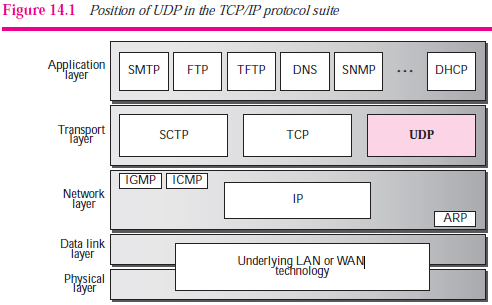
**User Datagram Protocol (UDP)**

Giao thức truyền gói tin của người dùng

**14.1 Introduction**

Giao thức UDP nằm giữa lớp Application và lớp Network, có nhiệm vụ làm trung gian giữa các ứng dụng trên lớp Application và những xữ lý ở lớp Network.



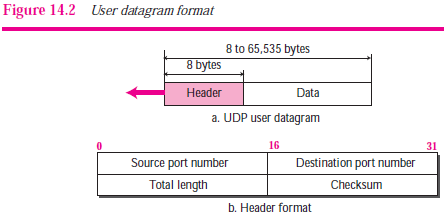
Lớp Transport có nhiều vai trò. Một trong số đó là tạo ra giao tiếp giữa các tiến trình; UDP sử dụng những port number để thực hiện nhiệm vụ này. Một vai trò khác của lớp Transport là để cung cấp những kỹ thuật kiểm soát tại lớp này. UDP thực hiện nhiệm vụ này ở mức tối thiểu. Không có kỹ thuật kiểm soát luồng các packet đã nhận được và cũng không có sự ghi nhận lại chúng. Tuy nhiên, UDP cũng cung cấp việc kiểm soát lỗi ở một mức độ nào đó. Nếu UDP phát hiện lỗi trong packet nhận được, nó sẽ loại bỏ packet đó mà không thông báo gì.

UDP là một giao thức transport không kết nối và không tin cậy. Nó không thêm bất cứ gì vào những dịch vụ của IP ngoại trừ việc cung cấp sự giao tiếp giữa các tiến trình thay cho giao tiếp giữa các host.

Nếu UDP tồi như thế, tại sao lại sử dụng nó? UDP là một giao thức đơn giản sử dụng ít chi phí nhất. Nếu một tiến trình muốn gửi một tin nhắn nhỏ và không quan tâm nhiều về độ tin cậy thì nó có thể sử dụng UDP. Gửi một tin nhắn nhỏ sử dụng UDP tốn ít sự tương tác qua lại giữa bên gửi và bên nhận hơn so với TCP.

**14.2 User Datagram**

Những packet UDP, gọi là user datagrams, có một header kích thước cố định là 8 bytes.



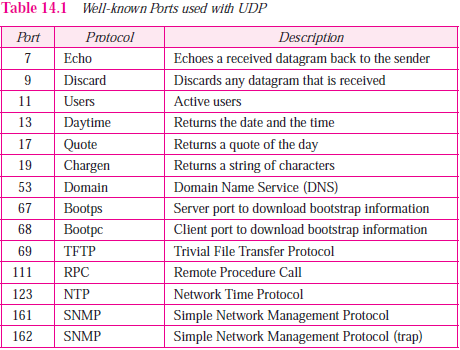
* **Source port number**. Đây là port được dùng bởi tiến trình của host gửi. Kích thước là 16 bits, tức là port number nằm trong khoảng 0 -> 65,535. Nếu host gửi là client, thì hầu hết trường hợp port number không cố định, được yêu cầu bởi tiến trình đang cần và được chọn bởi phần mêm UDP đang chạy trên host gửi. Nếu host gửi là server, thì hầu hết trường hợp port number làm một số có thể biết trước.
* **Destination port number**. Đây là port được dùng bởi tiến trình đang chạy trên host nhận. Kích thước là 16 bits. Nếu host nhận là server, thì hầu hết port number là có thể biết trước. Nếu host nhận là client, thì hầu hết port number làm một số không cố định. Trong trường hợp này, server sẽ copy port number đó của client đã nhận được trong request packet.
* **Length**. Đây là 1 field 16-bit dùng để xác định tổng độ dài của user datagram, header và cả dữ liệu. 16 bits có thể xác định tổng độ dài từ 0 -> 65,535 bytes. Tuy nhiên, tổng độ dài cần phải ít bởi vì một UDP user datagram được lưu trữ trong một IP datagram, với tổng độ dài là 65,535 bytes. Thực tế length field không cần thiết có trong một UDP datagram. Một user datagram được bao đóng trong IP datagram. Có một field trong IP datagram để xác định độ dài. Có một field khác trong IP datagram xác định độ dài của header. Vì vậy nếu ta trừ giá trị ở field thứ 2 cho field thứ 1, ta có thể suy ra độ dài của UDP datagram (được bao đóng trong IP datagram).  Tuy nhiên, người thiết kế UDP protocol cảm thấy hiệu quả hơn cho UDP đích để tính độ dài dữ liệu từ thông tin đã được cung cấp trong UDP user datagram hơn là yêu cầu phần mềm của IP để cung cấp thông tin này. Chúng ta nhớ rằng khi phần mềm của IP giao UDP user datagram cho lớp UDP, thì phải tháo IP header.
* **Checksum.** Field này dùng để phát hiện lỗi trong toàn bộ user datagram (header lẫn dữ liệu). Checksum sẽ được giới thiệu sau.

**14.3 UDP Services**

Chúng ta đã nói về những dịch vụ thông thường được cung cấp bởi giao thức lớp transport trong chap 13. Phần này, ta sẽ thảo luận về cụ thể những dịch vụ nào được UDP cung cấp.

**Process-to-Process Communication** (giao tiếp giữa tiến trình tới tiến trình)

UDP cung cấp giao tiếp giữa tiến trình tới tiến trình, đã nói trong Chapter 13 sử dụng socket, một sự kết hợp giữa những địa chỉ IP với port number. Nhiều port number được dùng bởi UDP như trong hình bên dưới.



**Connectionless Services** (những dịch vụ không dùng kết nối)

Như đã đề cập ở trên, UDP cung cấp một dịch vụ không kết nối. Nghĩa là mỗi user datagram được gửi bởi UDP là một datagram độc lập. Không có sự liên hệ nào giữa những user datagram khác nhau dù chúng xuất phát từ cùng 1 tiến trình và đến cùng một chương trình đích. Những user datagrams đó không được đánh số. Và cũng không có sự hình kết nối và không có hủy kết nối như trong trường hợp của TCP. Điều này có nghĩa là mỗi user datagram có thể di chuyển trên 1 đường độc lập.

Một khía cạnh của tính không kết nối là cái tiến trình mà sử dụng UDP không thể gửi 1 dòng dữ liệu tới UDP và mong đợi UDP cắt nhỏ chúng thành những user datagram có liên quan với nhau. Thay vào đó mỗi request phải nhỏ vừa đủ để nằm gọn trong một user datagram. Chỉ những tiến trình gửi những message ngắn, message phải ít hơn 65,507 bytes (65,535 trừ 8 bytes của UDP header và trừ 20 bytes của IP header), có thể dùng UDP.

**Flow Control** (kiểm soát mức độ gửi bên gửi cho phù hợp với bên nhận)

UDP là một giao thức rất đơn giản. Không có flow control, và vì vậy không có kỹ thuật cửa sổ trượt. Bên nhận có thể bị tràn buffer với lượng message được nhận. Thiếu flow control đồng nghĩa rằng tiến trình sử dụng UDP nên tự cung cấp dịch vụ này, nếu cần

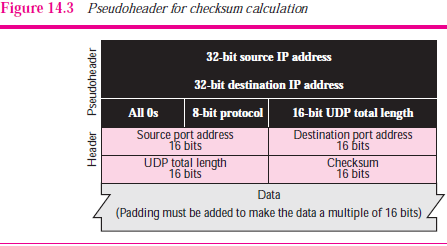
**Error Control** (kiểm soát lỗi)

Không có kỹ thuật kiểm soát trong UDP ngoại trừ checksum. Có nghĩa rằng bên gửi sẽ không thể biết message có bị lost hoặc trùng. Khi bên nhận phát hiện lỗi thông qua checksum, user datagram sẽ bị hủy bỏ mà không thông báo gì. Thiếu error control nghĩa là tiến trình dùng UDP nên tự cung cấp dịch vụ này nếu cần.

*Checksum*

Ta đã nói về thuật ngữ *checksum* và cách tính nó cho IP trong Chapter 7. Cách tính UDP checksum khác với cách trong IP. Ở đây checksum gồm 3 phần: 1 pseudoheader (header giả), UDP header, và dữ liệu từ lớp application.

**Pseudoheader** là một phần header của IP packet mà user datagram được bao đóng bên trong với một vài field chứa những số 0.



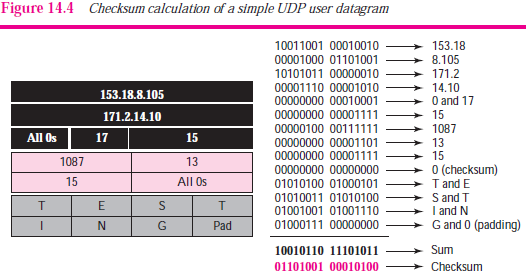
Nếu checksum không có pseudoheader, một user datagram có thể sẽ đến đích mà không bị vấn đề gì. Tuy nhiên, nếu IP header bị hư, nó có thể giao tới nhầm host.

Protocol field được thêm vào để đảm bảo packet thuộc về UDP chứ không phải TCP. Lát sau ta sẽ tìm hiểu xem liệu một tiến trình có thể dùng cả UDP hay TCP hay không, port number đích có thể giống nhau hay không. Giá trị trong protocol field cho UDP là 17. Nếu giá trị này bị thay đổi trong lúc truyền, việc tính checksum bên nhận sẽ phát hiện nó và UDP sẽ bỏ packet này. Nó không được gửi nhầm giao thức.

Chú ý: những điểm giống nhau giữa pseudoheader field và 12 bytes cuối của IP header.

Tùy chọn kèm theo của Checksum

Bên gửi UDP packet có thể chọn không tính checksum. Trong trường hợp này, checksum field được điền với tất cả là số 0 trước khi nó gửi đi. Trong trường hợp bên gửi tính checksum, nhưng nó kết quả toàn bộ là 0, thì checksum sẽ đổi tất cả thành 1 trước khi packet được gửi. Nói cách khác, bên gửi cộng 2 lần. Chú ý: điều này không tạo ra sự nhầm lẫn, rối rắm bởi vì giá trị của checksum không bao giờ tất cả số 1 trong tình huống bình thường.

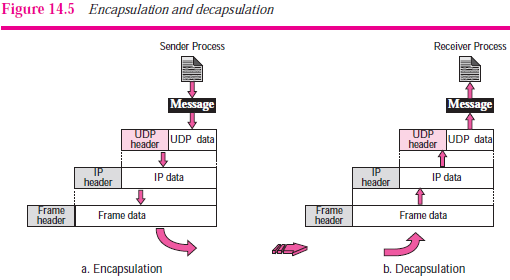


**Congestion Control** (kiểm soát cửa sổ trượt)

Bởi vì UDP là một giao thức không kết nối, nó không cung cấp congestion control. UDP cho rằng những packet được gửi là nhỏ và lẻ tẻ, và không thể tạo cửa sổ trượt trong network. Sự khẳng định này có thể hoặc không đúng ở hiện tại khi UDP được dùng cho truyền audio và video thời gian thực (real-time audio, real-time video).

**Encapsulation and Decapsulation** (đóng gói và mở gói)

Để gửi một message từ một tiến trình tới tiến trình khác, giao thức UDP gói gọn và mở gói những message



*Encapsulation*

Khi một tiến trình có một message cần gửi thông qua UDP, nó chuyển message đó tới UDP kèm theo một cặp địa chỉ sockets và thêm chiều dài của dữ liệu. UDP nhận dữ liệu và thêm UDP header. UDP tiếp theo sẽ chuyển user datagram tới IP với những địa chỉ socket. IP thêm header của chính nó, dùng giá trị 17 trong protocol field, để báo rằng dữ liệu này đến từ giao thức UDP. IP datagram đó tiếp theo được chuyển tới lớp data link. Lớp data link nhận IP datagram đó, rồi thêm vào header của chính nó, và chuyển tới lớp physical. Lớp physical mã hóa những bits thành những tín hiệu điện hoặc ánh sáng và gửi nó tới một cái máy từ xa.

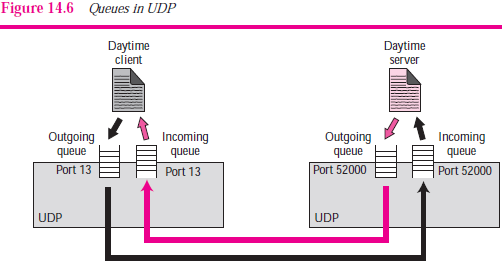
*Decapsulation*

Khi message đó đến tới host nhận, lớp physical sẽ giải mã những tín hiệu thành bits và chuyển lên lớp data link. Lớp data link dùng header để kiểm tra dữ liệu. Nếu không có lỗi, header sẽ được bỏ và datagram được chuyển tới IP. Phần mềm IP thực hiện việc kiểm tra. Nếu không có lỗi, header được bỏ và dữ liệu của ứng dụng cùng với địa chỉ socket bên gửi được chuyển tới tiến trình cần. Địa chỉ socket bên gửi được chuyển tới tiến trình trong trường hợp nó cần để phản hồi lại message đã nhận.

**Queuing**

Chúng ta vừa nói về những port mà chưa bàn đến cách chúng thực sự được thực hiện. Trong UDP, queues (những hàng đợi) được kết hợp với các ports.

Tại phía client, khi một tiến trình khởi động, nó yêu cầu một port number từ hệ điều hành. Một vài cách thực hiện thì tạo ra cả queue đi vào lẫn queue đi ra kết hợp với mỗi tiến trình. Những cách thực hiện khác thì chỉ tạo ra queue đi vào kết hợp với mỗi tiến trình.



Chú ý rằng thậm chí nếu một tiến trình muốn giao tiếp với nhiều tiến trình, nó tồn tại duy nhất một port number và một queue đi ra và một queue đi vào. Những queue này được mở bởi client, trong hầu hết trường hợp, mà được xác định bởi những port number không cố định. Những queue này chỉ hoạt động khi tiến trình đang chạy. Khi tiến trình kết thúc, những queue này sẽ bị hủy.

Tiến trình của client có thể gửi những message tới queue đi ra bằng cách dùng port number nguồn, được xác định trong yêu cầu. UDP loại bỏ những message từng cái từng cái một, và sau đó thêm UDP header, rồi giao chúng tới IP. Một queue đi ra có thể bị tràn. Nếu điều này xảy ra, hệ điều hành có thể yêu cầu tiến trình của client đợi trước khi gửi thêm bất kì message nào.

Khi một message tới một client, UDP kiểm tra để xem liệu một queue đi ra có được tạo cho cái port number mà được xác định trong port number field bên nhận của user datagram hay không. Nếu có một queue như vậy, UDP sẽ gửi user datagram đã nhận tới cuối queue. Nếu không có queue như vậy, UDP sẽ loại bỏ user datagram và yêu cầu giao thức ICMP gửi một *port unreachable* message tới server. Tất cả message đi tới cho một chương trình cụ thể trên client, dù chúng đến từ cùng hoặc khác server, đều được gửi tới cùng một queue. Một queue đi vào có thể bị tràn. Nếu điều này xảy ra, UDP loại bỏ user datagram và yêu cầu một port unreachable message để gửi tới server.

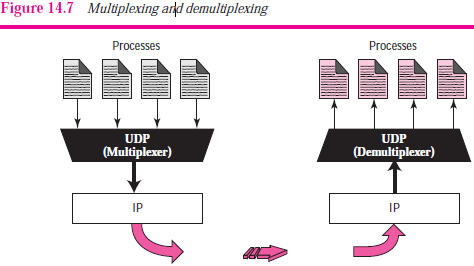
Tại phía server, kỹ thuật tạo các queue thì khác hẳn. Cách đơn giản nhất, một server yêu cầu queue đi vào và queue đi ra sử dụng port đã biết trước của nó khi khởi chạy. Những queue này luôn mở khi server đang chạy.

Khi một message đến server, UDP kiểm tra xem liệu một queue đi vào đã được tạo cho port number được xác định trong port number field nhận trong user datagram hay chưa. Nếu có, UDP gửi user datagram đã nhận tới cuối queue. Nếu chưa, UDP loại bỏ user datagram và yêu cầu giao thức ICMP gửi một port unreachable message tới client. Tất cả message đi tới cho một server cụ thê, dù chúng đến từ cùng hoặc khác client, được gửi trên cùng một queue. Một queue đi vào có thể bị tràn. Nếu điều này xảy ra, UDP loại bỏ user datagram và yêu cầu một port unreachable message để gửi tới client.

Khi một server muốn trả lờ một client, nó gửi những message tới queue đi ra sử dụng port number bên gửi ở trong request. UDP loại bỏ những message từng cái một và sau đó thêm UDP header, rồi giao chúng tới IP. Một queue đi ra có thể bị tràn. Nếu xảy ra, hệ điều hành yêu cầu server đợi trước khi gửi thêm bất kỳ message nào.

**Multiplexing and Demultiplexing** (ghép kênh và tách kênh)

Trong một host đang chạy một bộ giao thức TCP/IP, chỉ có duy nhất một UDP nhưng có thể có nhiều tiến trình muốn sử dụng những dịch vụ của UDP. Để xử lý tình huống này, ta dùng UDP ghép kênh và tách kênh.



*Multiplexing* (ghép kênh)

Tại bên gửi, có thể nhiều tiến trình cần gửi những user datagram. Tuy nhiên, chỉ có một UDP. Đây là quan hệ nhiều-một và cần ghép kênh. UDP chấp nhận những message từ các tiến trình khác nhau, phân biệt bằng port number chúng được gán. Sau khi thêm header, UDP chuyển các user datagram tới IP.

*Demultiplexing* (tách kênh)

Tại bên nhận, chỉ có một UDP. Tuy nhiên, chúng ta có thể có nhiều tiến trình cần nhận các user datagram. Đây là quan hệ một-nhiều và cần tách kênh. UDP nhận các user datagram từ IP. Sau khi kiểm tra lỗi và bỏ header, UDP giao từng message tới tiến trình thích hợp dựa vào các port number.

**Comparison between UDP and Generic Simple Protocol** (so sánh UDP và giao thức đơn giản thông thường)

Chúng ta có thể đối chiếu UDP với một giao thức đơn giản không kết nối đã thảo luận trong Chapter 13. Sự khác biệt duy nhất là UDP cung cấp một checksum tùy chọn để phát hiện packet hỏng tại phía bên nhận. Nếu checksum được thêm vào packet, UDP bên nhận có thể kiểm tra packet và bỏ nếu packet hỏng. Tuy nhiên, không có phản hồi nào gửi lại cho bên gửi.

