

CHƯƠNG 1

BUỔI THỰC HÀNH SỐ 1

Chương này giới thiệu về các công cụ quản trị mạng thường dùng trong hệ điều hành Linux (phiên bản Ubuntu 18.04) và công cụ hỗ trợ mô phỏng mạng Kathará trên nền tảng ảo hóa Docker. Trọng tâm của chương sẽ là các bài thực hành đơn giản được hướng dẫn chi tiết từng bước nhằm giúp sinh viên làm quen với việc sử dụng Kathará trong mô phỏng mạng máy tính.

1.1 CÔNG CỤ QUẢN TRỊ MẠNG TRÊN LINUX

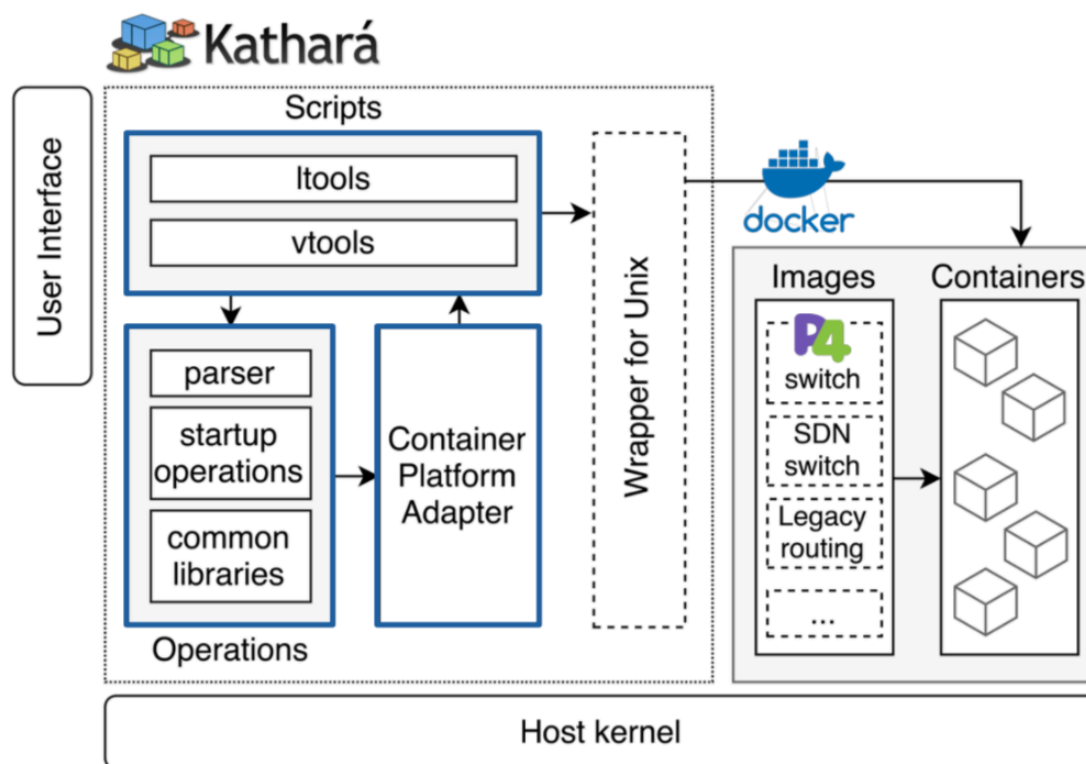
- ping: công cụ cho phép gửi 1 gói tin đến từ địa chỉ IP máy nguồn đến địa chỉ IP máy đích. Nếu như địa chỉ IP máy đích là tồn tại, ping trên máy đích sẽ tự động hồi đáp bằng 1 gói tin ngược lại máy nguồn. Cả 2 gói tin ping này đều chứa thông điệp ICMP - Internet Control Message Protocol.
- ifconfig: công cụ cho phép cấu hình giao diện mạng (network interface) của máy, ví dụ: đặt địa chỉ IP và netmask, tắt hoặc mở giao diện mạng.
- tcpdump: công cụ cho phép bắt các gói tin luân chuyển qua một hoặc nhiều giao diện mạng. Công cụ này cung cấp 2 chức năng lớn, là packet sniffing và packet analyze với thư viện lệnh phong phú.
- route: công cụ cho phép xem bảng dẫn đường hiện tại của host.
- traceroute: công cụ cho phép lần vết của dữ liệu luân chuyển qua host.

1.2 CÔNG CỤ MÔ PHỎNG MẠNG KATHARÁ

Kathará là một công cụ mã nguồn mở có chi phí thấp, hiệu suất cao được triển khai trên nền tảng ảo hóa Docker nhằm thực hiện giả lập (emulate) một hoặc nhiều hệ thống mạng từ đơn giản cho đến phức tạp. Thực chất *Kathará* là phiên bản nâng cao của *Netkit*, một công cụ mô phỏng mạng cũng đã đạt được những thành công trong hỗ trợ dạy và học Mạng máy tính tại các Trường Đại học có đào tạo chuyên ngành Công nghệ thông tin.

Điểm nổi trội của *Kathará* so với người tiền nhiệm của nó là nhờ vào nền tảng ảo hóa *Docker* để mô phỏng các thiết bị thường gặp trong mạng như Router, Switch, Web Server, DNS Server...dưới dạng các máy ảo Linux chỉ có giao diện dòng lệnh. Các thiết bị mạng (máy ảo) này hoạt động dưới hình thức là các bộ chứa (Container) được quản lý bởi *Docker*. Một số ưu điểm nổi bật

của môi trường ảo hóa dựa trên các bộ chứa mà *Docker* đem lại đó là: linh động, nhanh, nhẹ, đồng nhất và đóng gói. Các máy ảo được tạo ra có thể dễ dàng được tùy chỉnh (customize) lại theo ý đồ của người quản trị mạng. Chẳng hạn, triển khai Apache Spark cho các hệ thống ứng dụng tính toán dữ liệu lớn (Big Data). Mô hình tổng quan của môi trường mạng ảo được mô phỏng bởi *Kathará* trên nền tảng *Docker* được miêu tả trong hình 1.1.



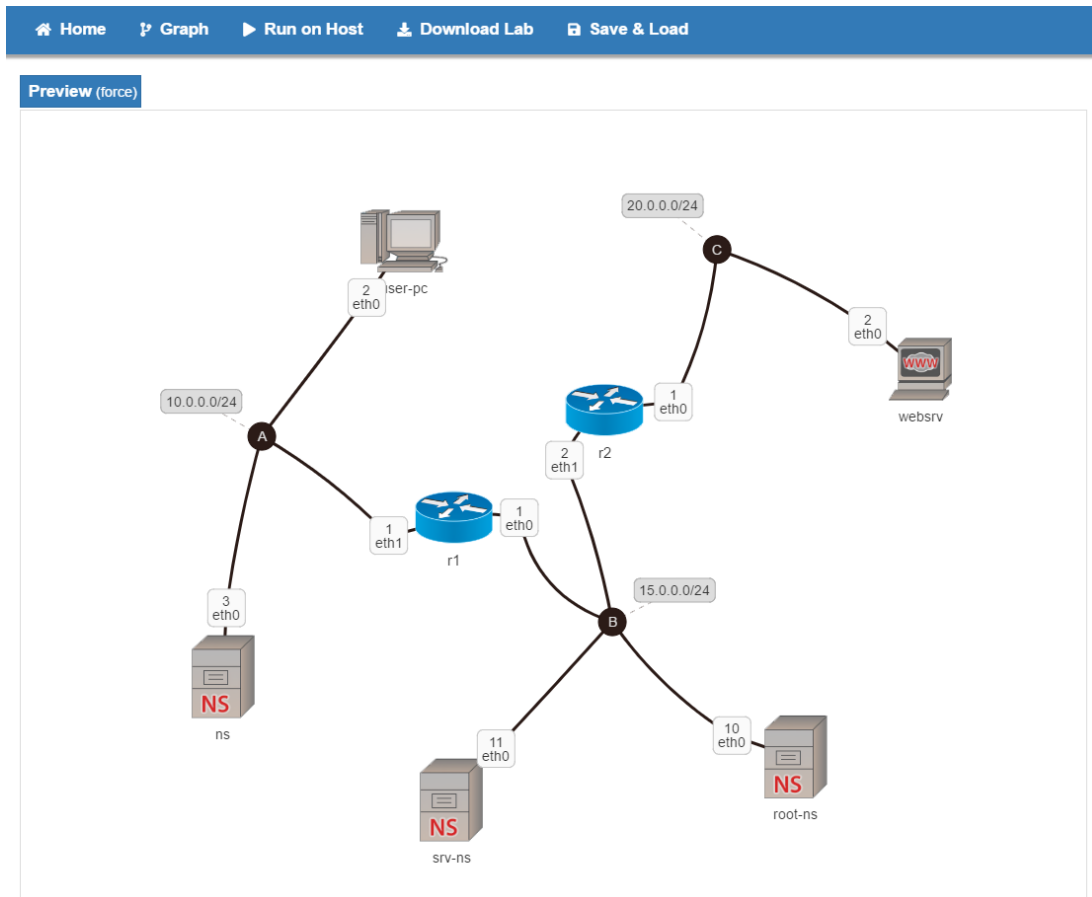
Hình 1.1 Mô hình môi trường mạng ảo *Kathará* trên *Docker*. Nguồn: *Kathará*

Trong các đánh giá về hiệu năng hoạt động của các phần mềm mô phỏng mạng ảo (Network Emulator), *Kathará* đạt được những đánh giá tốt theo nhiều tiêu chí khác nhau (về hiệu suất sử dụng CPU, tiêu hao bộ nhớ, thời gian khởi động và thực thi...). Chính vì vậy, trong tài liệu hướng dẫn thực hành Mạng máy tính CT112, chúng tôi lựa chọn giới thiệu và sử dụng *Kathará* mô phỏng các hệ thống mạng minh họa cho kiến thức lý thuyết về Mạng máy tính đã giảng dạy.

Việc sử dụng *Kathará* để mô phỏng mạng có thể được hỗ trợ thêm bởi công cụ *Netkit Lab Generator*¹. Đây là công cụ được phát triển kèm theo

¹ <http://www.kathara.org/tools/nlg/>.

Kathará với mục đích cung cấp môi trường đồ họa làm giảm tính phức tạp trong việc xây dựng mô hình mạng dưới dạng các máy ảo. Hình 1.2 dưới đây mô tả một mạng ảo được xây dựng bởi công cụ *Netkit Lab Generator*:

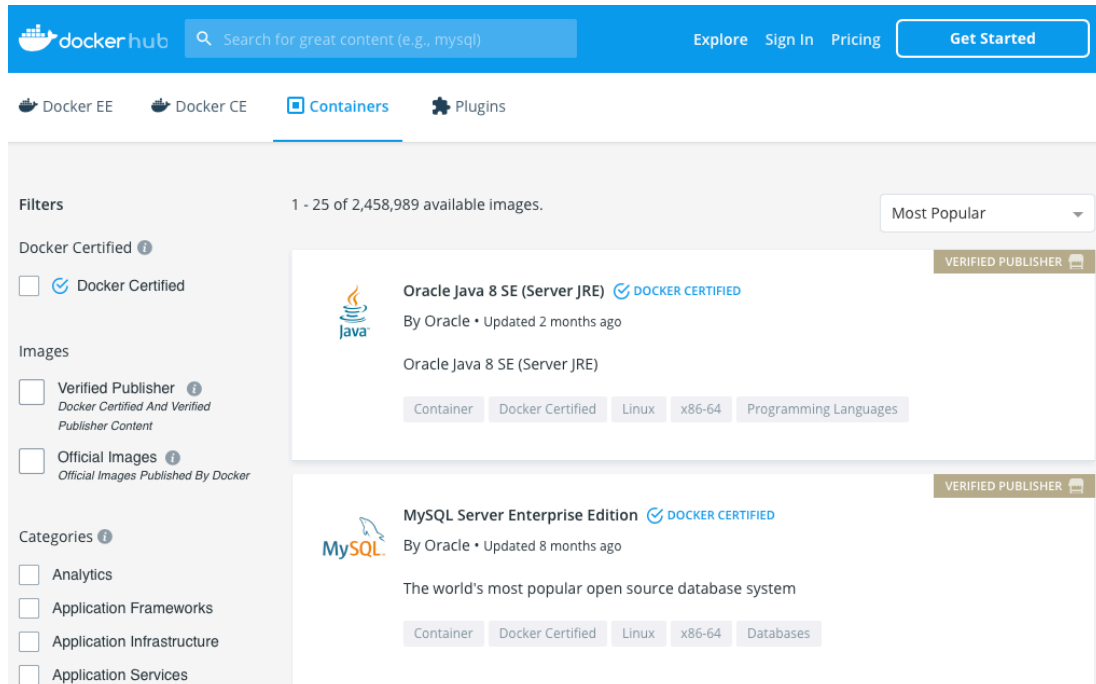


Hình 1.2 Hình thái mạng ảo cần mô phỏng được dựng trên *Netkit Lab Generator*

1.2.1 Các đặc điểm chính của *Kathará*

- Hoạt động trên nền tảng ảo hóa *Docker*.
- Máy thực (host machine) triển khai *Kathará* có thể chạy hệ điều hành Linux, Window hoặc MacOS. Tuy nhiên, môi trường phù hợp nhất và được khuyến khích triển khai *Kathará* đó là Linux.
- Các máy ảo (virtual machine) được xem là 1 nút (node) trong mô hình mạng được mô phỏng.

- Các máy ảo được tạo ra dưới dạng UML (User Mode Linux) và được quản lý như là các Containerized Applications² trên nền tảng *Docker* của máy thực. Các máy ảo này chiếm rất ít tài nguyên máy thực (CPU, Mem...)
- Các máy ảo UML mặc nhiên được tạo ra từ cùng 1 ảnh (image) có sẵn hoặc từ các ảnh mà người dùng xây dựng lại theo nhu cầu sử dụng. *Docker* cung cấp thư viện ảnh tại *Docker Hub*³ rất phong phú và sẵn dùng. Hình 1.3 là danh sách các ảnh thông dụng mà *Docker Hub* cung cấp:



Hình 1.3 Danh sách các ảnh được cung cấp bởi Docker Hub. Nguồn: Docker Hub

- Các máy ảo UML liên thông, kết nối với nhau dễ dàng do cùng hoạt động trong một môi trường ảo hóa là Docker. Ngoài ra, các máy ảo UML cũng có thể kết nối đến máy thực cục bộ (local machine) hoặc đến các máy thực ở xa (remote machine).
- Tại thời điểm hiện tại, Kathara chỉ cho phép mô phỏng mạng ảo với hình thái trục (bus topology) sử dụng chuẩn Ethernet II.

² <https://www.docker.com/resources/what-container>

³ <https://hub.docker.com/u/kathara/>

1.2.2 Hệ thống tập lệnh trong Kathará

Kathará cung cấp 2 tập lệnh với phần tiếp đầu ngữ (prefix) là: **v-commands** và **l-commands**. 2 tập lệnh này được sử dụng trên màn hình điều khiển (terminal) của máy thực. Trong đó:

- **v-commands** được sử dụng để tương tác với một máy ảo đơn lẻ và tất cả các liên kết mạng tới nó. Tập lệnh này chủ yếu bao gồm các lệnh như: chạy (run), dừng (stop), xóa bỏ (remove), truy vấn (info).
- **l-commands** được sử dụng để tương tác với nhiều hoặc tất cả các máy ảo cùng một lúc trong một mô hình mạng ảo. Tập lệnh này cho phép tự động hóa nhiều thao tác quản lý trên các máy ảo qua hệ thống thư mục chứa các file thiết lập cấu hình (configuration files) mà *Kathará* quy định. Hầu hết các lệnh được cung cấp bởi **v-commands** đều có trong **l-commands**.

Một số lệnh thông dụng trong tập lệnh **v-commands** và **l-commands** được trình bày trong Bảng 1.1

	v-commands	l-commands
start	Khởi động 1 máy ảo. Ví dụ: <code>vstart --eth 0:A pc1</code>	Khởi động một hoặc nhiều máy ảo trong 1 mạng ảo. Ví dụ: <code>lstart</code>
crash	Dừng (forcefully stopped) 1 máy ảo. Máy ảo đó sẽ dừng mà không quan tâm đến các hoạt động bên trong máy ảo đã hoàn thành hay chưa. Tài nguyên cấp phát cho máy ảo vẫn duy trì để tái khởi động. Ví dụ: <code>vcrash pc1</code>	Dừng (forcefully) một hoặc nhiều máy ảo trong mạng. Ví dụ: <code>lcrash pc1</code>
halt	Dừng (gracefully stopped) 1 máy ảo. Máy ảo đó sẽ dừng khi tất cả các hoạt động bên trong đã hoàn thành. Tài nguyên cấp phát cho máy ảo vẫn duy trì để tái khởi động. Ví dụ: <code>vhalt pc1</code>	Dừng (gracefully stopped) một hoặc nhiều máy ảo. Ví dụ: <code>lhalt</code>

<code>clean</code>	Dừng hoạt động của máy ảo và các kết nối liên quan đến máy ảo đó. Tài nguyên cấp phát cho máy ảo bị thu hồi. Ví dụ: <code>vclean pc1</code>	Dừng hoạt động của các máy ảo và các kết nối liên quan đến các máy ảo đó. Tài nguyên cấp phát cho các máy ảo bị thu hồi. Ví dụ: <code>lclean</code>
<code>config</code>	Giúp người dùng tạo ra các liên kết mới trên máy ảo (post creation) sau khi máy ảo đã khởi động. Ví dụ: <code>vconfig --eth 2:C pc1</code>	Không có
<code>list</code>	Giúp người dùng quan sát được các thông số hữu ích liên quan đến hoạt động của máy ảo trên máy thực (% CPU, %Mem...) Ví dụ: <code>vlist pc1</code>	Không có
<code>wipe</code>	Không có	Lệnh sử dụng tương tự như lệnh <code>lclean</code> . Ví dụ: <code>lwipe</code>
<code>restart</code>	Không có	Khởi động lại một hoặc nhiều máy ảo trong mạng. Ví dụ: <code>lrestart pc1</code>

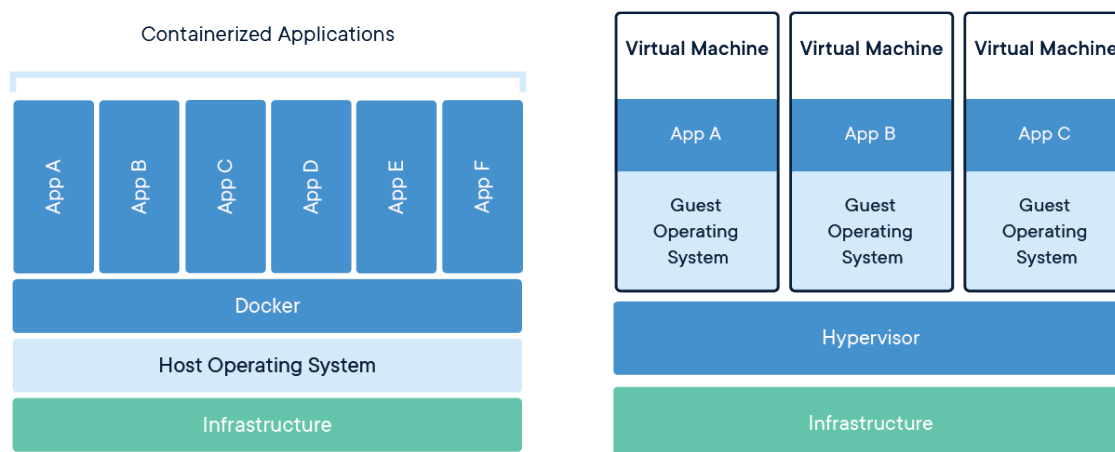
Bảng 1.1 Nhóm lệnh *v-commands* và *l-commands* của *Kathará*

1.2.3 Cài đặt Docker và *Kathará*

Trong các đánh giá về hiệu năng hoạt động của các phần mềm mô phỏng mạng ảo, *Kathará* đạt được những đánh giá tốt theo nhiều tiêu chí khác nhau (về hiệu suất sử dụng CPU, tiêu hao bộ nhớ, thời gian khởi động và thực thi...). Chính vì vậy, trong tài liệu hướng dẫn thực hành Mạng máy tính CT112, chúng tôi lựa chọn giới thiệu và sử dụng *Kathará* mô phỏng các hệ thống mạng minh họa cho kiến thức lý thuyết về Mạng máy tính đã giảng dạy.

- Đối với hệ điều hành Ubuntu 18.04, các thao tác cài đặt phải được thực hiện bằng tài khoản người dùng có thẩm quyền.
- Các lệnh cài đặt được thực hiện trên terminal của Ubuntu 18.04.

- *Docker, Python3* cũng như các thiết lập cần thiết cho môi trường thực hành Mạng máy tính CT112 đã được hoàn tất tại các phòng thực hành chuyên sâu về Mạng máy tính của khoa CNTT&TT, Đại học Cần Thơ. Vì vậy sinh viên không cần phải thực hiện lại các công việc dưới đây.
 - Tài khoản người dùng mà sinh viên được phép thao tác trên Ubuntu 18.04 là *student*. Lưu ý: Đây không phải là tài khoản có quyền cài đặt các gói phần mềm lên hệ thống.
- 1) Cài đặt *Docker* trên Ubuntu 18.04 (hoặc mới hơn) có thể tham khảo trong tài liệu này⁴. Hình 1.4 dưới đây thể hiện rõ sự khác biệt trong cách thức quản lý các ứng dụng kiểu Containerized Application trên Docker với ứng dụng trên máy ảo truyền thống.



Hình 1.4 Ứng dụng triển khai trên Docker và máy ảo truyền thống. Nguồn: Docker

Một số lưu ý đối với việc cài đặt và thực thi *Docker*:

- Khởi động *Docker* là bắt buộc để chạy *Kathará*. Để khởi động *Docker* bằng tay (manually) có thể dùng lệnh `systemctl` hoặc `service`. Ngoài ra *Docker* có thể khởi động tự động cùng hệ điều hành (automatically)⁵.
- Mặc nhiên chỉ có người dùng gốc (root) mới có quyền thao tác trên *Docker*. Tuy nhiên, sử dụng tài khoản root là không được khuyến khích mà thay vào đó sẽ sử dụng các tài khoản đã được thêm vào nhóm người dùng có quyền

⁴ <https://docs.docker.com/install/linux/docker-ce/ubuntu/>

⁵ <https://docs.docker.com/config/daemon/systemd/>

thao tác trên *Docker*. Lệnh thêm 1 người dùng (user) vào nhóm có quyền thao tác trên *Docker* như sau:

```
sudo usermod -aG docker your-user
```

- Có thể kiểm tra thử hoạt động của *Docker* bằng cách khởi động một Containerized Application từ ảnh là *hello-world*. Ảnh này không có sẵn trên máy cài đặt *Docker*, chính vì vậy *Docker* sẽ kết nối đến *Docker Hub* để tải ảnh về và sử dụng để khởi động Containerized Application. Lệnh sử dụng để thực thi công việc này là:

```
docker run hello-world
```

```
[vanlongs-MBP:~ vanlong$ docker run hello-world

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
$ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
https://hub.docker.com/

For more examples and ideas, visit:
https://docs.docker.com/get-started/

vanlongs-MBP:~ vanlong$ █
```

Hình 1.5 Kết quả thực thi lệnh *docker run hello-world*

- Nếu người dùng có nhu cầu thực thi một ứng dụng hay dịch vụ nào đó, chẳng hạn hệ điều hành Ubuntu trên một Containerized Application, thì có thể tìm kiếm phiên bản phù hợp của ứng dụng và thực thi giống như hướng dẫn trên.

```
docker search ubuntu
```

- 2) Cài đặt *Python 3* (trường hợp sử dụng phiên bản thấp hơn 18.04) bằng lệnh:

```
sudo apt-install python3
```

- 3) Đặt *Python 3* thay thế cho *Python 2.7* có sẵn bằng lệnh sau:

```
alias python=python3
```


- 4) Trường hợp *Python* bị thiếu thư viện *ipaddress* thì có thể tiến hành cài đặt trực tiếp như sau:

```
pip install ipaddress
```

- 5) Cài đặt *Wrapper* để đảm bảo tín an ninh (security) và riêng tư (privilege) trong vận hành giữa các mạng ảo trên *Docker*. Ngoài ra *Wrapper* còn rất cần thiết trong việc mở rộng sử dụng *Kathará* giữa các máy tính thực chia sẻ tài nguyên với nhau (shared computer).

```
sudo apt install build-essential
```

- 6) Cài đặt *Xterm terminal* để cho phép truy cập vào các máy ảo và điều khiển trực tiếp chúng. *Xterm* thường đã cài đặt sẵn trên các phiên bản của Ubuntu.

```
sudo apt install xterm
```

- 7) Truy cập: <https://github.com/KatharaFramework/Kathara/releases>. Chọn tải về Source code (zip). Giải nén file .zip đã tải về, được thư mục *Kathara-0.36.1*

- 8) Sao chép thư mục *Kathara-0.36.1* vào trong thư mục */home* của máy thực. Đặt lại tên cho thư mục vừa sao chép được là *KatharaMMT/*

- 9) Mở file *bashrc* của người dùng hiện hành trên máy thực bằng lệnh: *nano ~/.bashrc*. Di chuyển đến cuối file *.bashrc* và thêm vào nội dung sau đây để đặt biến môi trường cho *Kathará*:

```
export NETKIT_HOME=~/.KatharaMMT/bin
export PATH=$PATH:$NETKIT_HOME
export MANPATH=$NETKIT_HOME/man
```

- 10) Thực thi việc cài đặt *Kathará* lên *Docker* bằng lệnh sau đây:

```
$NETKIT_HOME/install
```

- 11) Sau khi hoàn tất quá trình cài đặt, *Docker* trên máy cục bộ đã có thông tin của ảnh mà *Kathará* dùng để tạo ra các máy ảo. Kiểm tra thông tin ảnh đang được quản lý bởi *Docker* bằng lệnh:

```
docker images
```

```
[vanlongs-MBP:~ vanlong$ docker images
REPOSITORY          TAG             IMAGE ID        CREATED         SIZE
kathara/testnet      latest         3e5f9a202c1d   4 months ago   824MB
alpine               latest         5cb3aa00f899   4 months ago   5.53MB
kathara/netkit_base  latest         90f86699ac62   6 months ago   824MB
hello-world          latest         fce289e99eb9   6 months ago   1.84kB
vanlongs-MBP:~ vanlong$
```

Hình 1.6 Kết quả thực thi lệnh *docker images*

12) Khởi động 1 máy ảo để thử nghiệm việc cài đặt. Nếu máy ảo được khởi động thành công nghĩa là quá trình cài đặt đã hoàn tất chính xác. Lệnh để khởi động 1 máy ảo (đặt tên là pc1) như sau:

```
vstart --eth 0:A pc1
```

```
[vanlongs-MBP:~ vanlong$ vstart --eth 0:A pc1
===== Starting Lab =====
[Password:
d9b644b0ed377e1e9422391f552253fb43f22a2e06dfc824a17be5a843045f2a
Applying brctl patch to link d9b644b0ed37
89a0d30e0964ad7afe759fbe3cfbbbaa0b6aeced1039b63cf7e034ce1cec4fb3
net.ipv4.conf.all.rp_filter = 0
net.ipv4.conf.default.rp_filter = 0
net.ipv4.conf.eth0.rp_filter = 0
net.ipv4.conf.lo.rp_filter = 0
netkit_501_pc1
vanlongs-MBP:~ vanlong$ Warning: locale not supported by Xlib, locale set to C
vanlongs-MBP:~ vanlong$ █
```

Hình 1.7 Kết quả thực thi lệnh `vstart --eth 0:A pc1`

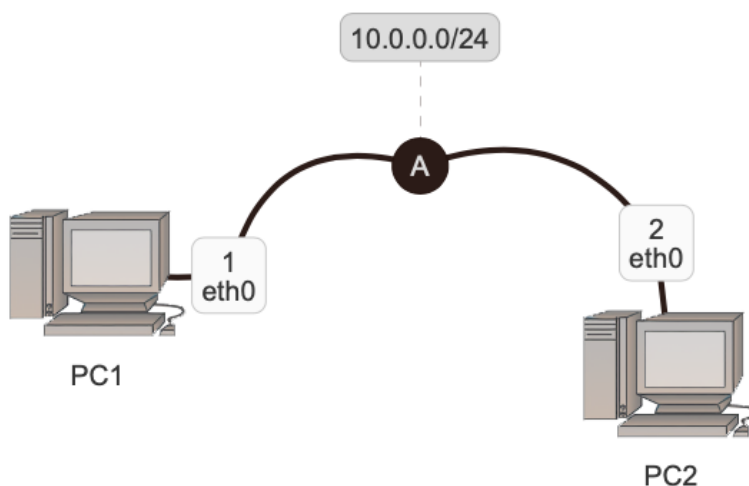
13) Trên máy thực, tắt và xóa máy ảo bằng lệnh:

```
vclean pc1
```

1.3 BÀI TẬP THỰC HÀNH

Các thao tác trong phần hướng dẫn thực hành được diễn giải theo từng bước một. Căn cứ vào công việc cụ thể trong nội dung thực hành, sinh viên có thể thực hiện 1 trong 2 cách sau: 1) sử dụng lệnh qua terminal hoặc 2) sử dụng thao tác trên GUI của Linux; hoặc kết hợp cả 2 cách. Các phần hướng dẫn thực hành được trình bày dưới đây sẽ sử dụng cách 1).

1.3.1 Bài tập 1



Hình 1.8 Mô hình mạng sử dụng cho Bài tập 1 và Bài tập 2

Mục tiêu: Xây dựng một mạng LAN đơn giản theo Phương pháp 1. Các bước thực hiện *Bài tập 1* được trình bày chi tiết như sau:

- 1) Quan sát mô hình mạng cần xây dựng và nhận diện các thiết bị, giao diện với các địa chỉ IP được gán. Mô hình mạng này được thực hiện bởi công cụ *Netkit Lab Gen*.
- 2) Tạo thư mục *BaiTap1* nằm trong workspace của sinh viên. Trên terminal của máy thực, di chuyển đến thư mục *BaiTap1* bằng lệnh:

```
cd /home/student/your_workspace/BaiTap1
```

- 3) Khởi tạo máy ảo pc1 bằng lệnh:

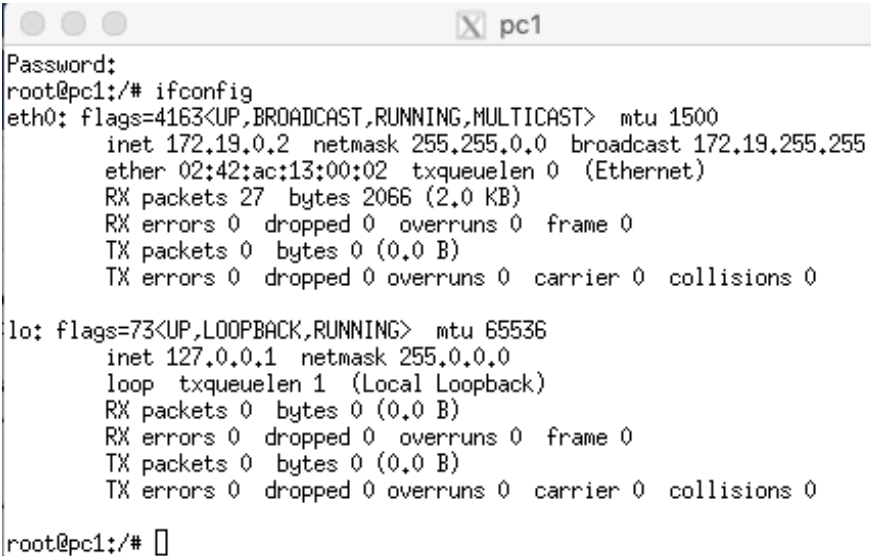
```
vstart --eth 0:A pc1.
```

Lệnh này tạo ra 1 máy ảo tên là pc1 với 1 giao diện mạng eth0. Giao diện eth0 giúp kết nối pc1 vào 1 nhánh mạng LAN (LAN segment) có tên là A.

Lặp lại công việc tương tự để tạo ra tiếp 1 máy ảo nữa là pc2

```
vstart --eth 0:A pc2
```

- 4) Trên giao diện *Xterm* của máy ảo pc1 hoặc pc2, gõ lệnh *ifconfig* để kiểm tra cấu hình mạng như hình 1.9



```
pc1
Password:
root@pc1:/# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.19.0.2 netmask 255.255.0.0 broadcast 172.19.255.255
    ether 02:42:ac:13:00:02 txqueuelen 0 (Ethernet)
    RX packets 27 bytes 2066 (2.0 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    loop txqueuelen 1 (Local Loopback)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

root@pc1:/#
```

Hình 1.9 Kết quả thực thi lệnh ifconfig trên máy ảo pc1

Câu hỏi:

- Có những giao diện mạng nào đã được tạo ra trong máy ảo?
- Địa chỉ IP của các giao diện mạng đó là bao nhiêu? Có đúng với địa chỉ IP cần gán mà Bài tập đã miêu tả hay không?

- 5) Đặt lại địa chỉ IP cho giao diện eth0 của pc1 bằng lệnh sau (sử dụng trên máy ảo pc1):

```
ifconfig eth0 10.0.0.1 netmask 255.255.255.0 broadcast 10.0.0.255
```

Gợi ý: sử dụng một lệnh đơn giản hơn (vẫn trên máy ảo pc1)

```
ifconfig eth0 10.0.0.1/24 up
```

- 6) Thực hiện đặt lại địa chỉ IP cho giao diện eth0 của pc2 tương tự như đã thực hiện với pc1. Kiểm tra lại một lần nữa bằng lệnh `ifconfig` trên 2 máy để đảm bảo việc gán địa chỉ IP mới đã thành công.
- 7) Trên pc1 thực hiện gửi gói tin ICMP đến pc2 bằng lệnh:

```
ping 10.0.0.2
```

Câu hỏi: Kết quả hiển thị trên màn hình của pc1 là gì?

- 8) Lần lượt thực hiện các thao tác sau:
- Sử dụng lệnh `traceroute` để kiểm tra thông tin đường đi của gói tin từ pc1 đến pc2. Kết quả hiển thị cho biết gì?
 - Sử dụng lệnh `route` để hiển thị thông tin bảng vạch đường của pc1 hoặc pc2 trong mạng LAN A. Kết quả hiển thị cho biết gì?
- 9) Trên máy thực, sử dụng lần lượt

```
vclean pc1
```

```
vclean pc2
```

để hủy 2 máy ảo vừa tạo và kết thúc *Bài tập 1*.

1.3.2 Bài tập 2

Mục tiêu: Xây dựng một mạng LAN đơn giản theo Phương pháp 2. Các bước thực hiện *Bài tập 2* được trình bày chi tiết như sau:

- 1) Sử dụng lại mô hình mạng đã cho ở *Bài tập 1*.
- 2) Tạo thư mục *BaiTap2* trong workspace của sinh viên. Thư mục sẽ này chứa các thư mục con và các file cấu hình (`.startup`, `lab.conf`) theo cấu trúc quy định của *Kathará*. Trên máy thực, di chuyển đến thư mục *BaiTap2* bằng

```
cd /home/student/your_workspace/BaiTap2
```

Cấu trúc thư mục *BaiTap2* được miêu tả như hình 1.10:



Hình 1.10 Các thư mục con và tệp tin trong thư mục BaiTap2

- File `lab.conf` chứa miêu tả về hình thái (topology) của một mạng ảo
- Thư mục `pc1` và `pc2` đại diện cho 2 máy ảo của mạng muốn đưa vào hoạt động. Máy ảo sẽ có tên đại diện là tên của thư mục, chẳng hạn: máy ảo số 1 có tên là `pc1`, máy ảo số 2 có tên là `pc2`. Lưu ý: Thư mục đại diện cho máy ảo không được chứa ký tự viết hoa (theo quy tắc khởi tạo Containerized Application của *Docker*).
- File `pc1.startup` và `pc2.startup` (gọi chung là các file `.startup`) là nơi chứa các cấu hình muốn áp dụng cho một máy ảo (`pc1` hoặc `pc2`) ngay khi máy ảo được khởi động cùng mạng ảo. Lưu ý: để đảm bảo rằng 1 máy ảo sẽ nhận được cấu hình mong muốn nhờ vào nội dung trong file `.startup` thì file `.startup` đó phải có tên trùng với tên của thư mục đại diện cho máy ảo. Ví dụ: máy ảo tên là `pc1` được thể hiện qua thư mục tên `pc1` và file tên `pc1.startup`

Để thực hiện tạo thư mục, file cũng như soạn thảo nội dung cho file, ngoài chế độ đồ họa GUI trên máy thực, người dùng có thể lựa chọn chế độ dòng lệnh để thực hiện

- Tạo mới thư mục: `mkdir <new_folder>`
- Tạo mới file: `touch <new_file>`
- Soạn thảo nội dung file: `nano <file_need_to_edit>`

3) Trên file `lab.conf`, soạn thảo nội dung mô tả hình thái mạng theo thiết kế

```
pc1[0]=A
pc2[0]=A
```

4) Trên file `pc1.startup`, cấu hình của `eth0` được miêu tả như sau:

```
ifconfig eth0 10.0.0.1/24 up
```

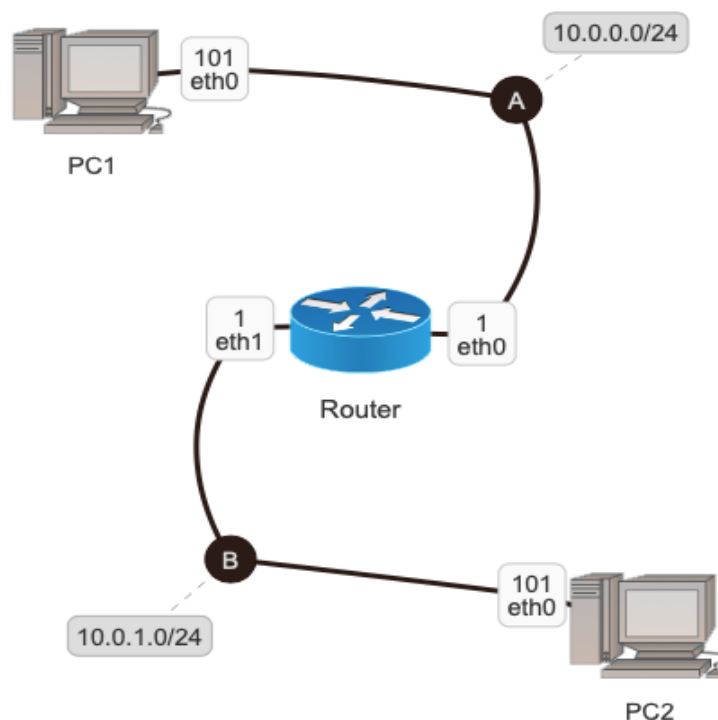
Trên file `pc2.startup`, cấu hình của `eth0` được miêu tả như sau:

```
ifconfig eth0 10.0.0.2/24 up
```

5) Trên máy thực, tại thư mục *BaiTap2* sử dụng lệnh `lstart` để khởi động mạng ảo *BaiTap2* đã tạo.

- Có thể sử dụng `1start` để khởi động từng máy ảo riêng lẻ trong trường hợp muốn kiểm tra từng máy ảo. Ví dụ: `1start pc1` hoặc `1start pc2`.
 - Việc khởi động từng máy ảo riêng lẻ trong một mạng ảo sẽ giúp sinh viên kiểm tra được tính đúng đắn của từng nút trong mô hình mạng trước khi khởi động toàn bộ mô hình mạng đó.
- 6) Trên `pc1`, lần lượt dùng các lệnh `ping`, `traceroute` và `route` để kiểm tra tính liên thông giữa `pc1` và `pc2` trong nhánh mạng LAN A giống như 7) và 8) của *Bài tập 1*
- 7) Trên máy thực, sử dụng lệnh `1wipe` để hủy 2 máy ảo vừa tạo. Kết thúc *Bài tập 2*.

1.3.3 Bài tập 3



Hình 1.11 Mô hình mạng sử dụng trong Bài tập 3

Mục tiêu: Xây dựng 2 nhánh mạng thuộc cùng một LAN được kết nối bởi 1 router. Các bước thực hiện *Bài tập 3* được trình bày chi tiết như sau:

- 1) Quan sát mô hình mạng cần xây dựng và nhận diện các thiết bị, giao diện với các địa chỉ IP được gán trên các máy ảo.
- 2) Tạo thư mục *BaiTap3* trong workspace của sinh viên. Thư mục sẽ này chứa các thư mục con và các file cấu hình (`.startup`, `lab.conf`) theo cấu trúc quy định của *Kathará*.

Trên máy thực, di chuyển đến thư mục *BaiTap3* bằng lệnh:

```
cd /home/student/your_workspace/BaiTap3
```

Cấu trúc thư mục *BaiTap3* được như hình 1.12:



Hình 1.12 Các thư mục con và tệp tin trong thư mục *BaiTap3*

- 3) Trên file *lab.conf*, soạn thảo nội dung mô tả hình thái mạng theo thiết kế

```
pc1[0]=A
pc2[0]=B
router[0]=A
router[1]=B
```

- 4) Trên file *pc1.startup*, chứa nội dung được miêu tả như sau

```
ifconfig eth0 10.0.0.101/24 up
route add default gw 10.0.0.1
```

- Lệnh *route add default gw*: thêm thông tin vạch đường mặc nhiên (default route) vào bảng vạch đường của một thiết bị.

- ✓ Tham số *gw* đại diện cho Gateway là hướng mà thiết bị sẽ gửi gói tin đến để các gói tin có thể đi ra bên ngoài mạng.
- ✓ Thông tin vạch đường mặc nhiên sẽ được sử dụng khi thiết bị (Router, PC) không tìm thấy bất kỳ thông tin vạch đường cụ thể nào đến đích trong bảng vạch đường.

- 5) Trên file *pc2.startup* chứa nội dung được miêu tả như sau

```
ifconfig eth0 10.0.1.101/24 up
route add default gw 10.0.1.1
```

- 6) Trên file *router.startup*, cấu hình của eth0 và eth1 được miêu tả như sau

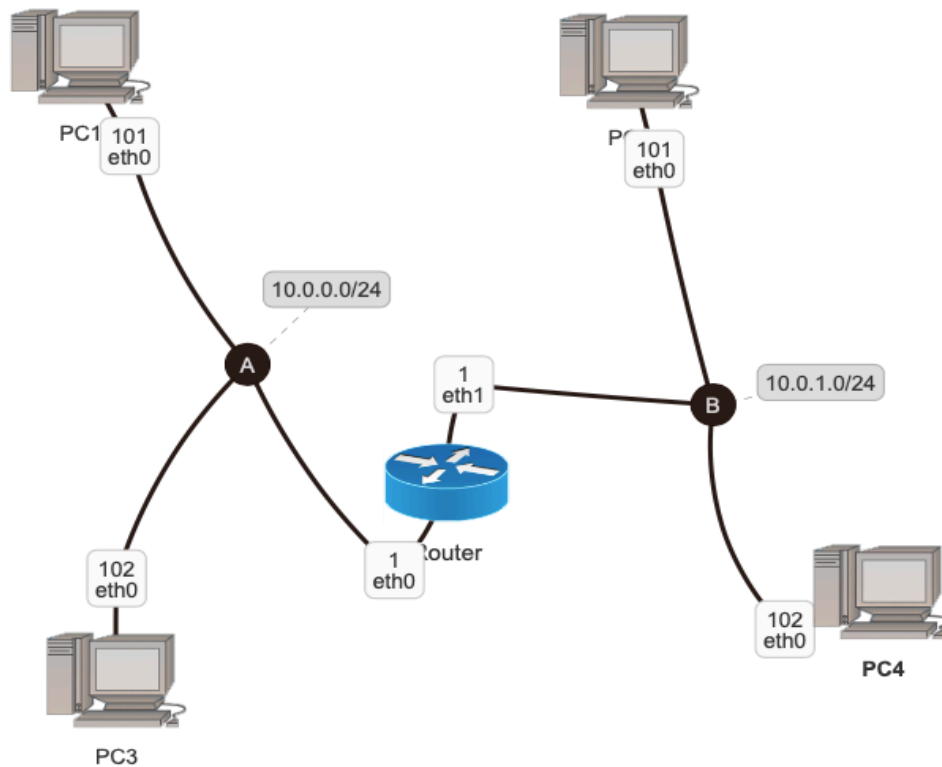
```
ifconfig eth0 10.0.0.1/24 up
ifconfig eth1 10.0.1.1/24 up
```

- 7) Trên máy thực, tại thư mục *BaiTap3* sử dụng lệnh *lstart* để khởi động mạng ảo *BaiTap3* đã tạo.

- 8) Trên pc1 lần lượt dùng các lệnh *ping*, *traceroute* và *route* để kiểm tra tính liên thông tới *router* và *pc2*.

- 9) Trên máy thực, sử dụng lệnh *lwiipe* để hủy mạng ảo *BaiTap3* vừa tạo. Kết thúc *Bài tập 3*.

1.3.4 Bài tập 4



Hình 1.13 Mô hình mạng sử dụng trong Bài tập 4

Mục tiêu: Xây dựng 2 nhánh mạng khác LAN, mỗi nhánh mạng có hai máy tính. Hai nhánh mạng khác LAN này được kết nối với nhau thông qua 1 router. Các bước thực hiện *Bài tập 4* được trình bày chi tiết như sau:

- 7) Quan sát mô hình mạng cần xây dựng và nhận diện các thiết bị, giao diện với các địa chỉ IP được gán.
- 8) Tạo thư mục *BaiTap4* trong workspace của sinh viên. Thư mục sẽ này chứa các thư mục con và các file cấu hình (*.startup*, *lab.conf*) theo cấu trúc quy định của *Kathará*.

Trên máy thực, di chuyển đến thư mục *BaiTap4* bằng lệnh

```
cd /home/student/your_workspace/BaiTap4
```

Gợi ý: Về cơ bản, nội dung *Bài tập 4* là phần mở rộng của *Bài tập 3*, vì vậy sinh viên có thể sao chép nội dung thư mục *BaiTap3* sang cho *BaiTap4* và bổ sung thêm phần khai báo, cấu hình cho máy pc3 và pc4

- 9) Thực hiện giống như hướng dẫn trong *Bài tập 3* đã làm.