Лабораторная работа 1. Практика по Linux

Выполнила студент гр. 5142704/30801 Порфирьева Е. В.

Шаг 1: создание виртуальных машин в PWD

С помощью возможностей https://labs.play-with-docker.com/ создадим три виртуальные машины, нажав ADD NEW INSTANCE.

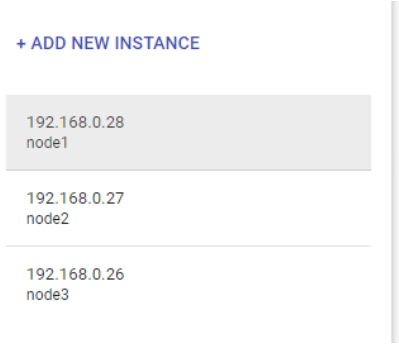

Рисунок 1 - Три виртуальные машины в Play With Docker

Шаг 2: генерация SSH-ключа

Для подключения к виртуальным машинам по ssh необходимо сгенерировать ключ.

Подключение к виртуальным машинам осуществляется с помощью команд, указанных для каждой ВМ в PWD:

1. ssh ip172-18-0-40-cnibn7q91nsg00b8iieg@direct.labs.play-with-docker.com
2. ssh ip172-18-0-50-cnibn7q91nsg00b8iieg@direct.labs.play-with-docker.com
3. ssh ip172-18-0-56-cnibn7q91nsg00b8iieg@direct.labs.play-with-docker.com

Шаг 3: Настройка маршрутов

Согласно схеме, приведенной на рисунке 2, необходимо настроить сеть:

- у Linux A и Linux C - по 1 адаптеру
- Linux B - по 2 адаптера

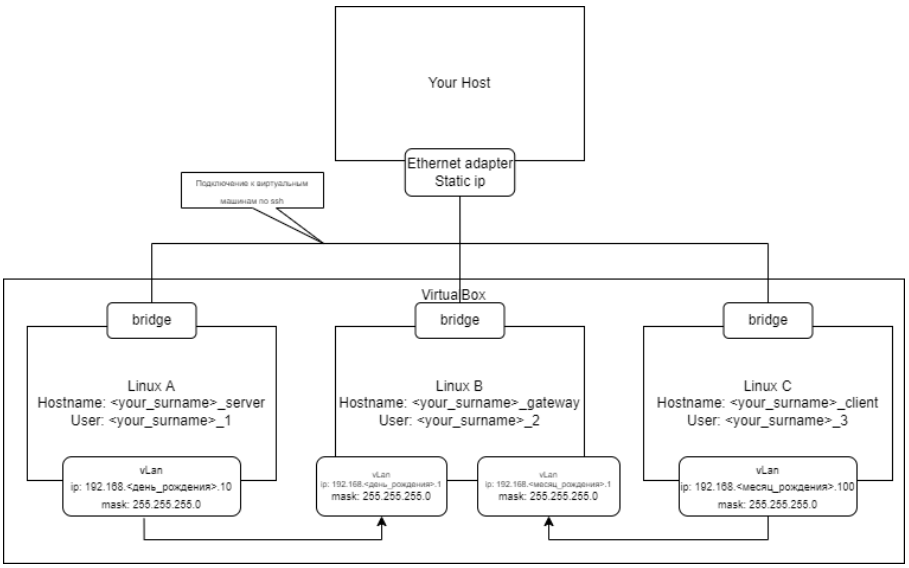

Рисунок 2 - Развертка трех ВМ по заданной схеме

По заданию каждому адаптеру необходимо назначить следующие ip-адреса:
Linux A: 192.168.4.10 / 24 (рис. 3)

Linux C: 192.168.5.100 / 24 (рис. 4)

Linux B1: 192.168.5.1 / 24 (рис. 5)

Linux B2: 192.168.4.1 / 24 (рис. 6)

```
# This is a sandbox environment. Using personal credentials   #
# is HIGHLY! discouraged. Any consequences of doing so are    #
# completely the user's responsibilites.                      #
#                                                             #
# The PWD team.                                               #
###############################################################
[node1] (local) root@192.168.0.28 ~
$ ip link add macvlan1 link eth0 type macvlan mode bridge
[node1] (local) root@192.168.0.28 ~
$ ip address add dev macvlan1 192.168.4.10/24
[node1] (local) root@192.168.0.28 ~
$ ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN qlen 1
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
2: docker0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN
    link/ether 02:42:c8:5c:b0:a1 brd ff:ff:ff:ff:ff:ff
    inet 172.17.0.1/16 brd 172.17.255.255 scope global docker0
       valid_lft forever preferred_lft forever
3: macvlan1@eth0: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN qlen 1
    link/ether 9a:d9:18:b5:2c:2b brd ff:ff:ff:ff:ff:ff
    inet 192.168.4.10/24 scope global macvlan1
       valid_lft forever preferred_lft forever
151846: eth0@if151847: <BROADCAST,MULTICAST,UP,LOWER_UP,M-DOWN> mtu 1500 qdisc noqueue state UP
    link/ether c6:69:52:13:fd:ab brd ff:ff:ff:ff:ff:ff
    inet 192.168.0.28/23 scope global eth0
       valid_lft forever preferred_lft forever
151850: eth1@if151851: <BROADCAST,MULTICAST,UP,LOWER_UP,M-DOWN> mtu 1500 qdisc noqueue state UP
    link/ether 02:42:ac:12:00:12 brd ff:ff:ff:ff:ff:ff
    inet 172.18.0.18/16 scope global eth1
       valid_lft forever preferred_lft forever
[node1] (local) root@192.168.0.28 ~
$
ip link set macvlan1 up
[node1] (local) root@192.168.0.28 ~
$
```

Рисунок 3 - Создание адаптера и установка IP - адреса для Linux A

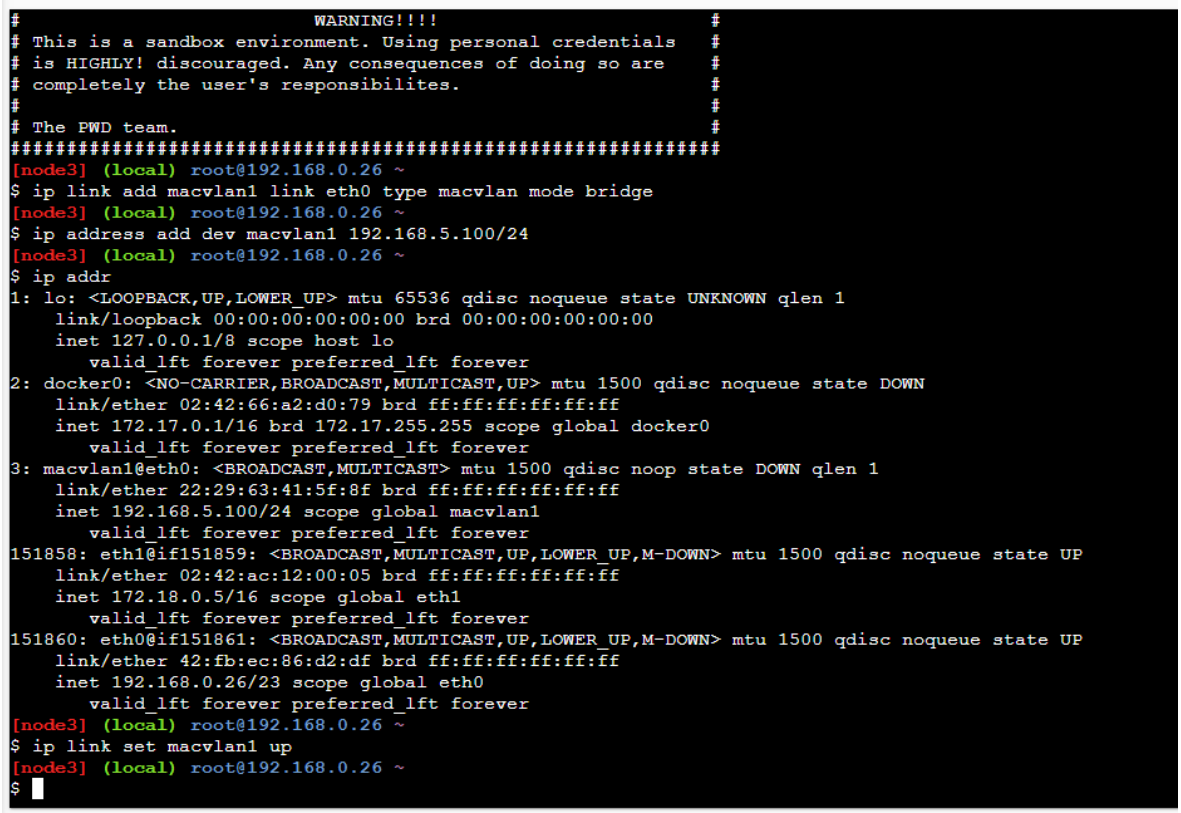

Рисунок 4 - Создание адаптера и установка IP - адреса для Linux C

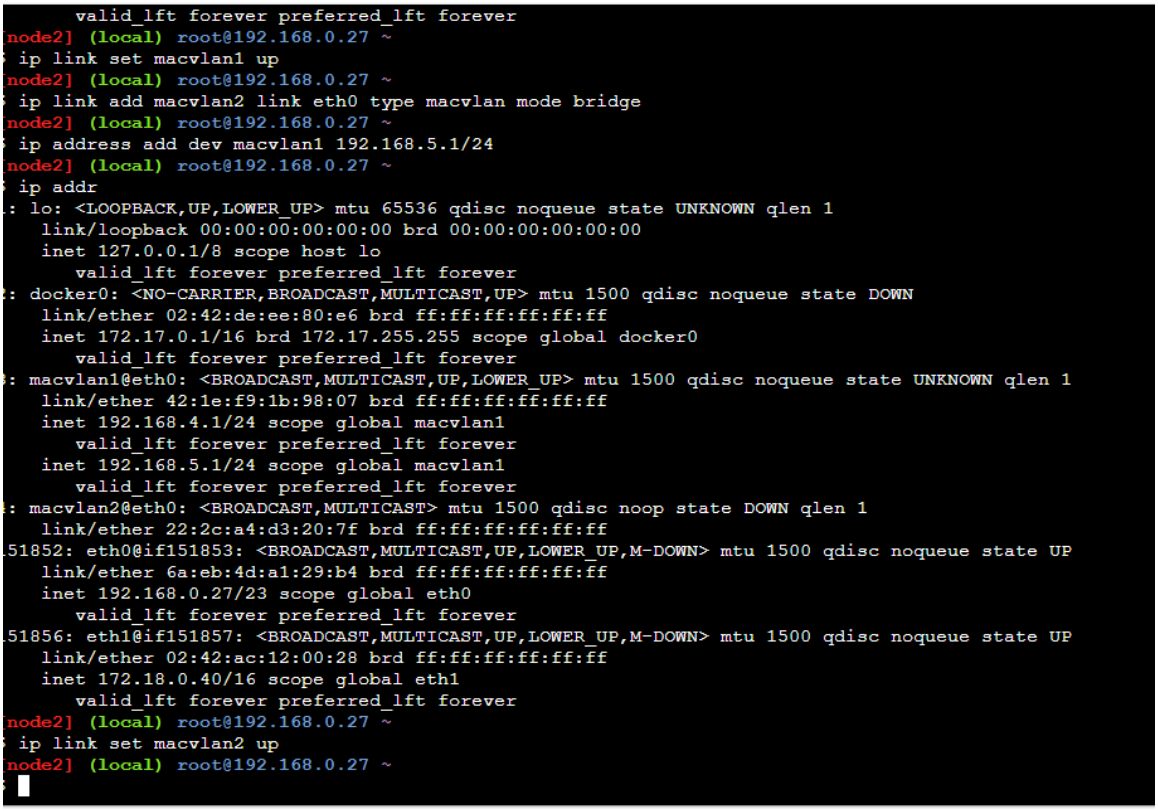

Рисунок 5 - Создание 1-го адаптера и установка IP - адреса для Linux B

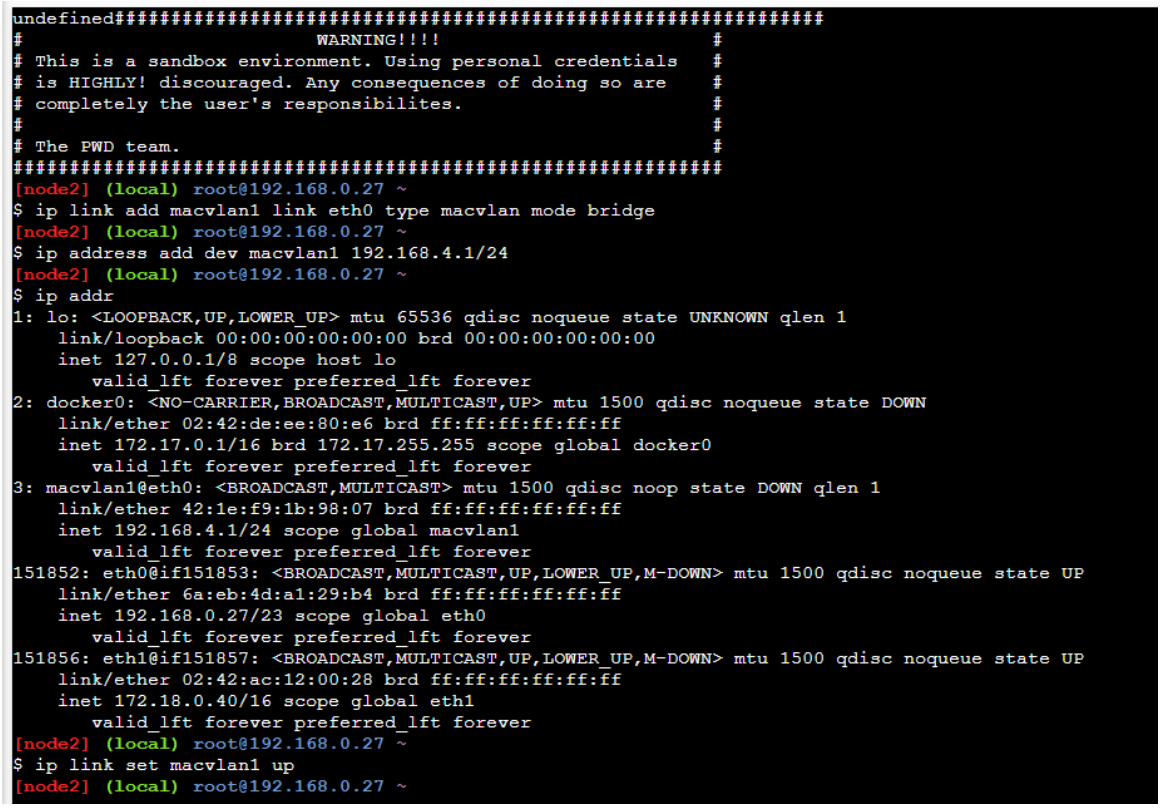

Рисунок 6 - Создание 2-го адаптера и установка IP - адреса для Linux B

После настройки сети на ВМ необходимо прописать маршруты у клиентов A и C к их подсетям через машину B при помощи команды: ip route add <subnet A vm>/<mask> via <gateway ip B vm>.

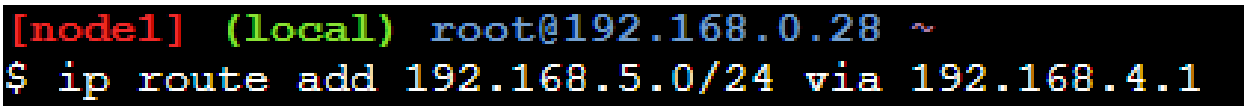

Рисунок 7 - Настройка маршрута передачи от A к C через B

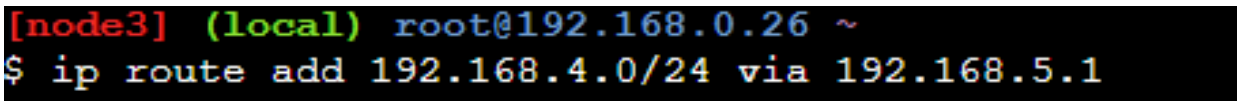

Рисунок 8 - Настройка маршрута передачи от C к A через B

Шаг 4: Настройка файрвола

На Linux B необходимо настроить файрвол таким образом, чтобы Linux B пропускал только http и только через порт 5000.

Это можно сделать при помощи tcpdump.

apk add tcpdump

tcpdump -i any -s 0 'tcp port http' -w /tmp/http.cap and 'port 5000'

```
[node2] (local) root@192.168.0.27 ~
$ apk add tcpdump
fetch https://dl-cdn.alpinelinux.org/alpine/v3.18/main/x86_64/APKINDEX.tar.gz
fetch https://dl-cdn.alpinelinux.org/alpine/v3.18/community/x86_64/APKINDEX.tar.gz
(1/2) Installing libpcap (1.10.4-r1)
(2/2) Installing tcpdump (4.99.4-r1)
Executing busybox-1.36.1-r2.trigger
OK: 469 MiB in 164 packages
[node2] (local) root@192.168.0.27 ~
$ tcpdump -i any -s 0 'tcp port http' -w /tmp/http.cap and 'port 5000'
tcpdump: data link type LINUX_SLL2
tcpdump: listening on any, link-type LINUX_SLL2 (Linux cooked v2), snapshot length 262144 bytes
```

Рисунок 9 – Настройка файрвола

Шаг 5: Организация клиент-серверного взаимодействия между A и C

После настройки маршрутов необходимо развернуть сервер на тачке A. Для этого нужно использовать библиотеку Flask

```
[node1] (local) root@192.168.0.28 ~
$ pip install flask
Collecting flask
  Downloading flask-3.0.3-py3-none-any.whl (101 kB)
     ---------------------------------------- 101.7/101.7 kB 10.0 MB/s eta 0:00:00
Collecting Werkzeug>=3.0.0 (from flask)
  Downloading werkzeug-3.0.4-py3-none-any.whl (227 kB)
     ---------------------------------------- 227.6/227.6 kB 21.7 MB/s eta 0:00:00
Collecting Jinja2>=3.1.2 (from flask)
  Downloading jinja2-3.1.4-py3-none-any.whl (133 kB)
     ---------------------------------------- 133.3/133.3 kB 29.9 MB/s eta 0:00:00
Collecting itsdangerous>=2.1.2 (from flask)
  Downloading itsdangerous-2.2.0-py3-none-any.whl (16 kB)
Collecting click>=8.1.3 (from flask)
  Downloading click-8.1.7-py3-none-any.whl (97 kB)
     ---------------------------------------- 97.9/97.9 kB 26.7 MB/s eta 0:00:00
Collecting blinker>=1.6.2 (from flask)
  Downloading blinker-1.8.2-py3-none-any.whl (9.5 kB)
Collecting MarkupSafe>=2.0 (from Jinja2>=3.1.2->flask)
  Downloading MarkupSafe-2.1.5-cp311-cp311-musllinux_1_1_x86_64.whl (33 kB)
Installing collected packages: MarkupSafe, itsdangerous, click, blinker, Werkzeug, Jinja2, flask
Successfully installed Jinja2-3.1.4 MarkupSafe-2.1.5 Werkzeug-3.0.4 blinker-1.8.2 click-8.1.7 flask-3.0.3 itsdangerous-2.2.0
```

Рисунок 10 - Установка Flask

Далее создаем файл app.py со следующим содержимым:

```
$ touch app.py
[node1] (local) root@192.168.0.28 ~
$ cat << EOF >app.py
> from flask import Flask, request

app = Flask(__name__)

data = {"username":"", "password":""}

@app.route("/")
def get_():
    return data

@app.route("/",methods = ['POST'])
def post():
    data_json=request.get_json()
    if data_json is None:
        return 'Invalid JSON data', 400

    data['username']=data_json['username']
    data['password']=data_json['password']
    print(f"Data received {data_json}")

    return [data['username'], data['password']]

@app.route("/", methods =['PUT'])
def put():
    str = request.args.get('password')
    print(f"New password received {str}")
    data['password'] = str
    return [data['username'], data['password']]

app.run(host='0.0.0.0', port=5000)

EOF
[node1] (local) root@192.168.0.28 ~
$
```

```
EOF
[node1] (local) root@192.168.0.28 ~
$ echo -e "Run the server\n"
Run the server

[node1] (local) root@192.168.0.28 ~
$ python app.py
 * Serving Flask app 'app'
 * Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on all addresses (0.0.0.0)
 * Running on http://127.0.0.1:5000
 * Running on http://172.18.0.18:5000
Press CTRL+C to quit
```

Рисунок 12 - Сервер на машине A

С помощью команды curl на машине C было послано 3 запроса на машину A в http:

```
[node3] (local) root@192.168.0.26 ~
$ curl "http://192.168.4.10:5000/"
{"password":"","username":""}
```

Рисунок - Отправка GET-запроса

```
[node3] (local) root@192.168.0.26 ~
$ curl -X POST -H "Content-Type: application/json" -d' {"username":"Lena","password":"xyz"}' http://192.168.4.10:5000
["Lena","xyz"]
[node3] (local) root@192.168.0.26 ~
```

Рисунок - Отправка POST-запроса

```
[node3] (local) root@192.168.0.26 ~
$ curl -X PUT http://192.168.4.10:5000?password=ber453
["Lena","ber453"]
[node3] (local) root@192.168.0.26 ~
```

Рисунок - Отправка PUT-запроса

```
192.168.5.100 - - [03/Oct/2024 10:36:53] "POST / HTTP/1.1" 200 -
Data received {'username': 'Lena', 'password': 'xyz'}
192.168.5.100 - - [03/Oct/2024 10:37:21] "POST / HTTP/1.1" 200 -
New password received ber453
192.168.5.100 - - [03/Oct/2024 10:38:39] "PUT /?password=ber453 HTTP/1.1" 200 -
```

Рисунок 14 - Запросы, пришедшие на сервер A