ĐẠI HỌC QUỐC GIA HÀ NỘI

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ

Nguyễn Văn Biền

NGHIÊN CỨU TÍNH TOÁN LƯỚI VÀ ÁP DỤNG GIẢI BÀI TOÁN TRONG AN TOÀN THÔNG TIN

KHOÁ LUẬN TỐT NGHIỆP ĐẠI HỌC HỆ CHÍNH QUY

Ngành: Công Nghệ Thông Tin

HÀ NỘI - 2010

ĐẠI HỌC QUỐC GIA HÀ NỘI

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ

Nguyễn Văn Biền

NGHIÊN CỨU TÍNH TOÁN LƯỚI VÀ ÁP DỤNG GIẢI BÀI TOÁN TRONG AN TOÀN THÔNG TIN

KHOÁ LUẬN TỐT NGHIỆP ĐẠI HỌC HỆ CHÍNH QUY

Ngành: Công Nghệ Thông Tin

Cán bộ hướng dẫn: PGS-TS Trịnh Nhật Tiến

Cán bộ đồng hướng dẫn: ThS Lương Việt Nguyên

HÀ NỘI - 2010

Mục Lục

DANH MỤC CÁC TỪ VIẾT TẮT

API : Application Programming Interface

CSDL : Cơ Sở Dữ Liệu

CPU : Center Processing Unit

GASS : Grid Access to Secondary

GGF : Global Grid Forum

GIMPS : the Great Interner Mersenne Prime Search

GIS : Grid Security System

GRAM : Grid Resource Allocation Manager

GT : Globus Toolkit

IPG : Information Power Grid

J2EE : Java 2 Enterprise Edition

MDS : Monitoring and Discovery Service

OSI : Open Systems Interconnection

OGSA : Open Grid Service Architecture

OGSI : Open Grid Service Infrastructure

QoS : Query of Service

SDK : Software Development Kit

VO : Virtual Organization

WSAS : Web Sphere Application Server

DANH MỤC HÌNH VẼ

MỞ ĐẦU

Cho đến nay tính toán lưới là một lĩnh vực mới mẻ và hấp dẫn trong ngành công nghệ thông tin. Với khả năng tận dụng các nguồn tài nguyên nhàn rỗi môi trường tính toán lưới có thể đem lại cách giải quyết tối ưu cho những bài toán lớn cả về mặt kinh tế lẫn thời gian thực hiện mà hiện nay các hệ thống siêu máy tính cũng như các cluster vẫn còn gặp một số khó khăn khi giải quyết. Mặc dù tính toán lưới đã đạt được một số kết quả nhất định nhưng các viện nghiên cứu và nhiều người quan tâm đến lĩnh vực công nghệ thông tin vẫn tập trung nghiện cứu để hướng tới một hệ thống lưới hoàn chỉnh trên phạm vi toàn cầu.

Tại Việt Nam công nghệ này có thể nói là vẫn còn khá mới mẻ, nó chỉ được biết tới trong các đề tài nghiên cứu khoa học, trong các viện chuyên môn mà chưa được xem xét nghiên cứu kỹ tại các trường đại học. Hiện nay đang có một số trung tâm nghiên cứu và bắt đầu triển khai công nghệ này như: Trung tâm của trường Đại học Khoa học Tự Nhiên – Đại học Quốc Gia Hà Nội, Trung tâm tính toán hiệu năng cao của đại học Bách Khoa Hà Nội, trung tâm của trường đại học Khoa học Tự Nhiên – Đại học Quốc Gia TP. Hồ Chí Minh.

Dù đã cố gắng tập trung cho khóa luận, nhưng do thời gian có hạn, cùng sự hạn chế của bản thân nên khóa luận này tập trung trình bày những kiến thức cơ bản nhất về công nghệ tính toán lưới, đồng thời khóa luận cũng trình bày một ứng dụng của tính toán lưới trong việc giải quyết bài toán trong an toàn thông tin.

Em xin được gửi lời cảm ơn chân thành nhất tới PGS. TS Trịnh Nhật Tiến, cùng ThS Lương Việt Nguyên đã tạo điều kiện và hướng dẫn em nhiệt tình để hoàn thành bài khóa luận này. Cũng nhân đây con xin gửi lời cảm ơn đến gia đình và bạn bè đã luôn quan tâm và động viên trong suốt quá trình hoàn thành khóa luận.

Chương 1 . TỔNG QUAN VỀ TÍNH TOÁN LƯỚI

1.1. GIỚI THIỆU TÍNH TOÁN LƯỚI

1.1.1. Nguồn gốc tính toán lưới

Cũng giống như các công nghệ tính toán khác, tính toán lưới xuất phát từ nhu cầu tính toán của con người. Thực tiễn ngày càng đặt ra những bài toán phức tạp hơn và do vậy các tổ chức cũng cần phải có năng lực tính toán mạnh mẽ hơn. Các tổ chức giải quyết vấn đề này bằng hai cách:

- ✓ Đầu tư thêm trang thiết bị, cơ sở hạ tầng tính toán (mua thêm máy chủ, máy trạm, siêu máy tính, cluster…). Tuy nhiên cách làm này có một nhược điểm là tốn kém tiền của, số trang thiết bị sẽ tỉ lệ thuận với độ phức tạp của bài toán.
- ✓ Có một cách làm khác hiệu quả hơn đó là phân bố lại tài nguyên hợp lý trong tổ chức hoặc thuê thêm các nguồn tài nguyên từ bên ngoài (tất nhiên là việc thuê này sẽ có chi phí ít hơn nhiều so với việc đầu tư mới trang thiết bị).

Cách giải quyết thứ hai này chính là mục tiêu và là nguồn gốc yêu cầu cho sự hình thành của tính toán lưới. Các nhà khoa học tại Argone National Labs thuộc đại học Chicago (Mỹ) là những người đầu tiên đề xuất ý tưởng về tính toán lưới. Cũng như nhiều ý tưởng cách mạng khác trong tin học như World Wide Web, siêu máy tính … tính toán lưới được hình thành bởi nhu cầu thực tế là mong muốn đạt tới giới hạn của khả năng tính toán.

1.1.2. Khái niệm tính toán lưới

Hiện nay tồn tại khá nhiều định nghĩa khác nhau về tính toán lưới và vẫn chưa có được một định nghĩa nào được coi là chuẩn. Khóa luận trình bày định nghĩa về tính toán lưới của Ian Foster, đây là định nghĩa sớm và chuẩn nhất về tính toán lưới, định nghĩa này được ông đưa ra trong một bài báo được mang tên “What is Grid ?”.

"*Grid là một loại hệ thống tính toán song song, phân tán cho phép chia sẻ, lựa chọn, kết hợp các tài nguyên phân tán theo địa lý, thuộc nhiều tổ chức khác nhau dựa trên tính sẵn sàng, khả năng, chi phí của chúng và yêu cầu về chất lượng dịch vụ (QoS) của người dùng để giải quyết các bài toán, ứng dụng có quy mô lớn trong khoa học, kỹ thuật và thương mại. Từ đó hình thành nên các "tổ chức ảo" (Virtual Organization (VO)), các liên minh tạm thời giữa các tổ chức và tập đoàn, liên kết với nhau để chia sẻ tài nguyên và / hoặc kỹ năng nhằm đáp ứng tốt hơn các cơ hội kinh doanh hoặc các dự án có nhu cầu lớn về tính toán và dữ liệu, toàn bộ việc liên minh này dựa trên các mạng máy tính"*

Ta cũng có thể hiểu rằng: tính toán lưới là một cơ sở hạ tầng tin học cụ thể báo gồn vả phần cứng và phần mềm cho phép người sử dụng khai thác các tài nguyên trên các máy trạm hay máy chủ với tốc độ cao với độ tin cậy, giá thành chấp nhận được và hệ thống có xu hướng trong suốt với người dùng. Tính toán lưới chính là bước phát triển tiếp theo của tính toán phân tán. Mục đích là tạo ra một máy tính ảo với người sử dụng, nó có khả năng tính toán lớn, thậm chí trên cả một siêu máy tính.

Ý tưởng về tính toán lưới rất có ý nghĩa thực tế. Bởi lẽ, hiện nay theo các nghiên cứu thì các máy tính cá nhân thường chỉ sử dụng từ 5-10% năng lực tính toán còn các máy chủ, siêu máy tính cũng chỉ sử dụng đến 20% năng lực tính toán, đây là sự phí phạm một nguồn tài nguyên tính toán rất lớn. Việc tận dụng hiệu quả các nguồn tài nguyên này có thể mạng lại một sức mạnh tính toán khổng lồ. Tính toán lưới sẽ là một giải pháp hữu hiệu khi mà mục đích sử dụng của nó tập trung vào sử dụng tốt hơn và có hiệu quả hơn các nguồn tài nguyên nhằm chia sẻ các ứng dụng và tăng cường sự hợp tác trong các dự án. Thuật ngữ "lưới" ở đây xuất phát từ lưới điện (electricity grid), ngụ ý rằng bất cứ một thiết bị tương thích nào đều có thể gắn vào trong lưới và được xếp ở một mức tài nguyên nào đó mà không cần quan tâm đến nguồn gốc của tài nguyên đó. Trong tương lai, tính toán lưới có thể cung cấp cho người sử dụng các dịch vụ đóng vai trò như là dịch vụ cơ sở hạ tầng mà chúng ta có thể sử dụng hàng ngày như: điện, nước, giao thông, …

Các nghiên cứu về tính toán lưới đã và đang được tiến hành là nhằm tạo ra một cơ sở hạ tầng lưới, cho phép dễ dàng chia sẻ và quản lý các tài nguyên đa dạng và phân tán trong môi trường lưới. Các thách thức mà công nghệ lưới hướng tới giải quyết bao gồm:

- ✓ ***Sự đa dạng và không đồng nhất của các tài nguyên***

Tài nguyên ở đây được hiểu theo nghĩa tổng quát, đó có thể là các tài nguyên phần cứng: tài nguyên tính toán, tài nguyên lưu trữ, các thiết bị đặc biệt khác, …; các tài nguyên phần mềm: các CSDL, các phần mềm đặc biệt bản quyền đắt giá, các đường truyền mạng,... Các tài nguyên này có thể khác nhau về mặt kiến trúc, giao diện, khả năng xử lý,…Việc tạo ra một giao diện thống nhất cho phép khai thác và sử dụng hiệu quả các nguồn tài nguyên này là hoàn toàn không dễ dàng.

- ✓ ***Sự đa dạng về chính sách quản lý tài nguyên***

Các tài nguyên không chỉ phụ thuộc về một tổ chức mà thuộc về nhiều tổ chức cùng tham gia vào lưới. Các tổ chức này phải tuân thủ một số quy định chung khi tham gia vào lưới còn nhìn chung là hoạt động độc lập tức là các tài nguyên này đều có quyền tự trị. Các tổ chức khác nhau thường có chính sách sử dụng hay cho thuê tài nguyên của họ khác nhau, do vậy cũng gây khó khăn cho việc quản lý.

- ✓ ***Sự phân tán của các tài nguyên***

Dễ nhận thấy rằng các tài nguyên khi tham gia vào lưới là không tập trung, có thể ở nhiều tổ chức nhiều vùng lãnh thổ khác nhau, miễn là các tài nguyên này có thể kết nối được với nhau vì vậy phải có cơ chế quản lý sự phân tán tài nguyên trong lưới.

- ✓ ***Vấn đề an toàn, bảo mật thông tin***

Môi trường lưới là một môi trường rất phức tạp, tuy rằng khi các tổ chức cá nhân cùng tham gia vào một mạng lưới thì sẽ có các quy định áp dụng cho họ nhưng cũng cần phải quan tâm đến việc bảo vệ an toàn thông tin cho các tổ chức khi tham gia vào lưới, đây phải là một ưu tiên hàng đầu cho những người xây dựng hệ thống lưới.

Trong bài báo "What Is Grid ?" Ian Foster cũng đã đưa ra ba đặc điểm của một hệ thống tính toán lưới:

1/. Kết hợp chia sẻ các nguồn tài nguyên không được quản lý tập trung

Grid tích hợp và phối hợp các tài nguyên, người dùng thuộc nhiều vùng quản lý khác nhau, nhiều đơn vị khác nhau trong một tổ chức và nhiều tổ chức khác nhau.
Công nghệ Grid tập trung giải quyết một số vấn đề bảo vệ tài nguyên, chính sách quản trị, chi phí, thành viên, …nảy sinh trong quá trình chia sẻ và sử dụng tài nguyên.

2/. Sử dụng các giao diện và giao thức chuẩn mang tính mở

Tính toán lưới sử dụng các chuẩn mở để chia sẻ qua mạng những tài nguyên phức tạp (trên các nền tảng kiến trúc phần mềm, phần cứng và ngôn ngữ lập trình khác nhau), nằm tại những điểm khác nhau tùy vào khu vực hành chính. Nói cách khác nó "ảo hóa" các tài nguyên tính toán.

Tính toán lưới thường bị nhầm với tính toán phân cụm, tuy nhiên có sự khác nhau giữa hai kiểu tính toán này: cụm tính toán là một tập đơn các nút tính toán tập trung trên một khu vực địa lý nhất định. Lưới tính toán gồm nhiều cụm tính toán và những tài nguyên khác (như mạng, các thiết bị lưu trữ).

3/. Cung cấp các dịch vụ có chất lượng cao

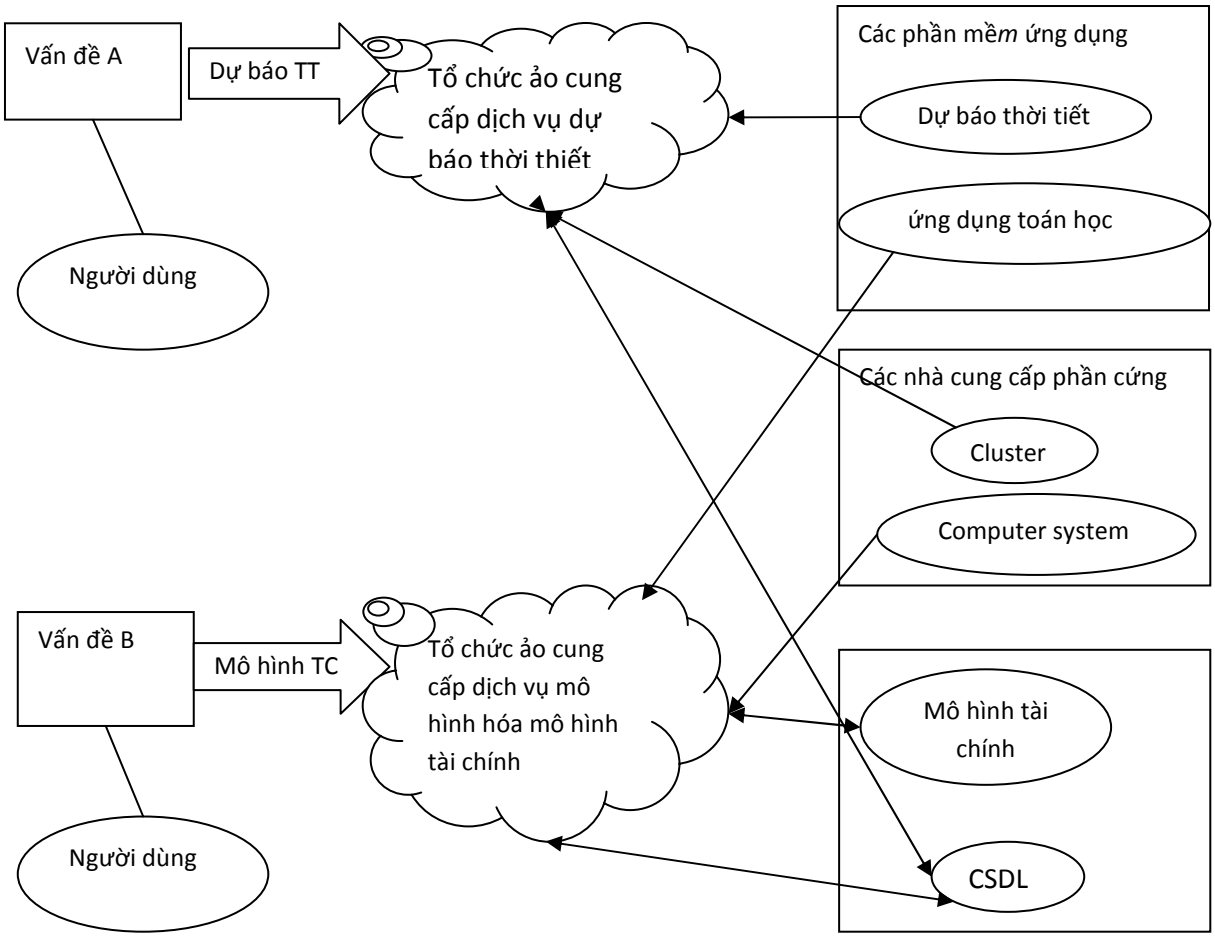
Tính toán lưới tạo ra một mô hình để giải quyết các bài toán tính toán lớn bằng cách sử dụng những tài nguyên rỗi (CPU, thiết bị lưu trữ) của một loạt các máy tính riêng rẽ, thường là máy để bàn. Hệ thống này được coi là một cụm "máy ảo", nhúng trong một môi trường liên lạc phân tán. Tính toán lưới tập trung vào khả năng hỗ trợ tính toán giữa các khu vực hành chính, điều này làm cho mô hình này khác biệt so với mô hình cụm tính toán và tính toán phân tán truyền thống.

Tính toán lưới cung cấp một giải pháp cho những bài toán về tính toán hiệu năng cao như tạo nếp protein, mô hình hoá tài chính, mô phỏng động đất và dự báo khí hậu thời tiết, … Ngoài ra tính toán lưới còn có thể giúp các tổ chức, doanh nghiệp sử dụng tối ưu các tài nguyên CNTT và tạo ra các dịch vụ tính toán theo nhu cầu cho khách hàng thương mại, trong đó khách hàng chỉ phải trả những gì họ đã sử dụng giống như điện và nước.

Tính toán lưới được thiết kế với mục tiêu giải các bài toán tính toán quá lớn cho một siêu máy tính, trong khi vẫn giữ được sự linh hoạt đối với những bài toán nhỏ hơn. Vì vậy tính toán lưới cung cấp một môi trường đa người dùng.

Mục tiêu thứ hai của tính toán lưới là khả năng khai thác tốt hơn những năng lực tính toán chưa được sử dụng và phục vụ cho những nhu cầu tính toán không ngừng của các bài toán khoa học lớn. Điều này dẫn đến việc sử dụng các cơ chế cấp phép an toàn, cho phép người dùng từ xa có thể điều khiển được các tài nguyên tính toán.

Khái niệm "tổ chức ảo" là một khái niệm rất quan trọng trong tính toán lưới. Tổ chức "ảo" là một tổ chức được lập ra để giải quyết một vấn đề nào đó. Thành phần của tổ chức ảo bao gồm nhiều tài nguyên thuộc về nhiều tổ chức thực khác nhau trong môi trường lưới cùng hoạt động vì một mục tiêu chung. Tùy theo mức độ của vấn đề cần giải quyết mà các tổ chức ảo có thể rất khác nhau về quy mô, phạm vi hoạt động và thời gian sống. Hình phía dưới minh họa về một tổ chức ảo. Có một người dùng cần giải quyết một bài toán lớn về dự báo thời tiết, anh ta thành lập một tổ chức ảo bằng cách thuê một số nguồn tài nguyên khác nhau từ một vài tổ chức khác nhau. Tương tự như vậy, một người dùng cần giải một bài toán về dự báo tài chính, anh ta cũng thành lập một tổ chức ảo để giải quyết bài toán này.



Hình 1 : Ví dụ mô hình tổ chức tính toán lưới

1.1.3. Lịch sử phát triển

Mặc dù hiện tại thì khái niệm về Grid vẫn còn khá mới mẻ, đặc biệt là tại Việt Nam. Nhưng khái niệm về Grid đã xuất hiện dưới dạng này và dạng khác trong lịch sử tính toán từ khá lâu. Ví dụ như ý tưởng "chia sẻ năng lực tính toán" đã xuất hiện từ những năm 60-70 của thế kỷ XX.

Năm 1965, những người phát triển hệ điều hành Mulitics (tiền nhân của hệ điều hành Unix) đã đề cập đến việc sử dụng năng lực tính toán như một tiện ích, một quan điểm rất gần với quan điểm về Grid như hiện nay. Đó là một hệ thống cung cấp năng lực tính toán tương tự như hệ thống cung cấp điện, nước hiện đang được sử dụng trong cuộc sống hằng ngày. Người dùng khi muốn sử dụng tài nguyên tính toán để xử lý công việc, chỉ cần cắm thiết bị vào hệ thống cung cấp, sử dụng và trả tiền giống như khi cắm thiết bị điện vào lưới điện.

Tuy nhiên đó mới là những ý tưởng về Grid nhưng nguồn gốc của Grid chính thức được xác định vào năm 1990, khi thuật ngữ "siêu tính toán" ra đời, dùng để mô tả các dự án kết nối các trung tâm siêu máy tính của Mỹ nhằm kết hợp sức mạnh của nhiều siêu máy tính lại với nhau.

Những năm 1997 – 1999, có một dự án phi lợi nhuận SETI@home là một trong những nhân tố khoa học nổi tiếng thúc đẩy vào việc tạo ra một dự án tính toán lưới đơn giản bằng cách thu thập các tài nguyên CPU chưa được sử dụng. Những người theo chủ nghĩa Grid thuần túy cho rằng CPUi@home thực chất là một ứng dụng tính toán phân tán, bởi nó hầu như không thúc đẩy việc sử dụng bất kỳ một khái niệm tính toán lưới nào. SETI@home không phải là dự án đầu tiên mở đường cho kỹ thuật này, việc tận dụng tài nguyên CPU trên máy tính cục bộ đã bắt đầu từ thập niên 1970 với những dự án phi lợi nhuận như DISTRIBUTED. NET, nhưng SETI@home nổi tiếng bởi dự án này được ứng dụng vào nhiều dự án khác như: tạo nếp Protein, nghiên cứu thuốc cho bệnh ung thư, giải các bài toán phức tạp và dự báo thời tiết, . . Hầu hết các dự án này đều được thực hiện dưới dạng các tiến trình chạy trên nền máy tính cá nhân, xử lý những dữ liệu nhỏ khi máy tính ở trạng thái chờ hoặc ít sử dụng tài nguyên.

Năm 1997, một trong những dịch vụ tính toán lưới thương mại đầu tiên đã được Entropia cung cấp, tới nay cũng có nhiều dịch vụ như vậy do các công ty hay các phòng thí nghiệm thực hiện. Điều khác biệt quan trọng giữ dự án "Lưới" và dự án "giống lưới" là trong lưới cho phép di trú các nhiệm vụ tính toán lưới tới tất cả các nút tính toán trên lưới để thực thi. Chẳng hạn như chương trình xử lý ảnh viễn vọng SETI@home chứa cả mã xử lý dữ liệu từ kính viễn vọng vô tuyến và mã để lấy dữ liệu từ cơ sở dữ liệu và trả lại kết quả. Hai mã này được trộn lẫn vào một chương trình.

Tính toán lưới hiện nay thì đang có xu hướng phát triển mạnh và được nhiều nhà nghiên cứu quan tâm. Hai nhóm gồm Globus Alliance (được sự tài trợ của một vài trường đại học tại Mỹ như đại học Chicago, đại học Berkeley,…) và Global Grid Forum (các thành viên bao gồm các hãng lớn như IBM, SUN, Microsoft,…) là các trung tâm nghiên cứu đáng chú ý hiện nay. Các nhóm này đã tạo ra các chuẩn mã nguồn mở và các giải pháp phần mềm cho công nghệ mới mẻ này. Đó là một nền tảng để các thành phần trong lưới có thể giao tiếp được với nhau. Trong đó:

- Globus Alliance tạo ra bộ công cụ Globus Toolkit (GT) mã nguồn mở, bao gồm các thư viện phần mềm và các dịch vụ cho phép người phát triển tạo ra các ứng dụng lưới. Thư viện GT cung cấp các hàm đảm bảo vấn đề như an ninh, cơ sở hạ tầng thông tin, quản lý tài nguyên lưới, tính tin cậy, tính khả chuyển, . . .
- Global Grid Forum quản lý các tiến trình chuẩn cho việc đặc tả kiến trúc các dịch vụ lưới OGSA (Open Grid Service Architecture) và OGSI (Open Grid Service Infrastructure).

Các chuẩn OGSA, OGSI và bộ công cụ Globus Toolkit giúp cho các nhà phát triển triển khai một cách thuận lợi các giải pháp tính toán lưới trong nhiều lĩnh vực nghiên cứu chuyên sâu ở Mỹ và Châu Âu như: dự án tìm kiếm các tín hiệu ngoài trái đất SETI@home, dự án về nghiên cứu bản đồ gen người, dự án IPG (Information Power Grid) của NASA, …Đó là những ứng dụng tiêu biểu cho sự thành công ban đầu của tính toán lưới trong giai đoạn nghiên cứu.

1.1.4. Các tổ chức tham gia phát triển tính toán lưới

1/. Các tổ chức phát triển chuẩn lưới

Đại diện cho nhóm này là diễn đàn lưới toàn cầu (GGF – Global Grid Forum) và các tổ chức chuẩn hóa quốc tế khác như OASIS (Organization for the Advancement of Structure Information Standards). W3C (World Wide Web Consortium), IETF (the Internet Engineering Task Force) và DMTF (the Distributed Management Task Force). Hiện nay một trong những hoạt động chính của GGF là phát triển chuẩn dịch vụ lưới OGSA.

2/. Các tổ chức phát triển bộ công cụ framework và các middleware

Bao gồm các trường đại học, các viện nghiên cứu. Các tổ chức này đã cho ra đời nhiều bộ công cụ phát triển lưới như Legion, Condor, Nimrod, Unicore, Globus, …

3/. Các tổ chức xây dựng và sử dụng các giải pháp lưới

Có thể kể ra một số lưới tiêu biểu trên thế giới như Nasa Information Power Grid của NASA, Science Grid của bộ quốc phòng Mỹ, dự án EuroGrid của liên minh Châu Âu với nhiều lưới con như Bio Grid, Metro Grid, Computer-Aided Engineering (CAD) Grid,…

4/. Các tổ chức đưa công nghệ lưới vào các sản phẩm thương mại

Trong nhóm này có nhiều đại gia trong ngành công nghiệp máy tính như IBM, SUN, HP, …Các hãng này đưa ra nhiều giải pháp khác nhau dựa trên nền công nghệ tính toán lưới. Hiện nay trên thế giới đã có sự phân biệt giữa công nghệ tính toán lưới mang tính hàn lâm và công nghệ tính toán lưới trong doanh nghiệp.

1.2. MỘT SỐ MÔ HÌNH TÍNH TOÁN KHÁC

1.2.1 World Wide Web (Web Computing)

WWW hiện nay đang phát triển mạnh mẽ và được sử dụng rộng khắp. Sử dụng các chuẩn mở và các giao thức mở (TCP, HTTP, XML, SOAP), WWW có thể được sử dụng để xây dựng các tổ chức ảo tuy nhiên nó thiếu một số đặc tính quan trọng như các cơ chế chứng thực một lần, ủy nhiệm, các cơ chế phối hợp sự kiện …

1.2.2. Hệ thống tính toán phân tán (Distributed Computing system)

Các công nghệ tính toán phân tán hiện tại bao gồm CORBA, J2EE và DCOM rất thích hợp cho các ứng dụng phân tán tuy nhiên chúng không cung cấp một nền tảng phù hợp cho việc chia sẻ tài nguyên giữa các thành viên của tổ chức ảo. Một số khó khăn có thể kể ra trong việc khai phá tài nguyên, đảm bảo an ninh và xây dựng động các tổ chức ảo. Thêm nữa việc tương tác giữa các công nghệ này cũng gặp phải khó khăn. Tuy nhiên, cũng đã có một số nghiên cứu nhằm mở rộng những công nghệ này cho môi trường lưới như Java JINI.

1.2.3. Nhà cung cấp dịch vụ ứng dụng và dịch vụ lưu trữ (Application and storage service provider)

Các nhà cung cấp ứng dụng và dịch vụ lưu trữ thường cung cấp cho người dùng ứng dụng cụ thể nào đó, cũng như không gian lưu trữ. Người dùng tương tác với nhà cung cấp dịch vụ thường thông qua mạng riêng ảo (VPN), hoặc đường truyền dành riêng, vì vậy nên loại bỏ được nhiều nguy cơ về an toàn bảo mật. Do vậy khi nói về các loại dịch vụ này, thì ngữ cảnh của chúng cũng hẹp hơn tính toán lưới rất nhiều.

1.2.4. Hệ thống tính toán ngang hàng

(Peer-to-Peer Computing system)

Tính toán ngang hàng cũng là một lĩnh vực của tính toán phân tán. Một số hệ thông tính toán ngang hàng phổ biến hiện nay là SETI@home, hay các mạng ngang hàng chia sẻ tệp tin như Napter, Kazaa, Morpheus, Gnutella. Những điểm khác biệt chính giữa tính toán ngang hàng và tính toán lưới là:

- ✓ Cộng đồng người sử dụng mà chúng hướng tới. Tính toán lưới có cộng đồng người dùng có thể nhỏ hơn, tuy nhiên tập trung nhiều vào các ứng dụng và có yêu cầu cao hơn về an ninh cũng như tính toàn vẹn của ứng dụng. Trong khi hệ thống mạng ngang hàng có thể có số người dùng rất lớn, bao gồm cả các người dùng đơn lẻ và các tổ chức tuy nhiên không đòi hỏi cao về an ninh, và mô hình chia sẻ tài nguyên cũng đơn giản hơn.
- ✓ Môi trường lưới liên kết các nguồn tài nguyên mạnh hơn, đa dạng hơn và chặt chẽ hơn.

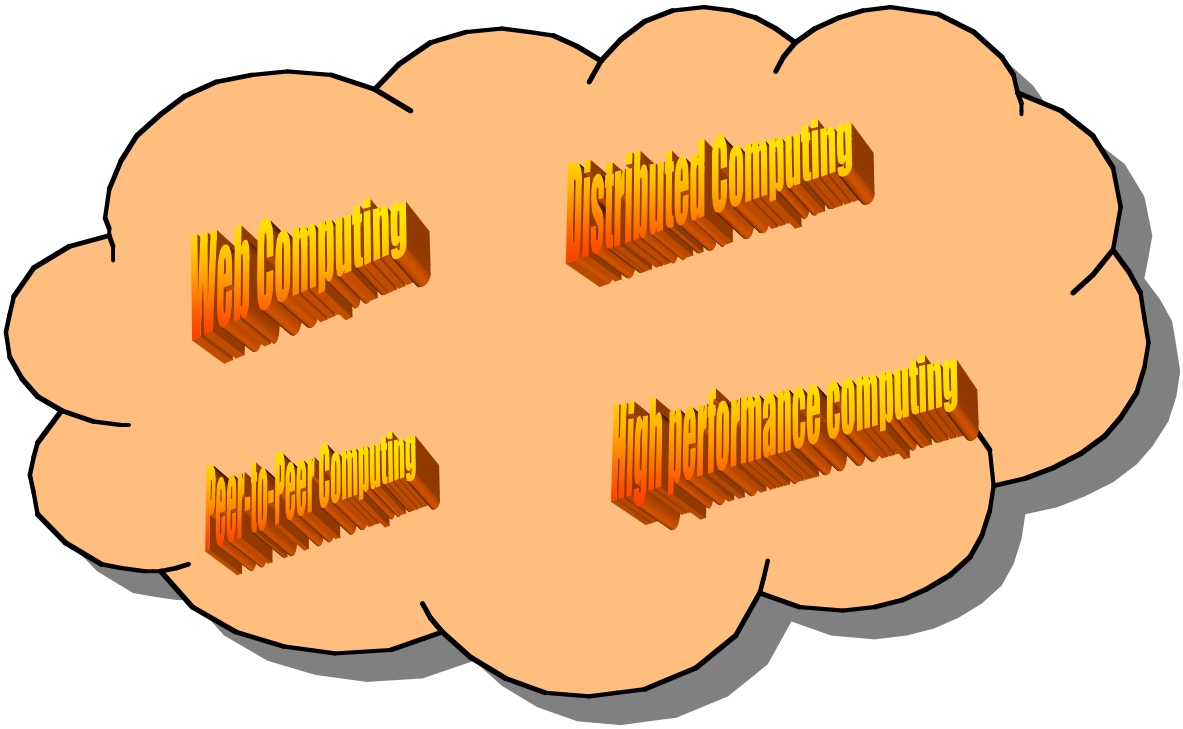
1.2.5. Công nghệ tính toán hiệu năng cao

(High performance computing)

Như đã nói ở trên, để giải quyết những bài toán lớn người ta có thể đầu tư cho cơ sở hạ tầng tính toán. Để giải quyết những bài toán rất lớn và phức tạp người ta phải nghiên cứu xây dựng hệ thống siêu tính toán. Các hướng nghiên cứu trong tính toán hiệu năng cao chủ yếu bao gồm:

- ✓ Nghiên cứu cơ chế tạo siêu máy tính tuần tự đơn bộ vi xử lý với tốc độ rất cao. Cách làm này gặp phải các giới hạn về vật lý như độ truyền dẫn của bán dẫn, tốc độ điện từ, nhiễu điện từ …nên không thể tăng mãi được.
- ✓ Để khắc phục khó khăn trên, người ta nghiên cứu chế tạo các siêu máy tính song song bao gồm nhiều bộ xử lý hoạt động song song trên một bảng mạch chủ. Cách làm này đòi hỏi phải có những phần mềm tương thích để tận dụng năng lực tính toán của hệ thống, ví dụ: hệ điều hành song song phân tán, trình biên dịch song song, ngôn ngữ lập trình song song ,…

Tuy nhiên việc nghiên cứu chế tạo ra các siêu máy tính nói chung mới chỉ được thực hiện ở các nước phát triển và giá thành của một hệ thống siêu máy tính như vậy (bao gồm cả phần cứng lẫn phần mềm hệ thống, công cụ phát triển) có thể lên đến hàng triệu đô la.



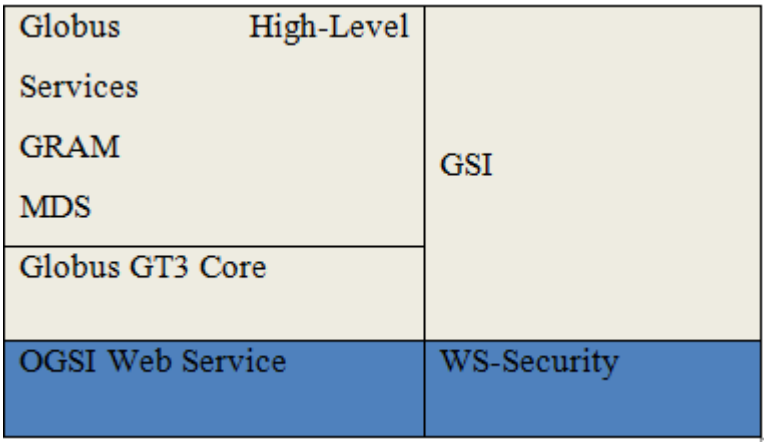
Hình 2: Các mô hình tính toán

1.3. MỘT SỐ CÔNG CỤ TÍNH TOÁN LƯỚI HIỆN NAY

Hiện nay trên thế giới có nhiều bộ công cụ phát triển hỗ trợ việc xây dựng lưới ở nhiều mức độ khác nhau. Tiêu biểu là:

1.3.1. Bộ công cụ Globus

Globus là một dự án nghiên cứu gồm nhiều tổ chức tham gia với mục tiêu ban đầu là tạo cơ sở hạ tầng và các dịch vụ cấp cao cho một lưới tính toán tuy nhiên hiện nay nó đã mở rộng phạm vi thành cơ sở hạ tầng cho phép chia sẻ nhiều loại tài nguyên đa dạng. Bộ công cụ GT đã trải qua nhiều giai đoạn phát triển với nhiều phiên bản, hiện nay phiên bản mới nhất là GT 5. 0. 1 phát hành vào tháng 3 năm 2010.



Hình 3 :Các dịch vụ cơ bản của GT (Globus Toolkit)

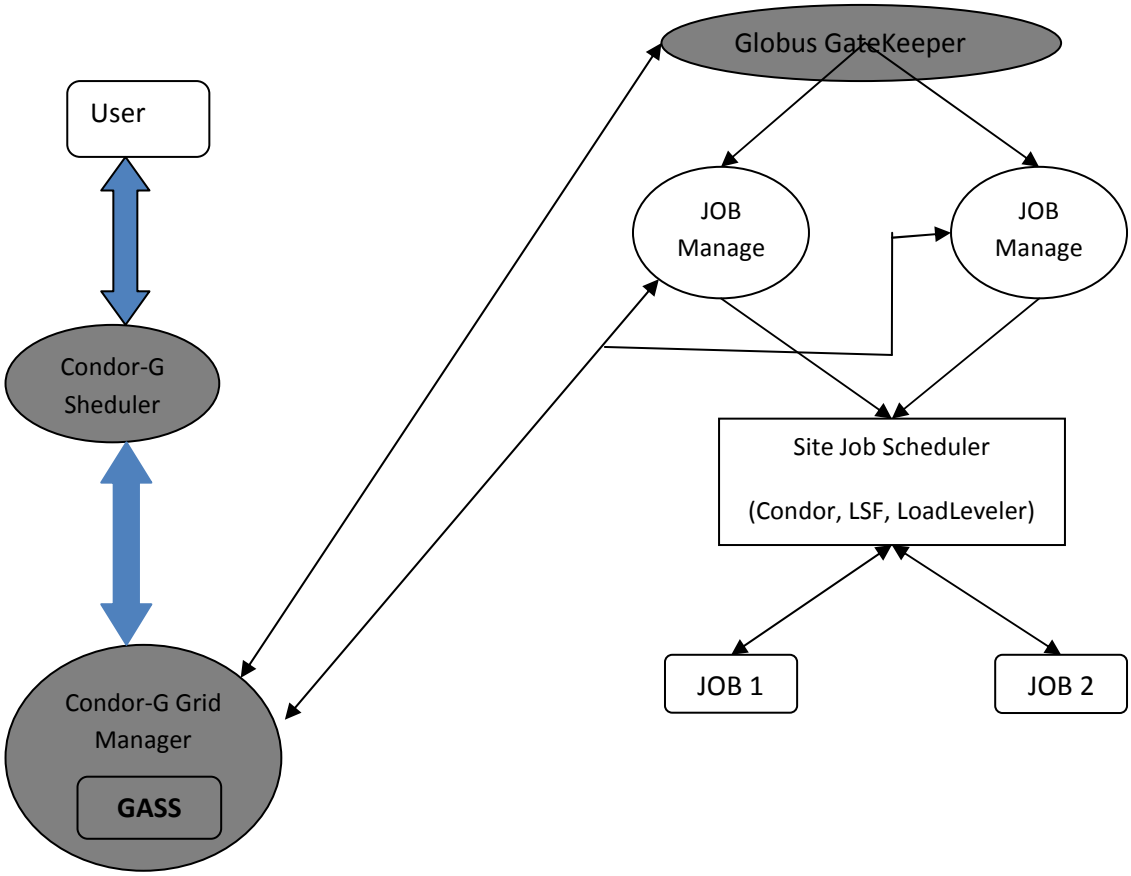
Như mô tả ở hình trên, Globus được thiết kế theo kiến trúc phân tầng với tầng dưới cũng chính là các dịch vụ Web. Tầng tiếp theo là các dịch vụ lõi, và tầng trên cùng là các dịch vụ cấp cao cung cấp các chức năng sử dụng cho người dùng. Có thể thấy xuyên suốt kiến trúc của GT là các cơ chế bảo mật, cũng dựa trên nền tảng bảo mật của dịch vụ Web.

1.3.2. Bộ công cụ Legion

Legion là một dự án phát triển middleware cho lưới do trường đại học Virginia phát triển. Bộ công cụ Legion Toolkit được phát hành lần đầu tiên vào năm 1997. Đây là bộ công cụ lưới hướng đối tượng. Năm 1998 công ty Avaki được thành lập đưa bộ công cụ này vào sử dụng cho mục đích thương mại.

1.3.3. Bộ công cụ Condor

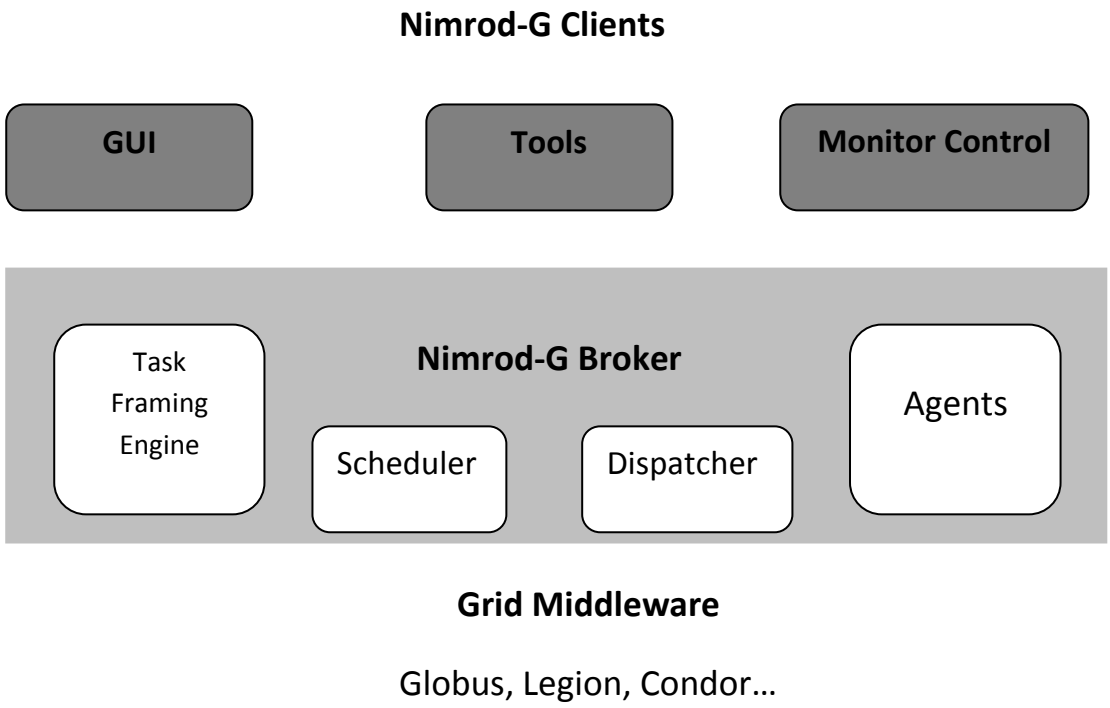
Condor là một công cụ cho phép tận dụng thời gian rảnh rỗi của các máy tính vào công việc tính toán, rất thích hợp cho các ứng dụng dạng nghiên cứu tham số và tính toán thông lượng cao trong đó các công việc thường không cần liên lạc với nhau. Condor G là một biến thể của Condor cho phép kết nối với các hệ thống lưới dựa trên GT.



Hình 4: Kết nối giữa Condor-G và GT

1.3.4. Bộ công cụ Nimrod

Nimrod là bộ công cụ cung cấp cho người dùng một giao diện để mô tả các ứng dụng dạng nghiên cứu tham số. Nimrod G là một biến thể của Nimrod cho phép liên kết với các hệ thống lưới trên nền GT.



Hình 5: Kiến trúc Nimrod G

1.3.5. Dự án Unicore

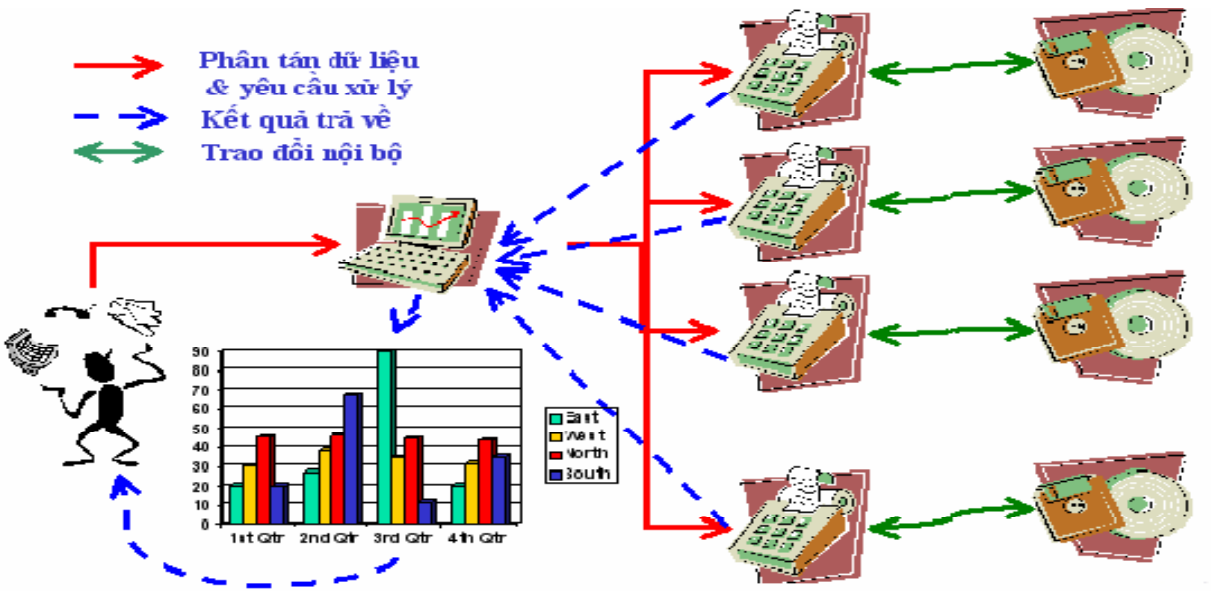
Unicore là dự án nghiên cứu lưới được tài trợ bởi bộ giáo dục Đức. Hiện nay có một dự án mang tên GRIP (Grid Interoperability Project) đã được thực hiện từ năm 2002, nhằm mang lại khả năng tương tác giữa Unicore và Globus.

1.4. PHÂN LOẠI LƯỚI TÍNH TOÁN

1.4.1. Lưới tính toán (Computation Grid)

Lưới tính toán là một loại của tính toán lưới, chủ yếu tập trung vào việc sử dụng năng lực tính toán của lưới. Ở loại lưới này, phần lớn các node là các nhóm máy tính có năng lực tính toán lớn, để phục vụ cho việc tính toán bài toán lớn.

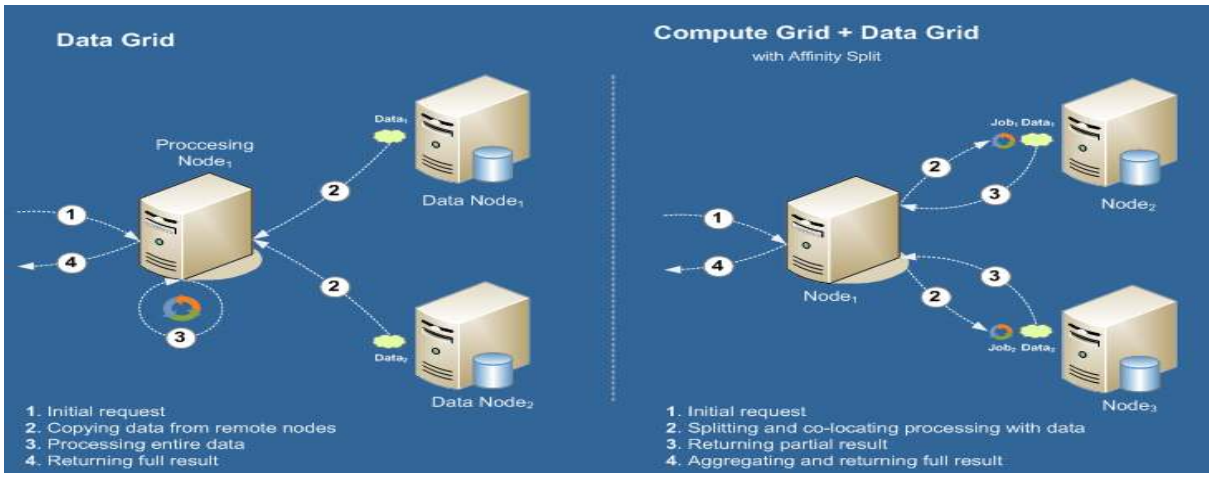
Hình thức thực hiện là chia tác vụ tính toán lớn thành nhiều công việc nhỏ thực thi song song trên các node của lưới. Việc phân tán các tác vụ của lưới sẽ làm giảm đáng kể thời gian xử lý, và làm tăng khả năng tận dụng của hệ thống. Thông thường hệ thống chính sẽ chia khối dữ liệu cần xử lý thành các phần nhỏ, sau đó phân phối đến các node trên grid. Mỗi node thực hiện xử lý dữ liệu, kết quả trả về hệ thống chính, tại đây sẽ tổng hợp và trình diễn kết quả toàn cục cho người dùng.



Hình 6: Lưới tính toán

1. 4. 2. Lưới dữ liệu (data grid)

Grid dữ liệu sẽ tập trung vào việc lưu trữ và cung cấp khả năng truy xuất dữ liệu của nhiều cá nhân, tổ chức khác nhau. Người dùng không cần biết chính xác vị trí dữ liệu khi thao tác với dữ liệu. Các cơ sở dữ liệu, đặc biệt là các cơ sở dữ liệu liên hợp đóng vai trò quan trọng trong grid dữ liệu nhất là khi có nhiều nguồn dữ liệu và xuất hiện nhu cầu kết hợp các thông tin từ các nguồn dữ liệu này. Grid dữ liệu có thể được sử dụng trong lĩnh vực khai phá dữ liệu (data mining), hoặc các hệ thống thương mại thông minh. Trong trường hợp này, không chỉ có hệ thống file hay các cơ sở dữ liệu mà toàn bộ dữ liệu của tổ chức cần tập hợp lại. Ở đây có thể kết hợp grid dữ liệu và grid tính toán.



Hình 7: data grid và data grid + compute grid

1. 4. 3. Lưới kết hợp (Scavenging grid)

Scavenging grid có thể được xem là một loại kết hợp giữa data grid và compute grid. Một scavenging thường được dùng với nhiều máy tính để bàn. Các máy tính sẽ được kiểm tra định kỳ để xem khi nào bộ xử lý và các tài nguyên khác rảnh rỗi để thực hiện các tác vụ grid. Chủ nhân của các máy để bàn này sẽ được quyền xác định khi nào thì sẽ chia sẻ máy tính của mình với mạng lưới.

1.5. LỢI ÍCH CỦA TÍNH TOÁN LƯỚI

1.5.1. Khai thác tận dụng các nguồn tài nguyên nhàn rỗi

Đây có thể được coi là lợi ích lớn nhất mà grid mang lại và cũng là lợi ích dễ nhìn thấy nhất khi triển khai một hệ thống grid. Hầu hết các tổ chức đều có một lượng lớn các tài nguyên tính toán nhàn rỗi là các máy tính trong tổ chức của mình (bao gồm cả máy chủ). Các máy tính cá nhân thường chỉ sử dụng hết 5% thời gian xử lý CPU, ngay cả các sever cũng thường rảnh rỗi. Grid có thể tối ưu sử dụng các tài nguyên nhàn rỗi này theo nhiều cách khác nhau.

Ví dụ: Gửi một công việc trên một máy tính đang bận rộn đến một máy tính nhàn rỗi hơn để xử lý, hoặc phân nhỏ một công việc rồi gửi các công việc con đến các máy tính nhàn rỗi khác cho xử lý song song, …

Một chức năng nữa của grid đó là cân bằng sử dụng tài nguyên tốt hơn. Một tổ chức thường gặp các vấn đề không mong đợi khi các hoạt động đòi hỏi thêm nhiều tài nguyên. Với grid, có thể chuyển hoạt động đến tài nguyên nhàn rỗi khác, hoặc có thể thêm tài nguyên mới một cách dễ dàng, từ đó làm tăng khả năng của hệ thống.

"Lưới" cho phép kết hợp nhiều không gian lưu trữ nhàn rỗi để tạo thành một không gian lưu trữ lớn hơn, được cấu hình để tăng hiệu suất, độ tin cậy hơn so với các máy tính đơn lẻ thông qua các cơ chế quản lý dữ liệu.

"Lưới" có thể quản lý nhiều loại tài nguyên, do đó có thể cho phép theo dõi tổng quan về các hoạt động sử dụng tài nguyên trong một tổ chức lớn, hỗ trợ hoạch định các chiến lược sử dụng tài nguyên.

1.5.2. Sử dụng bộ xử lý song song

Khả năng sử dụng CPU song song là một đặc tính tuyệt vời của grid, ngoài việc hỗ trợ các nhu cầu tính toán của các nhà khoa học, sức mạnh tính toán do grid cung cấp có thể giúp giải quyết các bài toán đòi hỏi năng lực xử lý lớn trong các ngành y dược, tính toán tài chính, kinh tế và khai thác dầu hỏa, dự báo thời tiết, công nghiệp vũ trụ, thiết kế sản phẩm,… và nhiều lĩnh vực khác.

1.5.3. Cho phép hợp tác trên toàn thế giới

Một trong những đóng góp quan trọng của tính toán lưới là cho phép đơn giản hóa hợp tác chia sẻ, làm việc giữa cộng đồng rộng lớn trên toàn thế giới.

Các công nghệ phân tán trước đây cũng cho phép hợp tác nhưng chỉ trong quy mô nhỏ, còn grid cho phép trên phạm vi toàn cầu khi đưa ra những chuẩn quan trọng cho phép các hệ thống không đồng dạng làm việc chung với nhau để tạo nên một hệ thống tính toán ảo cung cấp rất nhiều dạng tài nguyên khác nhau.

1.5.4. Cho phép chia sẻ tất cả các loại tài nguyên

Không chỉ cho phép chia sẻ các chu kỳ tính toán dữ liệu, grid còn cho phép chia sẻ tất cả các loại tài nguyên mà trước đây chưa được chia sẻ như băng thông mạng, các thiết bị đặc biệt, phần mềm, bản quyền và các dịch vụ, …

Ví dụ: một người muốn tăng băng thông kết nối internet của mình lên để thực hiện một ứng dụng khai thác dữ liệu, ứng dụng đó có thể được gửi đến nhiều máy tính trong grid có các kết nối internet riêng, từ đó băng thong truy cập internet của người đó sẽ tăng lên rất nhiều lần, …

1. 5. 5. Tăng tính tin cậy cho các hệ thống máy tính

Hiện nay, các hệ thống tính toán sử dụng các phần cứng chuyên dụng, đắt đỏ để tăng độ tin cậy. Ví dụ, có thể sử dụng các "chip" có các mạch dự phòng để có thể phục hồi lỗi khi có sự cố về phần cứng. Một máy tính có thể sử dụng các bộ vi xử lý đôi, cho phép "cắm nóng", để khi có một vi xử lý bị hỏng, có thể thay thế cái khác mà không làm ngưng hoạt động của hệ thống. Các giải pháp này làm tăng độ tin cậy của hệ thống, tuy nhiên với chi quá đắt khi phụ kiện đi kèm cũng phải nhân lên.

Trong tương lai, các hướng tiếp cận mới để giải quyết vấn đề độ tin cậy dựa nhiều hơn vào các công nghệ phần mềm hơn là các phần cứng đắt tiền. Grid là sự khởi đầu cho các công nghệ đó. Các hệ thống trong Grid thường rẻ và phân tán theo địa lý, do đó, nếu có sự cố về nguồn điện hay các lỗi hệ thống khác tại một vị trí, toàn bộ phần còn lại không bị ảnh hưởng.

Các phần mềm quản trị Grid có khả năng thực thi lại công việc trên một node khác khi phát hiện có lỗi trong hệ thống. Nếu quan trọng hơn nữa, trong các hệ thống theo thời gian thực, nhiều bản dự phòng của các công việc quan trọng có thể được chạy trên nhiều máy tính khác nhau trong Grid để đảm bảo độ tin cậy tối đa.

1. 5. 6. Tăng khả năng quản trị các hệ thống

Mục tiêu "ảo hóa" tất cả các tài nguyên và cung cấp giao diện quản lý đơn nhất các hệ thống hỗn tạp đem lại những cơ hội mới để quản trị tốt hơn trong các cơ sở hạ tầng công nghệ thông tin lớn, phân tán. Bên cạnh đó đối với tầm quản lý vĩ mô có nhiều dự án sử dụng hạ tầng thông tin, grid cho phép quản lý độ ưu tiên sử dụng tài nguyên của các dự án này.

Trước đây mỗi dự án thường chịu trách nhiệm quản lý một số tài nguyên, thường xảy ra tình trạng các các tài nguyên của dự án này đang nhàn rỗi trong khi dự án kia đang gặp vấn đề, thiếu tài nguyên do gặp các sự cố không lường trước được. Với tầm nhìn rộng hơn do grid cung cấp các tình huống trên có thể được giải quyết một cách dễ dàng.

Trên đây giới thiệu một số lợi ích cụ thể của việc áp dụng công nghệ tính toán lưới, tùy vào tình huống cụ thể việc áp dụng công nghệ tính toán lưới đem lại những lợi ích khác nhau. Vấn đề là phải hiểu rõ bản chất Grid, sử dụng tốt các công cụ nhằm khai thác tốt nhất trong các tình huống cụ thể.

Chương 2. CƠ SỞ HẠ TẦNG LƯỚI

2. 1. TÀI NGUYÊN TÍNH TOÁN LƯỚI

Để nghiên cứu về kiến trúc lưới, trước tiên chúng ta cần tìm hiểu về các tài nguyên mà lưới có thể tận dụng và yêu cầu trong việc xây dựng một hệ thống lưới.

2. 1. 1. Tài nguyên tính toán

Tài nguyên quan trọng nhất của Grid là các chu kỳ tính toán (computing cycles) được cung cấp bởi bộ vi xử lý của các thiết bị trong Grid. Các bộ vi xử lý không cần phải đồng nhất về tốc độ, kiến trúc hay phần mềm khác.

Có 3 cách để khai thác tài nguyên tính toán của Grid:

+ Chạy các ứng dụng hiện có trên node của Grid thay vì chạy trên máy tính cục bộ.

+ Thiết kế ứng dụng, tách các công việc thành các phần riêng rẽ, để có thể thực thi song song trên nhiều bộ xử lý trong Grid.

+ Chạy ứng dụng thực thi nhiều lần trên nhiều node khác nhau trong Grid.

2. 1. 2. Tài nguyên lưu trữ

Mỗi thiết bị trong Grid thường cung cấp một số dung lượng lưu trữ phục vụ cho việc thực thi ứng dụng trên Grid. Tài nguyên lưu trữ có thể là bộ nhớ trong, ổ đĩa cứng hoặc các thiết bị lưu trữ khác. Bộ nhớ trong thường dùng để lưu trữ dữ liệu tạm thời cho ứng dụng, trong khi các thiết bị lưu trữ ngoài có thể được sử dụng để tăng không gian lưu trữ, tăng hiệu suất, khả năng chia sẻ và đảm bảo tính tin cậy của dữ liệu.

2. 1. 3. Phương tiện liên lạc

Khả năng liên lạc giữa các máy tính phát triển nhanh chóng đã giúp cho công nghệ Grid trở nên hiện thực, do đó đây cũng là một tài nguyên quan trọng. Phương tiện liên lạc giúp việc liên lạc, trao đổi dữ liệu giữa các thành phần trong Grid và giao tiếp giữa Grid với bên ngoài.

Một số công việc đòi hỏi một lượng dữ liệu lớn, nhưng các dữ liệu này thường không nằm trên máy đang thực thi công việc. Khả năng về băng thông trong những trường hợp như vậy là một tài nguyên then chốt, ảnh hưởng đến khả năng của Grid.

Việc giao tiếp với bên ngoài được thực hiện thông qua mạng Internet. Grid có thể sử dụng các kết nối Internet để liên lạc giữa các node. Vì các kết nối này không chia sẻ một đường truyền riêng, nên làm tăng băng thông truy cập Internet.

Các đường truyền dự phòng là cần thiết để giải quyết tốt hơn các tình huống khi hư hỏng mạng và truyền dữ liệu lớn.

2. 1. 4. Phần mềm

Các phần mềm trong Grid có thể chỉ cài đặt trên một số node trong Grid. Thông qua Grid, khi một công việc cần đến phần mềm nào, nó sẽ gửi dữ liệu đến node đó và cho thực thi. Đây có thể là một giải pháp tốt để tiết kiệm chi phí về bản quyền phần mềm.

2. 1. 5. Các thiết bị đặc biệt

Đó là các thiết bị dùng trong khoa học, kỹ thuật như kính viễn vọng… dùng để thu thập các dữ liệu khoa học, phục vụ cho các bước phân tích, xử lý sau này.

2. 2. KIẾN TRÚC LƯỚI

2. 2. 1. Bản chất của kiến trúc lưới

Tổ chức ảo (Virtual Organization) là đơn vị cơ bản quan trọng nhất của hệ thống grid. Việc thiết lập, quản lý, khai thác các quan hệ chia sẻ tài nguyên giữa các tổ chức ảo đòi hỏi phải có kiến trúc hệ thống mới, kiến trúc Grid. Kiến trúc grid được xây dựng dựa trên quan niệm: "Để các tổ chức ảo hoạt động hiệu quả đòi hỏi phải thiết lập các quan hệ chia sẻ với bất kỳ đơn vị tham gia tiềm năng nào".

Để làm được điều này, vấn đề "liên kết hoạt động" (interoperability) cần phải được tập trung giải quyết. Trong môi trường mạng, "liên kết hoạt động" đồng nghĩa với việc sử dụng các giao thức (protocol) chung. Do đó, kiến trúc Grid là kiến trúc giao thức, với các giao thức xác định, người dùng và nhà cung cấp tài nguyên thương lượng, thiết lập, quản lý và khai thác các mối quan hệ chia sẻ tài nguyên.

Kiến trúc Grid phải là kiến trúc dựa chuẩn, hướng mở, để dễ mở rộng, liên kết hoạt động tốt, có tính khả chuyển (portability) cao. Các giao thức chuẩn sẽ giúp định nghĩa các dịch vụ (service) chuẩn, nhờ đó xây dựng dễ dàng các dịch vụ cao cấp hơn.

Sau khi có được kiến trúc Grid, việc tiếp theo là xây dựng các hàm API và các bộ SDK để cung cấp các công cụ cần thiết, nhằm phát triển các ứng dụng chạy trên nền Grid.

Sở dĩ "liên kết hoạt động" được xem là vấn đề cơ bản vì các mối quan hệ chia sẻ có thể phải được thiết lập giữa các bên tham gia khác nhau về các chính sách, giữa các môi trường khác nhau về nền tảng, ngôn ngữ, môi trường lập trình, …

Nếu không có nó, các thành viên trong VO sẽ thực hiện các chính sách chia sẻ song phương sẽ không chắc rằng các cơ chế sử dụng cho 2 thành viên này có thể mở rộng được cho các thành viên khác. Điều này khiến cho việc thành lập các VO động là không thể thực hiện, hoặc chỉ thành lập được VO theo một kiểu nào đó mà thôi. Cũng giống như Web đã làm bùng nổ việc chia sẻ thông tin bằng cách cung cấp các giao thức và cú pháp chuẩn (HTTP và HTML) dùng cho việc trao đổi thông tin, ở đây cũng cần các giao thức và cú pháp chuẩn để chia sẻ tài nguyên.

Để giải quyết vấn đề "liên kết hoạt động", việc xây dựng các giao thức là quan trọng. Vì giao thức xác định cách các thành phần phân tán trao đổi với nhau để đạt được một mục đích nào đó, xác định các cấu trúc thông tin cần thiết trong quá trình trao đổi. Các VO thường hay thay đổi, nên các cơ chế xác định, chia sẻ và sử dụng tài nguyên cần phải mềm dẻo, gọn nhẹ, để các thỏa thuận chia sẻ tài nguyên có thể được thiết lập, thay đổi một cách nhanh chóng. Các cơ chế chia sẻ không được ảnh hưởng đến các chính sách cục bộ, và phải cho phép các thành viên quản lý được tài nguyên của họ.

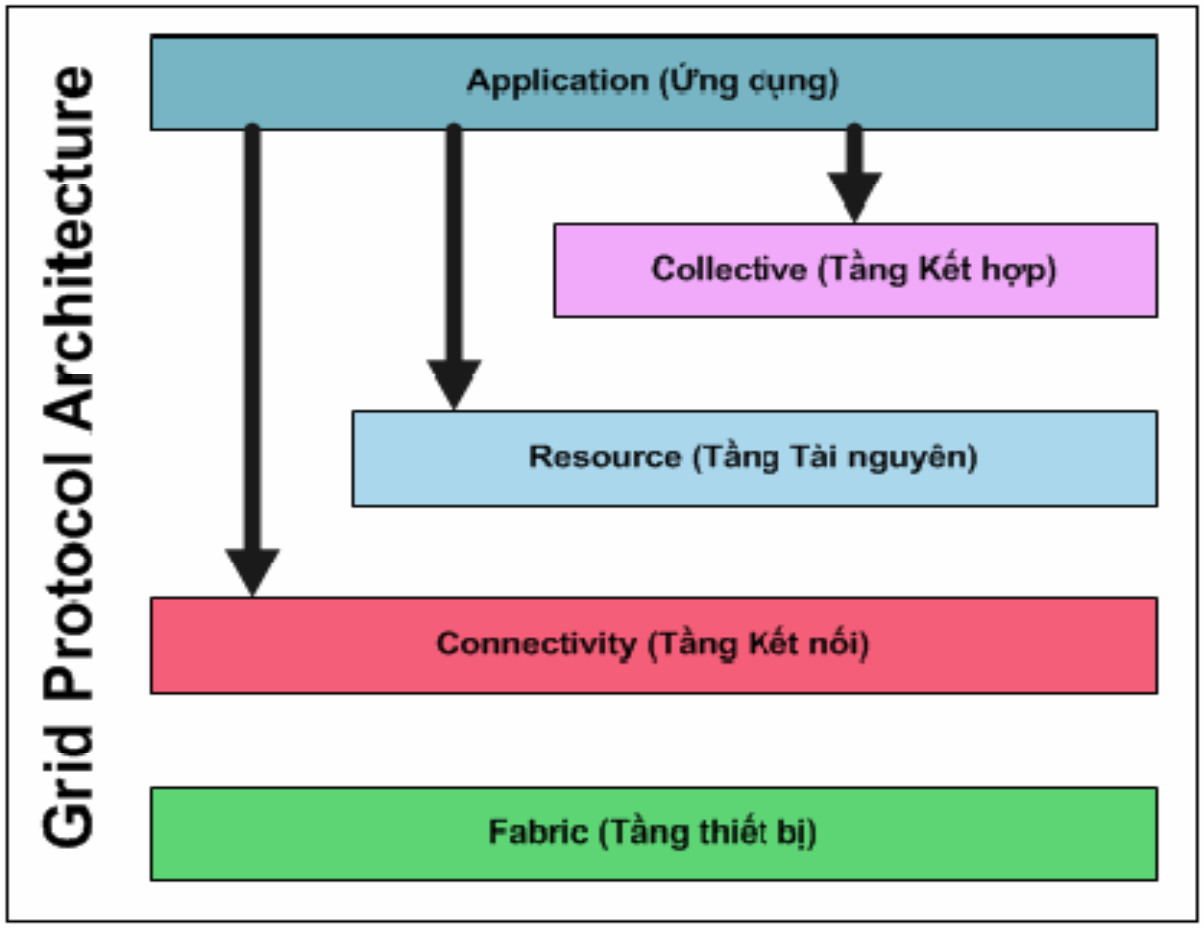
Vì các giao thức quy định việc giao tiếp giữa các thành viên chứ không quy định thành viên đó phải như thế nào, nên khi dùng các giao thức, các chính sách cục bộ được giữ lại. Do đó các giao thức được cần đến.

Khi đã có các giao thức, thì việc xây dựng các dịch vụ là cần thiết và quan trọng, các dịch vụ là bản cài đặt cụ thể của các giao thức. Việc xây dựng các dịch vụ cơ bản phục vụ truy cập đến tài nguyên tính toán, dữ liệu, tìm kiếm tài nguyên, lập lịch và đồng bộ hoá, sao chép dữ liệu, … cho phép xây dựng các dịch vụ cao cấp hơn cho ứng dụng đồng thời trừu tượng hoá các chi tiết về tài nguyên.

Cần phải xây dựng các bộ API và SDK, vì các nhà phát triển ứng dụng cần phải có công cụ để hỗ trợ phát triển các ứng dụng phức tạp trong môi trường Grid, người dùng cũng phải có khả năng thực thi được các ứng dụng này. Sức mạnh, tính đúng đắn của ứng dụng, chi phí phát triển và bảo trì là những mối quan tâm quan trọng. Các API và SDK có thể giúp tăng tốc việc phát triển mã, cho phép chia sẻ mã, tăng tính khả chuyển cho ứng dụng. Tất nhiên, API và SDK chỉ hỗ trợ thêm chứ không thể thay thế các giao thức được.

2.2.2. Kiến trúc lưới tổng quát

Kiến trúc lưới do Ian Foster đề xuất là một kiến trúc phân tầng được mô tả như hình dưới đây. Các thành phần trong cùng một tầng có đặc điểm, tính chất và có thể được xây dựng từ bất cứ tầng dưới nào. Các thành phần được phân tầng dựa theo vai trò của chúng trong grid, kiến trúc này là một kiến trúc mở. Kiến trúc chỉ quy định các yêu cầu chung nhất về thiết kế và triển khai với các mục chính là để tham khảo. Việc cài đặt cụ thể tùy thuộc vào từng dự án từng lĩnh vực cụ thể.



Hình 8: Kiến trúc lưới tổng quát

2.2.2.1. Tầng thiết bị (Fabric)

Đây là tầng thấp nhất trong kiến trúc phân tầng tính toán lưới có chức năng tương tự như tầng vật lý trong OSI. Nó bao gồm các tài nguyên được truy cập và sử dụng bởi các dịch vụ hay các ứng dụng thông qua giao thức lưới. Các tài nguyên này có thể là các tài nguyên tính toán, tài nguyên dữ liệu, tài nguyên mạng, thiết bị ngoại vi hoặc cao hơn là hệ thống tệp phân tán, các tài nguyên chuyên dụng …

Tương tự như các API trong hệ điều hành, tầng nền thực hiện các thao tác trên các tài nguyên cụ thể và chúng được gọi bởi các ứng dụng hay dịch vụ ở các tầng trên như tầng liên kết, tầng tài nguyên. Các hàm thực hiện ở tầng nền độc lập với nhau và các tài nguyên trên tầng nền có thể cho phép nhiều thao tác hay chức năng thực hiện đồng thời. Nếu ứng dụng cần ít các thao tác thì việc triển khai lưới càng dễ dàng.

Đối với các tài nguyên lưới thông thường việc tối thiểu là chúng phải hỗ trợ các hàm cho phép các ứng dụng hay dịch vụ ở mức trên có thể thực hiện các thao tác theo dõi, lấy thông tin trạng thái tài nguyên, hỗ trợ quản lý tài nguyên. Điều này rất quan trọng trong việc đảm bảo chất lượng dịch vụ lưới.

2.2.2.2. Tầng kết nối (Connectivity)

Tầng kết nối có nhiệm vụ định nghĩa các giao thức truyền thông và chứng thực cần thiết cho việc giao tiếp trong lưới. Các giao thức truyền thông cho phép thực hiện trao đổi dữ liệu giữa các tài nguyên trong tầng nền. Mô hình truyền thông lưới có nhiều điểm tương đồng so với mô hình giao thức TCP/IP đang dùng hiện nay. Các giao thức chứng thực cung cấp cơ chế mã hóa, giải mã, kiểm tra định danh của người dùng cũng như tài nguyên.

Trong lĩnh vực tính toán lưới, vấn đề bảo mật và an ninh rất quan trọng trong đó các giao thức chứng thực đóng một vai trò cơ bản. Việc chứng thực trong lưới thực hiện ở các điểm sau:

- ✓ Cơ chế chứng thực một lần (single sign on): Người dùng chỉ cần đăng nhập vào lưới một lần, và sử dụng các dịch vụ của hệ thống lưới với quyền hạn xác định.
- ✓ Cơ chế ủy quyền (delegation): Người dùng phải chịu trách nhiệm về các thao tác của mình như việc thực hiện các trình ứng dụng. Khi có các ứng dụng này có thể sử dụng các tài nguyên của lưới theo quyền hạn của người dùng đã ủy quyền. Mặt khác, bản thân trình ứng dụng cũng có thể ủy quyền cho một hay nhiều công việc con của nó.
- ✓ Tích hợp các giải pháp bảo mật địa phương mỗi node lưới để có những cơ chế bảo mật riêng. Các giao thức chứng thực phải có sự đồng bộ, kết hợp với các giải pháp địa phương.
- ✓ Chứng thực đa phương (mutual authorization): người dùng có thể cùng một lúc sử dụng tài nguyên trên các trạm khác nhau mà các nhà quản trị ở các trạm khong cần giao tiếp với nhau để xác định lại.

2. 2. 2. 3. Tầng tài nguyên (Resource)

Tầng tài nguyên được xây dựng trên tầng kết nối, có nhiệm vụ sử dụng các giao thức truyền thông và bảo mật của tầng kết nối để xây dựng dịch vụ, giao thức đàm phán khởi tạo theo dõi và điều khiển các thủ tục giao tiếp với các tài nguyên cụ thể. Việc điều khiển, theo dõi các tài nguyên được thực hiện bằng cách triệu gọi các hàm của tầng nền. Tầng tài nguyên bao gồm hai lớp giao thức cơ bản:

- ✓ Các giao thức thông tin

  Được sử dụng để lấy các thông tin cấu trúc, trạng thái của một tài nguyên nào đó.

- ✓ Các giao thức quản lý

  Các giao thức này có nhiệm vụ thực hiện việc đàm phán để có thể truy cập và sử dụng một tài nguyên nào đó. Các giao thức quản lý có nhiệm vụ xác lập quan hệ giữa người dùng lưới hay các ứng dụng lưới với các tài nguyên cụ thể. Vì vậy cần hết sức chú ý các chính sách, quyền hạn mà người dùng có thể thực hiện trên các tài nguyên này.

2. 2. 2. 4. Tầng kết hợp (Collective)

Nếu như tầng tài nguyên quan tâm tới các tài nguyên cụ thể thì tầng kết hợp được xây dựng có nhiệm vụ quản lý các tài nguyên ở mức hệ thống. Các giao thức trong tầng này không thực hiện trên một tài nguyên cụ thể nào mà nó thao tác trên tất cả các tài nguyên lưới tại các node khác nhau. Các dịch vụ được cung cấp bởi tầng kết hợp bao gồm:

- Dịch vụ thư mục: Cho phép người dùng lưới có thể quan sát, theo dõi được các tài nguyên trong hệ thống trên phạm vi tổ chức nào đó mà họ có thẩm quyền. Các thao tác cho phép như truy vấn, tìm kiếm tài nguyên theo yêu cầu.
- Dịch vụ môi giới, lập lịch, xác định tài nguyên: Các dịch vụ này cho phép người dùng có thể yêu cầu việc phân bố tài nguyên tới các ứng dụng, lập lịch cho các ứng dụng trên các tài nguyên đã được chấp nhận.
- Dịch vụ theo dõi và chẩn đoán: Cho phép theo dõi các yêu cầu của người dùng, phát hiện các lỗi và có những biện pháp phục hồi cụ thể.
- Dịch vụ nhân bản dữ liệu: Cho phép quản lý các tài nguyên lưu trữ các bản sao dữ liệu, nâng cao hiệu năng truy cập dữ liệu theo các tiêu chí như thời gian đáp ứng, độ tin cậy, chi phí.
- Các hệ thống hỗ trợ lập trình trong môi trường lưới: Xây dựng một mô hình lập trình phù hợp với môi trường lưới, sử dụng các dịch vụ ở mức thấp như tìm kiếm tài nguyên, phân bố tài nguyên, cơ chế bảo mật, ….
- Các dịch vụ tìm kiếm dịch vụ: Tìm kiếm và lựa chọn các dịch vụ tốt nhất cũng như môi trường thực hiện dựa vào tham số của các ứng dụng cần thực hiện.
- Các dịch vụ công tác: Hỗ trợ việc trao đổi thông tin trong cộng đồng những người dùng, có thể đồng bộ hay không đồng bộ. Ví dụ như các dịch vụ CAVERNsoft, Access Grid, các hệ thống chia sẻ phần mềm theo nhóm.

2.2.2.5. Tầng ứng dụng (Application)

Đây là tầng trên cùng trong kiến trúc phân tầng tính toán lưới. Các ứng dụng lưới này được xây dựng trên cơ sở triệu gọi các hàm, các dịch vụ được cung cấp bởi các tầng phía dưới. Vì vậy, ở tầng này ta phải thiết kế và cài đặt các dịch vụ, hàm cụ thể cho các thao tác như quản lý tài nguyên, truy cập dữ liệu, tìm kiếm tài nguyên, … để sao cho người dùng lưới cảm thấy hoàn toàn trong suốt.

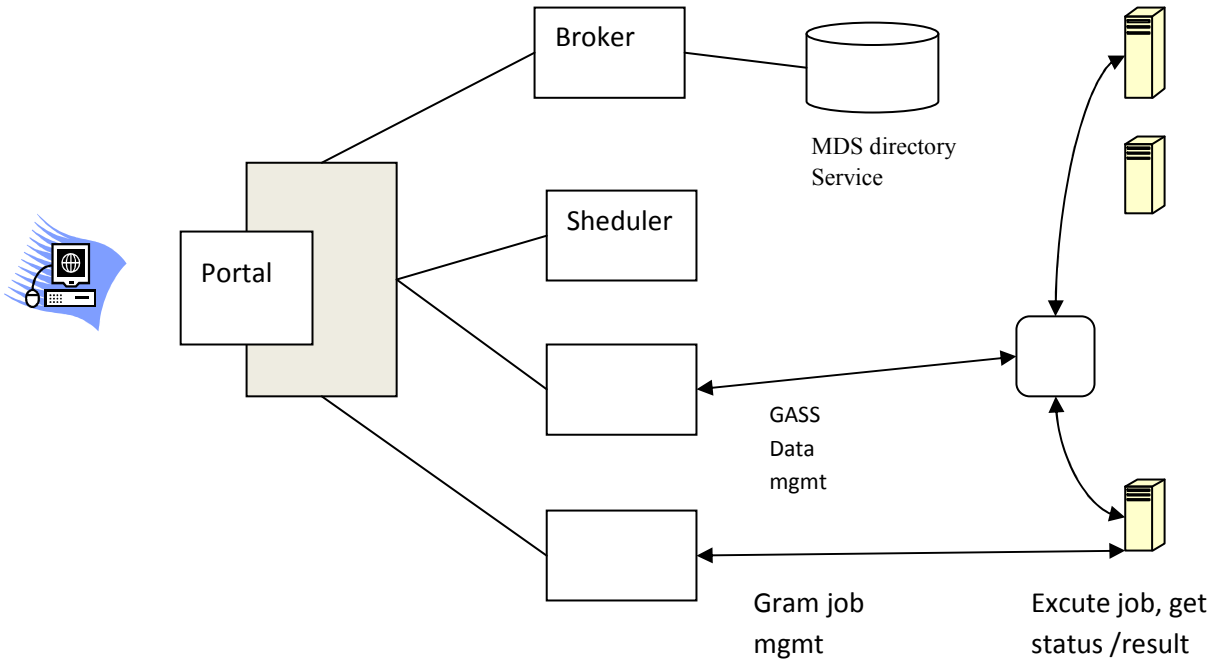
Người dùng yêu cầu chạy ứng dụng, nhận về kết quả mà không hề biết ứng dụng có được chạy ở đâu trên hệ thống lưới, sử dụng tài nguyên gì, ở đâu. Vì vậy, hệ thống lưới được coi như một máy tính ảo được kết hợp bởi nhiều tài nguyên khác nhau.

Như vậy, môi trường lưới hứa hẹn rất nhiều lợi thế không những cho người sử dụng mà còn cho cả các doanh nghiệp, tổ chức. Vấn đề cấp thiết đặt ra là cần phải xây dựng một nền tảng cho môi trường lưới hay nói cách khác là phải thiết kế cơ sở hạ tầng lưới, các thành phần và các dịch vụ cơ bản mà một lưới có thể cung cấp.

2. 3. CẤU TRÚC MỘT HỆ THỐNG LƯỚI

Hiện nay có nhiều giải pháp về phần mềm khác nhau để xây dựng một hệ thống lưới. Các giải pháp này tuy có nhiều điểm khác nhau về mặt kỹ thuật nhưng tương đối thống nhất về cấu trúc cơ bản của hệ thống. Điều này giúp cho việc hợp tác giữa các hệ thống lưới trong tương lai có thể trở nên dễ dàng hơn.

Khóa luận xin giới thiệu vắn tắt cấu trúc một hệ thống lưới ở mức tổng quan và cơ bản nhất, được đề xuất bởi IBM [4]. Hình 7 mô tả kiến trúc của một hệ thống lưới ở mức khái niệm, không quá đi sâu vào mặt chi tiết của từng thành phần:



Hình 9: Cấu trúc một hệ thống lưới do IBM đề xuất

Cổng giao tiếp (Portal): Cấu trúc hệ thống bên dưới được che giấu (trong suốt) đối với người dùng. Người dùng sẽ sử dụng hệ thống thông qua các ngõ vào (portal), các portal sẽ cung cấp cho người dùng các công cụ để thực thi ứng dụng, điều khiển hệ thống dưới các hình thức khác nhau như giao diện dòng lệnh, giao diện web…

Thành phần bảo mật (GSI – Grid Security System): Thành phần này sẽ kiểm tra, chứng thực thông tin người dùng cũng như kiểm tra quyền hạn của người dùng khi thao tác với hệ thống. Thành phần này còn chịu trách nhiệm mã hóa dữ liệu trao đổi giữa người dùng và hệ thống nếu có nhu cầu.

Thành phần lưu trữ thông tin hệ thống: MDS (Monitoring and Discovery Service): còn có thể được gọi là GIS (Grid Information Service) trong các tài liệu khác. Hệ thống lưới là một hệ thống biến động, các tài nguyên nằm phân tán, có thể tham gia hay rời bỏ hệ thống một cách thường xuyên. MDS là nơi giữ thông tin và luôn cập nhật những thay đổi về hệ thống.

Thành phần giao tiếp (Broker): Khi người dùng thực thi một ứng dụng, hệ thống cần xác định tài nguyên tính toán nào phù hợp nhất để thực thi ứng dụng đó. Vai trò này sẽ do broker đảm nhận. Broker sẽ giao tiếp với MDS để lấy những thông tin của hệ thống, dựa vào các giải thuật được cài đặt sẵn để lựa chọn tài nguyên phù hợp.

Bộ lập lịch (Scheduler): Sau khi đã xác định được tài nguyên đảm nhận ứng dụng, ứng dụng cần được điều phối lên tài nguyên để thực thi vào thời điểm thích hợp. Một tài nguyên vào một thời điểm có thể đảm nhận nhiều ứng dụng, quá trình điều phối trên tài nguyên sẽ do scheduler quyết định. Ở nhiều hệ thống, scheduler không được đặt ở mức chung toàn hệ thống mà ở mỗi tài nguyên, tài nguyên có chính sách riêng và tự quyết định hoạt động của mình.

GASS (Grid Access to Secondary Storage): Đảm trách nhiệm vụ vận chuyển ứng dụng và các dữ liệu cần thiết đến tài nguyên thực thi.

GRAM (Grid Resource Allocation Manager): Đảm nhận việc kích hoạt thực thi ứng dụng trên tài nguyên, kiểm tra trạng thái của ứng dụng (thất bại/thành công) và nhận kết quả trả về khi hoàn tất.

2. 4. LƯỚI HÓA ỨNG DỤNG

Để thiết kế một ứng dụng lưới hoàn chỉnh và đạt hiệu quả mong muốn, người thiết kế phải quan tâm tới nhiều vấn đề. Cần nhấn mạnh rằng, tất cả các vấn đề nêu trên là tổng quát và tùy thuộc vào loại quy mô và mức độ phức tạp của ứng dụng mà một số yếu tố có thể được xem nhẹ. Ví dụ như quá trình thiết kế ứng dụng phục vụ cho việc phân tích dữ liệu về thời tiết có thể tập trung vào cơ chế truyền thông, hiệu năng tính toán của các tài nguyên lưới và giao diện.

Do các tài nguyên xử lý là chuyên dụng và chỉ phục vụ cho một số lượng người dùng xác định nên vấn đề thông tin tài nguyên hay vấn đề lập lịch thực hiện không quá phức tạp. Ứng dụng càng có tính đa dạng, quy mô càng lớn và độ phức tạp cao thì quá trình phân tích thiết kế càng tốn nhiều công sức và chi phí. Chính vì vậy, cách tiếp cận lưới hóa các ứng dụng đã có để thích nghi với cơ sở hạ tầng lưới được xem là có chi phí nhỏ hơn và ít rủi ro hơn so với việc thiết kế hoàn toàn mới các ứng dụng đã có. Việc triển khai trên lưới BKGrid 2005 của ứng dụng khai phá dữ liệu Weka đi theo cách tiếp cận này. Tuy nhiên không phải ứng dụng "không lưới" nào cũng có thể được lưới hóa. Phần tiếp theo trình bày sáu bước để lưới hóa một ứng dụng.

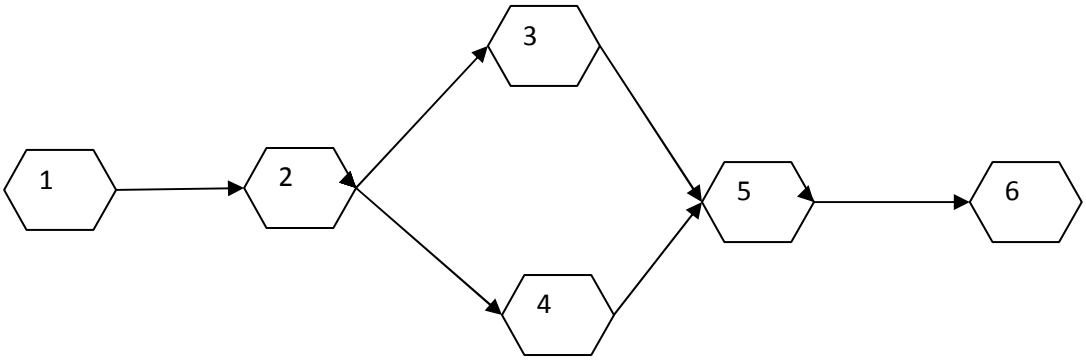
*Sáu bước lưới hóa ứng dụng

Một ứng dụng chỉ được coi là ứng dụng lưới khi nó tận dụng được các lợi thế do cơ sở hạ tầng tổ chức ảo đem lại để tăng tốc độ xử lý hay khả năng liên kết của lưới. Không chỉ có vậy, theo đặc tả OGSA thì ứng dụng lưới phải được chạy như là một dịch vụ Web trong môi trường lưới trong khi vẫn sử dụng được dịch vụ nền tảng được cung cấp bởi một cơ sở hạ tầng lưới. Ví vụ như nền tảng J2EE (Java 2 Enterprise Edition) chạy như một dịch vụ web trên nền phần đệm hỗ trợ lưới (grid enable middleware) WebSphere Application Server (WSAS) và sử dụng các dịch vụ được cung cấp bởi WSAS và cơ sở hạ tầng lưới.

Bước đầu tiên của quá trình lưới hóa là khảo sát quá trình thực hiện đơn giản của một ứng dụng trong một lưới. Sau đó là tìm hiểu cách các ứng dụng cần lưới hóa sử dụng hai bước đầu tiên để có thể thực thi như là các công việc theo lô đơn lẻ và dịch vụ hóa ứng dụng đó để phục vụ người sử dụng thông qua phần đệm.

Hai bước cuối cùng sẽ đưa ra một số đẳng thức khả thi để triển khai một ứng dụng sử dụng dịch vụ lưới theo cách song song. Mặc dù các bước được đề cập dưới đây là cần thiết trong quá trình lưới hóa ứng dụng nhưng không bắt buộc phải thực hiện đủ cả năm bước. Nhiều ứng dụng chỉ đạt được bước 5 và rất ít có thể đạt được bước 6. Bước 6 được chuyên biệt hóa cho một số ứng dụng được thiết kế từ ban đầu, để hỗ trợ cho mô hình xử lý song song kết nối chặt (tightly coupled) cho các chương trình mà các chương trình này tạo nên ứng dụng hoàn chỉnh.

Ngoài ra, ứng dụng không cần thiết phải thực hiện các bước một cách tuần tự, từng bước một, ví dụ như giữa bước 2: xứ lý theo lô đồng thời và bước 5 là dịch vụ song song có thể thực hiện một trong hai bước 3: xử lý theo lô song song hay bước 4: dịch vụ.



Hình 10: Mô hình lưới hóa ứng dụng

- Bước 1: Xử lý công việc theo lô
- Bước 2: Xử lý theo lô đồng thời
- Bước 3: Xử lý theo lô song song
- Bước 4: Dịch vụ
- Bước 5: Dịch vụ song song.
- Bước 6: Chương trình song song phụ thuộc chặt.

Các bước có thể được phân chia thành các giai đoạn như sau:

- Giai đoạn thực hiện:

  Bước 1, bước 2 và dạng thức đơn giản nhất của bước 3 với mục đích tập trung vào khả năng của một ứng dụng có thể được chạy trên lưới.

- Giai đoạn thích ứng:

  Dạng thức phức tạp hơn của bước 3, bước 4, bước 5, nhằm đáp ứng các chức năng và giá trị của một ứng dụng bằng cách lưới hóa ứng dụng với điều kiện không có nhiều thay đổi phần đệm của lưới. Ứng dụng tương tự có thể được cấu trúc để chạy trên môi trường phi lưới. Ứng dụng sau khi đã đạt đến giai đoạn này đã hội đủ các điều kiện để có thể trở thành một ứng dụng lưới.

- Giai đoạn khai thác:

  Ứng dụng đạt được bước 6 sẽ khai thác lưới hay cơ sở hạ tầng các cụm máy tính cho mục đích là thực hiện trên lưới. Các ứng dụng qua sáu bước không thể hoàn thành một cách chính xác và thành công nếu như không được chạy trên lưới.

Các bước có liên quan đến nhau theo nghĩa là bước sau được phát triển trên cở sở bước trước nó. Ví dụ: một ứng dụng hiện thời có thể được lưới hóa theo cách mà một đơn vị tính toán hướng tài nguyên của nó được chạy như một công việc theo lô trên bất cứ một tài nguyên tính toán nào trong lưới (bước 1). Lưới đó sẽ quyết định việc thực thi công việc vào thời gian chạy ứng dụng. Sau đó, bằng việc loại bỏ các tác nhân cản trở việc thực thi đồng thời, ứng dụng đó được coi như là một tập bao gồm các thể hiện công việc theo lô độc lập nhau được chạy đồng thời trên các node lưới (bước 2). Cách này làm cho toàn bộ ứng dụng chạy nhanh hơn và hiệu quả hơn, xử lý được nhiều đầu vào hơn là chỉ chạy duy nhất một thể hiện vào một thời điểm.

Ứng dụng có thể thực hiện tiếp thông qua một trong hai cách sau:

+ Ứng dụng có thể được thiết kế lại theo cách là phần công việc đã được lưới hóa không chạy như là tập các công việc song song mà được thực hiện như là một dịch vụ (bước 4).
+ Cuối cùng các thành phần của ứng dụng có thể được thiết kế lại ở mức cao hơn như là các dịch vụ có trạng thái, được gọi từ xa và thực hiện một cách song song (bước năm). Đối với một ứng dụng bình thường tức là ứng dụng được thiết kế để thực hiện trên các máy đơn, bước 5 là đủ cho quá trình lưới hóa. Bước 6 thông thường không đạt được cho các ứng dụng sẵn có bởi chi phí để thiết kế lại một ứng dụng có thể cao hơn những lợi ích mà ta có thể thu lại. Do đó, tốt hơn là để có một ứng dụng lưới hoàn chỉnh ta thiết kế lại các chương trình song song phụ thuộc chặt ngay từ lúc xây dựng ứng dụng.

Chương3. ÁP DỤNG TÍNH TOÁN LƯỚI GIẢI BÀI TOÁN TRONG AN TOÀN THÔNG TIN

3.1. BÀI TOÁN TÌM SỐ NGUYÊN TỐ MERSENNE

3. 1.1.Số nguyên tố và số hoàn thiện

3. 1. 1.1. Khái niệm số nguyên tố

1/. Định nghĩa

Số nguyên tố là số tự nhiên lớn hơn 1 và chỉ có hai ước số là 1 và chính nó. Số 2 là số nguyên tố đầu tiên và cũng là số nguyên tố chẵn duy nhất.

2/. Tính chất

- Ước tự nhiên nhỏ nhất khác một của một số tự nhiên là một số nguyên tố.
- Cho a là số tự nhiên, p là số nguyên tố. Nếu UCLN(a, p) = p thì a chia hết cho p. Nếu UCLN(a, p) = 1 thì a không chia hết cho p
- Một số tự nhiên bất kỳ không phải là số nguyên tố đều được phân tích thành tích của các số nguyên tố.
- Không tồn tại số nguyên tố lớn nhất.

3/. Ý nghĩa của số nguyên tố

Trong thời kỳ cổ đại thì số nguyên tố cũng đã thu hút rất nhiều sự quan tâm của các nhà toán học. Ban đầu người ta tập trung nghiên cứu số nguyên tố cho các lĩnh vực thần bí và số học.

Ngày nay khi công nghệ thông tin phát triển thì số nguyên tố lại đóng một vai trò rất quan trọng trong lĩnh vực này, đặc biệt là trong các hệ mã hóa dữ liệu, bảo mật thông tin. Ta có thể nhận thấy rằng trong các hệ mã hóa dữ liệu hiện đại đều sử dụng tính chất của số nguyên tố, và đặc biệt là cần sử dụng một hay nhiều số nguyên tố lớn để làm công cụ mã hóa dữ liệu.

3. 1. 1. 2. Kiểm tra một số nguyên tố

1/. Phương pháp đơn giản

Phương pháp đơn giản nhất để kiểm tra một số có phải là số nguyên tố không là ta xem số đó có chia hết cho một số tự nhiên trong khoảng từ 0 -$\sqrt{n}$. Phương pháp này có thể được mở rộng hơn thành phương pháp sàng lọc Eratosthenes.

2/. Phương pháp xác suất

Phép kiểm tra tính nguyên tố hay dùng nhất là thuật toán ngẫu nhiên. Giả sử có mệnh đề Q(p, a) nào đó đúng với mọi số nguyên tố *p* và một số tự nhiên *a* <= *p*. Nếu *n* là một số tự nhiên lẻ và mệnh đề Q(n, a) đúng với *a*<= *n* được lấy ngẫu nhiên, khi đó *a có khả năng* là số nguyên tố. Ta đưa ra một thuật toán, kết luận rằng *n* là số nguyên tố. Nó là thuật toán ngẫu nhiên hay thuật toán xác suất. Trong thuật toán loại này, dùng một kiểm tra ngẫu nhiên không bao giờ kết luận một số nguyên tố là hợp số nhưng có thể kết luận một hợp số là số nguyên tố.

Xác suất sai của phép kiểm tra có thể giảm xuống nhờ việc chọn một dãy độc lập các số *a*; nếu với mỗi số *a* xác suất để thuật toán kết luận một hợp số là số nguyên tố là nhỏ hơn một nửa thì sau *k* lần thử độc lập, xác suất sai là nhỏ hơn 2^{-k}, độ tin cậy của thuật toán sẽ tăng lên theo *k*.

Cấu trúc cơ bản của một phép kiểm tra ngẫu nhiên:

a) Chọn một số ngẫu nhiên *a*.

b) Kiểm tra một hệ thức nào đó giữa số *a* và số *n* đã cho. Nếu hệ thức sai thì chắc chắn *n* là một hợp số (số *a* là "bằng chứng" chứng tỏ *n* là hợp số) và dừng thuật toán.

c) Lặp lại bước a cho đến khi đạt được số lần đã định hoặc gặp bước b.

Sau một số lần kiểm tra, nếu không tìm được bằng chứng chứng tỏ *n* là hợp số thì ta kết luận *n* là số nguyên tố.

*Các phép kiểm tra tính nguyên tố ngẫu nhiên:

Phép kiểm tra tính nguyên tố của Fermat (Kiểm tra Fetmar. Đây là phép thử heuristic; tuy nhiên ít người sử dung phép thử này

Được sử dụng nhiều hơn là Kiểm tra Miller-Rabin và Kiểm tra Solovay-Strassen. Với mỗi hợp số n, ít nhất 3/4 (với kiểm tra Miller-Rabin) hoặc 1/2 (Với kiểm tra Solovay-Strassen) các số a là bằng chứng chứng tỏ n là hợp số).

3/. Phương pháp tất định

Vào năm 2002, Manindra Agrawal, Nitin Saxena và Neeraj Kayal đề xuất một giải thuật tất định kiểm tra tính nguyên tố, là kiểm tra AKS, có khả năng chạy trong $O((\log n)^{12})$. Trên thực tế thuật toán này chạy chậm hơn các phương pháp xác suất.

5/. Các phương pháp lý thuyết số

Có một vài phương pháp khác trong lý thuyết số để kiểm tra tính nguyên tố như kiểm tra Lucas-Lehmer và kiểm tra Proth. Chúng thường dựa vào việc phân tích $n + 1$, $n - 1$, hoặc những số khác. Tuy nhiên các phương pháp này không dùng cho các số tự nhiên n bất kỳ, mà chỉ cho các số có dạng đặc biệt nào đó Kiểm tra Lucas-Lehmer dựa trên tính chất: bậc(*multiplicative order*) của một số a modulo n là $n - 1$ với n là số nguyên tố và a là căn nguyên thuỷ (*primitive root*) modulo n. Nếu ta có thể biểu diễn a chỉ theo n,có thể thấy n là nguyên tố.

3. 1.1.3. Số nguyên tố Mersenne

Trước đây, nhiều người từng nghĩ 2^n-1 luôn là số nguyên tố cho mọi n nguyên tố, nhưng năm 1563 **Hudalricus Regius** đã chỉ ra rằng 2^{11}–1 = 2047 = 23*89 không phải là số nguyên tố.

Năm 1603 Pietro Cataldi đã kiểm chứng 1 cách chính xác rằng khi n=17, 19 thì 2^n-1 là số nguyên tố và dự đoán điều đó cũng đúng khi n=23;29;31;37 . Tuy nhiên vào năm 1640 Fermat đã chỉ ra suy đoán của Cataldi sai với trường hợp 23 và 37 rồi đến năm 1738 Euler cũng chỉ ra trường hợp n=29 là sai.

Năm 1644 một nhà toán học kiêm giáo sĩ người Pháp là **Marin Mersenne** (1588-1648) trong lời tựa của cuốn **Cogitata Physica-Mathematica** (1644) (Những khái niệm Toán học và Vật lí) đã sắp xếp ra 11 trị số của n để (2^n-1) là số nguyên tố, đó là các trị số: 2, 3, 5, 7, 13, 17, 19, 31, 67, 127 và 257.

Không lâu sau, có người còn chứng minh được nếu 2^n-1 là số nguyên tố thì n nhất định là số nguyên tố, nhưng điều ngược lại không đúng: nghĩa là khi n là số nguyên tố thì 2^n-1 không nhất định là số nguyên tố. Thí dụ như trong các trị số của n mà Mersenne đưa ra với n = 67, 257 thì 2^n-1 không phải là số nguyên tố. Trong quá trình tìm kiếm người ta cũng phát hiện ra rằng trong dãy trị số mà Mersenne đưa ra thì còn thiếu n = 61, 89, 107.

Để tưởng nhớ đến công lao của vị giáo sĩ này, người ta đã đặt tên ông cho loại số nguyên tố mà ông là người đã tìm ra những số đầu tiên.

Định nghĩa:

Số nguyên tố Mersenne là số nguyên tố có dạng 2^n-1*, trong đó n là số nguyên tố (và không có chiều ngược lại, nghĩa là n là số nguyên tố thì chưa chắc* 2^n-1 *đã là số nguyên tố)*

Hiện nay các số nguyên tố lớn nhất được tìm thấy phần lớn đều là các số nguyên tố Mersenne.

Danh sách các số nguyên tố Mersenne đã được tìm thấy

#	n	M_n	Số chữ số trong M_n	Ngày tìm được	Người tìm
1	2	3	1	*cổ đại*	*Hy Lạp cổ đại*
2	3	7	1	*cổ đại*	*Hy Lạp cổ đại*
3	5	31	2	*cổ đại*	*Hy Lạp cổ đại*
4	7	127	3	*cổ đại*	*Hy Lạp cổ đại*
5	13	8191	4	1456	*Khuyết danh*
6	17	131071	6	1588	Cataldi
7	19	524287	6	1588	Cataldi
8	31	2147483647	10	1750	Euler
9	61	2305843009213693951	19	1883	Pervushin
10	89	618970019...449562111	27	1911	Powers
11	107	162259276...010288127	33	1914	Powers
12	127	170141183...884105727	39	1876	Lucas
13	521	686479766...115057151	157	30 tháng 1 năm 1952	Robinson
14	607	531137992...031728127	183	30 tháng 1 năm 1952	Robinson
15	1 279	104079321...168729087	386	25 tháng 6 năm 1952	Robinson
16	2 203	147597991...697771007	664	7 tháng 10 năm 1952	Robinson
17	2 281	446087557...132836351	687	9 tháng 10 năm 1952	Robinson
18	3 217	259117086...909315071	969	8 tháng 9 năm 1957	Riesel
19	4 253	190797007...350484991	1 281	3 tháng 11 năm 1961	Hurwitz
20	4 423	285542542...608580607	1 332	3 tháng 11 năm 1961	Hurwitz
21	9 689	478220278...225754111	2 917	11 tháng 5 năm 1963	Gillies
22	9 941	346088282...789463551	2 993	16 tháng 5 năm 1963	Gillies
23	11 213	281411201...696392191	3 376	2 tháng 6 năm 1963	Gillies
24	19 937	431542479...968041471	6 002	4 tháng 3 năm 1971	Tuckerman
25	21 701	448679166...511882751	6 533	30 tháng 10 năm 1978	Noll & Nickel
26	23 209	402874115...779264511	6 987	9 tháng 2 năm 1979	Noll

27	44 497	854509824...011228671	13 395	8 tháng 4 năm 1979	Nelson & Slowinski
28	86 243	536927995...433438207	25 962	25 tháng 9 năm 1982	Slowinski
29	110 503	521928313...465515007	33 265	28 tháng 1 năm 1988	Colquitt & Welsh
30	132 049	512740276...730061311	39 751	20 tháng 9 năm 1983	Slowinski
31	216 091	746093103...815528447	65 050	6 tháng 9 năm 1985	Slowinski
32	756 839	174135906...544677887	227 832	19 tháng 2 năm 1992	Slowinski & Gage trên Cray-2 tại Phòng thí nghiệm Harwell [1]
33	859 433	129498125...500142591	258 716	10 tháng 1 năm 1994	Slowinski & Gage
34	1 257 787	412245773...089366527	378 632	3 tháng 9 năm 1996	Slowinski & Gage [2]
35	1 398 269	814717564...451315711	420 921	13 tháng 11 năm 1996	GIMPS / Joel Armengaud [3]
36	2 976 221	623340076...729201151	895 932	24 tháng 8 năm 1997	GIMPS / Gordon Spence [4]
37	3 021 377	127411683...024694271	909 526	27 tháng 1 năm 1998	GIMPS / Roland Clarkson [5]
38	6 972 593	437075744...924193791	2 098 960	1 tháng 6 năm 1999	GIMPS / Nayan Hajratwala [6]
39	13 466 917	924947738...256259071	4 053 946	14 tháng 11 năm 2001	GIMPS / Michael Cameron [7]
40*	20 996 011	125976895...855682047	6 320 430	17 tháng 11 năm 2003	GIMPS / Michael Shafer [8]
41*	24 036 583	299410429...733969407	7 235 733	15 tháng 5 năm 2004	GIMPS / Josh Findley [9]
42*	25 964 951	122164630...577077247	7 816 230	18 tháng 2 năm 2005	GIMPS / Martin Nowak [10]
43*	30 402 457	315416475...652943871	9 152 052	15 tháng 12 năm 2005	GIMPS / Curtis Cooper & Steven Boone [11]
44*	32 582 657	124575026...053967871	9 808 358	4 tháng 9 năm 2006	GIMPS / Curtis Cooper & Steven Boone [12]
45*	37 156 667	202254406...308220927	11 185 272	6 tháng 9 năm 2008	GIMPS / Hans-Michael Elvenich
46*	43 112 609	316470269...697152511	12 978 189	23 tháng 8 năm 2008	GIMPS / Edson Smith

3. 1. 1.4. Số hoàn thiện

Nhiều nền văn hóa cổ đại có liên quan đến mối quan hệ vừa một số với tổng của các ước số của nó, thường đưa ra các cách lý giải huyền bí. Ở đây chúng ta chỉ tập trung vào một điều đó là mối quan hệ.

Định nghĩa:

Một số nguyên dương n được gọi là số hoàn thiện (perfect number) nếu nó bằng tổng của tất cả các ước dương cộng lại (trừ ước số bằng chính nó).

Ví dụ: 6 là số hoàn thiện vì 6=1+2+3.

Tương tự các số 28, 496, 8218, …cũng là số hoàn thiện.

Hãy nhìn vào các số trên 6=2*3, 28=4*7, 496= 16*31, 8218 = 64*127, chúng ta có thể nhận ra rằng các số trên đều có dạng $2^{n-1}(2^n-1)$ với n =2, 3, 5, 7, …

Trong mỗi trường hợp thì 2^n-1 đều là số nguyên tố Mersenne. Vì vậy có thể dễ dàng có được định lý dưới đây:

Định lý 1

k là số hoàn thiện chẵn nếu và chỉ nếu nó có dạng $2^{n-1}(2^n-1)$ và (2^n-1) là số nguyên tố.

Định lý 2

Nếu (2^n-1) là số nguyên tố thì n cũng là số nguyên tố.

Như vậy việc tìm kiếm số nguyên tố Mersenne cũng là việc tìm kiếm số hoàn thiện chẵn.

Cũng nên lưu ý rằng các số hoàn thiện được liệt kê ở trên đều có số hàng đơn vị là 6, 8. Điều này có thể chứng minh một cách dễ dàng.

3. 1.1.5. Phương pháp kiểm tra số nguyên tố Lucas-Lehmer

Các số nguyên tố Mersenne (cũng như đối với các số hoàn thiện chẵn) được tìm bằng cách sử dụng định lý sau:

Định lý Lucas-Lehmer :

Đối với p là số nguyên tố lẻ, số Mersenne $2^p - 1$ là một số nguyên tố nếu và chỉ nếu $2^p - 1$ chia hết cho $S_{(p-1)}$. trong đó $S_{(n+1)} = S_n^2 - 2$ và $S_1 = 4$.

Chứng minh : tham khảo tại địa chỉ:

http://primes. utm. edu/notes/proofs/LucasLehmer. html

Đoạn giả mã với cách kiểm tra trên là:

Function():

Begin

S:=4 ;

For (i from 3 to p do s:=s^2 -2 mod 2^p -1 ;

If s==0 then

2^p – 1 is prime.

Else

2^p – 1 is composite

End ;

Cơ sở lý thuyết đối với phương pháp kiểm tra trên được đề xướng bởi Lucas vào những năm cuối 1870, và sau đó vào những năm 1930, Lehmer chuyển nó thành một phương pháp kiểm tra đơn giản.

Trong những năm 1810, Perter Barlow xuất bản cuốn sách về lý thuyết số khi đó số 2^{30} (2^{31}-1) là số hoàn thiện lớn nhất tìm ra được. Vào cuối những năm 1800, đã có ý tưởng về một siêu máy tính.

Sau khi tìm ra số nguyên tố Mersenne thứ 23 (tìm ra tại đại học Illionis), khoa Toán đã tự hào với vị chủ nhiệm của họ, tiến sĩ Batermam, số nguyên tố do ông tìm ra đã được phát hành tem.

Số nguyên tố Mersenne thứ 25 và 26 được tìm ra bởi các học sinh trường trung học Laura Nickel và Landon Curt Noll, những hiểu biết rất ít về toán học, đã sử dụng phương pháp kiểm tra của Lucas trên máy tính cục bộ của trường đại học. Quá trình tìm ra số nguyên tố đầu tiên đã được làm thành một chương trình truyền hình trên đài truyền hình quốc gia Mỹ và được đăng trên trang nhất của tờ NewYork Time.

Slowinski, làm việc cho công ty máy tính Cray đã viết một phiên bản của phương pháp kiểm tra Lucas và đã thuyết phục rất nhiều phòng thí nghiệm Cray trên khắp thế giới chạy phần mềm của mình. Ông đã trì hoãn việc thông báo số nguyên tố mà ông ta tìm ra tới tận khi ông được nhận chứng nhận bản quyền về việc tìm ra nó. Trong bảng các số nguyên tố Mersenne ta thấy được rằng có số thứ 30 và 1 nhưng lại thiếu số 29. Colquitt và Welsh đã làm việc cùng nhau để tìm ra số nguyên tố thứ 29.

George Woltman, một nhà tổ chức và lập trình viên xuất sắc. Ông ta bắt đầu quá trình tìm kiếm vào cuối năm 1995, ông ta đã tập hợp lại cơ sở dữ liệu riêng lẻ và kết hợp chúng thành một cơ sở dữ liệu chung nhất. Sau đó ông ta đặt CSDL, một chương trình tối ưu cao, miễn phí để tìm số nguyên tố trên Internet. Đây là điểm khởi đầu của GIMPS (the Great Internet Mersenne Prime Search – cộng đồng tìm kiếm số nguyên tố lớn Mersenne), nơi đã tìm số nguyên tố lớn nhất được biết đến, tại đó cũng đã kiểm tra lại tất cả các vùng chưa được khám phá giữ các số nguyên tố trước đó, kết hợp kết quả của hàng tá các chuyên gia và hàng nghìn người không chuyên và giới GIMPS đã đưa ra một phần mềm miễn phí có khả năng chạy được trên mọi nền.

Cuối năm 1997, Scott Lurowski (và những người khác) thiết lập lên mạng PrimeNet để tự động chọn lựa các thứ hạng và báo cáo kết quả tìm kiếm đối với GIMPS, hiện tại ai cũng có thể tham gia nghiên cứu này.

3.1.2. Áp dụng tính toán lưới tìm số nguyên tố Mersenne

3.1.2.1. Thuật toán kiểm tra số nguyên tố Mersenne

Các số Mersenne có dạng rất đơn giản $2^p - 1$, trong đó p là số mũ và sẽ được kiểm tra. Có thể dễ dang chứng minh rằng nếu $2^p - 1$ là số nguyên tố thì p là số nguyên tố.

a) Lập danh sách các dạng và các số mũ: Bước đầu tiên trong việc tìm kiếm số nguyên tố Mersenne là tạo một danh sách các số mũ p để kiểm tra.

b) Thử phân tích thừa số: Bước tiếp theo là loại trừ một số mũ bằng cách tìm ra các thừa số nhỏ hơn. Có rất nhiều thuật toán hiệu quả để chỉ ra rằng một số có chia hết cho $2^p - 1$ hay không.

   Đoạn code sinh ra các hệ số mà chương trình PrimeNet cung cấp tạo ra một sàng Eratosthenes với mỗi bit biểu diễn hệ số mũ 2kp + 1. Sàng này sẽ thực hiện loại trừ hệ số nào mà chia hết cho các danh sách 40000 số nguyên tố đầu tiên. Vì vậy các bit biểu diễn các số 3 hoặc 5 mod 8 đều bị xóa. Quá trình này sẽ loại bỏ 95% các thừa số mũ. Các hệ số mũ còn lại sẽ được kiểm tra bằng cách sử dụng thuật toán ở trên.

c) Kiểm tra Lucas –Lehmer nguyên thủy:

   Các phương pháp sử dụng ở trên là bước đầu tiên để thử tìm các ước số đối với số Mersenne, vì vậy loại trừ các mũ nguyên tố p từ danh sách kiểm tra trước khi thực hiện việc kiểm tra Lucas – Lehmer.

   Phương pháp kiểm tra Lucas – Lehmer nguyên thủy tương đối đơn giản:

   Với $p > 1$, số $(2^p - 1)$ là số nguyên tố nếu và chỉ nếu $S_{p-2} = 0$ trong dãy

   $S_0 = 4$, $S_N = S_{N-1}^2 - 2 \bmod (2^p - 1)$.

Ví dụ để chứng minh 2^7 -1 là số nguyên tố:

$S_0 = 4$; $S_1 = (4*4 – 2)$ mod 127 = 14

$S_2 = (14*14 – 2)$ mod 127 = 67

$S_3 = (67*67 – 2)$ mod 127 = 42

$S_4 = (42*42 – 2)$mod 127 = 111

$S_5 = (111*111 – 2)$ mod 127 = 0

Để cài đặt phương pháp Lucas – Lehmer một cách hiệu quả, cần phải tìm ra nhanh chóng bình phương một số lớn sau đó chia lấy phần dư cho $(2^p – 1)$. Bởi vậy cuối những năm 1960, thuật toán nhanh nhất để bình phương một số lớn là chia số lớn thành các phần nhỏ dưới dạng một mảng lớn, sau đó thực hiện biến đổi nhanh Fourier và biến đổi nghịch đảo.

3.1.2.2. Hướng dẫn sử dụng phần mềm PrimeNet v5. 0 và tham gia cộng đồng PrimeNet

Hệ thống kiểm tra số nguyên tố Mersenne GIMPS cho phép người sử dụng trên khắp thế giới đăng kí tham gia vào lưới tính toán kiểm tra số nguyên tố Mersenne trực tiếp trên Internet.

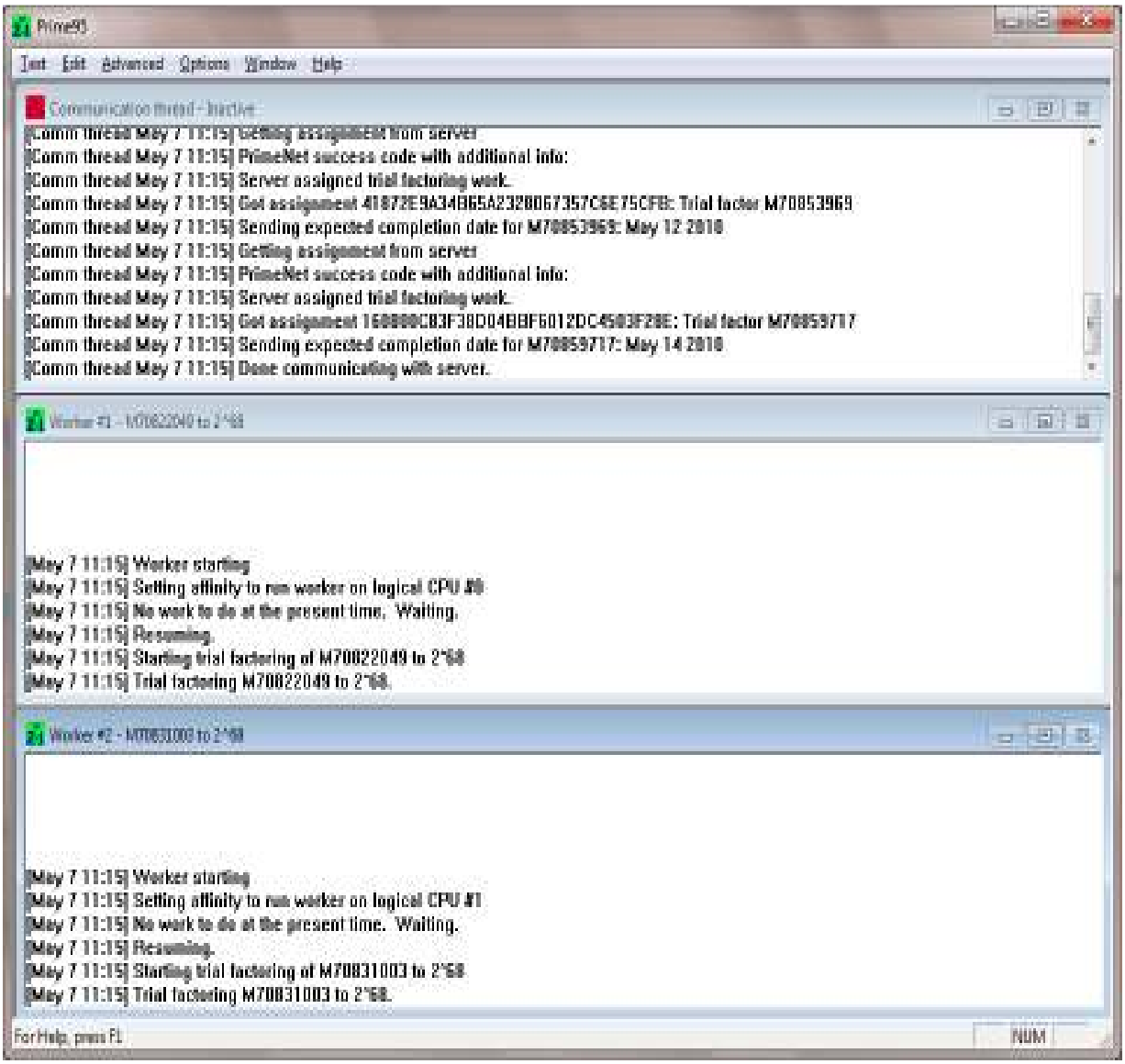
Hệ thống lưới bao gồm các nhóm người dùng, họ có quyền đăng ký vào bất kỳ nhóm nào.

Để tham gia hệ thống, yêu cầu với máy tính của người tham gia cần:

- Một máy tính tương đối hiện đại
- Máy tính có giờ rỗi
- Cần thời gian và tính kiên nhẫn
- Có kết nối internet

Các bước thực hiện:

- Đăng kí một tài khoản tại website http://www. mersenne. org
- Tải và cài đặt phần mềm PrimeNet
- Chạy phần mềm trên và nhập thông tin tài khoản đã đăng kí



Hình 11: Giao diện chạy chương trình PrimNET

Khi tham gia cộng đồng PrimeNet, người sử dụng đã trực tiếp tham gia lưới tính toán, kiểm tra số nguyên tố Mersenne, cũng là góp một phần vào việc tìm ra những số nguyên tố ngày một lớn hơn đóng góp cho khoa học công nghệ đặc biệt là toán học và công nghệ thông tin.

3.1.2.3. Lợi ích của tính toán lưới trong việc tìm số nguyên tố Mersenne

Ngày nay thì số nguyên tố đóng một vai trò quan trọng trong lĩnh vực công nghệ thông tin, đặc biệt là trong ngành an toàn và bảo mật thông tin. Có một điều mà những người làm trong lĩnh vực này nhận thấy đó là nhu cầu ngày càng cao về giá trị của các số nguyên tố lớn vì có một số hệ mã hóa mà độ an toàn của nó phụ thuộc vào độ lớn của số nguyên tố được dùng để tạo khóa, có những giải thưởng rất lớn dành cho những ai tìm được ra các số nguyên tố lớn. Ví dụ như có những giải thưởng trị giá đến 100.000 USD cho ai tìm được ra số nguyên tố lớn có 10 triệu chữ số đầu tiên.

Việc kiểm tra một số nguyên tố có giá trị nhỏ thì ta cảm thấy khá đơn giản, có thể dùng các phương pháp đơn giản như kiểm tra xem số đó có chia hết cho các số trong khoảng từ 2 đến căn bậc 2 của nó không ? Nhưng công việc này sẽ cực kỳ khó khăn khi ta phải kiểm tra một số lớn, thậm chí đến hàng chục triệu chữ số. Dù đã có nhiều thuật toán để cải thiện và giảm độ phức tạp của bài toán kiểm tra số nguyên tố nhưng để kiểm tra những số lớn đến hàng chục triệu chữ số sẽ rất mất nhiều thời gian nếu như dùng trên máy tính cá nhân.

Với phần mềm PrimeNet v5.0, nó sẽ sử dụng những lợi ích rất lớn của công nghệ tính toán lưới như tận dụng các CPU nhàn rỗi để tăng tốc độ tính toán , và đặc biệt là khả năng tính toán song song trong tính toán lưới. Với các máy tính thông thường thì nó chỉ chạy được các chương trình theo kiểu tuần tự vì vậy mà tốc độ sẽ không thể cao được cho dù với các bộ vi xử lý rất mạnh như hiện nay. Đối với tính toán lưới, nó sẽ phân tán việc kiểm tra số lớn đó thành các thành phần nhỏ và chia ra cho các node xử lý, vì vậy sẽ tăng tốc độ tính toán và giảm thời gian giải quyết bài toán.

3.2. ỨNG DỤNG GRID COMPUTING TRONG HỆ THỐNG PHÁT HIỆN XÂM NHẬP

3.2.1. Giới thiệu

Cùng với sự phát triển ngày càng phổ biến của mạng Ad Hoc, yêu cầu về bảo mật cho các host di động cũng được quan tâm nhiều hơn. Đặc tính dễ bị tổn xâm phạm của các mạng di động không có cơ sở hạ tầng cố định khiến cho khả năng ngăn chặn những truy nhập bất hợp pháp trở nên rất yếu kém. Phần này sẽ tập trung giới thiệu về hệ thống phát hiện xâm nhập dựa trên cơ sở tính toán lưới (Grid based Intrusion Detection System (G-IDS)). Khóa luận này sẽ trình bày một kiến trúc mới sử dụng những nguyên lý cơ bản của Grid Computing và áp dụng chúng vào các cơ chế phát hiện xâm nhập, nhằm bảo vệ các hệ thống mạng. Kiến trúc này được phát triển bởi các tác giả Pasquale Donadio, Antonio Cimmino và Giorgio Ventre.

3.2.2. Phân tích bài toán và hướng giải quyết

Các mạng không dây Ad Hoc bị giới hạn sử dụng bởi một cơ sở hạ tầng cố định nằm bên dưới, nhiều chức năng mạng được tích hợp trong các node di động có thể giúp tạo nên một mạng sở hữu riêng. Đặc tính không có chức năng quản lý mạng tập trung của mạng Ad Hoc khiến cho nó trở nên dễ bị tổn thương đối với những tấn công hiểm độc. Rất khó khăn để có thể đạt tới sự bảo mật tuyệt đối trong một mạng Ad Hoc không dây, bởi những hạn chế trong bảo vệ vật lý của mỗi node, các kết nối tự nhiên rời rạc, thiếu đơn vị quản lý và giám sát tập trung.

Hệ thống G-IDS hứa hẹn sẽ xử lý các vấn đề bảo mật quan trọng của một hệ thống mạng, đó là: phân tích lưu lượng thông tin thời gian thực và các hành động ứng xử. Tính toán lưu lượng thông tin là chìa khóa của an toàn mạng, nó không chứa các cảnh báo IDS nhưng có chứa các thông điệp từ những node kế cận, các ứng dụng và các thiết bị mạng khác. Việc phân tích lưu lượng thông tin mỗi ngày sẽ giúp phát hiện dấu vết các cuộc tấn công hiểm độc.

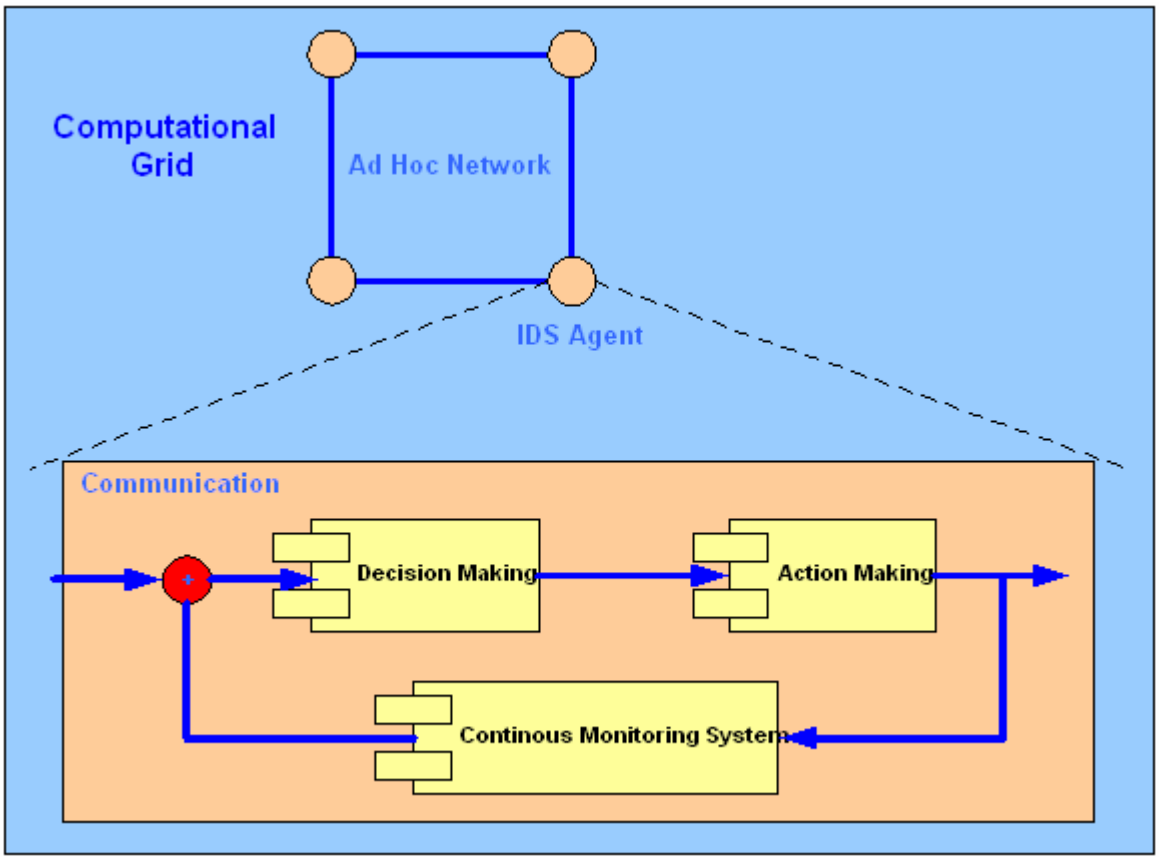
Ý tưởng của hệ thống là định nghĩa một bộ phân tích lưu lượng thông tin phân tán, thực hiện chia sẻ các kết quả phản hồi giữa các node kế cận trong mạng. Bộ phân tích này sẽ sử dụng các nguyên lý cơ bản của Grid Computing và áp dụng chúng trong hệ thống phát hiện xâm nhập.

3.2.3. Giải pháp Based IDS cho mạng AD HOC

Đặc điểm chính của G-IDS là nó có thể tập trung hoặc không tập trung. Một G-IDS tập trung là sự kết hợp thông thường giữa các cảm biến riêng lẻ. Các cảm biến này tập hợp và chuyển tất cả các dữ liệu tới hệ thống quản lý trung tâm, nơi dữ liệu IDS của mạng Ad Hoc được lưu trữ và xử lý. G-IDS không tập trung thường có thêm một hoặc nhiều hơn các thiết bị hợp tác để thực hiện chức năng báo cáo quá trình tập hợp và xử lý dữ liệu trên G-IDS. Ý tưởng đằng sau mục tiêu của G-IDS rất đơn giản: để định nghĩa một bộ phân tích lưu lượng thông tin phân tán, cần phải tính toán các nguồn tài nguyên và ghi lọc các thông điệp thực thi có liên quan đến các hành động an toàn trong thời gian thực. Các hệ thống phát hiện và đáp lại những xâm nhập có thể vừa phân tán vừa hợp tác để phù hợp với những yêu cầu của mạng không dây di động.

Trong kiến trúc được đề xướng, mọi node trong mạng Ad Hoc di động đều tham gia vào việc phát hiện và hồi đáp xâm nhập. Mỗi node chịu trách nhiệm phát hiện các dấu hiệu xâm nhập cục bộ và độc lập, nhưng các node kế cận có thể cộng tác điều tra trong một phạm vi rộng. Các G-IDS Agent riêng lẻ chạy độc lập và giám sát các hoạt động cục bộ. Nó phát hiện xâm nhập từ những vết tích cục bộ và khởi tạo các phản hồi. Nếu sự dị thường được phát hiện trong dữ liệu cục bộ, hoặc nếu bằng chứng không xác định, G-IDS Agent kế cận sẽ được kết nối tới Grid, thực thi lần đầu một số hành động mềm (soft actions) trên node bị "nhiễm" (một node bị ảnh hưởng bởi hành động tấn công) và nếu cần sẽ có các hành động cứng (hard actions) trên node bị nhiễm ban đầu và sau đó là các node kế cận (định tuyến, khôi phục và cấu hình lại hệ thống mạng).

Các G-IDS Agent sẽ tham gia hợp tác trong các hành động phát hiện xâm nhập tổng thể, chia sẻ phần cứng và các tài nguyên phần mềm do được kết nối tới cùng Grid. Các G-IDS Agent riêng lẻ này sẽ chạy trong mỗi node di động, tập hợp mẫu hệ thống G-IDS tổng thể để bảo vệ mạng không dây di động.



Hình 12: Hệ thống G-IDS tổng thể

Kiến trúc bảo mật G-IDS đã đề xuất định nghĩa một quá trình theo dõi liên tục, dựa trên ý tưởng: "Nếu bạn có thể phản hồi đủ nhanh, bạn có thể đá anh ta ra trước khi anh ta gây nguy hiểm..".

Mỗi G-IDS Agent của một node không dây kết nối tới mạng Ad Hoc cung cấp các dịch vụ sau đây:

1/. Theo dõi liên tục

Thực hiện một hệ thống theo dõi ở chế độ thụ động. Nó đại diện cho khối thành phần cơ bản của một cơ sở hạ tầng theo dõi mạng Ad Hoc cho phép chia sẻ những thống kê lưu lượng thông tin và các thông tin trên toàn bộ Grid. Module CoMo thực thi 2 mức theo dõi: mức hệ thống và mức ứng dụng.

2/. Ra quyết định

Đây là một phần của hệ thống điều khiển dây chuyền đóng, sử dụng dữ liệu từ dịch vụ CoMo, ra quyết định về những hành động thời gian thực cần thiết để xử lý những sự dị thường liên quan đến các xâm nhập mạng. Module này thực thi các logic cần thiết để phân tích hành vi và ra quyết định đối với các dị thường vừa phát hiện ra.

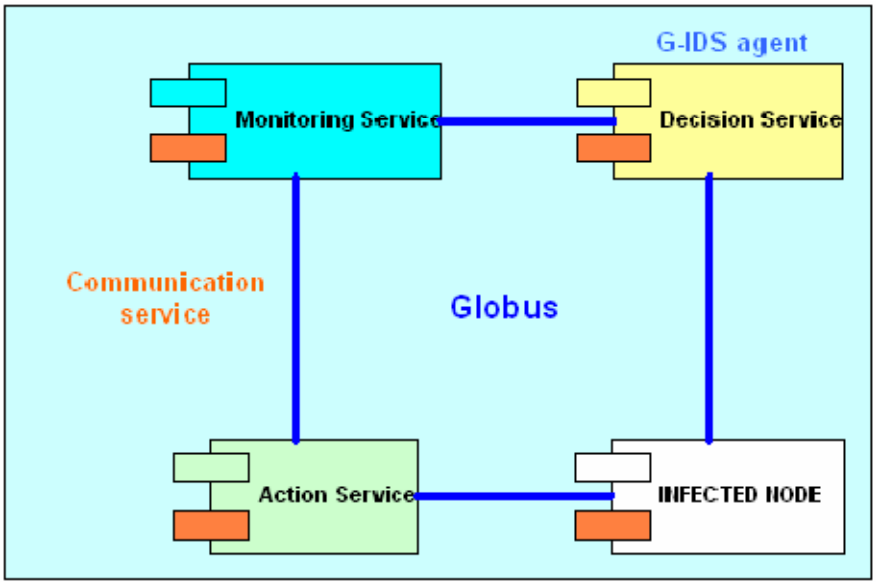
3/. Thực thi hành động

Tất cả các node sẽ có các module hành động chịu trách nhiệm xử lý tình trạng xâm nhập trên một host không dây.

4/. Truyền thông

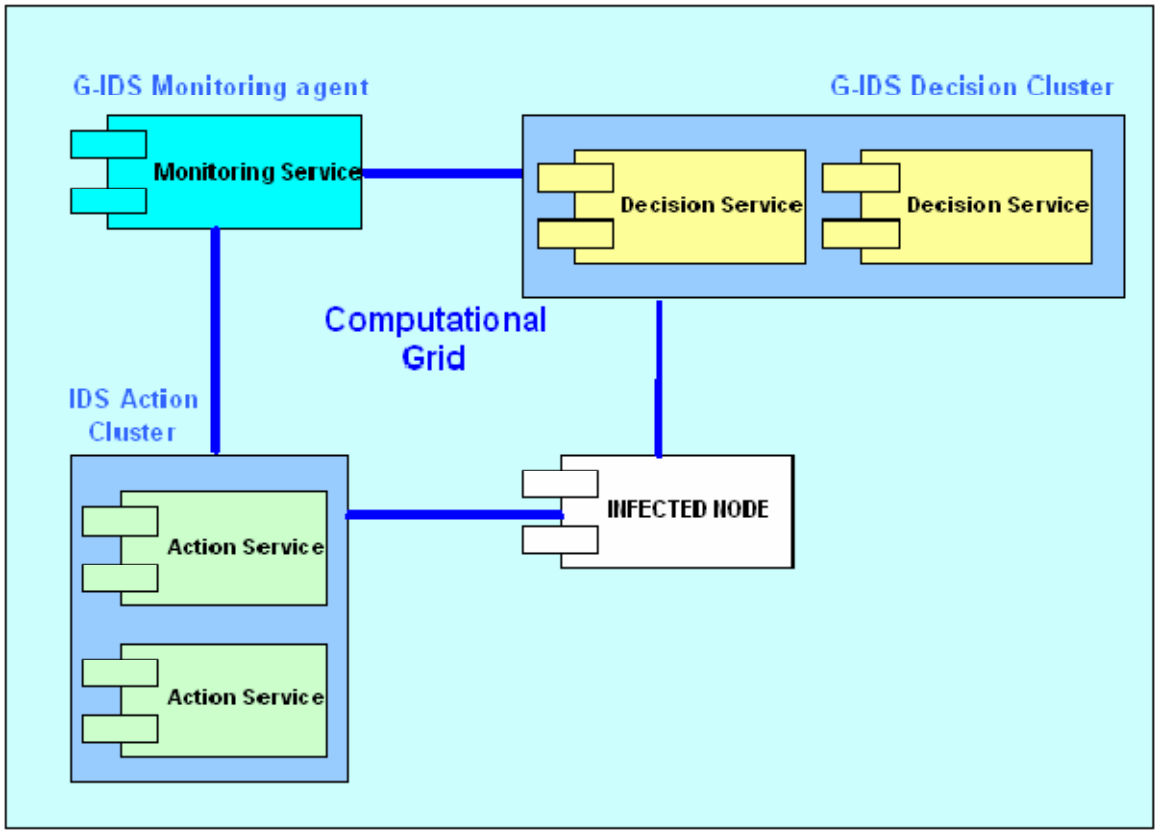
Mỗi node trao đổi thông tin về các hành vi lạ thường và hiểm độc trên một vài đoạn mạng (segment) hoặc trên các host nhất định, đồng thời đáp trả những hành vi xâm nhập đó. Dịch vụ truyền thông được thực hiện đầy đủ sử dụng framework mã nguồn mở GLOBUS.

Mỗi dịch vụ G-IDS đại diện cho một agent con kết nối tới Grid, có khả năng chia sẻ tài nguyên tính toán và các dịch vụ với các agent khác kết nối tới cùng một Grid



Hình 13: Hệ thống G-IDS tổng thể

Chiến lược đã đề xuất khiến cho tải của toàn bộ mạng rất nhỏ bởi vì khi cần thiết, một Agent hoặc một nhóm các Grid-enabled Agent (Cluster) được cấp phép để phân chia nhiệm vụ chức năng trong các "service" nhằm phục vụ cho một mục đích cụ thể.

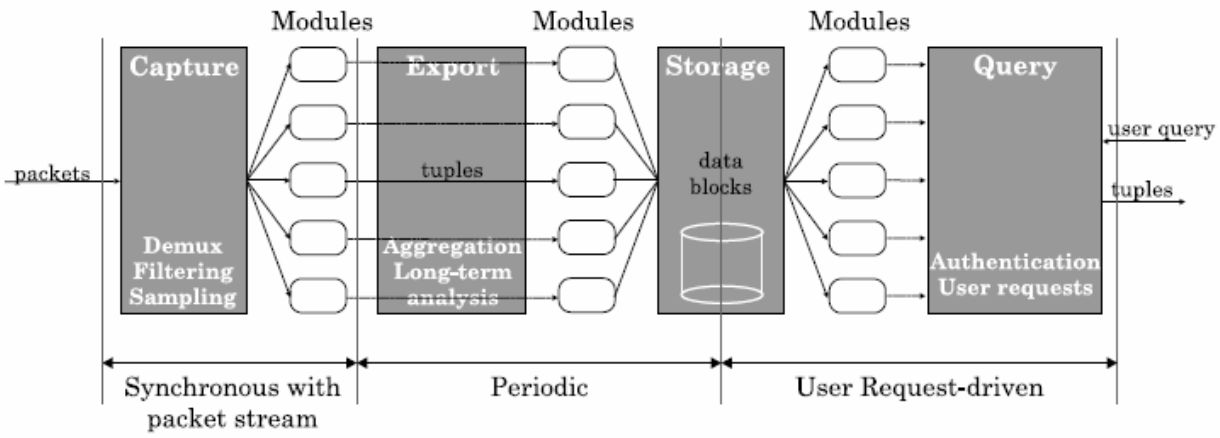


Hình 14: Phân tách nhiệm vụ trong G-IDS Cluster

Bằng cách này, tải làm việc của hệ thống G-IDS được phân tán ra các node, để giảm thiểu tiêu thụ điện năng và thời gian xử lý giữa các node.

3.2.4 Môi trường lưới bảo mật dựa trên việc tích hợp globus và como

CoMo được tạo bởi hai thành phần chính: các quá trình xử lý lõi (điều khiển đường dẫn dữ liệu xuyên qua G-IDS, bao gồm bắt gói tin, xuất bản, lưu trữ, quản lý yêu cầu và điều khiển tài nguyên); module plug-in (chịu trách nhiệm đối với các biến đổi khác nhau của dữ liệu). Dòng dữ liệu xuyên qua hệ thống CoMo được minh họa dưới đây:



Hình 15: Dòng dữ liệu trong CoMo

Các ô trắng chỉ ra các module plug-in, còn các ô xám đại diện cho các quá trình xử lý lõi. Một mặt CoMo tập hợp các gói tin trên đường dẫn được theo dõi. Những gói tin này được xử lý bởi các quá trình xử lý lõi kế tiếp nhau và cuối cùng được lưu trên một thiết bị nhớ. Mặt khác, dữ liệu sẽ được nhận từ thiết bị nhớ khi có yêu cầu của người dùng.

Các quá trình xử lý lõi nằm trong các thao tác di chuyển dữ liệu. Di chuyển dữ liệu trong trong một node không dây là nhiệm vụ tốn chi phí nhất: bộ nhớ, bus, băng thông.

Truyền thông giữa các quá trình xử lý lõi được điều khiển bởi một hệ thống chuyển thông điệp đơn hướng. Năm quá trình xử lý chính hợp thành lõi của CoMo như sau:

1/. Quá trình Bắt giữ (Capture)

Chịu trách nhiệm bắt gói tin, lọc gói tin, lấy mẫu và duy trì thông tin trạng thái trên mỗi module.

2/. Quá trình Xuất bản (Export)

Cho phép phân tích thuật ngữ dài của lưu lượng thông tin và cung cấp truy nhập tới thông tin mạng bổ sung (ví dụ: bảng định tuyến).

3/. Quá trình Lưu trữ (Storage)

Sắp xếp và quản lý truy cập tới bộ nhớ.

4/. Quá trình yêu cầu (Query)

Tiếp nhận các yêu cầu người dùng, tác động query trên lưu lượng thông tin (hoặc đọc các kết quả tiền tính toán) và trả lại kết quả.

5/. Quá trình giám sát (Supervisor)

Chịu trách nhiệm nắm bắt các lỗi (ví dụ: xử lý thất bại) và quyết định xem sẽ tải, chạy hay dừng các module plug-in phụ thuộc và các tài nguyên sẵn sàng hoặc dựa trên các chính sách truy cập hiện tại.

Các dịch vụ ra quyết định và thực thi quyết định được tính toán trong một tập các module plug-in. Các module có thể được nhìn nhận như là một chức năng lọc, nơi mà bộ lọc sẽ chỉ rõ các gói tin nên được thực thi.

Ví dụ: Nếu luật quyết định là "chặn một số gói tin định sẵn trên cổng 80" thì bộ lọc sẽ chỉ chặn các gói tin có cổng đích là 80.

Các quá trình lõi chịu trách nhiệm chạy các bộ lọc gói tin và liên kết với các module sử dụng một tập các hàm callback. Kiến trúc này hướng tới việc cung cấp một cơ chế tự động giúp một node Grid không dây thích nghi với các G-IDS Service khác nhau, đồng thời cung cấp cơ chế quản lý chính hệ thống Grid.

3.2.5. Lợi ích của tính toán lưới hệ thống chống xâm nhập

Một nguyên tắc trong hệ thống chống xâm nhập đó là yêu cầu phát hiện nhanh và đưa ra các giải pháp nhanh nhất cho việc chống xâm nhập. Việc tổng hợp và phân tích các sự kiện để đưa được ra các kết luận là một bài toán rất lớn với khối lượng xử lý nhiều. Vì một hệ thống càng muốn an toàn thì càng phải tổng hợp và phân tích kỹ các sự kiên và các truy cập trên hệ thống.

Trong các mạng lưới khác, thì việc chống xâm nhập chỉ được thực hiện trên từng máy cục bộ vì vậy mà tốc độ xử lý sẽ chậm chạp, việc đưa ra các hành động không thể tức thời đối với các truy cập trái phép. Còn trong hệ thống lưới, với việc tận dụng khả năng tính toán của các máy tính trong mạng đồng thời kết hợp với việc xử lý phân tán, các truy cập trên một máy có thể được phân tích trên nhiều máy tính khác nhau trong mạng và sử dụng cơ chế phân tán, vì vậy sẽ làm tăng tốc độ xử lý và đưa ra được các hành động nhanh và thích hợp nhất đối với từng truy cập. Do đó sẽ làm tăng khả năng bảo mật cho hệ thống.

KẾT LUẬN

Công nghệ Grid Computing ra đời đã đánh dấu một bước phát triển mới trong lĩnh vực tính toán hiệu năng cao, cho phép tận dụng năng lực xử lý, lưu trữ cùng các tài nguyên nhàn rỗi khác để cung cập một môi trường tính toán có năng lực xử lý lớn, khả năng năng lưu trữ dồi dào để giải quyết các bài toán phức tạp và cần năng lực tính toán cao trong khoa học và thương mại. Trong tương lai chắc chắn rằng cộng nghệ này sẽ rất phát triển, và được triển khai một cách mạnh mẽ, giúp người dùng có thể sử dụng máy tính giống như điện, nước, …

Kết quả chính của khóa luận gồm có:

1/. Tìm hiểu và nghiên cứu qua các nguồn tài liệu để hệ thống lại các vấn đề sau:

- Tổng quan về tính toán lưới
- Cơ sở hạ tầng tính toán lưới

2/. Trình bày ứng dụng của tính toán lưới trong một số bài toán an toàn thông tin

- Nêu ứng dụng của tính toán lưới trong việc kiểm tra số nguyên tố Mersenne
- Trình bày ứng dụng của tính toán lưới trong hệ thống phát hiện xâm nhập

TÀI LIỆU THAM KHẢO

[1] Ian Foster, The Grid, CLUSTERWORLD, vol 1, no. 1, 2001, pp. 1-2
[2] Ian Foster, Carl Kesselman, Steven Tuecke, The Anatomy of Grid, Intl J. Supercomputer Applications, 2001.
[3] Ian Foster, What is the Grid? A Three Point Checklist, Argonne National Laboratory & University of Chicago, 20/06/2002.
[4] Ian Foster, Carl Kesselman, Jeffrey M. Nick, Steven Tuecke, The Physiology of the Grid - An Open Grid Services Architecture for Distributed Systems Integration, Version: 6/22/2002.
[5] I. Foster, D. Gannon, H. Kishimoto, The Open Grid Services Architecture, GLOBAL GRID FORUM, 10/03/2004, http://forge. gridforum. org/projects/ogsawg
[6] S. Tuecke, K. Czajkowski, I. Foster, J. Frey, S. Graham, C. Kesselman, T. Maquire, T. Sandholm, D. Snelling, P. Vanderbilt, Open Grid Services Infrastructure (OGSI) Version 1. 0, GLOBAL GRID FORUM, 27/06/2003, http://www. ggf. org/ogsi-wg
[7] Luis Ferreira, Arun Thakore, Michael Brown, Fabiano Lucchese, Huang RuoBo, Linda Lin, Paul Manesco, Jeff Mausolf, Nasser Momtaheni, Karthik Subbian, Olegario, Hernandez, Grid Services Programming and Application Enablement, Redbooks, IBM Corp, 05/2004, www. ibm. com/redbooks

[8] Tìm hiểu nguồn tài liệu từ các trang web trên Internet như:

http://www. globus. org

http://www. ibm. com/redbook

http://www. ggf. org

http://w3c. org/webservice

http://mersenne. org