TRƯỜNG ĐẠI HỌC BÁCH KHOA

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH

BÁO CÁO LAB MÔN HỆ ĐIỀU HÀNH

LAB1: INTRODUCTION TO LINUX PROGRAMMING

Lớp: L02 – HK231

GV hướng dẫn: Trương Tuấn Phát

SV thực hiện: Trần Đình Phong

Thành phố Hồ Chí Minh - 2023

MỤC LỤC

1. Introduction to Linux Command Line Interface

1.1. Linux Shell

Question: List some other popular Linux shells and describe their highlighted features.

1. Bash (Bourne Again SHell):
    - Là shell mặc định trong hầu hết các bản phân phối Linux.
    - Cung cấp nhiều tính năng, bao gồm viết script, chỉnh sửa dòng lệnh, lịch sử lệnh và điều khiển công việc.
    - Hỗ trợ mở rộng shell, biến, vòng lặp và câu lệnh điều kiện.
    - Có khả năng tùy chỉnh và mở rộng cao.
2. Zsh (Z Shell):
    - Cung cấp nhiều tính năng tương tự như Bash, kèm theo những cải tiến bổ sung.
    - Hỗ trợ hoàn thành tab tiên tiến, sửa lỗi chính tả và globbing.
    - Cung cấp nhiều tuỳ chỉnh, giúp tùy chỉnh shell theo ý muốn.
    - Có sẵn các plugin và giao diện để nâng cao tính năng và diện mạo.
3. Fish (Friendly Interactive SHell):
    - Tập trung vào tính đơn giản và dễ sử dụng.
    - Cung cấp chức năng gợi ý tự động dựa trên lịch sử và hoàn thành lệnh.
    - Hỗ trợ tô màu cú pháp cho lệnh và kết quả.
    - Cung cấp cú pháp nhất quán và dễ hiểu.
4. Ksh (Korn Shell):
    - Cung cấp một tập tính năng lập trình phong phú và khả năng viết script mạnh mẽ.
    - Hỗ trợ chỉnh sửa dòng lệnh tiên tiến, quản lý lịch sử và điều khiển công việc.
    - Tương thích với cả shell Bourne (sh) và shell C (csh).

- Phổ biến trong môi trường Unix thương mại.

5. Tcsh (TENEX C SHell):
    - Phát triển từ shell C (csh) và cung cấp các tính năng bổ sung.
    - Hỗ trợ chỉnh sửa dòng lệnh tiên tiến, bao gồm thay thế lịch sử và hoàn thành lệnh.
    - Hỗ trợ viết script với các tính năng như vòng lặp, biến và câu lệnh điều kiện.
    - Được sử dụng bởi một số nhà phát triển và quản trị viên hệ thống ưa thích cú pháp shell C.
6. Dash:
    - Shell nhẹ và nhanh được thiết kế để tối ưu hiệu suất.
    - Được cài đặt làm shell hệ thống mặc định trong một số bản phân phối Linux tối giản.
    - Tập trung tuân thủ POSIX và tính đơn giản, phù hợp cho viết script và quá trình khởi động hệ thống.
    - Thiếu một số tính năng tương tác có sẵn trong các shell khác, chẳng hạn như chỉnh sửa dòng lệnh tiên tiến.

1.2. Linux Commands

Exercise: Try more options with the ls command and analyze its output. You can also try to combine multiple options into one, ex: -la.

Lệnh ls được sử dụng phổ biến trong Linux để liệt kê nội dung của thư mục. Dưới đây là một số tùy chọn thông dụng với lệnh ls:

1. -l: Hiển thị thông tin chi tiết về các tệp tin và thư mục, bao gồm quyền truy cập, sở hữu, kích thước và thời gian sửa đổi.
2. -a: Hiển thị tất cả các tệp tin và thư mục, bao gồm cả các tệp tin ẩn bắt đầu bằng dấu chấm (.).
3. -h: Hiển thị kích thước tệp tin dưới dạng đọc được cho con người.
4. -t: Sắp xếp các tệp tin theo thời gian sửa đổi, với các tệp tin được sửa đổi gần đây nhất hiển thị đầu tiên.

5. -r: Đảo ngược thứ tự sắp xếp, hiển thị các tệp tin theo thứ tự ngược lại.
6. -S: Sắp xếp các tệp tin theo kích thước, với các tệp tin lớn nhất hiển thị đầu tiên.

1.3. Command Redirection

Question: Compare the Output Redirection (>/») with the Piping (|) technique

"Output redirection" (>/») là một kỹ thuật được sử dụng để chuyển đầu ra của một lệnh đến một tập tin thay vì hiển thị nó trên terminal. Toán tử ">" ghi đè lên nội dung tập tin bằng đầu ra của lệnh, trong khi toán tử "»" thêm đầu ra vào cuối tập tin.

Ví dụ, lệnh "ls > file.txt" sẽ tạo một tập tin có tên là "file.txt" và đưa kết quả của lệnh "ls" vào trong tập tin đó. Nếu tập tin đã tồn tại, nó sẽ bị ghi đè hoặc thêm vào đầu ra của lệnh.

"Piping" (|) là một kỹ thuật cho phép đầu ra của một lệnh được chuyển đến làm đầu vào cho một lệnh khác. Toán tử "|"" cho phép kết nối đầu ra của một lệnh với đầu vào của một lệnh khác.

Ví dụ, lệnh "ls | grep keyword" sẽ hiển thị danh sách tất cả các tập tin và thư mục trong thư mục hiện tại và đầu ra được chuyển đến lệnh "grep" để tìm kiếm các tập tin và thư mục có từ khóa "keyword". Kết quả của lệnh "ls" không được lưu vào tập tin nào mà chỉ được chuyển đến lệnh "grep".

"Output redirection" cho phép chúng ta lưu đầu ra của một lệnh vào một tập tin, trong khi "Piping" cho phép chúng ta chuyển đầu ra của một lệnh làm đầu vào cho một lệnh khác để thực hiện một nhiệm vụ cụ thể.

1.4. SUDO

Question: Compare the sudo and the su command.

"sudo" là một lệnh được sử dụng để thực thi một lệnh với quyền của một người dùng khác. Nó yêu cầu người dùng nhập mật khẩu của người dùng đó để xác thực và cấp quyền truy cập.

Một trong những điểm nổi bật của "sudo" là cho phép cấu hình các quyền truy cập khác nhau cho các người dùng khác nhau, điều này giúp tăng cường bảo mật trên hệ thống.

"su" là một lệnh được sử dụng để đăng nhập vào hệ thống với quyền của một người dùng khác mà không cần cung cấp mật khẩu của người dùng đó (nếu bạn có quyền truy

cập). Nó yêu cầu nhập mật khẩu của người dùng hiện tại để xác thực và đăng nhập lại với quyền của người dùng mới.

Tuy nhiên, việc sử dụng "su" có thể gây ra một số vấn đề bảo mật, đặc biệt là khi nó được sử dụng bởi các người dùng có quyền cao hơn.

Tóm lại, "sudo" cho phép người dùng thực thi một lệnh với quyền của người dùng khác trong khi cung cấp tính linh hoạt và bảo mật hơn, trong khi "su" cho phép người dùng đăng nhập vào hệ thống với quyền của một người dùng khác mà không cần cung cấp mật khẩu của người dùng đó, nhưng có thể gây ra một số vấn đề bảo mật.

1.5. Permission on Linux

Question: Discuss about the 777 permission on critical services (web hostings, sensitive databases, ...).

Trong hệ thống Unix/Linux, quyền truy cập vào tệp và thư mục được quản lý bằng cách sử dụng mã số hóa permission. Trong đó, số đầu tiên đại diện cho quyền của chủ sở hữu tệp/thư mục, số thứ hai đại diện cho quyền của người dùng trong cùng nhóm, số cuối cùng đại diện cho quyền của tất cả các người dùng khác.

Mã 777 permission có nghĩa là tất cả các người dùng đều có quyền đọc, ghi và thực thi tệp/thư mục. Điều này có thể gây ra nhiều vấn đề bảo mật nếu áp dụng cho các dịch vụ quan trọng như web hosting và cơ sở dữ liệu nhạy cảm.

Nếu một hacker tấn công thành công vào một trong những dịch vụ này và có quyền truy cập vào các tệp/thư mục được cấp quyền 777, họ có thể tùy ý thay đổi, xóa hoặc thêm mới các tệp và thư mục. Điều này có thể dẫn đến mất dữ liệu, làm hỏng dịch vụ hoặc thậm chí là vi phạm quyền riêng tư của người dùng.

Do đó, cần phải đặt quyền truy cập phù hợp cho các dịch vụ quan trọng như web hosting và cơ sở dữ liệu nhạy cảm. Nên chỉ cấp quyền cần thiết cho các người dùng hoặc nhóm người dùng liên quan đến dịch vụ đó và hạn chế quyền truy cập cho các người dùng khác. Nếu có nhu cầu, có thể sử dụng các công cụ như Access Control Lists (ACLs) để quản lý quyền truy cập chi tiết hơn cho các tệp và thư mục.

2. Bash Script Programming

2.1. Practice Exercises

3. Introduction to GNU C Compiler

3.1. GNU Make

Question 1. What are the advantages of Makefile? Give examples?

Makefile mang lại một số lợi ích trong quá trình phát triển và xây dựng các dự án phần mềm. Dưới đây là một số lợi ích chính của việc sử dụng Makefile:

1. Tự động hóa: Makefile tự động hóa quá trình biên dịch và xây dựng dự án. Nó giúp đơn giản hóa việc thực hiện các bước biên dịch, liên kết và xây dựng một cách tự động, giúp tiết kiệm thời gian và giảm công sức cần thiết.

Ví dụ: Bạn có thể sử dụng Makefile để tự động biên dịch và xây dựng một dự án phần mềm có nhiều tệp mã nguồn và thư viện phụ thuộc. Thay vì phải gõ từng lệnh biên dịch và liên kết thủ công, bạn chỉ cần chạy lệnh make và Makefile sẽ tự động thực hiện các bước cần thiết.

2. Độc lập nền tảng: Makefile cho phép bạn xây dựng dự án trên nhiều nền tảng và hệ điều hành khác nhau. Bạn có thể định nghĩa các quy tắc và cờ biên dịch phù hợp với từng nền tảng, giúp dự án của bạn dễ dàng chạy trên các môi trường khác nhau mà không cần sửa đổi mã nguồn.

Ví dụ: Bạn có thể sử dụng Makefile để xây dựng một ứng dụng trên cả Windows và Linux. Bằng cách định nghĩa các quy tắc và cờ biên dịch phù hợp cho mỗi nền tảng trong Makefile, bạn có thể tự động biên dịch và xây dựng ứng dụng trên cả hai hệ điều hành mà không cần chỉnh sửa mã nguồn.

3. Tối ưu hóa tự động: Makefile giúp tối ưu hóa quá trình biên dịch và xây dựng bằng cách chỉ biên dịch lại các tệp mã nguồn được thay đổi hoặc phụ thuộc của chúng. Điều này giúp tránh việc biên dịch lại toàn bộ dự án khi chỉ có một số tệp thay đổi, giúp tiết kiệm thời gian và tăng hiệu suất quá trình xây dựng.

Ví dụ: Khi bạn chỉnh sửa một tệp mã nguồn trong dự án và chạy lệnh make, Makefile sẽ tự động tìm ra các tệp phụ thuộc đã thay đổi và chỉ biên dịch lại những phần cần thiết. Điều này giúp giảm thời gian biên dịch và xây dựng so với việc biên dịch lại toàn bộ dự án.

4. Quản lý dự án linh hoạt: Makefile giúp quản lý dự án một cách linh hoạt và dễ dàng. Bạn có thể định nghĩa các quy tắc tùy chỉnh để thực hiện các tác vụ như kiểm tra lỗi, chạy các bài kiểm tra tự động, tạo tài liệu, và nhiều hơn nữa.

Ví dụ: Bạn có thể thêm các quy tắc tùy chỉnh để thực hiện các công việc như kiểm tra lỗi, chạy các bài kiểm tra tự động, tạo tài liệu và nhiều tác vụ khác. Điều này giúp tạo ra một quy trình làm việc mạnh mẽ và linh hoạt cho dự án của bạn.

Tóm lại, Makefile mang lại nhiều lợi ích cho việc phát triển và xây dựng dự án phần mềm. Nó tự động hóa quá trình biên dịch và xây dựng, giúp đảm bảo tính độc lập nền tảng và tối ưu hóa tự động. Ngoài ra, Makefile cũng mang lại tính linh hoạt và quản lý dự án hiệu quả.

Question 2. Compiling a program in the first time usually takes a longer time in comparison with the next re-compiling. What is the reason?

Lý do chính tại sao việc biên dịch chương trình lần đầu thường mất thời gian lâu hơn so với việc biên dịch lại sau đó là do quá trình biên dịch ban đầu phải thực hiện nhiều công việc bổ sung. Dưới đây là một số lý do cụ thể:

1. Phân tích cú pháp và tạo cây cú pháp: Trong quá trình biên dịch ban đầu, trình biên dịch phải đọc và phân tích cú pháp của toàn bộ mã nguồn. Nó phải xác định cấu trúc ngôn ngữ, kiểm tra cú pháp và tạo ra cây cú pháp tương ứng. Quá trình này tốn thời gian và tài nguyên tính toán.
2. Tìm và liên kết các thư viện: Khi biên dịch lần đầu, trình biên dịch phải tìm và liên kết các thư viện mà chương trình phụ thuộc. Điều này bao gồm tìm kiếm và tải các tệp thư viện từ hệ thống tệp và xác định các kết nối và phụ thuộc giữa chúng. Quá trình này cũng đòi hỏi thời gian và tài nguyên.
3. Tạo mã máy: Sau khi phân tích cú pháp và liên kết đã hoàn thành, trình biên dịch tiến hành tạo mã máy tương ứng từ mã nguồn. Quá trình này bao gồm biên dịch mã nguồn thành mã trung gian, tối ưu hóa mã trung gian và sau đó tạo ra mã máy cuối cùng. Quá trình tạo mã máy có thể tốn thời gian đáng kể.

Khi đã biên dịch một lần, các tệp tạo ra như mã trung gian và mã máy được lưu trữ và không cần phải thực hiện lại từ đầu. Khi chúng ta chỉnh sửa một phần của mã nguồn và tiến hành biên dịch lại, trình biên dịch chỉ cần biên dịch các phần bị thay đổi và các phần

phụ thuộc liên quan. Việc này giúp tiết kiệm thời gian và tài nguyên so với việc biên dịch toàn bộ chương trình lại.

Tóm lại, việc biên dịch lần đầu tiên mất thời gian lâu hơn vì quá trình phân tích cú pháp, liên kết thư viện và tạo mã máy phải được thực hiện từ đầu. Nhưng khi đã biên dịch lần đầu, việc biên dịch lại chỉ tập trung vào các phần thay đổi và phụ thuộc liên quan, giúp tiết kiệm thời gian và tăng tốc quá trình biên dịch.

Question 3. Is there any Makefile mechanism for other programming languages? If it has, give an example?

Có, Makefile không chỉ hỗ trợ cho việc biên dịch các chương trình viết bằng ngôn ngữ C/C++ mà còn có thể sử dụng cho các ngôn ngữ lập trình khác. Điều này là do Makefile không phụ thuộc vào ngôn ngữ lập trình mà chỉ sử dụng các công cụ, trình biên dịch và trình liên kết cụ thể cho từng ngôn ngữ.

Ví dụ, để sử dụng Makefile cho Python, ta có thể sử dụng trình biên dịch Python như làm công cụ để thực thi Makefile. Các tác vụ cần thiết để biên dịch chương trình Python sẽ được định nghĩa trong Makefile, giúp tự động hóa việc biên dịch, kiểm tra lỗi và tạo các file thực thi.

Ngoài ra, Makefile cũng có thể được sử dụng cho các ngôn ngữ lập trình như Java, Ruby, Perl, và nhiều ngôn ngữ khác, tùy thuộc vào công cụ và trình biên dịch được sử dụng cho từng ngôn ngữ.

Link Github: https://github.com/Phong2114404/LAB1_OS.git