Министерство науки и высшего образования Российской Федерации
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Факультет цифровых трансформаций

Дисциплина:

«**Телекоммуникационные системы и технологии**»

ПРАКТИЧЕСКАЯ РАБОТА №1

«Консольные утилиты настройки сетевых компонентов Linux»

Выполнили:

Гаджиев С. И., Васильков Д. А., Лавренов Д. А. М3304

Pulpy

(подпись)

Проверила:

Дяченко Екатерина Олеговна

(отметка о выполнении)

(подпись)

Цель работы:

Получить практические навыки по конфигурированию сетевых интерфейсов (на примере протокола IPv4) в операционных системах Linux, ознакомится с утилитами командной строки, освоить современные сетевые менеджеры Linux.

Необходимо:

- Установленная на компьютере среда виртуализации ORACLE Virtual Box
- Две виртуальные машины Linux с Linux CentOS и Linux Debian

Порядок выполнения работы:

Для выполнения работы рекомендуется виртуальные машины Linux CentOS 7 и Linux Debian 11. Рекомендуется гипервизор VirtulaBox 7.

В работе используются две конфигурации:

1) Статическая адресация IP=10.100.0.2, MASK=255.255.255.0, GATE=10.100.0.1, DNS = 8.8.8.8
2) Динамическая – все параметры получаются автоматически с dhcp сервера.

Часть 1. Linux (CentOS или Debian 11)

1. Запустить виртуальную машину и авторизоваться в системе под администраторской учётной записью.
2. Создать скрипт, который позволяет пользователю (!):
    a. Узнать модель сетевой карты, канальную скорость, режим работы (Full или Half Duplex). Наличие физического подключения (линка), MAC адрес
    b. Получить информацию о текущей конфигурации IPv4 (ip, mask, gate, dns). c. Настроить сетевой интерфейс по сценарию #1
    d. Настроить сетевой интерфейс по сценарию #2.
    e. закрыть скрипт.
3. Скрипт не должен вносить изменения в системны конфигурационные файлы. Нельзя использовать менеджеры сети.

Часть 2. Работа с виртуальными интерфейсами CentOS через Network Manager

1. На виртуальной машине CentOS настройте эмуляцию сетевого интерфейса в Virtual Box на режим Внутренняя сеть.
2. Используя консольную утилиту Network Manager nmcli (!):
    a. Настройте сетевой интерфейс по сценарию №1.
    b. создайте виртуальный сетевой интерфейс с именем br0 и IP = IP=10.100.0.3. c. активируйте виртуальный интерфейс.
3. С помощью утилиты ping проверьте связь между реальным и виртуальным интерфейсом.
4. Определите MAC адрес виртуального интерфейса (!).

Часть 3. Работа с реальными интерфейсами Linux Debian 11 через netpaln

1. На виртуальной машине Debian 11 заранее убедитесь, что установлен пакет netpaln
2. Настройте эмуляцию сетевого интерфейса в Virtual Box на режим Внутренняя сеть (это должна быть таже сеть, что и в части 2).
3. Отредактируйте YAML файл конфигурации netplan так чтобы сетевой интерфейс имел два адреса 10.100.0.4 и 10.100.0.5, маска 255.255.255.0, а gateway 10.100.0.3 (!).
4. С помощью утилиты ping проверьте связь между адресами 10.100.0.2-10.100.0.5.
5. Выведите таблицу arp кэша. Найдите в ней записи обо всех адресах 10.100.0.2-10.100.0.5 (!).

Часть 4. Настройка объединения реальных сетевых интерфейсов в Linux

В этой части вы исследуете возможности модуля ядра Linux – bonding. Этот модуль позволяет объединять сетевые интерфейсы в группы, в целях обеспечения отказоустойчивости или увеличения пропускной способности. Выполнять задания части можно на любой из использованных виртуальных машин. Рекомендуется использовать менеджеры сети (nmcli или netplan), но можно прямо настроить конфигурационные файлы интерфейсов и (или) воспользоваться утилитой ifenslave.

1. Выключите выбранную вами виртуальную машину, сделайте снапшот. В конфигурацию машины добавьте сетевой интерфейс. Оби интерфейса переведите в режим «Сеть NAT» (так же возможно использовать режим «Сетевой мост», только в этом случае нужно убедиться, что в реальной сети работает автоматическое конфигурирование IP по dhcp).
2. Убедитесь, что загружен модуль bounding. Проверить работу модуля можно командой lsmod. Если модуль не загружен, запустите его командой modprobe (если автоматической загрузки модуля нет, то учтите, что после перезагрузки модуль надо загружать заново вручную).
3. С помощью выбранного вами средства настройте объединение сетевых корт в одну группу, так чтобы (!):
    a. карты чередовались при передаче пакетов,
    b. виртуальный интерфейс группы назывался bond007
    c. интерфейс bond007 получал конфигурацию IPv4 автоматически.
4. Активируйте интерфейс группы. Выведите на консоль информацию о IP и MAC адресах всех интерфейсов.
5. Выведите на консоль файл /proc/net/bonding/bond007 , ознакомитесь с его содержанием (этот динамический файл позволяет отлаживать bonding) (!).
6. На второй консоли выведите раза содержимое файла /pcroc/net/dev, ознакомьтесь с его содержимым.
7. Напишите скрипт, который выводит на консоль (!): a. Отметку текущего времени b. Имя интерфейса, значение Receive-packets, значение Transmitpackets.
8. На перовой консоли запустите ping -I bond007 8.8.8.8
9. На второй консоли с помощью вашего скрипта соберите трижды данные.

Проанализируйте результаты (!).

Выполнение:

(Все выполнение в артефактах)

Артефакты:

1. Скрипт Часть 1 п. 2

```bash
#!/bin/bash

if [ "$(id -u)" -ne "0" ]; then
  echo "Скрипт должен быть запущен от имени суперпользователя."
  exit 1
fi

get_network_info() {
  echo "\n"
  echo ------------------

  echo "Информация о сетевой карте:"

  echo -n "Модель сетевой карты: "
  ethtool -i eth0 | grep "driver:" | awk '{print $2}'

  echo -n "Канальная скорость и режим работы: "
  ethtool eth0 | awk '/Speed:/ {print $2}'
  ethtool eth0 | awk '/Duplex:/ {print $2}'

  echo -n "Наличие физического подключения (линка): "
  ethtool eth0 | awk '/Link detected:/ {print $3}'

  echo -n "MAC адрес: "
  ip link show eth0 | awk '/link\/ether/ {print $2}'

  echo ------------------
  echo "\n"
}
```

```
get_ipv4_info() {
  echo "Информация о текущей конфигурации IPv4:"

  IP_ADDR=$(ip addr show eth0 | awk '/inet / {print $2}' | cut -d/ -f1)
  MASK=$(ip addr show eth0 | awk '/inet / {print $2}' | cut -d/ -f2)
  GATEWAY=$(ip route show | awk '/default/ {print $3}')
  DNS=$(grep "nameserver" /etc/resolv.conf | awk '{print $2}' | paste -sd ", ")

  echo "\n"
  echo ------------------
  echo -n "IP адрес: "
  echo $IP_ADDR
  echo -n "Маска подсети: "
  echo $MASK
  echo -n "Шлюз: "
  echo $GATEWAY
  echo -n "DNS: "
  echo $DNS
  echo ------------------
  echo "\n"
}

```

```
clear_ip_addresses() {
  echo "Удаление старых IP-адресов с интерфейса eth0"
  IP_ADDRESSES=$(ip addr show eth0 | awk '/inet / {print $2}')
  for IP in $IP_ADDRESSES; do
    ip addr del $IP dev eth0
  done
}

configure_static() {
  echo "Настройка сетевого интерфейса по сценарию #1"
  clear_ip_addresses
  ip addr add 10.100.0.2/24 dev eth0
  ip route add default via 10.100.0.1
  echo "nameserver 8.8.8.8" > /etc/resolv.conf
}

configure_dhcp() {
  echo "Настройка сетевого интерфейса по сценарию #2"
  clear_ip_addresses
  dhclient eth0
}

```

```
75
76  while true; do
77      echo "1. Узнать информацию о сетевой карте"
78      echo "2. Получить информацию о текущей конфигурации IPv4"
79      echo "3. Настроить сетевой интерфейс по сценарию #1 (статическая адресация)"
80      echo "4. Настроить сетевой интерфейс по сценарию #2 (динамическая адресация)"
81      echo "5. Закрыть скрипт"
82
83      read -p "Введите номер действия: " choice
84
85      case $choice in
86          1) get_network_info ;;
87          2) get_ipv4_info ;;
88          3) configure_static ;;
89          4) configure_dhcp ;;
90          5) exit ;;
91          *) echo "Неправильный выбор. Попробуйте еще раз." ;;
92      esac
93  done
```

6. Команды и конфигурационные файлы (если таковые использовались) Части 4 п.5

```yaml
network:
  version: 2
  renderer: networkd
  ethernets:
    enp0s3:
      dhcp4: no
    enp0s8:
      dhcp4: no
  bonds:
    bond007:
      interfaces: [enp0s3, enp0s8]
      parameters:
        mode: balance-rr
      dhcp4: yes
```

```
bazis@Seti:~/Desktop/ITMO-Telecommunication-systems-and-technologies-labs-5-sem$ cat /proc/net/bonding/bond007
Ethernet Channel Bonding Driver: v6.1.0-25-amd64

Bonding Mode: load balancing (round-robin)
MII Status: up
MII Polling Interval (ms): 0
Up Delay (ms): 0
Down Delay (ms): 0
Peer Notification Delay (ms): 0

Slave Interface: enp0s3
MII Status: up
Speed: 1000 Mbps
Duplex: full
Link Failure Count: 0
Permanent HW addr: 08:00:27:79:81:19
Slave queue ID: 0

Slave Interface: enp0s8
MII Status: up
Speed: 1000 Mbps
Duplex: full
Link Failure Count: 0
Permanent HW addr: 08:00:27:28:42:cf
Slave queue ID: 0
```

7. Консольный вывод Части 4 п.5 8. Скрипт Части 4 п.7

```bash
#!/bin/bash

for i in {1..3}; do
        echo "current time: $(date)"
        echo "Interface bond007:"
        grep bond007 /proc/net/dev
        sleep 3
done
```

```
bazis@Seti:~/Desktop/ITMO-Telecommunication-systems-and-technologies-labs-5-sem/Lab1/Task4$ ./statistics.sh
current time: Tue Sep 17 11:44:22 PM MSK 2024
Interface bond007:
bond007:   46564     302    0    0    0     0          0         52    74820     422    0   50    0     0       0
     0
```

<u>Вопросы и задания:</u>

1. Как с помощью команды ip:

    a. назначить новый IPv4 адрес?

        Ответ: для назначения нового IPv4:

        sudo ip addr add <IP_ADDRESS>/<SUBNET_MASK> dev <INTERFACE>

        Пояснение:

- ***ip addr add***: команда для добавления нового адреса.
- ***<IP_ADDRESS>/<SUBNET_MASK>***: указываете IP-адрес и маску подсети. Например, 192.168.1.10/24.
- ***dev <INTERFACE>***: указывает интерфейс, к которому будет привязан новый адрес (например, eth0).

    Пример:
    sudo ip addr add 192.168.1.10/24 dev eth0

Эта команда добавляет адрес 192.168.1.10 с маской 255.255.255.0 на интерфейс eth0.

b. назначить новый MAC адрес?

Ответ: чтобы изменить MAC-адрес:

sudo ip link set dev <INTERFACE> address <NEW_MAC_ADDRESS>

Пояснение:

- ***ip link set***: команда для настройки интерфейса.
- ***dev <INTERFACE>***: указывает интерфейс, для которого меняется MAC-адрес.
- ***address <NEW_MAC_ADDRESS>:*** задает новый MAC-адрес.

Пример:

sudo ip link set dev eth0 address 00:11:22:33:44:55
Эта команда изменяет MAC-адрес интерфейса eth0 на 00:11:22:33:44:55.

c. назначить новый gateway?

Ответ: для установки нового шлюза:

sudo ip route add default via <GATEWAY_IP>

Пояснение:

- ***ip route add***: команда для добавления нового маршрута.
- ***default***: указывает, что это маршрут по умолчанию.
- ***via <GATEWAY_IP>:*** указывает IP-адрес шлюза.

Пример:

sudo ip route add default via 192.168.1.1
Эта команда устанавливает шлюз по умолчанию на 192.168.1.1.

d. вывести информацию arp кэше?

Ответ: чтобы просмотреть ARP-кэш:

ip neigh

Пояснение:

- ***ip neigh***: команда для отображения ARP-кэша, который показывает соответствия между IP-адресами и MAC-адресами.

e. очистить arp кэш?

Ответ: для очистки ARP-кэша:

sudo ip neigh flush all

Пояснение:

- ***ip neigh flush all***: команда для удаления всех записей ARP-кэша. Это может быть полезно, если вы хотите обновить ARP-таблицу..

f. включить интерфейс?

Ответ: чтобы включить интерфейс:

sudo ip link set dev <INTERFACE> up

Пояснение:

- ***ip link set***: команда для настройки интерфейса.
- ***dev <INTERFACE>***: указывает интерфейс, который нужно активировать.
- ***up***: команда для включения интерфейса.

Пример:

sudo ip link set dev eth0 up

Эта команда активирует интерфейс eth0.

g. выключить интерфейс?

Ответ: чтобы выключить интерфейс:

sudo ip link set dev <INTERFACE> down

Пояснение:

- ***down***: команда для деактивации интерфейса.

Пример:

sudo ip link set dev eth0 down

Эта команда отключает интерфейс eth0.

2. Как с помощью nmcli назначить на интерфейс статический IP адрес, маску и настроить default gateway?

Ответ:

Чтобы назначить статический IP-адрес, маску и шлюз с помощью nmcli, выполните следующие команды:

Пояснение:

- ***nmcli con mod***: команда для изменения конфигурации подключения.
- ***<CONNECTION_NAME>***: имя вашего сетевого соединения (можно получить с помощью nmcli con show).
- ***ipv4.addresses <IP_ADDRESS>/<SUBNET_MASK>***: задает статический IP-адрес и маску.
- ***ipv4.gateway <GATEWAY_IP>***: указывает IP-адрес шлюза.
- ***ipv4.method manual***: устанавливает метод получения IP-адреса как статический.
- ***nmcli con up <CONNECTION_NAME>:*** применяет изменения и активирует соединение.

Пример:

3. Как с помощью netplan назначить на интерфейс статический IP адрес, маску и настроить default gateway?

Ответ:

Для настройки статического IP через netplan, вам нужно отредактировать конфигурационный файл, который обычно расположен в /etc/netplan/. Вот пример конфигурации:

Пояснение:

- ***network***: ключевое слово, указывающее на начало конфигурации сети.
- ***version*****: 2**: версия спецификации конфигурации Netplan.
- ***ethernets***: определяет параметры Ethernet-интерфейсов.
- ***<INTERFACE>***: имя вашего интерфейса (например, eth0).
- ***dhcp4: no***: отключает DHCP для IPv4.
- ***addresses***: задает статический IP-адрес и маску.
- ***gateway4***: задает шлюз по умолчанию.
- ***nameservers***: указывает DNS-серверы.

Пример:
После изменения файла выполните команду: ***sudo netplan apply***

4. Какие режимы bonding стандартно существую в Linux? Опишите их назначение, возможности по отказоустойчивости и необходимость поддержки со стороны оборудования.

Ответ:

Bonding в Linux позволяет объединять несколько сетевых интерфейсов в один логический интерфейс для повышения отказоустойчивости и пропускной способности.

Стандартные режимы bonding:

1. **mode 0 (balance-rr)**:
    - **Назначение**: равномерное распределение нагрузки по всем интерфейсам.
    - **Отказоустойчивость**: отсутствует; если один интерфейс выходит из строя, трафик не может быть перенаправлен.
    - **Требования**: не требует поддержки со стороны оборудования.
2. **mode 1 (active-backup)**:
    - **Назначение**: один интерфейс активен, остальные в резерве.
    - **Отказоустойчивость**: обеспечивает отказоустойчивость; если активный интерфейс выходит из строя, происходит автоматическое переключение на резервный интерфейс.
    - **Требования**: не требует поддержки со стороны оборудования.
3. **mode 2 (balance-xor)**:
    - **Назначение**: использование XOR для определения интерфейса, который будет передавать пакеты.
    - **Отказоустойчивость**: отсутствует, но нагрузка распределяется.
    - **Требования**: желательно иметь поддержку со стороны коммутаторов.
4. **mode 3 (broadcast)**:
    - **Назначение**: все данные передаются на все интерфейсы.
    - **Отказоустойчивость**: обеспечивает отказоустойчивость, но неэффективен с точки зрения производительности.
    - **Требования**: не требует поддержки со стороны оборудования.
5. **mode 4 (802.3ad)**:
    - **Назначение**: агрегация каналов с использованием LACP (Link Aggregation Control Protocol).
    - **Отказоустойчивость**: высокая; трафик может быть распределён между несколькими активными интерфейсами.
    - **Требования**: требуется поддержка со стороны оборудования.
6. **mode 5 (balance-tlb)**:
    - **Назначение**: динамическое распределение нагрузки.
    - **Отказоустойчивость**: обеспечивает отказоустойчивость; если активный интерфейс выходит из строя, трафик перенаправляется.
    - **Требования**: не требует поддержки со стороны оборудования.
7. **mode 6 (balance-alb)**:
    - **Назначение**: сочетает режимы TLB и ALB, поддерживает входящие и исходящие соединения.
    - **Отказоустойчивость**: высокая; эффективно управляет пропускной способностью.
    - **Требования**: не требует поддержки со стороны оборудования.

5. Какие существуют и чем отличаются режимы работы адаптера (duplex)?

Ответ:

Режимы работы адаптера делятся на два основных типа:

1. **Half Duplex**:
    - **Определение**: данные могут передаваться в одном направлении в любой момент времени.
    - **Пример**: радиосвязь, где одна сторона говорит, а другая слушает.
    - **Недостатки**: может вызвать конфликты, если обе стороны пытаются передавать данные одновременно.
2. **Full Duplex**:
    - **Определение**: данные могут передаваться в обоих направлениях одновременно.
    - **Пример**: телефонная связь, где обе стороны могут говорить и слышать друг друга одновременно.
    - **Преимущества**: значительно повышает производительность и эффективность передачи данных, исключая конфликты.

Отличия:

- Half duplex ограничивает скорость передачи данных, так как в любой момент времени может работать только один канал. Full duplex позволяет максимизировать использование канала связи.

6. Какой, по-вашему, практический смысл в возможности назначения нескольких IP адресов на один интерфейс?

Ответ:

Назначение нескольких IP-адресов на один интерфейс может иметь несколько практических применений:

- **Разделение трафика**: Один интерфейс может обслуживать несколько приложений или сервисов, каждый из которых использует свой IP-адрес. Это может быть полезно для обеспечения изоляции трафика и управления нагрузкой.
- **Поддержка виртуальных сетей**: Виртуальные локальные сети (VLAN) могут использовать один физический интерфейс, назначая на него несколько IP-адресов, что позволяет логически разделять трафик без необходимости в дополнительных физических интерфейсах.
- **Обеспечение отказоустойчивости**: В случае выхода из строя одного IP-адреса можно быстро переключиться на другой. Это может быть полезно для высоконагруженных серверов или служб, требующих непрерывной доступности.
- **Переход на новую сеть**: Во время миграции сети можно назначить новый IP-адрес наряду со старым, чтобы обеспечить плавный переход и минимизировать время простоя.

7. Какой, по-вашему, практический смысл в возможности создания виртуальных интерфейсов?

Ответ:

Создание виртуальных интерфейсов имеет множество практических применений:

- **Изоляция сервисов**: Разные приложения могут использовать различные виртуальные интерфейсы для своих нужд. Это позволяет управлять трафиком, применяя различные настройки безопасности и QoS (Quality of Service) для каждого виртуального интерфейса.
- **Упрощение администрирования**: Виртуальные интерфейсы позволяют системным администраторам создавать отдельные логические интерфейсы для разных сетевых конфигураций, упрощая управление и мониторинг сетевых соединений.
- **Эффективное использование ресурсов**: Можно создать несколько виртуальных интерфейсов на одном физическом интерфейсе, что позволяет максимально использовать доступную пропускную способность без необходимости в дополнительных физических интерфейсах.
- **Поддержка многослойной архитектуры**: Виртуальные интерфейсы могут быть полезны в сложных сетевых архитектурах, таких как VPN, где необходимо иметь разные интерфейсы для разных сетей и протоколов.
- **Эксперименты и тестирование**: Виртуальные интерфейсы позволяют создавать тестовые среды без необходимости в дополнительном оборудовании, что делает тестирование более удобным и экономически эффективным.

Ход выполнения работы:

Часть 1. Linux (CentOS или Debian 11)

1. Запустить виртуальную машину и авторизоваться в системе под администраторской учетной записью.
2. Создать скрипт, который позволяет пользователю (!):
    a. Узнать модель сетевой карты, канальную скорость, режим работы (Full или Half Duplex). Наличие физического подключения (линка), MAC адрес
    b. Получить информацию о текущей конфигурации IPv4 (ip, mask, gate, dns).
    c. Настроить сетевой интерфейс по сценарию #1
    d. Настроить сетевой интерфейс по сценарию #2.

e. закрыть скрипт.

3. Скрипт не должен вносить изменения в системные конфигурационные файлы. Нельзя использовать менеджеры сети.

+—

[Ч1.2]. Создали скрипты info.sh и menu.sh, которые:

info.sh позволяет:

1. Узнать модель сетевой карты, канальную скорость, режим работы, наличие физического подключения, MAC адрес.
2. Узнать текущую конфигурацию IPv4
3. Настроить сетевой интерфейс по сценарию №1 (статическая адресация)
4. Настроить сетевой интерфейс по сценарию №2 (динамическая адресация)

menu.sh:

Создан для упрощения работы (смены конфигураций и получения данных о текущей)

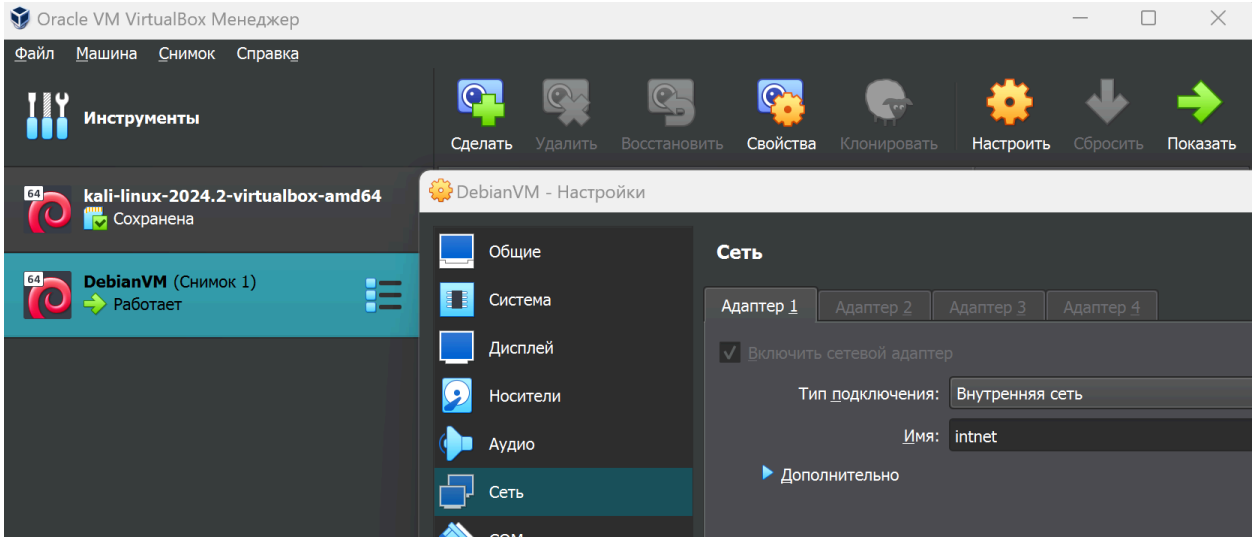
#—

Часть 2. Работа с виртуальными интерфейсами Linux CentOS через Network Manager

1. На виртуальной машине CentOS настройте эмуляцию сетевого интерфейса в Virtual Box на режим Внутренняя сеть.
2. Используя консольную утилиту Network Manager nmcli (!):
   a. Настройте сетевой интерфейс по сценарию №1.
   b. создайте виртуальный сетевой интерфейс с именем br0 и IP=10.100.0.3.
   c. активируйте виртуальный интерфейс.
3. С помощью утилиты ping проверьте связь между реальным и виртуальным интерфейсом.
4. Определите MAC адрес виртуального интерфейса (!).

+---

[Ч2.1].



[Ч2.2.a].

```
root@DebianVM:/home/vboxuser/labs/lab1# ./menu.sh
Options:
1. Find network card info
2. Show current IPv4 config
3. Set static config
4. Set dynamic config
5. Exit
Choose option: 3
Static configuration installed.
```

Под капотом:

```
function set_static_ip() {
        interface=$(ip -o -4 route show to default | awk '{print $5}')

        sudo ip addr flush dev $interface
        sudo ip addr add 10.100.0.2/24 dev $interface
        sudo ip route add default via 10.100.0.1
        echo "nameserver 8.8.8.8" | sudo tee /etc/resolv.conf > /dev/null
        echo "Static configuration installed."
}
```

[Ч2.2.b].

nmcli connection add type bridge con-name br0 ifname br0 ipv4.method manual ipv4.addresses 10.100.0.3/24

[Ч2.2.c].
nmcli connection up br0

[Ч2.3].

```
root@DebianVM:/home/vboxuser/labs/lab1/third# ping 10.100.0.3
PING 10.100.0.3 (10.100.0.3) 56(84) bytes of data.
64 bytes from 10.100.0.3: icmp_seq=1 ttl=64 time=0.029 ms
64 bytes from 10.100.0.3: icmp_seq=2 ttl=64 time=0.254 ms
64 bytes from 10.100.0.3: icmp_seq=3 ttl=64 time=0.056 ms
64 bytes from 10.100.0.3: icmp_seq=4 ttl=64 time=0.086 ms
^C
--- 10.100.0.3 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3506ms
rtt min/avg/max/mdev = 0.029/0.106/0.254/0.087 ms
root@DebianVM:/home/vboxuser/labs/lab1/third# ping 10.100.0.2
PING 10.100.0.2 (10.100.0.2) 56(84) bytes of data.
64 bytes from 10.100.0.2: icmp_seq=1 ttl=64 time=0.034 ms
64 bytes from 10.100.0.2: icmp_seq=2 ttl=64 time=0.058 ms
64 bytes from 10.100.0.2: icmp_seq=3 ttl=64 time=0.114 ms
^C
--- 10.100.0.2 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2030ms
rtt min/avg/max/mdev = 0.034/0.068/0.114/0.033 ms
```

[Ч2.4].

```
root@DebianVM:/home/vboxuser/labs/lab1/third# ip link show br0
3: br0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN mo
de DEFAULT group default qlen 1000
    link/ether ea:36:d6:af:91:6c brd ff:ff:ff:ff:ff:ff
```

#---

Часть 3. Работа с реальными интерфейсами Linux Debian 11 через netpaln

1. На виртуальной машине Debian 11 заранее убедитесь, что установлен пакет netpaln
2. Настройте эмуляцию сетевого интерфейса в Virtual Box на режим Внутренняя сеть (это должна быть таже сеть, что и в части 2).

3. Отредактируйте YAML файл конфигурации netplan так чтобы сетевой интерфейс имел два адреса 10.100.0.4 и 10.100.0.5, маска 255.255.255.0, а gateway 10.100.0.3 (!).
4. С помощью утилиты ping проверьте связь между адресами 10.100.0.2-10.100.0.5. 5. Выведите таблицу arp кэша. Найдите в ней записи обо всех адресах 10.100.0.2-10.100.0.5 (!).

+---

[Ч3.3]

#---