



МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
"МИРЭА - Российский технологический университет"

РТУ МИРЭА

Институт информационных технологий (ИТ)
Кафедра инструментального и прикладного программного обеспечения

**ОТЧЕТ
ПО ПРАКТИЧЕСКОЙ РАБОТЕ №2**

по дисциплине

«Технологии виртуализации клиент-серверных приложений»

Выполнил студент группы ИКБО-01-22

Прокопчук Р.О.

Принял преподаватель кафедры ИиППО

Волков М.Ю.

Практические работы выполнены

«__»_____2025 г.

«Зачтено»

«__»_____2025 г.

Москва
2025

Теоретическое введение

OVF (Open Virtualization Format) — открытый стандарт для хранения и распространения виртуальных машин. Данный стандарт описывает открытый, переносимый, расширяемый формат для распространения образов виртуальных машин. OVF не привязан к аппаратной архитектуре или реализации гипервизора.

Пакет OVF состоит из нескольких файлов, расположенных в одном каталоге, один из которых с расширением .ovf. .ovf файл представляет из себя XML-файл, который описывает упакованную виртуальную машину и метаданные (название, аппаратные требования, ссылки на другие файлы в пакете и описания). Весь пакет может быть распространен в формате OVA - TAR архив с OVA пакетом внутри.

Виртуальный сетевой адаптер — это программное обеспечение, которое работает как физический сетевой адаптер в операционной системе хоста (OS) или через приложение, установленное на конечной точке или сервере. Приложения и службы на устройстве или сервере могут получить доступ к виртуальному сетевому адаптеру, когда требуется второй сетевой интерфейс, но физический адаптер недоступен.

Сетевой мост — это компьютерное сетевое устройство, которое создает единую совокупную сеть из нескольких сетей или сетевых сегментов. Эта функция называется virtual bridging.

Клонирование — это создание точной копии виртуальной машины, как с теми же настройками, так и с нужными изменениями. Это может производиться в тестовых целях, когда необходимо провести изменения, но неизвестно, как поведет себя виртуальная машина. Создание копии позволяет определить поведение и избежать простоя сервисов.

Bridged Networking — тип сетевого взаимодействия позволяет привязать сетевой адаптер виртуальной машины к физическому сетевому интерфейсу компьютера, что дает возможность разделять ресурсы сетевой карты между хостовой и виртуальной системой. Виртуальная машина с таким типом

сетевого взаимодействия будет вести себя по отношению к внешней сети хостовой системы как независимый компьютер. Вы можете назначить такой машине собственный IP-адрес в домашней сети или сети организации, либо она получит его от внешнего DHCP-сервера.

Host-Only Networking — тип сетевого взаимодействия оптимален для целей тестирования программного обеспечения, когда вам требуется организовать виртуальную сеть в пределах хоста, а виртуальным машинам не требуется выход во внешнюю сеть. В виртуальной подсети действует DHCP-сервер, подключенный к виртуальному коммутатору VMnet1 и назначающий виртуальным машинам IP-адреса из заданного диапазона (по умолчанию 192.168.179.128 — 192.168.179.254).

NAT Networking — тип сетевого взаимодействия очень похож на Host-Only, за одним исключением: к виртуальному коммутатору VMnet8 подключается устройство трансляции IP-адресов (NAT). К этому коммутатору также подключается DHCP-сервер, раздающий виртуальным машинам адреса из заданного диапазона (по умолчанию 192.168.89.128 — 192.168.89.254) и, непосредственно, сами виртуальные машины. NAT-устройство позволяет осуществлять трансляцию IP-адресов, что позволяет виртуальным машинам инициировать соединения во внешнюю сеть, не предоставляя при этом механизма доступа к виртуальным машинам извне.

Шаблоны виртуальных машин представляют собой предустановленные и готовые к запуску виртуальные системы (чаще всего на базе бесплатных операционных систем), которые содержат в себе гостевую ОС, приложения, установленные в ней, необходимые для решения определенного круга задач, а также документацию, позволяющую понять, как работает шаблон виртуальной машины и какие функции он выполняет, а также предоставляющую всю необходимую информацию для его использования (пароли на вход в гостевую систему, инструкции пользователя, расположение компонентов приложений и т.п. В данный момент на рынке присутствует множество как бесплатных, так и коммерческих виртуальных шаблонов,

доступных для скачивания на сайтах производителей платформ. Условно их можно разделить на несколько категорий по сферам использования:

Администрирование

В эту категорию входят виртуальные шаблоны, обеспечивающие поддержку сетевого взаимодействия в инфраструктуре компании, управление рабочими станциями и серверами, а также различные утилиты для мониторинга сетевой активности.

Сервера приложений

Эта сфера применения виртуальных шаблонов одна из самых широких: предустановленные сервера приложений различных производителей могут быть подходящим образом настроены и оптимизированы, а пользователям остается лишь запустить виртуальную машину загрузить на нее контент.

Коммуникация и управление контентом

В данной области виртуальные шаблоны могут предоставлять различные сервисы систем управления контентом (Content Management System, CMS), систем управления отношений с клиентами (Client Relationship Management, CRM), «движки» для создания различных хранилищ знаний (Wiki) и репозитории.

Серверы баз данных и почтовые серверы

В эту группу входят в основном бесплатные серверы баз данных и почтовые серверы, готовые к использованию внешними приложениями и защищенные средствами безопасности. Они могут распространяться со всеми необходимыми настройками и готовы к использованию в производственной среде. Такая модель использования очень удобна в отношении простоты развертывания, тестирования и гибкости в отношении аппаратного обеспечения.

Безопасность и сетевое взаимодействие

В эту категорию входят виртуальные шаблоны, предоставляющие различные средства по защите сетевых соединений (брандмауэры), виртуальные машины с предустановленными антиспамовыми фильтрами и

антивирусами, которые очень удобно использовать для проверки потенциально опасных приложений.

Операционные системы

Гостевые системы в виртуальных машинах могут являться виртуальными шаблонами, поскольку некоторые системы сложно установить или сконфигурировать. Шаблоны виртуальных машин дарят отличную возможность для обучения работе с различными ОС и их модификациями. Виртуализация сетевых функций дает множество преимуществ, так как позволяет решать широкий спектр задач:

Сокращение затрат на оборудование

Развертывание виртуальных сетевых устройств и служб позволяет значительно снизить количество используемого физического оборудования и затрат на его приобретение.

Автоматизация

Многие административные задачи можно автоматизировать, уменьшить влияние человеческого фактора на возникновение ошибок, минимизировать простои ИТ-инфраструктуры.

Гибкое управление

Инструменты позволяют ускорить инициализацию сетевых ресурсов, повысить скорость реагирования, улучшить показатели доступности и непрерывности.

Безопасность

Удобное и быстрое управление политиками безопасности в отношении сетевых функций, управление защитой данных.

Постановка задачи

В данной практической работе будет настроено сетевое взаимодействие между двумя виртуальными машинами и хостом. Будут использоваться VM с ОС Kali Linux, VM с ОС Debian.

Получить виртуальную машину с образом Kali Linux можно с официального сайта <https://www.kali.org/get-kali/#kali-virtualmachines>, в скачиваемом zip архиве будет находиться виртуальная машина в формате vmdk.

После создания виртуальных машин нужно зайти в верхней панели в Редактор виртуальной сети (Правка -> редактор виртуальной сети).

В редакторе виртуальной сети нужно нажать кнопку “Изменить параметры”, при изменении параметров должна появиться виртуальная сеть с типом Bridged/Мост.

После создания виртуальной сети нужно изменить конфигурацию подключения на мост у обеих VM через подключение непосредственно к физической сети.

После настройки сетевого адаптера на обеих VM необходимо запустить их и получить информации о присвоенном ip адресе VM.

После того как IP у обеих VM стали известны можно провести серию пингов. В данном случае будут VM будет пинговать друг друга, а также одна из VM будет пропингована с хостовой машины.

После того, как сеть будет настроена, вам необходимо поднять сервер на любой из виртуальных машин и проверить подключение к сервису с хостовой и соседней виртуальной машины. Сервисом может выступать любое приложение, которое располагается в сети (Backend/Frontend). В качестве примера может выступать http.server для Python.

В результате отчет должен содержать скриншоты процесса настройки сети виртуальных машин, получения IP-адресов VM, процесса проверки соединения между машинами (ping), процесса проверки подключения к сервису на одной из VM с другой VM и хостовой машины.

Ход работы

Созданная виртуальная машина Kali Linux представлена на рисунке 1.

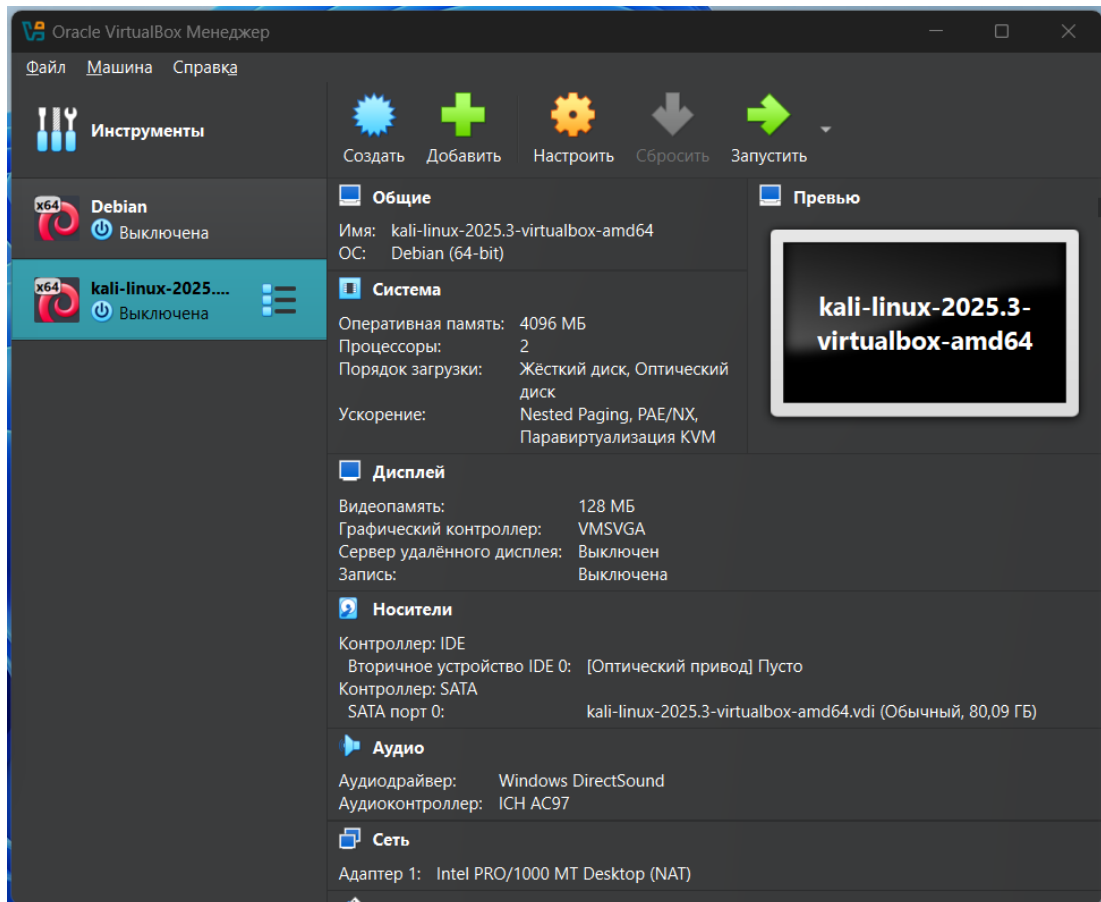


Рисунок 1 – ВМ на Kali Linux

Сетевые настройки обеих ВМ представлены на рисунках 2-3.

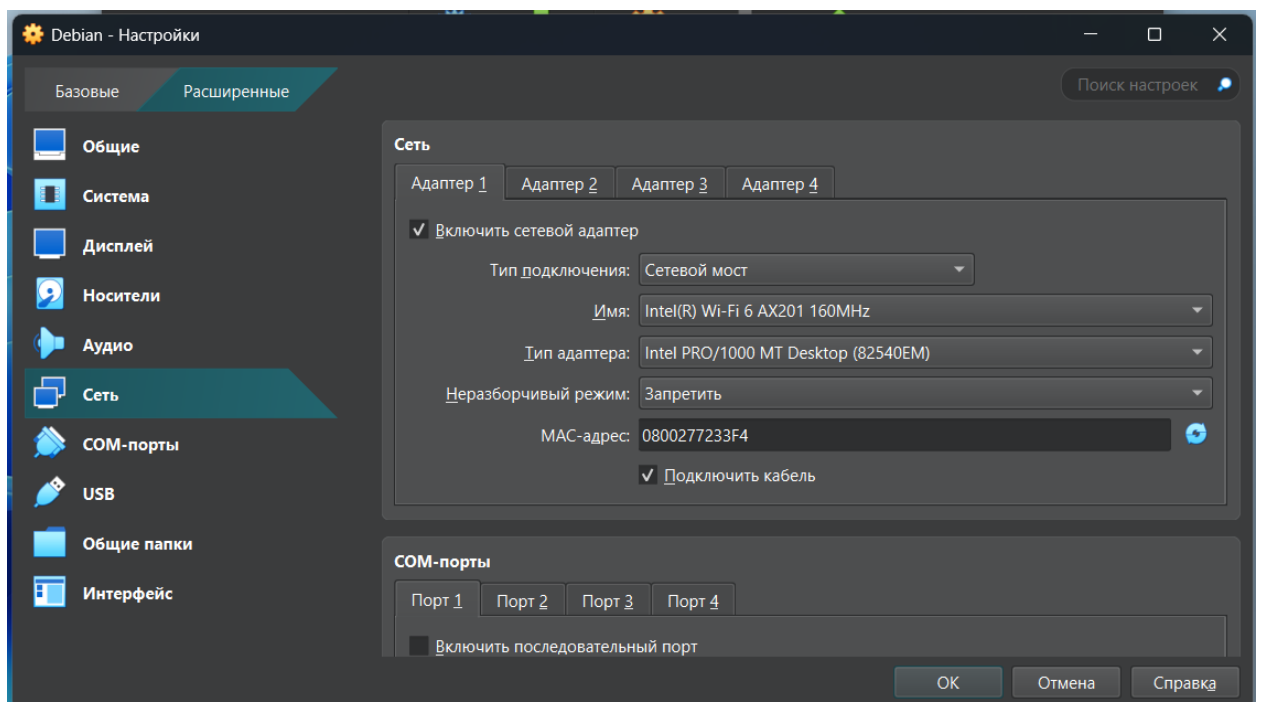


Рисунок 2 – Сетевые настройки Debian

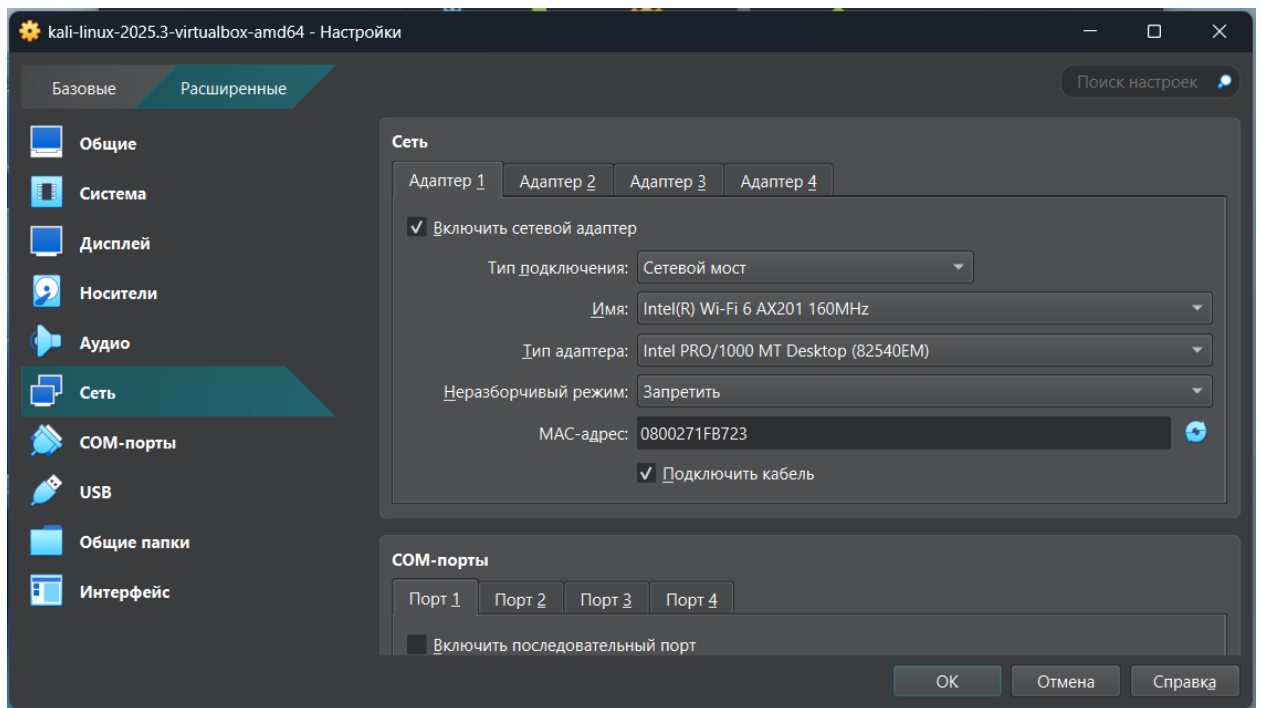


Рисунок 3 – Сетевые настройки Kali

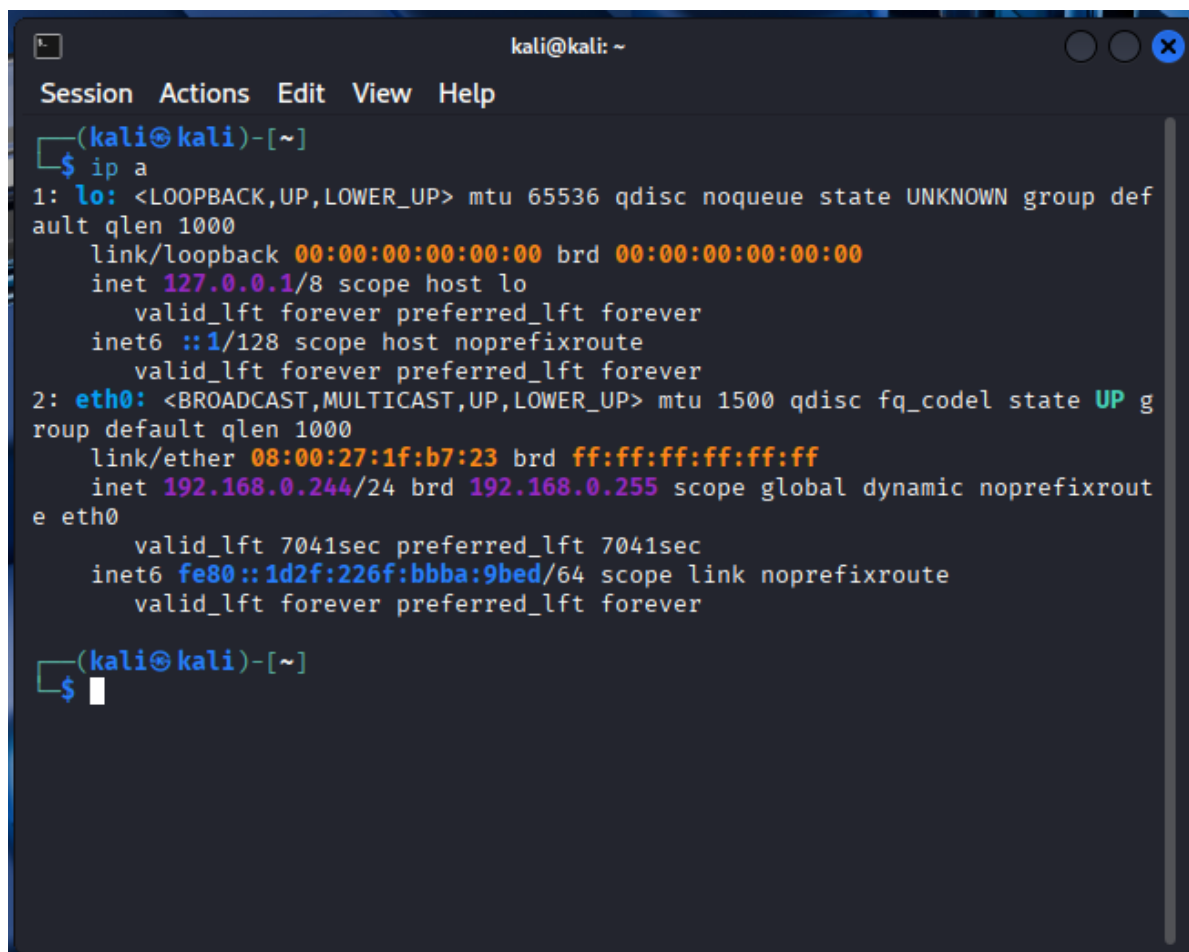
Информация информации об ip-адресах на обеих ВМ представлено на рисунках 4-5.

```

remselfy@vbox: ~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    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
    inet6 ::1/128 scope host noprefixroute
        valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:72:33:f4 brd ff:ff:ff:ff:ff:ff
    altname enx0800277233f4
    inet 192.168.0.104/24 brd 192.168.0.255 scope global dynamic noprefixroute enp0s3
        valid_lft 7066sec preferred_lft 7066sec
    inet6 fe80::a00:27ff:fe72:33f4/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
remselfy@vbox: ~$

```

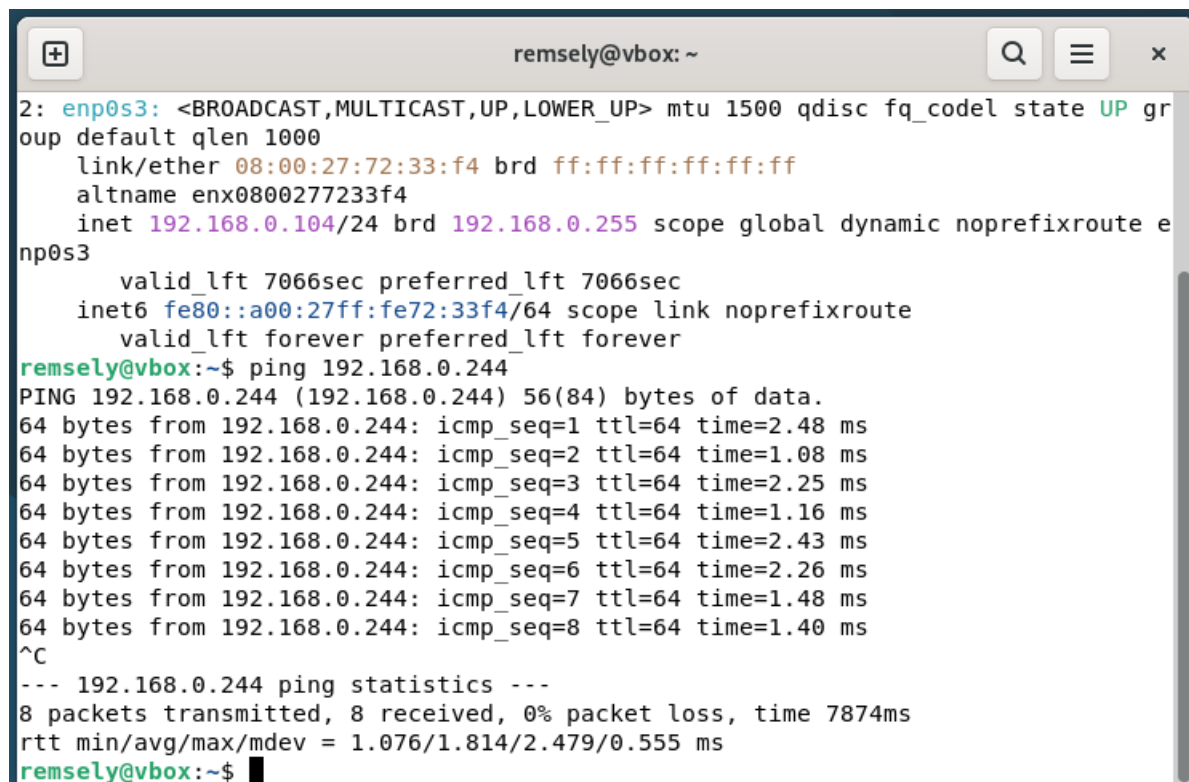
Рисунок 4 – Получение ip-адреса на Debian



```
kali@kali: ~  
Session Actions Edit View Help  
(kali@kali)-[~]  
$ ip a  
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000  
    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  
    inet6 ::1/128 scope host noprefixroute  
        valid_lft forever preferred_lft forever  
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000  
    link/ether 08:00:27:1f:b7:23 brd ff:ff:ff:ff:ff:ff  
    inet 192.168.0.244/24 brd 192.168.0.255 scope global dynamic noprefixroute eth0  
        valid_lft 7041sec preferred_lft 7041sec  
    inet6 fe80::1d2f:226f:bbba:9bed/64 scope link noprefixroute  
        valid_lft forever preferred_lft forever  
(kali@kali)-[~]  
$
```

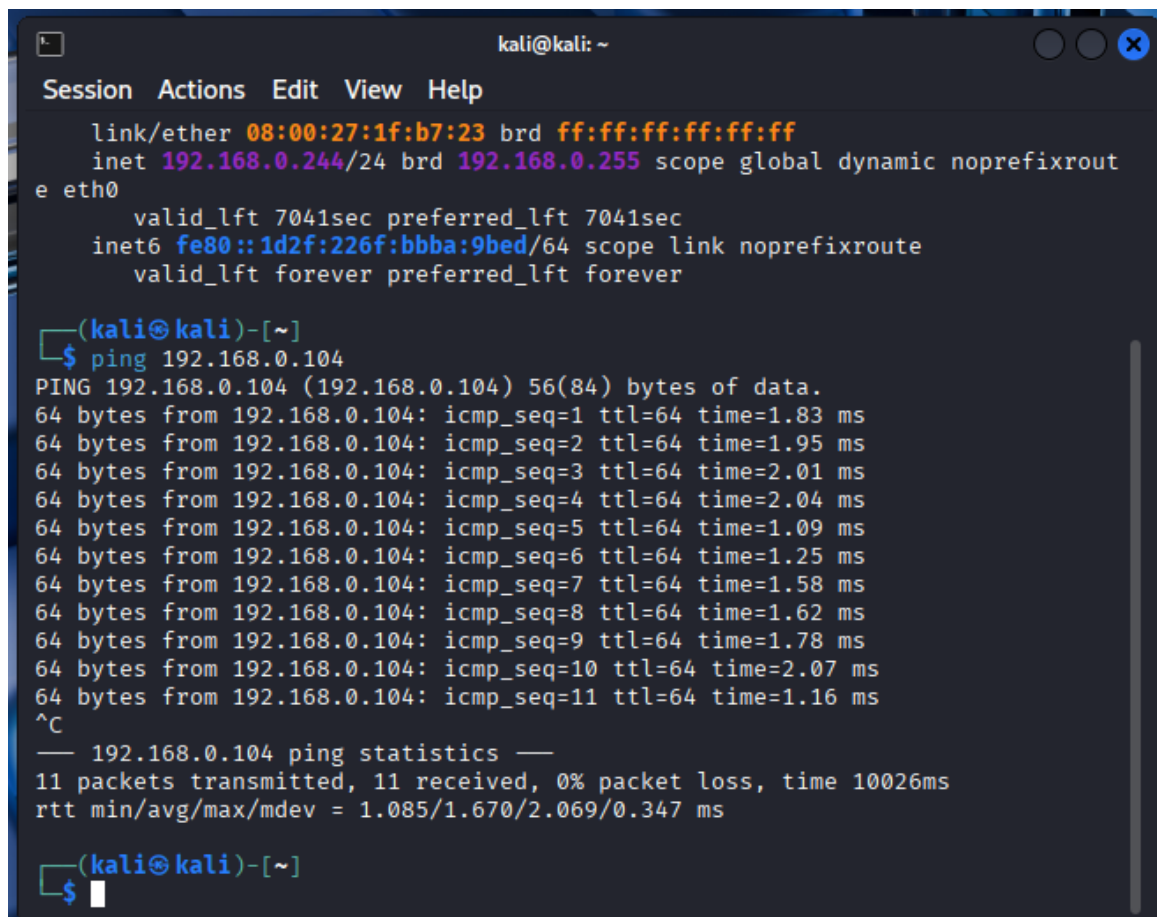
Рисунок 5 – Получение ip-адреса на Kali

Проверка соединения между машинами представлена на рисунках 6-7.



```
remsely@vbox: ~  
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000  
    link/ether 08:00:27:72:33:f4 brd ff:ff:ff:ff:ff:ff  
    altname enx0800277233f4  
    inet 192.168.0.104/24 brd 192.168.0.255 scope global dynamic noprefixroute enp0s3  
        valid_lft 7066sec preferred_lft 7066sec  
    inet6 fe80::a00:27ff:fe72:33f4/64 scope link noprefixroute  
        valid_lft forever preferred_lft forever  
remsely@vbox:~$ ping 192.168.0.244  
PING 192.168.0.244 (192.168.0.244) 56(84) bytes of data:  
64 bytes from 192.168.0.244: icmp_seq=1 ttl=64 time=2.48 ms  
64 bytes from 192.168.0.244: icmp_seq=2 ttl=64 time=1.08 ms  
64 bytes from 192.168.0.244: icmp_seq=3 ttl=64 time=2.25 ms  
64 bytes from 192.168.0.244: icmp_seq=4 ttl=64 time=1.16 ms  
64 bytes from 192.168.0.244: icmp_seq=5 ttl=64 time=2.43 ms  
64 bytes from 192.168.0.244: icmp_seq=6 ttl=64 time=2.26 ms  
64 bytes from 192.168.0.244: icmp_seq=7 ttl=64 time=1.48 ms  
64 bytes from 192.168.0.244: icmp_seq=8 ttl=64 time=1.40 ms  
^C  
--- 192.168.0.244 ping statistics ---  
8 packets transmitted, 8 received, 0% packet loss, time 7874ms  
rtt min/avg/max/mdev = 1.076/1.814/2.479/0.555 ms  
remsely@vbox:~$
```

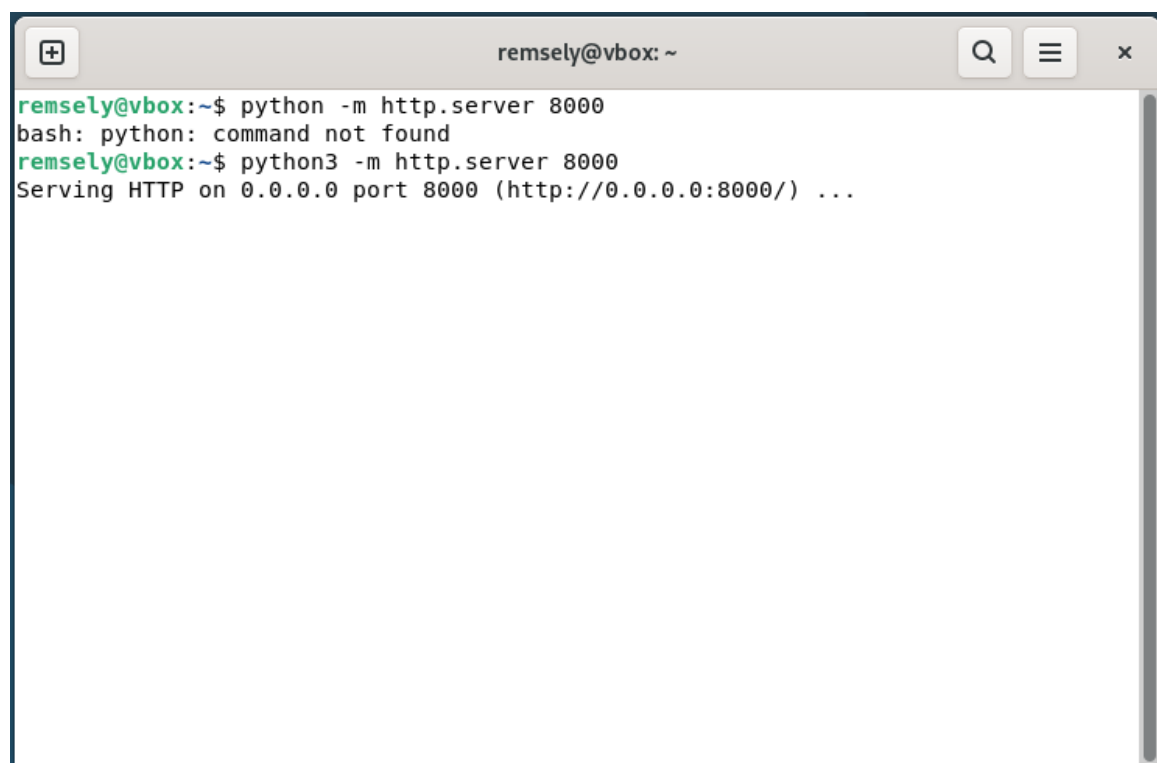
Рисунок 6 – Проверка соединения из Debian в Kali



```
kali@kali: ~  
Session Actions Edit View Help  
link/ether 08:00:27:1f:b7:23 brd ff:ff:ff:ff:ff:ff  
inet 192.168.0.244/24 brd 192.168.0.255 scope global dynamic noprefixroute  
eth0  
valid_lft 7041sec preferred_lft 7041sec  
inet6 fe80::1d2f:226f:bbba:9bed/64 scope link noprefixroute  
valid_lft forever preferred_lft forever  
  
(kali㉿kali)-[~]  
$ ping 192.168.0.104  
PING 192.168.0.104 (192.168.0.104) 56(84) bytes of data.  
64 bytes from 192.168.0.104: icmp_seq=1 ttl=64 time=1.83 ms  
64 bytes from 192.168.0.104: icmp_seq=2 ttl=64 time=1.95 ms  
64 bytes from 192.168.0.104: icmp_seq=3 ttl=64 time=2.01 ms  
64 bytes from 192.168.0.104: icmp_seq=4 ttl=64 time=2.04 ms  
64 bytes from 192.168.0.104: icmp_seq=5 ttl=64 time=1.09 ms  
64 bytes from 192.168.0.104: icmp_seq=6 ttl=64 time=1.25 ms  
64 bytes from 192.168.0.104: icmp_seq=7 ttl=64 time=1.58 ms  
64 bytes from 192.168.0.104: icmp_seq=8 ttl=64 time=1.62 ms  
64 bytes from 192.168.0.104: icmp_seq=9 ttl=64 time=1.78 ms  
64 bytes from 192.168.0.104: icmp_seq=10 ttl=64 time=2.07 ms  
64 bytes from 192.168.0.104: icmp_seq=11 ttl=64 time=1.16 ms  
^C  
— 192.168.0.104 ping statistics —  
11 packets transmitted, 11 received, 0% packet loss, time 10026ms  
rtt min/avg/max/mdev = 1.085/1.670/2.069/0.347 ms  
  
(kali㉿kali)-[~]  
$
```

Рисунок 7 – Проверка соединения из Kali в Debian

Запуск http-сервера на Debian представлен нарисунке 8.



```
remsely@vbox: ~  
remsely@vbox:~$ python -m http.server 8000  
bash: python: command not found  
remsely@vbox:~$ python3 -m http.server 8000  
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
```

Рисунок 8 – Запуск http-сервера на Debian

Подключение к серверу из браузеров обеих ВМ представлено на рисунках 9-10.

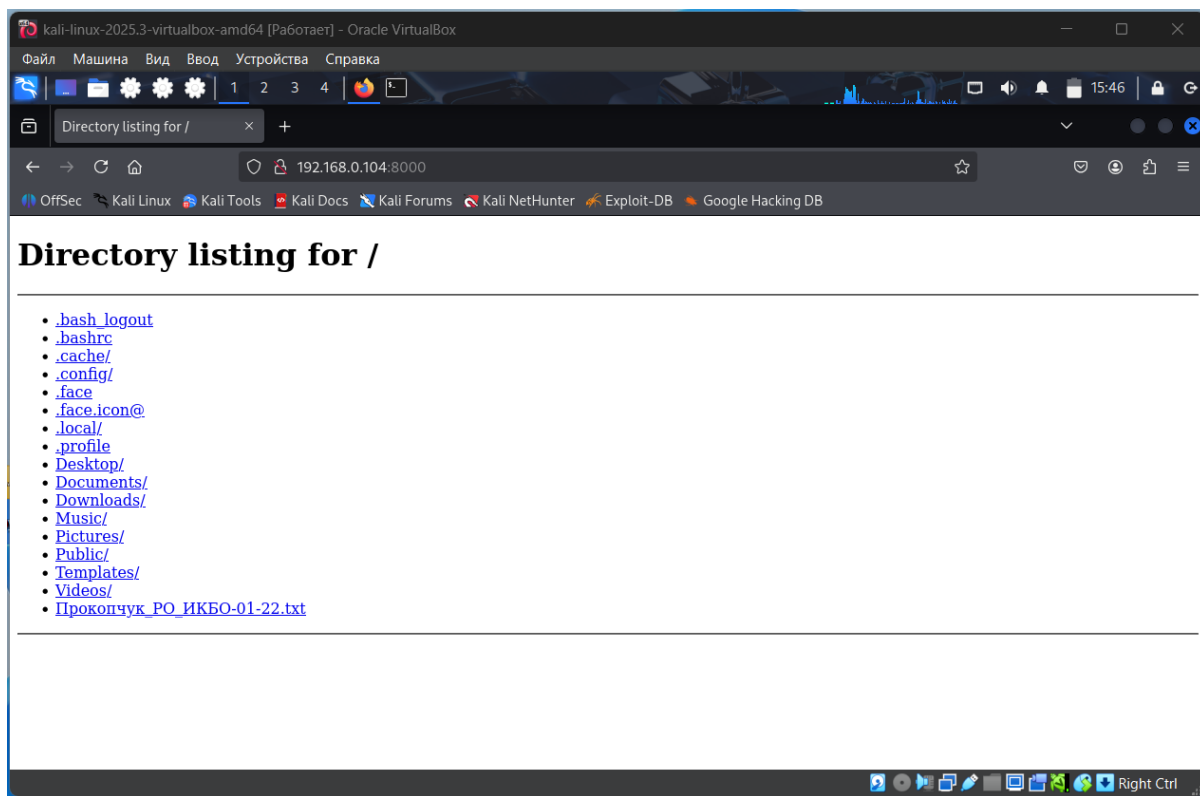


Рисунок 9 – Подключение к серверу с Kali

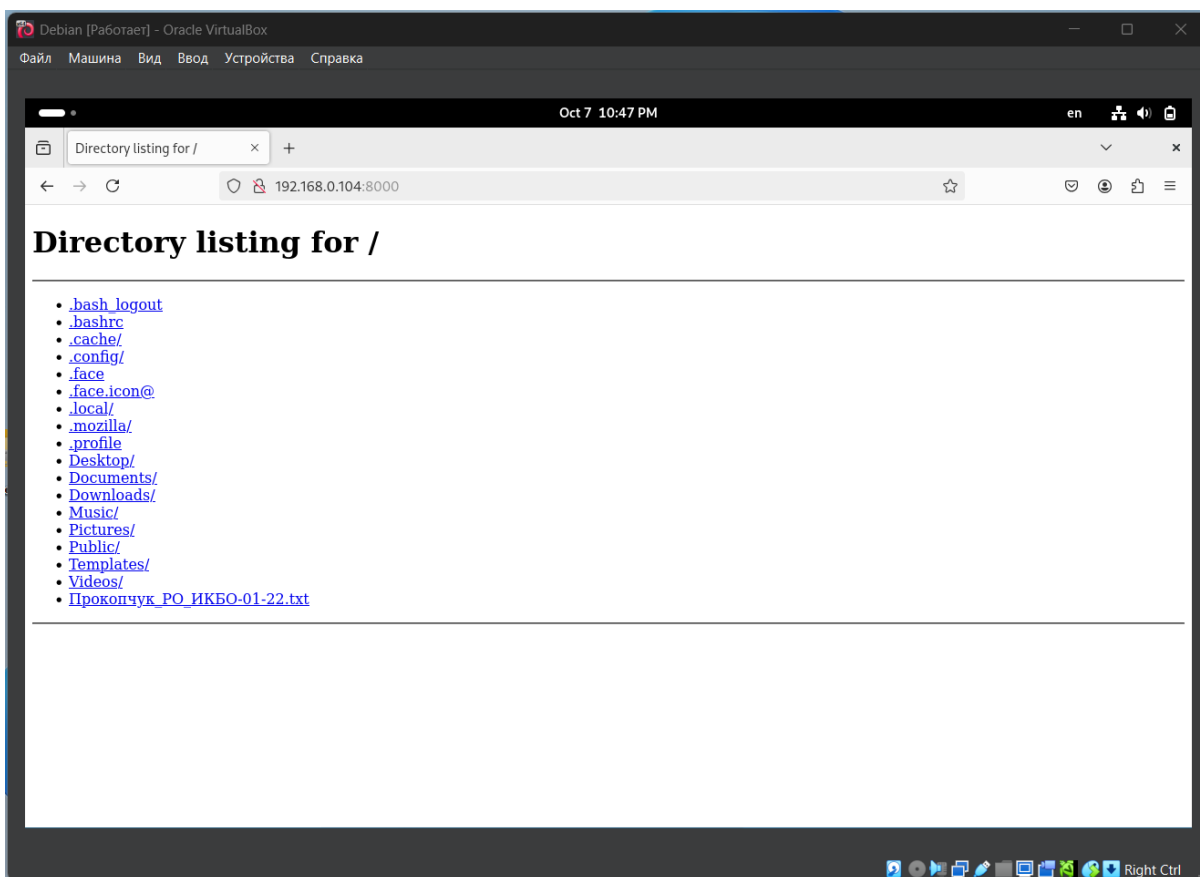


Рисунок 10 – Подключение к серверу с Debian (с хоста)

Вывод

В результате выполнения данной практической работы было настроено сетевое взаимодействие между двумя виртуальными машинами и хостом.