm để tính đường đi ngắn nhất không phải lúc nào cũng cho ra kết quả tối ưu. Lấy ví dụ, một con đường có ba trạm đi qua ba Ethernet có thể nhanh hơn nhiều so với con đường có hai trạm đi qua kết nối satellite. Để bù đắp cho những khác biệt kỹ thuật, nhiều cài đặt của RIP cho phép người quản lý cấu hình một cách nhân tạo nhiều trạm hơn khi đi qua các liên kết mạng có tốc độ chậm (ví dụ satellite).

Cả hai loại này RIP chủ động và thụ động đều lắng nghe các thông điệp quảng bá, và cập nhật bảng cơ sở dữ liệu của chúng theo thuật giải vector khoảng cách đã được mô tả trước đây. Lấy ví dụ trong Internet trên hình 16.2, bộ định tuyến R1 sẽ quảng bá một thông điệp trên mạng 2 có chứa cặp (1,1); Điều này có nghĩa rằng nó có thể đi đến mạng 1 với "chi phí" là 1. Các bộ định tuyến R2 và R5 sẽ nhận được quảng bá này và cài đặt một tuyến đường đến mạng 1 thông qua R1 (với chi phí là 2). Sau đó, các bộ định tuyến R2 và R5 sẽ bao gồm cặp (1, 2) khi chúng quảng bá các thông điệp RIP của chúng trên mạng 3. Cuối cùng thì, tất cả các bộ định tuyến và các máy tính sẽ cài đặt một tuyến đường đến mạng1.

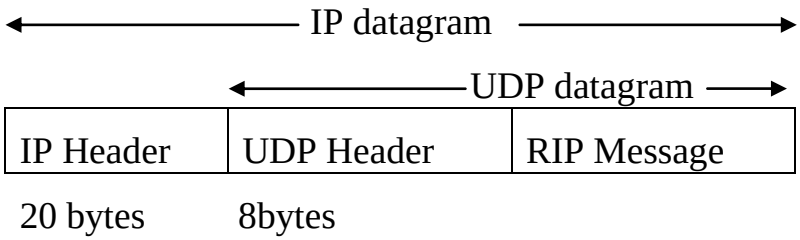
RIP xác định một vài quy tắc để hoàn thiện hiệu năng và độ tin cậy. Ví dụ, một khi mà bộ định tuyến biết được về một tuyến đường từ các bộ định tuyến khác, nó phải áp dụng phương pháp "hysteresis", có nghĩa là nó không thay thế tuyến đường này bằng môt tuyến đường có cùng chi phí. Trong ví dụ của chúng ta, nếu bộ định tuyến R2 và R5 cùng thông báo rằng mạng 1 có chi phí là 2, các bộ định tuyến R3 và R4 sẽ cài đặt tuyến đường đi qua bộ định tuyến nào mà thông báo trước. Chúng ta có thể tóm tắt:

Để ngăn ngừa sự lúng túng khi chọn giữa các tuyến đường có chi phí bằng nhau; RIP xác định rằng các tuyến đường đang tồn tại phải được giữ lại cho đến khi có tuyến đường mới với chi phí thấp hơn.

Điều gì sẽ xảy ra khi bộ định tuyến đầu tiên thông báo tuyến đường lại bị hỏng (ví dụ, máy bị hư)? RIP xác định rằng tất cả các đơn vị lắng nghe phải định thời hạn (timeout) cho các tuyến đường mà chúng được thông qua RIP. Khi một bộ định tuyến cài đặt một tuyến đường trong bảng của nó, nó khởi động một bộ đếm thời gian cho tuyến đường đó. Bộ đếm thời gian này phải được khởi động lại bất cứ khi nào bộ định tuyến nhận được thông điệp RIP khác về tuyến đường này. Tuyến đường này sẽ thông điệp RIP khác về tuyến đường này.Tuyến đường này sẽ không còn giá trị nếu sau 180 phút tuyến đường này không được thông báo lại.

RIP phải xử lý ba loại, do lỗi thuật giải cơ sở gây ra. Trước hết, bởi vì thuật giải không chính thức nhận biết các tuyến đường tạo ra vòng lặp, RIP phải giả định rằng các thành phần tham dự có thể tin cậy được hoặc cần phải để ý hầu ngăn ngừa các vòng lặp. Thứ hai, để ngăn ngừa tính bất ổn, RIP phải sử dụng một giá trị nhỏ để gán cho khoảng cách tối đa có thể có (cụ thể là 16). Như thế, với những Internet mà trong đó giá trị đếm số trạm có thể đạt đến 16, người quản lý phải chia trong đó giá trị đếm số trạm có thể đạt đến 16, người quản lý phải chia Internet ra thành những phân đoạn hoặc sử dụng một giao thức khác. Thứ ba, đếm đến vô hạn; trong đó xuất hiện những điều không nhất quán bởi vì các trực tiếp cập nhật việc định tuyến được nhân bản một cách chậm chạp qua mạng. việc chọn một giá trị giới hạn nhỏ (16) làm giới hạn vấn đề "đếm đến vô hạn" nhưng vẫn không loại bỏ hẳn được vấn đề này.

*Đóng gói thông điệp RIP*



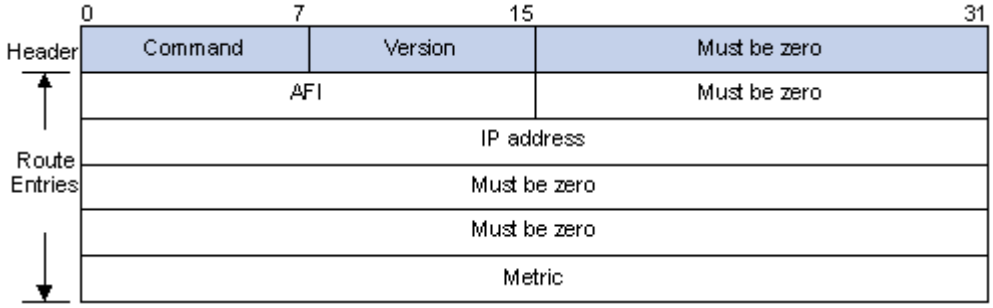

*Hình 8.22: Thông điệp RIP nằm trong gói dữ liệu UDP*

Khuôn dạng của thông điệp RIP như hình 8.23 dưới đây:

Command = 1 là một RIP request,

2 là RIP reply.

Một RIP request yêu cầu hệ thống khác gửi toàn bộ hoặc một phần bảng định tuyến của nó.




*Hình 8.23: Khuôn dạng của thông điệp RIP*

Một RIP reply bao gồm toàn bộ hoặc phần bảng định tuyến của nơi gửi.

Vùng VERSION chứa số phiên bản giao thức (là 1 trong trường hợp này), và được sử dụng tại nơi nhận để kiểm chứng rằng nó sẽ diễn dịch thông điệp một cách chính xác.

20 bytes tiếp theo xác định địa chỉ address family (luôn là 2 đối với các địa chỉ IP), một địa chỉ IP, và một ma trận được liên kết. Có đến 25 tuyến có thể được quảng bá trong một thông điệp RIP dùng 20 bytes này. Giới hạn 25 là để giữ kích thước tổng của thông điệp RIP là 20*25+ 4 = 504, nhỏ hơn 512 bytes.

Tính tổng quát của RIP cũng được thể hiện trong cách mà nó truyền địa chỉ mạng. Định dạng của địa chỉ không bị giới hạn để được sử dụng bởi TCP/IP; nó có thể được sử dụng với nhiều bộ giao thức mạng. Như đã trình bày trong hình 8.23, mỗi địa chỉ do RIP thông báo có thể có địa chỉ bao gồm tới 14 byte. Dĩ nhiên, các địa chỉ IP chỉ cần có 4 byte; RIP xác định rằng các byte còn lại phải là zero (những người thiết kế đã quyết định vị trí địa chỉ IP là từ byte thứ 3 đến thứ 6 trong vùng địa chỉ). Vùng có tên *FAMILY OF NET* i xác định họ của giao thức mà địa chỉ

mạng được diễn dịch trong đó RIP sử dụng các giá trị được gán cho các họ địa chỉ dưới hệ điều hành UNIX 48 SD (các địa chỉ IP được gán cho giá trị 2).

Cùng với các địa chỉ IP thông thường, RIP sử dụng quy ước rằng địa chỉ 0.0.0.0 để chỉ tuyến đường mặc định. RIP có đưa giá trị khoảng cách vào mọi tuyến đường mà nó thông báo, bao gồm cả tuyến đường mặc định. Như thế, có thể bố trí để hai bộ định tuyến thông báo tuyến đường mặc định theo những giá trị khoảng cách khác nhau, tạo ra một con đường chính và một con đường dự phòng.

DISTANCE TO NET, chứa một số nguyên là khoảng cách đến mạng được xác định. Các khoảng cách được tính theo số lượng bộ định tuyến cần đi qua, nhưng được giới hạn trong khoảng từ 1 đến 16, và giá trị 16 biểu thị cho vô hạn (nghĩa là, không tồn tại tuyến đường đi).

*Hoạt động*

Ta xem một tiến trình định tuyến dùng RIP, số hiệu cổng dùng cho RIP là UDP port 520. Khi daemon bắt đầu, nó xác định tất cả các giao diện đang hoạt động và gửi một gói dữ liệu "request packet" cho mỗi giao diện đó để yêu cầu các giao diện gửi cho nó bảng định tuyến đầy đủ của Router khác. Trên một liên kết PPP thì yêu cầu này được gửi tới đầu kia của liên kết. Yêu cầu này được quảng bá nếu mạng hỗ trợ nó. Cổng UDP đích là 520. Gói dữ liệu này có giá trị command=1 nhưng address family là 0 và metric là 16. Đây là một RIP request đặc biệt yêu cầu một bảng định tuyến đầy đủ từ đầu kia của liên kết.

Khi RIP request được nhận thì sau đó bảng định tuyến đầy đủ được gửi tới nơi gửi RIP request. Nếu không mỗi chỉ mục trong RIP request được xử lý: Nếu ta có một tuyến tới địa chỉ riêng, đặt metric theo giá trị của ta, nếu không đặt metric là 16 và có nghĩa là ta không có một tuyến tới đích đó. Khi đó trả lời respond được trả lại.

Trả lời RIP Respond được nhận và có thể cập nhật lại bảng định tuyến. Các tuyến mới có thể được thêm vào, các chỉ mục đang tồn tại có thể được sửa đổi hoặc bị xoá.

Cập nhật bảng định tuyến theo định kỳ: sau mỗi 30 giây, toàn bộ hoặc một phần của bảng định tuyến của Router được gửi tới mọi router láng giềng. Bảng định tuyến có thể được quảng bá hoặc gửi tới đầu kia của liên kết PPP

Việc cập nhật xảy ra bất cứ khi nào số metric cho một tuyến thay đổi. Mỗi tuyến có một thời gian sống nhất định (timeout). Nếu một hệ thống đang chạy RIP tìm một tuyến không được cập nhật khoảng 3 phút, metric của tuyến đó được đặt tới 16 và được đánh dấu xoá. Có nghĩa ta bỏ qua 6 lần cập nhật 30 giây từ router đã đưa ra tuyến đó.

Các metric được dùng bởi RIP là các host count. Host count của các giao diện nối trực tiếp là 1. Nếu một router đưa ra một tuyến tới một mạng khác với một host count là 1, metric cho mạng đó là 2, khi phải gửi gói dữ liệu tới router đó để truyền trên mạng. Khi mỗi router gửi các bảng định tuyến của nó tới các router láng riềng, thì các tuyến có thể được xác định tới mỗi mạng trong hệ thống. Nếu có nhiều đường dẫn trong hệ thống từ một router tới một mạng thì router sẽ chọn đường dẫn có ít host count nhất và bỏ qua các đường dẫn khác. Host count được giới hạn là 15, nghĩa là RIP có thể được dùng chỉ bên trong hệ thống, ở đây host count lớn nhất giữa các host là 15. Metric đặc biệt 16 chỉ ra là không có tuyến nào tồn tại đối với địa chỉ IP.

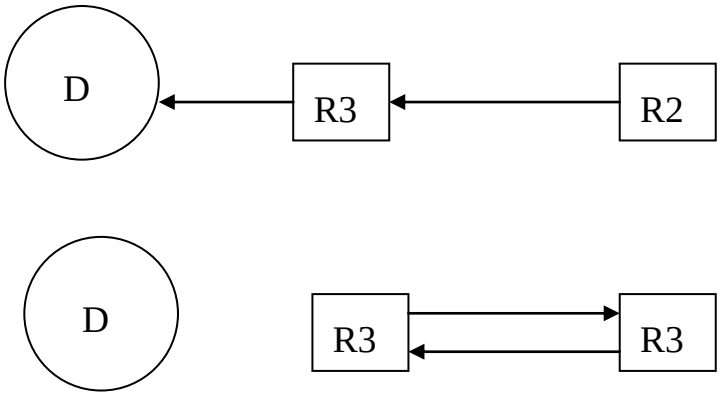
Bởi vì RIP không kiểm tra và sửa chữa những hiện tượng lặp trong chọn đường, những thông tin về những tuyến vượt quá 15 host sẽ được coi là không có giá trị, đó cũng là hạn chế của RIP khi sử dụng trong những mạng lớn.

*8.5.2.3. Các hạn chế của RIP 1.0*

Router dùng RIP cũng gặp phải những vấn đề độ tin cậy trong bảng định tuyến do thời gian đồng bộ thông tin giữa các router khi có sự thay đổi xảy ra trong mạng. Lý do chính là do sự đồng bộ thông tin chậm dẫn đến hiện tượng lặp, gọi là count to infinity như hình vẽ.

Router R3 có kết nối với mạng D. R2 nhận được thông tin do RIP ở R3 quảng bá tới, do đó nó có được thông tin chọn đường tới mạng D.

Khi liên kết của R3 tới mạng D không còn nó xoá bỏ thông tin trong bảng định tuyến của nó trong khi thông tin trong bảng định tuyến của R2 vẫn còn nên R2 lại gửi thông tin cập nhật này cho R3 nói rằng có thể tới được mạng D với số host là 2.




*Hình 8.24: Định tuyến lặp giữa hai router*

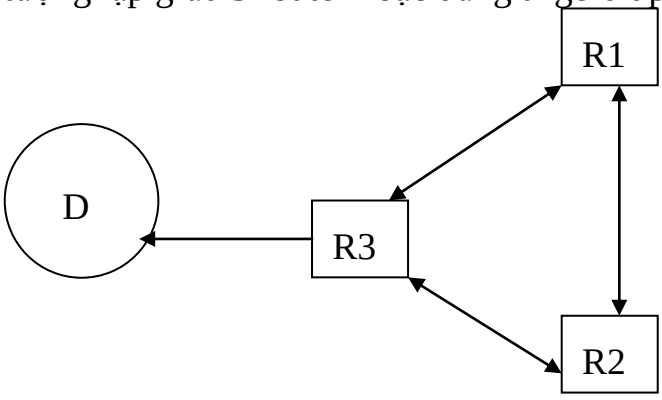
Do vậy R3 tính rằng nó có thể tới được mạng D với số host là 3, trong trường hợp này các gói dữ liệu đi tới mạng D được nhận bởi R3 và R2 sẽ truyền giữa 2 router đến khi TTL=0 gây ra hiện tượng định tuyến lặp (routing loop).

Khi 2 router tiếp tục cập nhật thông tin chọn đường, quá trình này còn mất thêm một khoảng thời gian nữa. Khi R2 không còn nhận được thông tin cập nhật từ R3 tới mạng D với giá trị cost 2, thông tin tới mạng D với giá trị cost 2 sau đó sẽ không có giá trị và sẽ bị xoá khỏi bảng định tuyến của R2 khi đó nhận được thông tin tới mạng D của R3 với cost 3 và cập nhật thông tin tới mạng D vào bảng định tuyến của nó với cost 4, tương tự như vậy R3 tăng cost của nó lên 5 cứ như vậy cho đến khi giá trị cost đạt tới 16, quá trình này mất vài phút.

Có một số công nghệ để khắc phục sự hội tụ chậm như Split horizon, Poison reverse và Trigered updates.

**Split horizon**

- Split horizon không cho phép một router cập nhật những thông tin mà nó đã gửi cho một router khác quay trở lại router này.
- Split horizon không có poison reverse cũng không khắc phục được hiện tượng lặp nếu trong trường hợp có 3 router như hình dưới.
- Nếu R3 mất liên kết với mạng D nó sẽ ngừng quảng bá thông tin tới mạng D cho R1 và R2. Khi đó Split horizon không cho phép R1 và R2 gửi thông tin tới D cho R3 do đó thông tin D của R1 và R2 sẽ mất giá trị khi hết thời gian (time out).
- Giả sử trong khi R2 vẫn còn thông tin tới D nhưng R1 đã bị mất, khi đó R1 sẽ nhận được thông tin cập nhật từ R2 (cost=3) do đó nó thông báo với R3 về thông tin cập nhật tới D (cost=4), khi thông tin trên R2 (cost=2) mất do timeout R3 sẽ cập nhật thông tin tới D do R2 (cost=5) cứ như vậy cho đến khi cost=16, để tránh trường hợp này phải thiết kế cấu hình mạng để không xảy ra hiện tượng lặp giữa 3 router hoặc dùng trigerd updates.




*Hình 8.25: Định tuyến lặp giữa 3 router*

**Poision Reverse**

Với Poision Reverse thay vì không truyền thông tin chọn đường cho máy đã quảng bá thông tin đó các router truyền thông tin này với cost bằng 16.

**Trigered updates**

Cho phép cập nhật kịp thời các thông tin khi có một thông tin chọn đường mới hoặc bị xoá, điều này cho phép giảm thiểu thời gian lưu giữ các thông tin không có giá trị, trong trường hợp trên khi R3 mất liên kết với mạng D nó sẽ cập nhật ngay lên các router R2, R1 thông tin này.

*8.5.2.4. Giải quyết vấn đề đếm đến vô hạn*

Với ví dụ trong hình 8.24, chúng ta có thể giải quyết vấn đề đếm đến vô hạn bằng cách sử dụng một kỹ thuật có tên là split horizon update. Khi sử dụng cách cắt theo hàng ngang, bộ định tuyến sẽ không nhân bản thông tin về tuyến đường ngược trở về bộ giao tiếp mà từ đó tuyến đường này đã đến. Trong ví dụ của chúng ta, việc cắt theo hàng ngang ngăn ngừa bộ định tuyến R2 khỏi việc thông báo một tuyến đường đi đến mạng 1 ngược trở về bộ định tuyến R1, vì vậy nếu R1 mất liên lạc với mạng 1, nó phải dừng việc thông báo về tuyến đường. Với kỹ thuật cắt theo hàng ngang, sẽ không có vòng lặp trong việc định tuyến ở mạng trong ví dụ này. Lý do là, sau một vài cập nhật việc định tuyến, tất cả các bộ định tuyến cùng đồng ý rằng mạng này là không thể đi đến được. Tuy nhiên, kỹ thuật cắt theo hàng ngang không ngăn cản được các vòng lặp trong tất cả các cách cấu hình mạng như trong một bài tập cuối chương.

Chúng ta cũng có thể nhìn nhận vấn đề đếm đến vô hạn theo một cách khác, đó là theo phương tiện luồng thông tin. Nếu một bộ định tuyến thông báo một con đường ngắn đi đến mạng nào đó, thì tất cả các bộ định tuyến khác (mà nhận được thông báo) sẽ nhanh chóng cài các bộ định tuyến này. Nếu một bộ định tuyến thôi không thông báo về một tuyến đường, thì giao thức phải phụ thuộc vào cơ chế bộ đếm thời gian trước khi đi đến kết luận là không còn đi theo con đường này nữa. Một là đã hết hạn (timeout), bộ định tuyến này sẽ tìm một tuyến đường khác và bắt đầu việc nhân bản thông tin đó. Tiếc thay, bộ định tuyến không thể biết được tuyến định tuyến mới này có phụ thuộc vào tuyến đường vừa bị loại bỏ không. Như thế, thông tin liên lạc không phải lúc nào cũng được nhanh chóng nhân bản. Ý tưởng này có thể được tóm tắt và giải thích ngắn gọn như sau:

Tin tốt đi nhanh; tin xấu đi chậm

Một kỹ thuật khác được sử dụng để giải quyết vấn đề đếm đến vô hạn sử dụng biến hold down. Biến hold down buộc bộ định tuyến tham gia bỏ qua thông tin về một mạng trong khoảng thời giam cố định ngay sau khi nhận được thông điệp khẳng định rằng không còn có thể đi đến mạng này được nữa. Thông thường, biến hold down được gán cho trị giá là 60 giây. Ý tưởng của kỹ thuật này là đợi để lâu để bảo đảm rằng tất cả các máy đều nhận được tin xấu và không vô ý nhận được thông điệp chứa thông tin đã cũ. Chúng ta cần lưu ý rằng tất cả các máy tham gia trong việc trao đổi RIP cần sử dụng cùng một ký hiệu cho biến hold down, nếu

không thì lại có thể bị vòng lặp định tuyến. Khuyết điểm của kỹ thuật hold down là nếu xảy ra các vòng lặp định tuyến, chúng sẽ vẫn tồn tại trong suốt thời hạn hold down. Quan trọng hơn, kỹ thuật hold down cũng giữ lại tất cả những tuyến đường không chính xác trong suốt thời hạn hold down, ngay cả khi có một tuyến đường thay thế.

Một kỹ thuật cuối cùng để giải quyết vấn đề đếm vô hạn được gọi là poison reverse. Một khi một kết nối biến mất, bộ định tuyến nơi thông báo kết nối này vẫn giữ lại tuyến đường này trong một khoảng vài lần cập nhật, và giảm giá trị vô hạn trong những lần quảng bá của nó. Để làm kỹ thuật poison reverse hiệu quả nhất, thì phải kết hợp với kỹ thuật triggerd update. Triggerd update buộc bộ định tuyến gửi tức thì một quảng bá khi nó nhận được tin xấu, thay vì phải đợi đến lần quảng bá định kỳ kế tiếp. Bằng cách gửi thông tin cập nhật tức khắc, bộ định tuyến giảm thiểu thời gian gây "tác hại" của tin xấu.

Tiếc thay, trong khi các kỹ thuật triggerd update, poison reverse, hold down và cắt theo hàng ngang đều giải quyết được một số vấn đề, thì đồng thời chúng cũng tạo ra những vấn đề khác. Lấy ví dụ, chúng ta hãy thử xem điều gì xảy ra với kỹ thuật triggerd update khi có nhiều bộ định tuyến cùng chia sẻ chung một mạng. Chỉ một lần quảng bá có thể thay đổi tất cả các bảng định tuyến của chúng, làm kích hoạt một loại mới các quảng bá. Nếu vòng thứ hai của các quảng bá thay đổi các bảng, nó sẽ lại kích hoạt thêm những quảng bá khác. Điều này gây ra sự khủng hoảng (hầu như tránh những đụng độ trên mạng cơ sở, RIP yêu cầu mỗi bộ định tuyến đợi một thời hạn ngẫu nhiên trước khi gửi một lệnh "triggerd update")

Việc sử dụng quảng bá, tiềm năng gây ra các vòng lặp định tuyến và việc sử dụng kỹ thuật hold down để ngăn ngừa vấn đề đếm vô hạn có thể làm cho RIP vô cùng kém hiệu quả trên mạng diện rộng. Việc quảng bá luôn luôn tiêu tốn rất nhiều băng thông. Ngay cả khi không xảy ra sự khủng hoảng, việc có nhiều máy quảng bá theo định kỳ có nghĩa là giao thông sẽ gia tăng khi số lượng bộ định tuyến gia tăng. Sự tiềm tàng của các vòng lặp định tuyến cũng thật là nguy hiểm khi mà dung lượng của đường truyền bị giới hạn. Một khi đường truyền bị tràn ngập bởi các packet (của vòng lặp), sẽ vô cùng khó khăn hoặc là không thể nào để các bộ định tuyến có thể trao đổi các thông điệp định tuyến nhằm phá vỡ các vòng lặp. Mặc dù có nhiều vấn đề đã được biết, có nhiều nhóm vẫn tiếp tục sử dụng RIP như là một IGP trong các mạng diện rộng.

*8.5.2.5. Giao thức định tuyến RIP version 2*

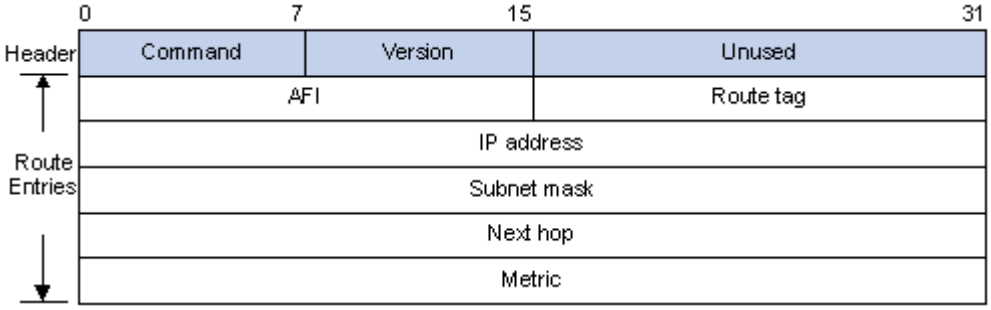
RIP v2 có khuôn dạng giống RIP nhưng nó bỏ qua trường "set to 0" như hình 8.26

Miền định tuyến (Routing domain) là phần định danh của tiến trình định tuyến ngầm định (routing daemon).

Route tag hỗ trợ EGP, nó nhớ một số hiệu hệ thống độc lập AS (Autonomous System) của EGP và BGP.

Mặt nạ mạng con (Subnet mask) của mỗi tuyến áp dụng cho địa chỉ IP tương ứng.

Địa chỉ Next host IP là địa chỉ các gói dữ liệu tới địa chỉ IP đích tương ứng cần được gửi tới. Giá trị 0 trong trường này có nghĩa là các gói dữ liệu tới đích nên được gửi đến hệ thống đã gửi thông điệp RIP.




*Hình 8.26: Gói RIP v2*

*8.5.2.6. Việc truyền các thông điệp RIP*

Các thông điệp RIP không chứa vùng độ dài tường minh và cũng không chứa tường minh biến đếm các mục. Thay vì thế, RIP giả định rằng cơ chế chuyển phát cơ sở sẽ báo cho nơi nhận độ dài của thông điệp gửi đến. Cụ thể, khi được sử dụng với TCP/IP, các thông điệp RIP dựa vào UDP để thông báo nơi nhận độ dài của thông điệp RIP hoạt động trên cổng số 520 của UDP. Mặc dù yêu cầu RIP có thể xuất phát từ những cổng UDP khác, nhưng cổng UDP đích của yêu cầu thì luôn luôn là 520, cũng như là cổng nguồn mà các thông điệp quảng bá RIP xuất phát.

*8.5.2.7. Khuyết điểm của biến đếm số trạm trong RIP*

Sử dụng RIP như là giao thức của bộ định tuyến nội sẽ giới hạn bộ định tuyến trong hai cách, trước hết, RIP ràng buộc bộ định tuyến vào giá trị của biến số đếm số trạm. Thứ hai, bởi vì nó sử dụng giá trị tương đối nhỏ của biến số đếm số trạm có thể hiện cho vô hạn, RIP đã gián tiếp giới hạn kích thước của Internet sử dụng nó. Cụ thể, RIP ràng buộc độ dài của Internet không được quá 16 (nghĩa là, khoảng cách tối đa). Như thế, một Internet sử dụng RIP có thể có nhiều lắm là 15 bộ định tuyến giữa hai máy bất kỳ.

Cần lưu ý rằng giới hạn về độ rộng của mạng không phải là giới hạn về tổng số bộ định tuyến cũng như không phải là giới hạn về độ dày đặc của mạng. Thực ra, hầu hết các mạng (trong viện đại học chẳng hạn) có độ rộng tương đối nhỏ ngay cả khi chúng có nhiều bộ định tuyến bởi vì chúng thường có *cấu hình phân cấp*. Lấy ví dụ, hãy thử xét một Internet thông thường của công ty. Hầu hết đều sử dụng cấu hình phân cấp bao gồm mạng backbone vào một nhóm làm việc, trong đó có

một mạng LAN. Mặc dù một công ty có thể bao gồm cả chục nhóm làm việc, nhưng độ rộng của toàn bộ Internet chỉ là 2. Ngay cả khi mỗi nhóm làm việc được mở rộng được mở rộng để thêm vào bộ định tuyến để nối thêm một hay nhiều mạng LAN, thì độ rộng tối đa cũng chỉ là 4. Tương tự, việc mở rộng cấu hình phân cấp một hay hai mức cũng chỉ làm tăng độ rộng thành 6. Như thế, giới hạn mà RIP áp đặt sẽ ảnh hưởng những hệ tự quản lớn hoặc những hệ tự quản không sử dụng cách tổ chức phân cấp.

Tuy nhiên, ngay cả trong trường hợp tốt nhất, biến đếm số trạm cũng chỉ cung cấp một thước đo đơn sơ về khả năng của mạng. Như thế việc sử dụng biến đếm số trạm không phải lúc nào cũng cho ra những tuyến đường có độ trì hoãn ít nhất hoặc có dung lượng cao nhất. Hơn thế nữa, việc tính các tuyến đường dựa trên cơ sở giá trị nhỏ nhất của biến đếm số trạm cùng bộc lộ khuyết điểm. Đó là, nó làm cho việc định tuyến tương đối tĩnh bởi vì các tuyến đường không thể đáp ứng với những thay đổi về mật độ giao thông trên mạng. Các phần tiếp theo sẽ tìm hiểu một cách tính khác, và giải thích vì sao cách tính theo biến đếm số trạm vẫn còn phổ biến mặc cho những giới hạn của chúng.

### 8.5.3. Giao thức Hello

Giao thức HELLO cung cấp cho ta một ví dụ về IGP mà sử dụng phương pháp tính tuyến đường khác với biến đếm số trạm. Mặc dù HELLO đã lỗi thời, nó có ý nghĩa đáng kể trong lịch sử của Internet bởi vì nó là IGP được sử dụng trong bộ định tuyến "fuzzball" ban đầu của backbone NSFNET (thuật ngữ fuzzball để chỉ những bộ định tuyến không có trong thương mại, bao gồm phần mềm giao thức được thiết kế đặc biệt chạy trên máy tính PDP11) HELLO đáng được trình bày ở đây bởi vì nó cung cấp một ví dụ về một giao thức mà sử dụng cách tính độ trì hoãn.

HELLO cung cấp hai chức năng: nó đồng bộ các đồng hồ trong một tập hợp máy tính và nó cho phép mỗi máy tính có thể tính những con đường có độ trì hoãn ngắn nhất để đi đến các đích như thế, các thông điệp HELLO chuyển tải thông tin timestamp cũng như là thông tin định tuyến. Ý tưởng cơ bản trong HELLO thật là đơn giản: mỗi máy tham gia vào những trao đổi HELLO sẽ duy trì một bảng của các ước tính tốt nhất của nó về các đồng hồ trong những máy lân cận. Trước khi truyền đi một packet, máy tính thêm vào timestamp của nó bằng cách sao chép giá trị đồng hồ hiện tại vào trong packet. Khi một packet đến nơi, nơi nhận sẽ tính một ước tính về độ trì hoãn trên đường truyền bằng cách lấy timestamp trong packet gửi đến trừ đi ước tính cục bộ của đồng hồ hiện tại trong máy lân cận. Theo định kỳ, các máy tính sẽ lấy thông tin từ các máy lân cận của chúng để thiết lập lại ước tính của các đồng hồ.

Các thông điệp HELLO cũng cho phép các máy tính tham gia tính những tuyến đường mới. Giao thức này sử dụng một mô hình đã được sửa đổi từ mô hình vector khoảng cách, trong đó sử dụng số đo về độ trì hoãn thay vì số trạm đi qua. Như thế, mỗi máy sẽ định kỳ gửi đến những máy lân cận một một bảng các đích mà nó có thể đếm và ước lượng thời hạn trì hoãn của mỗi đích. Khi một thông điệp gửi đến từ máy X, máy nhận kiểm tra mỗi giá trị trong thông điệp và thay đổi trạm kế thành X nếu tuyến đường đi qua X tốn ít chi phí hơn tuyến đường hiện tại (nghĩa là bất kỳ tuyến đường nào mà độ trì hoãn đi đến X cộng với độ trì hoãn độ trì hoãn từ X đến đích này lại ít hơn độ trì hoãn hiện tại để đi đến đích này).

### 8.5.4. Kết hợp RIP, Hello và BGP

Chúng ta đã quan sát được rằng chỉ một bộ định tuyến có thể sử dụng cả hai giao thức, giao thức cổng nội (IGP) để thu thập thông tin định tuyến bên trong hệ tự quản của nó và giao thức cổng ngoại (EGP) để thông báo các tuyến đường đến các hệ tự quản khác. Về nguyên lý, người ta có thể dễ dàng xây dựng chỉ một phần mềm mà kết hợp hai giao thức này, để có thể thu thập các tuyến đường và thông báo chúng mà không cần đến sự can thiệp của con người. Trong thực tế, những vướng mắc kỹ thuật cũng như là chính sách làm cho công việc này rất phức tạp.

Về mặt kỹ thuật, các giao thức IGP, như RIP và Hello, là những giao thức định tuyến. Một bộ định tuyến sử dụng các giao thức như thế để cập nhật bảng định tuyến của nó dựa vào thông tin nó lấy được từ các bộ định tuyến khác bên trong hệ tự quản của nó. Như thế, routed, chương trình UNIX mà cài đặt RIP, thông báo thông tin từ bảng định tuyến cục bộ và cũng thay đổi bảng định tuyến cục bộ khi nó nhận được các cập nhật. RIP tin tưởng rằng các bộ định tuyến ở trong cùng hệ tự quản chuyển đi dữ liệu chính xác.

Ngược lại, các giao thức cổng ngoại, như BGP không tin tưởng vào các bộ định tuyến trong những hệ tự quản khác. Kết quả là, các giao thức ngoại không thông báo tất cả các tuyến đường của bảng định tuyến cục bộ. Thay vì thế, các giao thức này lưu trữ một cơ sở dữ liệu về khả năng đi đến các mạng, và áp dụng các ràng buộc khi gửi và nhận thông tin. Việc bỏ qua các chính sách ràng buộc đó có thể ảnh hưởng đến việc định tuyến trong bối cảnh lớn hơn – có thể không đi đến được một số phần của Internet. Lấy ví dụ, nếu bộ định tuyến trong một hệ tự quản đang sử dụng RIP vô tình nhân bản một tuyến đường có chi phí thấp đến một mạng tai trường đại học Quốc Gia, trong khi không có tuyến đường như thế, thì các bộ định tuyến khác sử dụng RIP sẽ chấp nhận và cài đặt tuyến đường này. Sau đó chúng sẽ chuyển dữ liệu đi qua đại học Quốc Gia đến bộ định tuyến mà đã thực hiện sai sót này. Kết quả là, có lẽ rằng các máy tính trong hệ tự quản đó không thể nào đến được đại học Quốc Gia. Vấn đề trở lên nghiêm trọng hơn nếu các giao thức cổng ngoại không cài đặt các chính sách ràng buộc. Lấy ví dụ, nếu bộ định

tuyến ở biên trong hệ tự quản sử dụng BGP để nhân bản tuyến đường bất hợp lệ đến các hệ tự quản khác, thì một phần của Internet có thể không đi đến được mạng tại đại học Quốc Gia.

### 8.5.5. Định tuyến bên trong hệ tự quản

Chúng ta thấy rằng các EGP như là BGP cho phép một hệ tự quản thông báo thông tin định tuyến đến hệ khác. Tuy nhiên, cũng sẽ là hữu dụng khi cung cấp việc định tuyến bên trong một hệ tự quản, để mà các bộ định tuyến chọn những con đường có chi phí ít nhất. Để làm được điều này cần có thêm sự tin cậy lẫn nhau. Việc mở rộng ý niệm về sự tin cậy từ một hệ tự quản đến nhiều hệ tự quản là công việc phức tạp. Cách tiếp cận đơn giản nhất là nhóm các hệ tự quản lại theo kiểu phân cấp. Lấy ví dụ, chúng ta thử hình dung có ba hệ tự quản trong ba phân khoa khác nhau của một viện đại học lớn. Sẽ là điều tự nhiên khi nhóm ba phân khoa này lại bởi vì chúng cùng chia sẻ với nhau các yêu cầu quản lý. Động lực cho việc nhóm theo phân cấp chính là do ý niệm về sự tin cậy. Các bộ định tuyến bên trong một nhóm tin tưởng lẫn nhau với độ tin cậy cao hơn là các bộ định tuyến ở những nhóm khác.

Việc nhóm các hệ tự quản đòi hỏi sự mở rộng các giao thức định tuyến. Khi thông báo về khoảng cách, các giá trị phải được tăng lên khi đi qua biên giới từ nhóm này đến nhóm khác. Kỹ thuật này, đôi khi được gọi là biến đổi thước đo, phân chia các giá trị khoảng cách thành ba loại. Lấy ví dụ, giả sử các bộ định tuyến bên trong một hệ tự quản các giá trị khoảng cách nhỏ hơn 128. Chúng ta có thể đưa ra một quy tắc rằng khi truyền thông tin khoảng cách đi qua biên giới hệ tự quản bên trong một nhóm, giá trị khoảng cách phải được chuyển đổi qua vùng từ 128 đến 191. Cuối cùng, chúng ta cũng có thể đưa ra quy tắc rằng khi gửi giá trị khoảng cách qua biên giới giữa hai nhóm, các giá trị này phải được chuyển đổi qua vùng từ 192 đến 254. Tác dụng của các biến đổi này là hiển nhiên: đối với một mạng đích bất kỳ, bất kỳ con đường nào nằm hoàn toàn trong một hệ tự quản đều bảo đảm có chi phí thấp hơn con đường có đoạn nằm ngoài hệ tự quản này. Hơn thế nữa, trong số những con đường có phần nằm ngoài hệ tự quản, những con đường nào vẫn còn nằm trong một nhóm sẽ có chi phí thấp hơn những con đường đi qua biên giới của nhóm. Ưu điểm chính của kỹ thuật chuyển đổi thước đo là chúng cho phép mỗi hệ tự quản chọn một IGP và cũng cho phép các hệ khác có thể so sánh các chi phí định tuyến.

### 8.5.6. Giao thức định tuyến OSPF

#### *8.5.6.1. Đặc điểm và hoạt động của OSPF*

OSPF là một giao thức được thay thế cho RIP nhằm khắc phục được các hạn chế của giao thức RIP. OSPF là một giao thức dùng trạng thái liên kết (link state), trong khi đó RIP là một giao thức vector khoảng cách (distance vector), thông điệp được gửi đi bao gồm một vector khoảng cách (host count). Khi dùng RIP mỗi router cập nhật bảng định tuyến của nó dựa trên những vector khoảng cách mà nó nhận từ các router láng giềng.

Trong giao thức OSPF dùng trạng thái liên kết, một router không trao đổi khoảng cách với router láng giềng, thay vào đó mỗi router kiểm tra một cách tích cực các trạng thái liên kết của nó với mỗi router láng giềng, tức là có thể có nhiều tuyến đường tới một đích với các host count khác nhau nhưng nó không nhất thiết phải có một host count nhỏ nhất mà lúc ấy nó sẽ tính toán xem tuyến nào thuận lợi hơn thì nó sẽ chọn tuyến đó và nó gửi thông tin này tới các router khác.

Chẳng hạn như có một tuyến nào đó có số host count lớn hơn nhưng giải thông theo tuyến đó lại lớn hơn nhiều so với giải thông của tuyến có ít host count hơn thì nó sẽ chọn tuyến có giải thông lớn hơn. Sau đó các router láng giềng sẽ gửi thông tin ra toàn bộ hệ thống. Mỗi router nắm giữ thông tin về trạng thái liên kết này và xây dựng nên một bảng định tuyến đầy đủ. Một sự khác nhau quan trọng là giao thức trạng thái liên kết luôn hội tụ nhanh hơn giao thức vector khoảng cách, nghĩa là khả năng ổn định nhanh hơn sau khi có sự thay đổi, ví dụ như khi một router hoặc một liên kết bị hỏng. OSPF khác với RIP và các giao thức định tuyến khác là OSPF dùng IP trực tiếp. Nghĩa là nó không dùng TCP hoặc UDP để thiết lập liên kết. OSPF có giá trị riêng của trường protocol trong IP header.

*Đặc điểm OSPF*

OSPF có nhiều đặc điểm hơn hẳn RIP như sau:

- OSPF có thể tính toán một hoặc nhiều tuyến đối với mỗi loại dịch vụ IP. Nghĩa là đích nào có nhiều tuyến trong bảng định tuyến thì nó sẽ dành mỗi tuyến cho một loại dịch vụ IP.
- Khi các tuyến có giá trị cost cân bằng tới một đích, OSPF sẽ phân tán luồng dữ liệu cân bằng giữa các tuyến, gọi là Load balancing.
- OSPF hỗ trợ cho các mạng con (Subnet): Một mặt nạ mạng con (Subnet mask) được kết hợp với mỗi tuyến được quảng bá. Subnet mask cho phép một địa chỉ IP của bất kỳ lớp nào được phân ra thành nhiều mạng con khác nhau với các kích thước khác nhau. Các tuyến tới một host được quảng bá với một subnet

mask gồm toàn bit 1. Một tuyến mặc định (default) được quảng bá như một địa chỉ IP 0.0.0.0 với một Mask toàn bit 0.

- Một phương pháp xác thực đơn giản có thể được với giao thức này. Một mật khẩu dạng cleartext có thể được xác định như RIPv2.
- Các liên kết PPP giữa các router không cần một địa chỉ IP tại mỗi đầu của liên kết để tiết kiệm địa chỉ IP, gọi là mạng Unnumbered.
- Hệ thống dùng OSPF gửi thông điệp tới một nhóm địa chỉ dùng (Muiticasting) thay cho việc quảng bá ( quảng báing) để giảm thông lượng trên các đường truyền của hệ thống không dùng giao thức OSPF
- Không giống RIP, OSPF không bị hiện tượng định tuyến lặp (count to infinity), do đó số host của nó không bị giới hạn bởi 16 (tối đa lên tới 65535) do đó có thể chạy trong mạng có quy mô lớn hơn, cho phép gán một số lượng lớn các giá trị cost khác nhau phụ thuộc vào kiểu của mạng dựa trên một số đặc tính như giải thông. OSPF trao đổi thông tin định tuyến nhanh hơn RIP do không xảy ra trường hợp định tuyến lặp và do thông lượng cập nhật ngay lập tức, khắc phục hiện tượng mất liên kết hay hiện tượng lặp.
- OSPF trao đổi thông tin trên mạng ít hơn RIP, OSPF chỉ cập nhật thông tin khi có sự thay đổi trạng thái liên kết hoặc sau 30 phút do vậy chiếm ít thời gian lưu thông trên mạng hơn RIP (30 giây), dành giải thông cho dữ liệu.
- OSPF cho phép chọn đường trong các mạng có quy mô lớn nhờ có thêm một số tính năng linh hoạt sau:
- Loại dịch vụ (Type of Services): Chọn đường tuỳ theo kiểu dịch vụ, nhiều đường đi có thể được cấu hình cho những kiểu dịch vụ khác nhau, thí dụ như đường có thông lượng cao dành cho một số các dịch vụ có yêu cầu cao về dải thông...
- Cân bằng tải (Load Balancing): Do có nhiều đường sẵn sàng, dữ liệu được phân bổ những đường này sao cho sử dụng hạ tầng mạng được hiệu quả nhất.
- Chia AS (Subdivision of AS) quản lý mạng quy mô lớn bằng cách phân ra thành các vùng logic.
- An ninh (Security).
- Định tuyến cho host riêng cho mạng.

OSPF tổ chức theo kiểu hình cây, trên cùng là hệ thống AS trong hệ thống AS có các vùng, mỗi router chỉ nắm thông tin về cấu trúc mạng của một vùng nhất định (topological database) cùng các thông tin chọn đường trong vùng đó, cho phép nó quyết chọn đường theo các thông tin chọn đường trên mạng, dẫn đến 2 loại chọn đường sau:

Chọn đường trong AS (infraction routing) và chọn đường ngoài AS (inter routing). Các loại router OSPF:

- Router nội bộ (Internal router): chỉ có liên kết với một vùng, chỉ chạy một bản sao của thuật toán OSPF và lưu giữ thông tin về cấu hình trong vùng của nó.
- Router có liên kết ra bên ngoài (Area Border Router) có liên kết với nhiều hơn một vùng, chạy nhiều bản sao của thuật toán OSPF cùng với một bản sao về cấu hình mạng cho mỗi vùng mà nó liên kết và một bản sao cho AS xương sống (backbone AS). Để giảm thiểu lưu lượng chọn đường, Router có liên kết ra bên ngoài chỉ quan tâm đến cấu hình của vùng mà nó liên kết trong sự phân bổ của cả hệ thống AS. Những router khác trên cùng một backbone sẽ truyền những thông tin chọn đường từ router này tới các vùng khác.
- Các router nằm trên đường biên của hệ thống độc lập (AS boundary router) chuyển đổi thông tin chọn đường với các router ở các hệ thống AS khác. Các router này có những thông tin chọn đường ra ngoài hệ thống độc lập AS được quảng bá khắp hệ thống AS, các router trong hệ thống AS lưu giữ các thông tin trọn đường tới mỗi router.

* Như tên gọi của nó, đặc tả của giao thức được phổ biến rộng rãi việc làm cho nó trở thành một chuẩn mở mà mọi người có thể cài đặt mà không phải trả chi phí bản quyền, đã khuyến khích nhiều nhà sản xuất hỗ trợ OSPF. Kết quả là, nó đã trở lên phổ biến.

* OSPF bao gồm việc định tuyến theo kiểu của dịch vụ. Người quản lý có thể cài đặt nhiều tuyến đường đi đến một đích nào đó, mỗi tuyến đường dành cho một độ ưu tiên hay một loại dịch vụ. Khi gửi datagram đi, bộ định tuyến chạy OSPF sẽ sử dụng cả địa chỉ đích và vùng kiểm dịch vụ trong phần đầu IP để chọn tuyến đường OSPF là một trong cac giao thức TCP/ IP đầu tiên hỗ trợ việc định tuyến theo kiểu của dịch vụ.

* OSPF cung cấp việc *cân bằng giao thông*. Nếu người quản lý xác định nhiều tuyến đường đi đến một đích nào đó với cùng một chi phí, OSPF sẽ phân bổ giao thông đều nhau trên tất cả các tuyến đường này và OSPF cũng lại là một trong những IGP mở đầu tiên hỗ trợ việc cân bằng giao thông; Các giao thức như RIP tính chỉ một con đường đi đến mỗi đích.

* Để cho phép sự phát triển và làm cho các mạng tại mỗi đơn vị để quản lý hơn, OSPF cho phép một đơn vị phân chia các mạng và các bộ định tuyến của nó thành những tập hợp con gọi là *khu vực*. Mỗi khu vực là riêng biệt; kiến thức về cấu hình của một khu vực được che dấu đối với những khu vực khác. Như thế, nhiều nhóm bên trong một đơn vị có thể cùng hợp tác trong việc sử dụng OSPF

cho việc định tuyến mặc dù mỗi nhóm vẫn giữ lại khả năng thay đổi cấu hình mạng nội bộ của nó một cách độc lập.

* Giao thức OSPF xác định rằng tất cả những trao đổi giữa các bộ định tuyến có thể được *xác minh*. OSPF cho phép có những mô hình xác minh khác nhau và thậm chí cho phép một khu vực được quyền chọn một mô hình khác với khu vực khác. Ý tưởng nằm sau việc xác minh là để đảm bảo rằng chỉ những bộ định tuyến được tin cậy sẽ nhân bản thông tin định tuyến. Để hiểu được tại sao điều này có thể một vấn đề, chúng ta hãy thử xét xem điều gì có thể xảy ra khi sử dụng RIT1, mà không có việc xác minh. Nếu một người nghịch ngợm sử dụng máy tính cá nhân để nhân bản các thông điệp RIP thông báo về các tuyến đường có chi phí thấp, thì các bộ định tuyến khác và các máy tính sử dụng RIP sẽ thay đổi các tuyến đường của chúng và bắt đầu gửi datagram đến máy tính cá nhân này.

* OSPF bao gồm việc hỗ trợ cho các tuyến đường không phân lớp, theo máy cụ thể, và theo mạng con cụ thể, cũng như là các tuyến đường theo mạng phân lớp cụ thể. Có lẽ trong một Internet lớn sẽ cần đến tất cả các kiểu này.

* Để dung nạp các mạng như Ethernet, OSPF đã mở rộng thuật giải SPF đã trình bày trong phần đầu chương. Chúng ta đã mô tả thuật giải này sử dụng đồ thị điểm nối điểm và đã nói rằng mỗi bộ định tuyến chạy SPF sẽ định kỳ quảng bá các thông điệp trạng thái liên kết của mỗi máy lân cận và có thể đi đến được.

* Nếu có K bộ định tuyến nối vào một Ethernet, chúng ta sẽ quảng bá $K^2$ thông điệp trạng thái liên kết OSPF giảm thiểu số quảng bá bằng cách cho phép một cấu hình đồ thị phức tạp hơn, trong đó mỗi nút thể hiện cho bộ định tuyến hoặc cho mạng. Kết quả là, OSPF cho phép các mạng đa truy xuất (như Ethernet) có một *cổng được chỉ định* (nghĩa là một *bộ định tuyến được chỉ định*) để gửi đi các thông điệp trạng thái liên kết, đại diện cho tất cả các bộ định tuyến nối vào mạng này; những thông điệp này thông báo trạng thái của tất cả các liên kết từ mạng này đến các bộ định tuyến nối vào nó. OSPF cũng sử dụng khả năng quảng bá phần cứng, khi chúng tồn tại, để chuyển phát các thông điệp trạng thái liên kết.

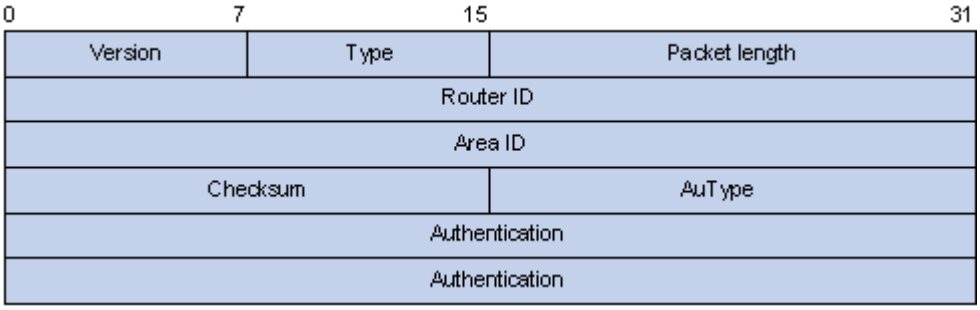
* Để cho phép độ uyển chuyển tối đa, OSPF cho phép người quản lý mô tả cấu hình mạng ảo mà có thể rất trừu tượng so với các chi tiết về những kết nối vật lý. Lấy ví dụ, người quản lý có thể cấu hình một liên kết ảo giữa hai bộ định tuyến này đòi hỏi việc thông tin liên lạc qua một mạng trung gian.

* OSPF cho phép các bộ định tuyến trao đổi thông tin định tuyến nó được học từ đơn vị khác (bên ngoài). Về cơ bản, một hay nhiều bộ định tuyến có các kết nối đến những đơn vị khác sẽ học được thông tin về các đơn vị đó và đưa nó vào khi gửi các thông điệp cập nhật. Định dạng của thông điệp phân biệt được thông tin lấy

được từ nguồn bên ngoài thông tin lấy được từ các bộ định tuyến bên trong đơn vị, vì vậy không có sự nhầm lẫn về nguồn hay độ tin cậy của các tuyến đường.

*8.5.6.2. Định dạng thông điệp OSPF*

Mỗi thông điệp OSPF bắt đầu bởi một phần đầu cố định 24 byte, như trong hình 16.7:

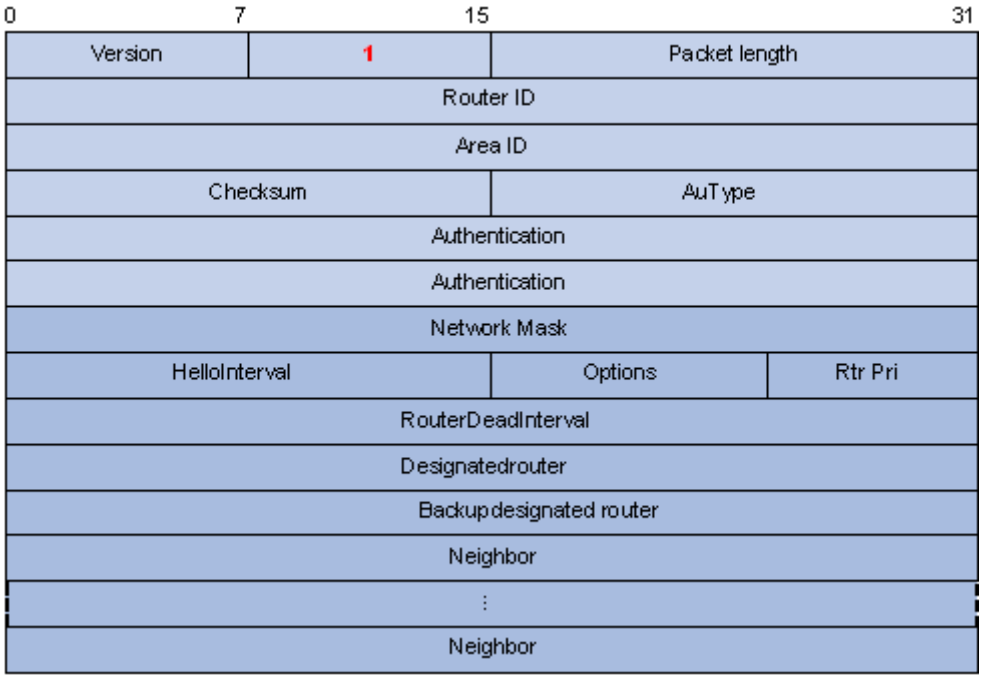



*Hình 8.28: Định dạng thông điệp OSPF*

- Vùng VERSION xác định phiên bản của giao thức.
- Vùng TYPE xác định kiểu của thông điệp, theo bảng sau:
    - 1: Hello
    - 2: Database description
    - 3: Link status request
    - 4: Link status update
    - 5: Link status acknowledgment
- Vùng có tên SOURCE ROUTER IP ADDRESS cho ta địa chỉ của nơi gửi,
- Vùng có tên AREA ID là con số định danh 32 bit của khu vực này.
- Bởi vì mỗi thông điệp có thể bao gồm việc xác minh, nên vùng AUTHENTICATION TYPE xác định mô hình xác minh nào được sử dụng hiện tại, ∅ có nghĩa là không xác minh và 1 có nghĩa là sử dụng một password đơn giản).

*8.5.6.3. Định dạng thông điệp Hello của OSPF*

OSPF gửi thông điệp Hello trên mỗi liên kết theo định kỳ để thiết lập và kiểm tra khả năng đi đến cac máy lân cận. Hình 8.29 trình bày định dạng của Hello.




*Hình 8.29: Định dạng thông điệp OSPF HELLO*

- Vùng NETWORK MASK chứa mặt nạ của mạng mà qua đó thông điệp được gửi đi.

- Vùng DEADINTERVAL cho giá trị thời gian tính bằng giây; Sau thời hạn đó nếu máy lân cận không đáp lời thì được xem như đã "chết".

- Vùng HELLO INTERVAL khoảng cách thời gian, tính bằng giây, giữa các thông điệp Hello. Vùng GWAY PRIO là độ ưu tiên của bộ định tuyến này, tính theo số nguyên và được sử dụng trong việc chọn máy dự phòng của bộ định tuyến được chỉ định.

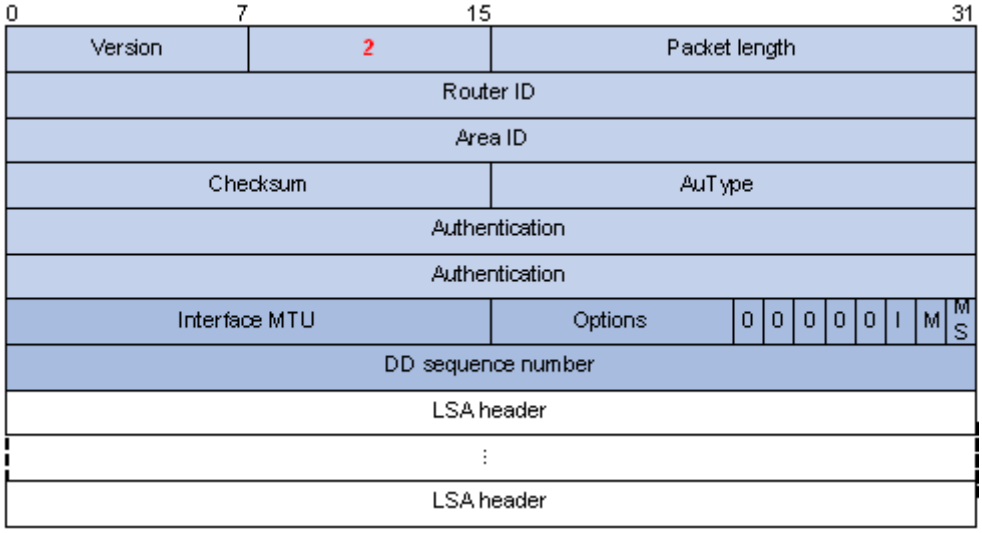
- Các vùng có tên DESIGNATED ROUTER và BACKUP DESIGNATED ROUTER chứa các địa chỉ IP của bộ định tuyến được chỉ định và bộ dự phòng cho bộ định tuyến được chỉ định của mạng mà qua đó thông điệp được gửi đi.

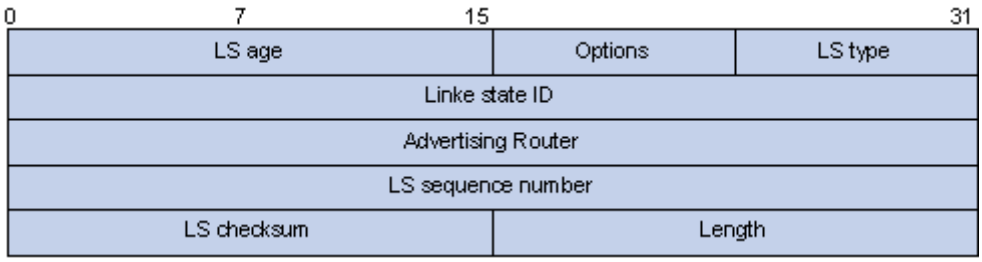
- Cuối cùng, các vùng có tên NEIGHBOR IP ADDRESS chứa địa chỉ IP của tất cả các máy lân cận mà nơi gửi vừa mới nhận các thông điệp hello từ đó.

### *8.5.6.4. Định dạng thông điệp "Database description" của OSPF*

Các bộ định tuyến trao đổi nhau thông điệp OSPF database description để khởi động cơ sở dữ liệu cấu hình mạng của chúng. Khi trao đổi, một bộ định tuyến đóng vai trò đóng vai trò chủ, còn những cái khác đóng vai trò thứ. Những cái thứ đáp lời lại mỗi thông điệp database description. Hình 8.30 trình bày định dạng của thông điệp.

Bởi vì nó có thể rất lớn, nên cơ sở dữ liệu cấu hình có thể được phân chia ra thành một số thông điệp bằng cách sử dụng các bit I và M. Bit I được lập lên 1 trong thông điệp khởi động; bit M được lập lên 1 nếu có thêm thông điệp tiếp theo sau. Bit S để chỉ rằng thông điệp được gửi đi máy chủ (1) hay máy thứ (0). Vùng DATABASE SEQUENCE NUMBER đánh số thứ tự các thông điệp để nơi nhận có thể biết được cái nào bị mất. Thông điệp khởi động chứa một số nguyên ngẫu nhiên R; các thông điệp tiếp theo sau chứa các số nguyên tiếp theo từ R.




*Hình 8.30: Định dạng thông điệp OSPF DD và LSA Header*

Các vùng từ LINK TYPE đến LINK AGE mô tả một liên kết trong cấu hình mạng; chúng được lập lại cho mỗi liên kết. Vùng LINK TYPE mô tả một liên kết theo các loại trong danh sách sau đây.

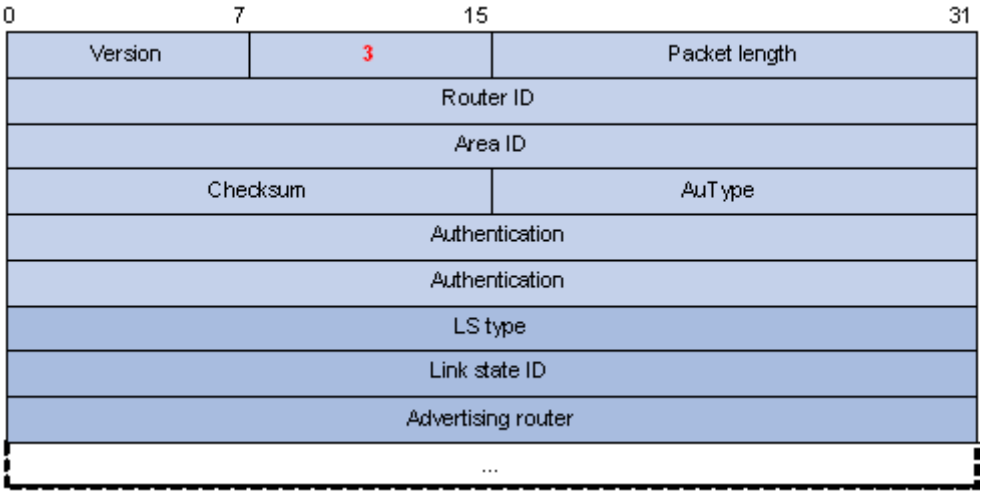
- 1: Router link
- 2: Network link
- 3: Summary link (IP net)
- 4: Summary link (link to border route)
- 5: External link (link to another site)

Vùng LINK ID cho ta định danh của liên kết (mà có thể là địa chỉ IP của bộ định tuyến hay mạng, tuỳ thuộc vào kiểu liên kết)

Vùng *ADVERTSING ROUTER* xác định địa chỉ của bộ định tuyến thông báo liên kết này và vùng *LINK SEQUENCE NUMBER*, chứa một số nguyên được phát sinh bởi bộ định tuyến đó để bảo đảm rằng các thông điệp không bị thất lạc hoặc là không bị mất thứ tự khi nhận. Vùng *LINK CHECKSUM* cung cấp sự bảo đảm hơn nữa thông tin liên kết không bị hư hỏng. Cuối cùng, cùng LINK AGE cùng giúp đỡ các thông điệp theo thứ tự nó cho ta thời gian, tính bằng giây từ khi liên kết được thiết lập.

*8.5.6.5. Định dạng thông điệp "Link Status Request" của OSPF*

Sau khi trao đổi các thông điệp "database description" với máy lân cận, bộ định tuyến có thể phát hiện ra rằng một phần của cơ sở dữ liệu của nó không được cập nhật (với thông tin mới nhất). Để yêu cầu các máy lân cận cung cấp thôgn tin cập nhật, bộ định tuyến gửi đi thông điệp "Link Status Request". Thông điệp này liệt kê các liên kết cụ thể, như trong hình 8.31. Các máy lân cận sẽ đáp lời với thông tin mới nhất mà nó có được về các liên kết đó. Có ba vùng được lặp lại cho mỗi liên kết về trạng thái nào được yêu cầu. Có thể cần nhiều hơn một thông điệp nếu danh sách yêu cầu quá dài.

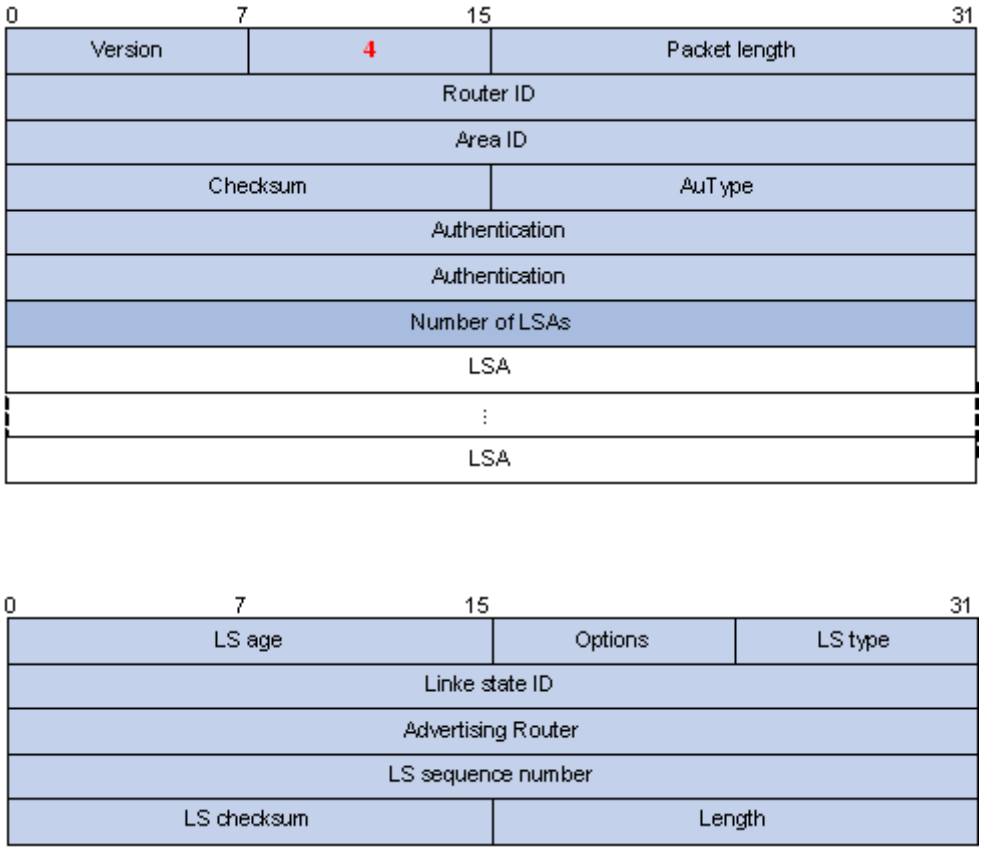



*Hình 8.31: Định dạng thông điệp OSPF LSR*

*8.5.6.6. Định dạng thông điệp "Link Status Update" của OSPF*

Các bộ định tuyến quảng bá trạng thái của các liên kết với thông điệp "Link Status Update". Mỗi cập nhật bao gồm một danh sách các thông báo, như trình bày trong hình 8.32

Mỗi thông báo về trạng thái liên kết có một phần đầu được định dạng như trong hình 8.32. Các giá trị được sử dụng trong mỗi vùng có nghĩa như trong thông điệp "database desciption".




*Hình 8.32: Định dạng thông điệp OSPF LSU và LSA header*

Tiếp theo sau phần đầu trạng thái liên kết là một trong bốn định dạng có thể có để mô tả các liên kết từ một bộ định tuyến đến một khu vực nào đó, các liên kết từ một bộ định tuyến đến một mạng cụ thể, các liên kết từ một bộ định tuyến đến các mạng vật lý mà kết hợp thành một mạng con IP, hay các liên kết từ một bộ định tuyến đến các mạng ở những đơn vị khác. Trong tất cả mọi trường hợp, vùng LINK TYPE trong phần đầu của trạng thái liên kết xác định định dạng nào đã được sử dụng. Như thế, bộ định tuyến mà nhận thông điệp "Link Status Update" biết được chính xác những đích được mô tả nào nằm bên trong đơn vị và những đích nào nằm ngoài.

### 8.5.7. Một số biện pháp đảm bảo an toàn định tuyến

Trên siêu xa lộ thông tin, các host được kết nối với nó có nguy cơ bị đe doạ. Hiện nay, khi Internet trở thành xa lộ không an toàn, router chưa thực sự có bức chắn bảo vệ ngăn ngừa. Đúng hơn là, tầm quan trọng còn đặt trên mức độ bảo vệ an toàn tại các host. Điều này tương tự như bảo vệ nhà, người ta khoá cửa lại để chống lại kẻ xâm phạm (host machines) hơn là ngăn chặn trên đường phố. Một số

có thể cho rằng chúng ta có thể vẫn đặt nhiều đội tuần tra an toàn và các trạm kiểm soát trên đường.

Sau đây là một số biện pháp an toàn có thể thực hiện bởi router nhằm chống kẻ trộm trên xa lộ.

**Padlock the routers (**khoá tại router**).** Chủ nhân của router có thể đặt router của họ trong vùng an toàn và khoá chúng lại. Ví dụ để thực hiện dự án có tên WorldPlus của AT&T, các router của Cisco được sử dụng để cho phép kết nối giữa các site WorldPlus. AT&T cung cấp dịch vụ StarWAN bao gồm các router. Nhà cung cấp dịch vụ thường đặt một khoá trong router của họ và chỉ có người quản trị site mới có quyền truy cập tới khoá.

Encrypt data files (mã hoá các tệp dữ liệu): Khi các tệp ứng dụng ra khỏi các máy, chúng có thể được mã hoá tại mức router nguồn và giải mã tại router đích.

Build firewalls (xây dựng các bức tường lửa). Firewall là các hệ thống máy tính, không phải các router. Một hệ thống firewall thay thế IP router với một máy tính không cho phép các gói chuyển tiếp. Bằng cách không cho phép các gói chuyển tiếp giữa các mạng, firewall hoạt động như một proxy cho những người dùng bảo vệ điều bất hạnh.

Bình thường để chuyển qua một router các gói dữ liệu được chuyển thông qua tầng IP mà không có hạn chế. Trong hệ thống firewall, không có gói nào được chuyển qua. Chỉ có các gói có địa chỉ do máy firewall kiểm soát mới được phép chuyển qua.

Filter out packets at the router level (lọc các gói tại mức router). Kỹ thuật này được thực hiện theo công nghiệp router. Nó được thực hiện trong một bảng. Người quản trị router có thể chỉ rõ cách thức người dùng muốn chống lại việc truy cập bằng cách chỉ ra tên host và các số hiệu cổng tương ứng.

Authenticate at the router protocol level (xác thực tại mức giao thức định tuyến). Giao thức RIP 2 và OSPF đều có trường dùng cho việc xác thực (authentication field) (xem phần giao thức RIP và OSPF).

Để có được các chế độ bảo vệ tốt hơn ở mức giao thức, các nghiên cứu trong tương lai có thể được thực hiện để tìm ra các thuật toán xác thực tốt hơn.

**Câu hỏi và bài tập**

8.1 Trình bày nguyên lý routing

8.2 Khái niệm bảng dẫn đường

8.3 Hệ tự quản và hệ đồng đẳng

8.4 Giao thức IGP và EGP

8.5 Hoạt động của RIP

8.6 Hoạt động của OSPF

8.7 Hoạt động của BGP

8.8 Sử dụng phần mềm Packet Tracert thực hành cấu hình các giao thức; RIP v1, RIP v2, OSPF, EIGRP theo các topo mạng khác nhau.

8.9 Sử dụng phần mềm Packet Tracert thực hành cấu hình routing tĩnh trên các topo mạng khác nhau.

# PHỤ LỤC 1

# THUẬT TOÁN DIJKSTRA TÌM ĐƯỜNG ĐI NGẮN NHẤT

## 1. Mô tả thuật toán

Đồ thị có trọng số là đồ thị G=(V,E) mà mỗi cạnh (hoặc cung) $e \in E$ được gán bởi một số thực m(e), gọi là trọng số của cạnh (hoặc cung) e.

Trong phần này, trọng số của mỗi cạnh được xét là một số dương và còn gọi là chiều dài của cạnh đó. Mỗi đường đi từ đỉnh u đến đỉnh v, có chiều dài là m(u,v), bằng tổng chiều dài các cạnh mà nó đi qua. Khoảng cách d(u,v) giữa hai đỉnh u và v là chiều dài đường đi ngắn nhất (theo nghĩa m(u,v) nhỏ nhất) trong các đường đi từ u đến v.

Cho đơn đồ thị liên thông, có trọng số G=(V,E). Tìm khoảng cách d($u_0$,v) từ một đỉnh $u_0$ cho trước đến một đỉnh v bất kỳ của G và tìm đường đi ngắn nhất từ $u_0$ đến v.

## 2. Phương pháp của thuật toán Dijkstra:

Xác định tuần tự đỉnh có khoảng cách từ $u_0$ đến các đỉnh khác.

Trước tiên, đỉnh có khoảng cách đến a nhỏ nhất chính là a, với d($u_0$,$u_0$)=0. Trong các đỉnh $v \neq u_0$, tìm đỉnh có khoảng cách $k_1$ đến $u_0$ là nhỏ nhất. Đỉnh này phải là một trong các đỉnh kề với $u_0$. Giả sử đó là $u_1$. Ta có: $d(u_0,u_1) = k_1$.

Trong các đỉnh $v \neq u_0$ và $v \neq u_1$, tìm đỉnh có khoảng cách $k_2$ đến $u_0$ là nhỏ nhất. Đỉnh này phải là một trong các đỉnh kề với $u_0$ hoặc với $u_1$. Giả sử đó là $u_2$. Ta có:

$d(u_0,u_2) = k_2$.

Tiếp tục như trên, cho đến bao giờ tìm được khoảng cách từ $u_0$ đến mọi đỉnh v của G.

Nếu $V=\{u_0, u_1, ..., u_n\}$ thì:

$0 = d(u_0,u_0) < d(u_0,u_1) < d(u_0,u_2) < ... < d(u_0,u_n)$.

## 3. Thuật toán Dijkstra

procedure Dijkstra (G=(V,E) là đơn đồ thị liên thông, có trọng số với trọng số dương)

{G có các đỉnh $a=u_0, u_1, ..., u_n=z$ và trọng số $m(u_i, u_j)$, với $m(u_i, u_j) = \infty$ nếu $(u_i, u_j)$ không là một cạnh trong G}

**for** i := 1 **to** n

$L(u_i) := \infty$

$L(a) := 0$

$S := V \setminus \{a\}$

$u := a$

**while** $S \neq \varnothing$

**begin**

**for** tất cả các đỉnh v thuộc S

**if** L(u) +m(u, v) < L(v) **then** L(v) := L(u)+m(u, v)

u := đỉnh thuộc S có nhãn L(u) nhỏ nhất

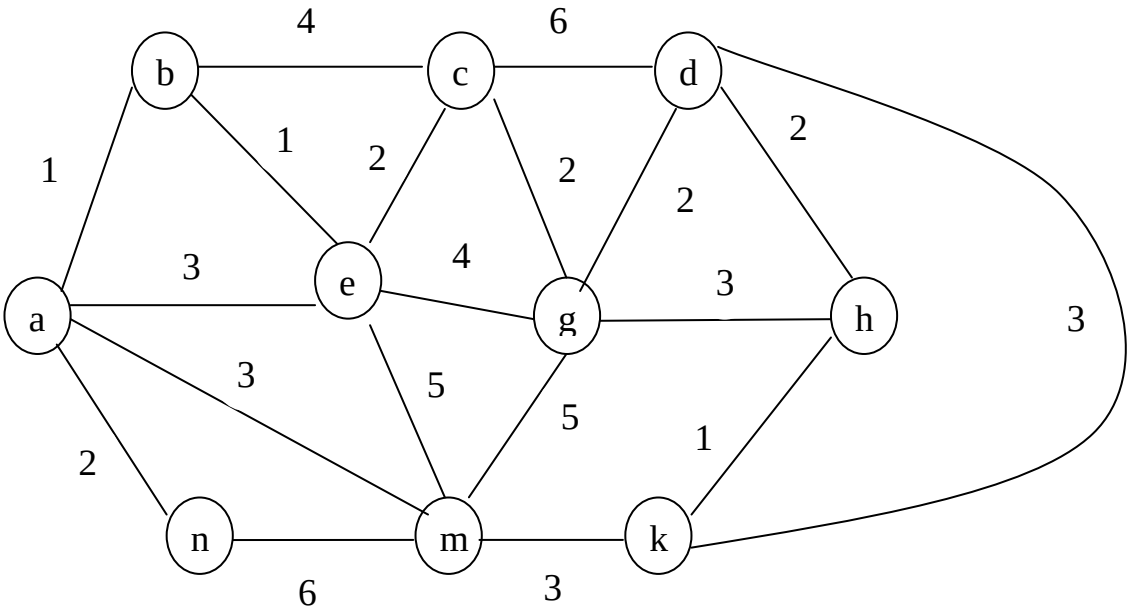
{L(u): độ dài đường đi ngắn nhất từ a đến u}

S := S \ {u}

**end**

## 4. Ví dụ

Tìm khoảng cách d(a,v) từ a đến mọi đỉnh v và tìm đường đi ngắn nhất từ a đến v cho trong đồ thị G sau.




**Bài làm**

| L(a) | L(b) | L(c) | L(d) | L(e) | L(g) | L(h) | L(k) | L(m) | L(n) |
|---|---|---|---|---|---|---|---|---|---|
| 0 | ∞ | ∞ | ∞ | ∞ | ∞ | ∞ | ∞ | ∞ | ∞ |
| - | 1 | ∞ | ∞ | 3 | ∞ | ∞ | ∞ | 3 | 2 |
| - | - | 5 | ∞ | 2 | ∞ | ∞ | ∞ | 3 | 2 |
| - | - | 5 | ∞ | 2 | ∞ | ∞ | ∞ | 3 | - |
| - | - | 4 | ∞ | - | 6 | ∞ | ∞ | 3 | - |
| - | - | 4 | ∞ | - | 6 | ∞ | 6 | - | - |
| - | - | - | 10 | - | 6 | ∞ | 6 | - | - |
| - | - | - | 8 | - | - | 9 | 6 | - | - |
| - | - | - | 8 | - | - | 7 | - | - | - |
| - | - | - | 8 | - | - | - | - | - | - |

# PHỤ LỤC 2

# THUẬT TOÁN BELLMAN-FORD

## 1. Mô tả bài toán

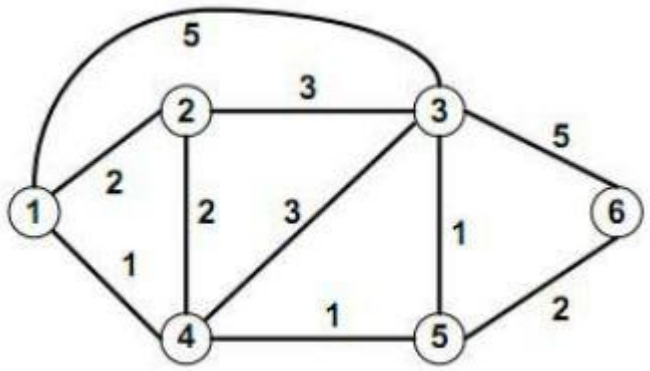
- **Đầu vào:**
  - Đồ thị G(V,E): với V là tập đỉnh, E là tập cạnh có trọng số.
  - Đỉnh nguồn S: $S \in V$
- **Ký hiệu:**
  - $D(h)_i$: đường đi ngắn nhất từ node nguồn S đến i có tối đa h đoạn.
  - $D_{ij}$: trọng số trên cạnh nối từ node i đến node j.
    - $D_{ij} = 0$ nếu i trùng j
    - $D_{ij} \neq 0$ nếu i khác j
- **Đầu ra:**
  - Đường đi ngắn nhất từ nguồn S đến tất cả các đỉnh còn lại

## 2. Thuật toán Bellman-Ford

- **Bước 1: Khởi động**
  - $D(l)_N = d_{SN}, \forall N \in V \setminus \{S\}$ **(**đường đi ngắn nhất từ S đến N có tối đa l đoạn)
- Bước 2: Cập nhật đường đi ngắn nhất
  - $D(h+l)_N = \min\{D(h)_j + d_{jN}\}$ với $\forall j \in V \setminus \{S\}$
- Bước 3: Lặp lại bước 2 cho đến khi không có đường đi mới nào ngắn hơn được tìm thấy thì dừng
- Kết quả: $D(h)_N$ sẽ là đường đi ngắn nhất từ node nguồn S đến node N

## 3. Ví dụ:

Cho đồ thị như sau:

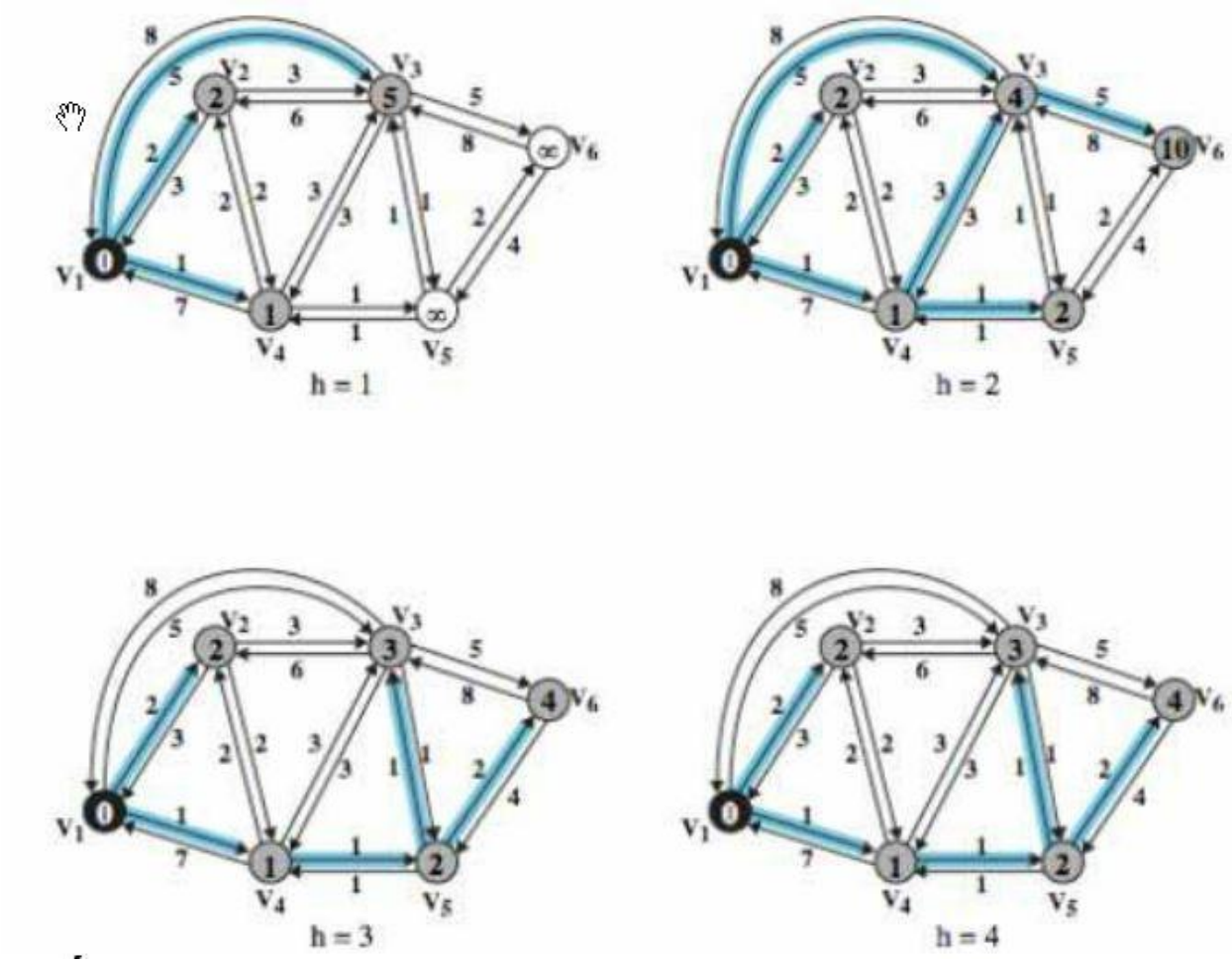


Tìm đường đi ngắn nhất từ đỉnh 1 tới đỉnh 6 với chi phí cho trên đồ thị

**Bài làm**

Ta có

| Lần chạy | $D(h)_2$ | Node 2 Path | $D(h)_3$ | Node 3 Path | $D(h)_4$ | Node 4 Path | $D(h)_5$ | Node 5 Path | $D(h)_6$ | Node 6 Path |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 1 – 2 | 5 | 1 – 3 | 1 | 1-4 | ∞ | … | ∞ | … |
| 2 | 2 | 1 - 2 | 4 | 1-4-3 | 1 | 1-4 | 2 | 1-4-5 | 10 | 1-3-6 |
| 3 | 2 | 1 – 2 | 3 | 1-4-5-3 | 1 | 1-4 | 2 | 1-4-5 | 4 | 1-4-5-6 |
| 4 | 2 | 1 - 2 | 3 | 1-4-5-3 | 1 | 1-4 | 2 | 1-4-5 | 4 | 1-4-5-6 |




Như vậy đường đi ngắn nhất là đi các đỉnh: 1 – 4 -5 -6

# TÀI LIỆU THAM KHẢO


[1] *Giáo trình Công nghệ mạng*, Viện đào tạo Công nghệ thông tin - Đại học Quốc gia Hà nội, (2010).

[2] Vũ Duy Lợi. *Giáo trình lý thuyết mạng thông tin máy tính*. Viện Công nghệ thông tin, (2010).

[3] Nguyễn Thúc Hải, *Mạng máy tính*, Nhà xuất bản giáo dục, (2008).

[4] Học viện bưu chính viễn thông, *Giáo trình mạng*, (2011).

[5] Nguyễn Quốc Cường, *Internetworking với TCP/IP 2 tập*, NXB Giáo dục, (2001).

[6] Andrew S.Tanenbaun, *Computer Networks*, Fourth Edition, Pretice Hall, (2003).

[7] Microsoft Corp. - *Network Esential* - Nhà xuất bản giáo dục, (1999).

[8] IBM Corporation, *TCP/IP Basic*, (1999).

[9] W. Richard Stevens-Gary R. Wright, *TCP/IP Illustrated*, Vol2, (2001).

[10] Microsoft Corporation, *TCPIP Network Managment*, (2003).

[11] Học viện bưu chính viễn thông, *Giáo trình mạng*, (2003).

[12] Học viện kỹ thuật mật mã, *Giáo trình mạng*, (2002).

[13] Đại học Bách khoa Hà nội, *Tập bài giảng TCP/IP*, (2006).

[14] Https://en.wikipedia.org/wiki/List_of_TCP_and_UDP_port_numbers.