

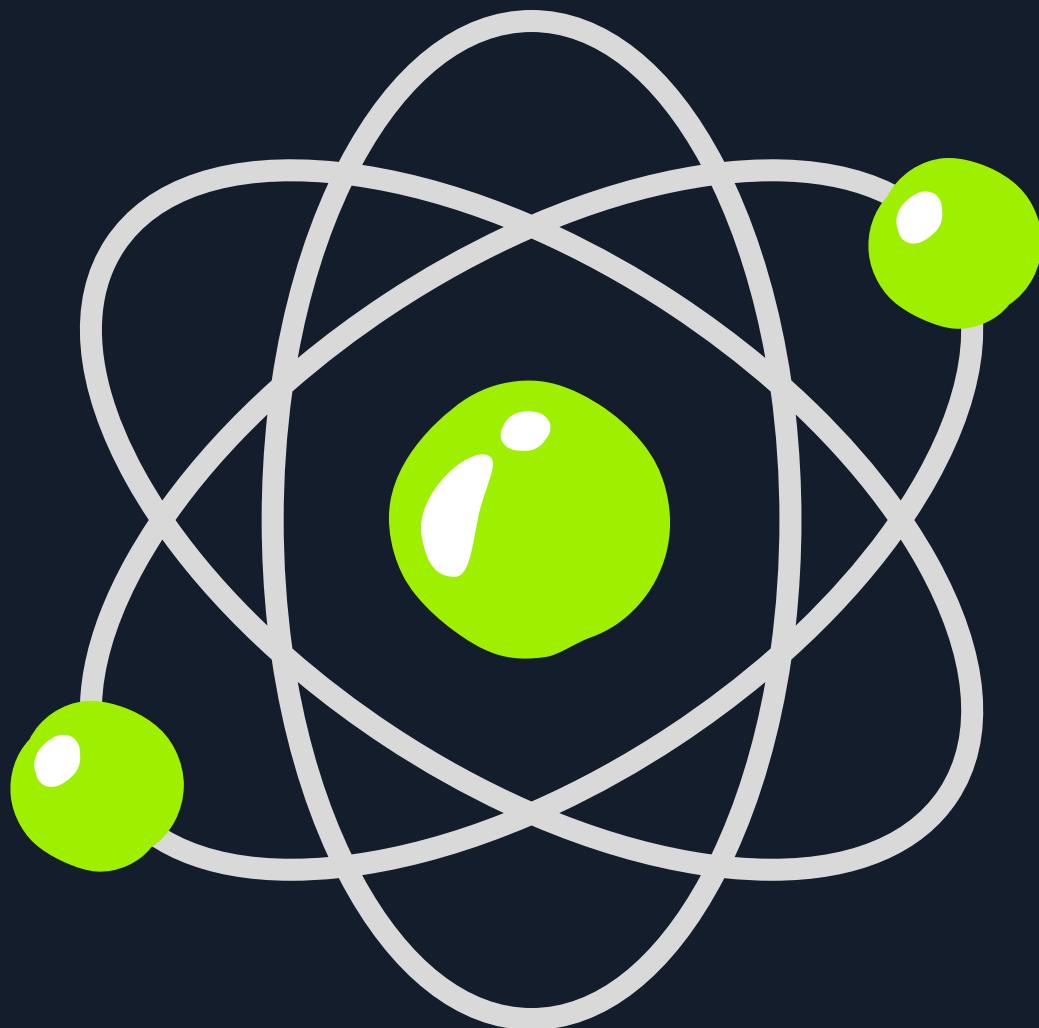
Nguyen Minh Hung  
Minhung



HACKTHEBOX

# Linux Structure

#LinuxFundamentals #Introduction #Part1



# History

Bắt đầu với việc phát hành hệ điều hành Unix của Ken Thompson và Dennis Ritchie (cả hai đều làm việc cho AT&T vào thời điểm đó) vào năm 1970. The Berkeley Software Distribution (BSD) được phát hành vào năm 1977, nhưng vì nó chứa Unix code do AT&T sở hữu nên một vụ kiện đã hạn chế sự phát triển của BSD. Richard Stallman bắt đầu dự án GNU (GNU Is Not Unix) vào năm 1983. Mục tiêu của ông là tạo ra một hệ điều hành giống Unix nhưng “free” (freedom), và một phần dự án của ông đã dẫn đến việc tạo ra giấy phép GNU General Public License (GPL).

Linux là một dự án cá nhân bắt đầu vào năm 1991 bởi một sinh viên Phần Lan tên là Linus Torvalds. Qua nhiều năm, Linux kernel đã chuyển từ một số lượng nhỏ các file được viết bằng ngôn ngữ C sang phiên bản mới nhất với hơn 23 triệu dòng code (không bao gồm comment), được cấp phép theo GNU General Public License v2.

Linux có sẵn trong hơn 600 bản phân phối (hệ điều hành dựa trên Linux kernel). Phổ biến và nổi tiếng nhất là Ubuntu, Debian, Fedora, OpenSUSE, elementary, Manjaro, Gentoo Linux, RedHat và Linux Mint.

# History

**Linux** thường được đánh giá là bảo mật hơn so với các hệ điều hành khác và mặc dù trước đây kernel của nó có nhiều lỗ hổng nhưng vấn đề này đã ngày càng ít xảy ra hơn. Nó ít bị nhiễm phần mềm độc hại hơn hệ điều hành Windows và được cập nhật rất thường xuyên. Linux cũng rất ổn định và thường mang lại hiệu suất rất cao cho end user. Tuy nhiên, nó sẽ có thể khó khăn hơn đối với những người mới bắt đầu cũng như không có nhiều hardware driver bằng Windows.

Vì Linux là mã nguồn mở và miễn phí, nó có thể được sửa đổi và phân phối thương mại hoặc phi thương mại bởi bất kỳ ai. Các hệ điều hành dựa trên Linux chạy trên máy chủ, máy tính lớn, máy tính để bàn, hệ thống nhúng như router, TV, game console, v.v. Hệ điều hành Android chạy trên điện thoại thông minh và máy tính bảng đều dựa trên **Linux kernel** và do đó, Linux có thể xem là hệ điều hành được cài đặt rộng rãi nhất.

Linux là một hệ điều hành giống như **Windows**, **iOS**, **Android** hoặc **macOS**. Hệ điều hành là phần mềm quản lý tất cả các tài nguyên phần cứng được liên kết với máy tính của chúng ta, nói cách khác một hệ điều hành quản lý toàn bộ giao tiếp giữa phần mềm và phần cứng. Ngoài ra, còn tồn tại nhiều bản phân phối (**distro**) khác nhau. Nó giống như những phiên bản của hệ điều hành Windows vậy.

# Philosophy

Linux tuân theo 5 nguyên tắc cốt lõi:

- **Everything is a file**: ví dụ như tệp `/etc/passwd` là một file lưu trữ tất cả người dùng đã đăng ký trên hệ thống.
- **Small, single-purpose programs**: Linux cung cấp những công cụ để làm tốt một việc, điều này giúp cho việc sử dụng hiệu quả tài nguyên hệ thống và các chương cũng trở nên dễ hiểu.
- **Ability to chain programs together to perform complex tasks**: Tích hợp và kết hợp các công cụ khác nhau để thực hiện những tác vụ lớn và phức tạp.
- **Avoid captive user interfaces**: Linux được thiết kế để hoạt động chủ yếu với shell (hoặc terminal), cho phép người dùng kiểm soát hệ điều hành tốt hơn.
- **Configuration data stored in a text file**: Tất cả các tệp cấu hình (configuration files) cho các dịch vụ chạy trên hệ điều hành Linux được lưu trữ trong một hoặc nhiều tệp văn bản (text file).

# Components

- **Bootloader:** Một đoạn code chạy để hướng dẫn quá trình khởi động hệ điều hành. Ubuntu sử dụng GRUB Bootloader.
- **OS Kernel:** Kernel là thành phần chính của hệ điều hành, quản lý tài nguyên cho I/O device ở cấp độ phần cứng.
- **Daemons:** Các service chạy ẩn dưới nền trong Linux gọi là “daemon”. Trong Unix, tên của daemon thường kết thúc bằng chữ cái “d“. Một số ví dụ thường gặp có thể kể đến như **inetd**, **httpd**, **mysqld**, **nfsd**, **sshd**, **containerd**...
- **OS Shell:** operating system shell là giao diện câu lệnh giữa hệ điều hành và người dùng. Giao diện này cho phép người dùng ra lệnh cho hệ điều hành phải làm gì. Các ngôn ngữ shell được sử dụng phổ biến là **Bash**, **Tcsh/Csh**, **Ksh**, **Zsh** và **Fish**.
- **Graphics server:** Cung cấp graphical sub-system (server) gọi là “X“ hoặc “X-server“, cho phép các chương trình đồ họa chạy local hoặc remote trên hệ thống X-window.
- **Window Manager:** là một graphical user interface (GUI), có nhiều tùy chọn bao gồm **GNOME**, **KDE**, **MATE** và **Cinnamon**.
- **Utilities:** là các chương trình thực hiện những chức năng cụ thể cho người dùng hoặc chương trình khác.

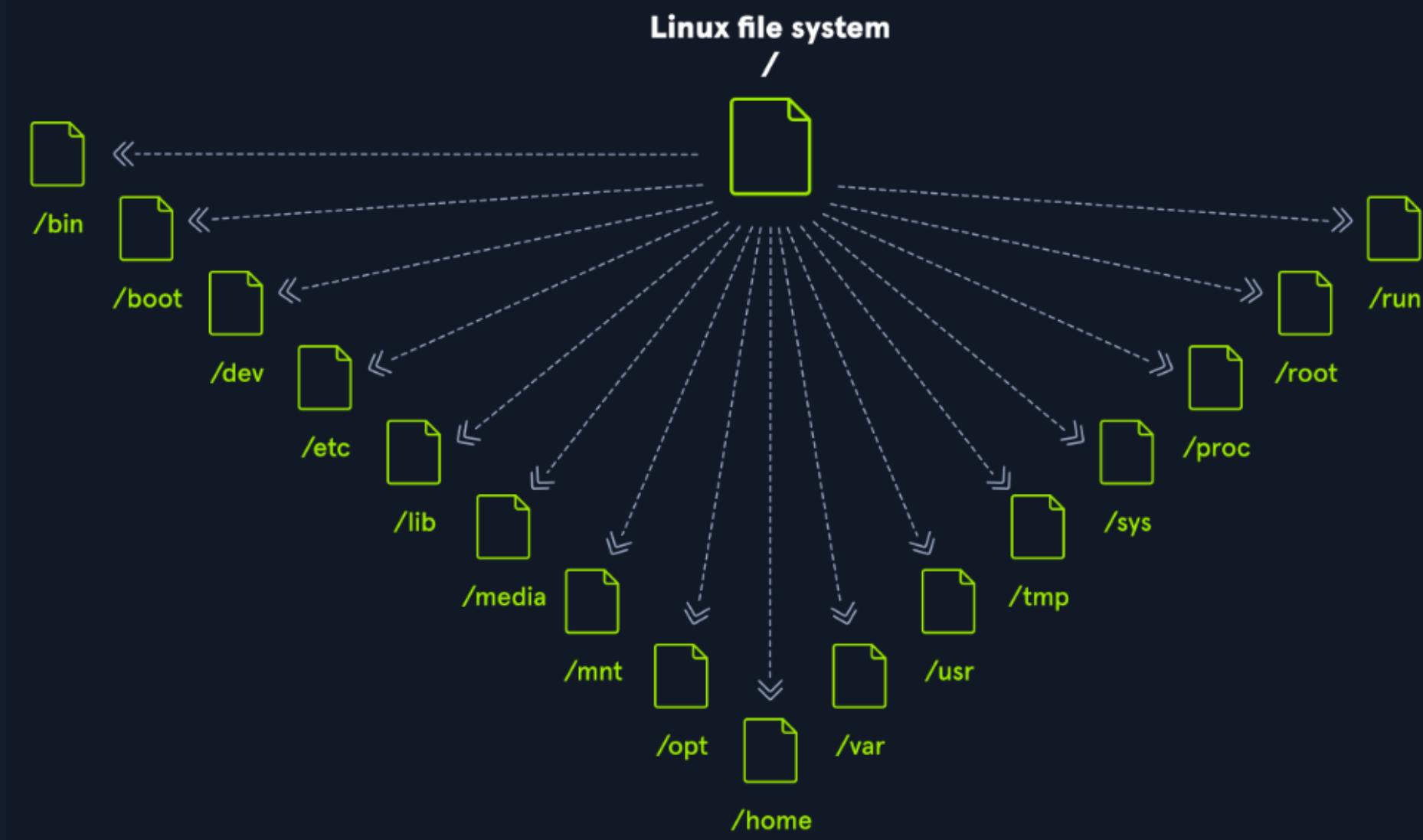
# Linux Architecture

Hệ điều hành Linux có thể được chia thành các lớp từ dưới lên theo góc độ tương tác với người dùng như sau:

- **Hardware layer**: Các thiết bị ngoại vi (peripherals) như RAM, ổ cứng, CPU...
- **Kernel layer**: lõi của hệ điều hành Linux, có chức năng ảo hóa (virtualize) và kiểm soát các tài nguyên (resource) phần cứng (hardware) như CPU, phân bổ bộ nhớ, truy cập dữ liệu... Kernel cung cấp cho mỗi quy trình virtual resource của riêng nó và ngăn chặn, giảm thiểu xung đột với các quy trình khác.
- **Shell layer**: Giao diện câu lệnh - command-line interface (CLI) để người dùng nhập câu lệnh thực thi các chức năng của kernel.
- **System Utility**: Cung cấp cho người dùng tất cả các chức năng của hệ điều hành.

# File System Hierarchy

Hệ điều hành Linux được cấu trúc theo dạng cây phân cấp (tree-like hierarchy), theo document của **Filesystem Hierarchy Standard (FHS)**. Cấu trúc tiêu chuẩn của Linux từ top-level directory như sau:



# File System Hierarchy

- **/** : top-level directory, root filesystem, chứa tất cả các file cần thiết để boot hệ điều hành trước khi filesystem khác được mount cũng như các file cần thiết để boot các filesystem khác. Sau khi khởi động, tất cả các filesystem khác được mount dưới dạng subdirectory của root.
- **/bin** : chứa các lệnh nhị phân thiết yếu.
- **/boot** : bao gồm static bootloader, thực thi kernel và các file cần thiết để khởi động hệ điều hành Linux.
- **/dev** : chứa device file để truy cập vào hardware device được gắn vào hệ thống.
- **/etc** : các tệp cấu hình (configuration file) của local system. Các configuration file của ứng dụng đã cài đặt cũng có thể được lưu ở đây.
- **/home** : mỗi user của system sẽ có một subdirectory để lưu trữ tại đây.
- **/lib** : các shared library file cần thiết để boot system.

# File System Hierarchy

- **/media** : các media device ngoài có thể tháo rời như USB được mount tại đây.
- **/mnt** : nơi mount tạm thời cho các filesystem thông thường.
- **/opt** : optional file của các công cụ bên thứ ba có thể được lưu tại đây.
- **/root** : giống **/home** nhưng dành cho root user.
- **/sbin** : chứa các tệp thực thi quản trị hệ thống (binary system file)
- **/tmp** : hệ điều hành, các chương trình lưu trữ các tệp tạm thời (temporary file). Thường bị xóa khi khởi động hệ thống và có thể bị xóa vào những thời điểm khác mà không có bất kỳ cảnh báo nào.
- **/usr** : chứa các tệp thực thi, library, man (manual) file...
- **/var** : chứa các tệp dữ liệu (data file) như logfile, email in-boxes, web app files, cron file...

\*tệp thực thi trên hệ điều hành Linux giống như các file đuôi “.exe” trên hệ điều hành Windows.

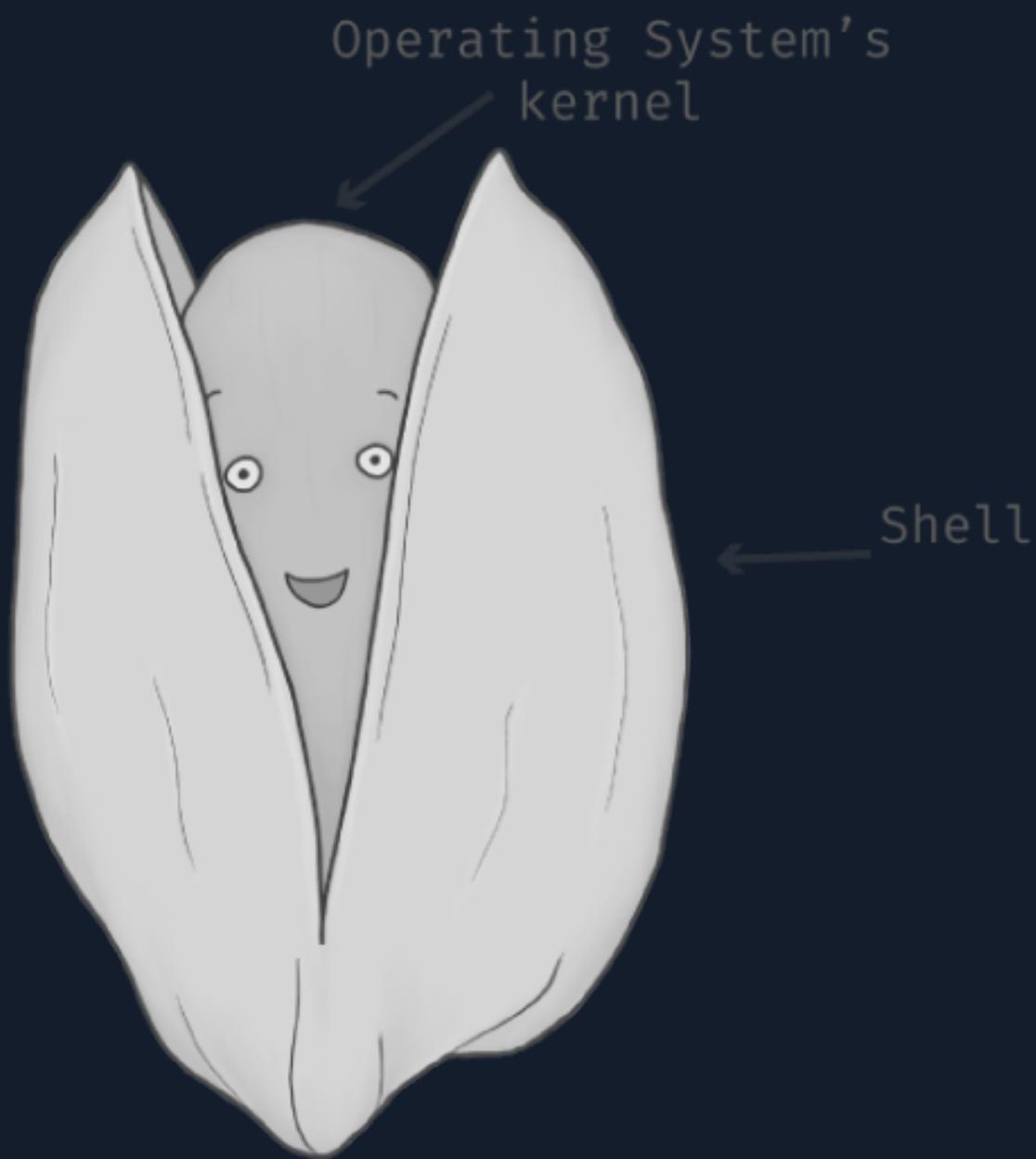
Nguyen Minh Hung  
Minhung



HACKTHEBOX

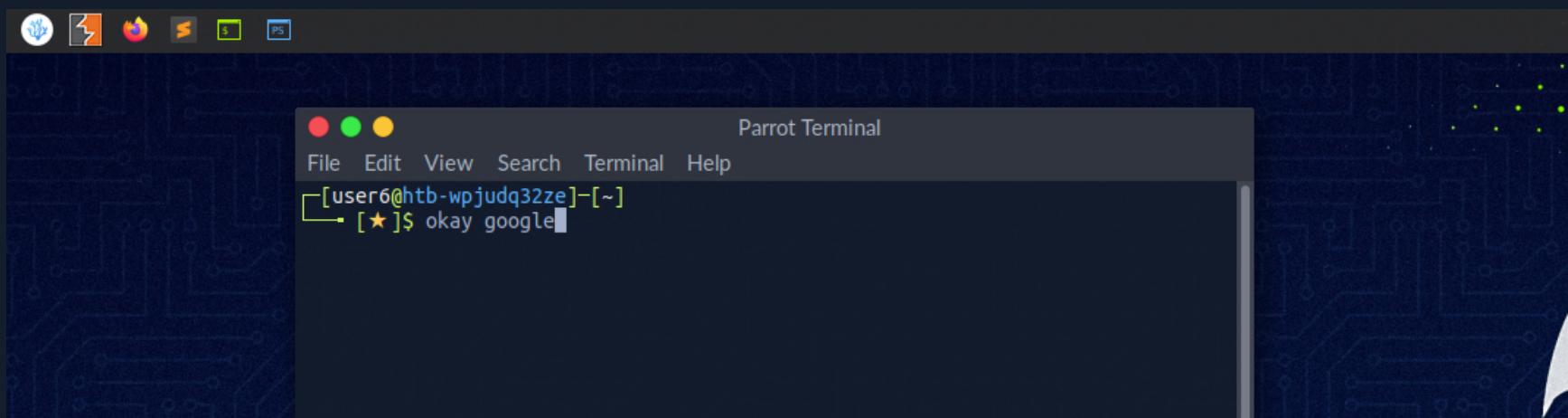
# Introduction to Shell

#LinuxFundamentals #Introduction #Part2



# Introduction to Shell

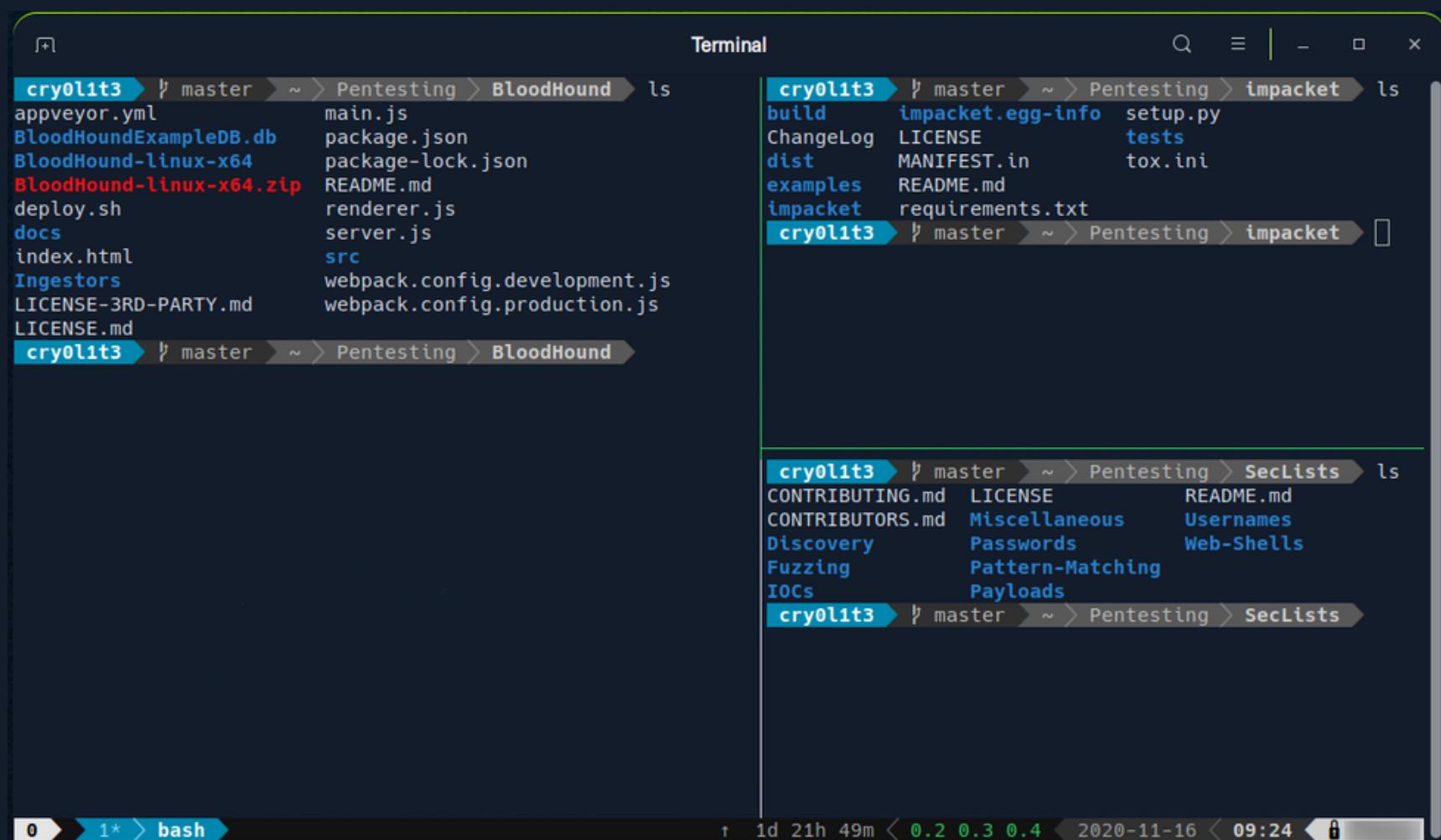
Một trong những điều quan trọng là phải học cách sử dụng Linux shell vì có nhiều server chạy Linux. Chúng thường được sử dụng vì Linux ít bị lỗi hơn so với các Windows server. Ví dụ như web server thường dùng Linux (Linux distro). Biết cách sử dụng hệ điều hành để kiểm soát một cách hiệu quả đòi hỏi phải hiểu và nắm vững phần thiết yếu của Linux, đó là **Shell**. Giao diện khi sử dụng Linux distro là Parrot OS:



Linux terminal, còn được gọi là **shell** hay command line, cung cấp giao diện input/output (I/O) dạng text giữa user và Linux kernel của system. Thuật ngữ console cũng có thể hay gấp nhưng sẽ không ở dạng cửa sổ mà là toàn bộ screen ở text mode. Trong cửa sổ terminal, các câu lệnh có thể được thực thi để điều khiển hệ thống. Ngắn gọn, một terminal đóng vai trò là giao diện cho trình thông dịch shell.

# Terminal Emulators

Terminal emulator thường được sử dụng cho việc này, đây là phần mềm mô phỏng chức năng của một terminal. Nó cho phép sử dụng các chương trình dạng text trên một giao diện đồ họa - graphical user interface (GUI). Có nhiều terminal emulator khác nhau như **GNOME Terminal**, **XFCE4 Terminal**, **XTerm**... Ngoài ra còn có command-line interface (CLI) với những terminal bổ sung bên trong một terminal, được gọi là **multiplexer**. Có thể kể đến những multiplexer như **Tmux**, **GNU Screen**, **Konsole** hay **Terminator**. Các terminal emulator và multiplexer là tiện ích mở rộng (extension) cho terminal, giúp chia nhỏ terminal trong một cửa sổ, làm việc trên nhiều directory, tạo các không gian làm việc (workspace).



## multiplexer Timux

# Shell

Shell được sử dụng phổ biến nhất trong Linux là **Bourne-Again Shell (BASH)** và là một phần của dự án GNU. Mọi thứ chúng ta làm thông qua GUI, đều có thể làm với shell. Shell đem đến nhiều khả năng hơn để tương tác với các chương trình và quy trình (process) để lấy thông tin nhanh hơn. Bên cạnh đó, nhiều process có thể dễ dàng được tự động hóa với các tệp chứa các câu lệnh (script) giúp công việc thủ công trở nên dễ dàng hơn rất nhiều.

Bên cạnh Bash, còn có nhiều shell khác như **Tcsh/Csh, Ksh, Zsh, Fish shell...**



Bourne-Again Shell (BASH)

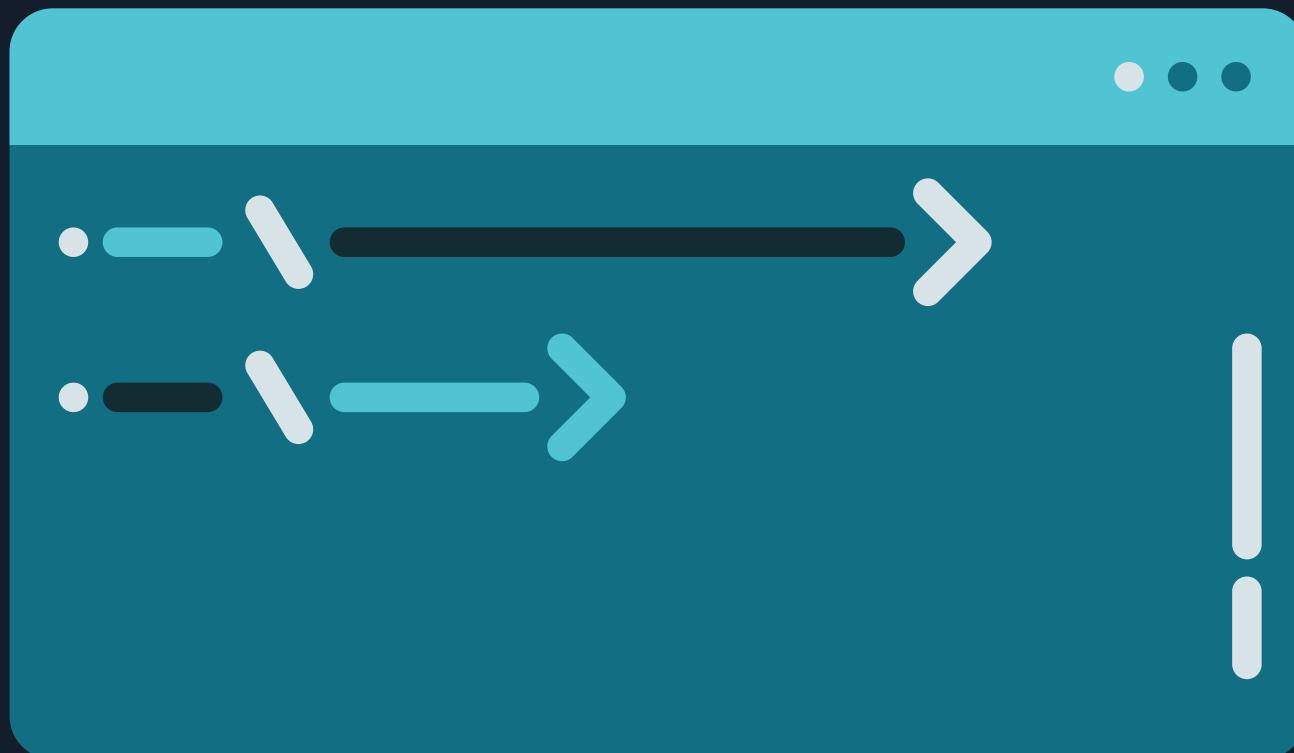
Nguyen Minh Hung  
Minhung



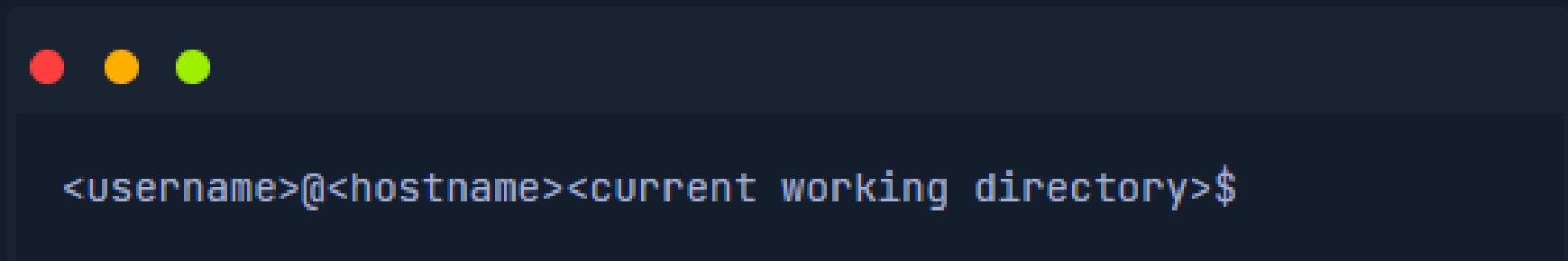
HACKTHEBOX

# Prompt Description

#LinuxFundamentals #TheShell #Part3

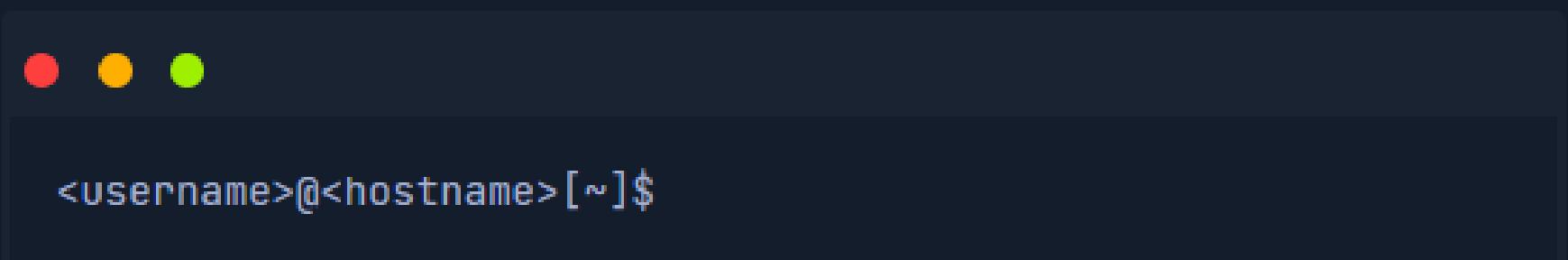


Dấu nhắc lệnh (bash prompt) theo mặc định, bao gồm các thông tin như username, hostname và directory đang làm việc hiện tại. Định dạng có thể trông giống như thế này:



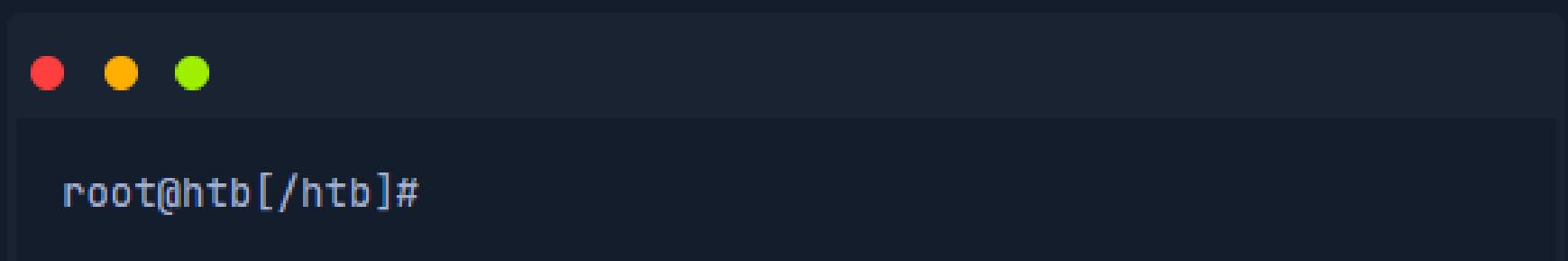
```
<username>@<hostname><current working directory>$
```

Home directory của user được ký hiệu bằng dấu `<~>` và là directory mặc định khi đăng nhập.



```
<username>@<hostname>[~]$
```

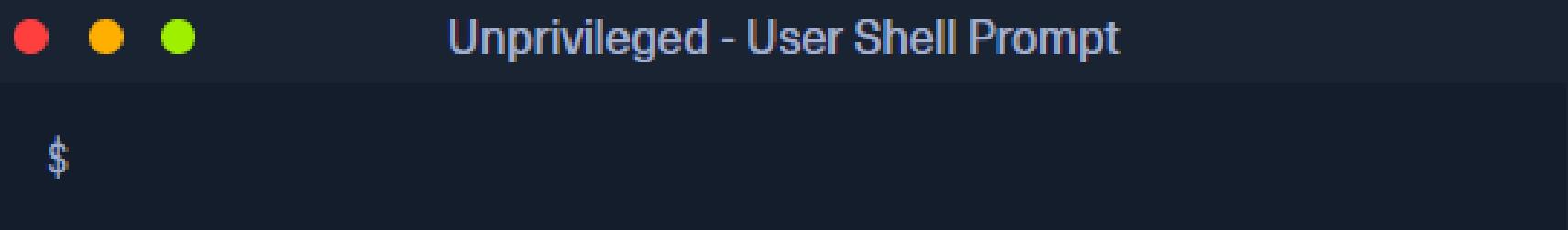
Ký hiệu dollar `<$>`, trong trường hợp này là viết tắt của tên user. Ngay sau khi chúng ta đăng nhập bằng quyền root, ký tự này sẽ chuyển thành ký hiệu hash `<#>` và trông như sau:



```
root@htb[/htb]#
```

Tuy nhiên, khi chúng ta tải lên và chạy shell trên một hệ thống mục tiêu, có thể sẽ không nhìn thấy username, hostname và directory hiện tại. Điều này có thể là do PS1 variable trong môi trường không được cài đặt chính xác. Trong trường hợp này, chúng ta sẽ thấy các prompt như sau:

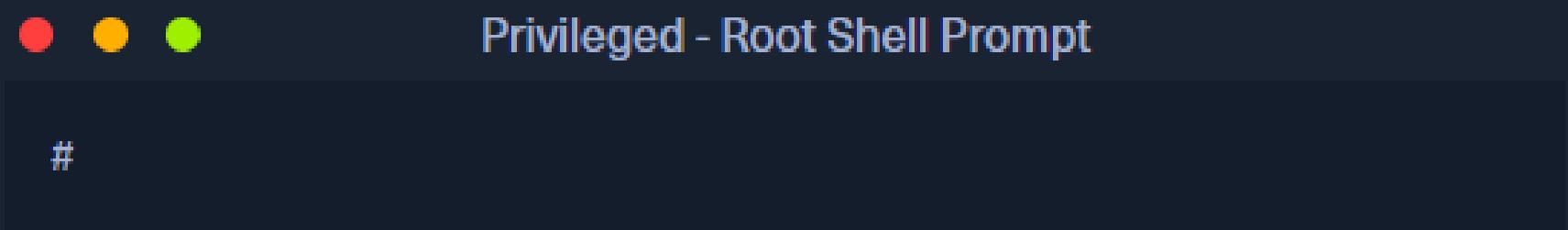
- Unprivileged - User Shell Prompt (Không có đặc quyền)



● ● ● **Unprivileged - User Shell Prompt**

\$

- Privileged - Root Shell Prompt (Có đặc quyền)



● ● ● **Privileged - Root Shell Prompt**

#

Việc chúng ta thấy ở đây giống như khi ta làm việc trên GUI của Windows. Chúng ta đăng nhập với một user trên máy tính và biết mình đang ở thư mục nào khi điều hướng. Bash prompt cũng có thể được tùy chỉnh và thay đổi theo nhu cầu của chúng ta. Việc điều chỉnh dấu nhắc nằm ngoài phạm vi nội dung phần này. Nhưng chúng ta có thể xem xét [bashrcgenerator](#) và [powerline](#), điều này cho phép khả năng điều chỉnh prompt theo nhu cầu của mình.

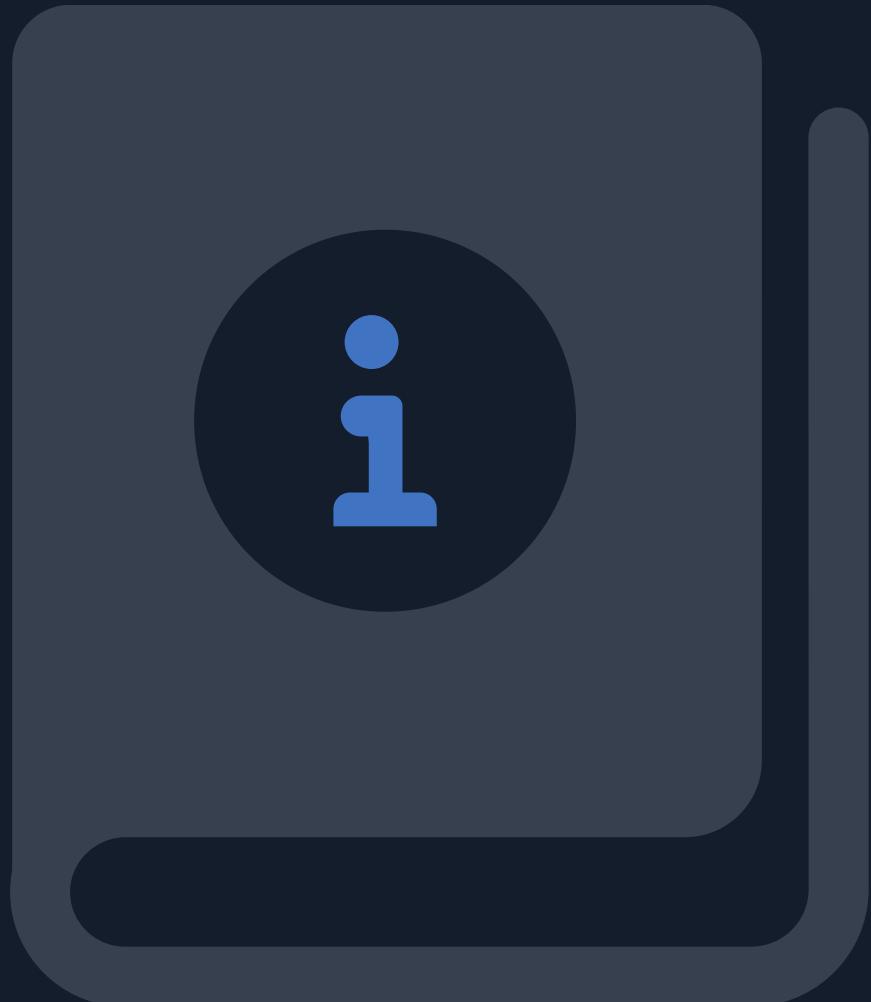
Nguyen Minh Hung  
Minhung



HACKTHEBOX

# Getting Help

#LinuxFundamentals #TheShell #Part4



Chúng ta sẽ luôn gặp phải các tool có các parameter tùy chọn mà ta không nhớ hay các tool chưa từng gặp trước đây. Do đó, điều quan trọng là phải biết cách để tự giúp mình làm quen với những tool này.

Hai cách đầu tiên là các trang manual và tính năng help:

- “**man**“ (manual) là một công cụ trợ giúp cung cấp tài liệu chi tiết về các lệnh, từ vựng và tùy chọn của các chương trình trong hệ thống. Tài liệu này được viết bằng ngôn ngữ chuyên môn và có thể khá dài.
- “**help**“ là một lệnh giúp cho người dùng xem thông tin trợ giúp cho các lệnh được cung cấp bởi shell. Nó thường hiển thị một danh sách ngắn gọn các lệnh có sẵn và mô tả tổng quan về chức năng của chúng. Thông tin trợ giúp này ít chi tiết hơn so với “man“ và có thể không đầy đủ hoặc chính xác.

Nếu bạn cần thông tin chi tiết và đầy đủ về một lệnh, hãy sử dụng “man“. Nếu bạn chỉ cần một tổng quan ngắn gọn về các lệnh, hãy sử dụng “help“.

# man

Syntax:

```
● ● ● Syntax:  
holdennguyen6174@htb[/htb]$ man <tool>
```

cùng xem qua ví dụ với tool curl:

```
● ● ● Example:  
holdennguyen6174@htb[/htb]$ man curl
```

```
curl(1)                                     Curl Manual  
  
NAME  
      curl - transfer a URL  
  
SYNOPSIS  
      curl [options] [URL...]  
  
DESCRIPTION  
      curl  is  a  tool  to  transfer  data  from  or  to  a  server,  using  one  of  the  supported  
      IMAP, IMAPS, LDAP, LDAPS, POP3, POP3S, RTMP, RTSP, SCP, SFTP, SMB, SMBS, SMTP  
      curl offers a busload of useful tricks like proxy support, user authentication, F1  
      curl  is  powered  by  libcurl  for  all  transfer-related  features.  See  libcurl(3)  for  
      Manual page curl(1) line 1 (press h for help or q to quit)
```

# --help

Syntax:



Syntax:

```
holdennguyen6174@htb[/htb]$ <tool> --help
```

ví dụ với tool curl:



Example:

```
holdennguyen6174@htb[/htb]$ curl --help
```

```
Usage: curl [options...] <url>
      --abstract-unix-socket <path> Connect via abstract Unix domain socket
      --anyauth           Pick any authentication method
      -a, --append         Append to target file when uploading
      --basic             Use HTTP Basic Authentication
      --cacert <file>    CA certificate to verify peer against
      --capath <dir>     CA directory to verify peer against
      -E, --cert <certificate[:password]> Client certificate file and password
<SNIP>
```

# -h

Chúng ta cũng có thể sử dụng phiên bản ngắn gọn của help

Syntax:



Syntax:

```
holdennguyen6174@htb[/htb]$ <tool> -h
```

ví dụ với tool curl:



Example:

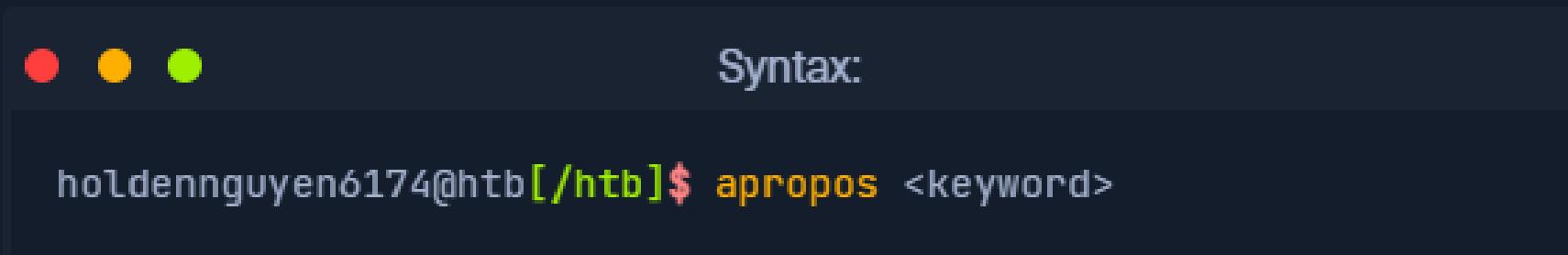
```
holdennguyen6174@htb[/htb]$ curl -h

Usage: curl [options...] <url>
      --abstract-unix-socket <path> Connect via abstract Unix domain socket
      --anyauth              Pick any authentication method
      -a, --append            Append to target file when uploading
      --basic                Use HTTP Basic Authentication
      --cacert <file>        CA certificate to verify peer against
      --capath <dir>          CA directory to verify peer against
      -E, --cert <certificate[:password]> Client certificate file and password
<SNIP>
```

# apropos

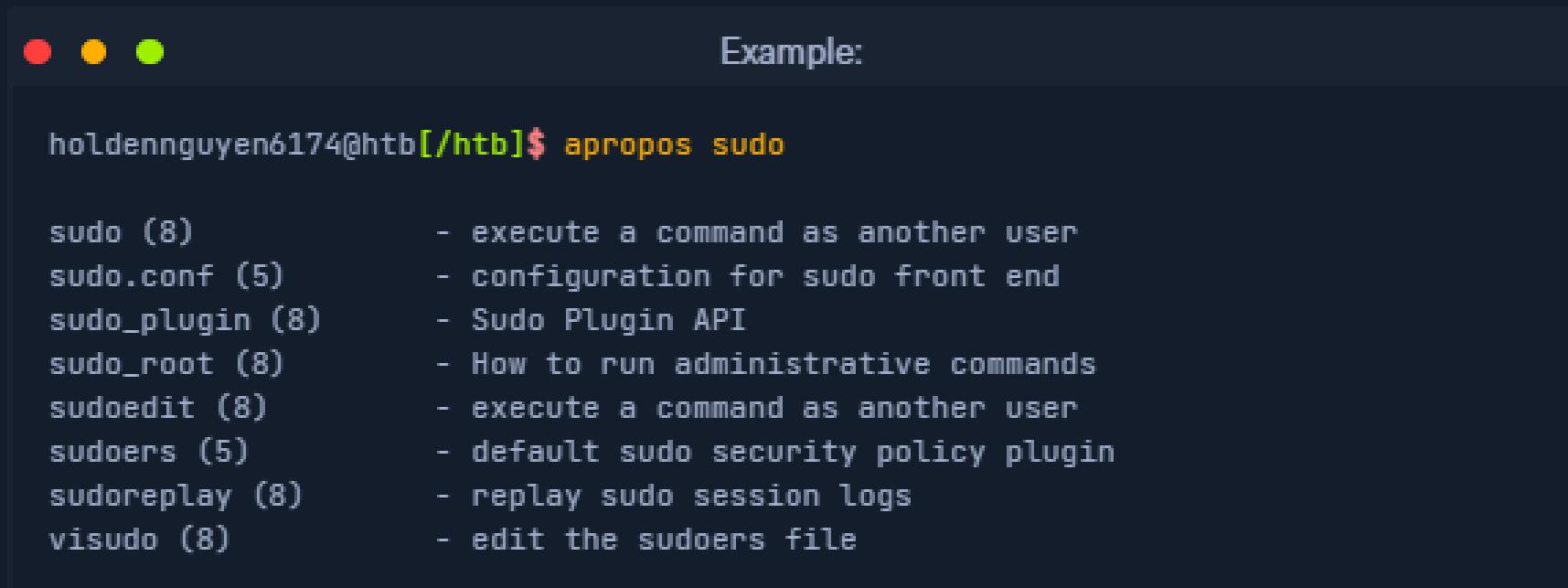
“apropos“ là một lệnh cho phép tìm kiếm tài liệu về các lệnh và chương trình trong hệ thống. Nó tìm kiếm từ khóa trong tên lệnh và mô tả của các lệnh rồi trả về danh sách các lệnh có liên quan đến từ khóa đó. Bạn có thể sử dụng “apropos“ để tìm kiếm một lệnh cụ thể hoặc tìm hiểu về một chức năng cụ thể. Kết quả tìm kiếm của “apropos“ có thể được sử dụng với “man“ để xem tài liệu chi tiết về các lệnh.

Syntax:



holdennguyen6174@htb[/htb]\$ apropos <keyword>

ví dụ với từ khóa sudo:



holdennguyen6174@htb[/htb]\$ apropos sudo

sudo (8)	- execute a command as another user
sudo.conf (5)	- configuration for sudo front end
sudo_plugin (8)	- Sudo Plugin API
sudo_root (8)	- How to run administrative commands
sudoedit (8)	- execute a command as another user
sudoers (5)	- default sudo security policy plugin
sudoreplay (8)	- replay sudo session logs
visudo (8)	- edit the sudoers file

Và còn một nguồn khá hữu dụng để tìm kiếm trợ giúp nếu như bạn gặp khó khăn trong việc hiểu một lệnh dài đó là:

The screenshot shows the explainshell.com website interface. At the top, there's a navigation bar with 'about' and a search bar containing 'curl -sL'. A 'theme' dropdown is also present. Below the search bar, the title 'curl(1)' is shown with a blue bracket underlining the '-sL' part. A callout box highlights the '-sL' option with the text: 'transfer a URL'. Another callout box provides detailed documentation for '-sL, --silent': 'Silent or quiet mode. Don't show progress meter or error messages. Makes Curl mute.' A third callout box details '-L, --location': '(HTTP/HTTPS) If the server reports that the requested page has moved to a different location (indicated with a Location: header and a 3XX response code), this option will make curl redo the request on the new place. If used together with -i, --include or -I, --head, headers from all requested pages will be shown. When authentication is used, curl only sends its credentials to the initial host. If a redirect takes curl to a different host, it won't be able to intercept the user+password. See also --location-trusted on how to change this. You can limit the amount of redirects to follow by using the --max-redirs option.' It also notes that when curl follows a redirect and the request is not a plain GET, it will do the following request with a GET if the HTTP response was 301, 302, or 303. If the response code was any other 3xx code, curl will re-send the following request using the same unmodified method. At the bottom of the page, there's a link 'source manpages: curl'.

<https://explainshell.com/>

Nguyen Minh Hung  
Minhung



HACKTHEBOX

# System Information

#LinuxFundamentals #TheShell #Part5



Để làm việc với nhiều hệ thống Linux khác nhau, chúng ta cần tìm hiểu cấu trúc và các thông tin về hệ thống, tiến trình, cấu hình... và các tham số tương ứng. Dưới đây là danh sách các tool cần thiết sẽ giúp chúng ta có được những thông tin trên. (Hầu hết chúng đều sẽ được cài đặt theo mặc định)

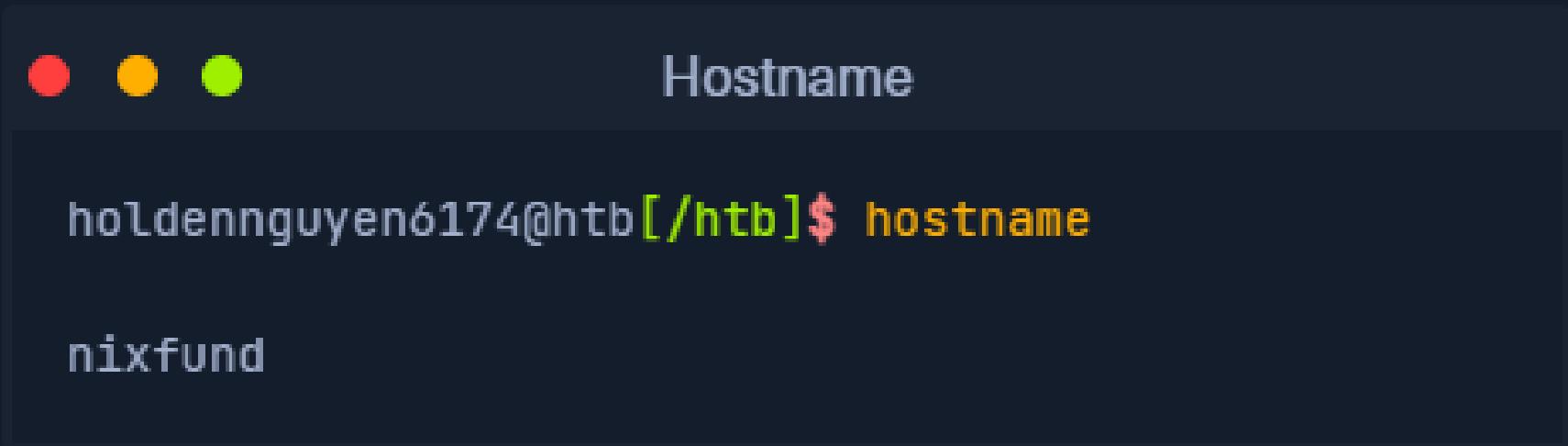
- **whoami**: hiển thị username hiện tại.
- **id**: trả về thông tin chi tiết của user (uid, gid, groups...)
- **hostname**: trả về tên host của máy tính đang sử dụng.
- **uname**: xuất thông tin của hệ điều hành và phần cứng.
- **pwd**: hiển thị tên directory hiện tại.
- **ifconfig**: dùng để xem hoặc gán một địa chỉ cho network interface hoặc cấu hình các tham số network interface.
- **ip**: dùng để quản lý hoặc xem thông tin về mạng. Cung cấp giao diện dễ dàng và nhiều tính năng hơn so với “ifconfig” để xem và cấu hình các network interface, địa chỉ IP, route table, và các thiết lập mạng khác.

- **netstat**: hiển thị trạng thái mạng, network status.
- **ss**: giúp điều tra kết nối mạng đang hoạt động (sockets).
- **ps**: hiển thị trạng thái các process đang hoạt động.
- **who**: xem danh sách user đã đăng nhập vào hệ thống.
- **env**: xem danh sách các biến môi trường và giá trị của nó.
- **lsblk**: liệt kê các thiết bị lưu trữ block như đĩa cứng, usb, dvd... đang kết nối với hệ thống.
- **lsusb**: liệt kê các thiết bị usb đang kết nối với hệ thống.
- **lsof**: liệt kê các file đang được mở bởi các process.
- **lspci**: liệt kê các thiết bị trên bus PCI (Peripheral Component Interconnect) của máy tính.

Cùng xem qua một số ví dụ cụ thể sau đây:

# hostname

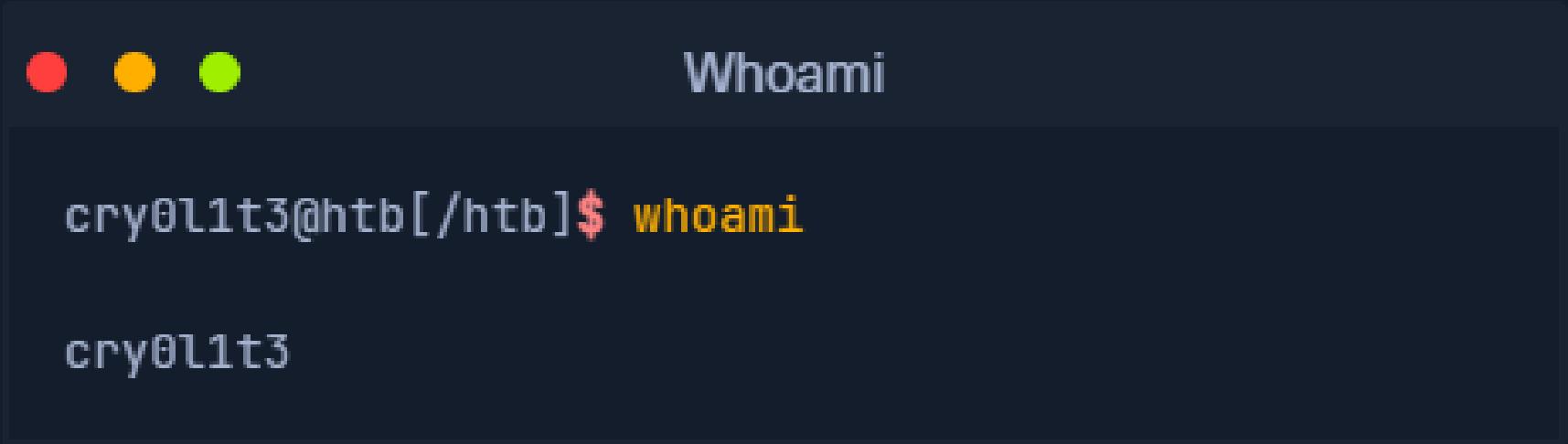
Lệnh **hostname** đơn giản chỉ trả về tên của máy tính hay host mà chúng ta đã đăng nhập



```
● ● ● Hostname  
holdennguyen6174@htb[/htb]$ hostname  
nixfund
```

# whoami

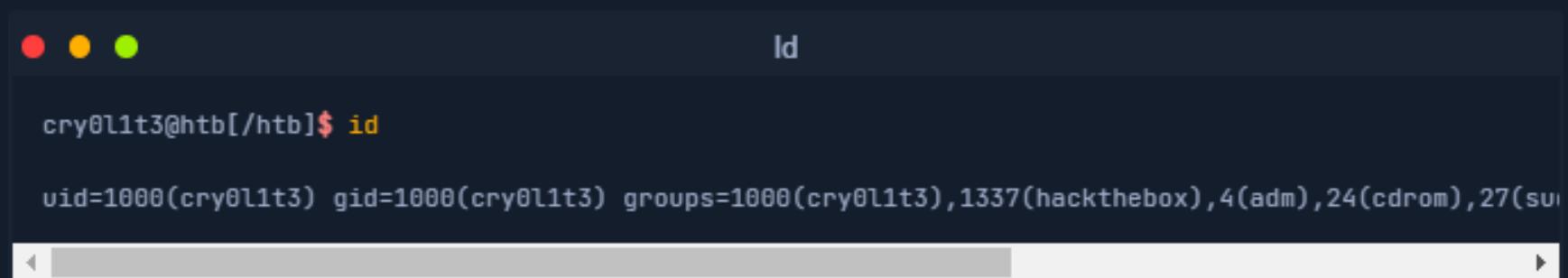
Lệnh này có thể được sử dụng cả trên Windows lẫn Linux để xác định tên của user hiện tại. Trong quá trình đánh giá bảo mật, chúng ta có thể sẽ sử dụng **reverse shell** trên một host, và điều đầu tiên cần làm là xem mình đang sử dụng user nào. Từ đó tìm hiểu xem user này có đặc quyền hay quyền truy cập nào đặc biệt không.



```
● ● ● Whoami  
cry0l1t3@htb[/htb]$ whoami  
cry0l1t3
```

# id

Lệnh **id** mở rộng hơn **whoami**, xuất ra các group membership và ID. Điều này có thể hữu ích đối với việc kiểm tra xâm nhập, xem user có những quyền truy cập nào và giúp system admin kiểm tra quyền của user và group.



```
● ● ●
cry0l1t3@htb[~/htb]$ id
uid=1000(cry0l1t3) gid=1000(cry0l1t3) groups=1000(cry0l1t3),1337(hackthebox),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),116(lpadmin),126(sambashare)
```

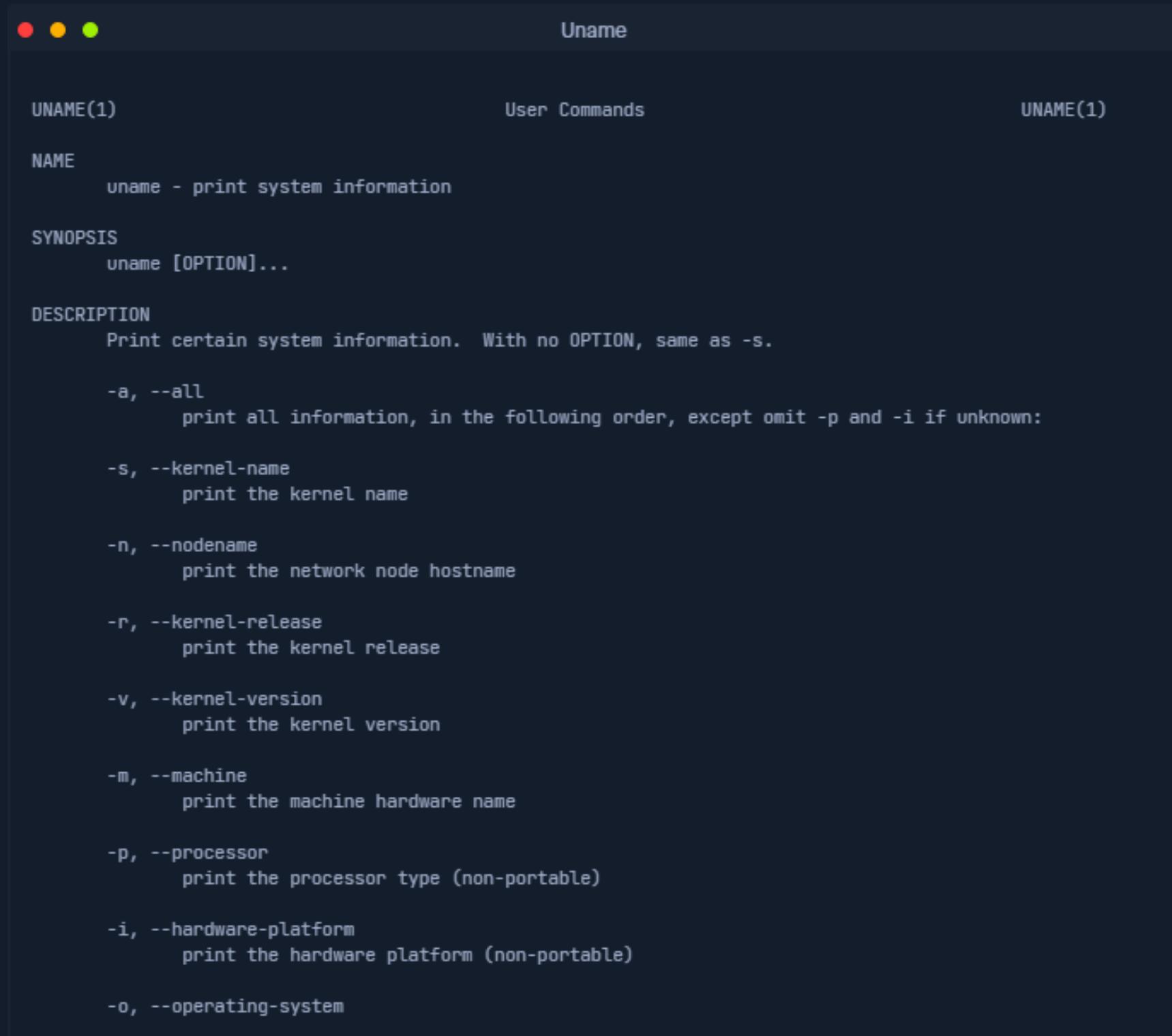
Như trong ví dụ trên, với output:

```
uid=1000(cry0l1t3) gid=1000(cry0l1t3)
groups=1000(cry0l1t3),1337(hackthebox),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),116(lpadmin),126(sambashare)
```

- **hackthebox** sẽ là group bất thường cần lưu ý
- **adm** group đồng nghĩa việc user có thể đọc log files trong **/var/log** và có khả năng cấp quyền cho những thông tin nhạy cảm.
- thuộc group **sudo** sẽ giúp user này chạy một số hoặc tất cả các lệnh dưới quyền **root** user. Đặc quyền sudo sẽ khiến user này thêm quyền hoặc có toàn quyền quản lý như admin, và cần phải được kiểm tra tư cách để xóa những quyền truy cập không cần thiết nếu đây chỉ là một user bình thường.

# uname

Hãy tìm hiểu sâu hơn về lệnh **uname**. Nếu chúng ta sử dụng **man uname** trong terminal, sẽ hiển thị trang manual của lệnh **uname**, từ đó hiển thị các option khả thi của command:

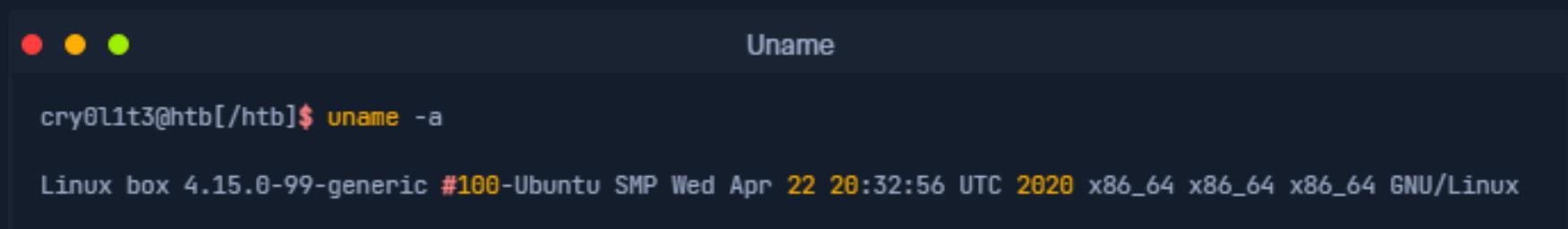


The screenshot shows a terminal window with a dark background and light-colored text. The title bar says "Uname". The window contains the man page for the "uname" command. The sections shown are:

- NAME**: `uname - print system information`
- SYNOPSIS**: `uname [OPTION]...`
- DESCRIPTION**:
  - `-a, --all`: print all information, in the following order, except omit `-p` and `-i` if unknown:
  - `-s, --kernel-name`: print the kernel name
  - `-n, --nodename`: print the network node hostname
  - `-r, --kernel-release`: print the kernel release
  - `-v, --kernel-version`: print the kernel version
  - `-m, --machine`: print the machine hardware name
  - `-p, --processor`: print the processor type (non-portable)
  - `-i, --hardware-platform`: print the hardware platform (non-portable)
  - `-o, --operating-system`

# uname

Chạy `uname -a` sẽ in ra tất cả thông tin về máy tính/host theo thứ tự cụ thể: kernel name, hostname, kernel release, kernel version, machine hardware name và operating system. `-a` flag sẽ bỏ qua `-p` (processor type) và `-i` (hardware platform) nếu chúng không xác định.



```
cry0l1t3@htb[~/htb]$ uname -a
Linux box 4.15.0-99-generic #100-Ubuntu SMP Wed Apr 22 20:32:56 UTC 2020 x86_64 x86_64 x86_64 GNU/Linux
```

Trong output ở ví dụ trên:

Linux box 4.15.0-99-generic #100-Ubuntu SMP Wed Apr 22  
20:32:56 UTC 2020 x86\_64 x86\_64 x86\_64 GNU/Linux

- kernel name là `Linux`.
- hostname là `box`.
- kernel release là `4.15.0-99-generic`.
- kernel version là `#100-Ubuntu SMP Wed Apr 22 20:32:56 UTC 2020...`

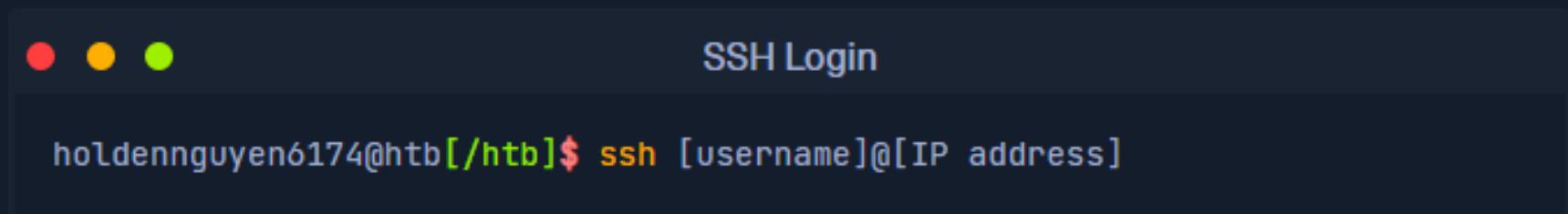
Chạy `uname` với mỗi option trong manual page sẽ cung cấp những output cụ thể có thông tin mà chúng ta quan tâm.

# Logging In via SSH

Secure Shell (SSH) đề cập đến một giao thức cho phép client truy cập và thực hiện các lệnh trên các máy tính từ xa.

Trên các host và server dựa trên Linux hoặc chạy hệ điều hành giống như Unix, SSH là một trong những công cụ tiêu chuẩn luôn được cài đặt và là lựa chọn ưa thích của nhiều administrator để cấu hình và quản trị máy tính thông qua truy cập từ xa.

Nó là một giao thức lâu đời và đã được công nhận đồng thời không đòi hỏi một giao diện đồ họa cho người dùng (GUI). Ví thế nó hoạt động rất hiệu quả và chiếm ít tài nguyên.



Nguyen Minh Hung  
Minhung



HACKTHEBOX

# User Management

#LinuxFundamentals #SystemManagement #Part6



Quản lý người dùng là một phần thiết yếu của việc quản trị hệ thống Linux. Đôi khi chúng ta cần tạo user mới hoặc thêm user khác vào một group cụ thể. Một khả năng khác là thực thi các lệnh với tư cách là một người dùng khác.

Sẽ luôn gặp các yêu cầu về user của chỉ một group riêng mới có quyền (permission) xem hoặc chỉnh sửa các file hay directory cụ thể.

Cùng xem qua một ví dụ về cách chạy lệnh bằng một user khác:



### Execution as a user

```
holdennguyen6174@htb[/htb]$ cat /etc/shadow
```

```
cat: /etc/shadow: Permission denied
```



### Execution as root

```
holdennguyen6174@htb[/htb]$ sudo cat /etc/shadow
```

```
root:<SNIP>:18395:0:99999:7:::  
daemon:*:17737:0:99999:7:::  
bin:*:17737:0:99999:7:::  
<SNIP>
```

Dưới đây là danh sách các lệnh giúp bạn hiểu rõ hơn việc xử lý, quản lý người dùng:

- **sudo**: cho phép user chạy lệnh với quyền cao hơn, thường là quyền **root**.
- **su**: tiện ích yêu cầu thông tin đăng nhập phù hợp của user thông qua **PAM** và chuyển sang ID user đó (user mặc định sẽ là superuser) để chạy shell.
- **useradd**: tạo hoặc cập nhật thông tin cho user .
- **userdel**: xóa user cùng các file liên quan.
- **usermod**: thay đổi thông tin của user.
- **addgroup**: tạo một group mới trên hệ thống.
- **delgroup**: xóa một group khỏi hệ thống.
- **passwd**: thay đổi mật khẩu của user.

Nguyen Minh Hung  
Minhung



HACKTHEBOX

# Package Management

#LinuxFundamentals #SystemManagement #Part7



Dù bạn làm việc với tư cách một system administrator, sử dụng máy chạy Linux ở nhà hay xây dựng/nâng cấp/quản trị bản phân phối kiểm thử xâm nhập (**pentest**), điều quan trọng luôn là nắm vững các Linux package manager có sẵn và các cách khác nhau để sử dụng chúng cài đặt, cập nhật hoặc gỡ bỏ package. **Package** là kho lưu trữ các file nhị phân của phần mềm, configuration file, thông tin về các **dependency** và theo dõi các bản cập nhật, nâng cấp. Một package cần phải sử dụng các package khác để hoạt động đúng cách. Trên Linux, ứng dụng hay phần mềm đều sẽ được cài đặt từ các package.

Các tính năng mà hầu hết mọi **package management system** cung cấp là:

- Tải package.
- Dependency resolution (tìm kiếm và cài đặt chính xác các mối quan hệ giữa các package để phần mềm có thể hoạt động).
- Định dạng tiêu chuẩn cho các binary package.
- Xác định vị trí cài đặt và cấu hình chung.
- Thiết lập cấu hình và tính năng bổ sung liên quan đến hệ thống.
- Kiểm soát chất lượng.

Chúng ta có thể sử dụng nhiều công cụ package management khác nhau với các loại file như “.deb”, “.rpm”... Yêu cầu của package management là phần mềm được cài đặt phải có sẵn dưới dạng package tương ứng. Thông thường, package management system được tạo, cung cấp và duy trì tập trung trong các [Linux distro](#). Theo cách này, phần mềm được tích hợp trực tiếp vào hệ thống và các directory khác nhau của nó sẽ được phân phối trên toàn hệ thống.

Công cụ package management thay đổi hệ thống để cài đặt package lấy từ package management system, nếu nó nhận ra cần thêm các package bổ sung để package chuẩn bị cài đặt có thể hoạt động bình thường thì một phần phụ thuộc (dependency) sẽ được đưa vào và cảnh báo administrator hoặc cố gắng tải lại phần mềm bị thiếu từ kho lưu trữ (repository) để cài đặt trước.

Nếu một phần mềm đã cài đặt bị xóa, công cụ package management sẽ lấy lại thông tin của package, sửa đổi nó dựa trên configuration và xóa các file.

Dưới đây là danh sách ví dụ về các chương trình như vậy:

- **dpkg**: Debian Package - một công cụ để cài đặt, xây dựng, gỡ bỏ và quản lý các package. Đây là công cụ cấp thấp dạng nhị phân (low-level) sử dụng trong các Linux distro dựa trên Debian.
- **apt**: Advanced Packaging Tool - là một công cụ cấp cao (high-level) xây dựng từ **dpkg** và được sử dụng nhiều hơn vì có thể tìm các package từ xa và giải quyết các package có mối quan hệ phức tạp (dependency resolution).
- **aptitude**: một front-end cho **apt**, với giao diện và một số tính năng bổ sung.
- **snap**: cài đặt, cấu hình, làm mới và xóa các snap package. Phân phối an toàn các ứng dụng và tiện ích cho cloud, server, desktop và IoT.
- **pip**: là Python package installer được khuyên dùng để cài đặt các Python package không có sẵn trong Debian. Có thể hoạt động với các version control repository (**Git**, **Mercurial** và **Bazaar**), output nhiều log hoặc ghi log ra file và tránh việc cài đặt lở dở bằng cách tải xuống tất cả các yêu cầu trước khi tiến hành cài đặt.
- **git**: một **version control system** cung cấp high-level operation và full access bên trong git package.

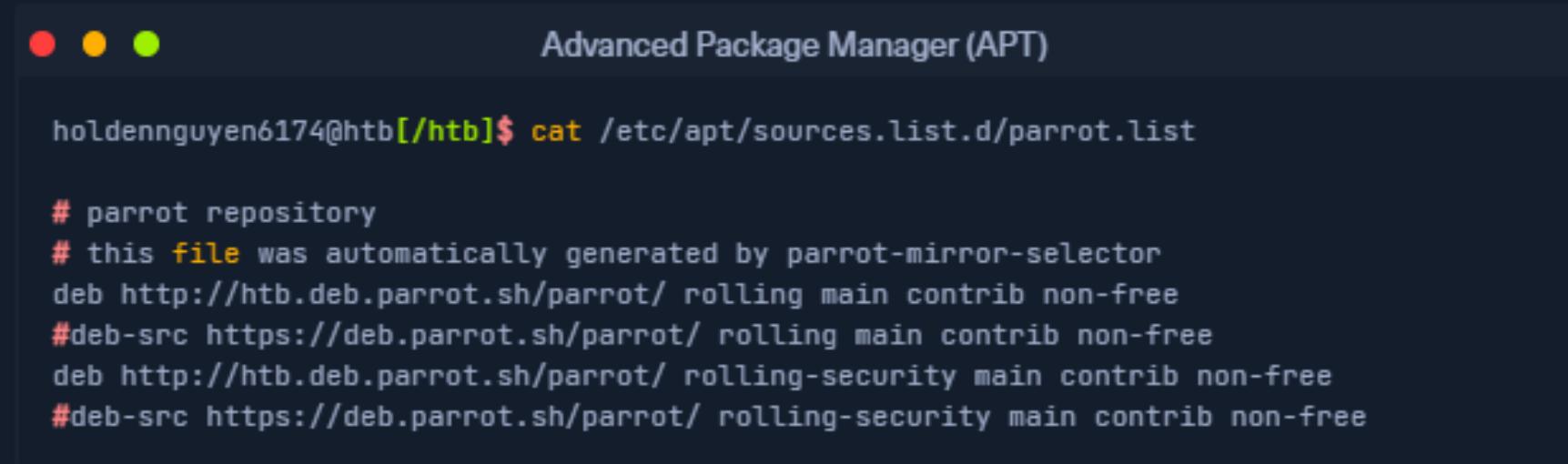
# APT

Các Linux distro dựa trên Debian sử dụng **apt** package manager. Một package là một file lưu trữ nhiều “.deb“ file. Tiện ích **dpkg** được sử dụng để cài đặt các chương trình từ “.deb“ file được liên kết. **Apt** làm cho việc cập nhật và cài đặt chương trình dễ dàng hơn vì nhiều chương trình có thêm **dependencies** (các mối quan hệ phụ thuộc vào package khác). Khi cài đặt chương trình từ file “.deb“ độc lập, chúng ta có thể gặp sự cố dependency và cần tải cũng như cài đặt một hoặc nhiều package bổ sung. **Apt** làm cho việc này trở nên dễ dàng và hiệu quả hơn bằng cách đóng gói (packaging) tất cả các dependencies cần thiết để cài đặt một chương trình.

Các Linux distro đều sử dụng các kho phần mềm (software repositories) để cập nhật thường xuyên. Khi chúng ta cập nhật một chương trình hoặc cài đặt một chương trình mới, hệ thống sẽ truy vấn đến các repository này để tìm package mong muốn. Các repository có thể được dán nhãn là ổn định (stable), thử nghiệm (testing) hoặc chưa ổn định (unstable). Hầu hết Linux distro sẽ sử dụng “main“ repository hoặc repository ổn định nhất. Điều này có thể được kiểm tra bằng cách xem nội dung của file **/etc/apt/sources.list**.

# APT

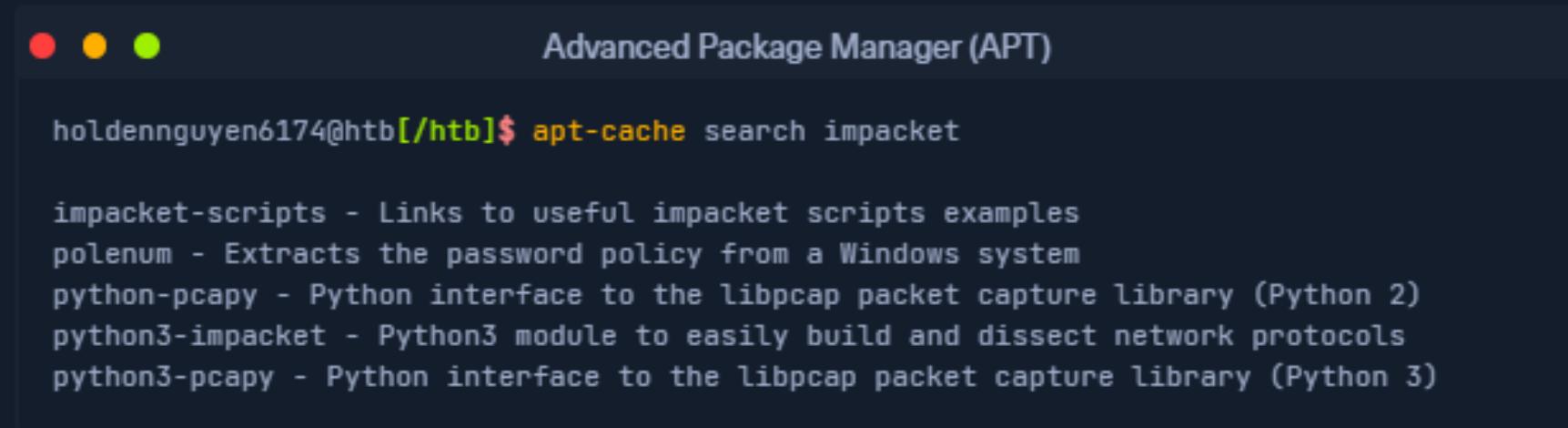
Danh sách repository của Parrot OS ở  
`/etc/apt/sources.list.d/parrot.list`



```
holdennguyen6174@htb[/htb]$ cat /etc/apt/sources.list.d/parrot.list

# parrot repository
# this file was automatically generated by parrot-mirror-selector
deb http://htb.deb.parrot.sh/parrot/ rolling main contrib non-free
#deb-src https://deb.parrot.sh/parrot/ rolling main contrib non-free
deb http://htb.deb.parrot.sh/parrot/ rolling-security main contrib non-free
#deb-src https://deb.parrot.sh/parrot/ rolling-security main contrib non-free
```

apt sử dụng cơ sở dữ liệu (database) gọi là apt cache, cung cấp thông tin về các package đã cài đặt trên hệ thống khi ngoại tuyến (offline). Ví dụ chúng ta có thể dùng `apt-cache` để tìm tất cả các package liên quan đến `impacket`.



```
holdennguyen6174@htb[/htb]$ apt-cache search impacket

impacket-scripts - Links to useful impacket scripts examples
polenum - Extracts the password policy from a Windows system
python-pcap - Python interface to the libpcap packet capture library (Python 2)
python3-impacket - Python3 module to easily build and dissect network protocols
python3-pcap - Python interface to the libpcap packet capture library (Python 3)
```

# APT

Chúng ta có thể xem thông tin bổ sung về một package:

```
● ● ● Advanced Package Manager (APT)

holdennguyen6174@htb[/htb]$ apt-cache show impacket-scripts

Package: impacket-scripts
Version: 1.4
Architecture: all
Maintainer: Kali Developers <devel@kali.org>
Installed-Size: 13
Depends: python3-impacket (>= 0.9.20), python3-ldap3 (>= 2.5.0), python3-ldapdomaindump
Breaks: python-impacket (<< 0.9.18)
Replaces: python-impacket (<< 0.9.18)
Priority: optional
Section: misc
Filename: pool/main/i/impacket-scripts/impacket-scripts_1.4_all.deb
Size: 2080
<SNIP>
```

Hay liệt kê tất cả các package đã được cài đặt:

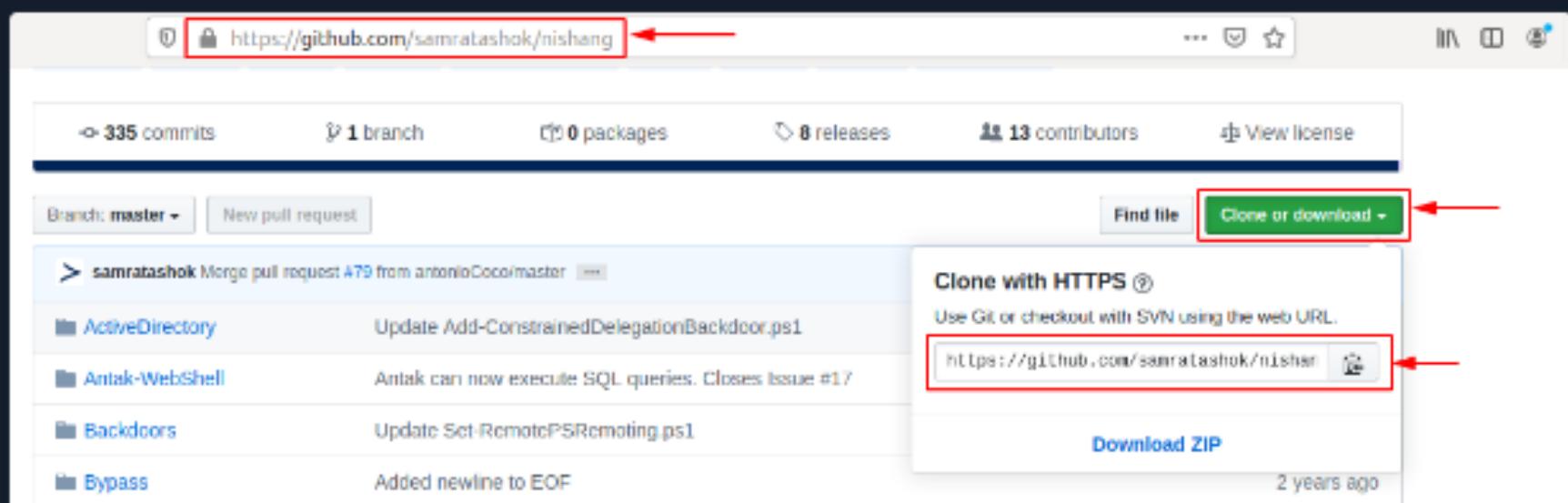
```
● ● ● Advanced Package Manager (APT)

holdennguyen6174@htb[/htb]$ apt list --installed

Listing... Done
accountsservice/rolling,now 0.6.55-2 amd64 [installed,automatic]
adapta-gtk-theme/rolling,now 3.95.0.11-1 all [installed]
adduser/rolling,now 3.118 all [installed]
adwainita-icon-theme/rolling,now 3.36.1-2 all [installed,automatic]
aircrack-ng/rolling,now 1:1.6-4 amd64 [installed,automatic]
<SNIP>
```

# Git

Khi đã cài đặt **git**, chúng ta có thể sử dụng nó để tải các tool từ Github. Đầu tiên cần truy cập đến repository của dự án và sao chép **Github link** trước khi sử dụng git để tải xuống.



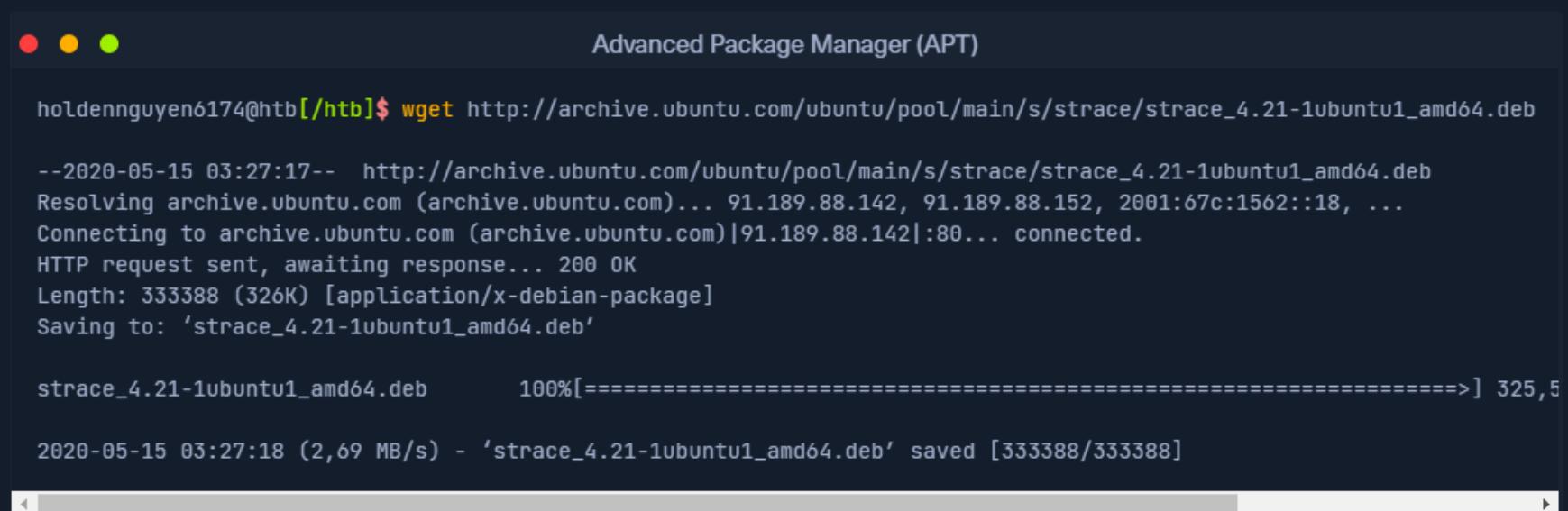
Trước khi tải xuống dự án, các script, list của nó nên tạo một thư mục cụ thể để chứa tất cả.

```
holdennguyen6174@htb[/htb]$ mkdir ~/nishang/ && git clone https://github.com/samratashok/nishang.git ~/nishang

Cloning into '/opt/nishang/'...
remote: Enumerating objects: 15, done.
remote: Counting objects: 100% (15/15), done.
remote: Compressing objects: 100% (13/13), done.
remote: Total 1691 (delta 4), reused 6 (delta 2), pack-reused 1676
Receiving objects: 100% (1691/1691), 7.84 MiB | 4.86 MiB/s, done.
Resolving deltas: 100% (1055/1055), done.
```

# DPKG

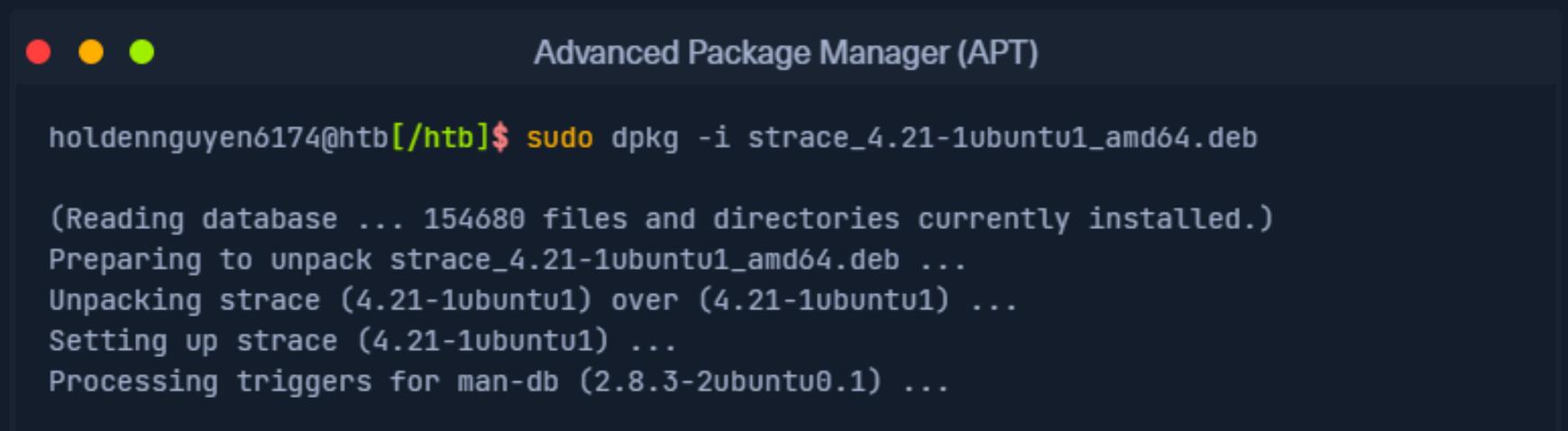
Chúng ta có thể tải xuống riêng các program và tool từ các repository. Ví dụ này sẽ tải “**strace**“ cho Ubuntu 18.04 LTS



```
holdennguyen6174@htb[/htb]$ wget http://archive.ubuntu.com/ubuntu/pool/main/s/strace/strace_4.21-1ubuntu1_amd64.deb
--2020-05-15 03:27:17-- http://archive.ubuntu.com/ubuntu/pool/main/s/strace/strace_4.21-1ubuntu1_amd64.deb
Resolving archive.ubuntu.com (archive.ubuntu.com)... 91.189.88.142, 91.189.88.152, 2001:67c:1562::18, ...
Connecting to archive.ubuntu.com (archive.ubuntu.com)|91.189.88.142|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 333388 (326K) [application/x-debian-package]
Saving to: 'strace_4.21-1ubuntu1_amd64.deb'

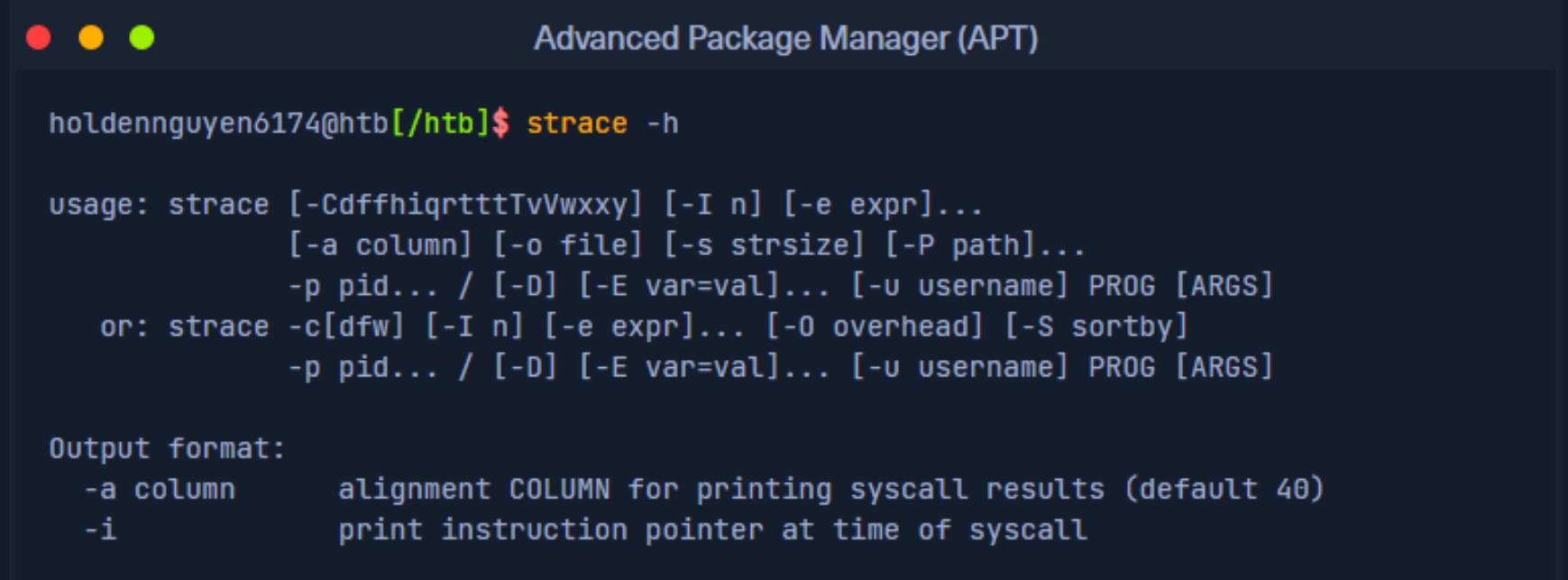
strace_4.21-1ubuntu1_amd64.deb      100%[=====] 325,5
2020-05-15 03:27:18 (2,69 MB/s) - 'strace_4.21-1ubuntu1_amd64.deb' saved [333388/333388]
```

Tương tự **apt**, chúng ta có thể sử dụng **dpkg** cho ví dụ này:



```
holdennguyen6174@htb[/htb]$ sudo dpkg -i strace_4.21-1ubuntu1_amd64.deb
(Reading database ... 154680 files and directories currently installed.)
Preparing to unpack strace_4.21-1ubuntu1_amd64.deb ...
Unpacking strace (4.21-1ubuntu1) over (4.21-1ubuntu1) ...
Setting up strace (4.21-1ubuntu1) ...
Processing triggers for man-db (2.8.3-2ubuntu0.1) ...
```

và tool đã được cài xong và có thể kiểm tra hoạt động:



```
holdennguyen6174@htb[/htb]$ strace -h
usage: strace [-CdfhhqrrttTvvVwxxxy] [-I n] [-e expr]...
               [-a column] [-o file] [-s strsize] [-P path]...
               -p pid... / [-D] [-E var=val]... [-u username] PROG [ARGS]
or: strace -c[dfw] [-I n] [-e expr]... [-O overhead] [-S sortby]
               -p pid... / [-D] [-E var=val]... [-u username] PROG [ARGS]

Output format:
-a column      alignment COLUMN for printing syscall results (default 40)
-i              print instruction pointer at time of syscall
```

Nguyen Minh Hung  
Minhung



HACKTHEBOX

# Service & Process Management

#LinuxFundamentals #SystemManagement #Part8



Nhìn chung có hai loại service:

- Internal, các service liên quan được yêu cầu khi khởi động hệ thống, chẳng hạn như thực hiện các tác vụ liên quan đến hardware.
- Service do người dùng cài đặt, thường bao gồm tất cả các server service.

Các service này chạy dưới background và không có bất kỳ tương tác nào với user. Chúng còn được gọi là **daemon** và được xác định bằng chữ “d” ở cuối tên chương trình.

Ví dụ: **sshd** hoặc **systemd**...

Hầu hết các Linux distro hiện đã chuyển sang **systemd**, **daemon** này là một **Init process** chạy đầu tiên và cũng bởi thế nên có **process ID (PID) = 1**. Daemon này giám sát và đảm nhận việc khởi động, dừng các service khác. Tất cả các process đều có một PID được chỉ định có thể xem trong **/proc/** với số tương ứng. Một process như vậy có thể có **parent process ID**, và nếu có, process đó sẽ được gọi là **child process**.

Bên cạnh **systemctl**, chúng ta có thể sử dụng **update-rc.d** để quản lý các **SysV init script link**. Hãy xem qua một số ví dụ. Chúng ta sẽ sử dụng **OpenSSH** server trong các ví dụ này. (Nếu máy chưa có OpenSSH, bạn hãy cài đặt để có thể thực hành)

# Systemctl

Sau khi cài đặt OpenSSH, chúng ta có thể chạy service với câu lệnh như sau:



```
holdennguyen6174@htb[/htb]$ systemctl start ssh
```

Sau khi khởi động, kiểm tra xem service đang chạy có bị lỗi hay không:



```
holdennguyen6174@htb[/htb]$ systemctl status ssh
```

- ssh.service - OpenBSD Secure Shell server
   
Loaded: loaded (/lib/systemd/system/ssh.service; enabled; vendor preset: enabled)
   
Active: active (running) since Thu 2020-05-14 15:08:23 CEST; 24h ago
   
Main PID: 846 (sshd)
   
Tasks: 1 (limit: 4681)
   
CGroup: /system.slice/ssh.service
 └─846 /usr/sbin/sshd -D

```
Mai 14 15:08:22 inlane systemd[1]: Starting OpenBSD Secure Shell server...
Mai 14 15:08:23 inlane sshd[846]: Server listening on 0.0.0.0 port 22.
Mai 14 15:08:23 inlane sshd[846]: Server listening on :: port 22.
Mai 14 15:08:23 inlane systemd[1]: Started OpenBSD Secure Shell server.
Mai 14 15:08:30 inlane systemd[1]: Reloading OpenBSD Secure Shell server.
Mai 14 15:08:31 inlane sshd[846]: Received SIGHUP; restarting.
Mai 14 15:08:31 inlane sshd[846]: Server listening on 0.0.0.0 port 22.
Mai 14 15:08:31 inlane sshd[846]: Server listening on :: port 22.
```

Thêm OpenSSH vào **SysV script** để hệ thống chạy service này ngay sau khi khởi động bằng cách link nó với lệnh sau:



```
holdennguyen6174@htb[/htb]$ systemctl enable ssh
```

```
Synchronizing state of ssh.service with SysV service script with /lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable ssh
```

# Systemctl

Sau khi reboot lại system, OpenSSH server sẽ tự động được chạy. Chúng ta có thể kiểm tra bằng tool gọi là **ps**:

```
holdennguyen6174@htb[/htb]$ ps -aux | grep ssh
root      846  0.0  0.1  72300  5660 ?        Ss   Mai14  0:00 /usr/sbin/sshd -D
```

Cũng có thể sử dụng **systemctl** để liệt kê toàn bộ service:

```
holdennguyen6174@htb[/htb]$ systemctl list-units --type=service
UNIT                                     LOAD ACTIVE SUB     DESCRIPTION
accounts-daemon.service                 loaded active running Accounts Service
acpid.service                           loaded active running ACPI event daemon
apache2.service                         loaded active running The Apache HTTP Server
apparmor.service                        loaded active exited AppArmor initialization
apport.service                          loaded active exited LSB: automatic crash repor
avahi-daemon.service                   loaded active running Avahi mDNS/DNS-SD Stack
bolt.service                            loaded active running Thunderbolt system service
```

Có thể service sẽ không chạy do gặp lỗi. Để xem vấn đề, chúng ta sử dụng tool **journalctl** để xem log:

```
holdennguyen6174@htb[/htb]$ journalctl -u ssh.service --no-pager
-- Logs begin at Wed 2020-05-13 17:30:52 CEST, end at Fri 2020-05-15 16:00:14 CEST. --
Mai 13 20:38:44 inlane systemd[1]: Starting OpenBSD Secure Shell server...
Mai 13 20:38:44 inlane sshd[2722]: Server listening on 0.0.0.0 port 22.
Mai 13 20:38:44 inlane sshd[2722]: Server listening on :: port 22.
Mai 13 20:38:44 inlane systemd[1]: Started OpenBSD Secure Shell server.
Mai 13 20:39:06 inlane sshd[3939]: Connection closed by 10.22.2.1 port 36444 [preauth]
Mai 13 20:39:27 inlane sshd[3942]: Accepted password for master from 10.22.2.1 port 36452 ssh2
Mai 13 20:39:27 inlane sshd[3942]: pam_unix(sshd:session): session opened for user master by (uid=0)
Mai 13 20:39:28 inlane sshd[3942]: pam_unix(sshd:session): session closed for user master
Mai 14 02:04:49 inlane sshd[2722]: Received signal 15; terminating.
Mai 14 02:04:49 inlane systemd[1]: Stopping OpenBSD Secure Shell server...
Mai 14 02:04:49 inlane systemd[1]: Stopped OpenBSD Secure Shell server.
-- Reboot --
```

# Kill a process

Một process có thể ở các trạng thái sau:

- Running.
- Waiting. (chờ một event hoặc system resource)
- Stopped.
- Zombie. (stopped nhưng vẫn là entry trong process table)

Các process có thể điều khiển bằng kill, pkill, pgrep và killall.

Để tương tác với một process, chúng ta phải gửi một tín hiệu tới nó. Có thể xem tất cả các tín hiệu bằng lệnh sau:

```
holdennguyen6174@htb[/htb]$ kill -l
 1) SIGHUP      2) SIGINT      3) SIGQUIT     4) SIGILL      5) SIGTRAP
 6) SIGABRT    7) SIGBUS      8) SIGFPE      9) SIGKILL    10) SIGUSR1
 11) SIGSEGV    12) SIGUSR2     13) SIGPIPE     14) SIGALRM    15) SIGTERM
 16) SIGSTKFLT   17) SIGCHLD     18) SIGCONT     19) SIGSTOP    20) SIGTSTP
 21) SIGTTIN    22) SIGTTOU     23) SIGURG      24) SIGXCPU    25) SIGXFSZ
 26) SIGVTALRM   27) SIGPROF     28) SIGWINCH    29) SIGIO      30) SIGPWR
 31) SIGSYS     34) SIGRTMIN   35) SIGRTMIN+1  36) SIGRTMIN+2  37) SIGRTMIN+3
 38) SIGRTMIN+4  39) SIGRTMIN+5  40) SIGRTMIN+6  41) SIGRTMIN+7  42) SIGRTMIN+8
 43) SIGRTMIN+9  44) SIGRTMIN+10 45) SIGRTMIN+11 46) SIGRTMIN+12 47) SIGRTMIN+13
 48) SIGRTMIN+14 49) SIGRTMIN+15 50) SIGRTMAX-14 51) SIGRTMAX-13 52) SIGRTMAX-12
 53) SIGRTMAX-11 54) SIGRTMAX-10 55) SIGRTMAX-9  56) SIGRTMAX-8  57) SIGRTMAX-7
 58) SIGRTMAX-6  59) SIGRTMAX-5  60) SIGRTMAX-4  61) SIGRTMAX-3  62) SIGRTMAX-2
 63) SIGRTMAX-1  64) SIGRTMAX
```

Ví dụ nếu một chương trình bị đơ, chúng ta có thể cưỡng ép kill nó với lệnh sau:

```
holdennguyen6174@htb[/htb]$ kill 9 <PID>
```

# Background a Process

Đôi khi cần phải chuyển process vừa chạy về chế độ background để tiếp tục sử dụng session hiện tại để tương tác với system hoặc bắt đầu các process khác. Chúng ta có thể thực hiện việc này bằng phím tắt [Ctrl + Z]. Như đã đề cập ở trang trước, khi đó chúng ta sẽ gửi tín hiệu **SIGTSTP** đến kernel, tín hiệu này sẽ tạm dừng process.



```
holdennguyen6174@htb[/htb]$ ping -c 10 www.hackthebox.eu
holdennguyen6174@htb[/htb]$ vim tmpfile
[Ctrl + Z]
[2]+  Stopped                  vim tmpfile
```

Lúc này, để xem tất cả background process, sử dụng lệnh:



```
holdennguyen6174@htb[/htb]$ jobs
[1]+  Stopped                  ping -c 10 www.hackthebox.eu
[2]+  Stopped                  vim tmpfile
```

Phím tắt [Ctrl + Z] tạm dừng process, và chúng sẽ không thể tiếp tục thực thi.

# Background a Process

Để process ở background tiếp tục chạy, hãy nhập lệnh bg

```
holdennguyen6174@htb[/htb]$ bg  
holdennguyen6174@htb[/htb]$ --- www.hackthebox.eu ping statistics ---  
10 packets transmitted, 0 received, 100% packet loss, time 113482ms  
[ENTER]  
[1]+ Exit 1 ping -c 10 www.hackthebox.eu
```

Một tùy chọn khác đó là tự động thiết lập process với ký hiệu AND (&) ở cuối command

```
holdennguyen6174@htb[/htb]$ ping -c 10 www.hackthebox.eu &  
[1] 10825  
PING www.hackthebox.eu (172.67.1.1) 56(84) bytes of data.
```

Khi process chạy xong, chúng ta sẽ thấy kết quả

```
holdennguyen6174@htb[/htb]$ --- www.hackthebox.eu ping statistics ---  
10 packets transmitted, 0 received, 100% packet loss, time 9210ms  
[ENTER]  
[1]+ Exit 1 ping -c 10 www.hackthebox.eu
```

# Foreground a Process

Chúng ta có thể sử dụng lệnh **jobs** để liệt kê tất cả background process. Background process không yêu cầu user tương tác, và chúng ta có thể tiếp tục sử dụng shell session đó mà không cần đợi cho đến khi process đó chạy xong.

```
● ● ●  
holdennguyen6174@htb[/htb]$ jobs  
[1]+  Running                  ping -c 10 www.hackthebox.eu &
```

Nếu muốn đưa background process quay lại foreground và tiếp tục tương tác với nó, chúng ta có thể sử dụng **fg <ID>**

```
● ● ●  
holdennguyen6174@htb[/htb]$ fg 1  
ping -c 10 www.hackthebox.eu  
  
--- www.hackthebox.eu ping statistics ---  
10 packets transmitted, 0 received, 100% packet loss, time 9206ms
```

# Execute Multiple Commands

Có ba cách để chạy nhiều lệnh lần lượt. Các lệnh chia ra bởi:

- Dấu chấm phẩy - Semicolon (;
- Hai dấu AND - **ampersand** (&&)
- Dấu xuyệt đứng - pipe (|)

Semicolon (;) ngăn giữa các lệnh sẽ tiếp tục thực hiện lệnh sau mặc cho kết quả hay lỗi xảy ra ở lệnh trước đó. Ví dụ, với lệnh thứ hai là ls cùng một file không tồn tại, chúng ta sẽ gặp lỗi và lệnh thứ ba sẽ tiếp tục được thực thi:

```
holdennguyen6174@htb[/htb]$ echo '1'; ls MISSING_FILE; echo '3'

1
ls: cannot access 'MISSING_FILE': No such file or directory
3
```

Với double AND (&&), nếu có lỗi xảy ra, lệnh tiếp theo sẽ không được thực thi và toàn bộ process sẽ dừng lại:

```
holdennguyen6174@htb[/htb]$ echo '1' && ls MISSING_FILE && echo '3'

1
ls: cannot access 'MISSING_FILE': No such file or directory
```

Pipes (|) sẽ không chỉ ảnh hưởng việc có lỗi hay không mà còn phụ thuộc vào kết quả của lệnh trước đó. Chúng ta sẽ tìm hiểu kỹ hơn ở nội dung **File Descriptiors and Redirections**.

Nguyen Minh Hung  
Minhung



HACKTHEBOX

# Working with Web Services

#LinuxFundamentals #SystemManagement #Part9



Một phần thiết yếu nữa đó là giao tiếp với các **web server**. Có nhiều cách khác nhau để thiết lập web server trên hệ điều hành Linux. Một trong những web server phổ biến và được sử dụng nhiều nhất bên cạnh **IIS** và **Nginx** là **Apache**.

Đối với Apache web server, chúng ta có thể sử dụng các module thích hợp, mã hóa giao tiếp giữa trình duyệt web và web server (**mod\_ssl**), sử dụng làm máy chủ proxy (**mod\_proxy**) hoặc thực hiện các thao tác phức tạp đối với HTTP header data (**mod\_headers**) và URLs (**mod\_rewrite**).

Apache cung cấp khả năng tạo các trang web một cách linh hoạt bằng cách sử dụng các ngôn ngữ **server-side scripting**. Các ngôn ngữ thường được sử dụng là **PHP**, **Perl** hoặc **Ruby**. Ngoài ra còn có **Python**, **JavaScript**, **Lua** và **.NET** cũng có thể được sử dụng cho việc này.

# Apache

Chúng ta có thể cài đặt Apache web server với lệnh sau:

```
holdennguyen6174@htb[/htb]$ apt install apache2 -y

Reading package lists... Done
Building dependency tree
Reading state information... Done
Suggested packages:
  apache2-doc apache2-suexec-pristine | apache2-suexec-custom
The following NEW packages will be installed:
  apache2
0 upgraded, 1 newly installed, 0 to remove and 17 not upgraded.
Need to get 95,1 kB of archives.
After this operation, 535 kB of additional disk space will be used.
Get:1 http://de.archive.ubuntu.com/ubuntu bionic-updates/main amd64 apache2 amd64 2.4.29
Fetched 95,1 kB in 0s (270 kB/s)
<SNIP>
```

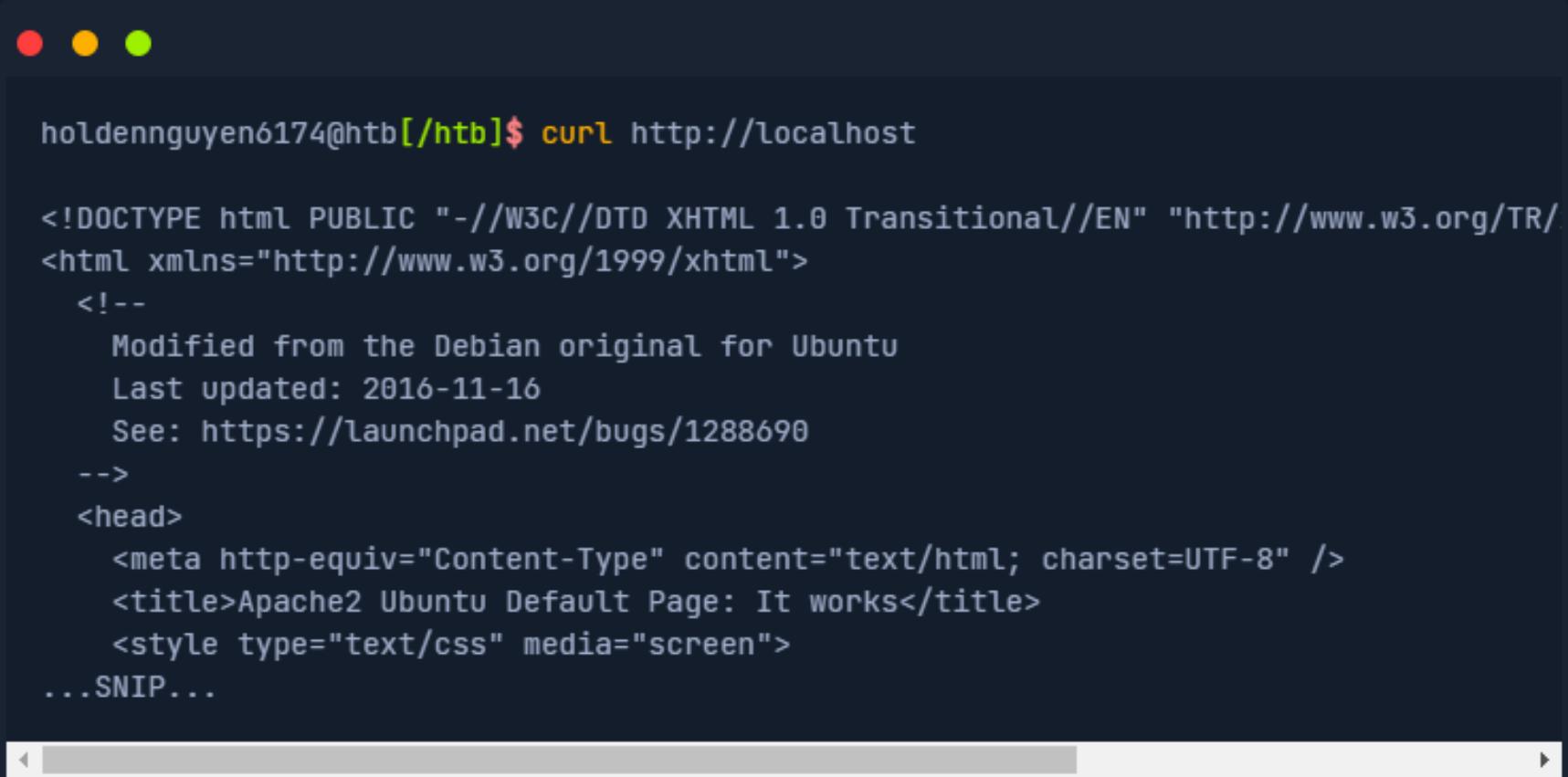
Sau khi chạy, chúng ta có thể sử dụng trình duyệt web chuyển đến trang mặc định (<http://localhost>).



Đây là trang mặc định (default page) sau khi cài đặt và chạy service, đảm bảo rằng web server đang hoạt động đúng.

# CURL

cURL là một tool cho phép chúng ta gửi và nhận file qua các giao thức như **HTTP**, **HTTPS**, **FTP**, **SFTP**, **FTPS** và **SCP**. Công cụ này cung cấp khả năng kiểm soát và kiểm tra các trang web từ xa. Bên cạnh nội dung server, chúng ta cũng có thể xem các từng request trong giao tiếp giữa client và server (-v hay --verbose)... Thông thường cURL sẽ được cài đặt sẵn trên hầu hết các Linux system. Việc làm quen với công cụ này có thể giúp làm một số process dễ dàng hơn nhiều sau này.



A terminal window showing the output of a curl command. The command is `curl http://localhost`. The output is the Apache2 Ubuntu Default Page: It works, which includes the DOCTYPE declaration, XML namespace, and a comment indicating it was modified from the Debian original for Ubuntu. It also shows the last update date (2016-11-16) and a bug reference (https://Launchpad.net/bugs/1288690). The page includes meta tags for content type and charset, a title, and a style block. A '...SNIP...' indicates more content follows.

```
holdennguyen6174@htb[/htb]$ curl http://localhost
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/
<html xmlns="http://www.w3.org/1999/xhtml">
<!--
    Modified from the Debian original for Ubuntu
    Last updated: 2016-11-16
    See: https://Launchpad.net/bugs/1288690
-->
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
    <title>Apache2 Ubuntu Default Page: It works</title>
    <style type="text/css" media="screen">
...
SNIP...
```

Trong thẻ **<title>** chúng ta có thể thấy rằng đó là văn bản giống như từ trình duyệt của chúng ta ở trang trước. Chúng ta có thể kiểm tra source code của trang web và lấy thông tin từ đó. Vấn đề này sẽ được tìm hiểu kỹ hơn ở nội dung khác.

# Wget

Một giải pháp thay thế cho curl là công cụ **wget**. Với tool này, chúng ta có thể tải xuống các file từ FTP hoặc HTTP server trực tiếp từ terminal và nó có thể đóng vai trò là một trình quản lý tải xuống khá tốt.

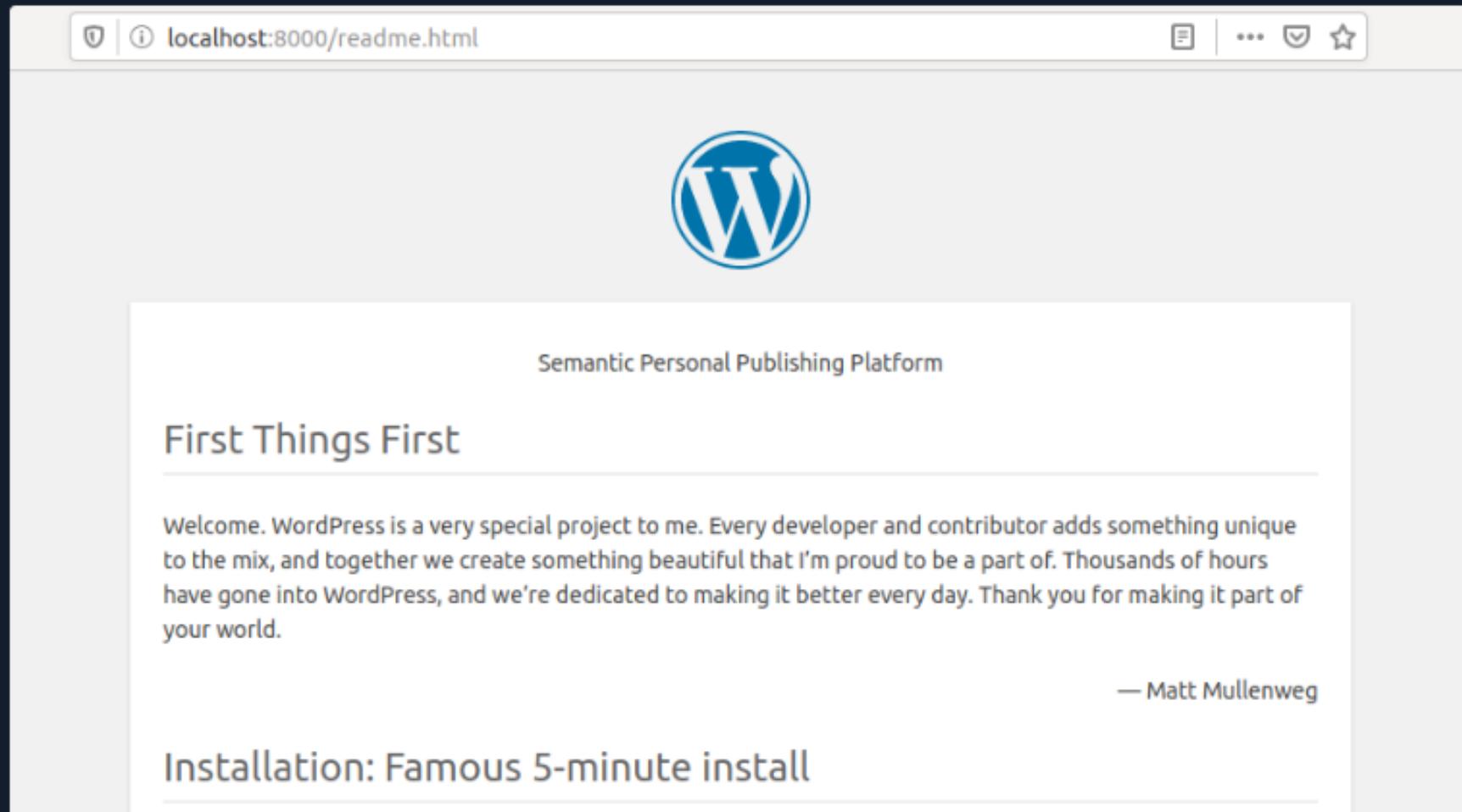
Nếu chúng ta sử dụng wget theo cách tương tự, sự khác biệt so với curl là nội dung website được tải xuống và lưu trữ local giống như trong ví dụ sau:

```
holdennguyen6174@htb[/htb]$ wget http://localhost  
--2020-05-15 17:43:52-- http://localhost/  
Resolving localhost (localhost)... 127.0.0.1  
Connecting to localhost (localhost)|127.0.0.1|:80... connected.  
HTTP request sent, awaiting response... 200 OK  
Length: 10918 (11K) [text/html]  
Saving to: 'index.html'  
  
index.html          100%[=====] 10,66K  --.-KB  
2020-05-15 17:43:52 (33,0 MB/s) - 'index.html' saved [10918/10918]
```

# Python 3

Một lựa chọn khác được sử dụng khi chuyển data là Python 3. Root directory của web server là nơi lệnh được thực thi để khởi động server. Đối với ví dụ dưới đây, chúng ta đang ở trong một directory cài đặt WordPress và chứa “`readme.html`“. Hãy khởi động Python3 web server và truy cập nó trên trình duyệt

```
holdennguyen6174@htb[~/htb]$ python3 -m http.server
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
```



Xem các request ở phần event của Python 3 web server:

```
holdennguyen6174@htb[~/htb]$ python3 -m http.server
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
127.0.0.1 - - [15/May/2020 17:56:29] "GET /readme.html HTTP/1.1" 200 -
127.0.0.1 - - [15/May/2020 17:56:29] "GET /wp-admin/css/install.css?ver=20100228 HTTP/1.1" 200 -
127.0.0.1 - - [15/May/2020 17:56:29] "GET /wp-admin/images/wordpress-logo.png HTTP/1.1" 200 -
127.0.0.1 - - [15/May/2020 17:56:29] "GET /wp-admin/images/wordpress-logo.svg?ver=20131107 HTTP/1.1" 200 -
```

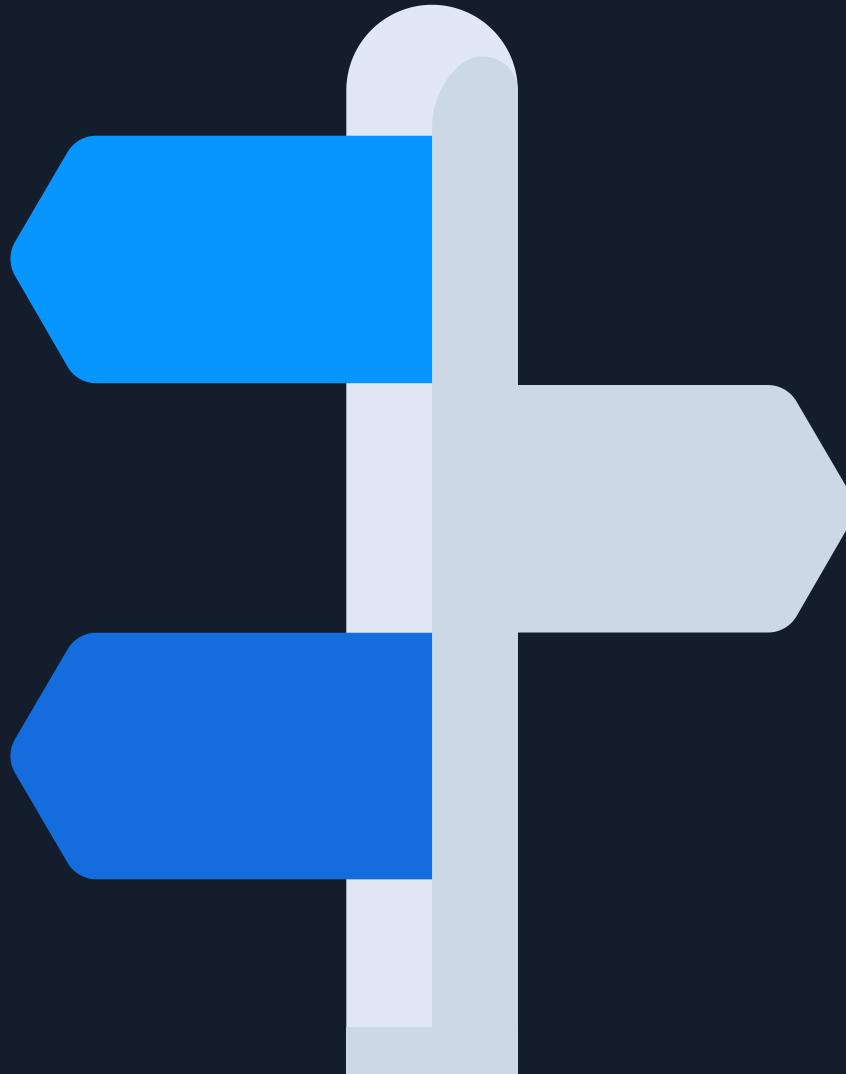
Nguyen Minh Hung  
Minhung



HACKTHEBOX

# Navigation

#LinuxFundamentals #Workflow #Part10



Trước khi khám phá hệ thống, chúng ta nên tìm hiểu xem mình đang ở directory nào. Có thể sử dụng lệnh **pwd**:

```
cry0l1t3@htb:[~]$ pwd  
/home/cry0l1t3
```

Lệnh **ls** sẽ cần thiết cho việc điều hướng. Nó có nhiều option bổ sung giúp hiển thị nội dung bên trong directory hiện tại.

```
cry0l1t3@htb:[~]$ ls  
Desktop Documents Downloads Music Pictures Public Templates Videos
```

```
cry0l1t3@htb:[~]$ ls -l  
  
total 32  
drwxr-xr-x 2 cry0l1t3 cry0l1t3 4096 Nov 13 17:37 Desktop  
drwxr-xr-x 2 cry0l1t3 cry0l1t3 4096 Nov 13 17:34 Documents  
drwxr-xr-x 3 cry0l1t3 cry0l1t3 4096 Nov 15 03:26 Downloads  
drwxr-xr-x 2 cry0l1t3 cry0l1t3 4096 Nov 13 17:34 Music  
drwxr-xr-x 2 cry0l1t3 cry0l1t3 4096 Nov 13 17:34 Pictures  
drwxr-xr-x 2 cry0l1t3 cry0l1t3 4096 Nov 13 17:34 Public  
drwxr-xr-x 2 cry0l1t3 cry0l1t3 4096 Nov 13 17:34 Templates  
drwxr-xr-x 2 cry0l1t3 cry0l1t3 4096 Nov 13 17:34 Videos
```

Đôi khi chúng ta sẽ không thể thấy hết mọi thứ trong một thư mục. Để liệt kê nội dung bên trong một directory, chúng ta không nhất thiết phải điều hướng tới đó trước. Có thể sử dụng “ls” với đường dẫn cụ thể tới nơi chúng ta cần biết nội dung.



```
cry0l1t3@htb[~]$ ls -l /var/  
  
total 52  
drwxr-xr-x  2 root root      4096 Mai 15 18:54 backups  
drwxr-xr-x 18 root root      4096 Nov 15 16:55 cache  
drwxrwsrwt  2 root whoopsie  4096 Jul 25  2018 crash  
drwxr-xr-x  66 root root     4096 Mai 15 03:08 lib  
drwxrwsr-x  2 root staff    4096 Nov 24  2018 local  
<SNIP>
```

Chúng ta có thể làm điều tương tự nếu muốn đi tới một directory bằng cách sử dụng lệnh `cd`. Ví dụ như đến `/dev/shm`. Tất nhiên, chúng ta có thể đi tới directory `/dev` trước rồi mới tiếp tục `cd` tới `/shm`. Tuy nhiên, trực tiếp nhập toàn bộ đường dẫn và nhảy thẳng tới đó không phải là ý tồi.



```
cry0l1t3@htb[~]$ cd /dev/shm  
  
cry0l1t3@htb[/dev/shm]$
```

Chúng ta cũng có thể nhanh chóng trở lại directory trước đó bằng lệnh “cd -“

```
● ● ●  
cry0l1t3@htb[/dev/shm]$ cd -  
cry0l1t3@htb[~]$
```

Shell cũng cung cấp cho chúng ta chức năng tự động hoàn thành câu lệnh giúp cho việc điều hướng dễ dàng hơn. Nếu nhập `cd /dev/s` và nhấn [TAB] 2 lần, chúng ta sẽ nhận được tất cả mục bắt đầu bằng chữ cái “s“ trong directory `/dev/`.

```
● ● ●  
cry0l1t3@htb[~]$ cd /dev/s [TAB 2x]  
shm/ snd/
```

Nếu thêm chữ cái “h“ kế tiếp “s“, shell sẽ tự động hoàn thành input do không còn thư mục nào trong directory này bắt đầu bằng “sh“ ngoài `shm/`.

Tiếp đó, chúng ta hiển thị toàn bộ nội dung trong directory `/dev/shm`. Sẽ chỉ thấy những content dưới đây.

```
cry0l1t3@htb[/dev/shm]$ ls -la  
  
total 0  
drwxrwxrwt  2 root root  40 Mai 15 18:31 .  
drwxr-xr-x 17 root root 4000 Mai 14 20:45 ..
```

Mục đầu tiên với một dấu chấm (`.`) biểu thị directory hiện tại ta đang ở. Mục thứ hai với hai dấu chấm (`..`) biểu thị thư mục cha là `/dev`. Điều đó có nghĩa chúng ta có thể chuyển đến thư mục cha bằng lệnh sau:

```
cry0l1t3@htb[/dev/shm]$ cd ..  
  
cry0l1t3@htb[/dev]$
```

Shell của chúng ta có thể sẽ đầy những lệnh đã nhập, có thể xóa chúng ở shell bằng lệnh `clear`. Hãy thử kết hợp với lệnh điều hướng tới `/dev/shm` với lệnh điều hướng bằng `&&`.

```
cry0l1t3@htb[/dev]$ cd shm && clear
```

Nguyen Minh Hung  
Minhung



HACKTHEBOX

# Working with Files and Directories

#LinuxFundamentals #Workflow #Part11



# Create, Move, and Copy

Chúng ta hãy làm việc với file và directory, tìm hiểu cách tạo, đổi tên, di chuyển, sao chép và xóa. Đầu tiên hãy tạo một file trống và một directory. Sử dụng lệnh **touch** để tạo một file trống và **mkdir** để tạo một directory.



## Syntax - touch

```
holdennguyen6174@htb[/htb]$ touch <name>
```



## Syntax - mkdir

```
holdennguyen6174@htb[/htb]$ mkdir <name>
```

Ví dụ này đặt tên file là **info.txt** và directory là **Storage**:



## Create an Empty File

```
holdennguyen6174@htb[/htb]$ touch info.txt
```



## Create a Directory

```
holdennguyen6174@htb[/htb]$ mkdir Storage
```

Khi chúng ta muốn có những thư mục cụ thể trong từng thư mục, sẽ rất tốn thời gian tạo lệnh này cho từng directory. Lệnh `mkdir` có một option `-p` để thêm các parent directory.

● ● ● Create a Directory

```
holdennguyen6174@htb[/htb]$ mkdir -p Storage/local/user/documents
```

Có thể xem toàn bộ cấu trúc sau khi tạo các parent directory bằng công cụ `tree`.

● ● ● Create a Directory

```
holdennguyen6174@htb[/htb]$ tree .
```

```
.  
└── info.txt  
└── Storage  
    └── local  
        └── user  
            └── documents
```

4 directories, 1 file

Chúng ta cũng có thể tạo file trực tiếp trong directory bằng cách chỉ định đường dẫn nơi tệp sẽ được lưu trữ. Mẹo nhỏ là sử dụng dấu chấm đơn (.) để báo cho system biết chúng ta muốn bắt đầu từ directory hiện tại.

Vì vậy lệnh tạo một file trống có tên `userinfo.txt` sẽ như sau:

```
● ● ● Create userinfo.txt
holdennguyen6174@htb[/htb]$ touch ./Storage/local/user/userinfo.txt

● ● ● Create userinfo.txt
holdennguyen6174@htb[/htb]$ tree .
.
└── info.txt
└── Storage
    └── local
        └── user
            ├── documents
            └── userinfo.txt

4 directories, 2 files
```

Và với lệnh `mv`, chúng ta có thể di chuyển cũng như đổi tên file và directory. Cú pháp sẽ trông như sau:

```
● ● ● Syntax - mv
holdennguyen6174@htb[/htb]$ mv <file/directory> <renamed file/directory>
```

Trước tiên, hãy đổi tên file `info.txt` thành `information.txt` và sau đó chuyển file đó vào directory `Storage`:

```
● ● ● Rename File
holdennguyen6174@htb[/htb]$ mv info.txt information.txt
```

Bây giờ chúng ta sẽ tạo một file có tên `readme.txt` trong directory hiện tại, sau đó copy file `information.txt` và `readme.txt` vào `Storage/ directory`.

Tạo file `readme.txt`:

```
Create readme.txt
```

```
holdennguyen6174@htb[/htb]$ touch readme.txt
```

Di chuyển file tới Directory cụ thể:

```
Move Files to Specific Directory
```

```
holdennguyen6174@htb[/htb]$ mv information.txt readme.txt Storage/
```

```
Move Files to Specific Directory
```

```
holdennguyen6174@htb[/htb]$ tree .
```

```
└── Storage
    ├── information.txt
    └── local
        └── user
            ├── documents
            └── userinfo.txt
    └── readme.txt
```

```
4 directories, 3 files
```

Giả sử chúng ta muốn thêm một file `readme.txt` nằm trong `local/ directory`. Có thể copy với đường dẫn chỉ định như sau:



### Copy `readme.txt`

```
holdennguyen6174@htb[/htb]$ cp Storage/readme.txt Storage/local/
```

Bây giờ sử dụng tool `tree` để kiểm tra lại vị trí của file lần nữa:



### Copy `readme.txt`

```
holdennguyen6174@htb[/htb]$ tree .
```

```
└── Storage
    ├── information.txt
    └── local
        └── readme.txt
            └── user
                └── documents
                    └── userinfo.txt
    └── readme.txt
```

```
4 directories, 4 files
```

Nguyen Minh Hung  
Minhung



HACKTHEBOX

# Editing Files

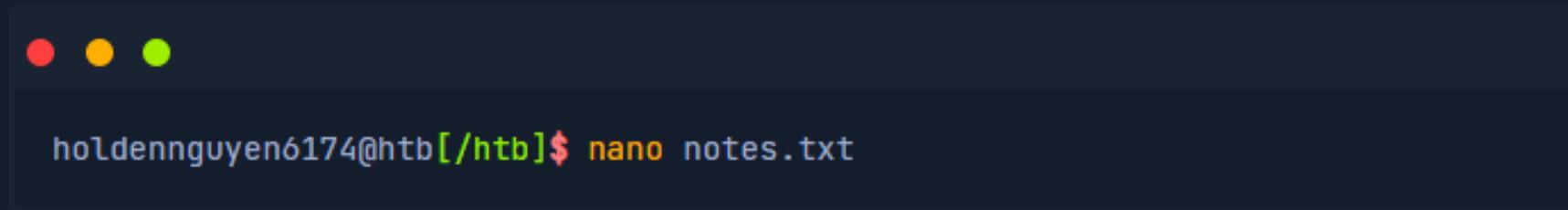
#LinuxFundamentals #Workflow #Part12



# Nano Editor

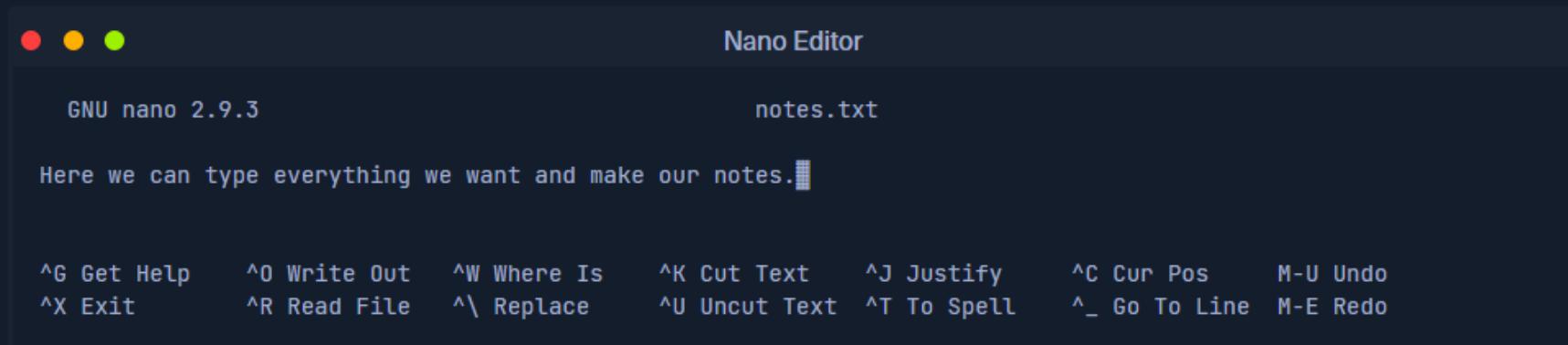
Có một số cách để chỉnh sửa một file. Một trong những text editor phổ biến nhất cho việc này là **Vi** và **Vim**. Ít được sử dụng hơn, có thêm **Nano**. Trước tiên, chúng ta sẽ tìm hiểu về Nano vì nó dễ hiểu hơn một chút.

Chúng ta có thể trực tiếp tạo một file mới với trình soạn thảo văn bản Nano bằng cách chỉ định trực tiếp tên của file làm parameter đầu tiên. Trong trường hợp này, ví dụ sẽ tạo một file mới có tên là **note.txt**:



```
holdennguyen6174@htb[/htb]$ nano notes.txt
```

Lúc này chúng ta sẽ thấy một “**pager**” được mở, và có thể tự do nhập, thêm văn bản bất kỳ. Shell của chúng ta sẽ trông như sau:



The screenshot shows the GNU nano 2.9.3 text editor interface. The title bar says "Nano Editor". The status bar at the bottom shows "GNU nano 2.9.3" on the left and "notes.txt" on the right. In the main area, there is a single line of text: "Here we can type everything we want and make our notes." Below the text area is a menu of keyboard shortcuts:

<b>^G</b> Get Help	<b>^O</b> Write Out	<b>^W</b> Where Is	<b>^K</b> Cut Text	<b>^J</b> Justify	<b>^C</b> Cur Pos	M-U Undo
<b>^X</b> Exit	<b>^R</b> Read File	<b>^\\</b> Replace	<b>^U</b> Uncut Text	<b>^T</b> To Spell	<b>^_</b> Go To Line	M-E Redo

Hai dòng phía dưới với mô tả ngắn, dấu mũ - **caret** (^) là viết tắt của phím **[CTRL]**. Ví dụ nếu nhấn **[CTRL + W]**, dòng tìm kiếm sẽ xuất hiện ở cuối và chúng ta có thể nhập một hoặc nhiều từ đang tìm kiếm.

Nếu tìm kiếm từ “we“ và nhấn [ENTER], con trỏ sẽ di chuyển đến từ đầu tiên khớp với từ khóa tìm kiếm như hình dưới đây:

The screenshot shows the Nano Editor interface with the title "Nano Editor". The status bar indicates "GNU nano 2.9.3" and the file name "notes.txt". The main area contains the text "Here we can type everything we want and make our notes.". Below the text, the search results are displayed: "Search: notes". The command line shows various keyboard shortcuts for navigation and search.

```

Nano Editor

GNU nano 2.9.3          notes.txt

Here we can type everything we want and make our notes.

Search: notes
^G Get Help   M-C Case Sens  M-B Backwards  M-J FullJustify ^W Beg of Par  ^Y First Line  ^P PrevHistory
^C Cancel     M-R Regexp    ^R Replace      ^T Go To Line   ^O End of Par  ^V Last Line   ^N NextHistory

```

Để chuyển con trỏ sang từ khớp kế tiếp, nhấn lại [CTRL + W] và xác nhận bằng [ENTER] mà không cần bổ sung thêm bất kỳ thông tin nào.

The screenshot shows the Nano Editor interface with the title "Nano Editor". The status bar indicates "GNU nano 2.9.3" and the file name "notes.txt". The main area contains the text "Here we can type everything we want and make our notes.". Below the text, the search results are displayed: "Search [we]:". The command line shows various keyboard shortcuts for navigation and search.

```

Nano Editor

GNU nano 2.9.3          notes.txt

Here we can type everything we want and make our notes.

Search [we]:
^G Get Help   M-C Case Sens  M-B Backwards  M-J FullJustify ^W Beg of Par  ^Y First Line  ^P PrevHistory
^C Cancel     M-R Regexp    ^R Replace      ^T Go To Line   ^O End of Par  ^V Last Line   ^N NextHistory

```

Bây giờ chúng ta có thể lưu file bằng cách nhấn [CTRL + O] và xác nhận tên tệp bằng [ENTER].

The screenshot shows the Nano Editor interface with the title "Nano Editor". The status bar indicates "GNU nano 2.9.3" and the file name "notes.txt". The main area contains the text "Here we can type everything we want and make our notes.". Below the text, a save dialog is displayed: "File Name to Write: notes.txt". The command line shows various keyboard shortcuts for navigation and search.

```

Nano Editor

GNU nano 2.9.3          notes.txt

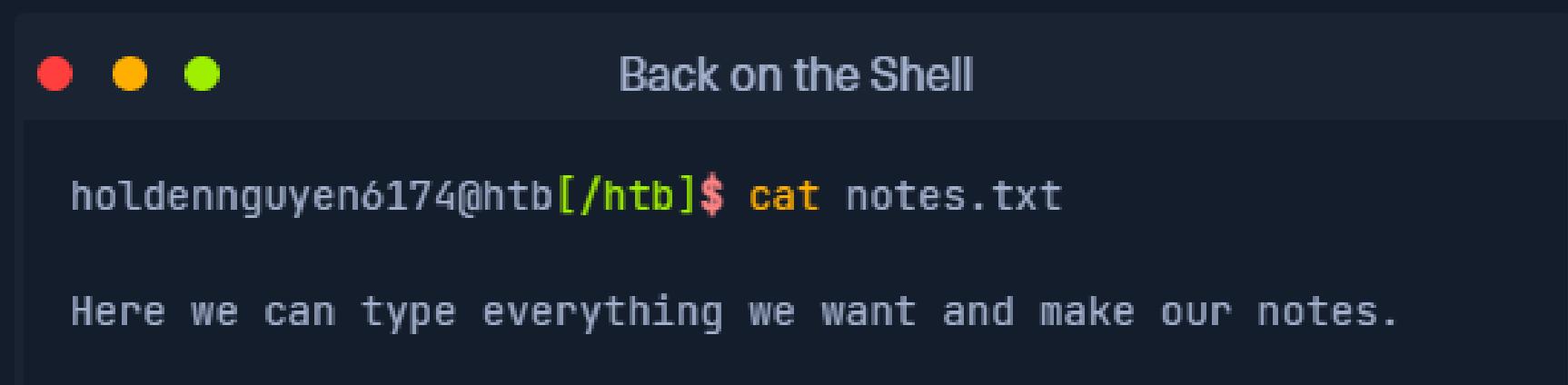
Here we can type everything we want and make our notes.

File Name to Write: notes.txt
^G Get Help   M-C Case Sens  M-B Backwards  M-J FullJustify ^W Beg of Par  ^Y First Line  ^P PrevHistory
^C Cancel     M-R Regexp    ^R Replace      ^T Go To Line   ^O End of Par  ^V Last Line   ^N NextHistory

```

Sau khi đã lưu file, chúng ta có thể rời trình chỉnh sửa bằng [CTRL + X]

Trở lại với shell, để xem nội dung của file, chúng ta sử dụng lệnh cat.



Back on the Shell

```
holdennguyen6174@htb[/htb]$ cat notes.txt
Here we can type everything we want and make our notes.
```

Có nhiều file trên hệ thống Linux có thể đóng vai trò thiết yếu đối cho chúng ta với tư cách là người kiểm thử xâm nhập - penetration tester ví dụ như tệp “/etc/passwd“.

# VIM

Vim là trình soạn thảo mã nguồn mở (open-source) dành cho tất cả các loại văn bản ASCII, giống như Nano. Nó là một bản sao cải tiến của Vi trước đó. Đây là một trình soạn thảo cực kỳ mạnh mẽ tập trung vào yếu tố cần thiết, cụ thể là soạn thảo văn bản.

Đối với các tác vụ khác, Vim cung cấp giao diện cho các chương trình bên ngoài, chẳng hạn như grep, awk, sed... có thể xử lý các tác vụ cụ thể của chúng tốt hơn nhiều so với chức năng tương ứng có thể được triển khai trực tiếp trong trình chỉnh sửa. Điều này làm cho nó trở nên nhỏ gọn, nhanh, mạnh, linh hoạt và ít bị lỗi hơn.

Vim tuân theo nguyên tắc Unix: nhiều chương trình nhỏ chuyên dụng đã được kiểm nghiệm và chứng minh tốt, khi được kết hợp lại và giao tiếp với nhau tạo ra một hệ thống linh hoạt và mạnh mẽ.

# VIM



holdennguyen6174@htb[/htb]\$ vim



```
1 $  
~  
~          VIM - Vi IMproved  
~  
~          version 8.0.1453  
~          by Bram Moolenaar et al.  
~ Modified by pkg-vim-maintainers@lists.alioth.debian.org  
~ Vim is open source and freely distributable  
~  
~          Sponsor Vim development!  
~ type :help sponsor<Enter>    for information  
~  
~          type :q<Enter>          to exit  
~          type :help<Enter> or  <F1> for on-line help  
~          type :help version8<Enter> for version info  
~  
~  
0,0-1      All
```

# VIM

Trái ngược với Nano, Vim là một trình chỉnh sửa có thể phân biệt giữa văn bản và lệnh được nhập. Vim cung cấp tổng cộng 6 chế độ cơ bản giúp công việc của chúng ta dễ dàng hơn và làm cho trình chỉnh sửa này trở nên mạnh mẽ:

- **Normal**: Ở chế độ này, tất cả input được coi là lệnh biên tập. Vì vậy các ký tự đã nhập sẽ không được chèn vào bộ đệm của trình soạn thảo giống như các editor khác. Sau khi chạy Vim editor, chúng ta thường ở chế độ Normal.
- **Insert**: Với một vài ngoại lệ, tất cả các ký tự đã nhập sẽ được chèn vào bộ đệm của trình soạn thảo.
- **Visual**: Chế độ trực quan được sử dụng để đánh dấu một phần liền kề của văn bản, phần này sẽ được tô sáng trực quan. Bằng cách định vị con trỏ, chúng ta thay đổi khu vực đã chọn. Sau đó khu vực được đánh dấu có thể được chỉnh sửa theo nhiều cách khác nhau như xóa, sao chép hoặc thay thế.
- **Command**: Cho phép chúng ta nhập các lệnh một dòng ở cuối trình soạn thảo. Điều này có thể được sử dụng để sắp xếp, thay thế các phần văn bản hoặc xóa chúng.
- **Replace**: Văn bản mới nhập sẽ ghi đè lên các ký tự văn bản hiện có, trừ khi không có ký tự cũ nào ở vị trí con trỏ hiện tại. Sau đó văn bản mới nhập sẽ được thêm vào.

# VIM

Khi Vim được mở, chúng ta có thể chuyển sang chế độ Command bằng cách nhập “:“ và nhập tiếp “q“ để đóng



The screenshot shows a terminal window with three colored dots (red, yellow, green) at the top left. The title bar says "Vim". The main content is the Vim startup message:

```
1 $  
~  
~ VIM - Vi IMproved  
~  
~ version 8.0.1453  
~ by Bram Moolenaar et al.  
~ Modified by pkg-vim-maintainers@lists.alioth.debian.org  
~ Vim is open source and freely distributable  
~  
~ Sponsor Vim development!  
~ type :help sponsor<Enter> for information  
~  
~ type :q<Enter> to exit  
~ type :help<Enter> or <F1> for on-line help  
~ type :help version8<Enter> for version info  
~  
:q
```

# VIM

Vim cung cấp một công cụ tuyệt vời gọi là **vimtutor** để thực hành và làm quen với trình soạn thảo. Thoạt đầu có vẻ hơi khó khăn và phức tạp, nhưng bạn sẽ chỉ cảm thấy như vậy trong một thời gian ngắn. Hiệu quả mà chúng ta đạt được từ Vim khi đã quen với nó là rất lớn.

```

● ● ● VimTutor
holdennguyen6174@htb[/htb]$ vimtutor

● ● ● VimTutor
=====
=   W e l c o m e   t o   t h e   V I M   T u t o r   -   V e r s i o n   1 . 7   =
=====

Vim is a very powerful editor that has many commands, too many to
explain in a tutor such as this. This tutor is designed to describe
enough of the commands that you will be able to easily use Vim as
an all-purpose editor.

The approximate time required to complete the tutor is 25-30 minutes,
depending upon how much time is spent with experimentation.

ATTENTION:
The commands in the lessons will modify the text. Make a copy of this
file to practice on (if you started "vimtutor" this is already a copy).

It is important to remember that this tutor is set up to teach by
use. That means that you need to execute the commands to learn them
properly. If you only read the text, you will forget the commands!

Now, make sure that your Caps-Lock key is NOT depressed and press
the j key enough times to move the cursor so that lesson 1.1
completely fills the screen.

=====

```

Nguyen Minh Hung  
Minhung



HACKTHEBOX

# Find Files and Directories

#LinuxFundamentals #Workflow #Part13



# Which

Một trong những công cụ phổ biến đó là **which**. Tool này trả về đường dẫn (path) đến file hoặc link sẽ được thực thi. Điều này cho phép chúng ta xác định xem các chương trình cụ thể như **cURL**, **netcat**, **wget**, **python**, **gcc** có sẵn trên hệ điều hành hay không. Hãy sử dụng nó để tìm kiếm Python trong ví này:

```
holdennguyen0174@htb[/htb]$ which python  
/usr/bin/python
```

Nếu chương trình chúng ta tìm không tồn tại, sẽ không có kết quả nào hiển thị.

# Find

Một công cụ hữu ích khác là **find**. Ngoài chức năng tìm kiếm file và directory, công cụ này còn có chức năng lọc kết quả. Chúng ta có thể sử dụng các tham số bộ lọc như kích thước của file hoặc date. Chúng ta cũng có thể chỉ định việc chỉ tìm kiếm file hoặc directory.



## Syntax - find

```
holdennguyen6174@htb[/htb]$ find <location> <options>
```

Hãy xem một ví dụ về lệnh find với nhiều option:



## Syntax - find

```
holdennguyen6174@htb[/htb]$ find / -type f -name *.conf -user root -size +20k -newermt 2020-03-03 -exec ls -al {} \; 2>/dev/null
-rw-r--r-- 1 root root 136392 Apr 25 20:29 /usr/src/linux-headers-5.5.0-1parrot1-amd64/include/config/auto.conf
-rw-r--r-- 1 root root 82290 Apr 25 20:29 /usr/src/linux-headers-5.5.0-1parrot1-amd64/include/config/tristate.conf
-rw-r--r-- 1 root root 95813 May  7 14:33 /usr/share/metasploit-framework/data/jtr/repeats32.conf
-rw-r--r-- 1 root root 60346 May  7 14:33 /usr/share/metasploit-framework/data/jtr/dynamic.conf
-rw-r--r-- 1 root root 96249 May  7 14:33 /usr/share/metasploit-framework/data/jtr/dumb32.conf
-rw-r--r-- 1 root root 54755 May  7 14:33 /usr/share/metasploit-framework/data/jtr/repeats16.conf
-rw-r--r-- 1 root root 22635 May  7 14:33 /usr/share/metasploit-framework/data/jtr/korelogic.conf
-rwxr-xr-x 1 root root 108534 May  7 14:33 /usr/share/metasploit-framework/data/jtr/john.conf
-rw-r--r-- 1 root root 55285 May  7 14:33 /usr/share/metasploit-framework/data/jtr/dumb16.conf
-rw-r--r-- 1 root root 21254 May  2 11:59 /usr/share/doc/sqlmap/examples/sqlmap.conf
-rw-r--r-- 1 root root 25086 Mar  4 22:04 /etc/dnsmasq.conf
-rw-r--r-- 1 root root 21254 May  2 11:59 /etc/sqlmap/sqlmap.conf
```

# Find

Bây giờ chúng ta hãy xem xét kỹ hơn các option đã sử dụng ở lệnh trong trang trước:

- **-type f** : xác định loại đối tượng tìm kiếm, trong trường hợp này, '**f**' là viết tắt của '**file**'.
- **-name \*.conf** : với '**-name**', chúng ta cung cấp tên của file cần tìm. Dấu hoa thị **asterisk (\*)** là viết tắt của 'tất cả' các file có phần mở rộng '**.conf**'.
- **-user root** : tùy chọn lọc này sẽ chỉ tìm các file có owner là root user.
- **-size +20k** : lọc tất cả các file đã được định vị từ các option trước và chỉ lấy các file dung lượng lớn hơn 20KiB.
- **-newermt 2020-03-03** : với option này, chúng ta đặt ngày. Chỉ các file mới hơn ngày được chỉ định mới hiển thị.
- **-exec ls -al {} \;** : tùy chọn này thực thi lệnh đã chỉ định, sử dụng dấu ngoặc nhọn **curly brackets** '{}' làm trình giữ chỗ cho mỗi kết quả. Dấu gạch chéo **backslash** '\' giúp shell thông dịch tiếp ký tự tiếp theo vì nếu không có nó, dấu chấm phẩy **semicolon** ';' sẽ chấm dứt câu lệnh và không đến được phần chuyển hướng còn lại.
- **2>/dev/null** : đây là phần chuyển hướng **STDERR** sang '**null device**' mà chúng ta sẽ quay lại trong phần tiếp theo. Việc chuyển hướng này đảm bảo rằng không có lỗi nào được hiển thị trong terminal. Chuyển hướng này **không phải** là một option của lệnh '**find**'.

# Locate

Sẽ mất khá nhiều thời gian để tìm các file và directory trong toàn bộ hệ thống cũng như phải thực hiện nhiều cách tìm kiếm khác nhau. Lệnh **locate** cung cấp một cách nhanh hơn, trái ngược với **find**, **locate** hoạt động với local database chứa toàn bộ thông tin về các file và directory tồn tại. Chúng ta cần cập nhật database với lệnh sau:



## Syntax - find

```
holdennguyen6174@htb[/htb]$ sudo updatedb
```

Nếu giờ muốn tìm tất cả các file có phần mở rộng **'.conf'**, việc tìm kiếm sẽ cho ra kết quả nhanh hơn nhiều so với dùng **find**.



## Syntax - find

```
holdennguyen6174@htb[/htb]$ locate *.conf
```

```
/etc/GeoIP.conf
/etc/NetworkManager/NetworkManager.conf
/etc/UPower/UPower.conf
/etc/adduser.conf
<SNIP>
```

Tuy nhiên tool này không có nhiều option lọc để sử dụng. Vì vậy việc sử dụng **locate** hay **find** luôn đáng được cân nhắc tùy thuộc vào việc chúng ta tìm kiếm điều gì.

Nguyen Minh Hung  
Minhung



HACKTHEBOX

# File Descriptors and Redirections

#LinuxFundamentals #Workflow #Part14



## File Descriptors

Một bộ mô tả - **file descriptor (FD)** trong hệ điều hành Unix/Linux là một indicator về kết nối được quản lý bởi kernel để thực hiện các thao tác **Input/Output (I/O)**. Trong các hệ điều hành Windows, nó được gọi là **filehandle**. Đó là kết nối (thường là tới một file) từ Operating system để thực hiện các I/O operation (Input/Output of Bytes).

Theo mặc định, ba file descriptor đầu tiên trong Linux là:

1. Data Stream for Input (Luồng dữ liệu cho đầu vào)  
**STDIN - 0**
2. Data Stream for Output (Luồng dữ liệu cho đầu ra)  
**STDOUT - 1**
3. Data Stream for Output liên quan đến lỗi xảy ra  
**STDERR - 2**

## STDIN and STDOUT

Ví dụ với lệnh **cat**. Khi chạy, chúng ta cung cấp đầu vào tiêu chuẩn (**STDIN - FD 0**), **khung xanh lá**, ở đây là “Think Outside The Box“. Ngay khi xác nhận bằng **[ENTER]**, nó sẽ được trả về terminal dạng đầu ra tiêu chuẩn (**STDOUT - FD 1**), **khung màu đỏ**.

```
System  🔥  🚧  🌐  VPN: (10.10.14.125) [2.48 ms]
htb-student@nixfund: ~
File Edit View Search Terminal Help
htb-student@nixfund:~$ cat
Think Outside The Box
Think Outside The Box
```

## STDIN and STDERR

Ví dụ tiếp theo, bằng cách sử dụng lệnh `find`, chúng ta sẽ thấy đầu ra tiêu chuẩn (`STDOUT - FD 1`) được đánh dấu xanh lá và lỗi tiêu chuẩn (`STDERR - FD 2`) được đánh dấu đỏ.

```
holdennguyen6174@htb[/htb]$ find /etc/ -name shadow
```

```
System VPN: (10.10.14.125) [2.71 ms]
```

```
htb-student@nixfund: ~
```

```
File Edit View Search Terminal Help
```

```
htb-student@nixfund:~$ find /etc/ -name shadow
find: '/etc/dovecot/private': Permission denied
/etc/shadow
find: '/etc/ssl/private': Permission denied
find: '/etc/polkit-1/localauthority': Permission denied
htb-student@nixfund:~$
```

Lỗi hiển thị “`Permission denied`” được đánh dấu, chúng ta có thể kiểm điều này bằng cách chuyển hướng FD cho các lỗi (`FD 2 - STDERR`) tới “`/dev/null`”. Bằng cách này, chúng ta chuyển hướng (redirection) các kết quả lỗi đến “`null device`”, để loại bỏ tất cả các dữ liệu đó.

```
holdennguyen6174@htb[/htb]$ find /etc/ -name shadow 2>/dev/null
```

```
System VPN: (10.10.14.125) [2.58 ms]
```

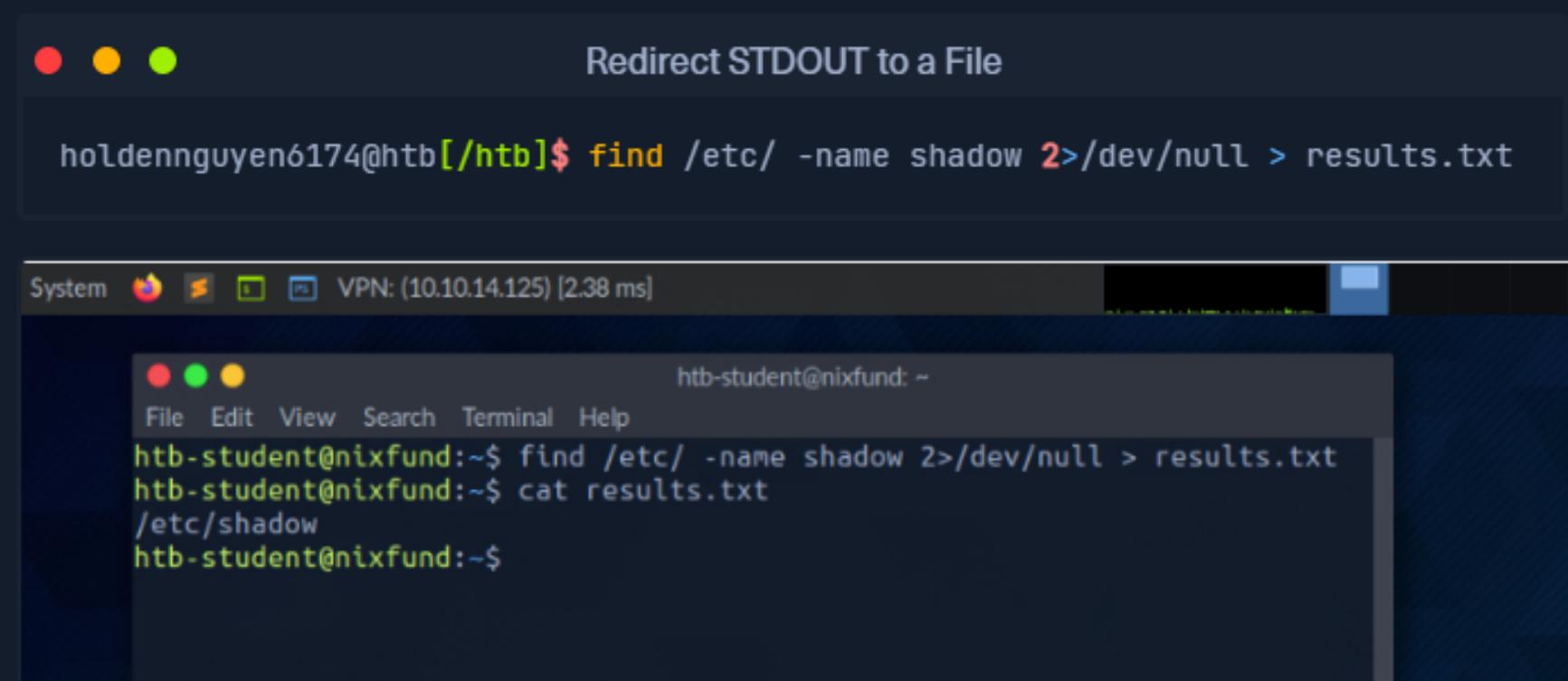
```
htb-student@nixfund: ~
```

```
File Edit View Search Terminal Help
```

```
htb-student@nixfund:~$ find /etc/ -name shadow 2>/dev/null
/etc/shadow
htb-student@nixfund:~$
```

## Redirect STDOUT to a File

Giờ đây chúng ta biết cách để tất cả các lỗi (STDERR) “**Permission denied**“ không còn được hiển thị nữa. Kết quả duy nhất hiển thị là đầu ra tiêu chuẩn (STDOUT), và chúng ta cũng có thể chuyển hướng nó đến một file có tên **result.txt**. Nội dung file này sẽ chỉ chứa STDOUT mà không có STDERR.

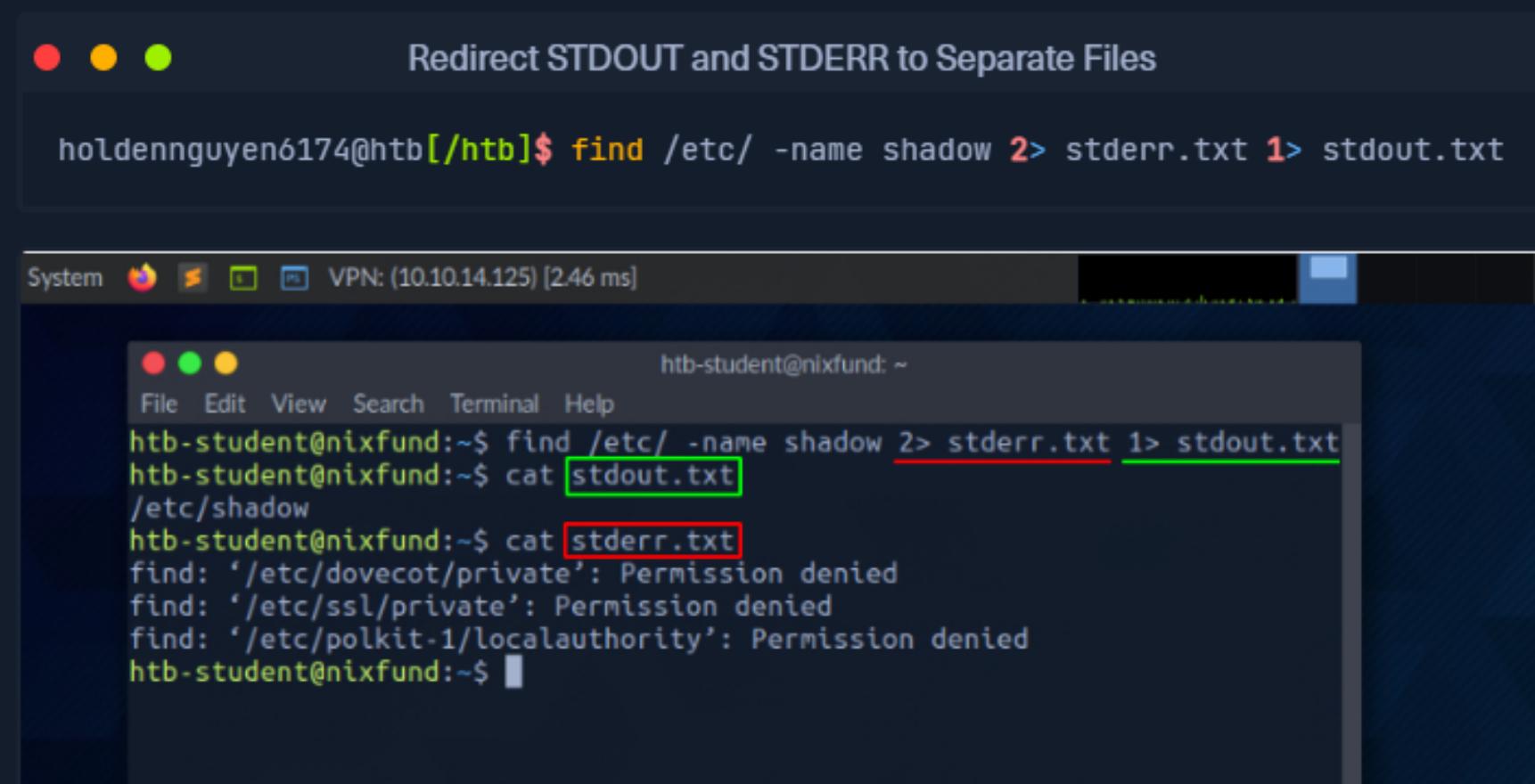


The screenshot shows a terminal window titled "Redirect STDOUT to a File". The terminal is running on a system with the IP address 10.10.14.125. The command entered is "find /etc/ -name shadow 2>/dev/null > results.txt". The output of the command is displayed in the terminal window, showing the path "/etc/shadow".

```
holdennguyen6174@htb[/htb]$ find /etc/ -name shadow 2>/dev/null > results.txt
htb-student@nixfund:~$ cat results.txt
/etc/shadow
htb-student@nixfund:~$
```

## Redirect STDOUT and STDERR to Separate Files

Chú ý rằng chúng ta đã không sử dụng một số trước dấu lớn hơn (>) ở ví dụ trước. Đó là bởi chúng ta đã chuyển hướng tất cả các STDERR sang “null device“ trước đó, và output duy nhất lúc này chỉ có đầu ra tiêu chuẩn (FD 1 - STDOUT). Để kỹ lưỡng hơn, hãy chuyển hướng lỗi tiêu chuẩn (FD 2 - STDERR) và đầu ra tiêu chuẩn (FD 1 - STDOUT) sang các file khác nhau



The screenshot shows a terminal window titled "Redirect STDOUT and STDERR to Separate Files". The terminal is running on a system with the IP address 10.10.14.125. The command entered is:

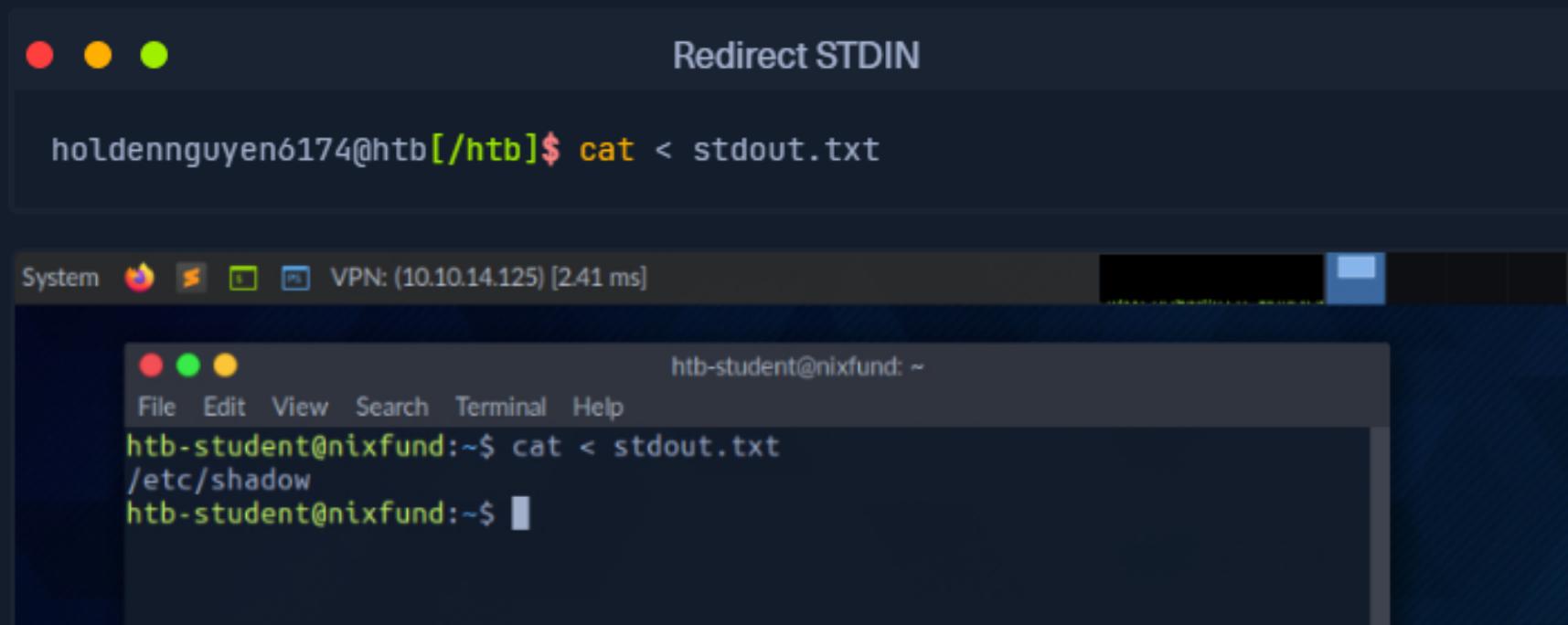
```
holdennguyen0174@htb[~/htb]$ find /etc/ -name shadow 2> stderr.txt 1> stdout.txt
```

The terminal window displays the command and its output. The output shows the contents of the "shadow" file from the "/etc" directory, and two error messages indicating permission denied for private files in "/etc/dovecot/private", "/etc/ssl/private", and "/etc/polkit-1/localauthority".

```
htb-student@nixfund: ~
File Edit View Search Terminal Help
htb-student@nixfund:~$ find /etc/ -name shadow 2> stderr.txt 1> stdout.txt
htb-student@nixfund:~$ cat stdout.txt
/etc/shadow
htb-student@nixfund:~$ cat stderr.txt
find: '/etc/dovecot/private': Permission denied
find: '/etc/ssl/private': Permission denied
find: '/etc/polkit-1/localauthority': Permission denied
htb-student@nixfund:~$
```

## Redirect STDIN

Như đã thấy, kết hợp với file descriptor, chúng ta có thể chuyển hướng lỗi và đầu ra với dấu lớn hơn (>). Điều này cũng hoạt động tương tự với dấu nhỏ hơn (<). Tuy nhiên, dấu nhỏ hơn đóng vai trò là đầu vào tiêu chuẩn (FD 0 - STDIN). Các ký tự này có thể được coi là “direction“ ở dạng mũi tên cho chúng ta biết dữ liệu sẽ được chuyển “từ đâu“ và “đến đâu“. Chúng ta sẽ dùng lệnh `cat` để sử dụng nội dung của file “stdout.txt“ dưới dạng STDIN.



```
holdennguyen0174@htb[/htb]$ cat < stdout.txt
```

```
System  🔍  🌐  VPN: (10.10.14.125) [2.41 ms]
```

```
htb-student@nixfund:~
```

```
File Edit View Search Terminal Help
```

```
htb-student@nixfund:~$ cat < stdout.txt
```

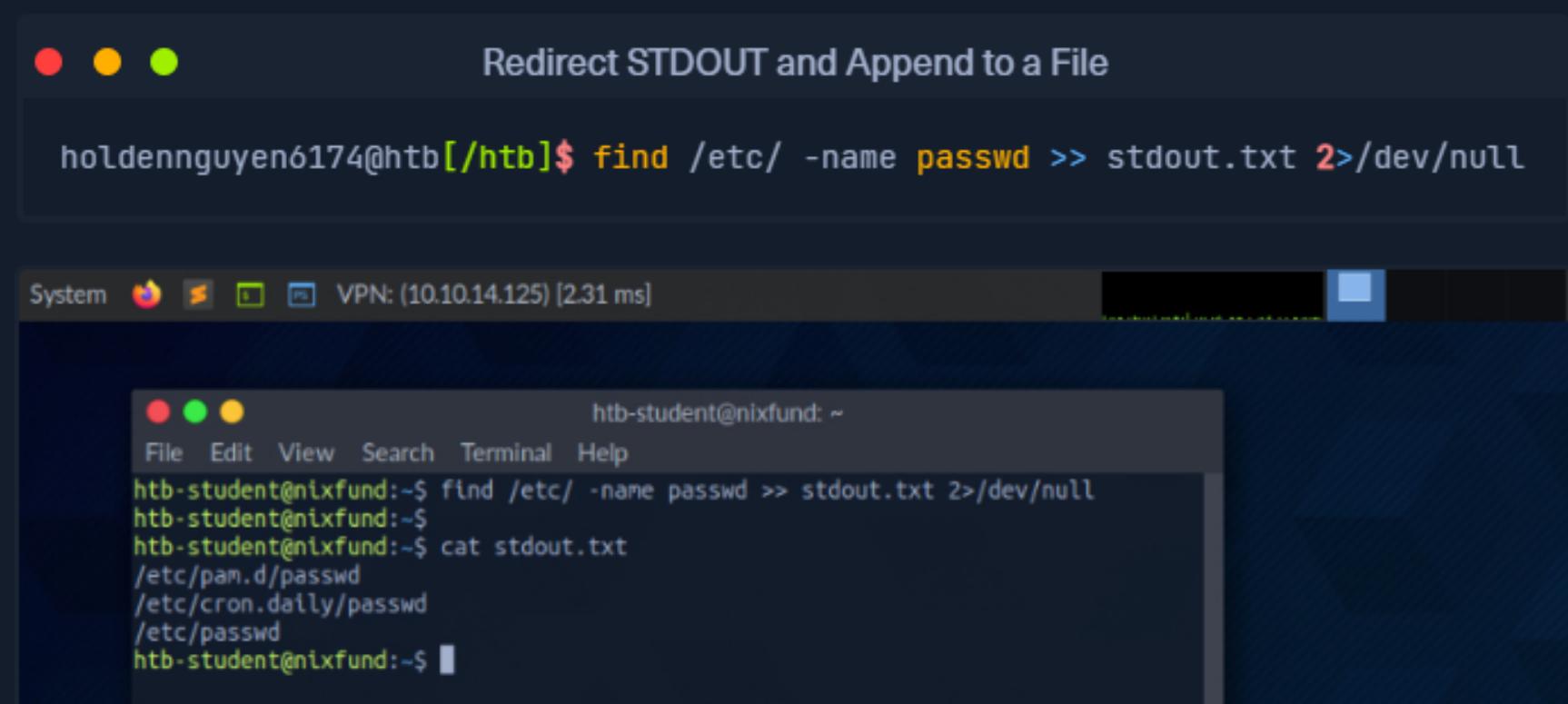
```
/etc/shadow
```

```
htb-student@nixfund:~$
```

## Redirect STDOUT and Append to a File

Khi sử dụng dấu lớn hơn (>) để chuyển hướng **STDOUT**, một file mới sẽ tự động được tạo nếu nó chưa tồn tại. Nhưng giả sử file này đã tồn tại, lúc này nó sẽ bị ghi đè lên mà không yêu cầu xác nhận.

Nếu chúng ta muốn nối thêm **STDOUT** vào file hiện có của mình, có thể sử dụng hai dấu lớn hơn (>>).



```
holdennguyen6174@htb:[/htb]$ find /etc/ -name passwd >> stdout.txt 2>/dev/null
```

```
System  🔥  🌐  📡  VPN: (10.10.14.125) [2.31 ms]
```

```
htb-student@nixfund: ~
File Edit View Search Terminal Help
htb-student@nixfund:~$ find /etc/ -name passwd >> stdout.txt 2>/dev/null
htb-student@nixfund:~$ cat stdout.txt
/etc/pam.d/passwd
/etc/cron.daily/passwd
/etc/passwd
htb-student@nixfund:~$
```

## Redirect STDIN Stream to a File

Chúng ta cũng có thể sử dụng hai dấu nhỏ hơn (`<<`) để thêm một luồng đầu vào tiêu chuẩn (**STDIN**). Để làm được điều đó, có thể sử dụng chức năng gọi là End-Of-File (**EOF**) của file Linux system, chức năng này sẽ xác định kết thúc của input. Ví dụ này, chúng ta sẽ sử dụng lệnh **cat** để đọc đầu vào trực tiếp của chúng ta thông qua stream và chuyển hướng nó tới file có tên “**stream.txt**”.

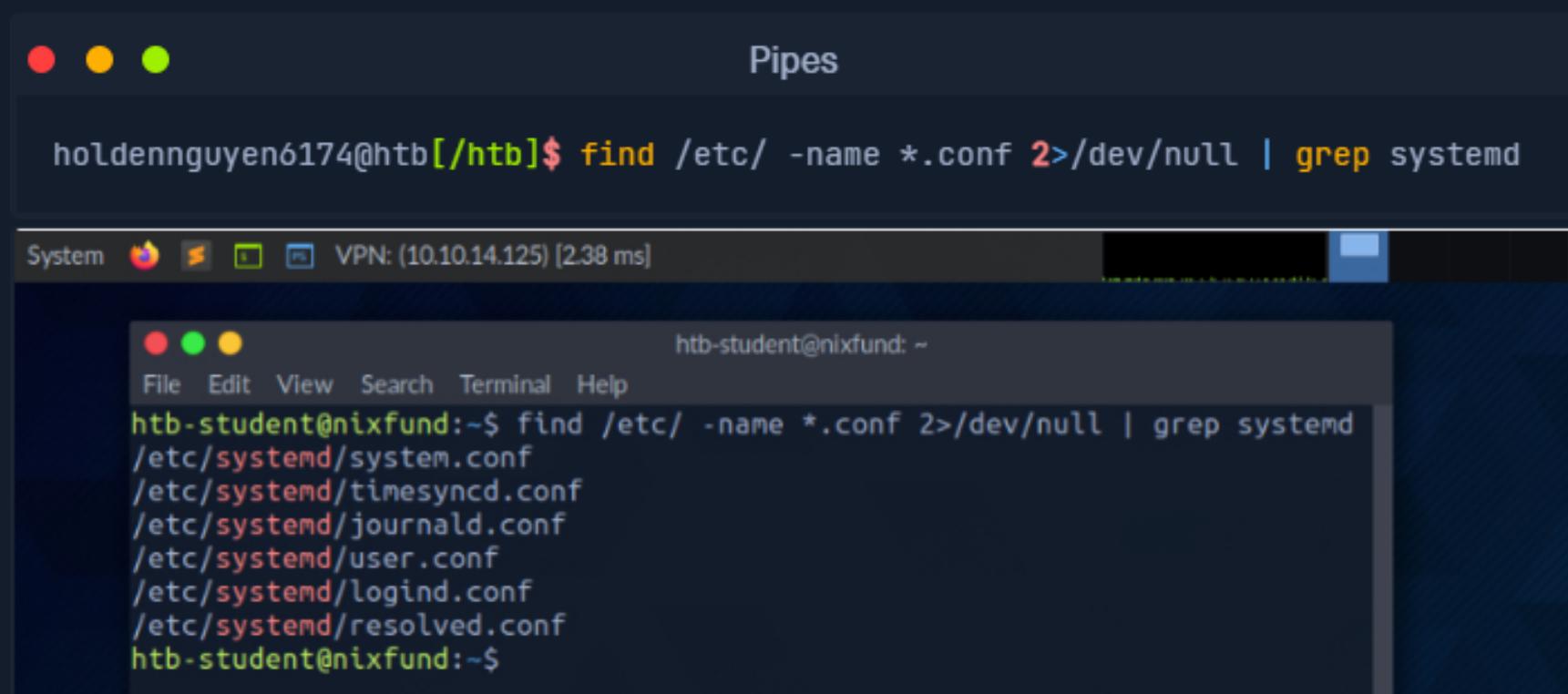
The screenshot shows a terminal window titled "Redirect STDIN Stream to a File". The terminal session starts with the command `htdennguyen6174@htb[/htb]$ cat << EOF > stream.txt`. In the terminal window, the user types the input "Hack", "The", "Box", and "EOF" sequentially, which are then written to the file "stream.txt". Finally, the user runs the command `cat stream.txt` to verify that the contents were correctly read from the file and displayed back to the screen.

```
System  🔥  VPN: (10.10.14.125) [2.37 ms]  htbs-student@nixfund: ~
```

```
htb-student@nixfund:~$ cat << EOF > stream.txt
> Hack
> The
> Box
> EOF
htb-student@nixfund:~$ cat stream.txt
Hack
The
Box
htb-student@nixfund:~$
```

## Pipes

Một cách khác để chuyển hướng **STDOUT** là sử dụng **pipes** (|). Chúng rất hữu ích khi muốn **STDOUT** từ một chương trình được xử lý bởi một chương trình khác. Thường sẽ được dùng nhiều với công cụ **grep** để lọc **STDOUT** theo mẫu đã xác định. Trong ví dụ này, sử dụng lệnh **find** để tìm các tệp trong **/etc/** có phần mở rộng “**.conf**”. Mọi lối đều được chuyển hướng đến “**null device**” (**/dev/null**). Sử dụng **grep** lọc kết quả và chỉ những dòng chứa mẫu “**systemd**” mới được hiển thị.



```
holdennguyen6174@htb[/htb]$ find /etc/ -name *.conf 2>/dev/null | grep systemd
```

```
System   VPN: (10.10.14.125) [2.38 ms]
```

```
htb-student@nixfund: ~
```

```
File Edit View Search Terminal Help
```

```
htb-student@nixfund:~$ find /etc/ -name *.conf 2>/dev/null | grep systemd
/etc/systemd/system.conf
/etc/systemd/timesyncd.conf
/etc/systemd/journald.conf
/etc/systemd/user.conf
/etc/systemd/logind.conf
/etc/systemd/resolved.conf
htb-student@nixfund:~$
```

Không chỉ một lần, còn có thể chuyển hướng tiếp sang một chương trình khác. Ví dụ dùng tool **wc** đếm tổng số kết quả.



```
holdennguyen6174@htb[/htb]$ find /etc/ -name *.conf 2>/dev/null | grep systemd | wc -l
```

```
System   VPN: (10.10.14.125) [2.80 ms]
```

```
htb-student@nixfund: ~
```

```
File Edit View Search Terminal Help
```

```
htb-student@nixfund:~$ find /etc/ -name *.conf 2>/dev/null | grep systemd | wc -l
6
htb-student@nixfund:~$
```

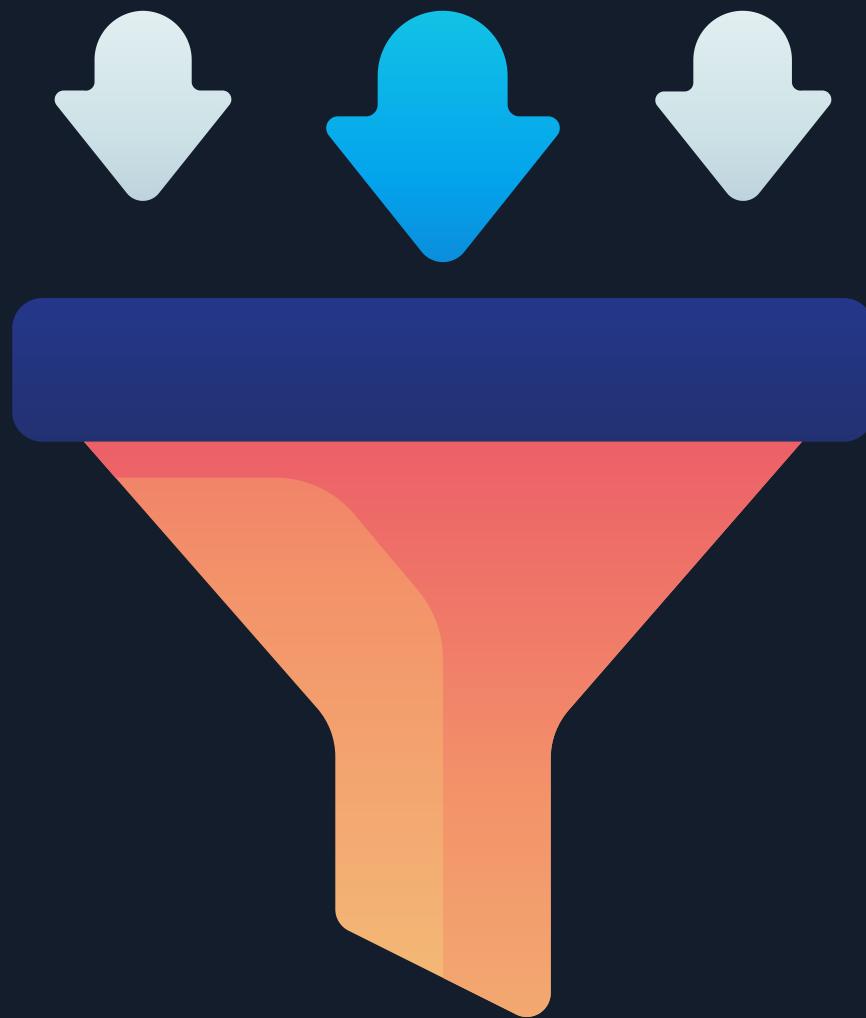
Nguyen Minh Hung  
Minhung



HACKTHEBOX

# Filter Contents

#LinuxFundamentals #Workflow #Part15



more và less là hai công cụ pager cho phép chúng ta xem nội dung file ở chế độ xem tương tác.

## More



```
holdennguyen0174@htb[htb]$ more /etc/passwd
```

Giúp xem nội dung theo trang (pager) ở chế độ tương tác



```
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
<SNIP>
--More--
```

Rời khỏi pager với phím [Q], output vẫn sẽ nằm trong terminal.

## Less



```
holdennguyen0174@htb[htb]$ less /etc/passwd
```



```
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
<SNIP>
:
```

Khi thoát công cụ less với phím [Q], khác với more, output sẽ không còn ở terminal.

## Head

Đôi khi chúng ta chỉ quan tâm những vấn đề ở đầu hoặc cuối nội dung file. Nếu chỉ muốn lấy những dòng đầu tiên của file, có thể sử dụng **head**. Mặc định, **head** sẽ in ra 10 dòng đầu tiên của file hoặc đầu vào đã nhập nếu không có chỉ định khác.



```
holdennguyen6174@htb[/htb]$ head /etc/passwd

root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
```

## Tail

Nếu chỉ muốn xem phần cuối của file hoặc kết quả, hãy dùng tool đối lập với **head** là **tail**, sẽ được trả về 10 dòng cuối cùng.



```
holdennguyen6174@htb[/htb]$ tail /etc/passwd

miredo:x:115:65534::/var/run/miredo:/usr/sbin/nologin
usbmux:x:116:46:usbmux daemon,,,:/var/lib/usbmux:/usr/sbin/nologin
rtkit:x:117:119:RealtimeKit,,,:/proc:/usr/sbin/nologin
nm-openvpn:x:118:120:NetworkManager OpenVPN,,,:/var/lib/openvpn/chroot:/usr/sbin/nologin
nm-openconnect:x:119:121:NetworkManager OpenConnect plugin,,,:/var/lib/NetworkManager:/usr/sbin/nologin
pulse:x:120:122:PulseAudio daemon,,,:/var/run/pulse:/usr/sbin/nologin
beef-xss:x:121:124::/var/lib/beef-xss:/usr/sbin/nologin
lightdm:x:122:125:Light Display Manager:/var/lib/lightdm:/bin/false
do-agent:x:998:998:/home/do-agent:/bin/false
user0:x:1000:1000:,,,:/home/user0:/bin/bash
```

## Sort

Đôi khi chúng ta muốn sắp xếp các kết quả mong muốn theo thứ tự bảng chữ cái hoặc số để có cái nhìn tổng quan hơn. Đối với điều này, có thể dùng công cụ gọi là **sort**.

```
holdennguyen6174@htb[/htb]$ cat /etc/passwd | sort

_apt:x:104:65534::/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
cry0l1t3:x:1001:1001::/home/cry0l1t3:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
dnsmasq:x:107:65534:dnsmasq,,,:/var/lib/misc:/usr/sbin/nologin
dovecot:x:114:117:Dovecot mail server,,,:/usr/lib/dovecot:/usr/sbin/nologin
dovenuill:x:115:118:Dovecot login user,,,:/nonexistent:/usr/sbin/nologin
ftp:x:113:65534::/srv/ftp:/usr/sbin/nologin
games:x:5:60:games:/usr/games:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
htb-student:x:1002:1002::/home/htb-student:/bin/bash
<SNIP>
```

Giờ đây output đã được sắp xếp theo thứ tự alphabet.

## Grep

Chúng ta thường xuyên tìm kiếm các kết quả cụ thể chứa các mẫu định sẵn. Một trong những công cụ cho việc này là **grep**, cung cấp nhiều tính năng khác nhau.

Ví dụ bằng việc tìm user dùng shell mặc định là “/bin/bash”



```
holdennguyen6174@htb[/htb]$ cat /etc/passwd | grep "/bin/bash"

root:x:0:0:root:/root:/bin/bash
mrb3n:x:1000:1000:mrb3n:/home/mrb3n:/bin/bash
cry0l1t3:x:1001:1001::/home/cry0l1t3:/bin/bash
htb-student:x:1002:1002::/home/htb-student:/bin/bash
```

Một khả năng khác đó là loại trừ các kết quả cụ thể. Đối với việc này, option “**-v**” được sử dụng với **grep**. Ví dụ dưới đây sẽ loại trừ tất cả user đã tắt shell mặc định bằng tên “/bin/false” hoặc “/usr/bin/nologin”



```
holdennguyen6174@htb[/htb]$ cat /etc/passwd | grep -v "false|\nlogin"

root:x:0:0:root:/root:/bin/bash
sync:x:4:65534:sync:/bin:/bin/sync
postgres:x:111:117:PostgreSQL administrator,,,:/var/lib/postgresql:/bin/bash
user6:x:1000:1000:,,,:/home/user6:/bin/bash
```

## Cut

Các kết quả thường sẽ có các dấu phân cách, để xóa chúng và hiển thị các từ trên một dòng ở vị trí cụ thể, có thể dùng công cụ **cut**. Sử dụng option “**-d**“ và đặt dấu phân cách thành ký tự dấu hai chấm colon (:), xác định bằng option “**-f**“ vị trí trong dòng mà chúng ta muốn output.

```
holdennguyen6174@htb[/htb]$ cat /etc/passwd | grep -v "false\|nologin" | cut -d ":" -f1
root
sync
mrb3n
cry0l1t3
htb-student
```

## Tr

Một cách khác để thay thế các ký tự nhất định từ một dòng bằng các ký tự chỉ định đó là dùng công cụ **tr**. Chúng ta sẽ nhập ký tự muốn thay thế là dấu hai chấm (:) và ký tự chỉ định là dấu cách ( ) như ví dụ sau đây

```
holdennguyen6174@htb[/htb]$ cat /etc/passwd | grep -v "false\|nologin" | tr ":" " "
root x 0 0 root /root /bin/bash
sync x 4 65534 sync /bin /bin/sync
mrb3n x 1000 1000 mrb3n /home/mrb3n /bin/bash
cry0l1t3 x 1001 1001 /home/cry0l1t3 /bin/bash
htb-student x 1002 1002 /home/htb-student /bin/bash
```

## Column

Như ở ví dụ trang trước, kết quả thường trình bày không rõ ràng nên công cụ **column** rất phù hợp để hiển thị chúng ở dạng bảng bằng cách sử dụng option “**-t**”.

```
holdennguyen6174@htb[/htb]$ cat /etc/passwd | grep -v "false\|nologin" | tr ":" " " | column -t
root      x  0    0    root          /root      /bin/bash
sync      x  4  65534  sync         /bin       /bin/sync
mrb3n     x 1000 1000  mrb3n        /home/mrb3n /bin/bash
cry0l1t3  x 1001 1001  /home/cry0l1t3 /bin/bash
htb-student x 1002 1002  /home/htb-student /bin/bash
```

## Awk

Mỗi dòng kết quả ở ví dụ trên vẫn còn khá nhiều dữ kiện, để nó trở nên đơn giản cho việc sort kết quả. công cụ (**g**)**awk** giúp hiển thị kết quả đầu tiên (**\$1**) và cuối cùng (**\$NF**) của dòng:

```
holdennguyen6174@htb[/htb]$ cat /etc/passwd | grep -v "false\|nologin" | tr ":" " " | awk '{print $1, $NF}'
root /bin/bash
sync /bin/sync
mrb3n /bin/bash
cry0l1t3 /bin/bash
htb-student /bin/bash
```

## Sed

Sẽ có lúc chúng ta muốn thay đổi cụm từ cụ thể trong toàn bộ file hoặc đầu vào tiêu chuẩn (STDIN). Một trong những tool có thể dùng để chỉnh sửa luồng (stream editor) là **sed**. Chúng ta thường sẽ dùng **sed** để thay thế văn bản. Nó sẽ tìm các mẫu đã chỉ định và thay thế bằng mẫu khác cũng do ta chỉ định. Hãy thay thế từ “**bin**“ bằng “**HTB**“ ở ví dụ trước đó  
**sed 's/bin/HTB/g'**

“**s**“ flag ở đầu là viết tắt của lệnh thay thế. Sau đó, chỉ định mẫu muốn thay thế. Sau dấu gạch chéo slash (/), nhập mẫu chỉ định để thế vào ở vị trí thứ ba. Cuối cùng, sử dụng “**g**“ flag là viết tắt của thay thế tất cả các từ trùng khớp.

```
holdennguyen6174@htb[/htb]$ cat /etc/passwd | grep -v "false\|nologin" | tr ":" " " | awk '{print $1, $NF}' | sed 's/bin/HTB/g'

root /HTB/bash
sync /HTB/sync
mrb3n /HTB/bash
cry0l1t3 /HTB/bash
htb-student /HTB/bash
```

## Wc

Cuối cùng nhưng không kém phần quan trọng, sẽ rất hữu ích nếu ta biết được có bao nhiêu cặp đã được thay thế. Để tránh đếm các dòng hay ký tự theo cách thủ công, sử dụng tool **wc**. Với option “**-l**“, chúng ta sẽ chỉ đếm số dòng.

```
holdennguyen6174@htb[/htb]$ cat /etc/passwd | grep -v "false\|nologin" | tr ":" " " | awk '{print $1, $NF}' | wc -l

5
```

Nguyen Minh Hung  
Minhung



HACKTHEBOX

# Permission Management

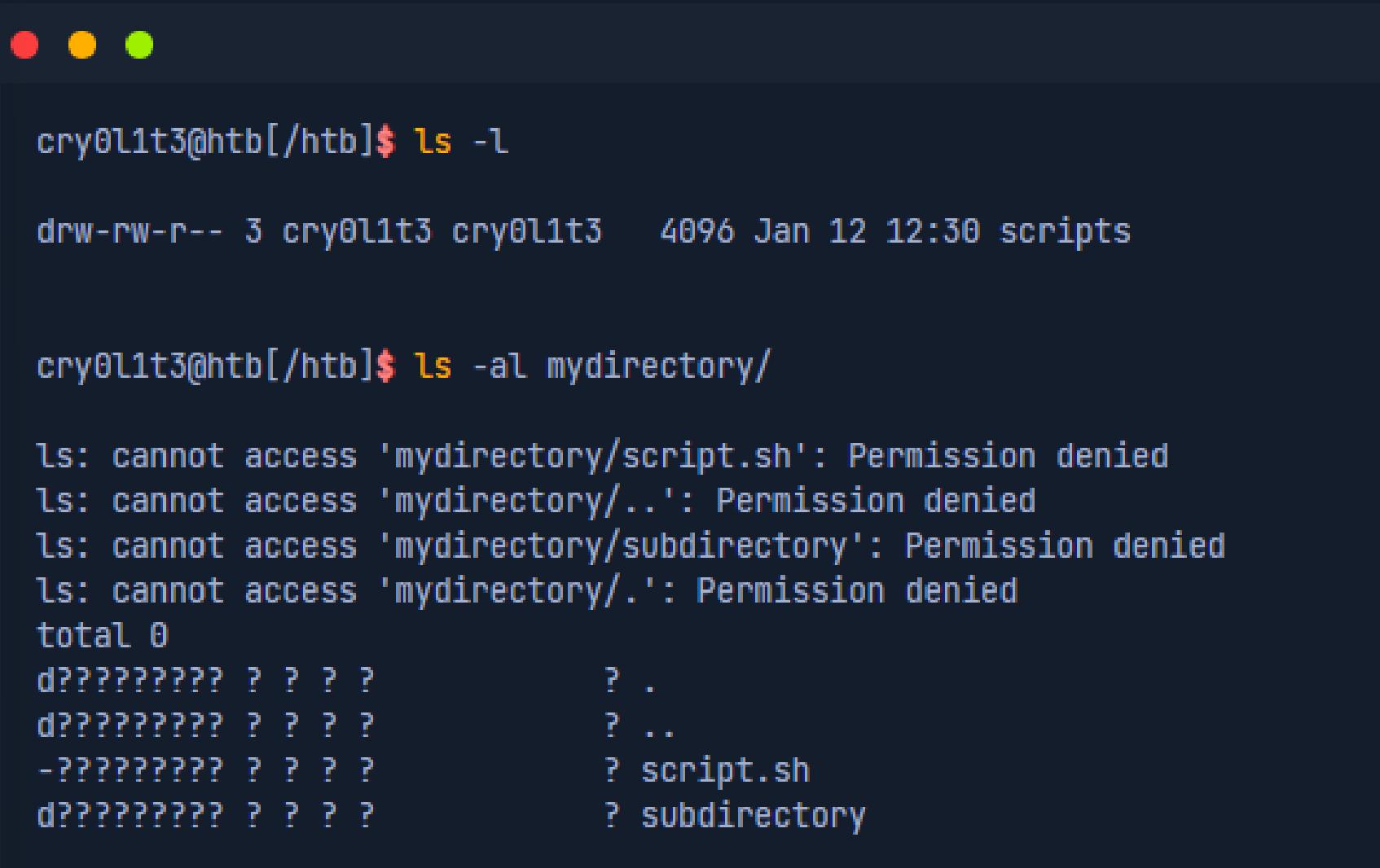
#LinuxFundamentals #Workflow #Part16



Trong Linux, permission (quyền) được gán cho user và group. Mỗi user có thể là thành viên của các nhóm khác nhau và membership trong các group này cung cấp cho user những permission bổ sung.

Mỗi file và directory thuộc về một user hoặc một group cụ thể. Khi tạo các file hoặc directory mới, nó sẽ thuộc về user hiện tại cũng như group của user đó.

Khi user muốn truy cập nội dung của một Linux directory, họ phải có quyền thực thi (**execute permission**) trên thư mục đó. Nếu không có quyền này, user không thể truy cập và sẽ nhận thông báo lỗi “**Permission Denied**“.



Cần có quyền **execute** để duyệt một thư mục, bất kể cấp độ user. Tuy nhiên, quyền **execute** trên một directory không đồng nghĩa việc user có thể thực thi hoặc chỉnh sửa bất kỳ file hay nội dung nào bên trong directory đó, nó chỉ cho phép duyệt qua file và truy cập vào nội dung bên trong directory thôi.

Để thực thi file trong directory, cần quyền **execute** trên đúng file đó. Để chỉnh sửa nội dung một directory (tạo, xóa hay đổi tên file, thư mục con), cần quyền **write** trên directory đó.

Permission trên Linux system dựa trên hệ bát phân và có ba loại khác nhau mà một file hoặc directory có thể được gán:

- (r) - Read (Đọc)
  - (w) - Write (Chỉnh sửa)
  - (x) - Execute (Thực thi)

Các permission có thể được cài đặt cho owner, group và các đối tượng khác (others) như ví dụ dưới đây.

## Change Permissions

Chúng ta có thể sửa đổi permission bằng cách sử dụng lệnh **chmod**, cùng nhóm quyền tham chiếu (**u** - owner, **g** - Group, **o** - others, **a** - All users) và **[+]** hoặc **[-]** để thêm hoặc xóa các quyền được chỉ định. Trong ví dụ sau, một user tạo một shell script mới được sở hữu bởi user đó, không thể thực thi, và chúng ta sẽ thêm read/write permission cho tất cả user.



```
cry0l1t3@htb[~/htb]$ ls -l shell  
-rwxr-x--x    1 cry0l1t3  htbtteam 0 May  4 22:12 shell
```

Chúng ta có thể thêm **read** permission cho tất cả user và quan sát kết quả dưới đây



```
cry0l1t3@htb[~/htb]$ chmod a+r shell && ls -l shell  
-rwxr-xr-x    1 cry0l1t3  htbtteam 0 May  4 22:12 shell
```

Chúng ta cũng có thể thiết lập read permission cho tất cả user bằng cách sử dụng hệ số tám (**octal value**) để gán.

```
cry0l1t3@htb[~/htb]$ chmod 754 shell && ls -l shell
-rwxr-xr-- 1 cry0l1t3 htbtteam 0 May 4 22:12 shell
```

Hãy xem xét tất cả các cách biểu diễn tương ứng dưới đây để hiểu hơn về cách gán permission:

<b>Binary Notation:</b>	4 2 1   4 2 1   4 2 1
<b>Binary Representation:</b>	-----
<b>Octal Value:</b>	1 1 1   1 0 1   1 0 0
<b>Permission Representation:</b>	-----
	7   5   4
	-----
	r w x   r - x   r - -

Nếu chúng ta tính tổng các bit thiết lập ở phần **Binary Representation** với giá trị mỗi bit ở phần **Binary Notation**, sẽ ra được **Octal Value**. **Permission Representation** đại diện cho các bit thiết lập ở **Binary Representation** ở dạng ký tự, với mục đích nhận ra cá permission đã thiết lập một cách dễ dàng.

## Change Owner

Để thay đổi owner và group được gán của một file hoặc directory, chúng ta có thể sử dụng lệnh **chown**. Cú pháp như sau:



### Syntax - chown

```
cry0l1t3@htb[/htb]$ chown <user>:<group> <file/directory>
```

Trong ví dụ này, “shell“ có thể được thay thế bằng file hoặc thư mục bất kỳ.



### Syntax - chown

```
cry0l1t3@htb[/htb]$ chown root:root shell && ls -l shell  
-rwxr-xr-- 1 root root 0 May  4 22:12 shell
```

## SUID & GUID

Bên cạnh việc gán permission trực tiếp cho user và group, chúng ta cũng có thể cấu hình các quyền đặc biệt cho file bằng cách cài đặt các bit **Set User ID (SUID)** và **Set Group ID (GUID)**. Những **SUID/GUID** bit cho phép user có thể chạy chương trình với quyền của user khác. Administrator thường sử dụng cách này để cấp user quyền đặc biệt cho một số ứng dụng hoặc file nhất định. Chữ cái “**s**“ được dùng thay thế cho “**x**“. Khi thực thi một chương trình, SUID/GUID của file owner sẽ được sử dụng.

Thường có trường hợp administrator chưa hiểu về ứng dụng nhưng vẫn gán SUID/GUID bit, dẫn đến một rủi ro bảo mật cao. Chẳng hạn như những chương trình có tính năng cho phép thực thi một shell từ pager, ví dụ như “**journalctl**“.

Nếu administrator thiết lập SUID bit cho “**journalctl**“, bất kỳ user nào với quyền truy cập ứng dụng này đều có thể thực thi shell như **root** user. Những thông tin thêm về vấn đề này và những ứng dụng khác có thể được tìm thấy tại [GTFObins](#).

## Sticky Bit

Sticky bit là một loại permission của file trong Linux có thể thiết lập trên directory. Loại permission này cung cấp thêm một lớp bảo mật khi kiểm soát việc xóa và đổi tên các file trong một thư mục. Nó thường được sử dụng trên các thư mục được chia sẻ bởi nhiều user để ngăn một user vô tình xóa hoặc đổi tên các file quan trọng đối với người khác.

Ví dụ: trong một directory chính được chia sẻ, nơi nhiều user có quyền truy cập vào cùng thư mục, administrator có thể đặt sticky bit trên thư mục này để đảm bảo rằng chỉ owner của file, owner của directory hoặc root user mới có thể xóa hoặc đổi tên các file trong thư mục. Điều này có nghĩa là những user khác không thể xóa hay đổi tên các file trong directory này vì họ không có các permission cần thiết. Cách này cung cấp một lớp bảo mật bổ sung để bảo vệ các file quan trọng, vì chỉ những user có quyền truy cập cần thiết mới có thể xóa hoặc đổi tên file. Đặt sticky bit trên một directory đảm bảo rằng chỉ owner, owner của thư mục hoặc root user mới có thể thay đổi các file trong directory.

Khi một sticky bit được đặt trên một directory, nó được biểu thị bằng chữ cái “**t**” trong execute permission của permission thư mục. Ví dụ như nếu một directory có permission “**rwxrwxrwt**”, điều đó có nghĩa là sticky bit đã được đặt, tăng mức độ bảo mật bổ sung để không ai khác ngoài owner hoặc root user có thể xóa hoặc đổi tên các file hoặc directory trong thư mục này.



### Syntax - chown

```
cry0l1t3@htb[ /htb]$ ls -l
drw-rw-r-t 3 cry0l1t3 cry0l1t3 4096 Jan 12 12:30 scripts
drw-rw-r-T 3 cry0l1t3 cry0l1t3 4096 Jan 12 12:32 reports
```

Ở ví dụ này, chúng ta có thể thấy cả hai thư mục đều có sticky bit. Tuy nhiên thư mục **reports** có chữ “**T**” viết hoa và thư mục **scripts** có chữ “**t**” thường.

Nếu sticky bit được viết hoa (**T**), thì điều này có nghĩa là tất cả người dùng khác không có quyền **execute** (**x**) và do đó, không thể xem nội dung của thư mục cũng như không thể chạy kỳ chương trình nào từ thư mục đó. Sticky bit chữ thường (**t**) là sticky bit nơi quyền **execute** (**x**) đã được đặt.

Nguyen Minh Hung  
Minhung



HACKTHEBOX

# Linux Shortcut

#LinuxFundamentals #Tip&Trick



**TIPS &  
TRICKS**



## Auto-Complete

[TAB] - Tự động hoàn thành, đưa ra những option gợi ý khác nhau dựa trên STDIN chúng ta cung cấp. Có thể đề xuất cụ thể các dictionary trong môi trường chúng ta đang thao tác, các lệnh bắt đầu với số ký tự giống với những gì đã nhập.

## Cursor Movement

[CTRL] + A - Di chuyển con trỏ về **điểm bắt đầu** dòng hiện tại.

[CTRL] + E - Di chuyển con trỏ đến **điểm cuối** dòng hiện tiện.

[CTRL] + [⬅]/[➡] - Nhảy tới **điểm bắt đầu** của từ hiện tại/trước đó.

[ALT] + B/ F - Nhảy lùi/tiến một từ.

## Erase The Current Line

[CTRL] + U - Xóa mọi thứ từ vị trí hiện tại của con trỏ đến **điểm bắt đầu** dòng.

[CTRL] + K - Xóa mọi thứ từ vị trí hiện tại của con trỏ đến **vị trí cuối** dòng.

[CTRL] + W - Xóa từ đứng trước vị trí con trỏ.

## Paste Erased Contents

[CTRL] + Y - Dán văn bản hoặc từ đã xóa.

## Ends Task

[CTRL] + C - Kết thúc task/process hiện tại bằng cách gửi tín hiệu SIGINT. Ví dụ: đây có thể là quá trình scan chạy bằng một tool. Nếu chúng ta đang xem quá trình scan, có thể dừng nó/loại bỏ process đó bằng phím tắt này. Process sẽ bị loại bỏ mà không yêu cầu chúng ta xác nhận.

## End-of-File (EOF)

[CTRL] + D - Đóng STDIN pipe, còn được gọi là End-of-File (EOF) hoặc End-of-Transmission.

## Clear Terminal

[CTRL] + L - Dọn sạch terminal. Một cách khác cho phím tắt này đó là lệnh clear.

## Background a Process

[CTRL] + Z - Tạm dừng process hiện tại bằng cách gửi tín hiệu SIGTSTP.

## Search Through Command History

[CTRL] + R - Tìm kiếm câu lệnh đã nhập phù hợp với mẫu tìm kiếm chỉ định thông qua lịch sử các câu lệnh.

[↑] / [↓] - Chuyển đến lệnh trước/tiếp theo trong lịch sử các câu lệnh.

## Switch Between Applications

[ALT] + [TAB] - Chuyển đổi giữa các ứng dụng đã mở.

## Zoom

[CTRL] + [+] - Phóng to.

[CTRL] + [-] - Thu nhỏ.

Nguyen Minh Hung  
Minhung



HACKTHEBOX

# Linux Security

#LinuxFundamentals #Tip&Trick



Tất cả hệ thống máy tính đều có nguy cơ bị xâm nhập. Một số có nhiều rủi ro hơn những cái khác, chẳng hạn như **web server** kết nối internet lưu trữ nhiều ứng dụng web phức tạp. Các Linux system cũng ít bị nhiễm virus hơn so với hệ điều hành Windows và không cung cấp bề mặt tấn công lớn như các **Active Directory domain-joined host**. Tuy nhiên, điều cần thiết là vẫn phải luôn có một số nguyên tắc cơ bản nhất định để bảo mật Linux system bất kỳ.

Một trong những biện pháp bảo mật quan trọng nhất của hệ điều hành Linux là luôn cập nhật hệ điều hành và các package đã cài đặt. Thực hiện điều này bằng lệnh như:



```
holdennguyen6174@htb[/htb]$ apt update && apt dist-upgrade
```

Nếu các quy tắc firewall không được thiết lập phù hợp ở network level, chúng ta có thể sử dụng Linux **firewall** hoặc **iptables** để hạn chế traffic truy cập vào và ra khỏi host.

Nếu SSH được mở trên server, cấu hình nên được thiết lập để không cho phép đăng nhập bằng mật khẩu cũng như không cho phép root user đăng nhập qua SSH. Điều quan trọng nữa đó là tránh đăng nhập và quản trị hệ thống với tư cách là root user bất cứ khi nào có thể và quản lý đầy đủ các quyền truy cập. Quyền truy cập của user phải được xác định dựa trên nguyên tắc đặc quyền tối thiểu (least privilege).

Ví dụ: nếu user cần chạy một lệnh với root permission, thì lệnh đó phải được chỉ định trong cấu hình sudoers thay vì cấp cho họ toàn quyền sudo.

Một cơ chế bảo vệ phổ biến khác có thể được sử dụng là fail2ban. Công cụ này đếm số lần đăng nhập không thành công, nếu user đã đạt đến số lượng tối đa, host đã cố kết nối sẽ được xử lý theo quy định đã cấu hình.

Thêm một điều quan trọng nữa là phải kiểm tra hệ thống định kỳ để đảm bảo rằng không tồn tại các vấn đề có thể tạo điều kiện cho việc leo thang đặc quyền (privilege escalation), chẳng hạn như out-of-date kernel, các vấn đề về user permission, các world-writable file, các công việc định kỳ (cron jobs) bị cấu hình sai hay service bị cấu hình sai. Nhiều administrator thường quên mất khả năng một số phiên bản kernel phải được cập nhật thủ công.

Một tùy chọn khác để hạn chế truy cập đến hệ thống Linux là **Security-Enhanced Linux (SELinux)** hoặc **AppArmor**. Đây là module bảo mật của kernel có thể được sử dụng cho các chính sách kiểm soát truy cập bảo mật. Trong SELinux, mọi process, file, directory và object đều được gắn nhãn.

Các policy với quy tắc sẽ được tạo để kiểm soát quyền truy cập giữa các process đến object đã gắn nhãn và được thực thi bởi kernel. Điều này có nghĩa là quyền truy cập có thể được thiết lập để kiểm soát user và ứng dụng nào có thể truy cập tài nguyên nào. SELinux cung cấp các điều khiển truy cập rất chi tiết, chẳng hạn như chỉ định ai có thể thêm vào file hoặc di chuyển file.

Ngoài ra còn có các ứng dụng và dịch vụ khác nhau như **Snort**, **chkrootkit**, **rkhunter**, **Lynis** và những ứng dụng và dịch vụ khác có thể đóng góp vào tính bảo mật của Linux.

Một số cài đặt bảo mật nên được thực hiện chẳng hạn như:

- Gỡ bỏ hoặc vô hiệu hóa tất cả các service và software.
- Xóa tất cả service dùng cơ chế xác thực không mã hóa.
- Đảm bảo Network Time Protocol (NTP) được bật và Syslog đang chạy.
- Đảm bảo rằng mỗi user có tài khoản riêng.
- Thúc đẩy việc sử dụng mật khẩu mạnh.
- Thiết lập thời hạn cho mật khẩu và hạn chế sử dụng mật khẩu trước đó.
- Khóa tài khoản người dùng sau khi đăng nhập thất bại.
- Vô hiệu hóa tất cả các SUID/SGID bit không mong muốn.

Danh sách kể trên chắc chắn vẫn chưa thể đầy đủ, vì an toàn không phải là một sản phẩm mà là quá trình. Điều này có nghĩa là các bước cụ thể phải luôn được thực hiện để bảo vệ hệ thống tốt hơn và điều đó phụ thuộc vào mức độ hiểu biết của người quản trị hệ thống của họ. Các administrator càng quen thuộc với hệ thống và càng được đào tạo thường xuyên thì các biện pháp phòng ngừa và bảo mật của họ sẽ càng tốt và an toàn hơn.

# TCP Wrappers

**TCP wrapper** là một cơ chế bảo mật được sử dụng trong các hệ thống Linux cho phép quản trị viên hệ thống kiểm soát service nào được phép truy cập vào hệ thống. Nó hoạt động bằng cách hạn chế quyền truy cập vào một số dịch vụ nhất định dựa trên **host name** hoặc **IP address** của user yêu cầu quyền truy cập. Khi một client cố gắng kết nối với service, trước tiên, system sẽ tham khảo các quy tắc được chỉ định trong **TCP wrapper configuration file** để xác định địa chỉ IP của client. Nếu địa chỉ IP đúng với tiêu chí trong configuration file, thì system sẽ cấp cho client quyền truy cập vào service. Tuy nhiên, nếu các tiêu chí không được đáp ứng, kết nối sẽ bị từ chối, cung cấp thêm một lớp bảo mật cho dịch vụ. TCP wrapper sử dụng các configuration file sau:

- **/etc/hosts.allow**
- **/etc/hosts.deny**

Tóm lại, file **/etc/hosts.allow** chỉ định service và host nào được phép truy cập vào hệ thống, trong khi file **/etc/hosts.deny** chỉ định service và host nào không được phép truy cập. Các file này có thể được cấu hình bằng cách thêm vào các quy tắc cụ thể bên trong file.

# TCP Wrappers

## /etc/hosts.allow

● ● ●

/etc/hosts.allow

```
holdennguyen6174@htb[/htb]$ cat /etc/hosts.allow

# Allow access to SSH from the local network
sshd : 10.129.14.0/24

# Allow access to FTP from a specific host
ftpd : 10.129.14.10

# Allow access to Telnet from any host in the inlanefreight.local domain
telnetd : .inlanefreight.local
```

## /etc/hosts.deny

● ● ●

/etc/hosts.deny

```
holdennguyen6174@htb[/htb]$ cat /etc/hosts.deny

# Deny access to all services from any host in the inlanefreight.com domain
ALL : .inlanefreight.com

# Deny access to SSH from a specific host
sshd : 10.129.22.22

# Deny access to FTP from hosts with IP addresses in the range of 10.129.22.0 to 10.129.22.255
ftpd : 10.129.22.0/24
```

Điều quan trọng cần nhớ là **thứ tự** của các quy tắc trong file rất quan trọng. Quy tắc đầu tiên khớp với service và host được yêu cầu là quy tắc sẽ được áp dụng. Cũng cần lưu ý rằng các **TCP wrapper** không phải là sự thay thế cho **firewall**, vì chúng bị giới hạn bởi thực tế là chỉ có thể kiểm soát quyền truy cập vào các **service** chứ không phải các **port**.



# HTB ACADEMY

## Course: Linux Fundamentals



Facebook: Minhung  
Instagram: \_\_minhung\_\_

