MỤC LỤC

[MỞ ĐẦU 2](#_Toc285716783)

[1. Giao thức TCP và Mạng vệ tinh 3](#_Toc285716784)

[1.1. Cơ bản về giao thức TCP 3](#_Toc285716785)

[1.1.1. Khái niệm 3](#_Toc285716786)

[1.1.2. Cấu trúc gói tin TCP 3](#_Toc285716787)

[1.1.3. Kết nối TCP 5](#_Toc285716788)

[1.1.4. Mô hình hoạt động của giao thức TCP 6](#_Toc285716789)

[1.2. Cơ bản về hệ thống mạng vệ tinh 9](#_Toc285716790)

[1.2.1. Đặc tính của mạng vệ tính 9](#_Toc285716791)

[1.2.2. Kiến trúc hệ thống vệ tinh 10](#_Toc285716792)

[1.2.3. Kết nối Internet qua vệ tinh 11](#_Toc285716793)

[2. Giới thiệu phần mềm mô phỏng - giả lập mạng NS-2 13](#_Toc285716794)

[2.1. Tổng quan về NS-2 13](#_Toc285716795)

[2.1.1. Khái niệm 13](#_Toc285716796)

[2.1.2. Kiến trúc NS-2 13](#_Toc285716797)

[2.2. Các thành phần cơ bản trong NS-2 15](#_Toc285716798)

[2.3. Cài đặt và thiết lập NS-2 19](#_Toc285716799)

[2.3.1. Cài đặt NS-2 trên nền LINUX 19](#_Toc285716800)

[2.3.2. Mô phỏng mạng đơn giản bằng NS-2 22](#_Toc285716801)

[KẾT LUẬN 24](#_Toc285716802)

[TÀI LIỆU THAM KHẢO 25](#_Toc285716803)

# MỞ ĐẦU

Ngày nay, công nghệ tiên tiến cho phép con người có thể vươn xa ra ngoài Trái đất, xây dựng những hệ thống vệ tinh ngoài vũ trụ để phục vụ cho các yêu cầu của cuộc sống hiện đại. Trong đó, hệ thống thông tin vệ tinh đã và đang đáp ứng được phần nào những yêu cầu đó. Thông qua nó, con người ở mọi nơi trên thế giới có kết nối, trao đổi thông tin với nhau trên một tốc độ đường truyền tốt mà không gặp những vấn đề về địa lý, địa hình như những hệ thống thông tin khác. Tuy nhiên, hệ thống thông tin vệ tinh còn có những vấn đề như độ trễ hay nhiễu do khoảng không gian tạo ra, điều này làm giảm hiệu suất mạng, gây khó khăn cho người sử dụng. Việc tối ưu hóa ứng dụng TCP trên kênh kết nối vệ tinh, sử dụng các lựa chọn tiêu chuẩn có thể làm tăng hiệu suất truyền dữ liệu và giảm tắc nghẽn; qua đó có thể khắc phục phần nào những vấn đề của kênh thông tin vệ tinh.

Dưới đây là phần tìm hiểu của em về giao thức TCP, hệ thống thông tin vệ tinh và phần mềm mô phỏng NS-2. Em xin cảm ơn GS.TS Nguyễn Thúc Hải đã hướng dẫn em hoàn thiện báo cáo này.

Sinh viên thực hiện

Chu Quang Tú

# Giao thức TCP và Mạng vệ tinh

## Cơ bản về giao thức TCP

### Khái niệm

Giao thức TCP (*Transmission Control Protocol* - Giao thức điều khiển truyền vận) là một trong các giao thức cốt lõi của bộ giao thức TCP/IP. Bằng việc sử dụng sử dụng TCP, các ứng dụng trên các máy chủ được nối mạng có thể tạo các kết nối với nhau, mà qua đó chúng có thể trao đổi dữ liệu hoặc các gói tin. Giao thức này đảm bảo chuyển giao dữ liệu tới nơi nhận một cách đáng tin cậy và đúng thứ tự. TCP còn phân biệt giữa dữ liệu của nhiều ứng dụng (chẳng hạn, dịch vụ Web và dịch vụ thư điện tử) đồng thời chạy trên cùng một máy chủ.

### Cấu trúc gói tin TCP

Cấu trúc gói tin TCP bao gồm hai phần là phần tiêu đề (header) và phần dữ liệu. Minh họa như hình dưới đây :

|  |
| --- |
|  |
| Cấu trúc gói tin TCP |

Trong đó ý nghĩa các trường số liệu là :

* *Source Port và Destination Port :* Số hiệu của cổng TCP tại máy gửi và máy nhận
* *Sequence Number :* Số tuần tự phát, có chức năng định danh byte đầu tiên của phần số liệu TCP trong dòng số liệu từ thực thể TCP gửi đến thực thể TCP nhận. Trường này có 2 nhiệm vụ. Nếu cờ SYN bật thì nó là số thứ tự gói ban đầu và byte đầu tiên được gửi có số thứ tự này cộng thêm 1. Nếu không có cờ SYN thì đây là số thứ tự của byte đầu tiên.
* *Acknowledgement Number :*  Vị trí tương đối của byte cuối cùng đã nhận đúng bởi thực thể gửi gói ACK cộng thêm 1. Giá trị trường này còn được gọi là số tuần tự thu. Giá trị này đúng khi bit cờ ACK bằng 1.
* *Data Offset :* Khoảng cách tương đối của trường số liệu với phần tiêu đề của TCP (TCP Header) tính theo từ 32 bit.
* *Reserved :* Luôn đặt là 0, sử dụng trong tương lai.
* *Control Bits* : Bao gồm 6 cờ:
  + URG : Cờ cho trường Urgent pointer
  + ACK : Cờ cho trường Acknowledgement
  + PSH : Hàm Push
  + RST : Thiết lập lại đường truyền
  + SYN : Đồng bộ lại số thứ tự
  + FIN : Không gửi thêm số liệu
* *Window :* Quy định tổng số byte dữ liệu mà thực thể nhận có thể nhận được, đồng nghĩa với độ lớn của bộ đệm nhận, tính khởi đầu từ giá trị trường số tuần tự thu (Acknowledgement Number).
* *Checksum :*  16 bit kiểm tra cho cả phần header và dữ liệu.
* *Urgent Pointer :* Vị trí tương đối của byte trong trường số liệu TCP cần được xử lý đầu tiên. Nếu cờ URG bật thì giá trị trường này chính là số từ 16 bít mà số tuần tự phát (*sequence number*) cần dịch trái.
* *Option :* Trường tùy chọn. Nếu có, thì độ dài là bội số của 32 bit
* *Padding :* Phần vá thêm để phần tiêu đề của gói TCP có độ lớn là bội của 4 byte

### Kết nối TCP

TCP là giao thức trao đổi số liệu, đảm bảo độ tin cậy và chính xác giữa hai thực thể cuối cùng trong mạng. Việc thiết kế và thực hiện giao thức TCP là phức tạp bởi vì :

* TCP phải quản lý đúng số tuần tự tính theo đơn vị byte của dòng dữ liệu.
* TCP phải tối ưu hóa khả năng dải thông của mạng bằng cách giám sát và điều khiển lưu lượng số liệu từ thực thể gửi đến thực thể nhận.

TCP bảo đảm số liệu tin cậy và chính xác giữa các thực thể cuối cùng trong mạng nhờ các yếu tố sau :

* *Đối thoại khi thu phát :* Mỗi khi trao đổi dữ liệu, bên nhận phải thông báo nhận đúng sau một khoảng thời gian nhất định. Nếu không, gói số liệu được bên gửi coi là nhận sai và sẽ phát lại.
* *Kiểm tra số liệu thu, phát :* Số liệu gửi đi được kiểm tra bằng thuật toán quy định. Byte kiếm tra (checksum) được gửi đi cùng với số liệu phát và được so sánh với byte checksum bên nhận. Nếu sai lệch, có nghĩa là có lỗi xảy ra trong quá trình truyền, bên sai thông báo kết quả nhận cho bên gửi và yêu cầu gửi lại.
* *Kiểm tra số tuần tự :* Do các gói TCP được truyền đi bằng các gói IP và các gói IP có thể đến đích không theo thứ tự phát ( do IP là giao thức không hướng kết nối) nên thực thể TCP nhận phải lập lại trật tự các gói số liệu thu được và hủy bỏ các gói số liệu trùng lặp khi cần và chuyển các gói số liệu đó theo đúng trật tự phát cho các ứng dụng.
* *Điều khiển lưu lượng :*  Mỗi thực thể của kết nối TCP đều có một vùng đệm hạn chế. Thực thể TCP nhận chỉ cho phép thực thể TCP phát gửi lượng số liệu đủ với vùng đệm nhận của mình. Điều này cho phép ngăn thực thể TCP gửi nhanh làm tràn vùng đệm của thực thể TCP nhận chậm.

### Mô hình hoạt động của giao thức TCP

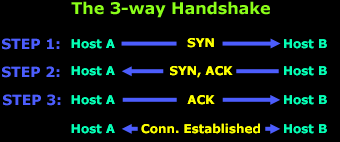
TCP đòi hỏi thiết lập kết nối trước khi bắt đầu gửi dữ liệu và kết thúc kết nối khi việc gửi dữ liệu hoàn tất. Cụ thể, các kết nối TCP có ba pha: thiết lập kết nối, truyền dữ liệu và kết thúc kết nối.

***Thiết lập kết nối***

Để thiết lập một kết nối, TCP sử dụng một quy trình bắt tay 3 bước (*3-way handshake*). Trước khi client thử kết nối với một server, server phải đăng ký một cổng và mở cổng đó cho các kết nối, đây được gọi là mở bị động. Một khi mở bị động đã được thiết lập thì một client có thể bắt đầu mở chủ động. Để thiết lập một kết nối, quy trình bắt tay 3 bước xảy ra như sau:

* Client yêu cầu mở cổng dịch vụ bằng cách gửi gói tin *SYN* (gói tin **TCP)** tới server, trong gói tin này, tham số *sequence number* được gán cho một giá trị ngẫu nhiên X.
* Server hồi đáp bằng cách gửi lại phía client bản tin *SYN-ACK*, trong gói tin này, tham số *acknowledgment number* được gán giá trị bằng X + 1, tham số *sequence number* được gán ngẫu nhiên một giá trị Y
* Để hoàn tất quá trình bắt tay ba bước, client tiếp tục gửi tới server bản tin ACK, trong bản tin này, tham số *sequence number* được gán cho giá trị bằng X + 1 còn tham số *acknowledgment number* được gán giá trị bằng Y + 1

Tại thời điểm này, cả client và server đều được xác nhận rằng, một kết nối đã được thiết lập. Minh họa của quá trình bắt tay 3 bước :



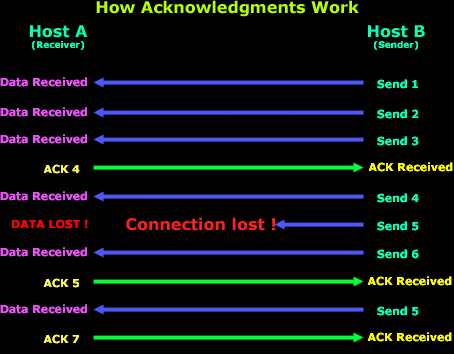
***Truyền dữ liệu***

Một số đặc điểm cơ bản truyền dữ liệu của giao thức TCP :

* Truyền dữ liệu không lỗi (do có cơ chế sửa lỗi và truyền lại)
* Truyền các gói dữ liệu theo đúng thứ tự.
* Truyền lại các gói dữ liệu mất trên đường truyền.
* Loại bỏ các gói dữ liệu trùng lặp.
* Cơ chế hạn chế tắc nghẽn đường truyền.

Ở hai bước đầu tiên trong ba bước bắt tay, hai máy tính trao đổi một số thứ tự gói ban đầu (*Initial Sequence Number* - ISN). Số này có thể chọn một cách ngẫu nhiên. Số thứ tự này được dùng để đánh dấu các khối dữ liệu gửi từ mỗi máy tính. Sau mỗi byte được truyền đi, số này lại được tăng lên. Nhờ vậy ta có thể sắp xếp lại chúng khi tới máy tính kia bất kể các gói tới nơi theo thứ tự thế nào.

Trên lý thuyết, mỗi byte gửi đi đều có một số thứ tự và khi nhận được thì máy tính nhận gửi lại tin báo nhận (ACK). Trong thực tế thì chỉ có byte dữ liệu đầu tiên được gán số thứ tự trong trường số thứ tự của gói tin và bên nhận sẽ gửi tin báo nhận bằng cách gửi số thứ tự của byte đang chờ.



Số thứ tự và tin báo nhận giải quyết được các vấn đề về lặp gói tin, truyền lại những gói bị hỏng hoặc mất và các gói tin đến sai thứ tự. Để phục vụ mục đích kiểm tra, các gói tin có trường giá trị tổng kiểm (*checksum).* Tuy nhiên, hiện tại kỹ thuật kiểm tra tổng trong TCP không đủ mạnh. Các tầng liên kết dữ liệu với xác suất lỗi bit cao có thể cần được bổ sung các khả năng phát hiện lỗi tốt hơn.

Ngoài ra, để chống tắc nghẽn. tin báo nhận là tín hiệu về tình trạng đường truyền giữa 2 máy tính. Từ đó, hai bên có thể thay đổi tốc độ truyền nhận dữ liệu phù hợp với điều kiện. Vấn đề này thường được đề cập là điều khiển lưu lượng, kiểm soát tắc nghẽn. TCP sử dụng một số cơ chế nhằm đạt được hiệu suất cao và ngăn ngừa khả năng nghẽn mạng. Các cơ chế này bao gồm: cửa sổ trượt (*sliding window*), thuật toán *slow-start*, thuật toán tránh nghẽn mạng (*congestion avoidance*), thuật toán truyền lại và phục hồi nhanh ...

***Kết thúc kết nối***

Để kết thúc kết nối hai bên sử dụng quá trình bắt tay 4 bước và chiều của kết nối kết thúc độc lập với nhau. Khi một bên muốn kết thúc, nó gửi đi một gói tin FIN và bên kia gửi lại tin báo nhận ACK. Vì vậy, một quá trình kết thúc tiêu biểu sẽ có 2 cặp gói tin trao đổi.

Một kết nối có thể tồn tại ở dạng "nửa mở": một bên đã kết thúc gửi dữ liệu nên chỉ nhận thông tin, bên kia vẫn tiếp tục gửi.

## Cơ bản về hệ thống mạng vệ tinh

Hệ thống thông tin vệ tinh là sự kết hợp của hai thành phần chính : Vệ tinh và trạm mặt đất.

* *Vệ tinh :* Được phóng ra ngoài Trái đất. Vệ tinh có hệ thống chuyển tiếp thông tin nên có thể thu nhận tín hiệu từ trạm mặt đất. Mặt khác, chúng còn được sử dụng để phát xuống trạm nhận tín hiệu thông qua hệ thống khuếch đại tín hiệu mạnh.
* *Trạm mặt đất :* Được đặt trên mặt đất, bao gồm hai chức năng chính : Truyền thông lên vệ tinh và nhận thông tin từ vệ tinh.

Truyền thông vệ tinh được sử dụng rộng rãi trong nhiều lĩnh vực như mạng điện thoại truyền thống, truyền tín hiệu truyền hình, truyền thông trong hàng hải ...

### Đặc tính của mạng vệ tính

Ưu điểm của mạng thông tin vệ tinh là dung lượng thông tin lớn, do sử dụng băng tần công tác rộng và kỹ thuật đa truy nhập cho phép đạt thông lượng lớn trong thời gian ngắn mà ít loại hình thông tin khác có thể đạt được. Mặt khác, do liên lạc giữa vệ tinh và trạm mặt đất là trực tiếp. nên độ tin cậy và chất lượng thông tin của mạng vệ tinh cao hơn các loại hình thông tin khác. Ngoài ra, mạng vệ tinh còn cung cấp vùng bao phủ rộng, cho phép người sử dụng có khả năng ứng dụng thông tin di động và thông tin liên lạc toàn cầu.

Tuy nhiên, ngoài những ưu điểm kể trên, hệ thống thông tin vệ tinh vẫn còn một thách thức lớn với hiệu suất của các ứng dụng Internet bởi những nhược điểm sau :

* *Độ trễ :* Trong hệ thông thông tin vệ tinh địa tĩnh GEO, độ trễ tối thiểu là 250ms; và đôi khi việc đóng gói số liệu, hàng chờ số liệu, hệ thống chuyển mạch bên trong có thể tạo thêm độ trễ kết nối giữa các trạm đầu cuối làm cho độ trễ lên đến 400ms. Ước lượng độ trễ này cao hơn khoảng 25 lần độ trễ kết nối mạng trục Internet Bắc-Nam Việt Nam (khoảng 16ms - Số liệu từ trung tâm VDC1). Độ trễ này làm ảnh hưởng đến các ứng dụng có tính tương tác cần sự bắt tay làm việc giữa hai phía - ví dụ như giao thức TCP.
* *Nhiễu :* Cường độ tín hiệu sóng radio tương ứng với khoảng cách truyền đi. Khoảng cách không gian giữa trạm mặt đất và vệ tinh là rất lớn, vì thế nhiễu sinh ra, làm giảm cường độ tín hiệu thông tin.

Ngoài những nhược điểm trên, hệ thống thông tin còn đòi hỏi đầu tư ban đầu cao trong khi thời gian làm việc vệ tinh là tương đối ngắn ( từ 7 - 15 năm). Thêm vào đó, chi phí bảo dưỡng hệ thống rất tốn kém và vấp phải nhiều khó khăn về mặt kỹ thuật.

### Kiến trúc hệ thống vệ tinh

Hệ thống vệ tinh có nhiều kiến trúc, tuy nhiên có một vài kiến trúc mạng mà trong đố liên kết vệ tinh có thể tích hợp vào mạng Internet.

* *Mạng vệ tinh phi đối xứng :* Một vài mạng vệ tinh cung cấp dải thông phi đối xứng. Đó là : dữ liệu truyền theo một chiều lớn hơn dữ liệu truyền theo chiều ngược lại bởi giới hạn của năng lực truyền thông và kích cỡ anten tại mỗi đầu cuối của kết nối. Bản chất của hầu hết lưu lượng TCP là phi đối xứng với luồng chuyển dữ liệu theo một hướng và trả lời theo hướng ngược lại.
* *Mạng vệ tinh lai ghép:*  Trong nhiều trường hợp tổng quát, những liên kết vệ tinh có thể được đặt tại bất kỳ điểm nào trong kiến trúc liên kết mạng. Trong trường hợp này, liên kết vệ tinh chỉ đóng vai trò liên kết giữa hai cổng kết nối khác.
* *Mạng vệ tinh điểm-điểm :* Trong mạng vệ tinh điểm-điểm, chỉ có duy nhất chặng kết nối trong mạng và bằng liên kết vệ tinh. Đó là môi trường vệ tinh thuần nhất, đặc trưng nhất về các vấn đề kết nối vệ tinh.

### Kết nối Internet qua vệ tinh

Ngày nay, giao thức TCP được sử dụng rộng rãi và chiếm phần lớn trong các ứng dụng Internet. Hiệu suất của giao thức TCP thông qua môi trường mạng có độ trễ lớn sẽ ảnh hưởng trực tiếp đến truy nhập Internet sử dụng vệ tinh địa tĩnh GEO. Giao thức TCP làm việc tốt với liên kết vệ tinh bằng cách hoạt động ở tốc độ thấp; Vì thế làm mất đi đặc tính của hiệu suất làm việc với kênh truyền số liệu tốc độ cao mà đáng lẽ có được bởi khả năng của thông tin vệ tinh. Ta có thể cái thiện hiệu suất bằng việc sử dụng các cơ chế điều khiển của TCP một cách phù hợp như :

* *Kích thước cửa sổ :* Điều khiển lưu lượng của TCP được bắt đầu với khái niệm "kích thước cửa sổ" trong kết nối TCP. Nó xác định có thể có bao nhiêu dữ liệu có thể đang lưu chuyển mà chưa xử lý. Trong mạng có độ trễ lớn, có thể nhiều phân đoạn chưa có trả lời. Trên lý thuyết, dung lương trên thời gian trễ của kênh truyền (bandwidth delay) cho ta độ lớn của dữ liệu có thể truyền mà không cần trả lời. Nhưng thực tế, bộ nhớ và tài nguyên của các hệ thống làm giới hạn kích thước cửa sổ. Với chuẩn TCP, kích thước cửa sổ lớn nhất là 64KB. Trong quá trình triển khai thực tế, kích thước giới hạn của cửa sổ thường là 32KB.

Để có thể sử dụng tối đa giải thông của mạng vệ tinh, giao thức TCP cần kích thước cửa sổ lớn hơn rất nhiều. Ví dụ : Trên liên kết vệ tinh, với tổng thời gian trễ toàn phần là 0.8 giây, dải thông của kết nối vệ tinh là 1.54Mbps, theo lý thuyết, kích thước cửa sổ tối ưu là 154KB - lớn hơn rất nhiều so với kích thước tối đa 32KB hay 64KB.

* *Sự thích nghi dải thông :* Giao thức TCP có thể thích ứng với dải thông có thể của mạng bằng việc tăng kích thước cửa sổ, tránh tắc nghẽn và giảm kích thước cửa sổ cho phù hợp. Tốc độ thích nghi tương xứng với độ trễ. Trong mạng vệ tinh với thời gian trễ lớn, thích nghi dải thông cũng mất nhiều thời gian hơn, và vì thế điều khiển chống tắc nghẽn gần như không hiệu quả.
* *Phát lại có lựa chọn :* Thứ tự các thông tin phản hồi của TCP chuẩn có tính tích lũy. Tức là nếu một phân đoạn bị mất, TCP gửi sẽ truyền lại tất cả dữ liệu trên phân đoạn bị mất mà không quan tâm đến các phân đoạn đã truyền thành công sau đó. Giao thức TCP coi việc mất một phân đoạn là thể hiện có xung đột và giảm kích thước cửa sổ đi một nửa.
* *Slow-start :* Khi kết nối TCP bắt đầu lần đầu tiên, hoặc rỗi không truyền số liệu đã lâu, nó cần nhanh chóng xác định khả năng dải thông của mạng. TCP thực hiện việc này bằng cách khởi tạo kích thước cửa sổ của một phân đoạn (thông thường là 512 Byte), sau đó tăng kích thước cửa sổ mỗi khi gói tin được gửi đi thành công và thông tin phản hồi trở lại, cho đến khi phát hiện trạng thái bão hòa của mạng (thể hiện bởi việc gói tin truyền đi bị hủy bỏ). Một mặt , *slow-start* tránh tắc nghẽn mạng trước khi nó xác định được khả năng dải thông làm việc tốt của mạng. Mặt khác, tận dụng dải thông là chức năng tối ưu mà *slow-start* thực hiện.

# Giới thiệu phần mềm mô phỏng - giả lập mạng NS-2

## Tổng quan về NS-2

### Khái niệm

NS-2 là một bộ mô phỏng có cấu trúc hướng đối tượng. Nó được xây dượng dựa trên hai ngôn ngữ hướng đổi tượng là C++ và phiên bản hướng đối tượng của Tcl ( hay còn gọi là OTcl). NS có thể được mở rộng bởi người sử dụng ( tức là người dùng có thể lập trình tùy ý trên nền của bộ mô phỏng NS ). Thông qua cách tiếp cận này, ta có thể coi mỗi một mô hình mô phỏng như một chương trình hơn là các mô hình tĩnh - không thể thay đổi. Mục đích của NS-2 là tạo ra một môi trường giả lập cho việc nghiên cứu, kiểm tra, thiết kế các giao thức, các kiến trúc mới, so sánh các giao thức và tạo ra các mô hình mạng phức tạp.

Phiên bản thứ nhất của NS được phát triển vào năm 1995 và phiên bản thứ hai ra đời năm 1996. NS-2 là phần mền mã nguồn mở có thể chạy được trong môi trường Linux và Window.

### Kiến trúc NS-2

Hình dưới đây mô tả kiến trúc chung của bộ mô phỏng NS-2. Người sử dụng thiết kế và chạy các mô phỏng trong Tcl. Còn Tcl dùng các đối tượng mô phỏng trong OTcl. Các đối tượng bộ lập lịch sự kiện (Event Scheduler) và hầu hết các đối tượng thành phần mạng (Network Components) được thực thi bằng C++ và sẵn có cho OTcl qua một liên kết OTcl. Liên kết OTcl này được thực thi bằng TclCL. Tất cả đã làm nên NS, bộ biên dịch Tcl mở rộng hướng đối tượng và các thư viện mô phỏng mạng.

|  |
| --- |
|  |
| Kiên trúc chung của NS-2 |

NS là bộ mô phỏng mạng, vì thế nó phải đáp ứng các yêu cầu thiết thực của việc mô phỏng. Một mặt, mô phỏng chi tiết các giao thức - đây là công việc yêu cầu một ngôn ngữ lập trình hệ thống có thể thao tác một cách hiệu quả đến các byte, các header, và các thuật toán triển khai có thể chạy trên một tập dữ liệu lớn. Với nhiệm vụ này thì tốc độ chạy chương trình và thời gian xoay vòng là ít quan trọng hơn. Mặt khác, phần lớn các nghiên cứu về mạng bao gồm các tham số hoặc các cấu hình ít thay đổi, hay nghiễn cứu nhanh chóng một số kịch bản. Với các trường hợp đó, thời gian lặp lại (thay đổi mẫu và chạy lại) là quan trọng hơn.

Trong khi đó, NS có một thư viện đồ sộ các đối tượng giao thức và mạng. Có hai nhánh các lớp trong NS : Nhánh gồm mã C++ đã được biên dịch và nhánh gồm mã OTcl đã được thông dịch, giữa chúng có sự tương ứng 1-1.

Nhánh gồm mã ngôn ngữ C++ đã được biên dịch cho phép ta đạt hiệu quả mô phỏng với thời gian thực hiện nhanh hơn. Điều này đặc biệt hữu ích cho việc định nghĩa và mô phỏng hoạt động một cách thật chi tiết của các giao thức cụ thể, cho phép ta giảm thời gian xử lý sự kiện và gói tin.

Sau đó, ở đoạn mã OTcl được nhập bởi người dùng, ta có thể định nghĩa một cấu trúc mạng cụ thể, các giao thức và ứng dụng riêng biệt mà ta muốn mô phỏng ( đặc tính của chúng được định nghĩa ở bên nhánh C++ đã được biên dịch) và dạng của dữ liệu đầu ra muốn nhận được tự bộ mô phỏng. OTcl có thể tận dụng các đối tượng đã được biên dịch trong C++ thông qua một liên kết OTcl - được thực hiện bằng việc sử dụng Tclcl, liên kết OTcl này tạo ra một ánh xạ giữa đối tượng OTcl với mỗi đối tượng trong C++.

Bằng cách sử dụng C++/OTcl, bộ mô phỏng mạng NS-2 hoàn toàn là hướng đối tượng. Đồng thời, thông qua Tclcl, NS cung cấp sự kết dính để tạo nên các đối tượng và các biến trong cả hai ngôn ngữ.

## Các thành phần cơ bản trong NS-2

Bộ mô phỏng NS-2 ngoài phần mô phỏng NS, còn có các công cụ thành phần bổ trợ cho việc mô phỏng như công cụ trình diễn mô phỏng bằng hình ảnh NAM,

***Bộ mô phỏng NS***

Thực hiện chức năng mô phỏng, bằng hai ngôn ngữ C++ và OTcl.

Bộ mô phỏng NS thực thi những tính năng sau:

* Các kỹ thuật quản lý hàng đợi Router như DropTail, RED, CBQ,
* Multicasting
* Mô phỏng mạng không dây
  + Được phát triển bởi Sun Microsystems + UC Berkeley (Dự án Daedalus)
  + Thuộc mặt đất (di động, adhoc, GPRS, WLAN, BLUETOOTH), vệ tinh
  + Chuẩn IEEE 802.11 có thể được mô phỏng, các giao thức Mobile-IP và Adhoc như DSR, TORA, DSDV và AODV
* Hành vi nguồn traffic – www, CBR, VBR
* Các agent truyền tải – UDP, TCP
* Định tuyến
* Luồng packet
* Mô hình mạng
* Các ứng dụng – Telnet, FTP, Ping
* Các packet tracing trên tất cả các link và trên các link xác định

***Công cụ hiển thị hình ảnh NAM***

Khi mô phỏng kết thúc, nếu các script OTcl đầu vào yêu cầu thì NS sẽ tạo ra các tệp văn bản lưu chi tiết dữ liệu mô phỏng. Dữ liệu đó có thể được dùng cho việc phân tích hay được dùng như đầu vào cho các công cụ trình diễn mô phỏng đồ họa NAM.

NAM thực hiện lại những sự kiện từ tệp dò vết nam (nam tracefile). Tệp này có thể là khổng lồ khi thời gian mô phỏng lớn và sự kiện xảy ra nhiều. Vì thế, phải hết sức cẩn thận. Ngoài ra, cần chú ý rằng các thông tin đồ họa lại không được dùng để phân tích mô phỏng một cách chính xác.

Có thể tóm tắt các tính năng của NAM như sau:

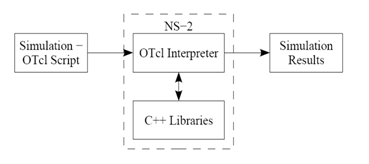
* Cung cấp trình diễn ảo cho mạng đã được tạo
* Có thể thi hành trực tiếp từ kịch bản OTcl
* NAM có giao diện đồ hoạ với các điều khiển bao gồm play (chạy), stop (ngưng), fast forward (chạy tiếp nhanh), rw (lùi lại), pause (tạm ngưng), điều khiển tốc độ trình diễn và tính năng giám sát packet
* Biểu diễn thông tin như throughput (thông lượng), số packet trên từng link ...
* Cung cấp giao diện kéo và thả cho việc tạo ra các mô hình mạng (topology)

Dưới đây là một minh họa về cửa sổ làm việc của NAM

|  |
| --- |
|  |
| Công cụ trình diễn mô phỏng bằng hình ảnh NAM |

***Phân tích***

Ở khía cánh người dùng, NS-2 được xem là một trình thông dịch OTcl, sử dụng kịch bản OTcl làm đầu vào và đưa ra tệp dò vết là kết quả mô phỏng.

******

Người sử dụng NS-2 sẽ dùng các tệp dò vết đó (kết quả mô phỏng) , tiến hành phần tích các dữ liệu thu được để đưa ra kết luận về kịch bản đã sử dụng mô phỏng. Việc phân tích này được hỗ trợ bằng các phần mềm như *XGRAPH* và *TRACEGRAPH*.

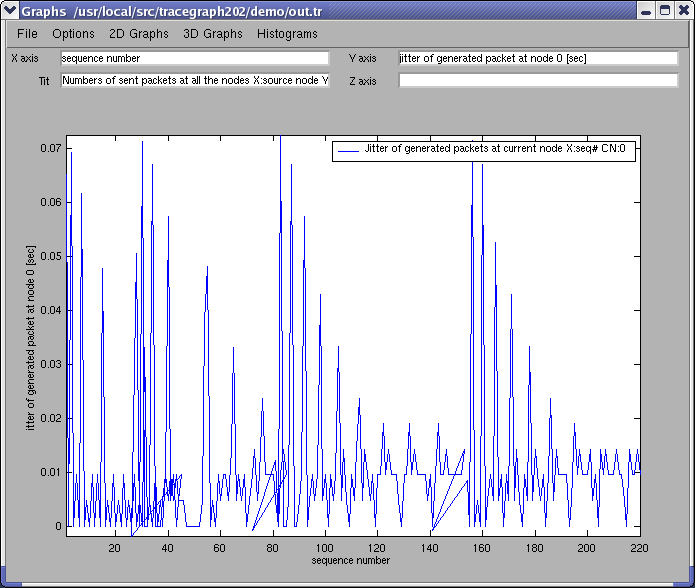
TRACEGRAPH là bộ phân tích tệp dò vết trace. *TRACEGRAPH* chạy trong hệ điều hành Windows, Linux, Unix và yêu cầu hệ thống có cài đặt Matlab 6.0 (hoặc các phiên bản cao hơn).

*TRACEGRAPH* hỗ trợ các định dạng file trace như sau:

* Wired (có dây)
* Satellite (vệ tinh)
* Wireless (không dây)

*TRACEGRAPH* phiên bản 2.02 có các tính năng sau:

* 238 đồ thị 2D (hai chiều)
* 12 đồ thị 3D (ba chiều)
* Các đồ thị và các thống kê về Delay (trì hoãn), jitter (độ rung pha), processing times (số lần xử lý), Round Trip Times (số lần khứ hồi), intermediate nodes (số node trung gian) và throughput (thông lượng)
* Các đồ thị và các thống kê cho toàn network (mạng), link (liên kết) và node
* Tất cả các kết quả có thể được lưu vào file dạng text (dạng văn bản), các đồ thị có thể lưu dưới dạng file jpeg hay tiff.
* Thông tin của trục x,y,z: minimum (giá trị nhỏ nhất), mean (giá trị trung bình), maximum (giá trị lớn nhất), standard deviation (độ lệch tiêu chuẩn) và median (số trung bình)
* Nếu các đồ thị lưu trong file text thì trong file này có thể dựa vào thông tin 2 hay 3 cột để vẽ lại đồ thị.
* Xử lý các file kịch bản để phân tích một cách tự động



***Giao diện TRACEGRAPH***

## Cài đặt và thiết lập NS-2

NS là phần mềm mã nguồn mở, có thể cài đặt và sử dụng trên các nền tảng Linux, Windows hay MAC OS. Trong đó, NS-2 thể hiện đầy đủ chức năng nhất khi chạy trên nền Linux. Dưới đây là phần cài đặt và thiết lập NS-2 (phiên bản 2.34) trên hệ điều hành Ubuntu 10.10.

### Cài đặt NS-2 trên nền LINUX

Có hai phương pháp cài đặt NS-2. Một là cài đặt gói Allinone (đã bao gồm tất cả các thành phần của bộ mô phỏng NS như NS, Tcl, OTcl, NAM ...). Cách thứ hai là cài đặt từng gói cần thiết tùy theo nhu cầu sử dụng. Dưới đây là phần trình bày cách cài đặt gói NS-Allinone-2.34.

Mở cửa sổ Terminal của Ubunutu và tải về gói cài đặt ns-allinone-2.34 :

*wget http://nchc.dl.sourceforge.net/sourceforge/nsnam/ns-allinone-2.34.tar.gz*

Sau đó, đưa gói cài đặt về thư mục Home (Ở đây là home/tucq/), tiến hành giải nén

*tar -xzvf ns-allinone-2.34.tar.gz*

Di chuyển về thư mục chứa bộ cài đặt :

*cd ns-allinone-2.34*

Sau đó, tiến hành tải về các gói phần mềm hỗ trợ cài đặt bộ mô phỏng :

*sudo apt-get install -f build-essential libxt-dev libxt6 libsm-dev libsm6 libice-dev libice6 libxmu-dev*

Khi đã cài đặt thành công các phần mềm hỗ trợ, thực hiện lệnh sau để cài đặt :

*./install*

Khi việc cài đặt hoàn tất, ta cần đặt đường dẫn cho các gói phần mềm. Chỉnh sửa trong file /.bashrc :

*gedit ~/.bashrc*

Thêm vào cuối file bashrc đường dẫn như sau :

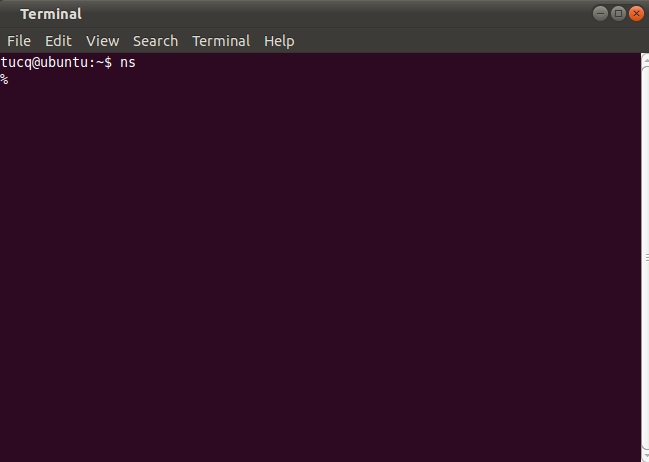
|  |
| --- |
| # LD\_LIBRARY\_PATH |
| OTCL\_LIB=/home/tucq/ns-allinone-2.34/otcl-1.13 |
| NS2\_LIB= /home/tucq/ns-allinone-2.34/lib |
| X11\_LIB=/usr/X11R6/lib |
| USR\_LOCAL\_LIB=/usr/local/lib |
| export LD\_LIBRARY\_PATH=$LD\_LIBRARY\_PATH:$OTCL\_LIB:$NS2\_LIB:$X11\_LIB:$USR\_LOCAL\_LIB |
|  |
| # TCL\_LIBRARY |
| TCL\_LIB= /home/tucq/ns-allinone-2.34/tcl8.4.18/library |
| USR\_LIB=/usr/lib |
| export TCL\_LIBRARY=$TCL\_LIB:$USR\_LIB |
|  |
| # PATH |
| XGRAPH=/home/tucq/ns-allinone-2.34/bin:/home/tucq/ns-allinone-2.34/tcl8.4.18/unix:/home/tucq/ns-allinone-2.34/tk8.4.18/unix |
| NS=/home/tucq/ns-allinone-2.34/ns-2.34/ns |
| NAM=/home/tucq/ns-allinone-2.34/nam-1.13/nam |
| PATH=$PATH:$XGRAPH:$NS:$NAM |

Cuối cùng, di chuyển đến thư mục ns-2.34 trong thư mục cài đặt ns-allinone-2.34 để tiến hành xác thực việc cài đặt đã hoàn tất chưa.

*cd ~/ns-allinone-2.33/ns-2.33*

*./validate*

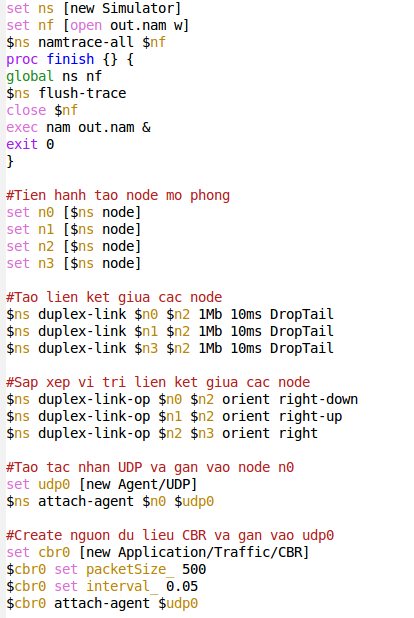
Gõ từ khóa "ns" vào Terminal, nếu hiển thị như sau nghĩa là việc cài đặt đã thành công :

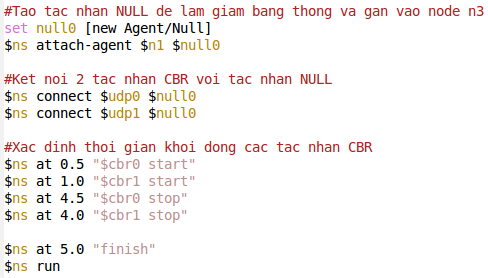


### Mô phỏng mạng đơn giản bằng NS-2

Dưới đây là phần mô phỏng mô hình mạng đơn giản bằng NS-2. Do kiến thức còn non kém và thời gian tìm hiểu chưa nhiều, nên phần mô phòng còn đơn giản và sơ sài.

Phần mã OTcl được thực hiện trên GNU Emac23 với bộ mô phỏng NS-2 v2.34 trên nền hệ điều hành Ubuntu 10.10





|  |
| --- |
|  |
| Mô phỏng hoạt động trên NAM |

# KẾT LUẬN

Trên đây là những tìm hiểu của em về mạng thông tin vệ tinh, giao thức TCP và phần mềm mô phỏng NS-2. Đây là những kiến thức em tìm hiểu và thu thập được trong thời gian thực tập, là những kiến thức cơ bản bổ trợ cho em trong việc thực hiện đồ án tốt nghiệp với đề tài "Hiệu suất TCP trên kênh thông tin vệ tinh". Do kiến thức còn hạn chế và thời gian tìm hiểu không nhiều nên phần tìm hiểu của em còn đơn giản , sơ sài . Kính mong thầy cô thông cảm.

# TÀI LIỆU THAM KHẢO

[1] RFC 793 - Transmission Control Protocol

[2] Andreas Schilke - TCP over Satellite Links (Seminar ``Broadband Networking Technology'' ) - 1997

[3] Mahbub Hassan, Raj Jain - High Performance TCP/IP Networking - 2004

[4] Network Simulator (NS-2), http://www.isi.edu/nsnam/ns/

[5] http://en.wikipedia.org/wiki/Transmission\_Control\_Protocol