

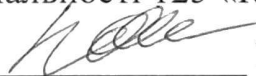
Міністерство освіти і науки України
Державний університет «Житомирська політехніка»
Факультет інформаційно-комп'ютерних технологій
Кафедра комп'ютерної інженерії та кібербезпеки

ПОЯСНЮВАЛЬНА ЗАПИСКА

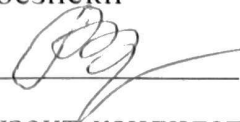
до випускної кваліфікаційної роботи магістра
на тему:

«Проект підсистеми захисту мережної інфраструктури
підприємства, побудованої з використанням технологій
"тонких" клієнтів»

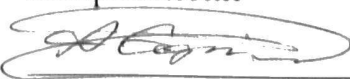
Виконав студент 2-го курсу, групи КБм-21-1
спеціальності 125 «Кібербезпека»

 Колесник О. С.

Керівник доктор технічних наук, доцент,
професор кафедри комп'ютерної інженерії та
кібербезпеки

 Воротніков В. В.

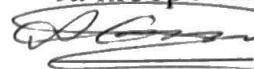
Рецензент кандидат технічних наук, доцент,
завідувач кафедри комп'ютерної інженерії та
кібербезпеки

 Єфіменко А. А.

ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»
ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ
КАФЕДРА КОМП'ЮТЕРНОЇ ІНЖЕНЕРІЇ ТА КІБЕРБЕЗПЕКИ
СПЕЦІАЛЬНІСТЬ 125 «КІБЕРБЕЗПЕКА»

ЗАТВЕРДЖУЮ

Завідувач кафедри
комп'ютерної інженерії
та кібербезпеки

 Єфіменко А. А.
«26» жовтня 2022 р.

ЗАВДАННЯ

на випускню кваліфікаційну роботу

Студента **Колесника Олексія Сергійовича**

Тема роботи: Проект підсистеми захисту мережної інфраструктури підприємства,
побудованої з використанням технологій "тонких" клієнтів

затверджена Наказом по університету №501/с від «25» жовтня 2022 р.

Термін здачі студентом закінченої роботи – «19» грудня 2022 р.

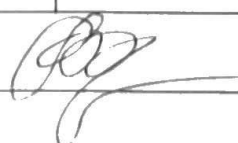
Вихідні дані для роботи:

1. Фізична топологія мережі
2. Мережевого сховище Truenas Scale 21.08
3. Маршрутизатор Mikrotik RB4011 (RouterOS v6)
4. Операційна системи Linux Ubuntu 21.04
5. Програмне забезпечення Cubic (Custom Ubuntu ISO Creator)
6. Графічний симулятор мереж GNS3

Консультанти з випускної кваліфікаційної роботи із зазначенням розділів, що їх стосуються

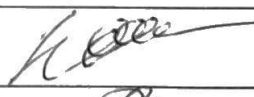
Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв
1	Воротніков В. В.	06.11.2022	06.11.2022
2	Воротніков В. В.	20.11.2022	20.11.2022
3	Воротніков В. В.	11.12.2022	11.12.2022

Керівник

 Воротніков В. В.

Календарний план

№ з/п	Назва етапів випускної кваліфікаційної роботи	Термін виконання етапів роботи	Примітка
1	Постановка завдання	1 жовтня 2022 – 4 жовтня 2022	Виконано
2	Пошук, огляд та аналіз літературних джерел щодо поставленого завдання підсистеми захисту мережної інфраструктури підприємства, побудованої з використанням технологій "тонких" клієнтів	5 жовтня 2022 – 7 жовтня 2022	Виконано
3	Розробка загального макету мережі організації з використанням технологій "тонких" клієнтів	8 жовтня 2022 – 11 жовтня 2022	Виконано
4	Проектування підключення локальної мережі до машин клієнтів та вибір обладнання для реалізації	12 жовтня 2022 – 15 жовтня 2022	Виконано
5	Проектування у врахуванням безпеки між процесами первинного завантаження	16 жовтня 2022 – 21 жовтня 2022	Виконано
6	Проектування та створення сценаріїв тестування функціоналу підсистеми	22 жовтня 2022 – 30 жовтня 2022	Виконано
7	Базове налагодження пристроїв мережі	31 жовтня 2022 – 4 листопада 2022	Виконано
8	Налагодження підсистеми захисту мережної інфраструктури підприємства, побудованої з використанням технологій "тонких" клієнтів	31 жовтня 2022 – 8 листопада 2022	Виконано
9	Тестування завантаження "тонких" клієнтів через мережу	9 листопада 2022 – 15 листопада 2022	Виконано
10	Аналіз результатів тестування	16 листопада 2022 – 23 листопада 2022	Виконано
11	Написання пояснювальної записки	24 листопада 2022 – 18 грудня 2022	Виконано
12	Дата подачі роботи	19 грудня 2022	Виконано
13	Отримання відгуку керівника та рецензії на кваліфікаційну роботу	21 грудня 2022	Виконано
14	Підготовка презентації та виступу	22 грудня 2022	Виконано
15	Захист	24 грудня 2022	Виконано

Студент  Колесник О.С.
Керівник  Воротніков В.В.

ABSTRACT

The qualifying work consists of the project of the subsystem of secure enterprise network infrastructure, built with using of “thin” client technology and explanatory note text. The explanatory note includes 69 pages, 47 pictures and 2 tables.

The purpose of this work is development within the project of the subsystem of secure enterprise network infrastructure.

This qualifying work describes the instructions on how to configure of network booting.

Network booting will separate on its own, as the boot process files on the server will be planned to be separated so that two simultaneous boot processes do not interrupt each other and work of the network. This means that the network booting will not form a queue – a few the same OS from the same server and different OS for smart cards can be booted simultaneously.

For maintenance of booting, from a network perspective, a configuration on the router to forward from the DHCP-server to the next server is required. During netbooting, the TFTP-server is assumed to be the next. This completes the router's influence on the netbooting act, except maintain of the network functioning and define boot file name, if it is available.

The qualifying work contains three main sections: theoretical information, the network engineering, choosing of devices for the network and all the necessary configurations for this.

KEYWORD: PXE, KICKSTART, PXEBOOT, BOOT, NETBOOTING, NETWORK BOOTING, "THIN" CLIENT

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ	7
ВСТУП	8
РОЗДІЛ 1 ТЕОРЕТИЧНІ ОСНОВИ ПОБУДОВИ ТА АКТУЛЬНІСТЬ ПРОЕКТУ ПІДСИСТЕМИ ЗАХИСТУ МЕРЕЖНОЇ ІНФРАСТРУКТУРИ ПІДПРИЄМСТВА, ПОБУДОВАНОЇ З ВИКОРИСТАННЯМ ТЕХНОЛОГІЙ "ТОНКИХ" КЛІЄНТІВ	10
1.1 ТЕОРЕТИЧНІ АСПЕКТИ ОРГАНІЗАЦІЇ МЕРЕЖІ З ВИКОРИСТАННЯМ ТЕХНОЛОГІЙ "ТОНКИХ" КЛІЄНТІВ	10
1.2 ТЕОРЕТИЧНИЙ АНАЛІЗ ЗАХИСТУ СУЧАСНОГО СТАНУ РОЗВИТКУ ТЕХНОЛОГІЙ "ТОНКИХ" КЛІЄНТІВ	15
1.3 МОЖЛИВОСТІ ЗАХИЩЕНОГО МЕРЕЖЕВОГО ЗАВАНТАЖЕННЯ	22
РОЗДІЛ 2 ПРОЕКТУВАННЯ МЕРЕЖІ З МОЖЛИВІСТЮ ЗАХИЩЕНОГО ПІДКЛЮЧЕННЯ ДО МЕРЕЖІ ТА ЗАВАНТАЖУВАННЯ "ТОНКИХ" КЛІЄНТІВ	25
2.1 ПРИНЦИПИ ПРОЕКТУВАННЯ МЕРЕЖІ	25
2.2 ЛОГІЧНЕ ПРОЕКТУВАННЯ МЕРЕЖІ	27
2.3 ВИБІР ОБЛАДНАННЯ ТА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ ОРГАНІЗАЦІЇ РОБОТИ МЕРЕЖІ.....	30
РОЗДІЛ 3 НАЛАГОДЖЕННЯ ПІДСИСТЕМИ ЗАХИСТУ МЕРЕЖНОЇ ІНФРАСТРУКТУРИ ПІДПРИЄМСТВА, ПОБУДОВАНОЇ З ВИКОРИСТАННЯМ ТЕХНОЛОГІЙ "ТОНКИХ" КЛІЄНТІВ	37
3.1 НАЛАШТУВАННЯ МЕРЕЖІ ТА ПІДГОТОВКА ІНФРАСТРУКТУРИ ПІДПРИЄМСТВА, ПОБУДОВАНОЇ З ВИКОРИСТАННЯМ ТЕХНОЛОГІЙ "ТОНКИХ" КЛІЄНТІВ	37
3.2 ТЕСТУВАННЯ МЕРЕЖІ НА БАЗІ СТВОРЕНОЇ ІНФРАСТРУКТУРИ ДЛЯ ЗАВАНТАЖЕННЯ "ТОНКИХ" КЛІЄНТІВ	50
3.3 АНАЛІЗ РЕЗУЛЬТАТІВ ТЕСТУВАННЯ МЕРЕЖЕВОГО ЗАВАНТАЖЕННЯ.....	59
ВИСНОВКИ.....	68
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ.....	71
ДОДАТКИ.....	74

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

ОС – операційна система
ПЗ – програмне забезпечення
ПК – персональний комп'ютер
ФС – файлова система
ACL – Access Control List
DHCP – Dynamic Host Configuration Protocol
IOT – Internet of Things
IP – Internet Protocol
ISP – Internet Service Provider
ISO - International Standards Organization
LAN – Local Area Network
MAC – Media Access Control
NFS - Network File System
POE – Power over Ethernet
PXE – Preboot Execution Environment
RAM - Random Access Memory
ROM - Read Only Memory
TCP – Transmission Control Protocol
TFTP – Trivial File Transfer Protocol
UDP – User Datagram Protocol
UI – User Interface
UUID - Universally Unique Identifier
VLAN – Virtual Local Area Network
WAN – Wide Area Network

ВСТУП

Актуальність теми. В умовах множинних інформаційних потоків, які фундаментально навівають умови нинішніх реалій, дуже зручно при вирішенні малих чи масштабних задач, коли відсутня необхідність вирішувати, яким буде робочий простір, а саме ОС, на ПК - дуже зручно, коли більше нема потреби бігати з флешкою від одного робочого місця до іншого. У цій роботі буде розглянуто питання налагодження завантаження робочого простору ОС у інфраструктурі, тобто "тонких" клієнтів. Як це може впросити життя або як це може зашкодити. Тому коли ми піднімаємо питання щодо масштабування, слід ураховувати наскільки масштабними можуть бути наслідки, що витікають звідси.

Метою кваліфікаційної роботи є розробка проекту підсистеми захисту мережної інфраструктури підприємства, побудованої з використанням технологій "тонких" клієнтів. Встановлена мета обумовлює підвищення надійності завантаження ОС. Необхідно забезпечити безпечне завантаження, яке унеможливить доступ до даних, що знаходяться локально на пристрої, що завантажується. Процес завантаження буде зроблено таким чином, що користувач не повинен слідкувати за завантаженням та встановленням – це буде зроблено автоматично. Підтримка завантаженої ОС буде здійснюватися з хаб-сервера, на якому будуть створюватись користувачі з необхідними на те правами доступу, та кожному буде виділено область даних відповідно до користувача.

Об'єктом дослідження є процес захищеного використання технологій "тонких" клієнтів. Важливо забезпечити безперешкодне підключення та швидкий доступ до робочого простору, щоб була надана змога виконувати свою роботу не вражаючи локальні медіа, чи будь-які дані кожного з пристроїв, що завантажуються. Таке дослідження вимушує при виборі завантажувача ОС звернути увагу на ФС цієї системи. Наша організація має містити, що найменше:

- DHCP-сервер (на бріджовому маршрутизаторі);
- TFTP-сервер (для налагодження первинного завантаження ОС та передачі файлів);

- NFS-сервер (на цьому етапі завантаження буде монтувати вже сформовану кореневу ФС, щоб не витратити час на її розгортання);
- "Тонкі" клієнти (завантажуванні ПК);
- Хаб-сервер (дата центр для зберігання файлів кореневої ФС та керування виділенням робочого простору клієнтів);
- Ethernet-підключення (фізичне підключення для такого завантаження необхідне та потребує пряме підключення до локальної мережі).

Головна небезпека полягає в тому, що вузлу можливо підключитися до пристрою та безперешкодно завантажитися, а в гірших випадках замінити дані первинного завантаження, тим самим можливий витік та пошкодження даних пристроїв, котрі завантажуються. На далі у майбутньому залишатимуться ризики використання тих чи інших вразливостей. Кіберзлочинці цілком можуть дістатися до ваших файлів і вкрасти якусь конфіденційну інформацію, яку потім можуть збути з вигодою для себе, або використати її як інструмент шантажу тощо.

Предметом дослідження є методи і засоби використання власних ОС, захищеність даних, та й в цілому доцільно розглянути питання безпечного завантаження ОС з серверу через мережу в рамках одного обмеженого робочого простору. Необхідно подбати про безпеку користувачів локальної мережі, які будуть завантажувати на свої пристрої нашу систему/системи. Вирішити, що в нашому випадку буде надійніше: видавати права доступу користувачам з подальшою підтримкою на самому сервері чи краще вбудовувати управління відразу в ОС для унеможливлення скористатися таким уразливим функціоналом.

Апробація результатів:

1. Колесник О.С., Вакалюк Т.А. Проект підсистеми захищеного багатоточкового підключення до мережі провайдера інтернет // Тези IV Всеукраїнської науково-практичної інтернет-конференції здобувачів вищої освіти і молодих учених «Інформаційно-комп'ютерні технології: стан, досягнення та перспективи розвитку», м. Житомир, 25-26 листопада 2021 року – Житомир: Житомирська політехніка, 2021. С. 60-61.

					КБ.КВ.Б – 125 – 22– ПЗ	
						9

РОЗДІЛ 1 ТЕОРЕТИЧНІ ОСНОВИ ПОБУДОВИ ТА АКТУЛЬНІСТЬ ПРОЕКТУ ПІДСИСТЕМИ ЗАХИСТУ МЕРЕЖНОЇ ІНФРАСТРУКТУРИ ПІДПРИЄМСТВА, ПОБУДОВАНОЇ З ВИКОРИСТАННЯМ ТЕХНОЛОГІЙ "ТОНКИХ" КЛІЄНТІВ

1.1 Теоретичні аспекти організації мережі з використанням технології "тонких" клієнтів

На сьогоднішній день, при створенні мережі малої чи великої компанії виникає важливе питання, від якого надалі залежить працездатність самої ж компанії, а саме буде описуватись процес завантаження бездискових Linux-станцій за допомогою PXE, а також як створити власну збірну ОС з усіма попередніми завантаженнями утиліт та інструментів з хаб-серверу для таких станцій для використання у подальшій роботі, покривати та досягати цілі створення такого офісного робочого простору. Розглядатиметься процес налаштування всіх необхідних компонентів (серверів - TFTP, DHCP, NFS), а також наведення вдалих прикладів конфігураційних файлів.

Перш ніж почати обирати обладнання чи інструменти для роботи з цим питанням, необхідно розглянути як цей процес взагалі відпрацьовує, його етапи та як передається керування при завантаженні. Тому що основна ідея закладається саме в автоматизованому мережевому завантаженні, через це захищеність може стати ще більш делікатним моментом для усієї інфраструктури, що використовує дану технологію.

Для забезпечення процедур завантаження використовується механізм **PXE:**

PXE (Preboot Execution Environment) - середовище для завантаження комп'ютерів за допомогою мережевої картки без використання жорстких дисків, накопичувачів, компакт-дисків та інших пристроїв, які застосовуються під час процесу завантаження ОС. Для організації завантаження системи завантаження через PXE використовуються що найменше такі протоколи: IP, TCP, UDP, DHCP та TFTP.

Сам же PXE-код прописаний на мережевій карті, очікує команду отримувати завантажувати по мережі, після чого передається керування безпосередньо до PXE.

Завантаження Linux з використанням PXE

Завантаження Linux-системи за допомогою PXE можна виконати за допомогою існуючого завантажувача PXELINUX, який було реалізовано на базі з урахуванням SYSLINUX. PXELINUX завантажується за допомогою PXE-коду в мережевій карті, а потім передається керування до нього. Код PXELINUX не потрібно прописувати в чіп на карті мережі (рисунок 1.1).

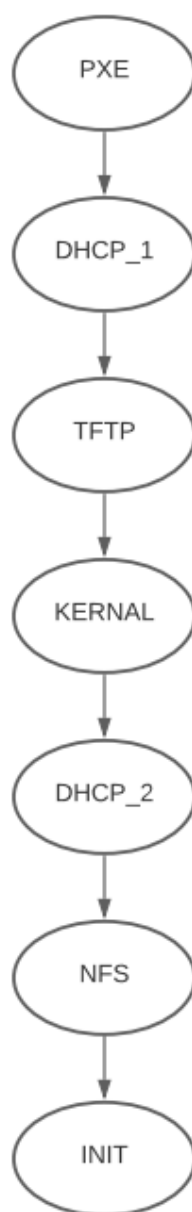


Рисунок 1.1. Процес первинного завантаження PXE

- PXE – З початку виконується завантаження PXE-завантажувача. Прошивка PXE, вбудована в мережеву карту - виконується зчитування та підкачка завантажувача `pxelinux`.
- DHCP_1 – Далі виконується ініціалізація DHCP - отримання IP-адреси з необхідної підмережі у якій знаходиться TFTP-сервер. Завантажувач `pxelinux` робить DHCP-запит і за його властивостями отримує IP-адресу та здатен смикнути адресу TFTP-сервера. За замовчуванням адреса нашого TFTP-сервера релевантна адресі DHCP-сервер.
- TFTP – На цьому етапі виконується звернення до TFTP-сервера для отримання ядра системи Linux. Завантажувач `pxelinux` звертається до TFTP-сервера із запитом на отримання ядра Linux (за необхідності із запитом на отримання образу `initrd`).
- KERNEL – Далі формується запуск ядра Linux. Після того як ядро Linux передано та завантажено – права на керування процесом передаються до нього.
- DHCP_2 – Ініціалізацію було виконано та процесом було досягнуто отримання IP-адреси DHCP. Ядро Linux робить запит відносно DHCP, за допомогою якого отримує необхідну IP-адресу виділених ним із його таблиці маршрутизації. Дістається адреса NFS-сервера, на якому знаходиться коренева ФС, а також шлях до розташування цієї ФС відносно фізичного шляху на жорсткому диску.
- NFS – Виконується монтування кореневої ФС. По завершенню коренева ФС буде передана та вмонтована на машину, котра завантажується, для пришвидшення розгортання ОС з усіма пред-установленнями.
- INIT – На завершальному етапі викликається процес `init`. На кореневій ФС знаходиться файл `/sbin/init`, якому передається остаточне керування, так як це перший користувацький процес в Unix-подібних ОС [1].

Далі розглянемо вже більш детально взаємозв'язок між процесами первинного завантаження ОС на зображенні (рисунок 1.2):

					КБ.КВ.Б – 125 – 22– ПЗ	
						12

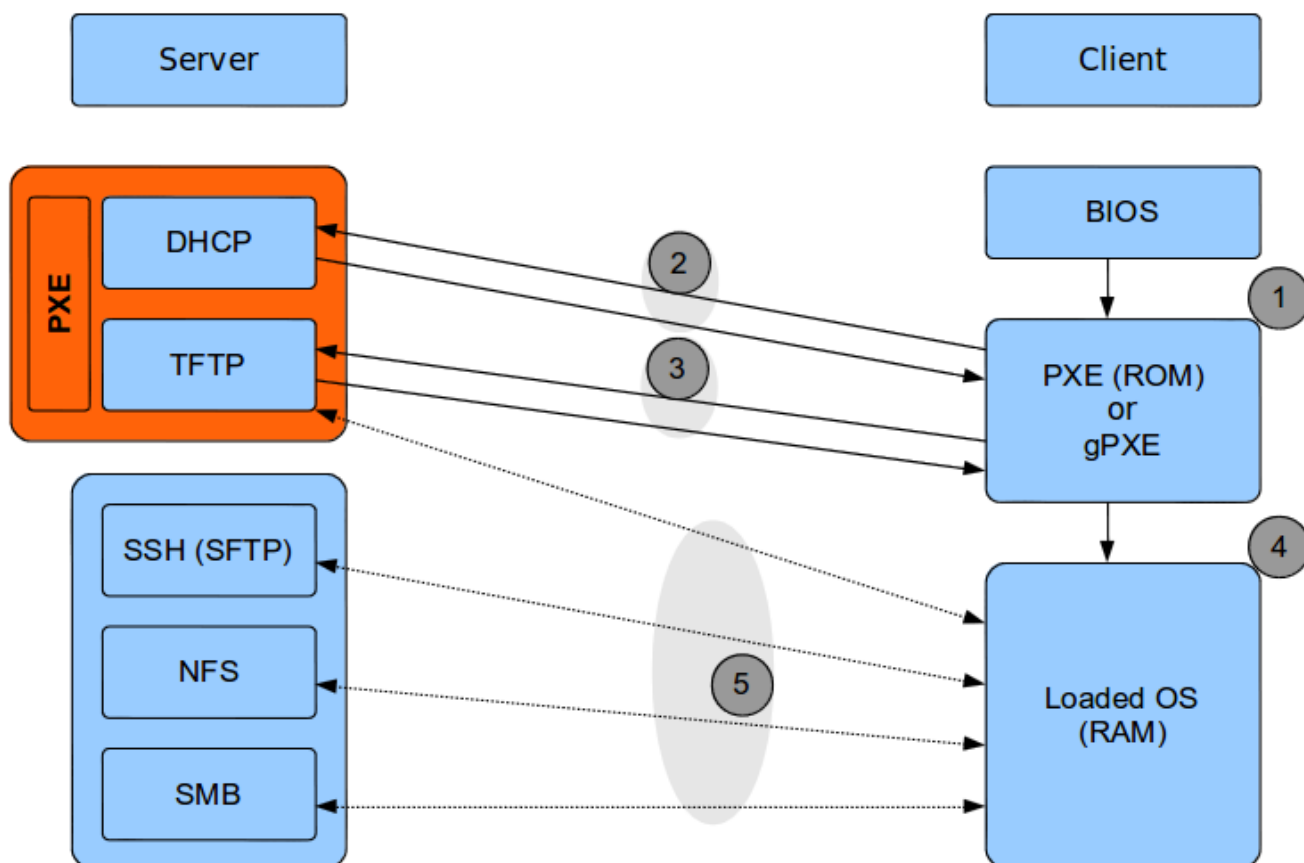


Рисунок 1.2. Взаємозв'язок між процесами PXE первинного мережевого завантаження

- 1) Виконується коректне ввімкнення клієнтського комп'ютера. Тоді BIOS сканує пристрої. Потім слід спробувати підключити зчитування завантажувача з відповідного пристрою, як зазначено вище про порядок / послідовність завантаження. При цьому буде завантажений PXE з ROM'у мережевої карти.
- 2) Далі вже PXE (на клієнтській стороні) передає запит на отримання IP-адреси. Відповідь DHCP-сервера міститиме необхідну інформацію для подальших етапів завантаження, а саме:
 - IP-адресу DHCP-серверу, який перекине нас на наступний сервер;
 - IP-адресу TFTP-серверу, який автоматично вкаже на файл, з якого розпочнеться ланцюг завантажень;
 - *Pxelinux.0* - назва файлу завантажувача PXE, можна вважати, що разом із цим буде передана подальша необхідна інформація для наступних дій.

- 3) На момент, коли вже виконується PXE на стороні клієнта виконує запит на отримання файлу *pxelinux.0* через TFTP-сервер, файл буде завантажено в оперативну пам'ять з подальшим управлінням *pxelinux.0* цим процесом. Так як завантаження буде виконано в RAM без корегувань й взаємодії з жорстким накопичувачем клієнтської машини - його наявність необов'язкова при такій конфігурації.

Pxelinux.0 отримує файл конфігурації завантаження з TFTP-серверу з таким списком індикаторів як (на випадок, щоб перейти до наступного, якщо попередній відсутній у конфігурації):

- UUID
- MAC
- IP-адреса в шістнадцятковій системі числення (Capital Hex)
- Унікальний мережевий номер IP фіксований підмережою (клас IP адреси)
- Переадресовує на *default* файл

- 4) В залежності від конфігурації далі вибір залишатиметься за користувачем на запуск завантаження ОС через меню або автоматичне завантаження за замовчанням єдиної у нашому випадку ОС. *Pxelinux.0* вимагатиме запит на відповідне ядро RAM-диску з TFTP-серверу, та завантажуватиме в RAM.

І після цього передає параметри та керування завантаженому ядру *kernal*.

Нижче прикріплено зображення для більш наглядного уявлення успішного мережевого завантаження, як приклад Live CD ISO образу (рисунок 1.3):

```
Listening on LPF/ens3/02:56:e6:08:2d:e3
Sending on LPF/ens3/02:56:e6:08:2d:e3
Sending on Socket/fallback
DHCPDISCOVER on ens3 to 255.255.255.255 port 67 interval 3 (xid=0x83e74430)
DHCPOFFER of 10.228.68.73 from 10.228.68.5
DHCPREQUEST for 10.228.68.73 on ens3 to 255.255.255.255 port 67 (xid=0x3044e783)
DHCPACK of 10.228.68.73 from 10.228.68.5 (xid=0x83e74430)
bound to 10.228.68.73 -- renewal in 286 seconds.
Connecting to 10.228.68.91 (10.228.68.91:80)
ubuntu-20.04.1-live- 0% | | 462k 0:34:48 ETA
ubuntu-20.04.1-live- 2% | | 22.2M 0:01:20 ETA
ubuntu-20.04.1-live- 4% |*| 39.7M 0:01:05 ETA
ubuntu-20.04.1-live- 6% |**| 59.6M 0:00:57 ETA
ubuntu-20.04.1-live- 8% |***| 77.7M 0:00:53 ETA
ubuntu-20.04.1-live- 10% |****| 97.7M 0:00:50 ETA
ubuntu-20.04.1-live- 12% |*****| 114M 0:00:49 ETA
ubuntu-20.04.1-live- 14% |*****| 132M 0:00:47 ETA
ubuntu-20.04.1-live- 16% |*****| 149M 0:00:45 ETA
ubuntu-20.04.1-live- 18% |*****| 166M 0:00:45 ETA
ubuntu-20.04.1-live- 20% |*****| 183M 0:00:43 ETA
ubuntu-20.04.1-live- 21% |*****| 200M 0:00:42 ETA
ubuntu-20.04.1-live- 23% |*****| 214M 0:00:42 ETA
```

Рисунок 1.3 – Успішний запуск завантаження через налаштування UEFI PXE

					КБ.КВ.Б – 125 – 22– ПЗ	14

- 5) На цьому етапі ядро завантажує відсутні частини з TFTP-серверу або альтернативних релевантних йому серверних протоколів в залежності від залежних для нього функцій і властивостей [2].

1.2 Теоретичний аналіз захисту сучасного стану розвитку технологій "тонких" клієнтів

Тепер коли більш детально розглянуто питання взаємодії між процесами первинного завантаження PXE, необхідно не забувати про аспект безпеки. Для якого треба хоча б мати уявлення про безпеку та надійність наступних запланованих налаштувань завантаження ОС.

Впровадження безпеки обумовлене тим, що зловмисники можуть зловживати механізмами пре-завантаження ОС як способом встановити нестабільність у систем. Не мало важно враховувати, що під час процесу завантаження, насамперед завантажується прошивка комп'ютера, а слідом відбувається завантаження усіляких необхідних служб автозапуску. І потім вже виконується завантаження операційної системи, та це може зіграти немалий жарт через низькорівневу вразливість даного процесу. Машинна прошивка та служби контролюють процесом завантаження до тих пір, поки виконання на себе не візьме ОС з усіма своїми подальшими процесами.

Такі вразливості можуть використовуватися кіберзлочинцями, які мають належний досвід у написанні драйверів, або мають навички управління процесами на низьких рівнях моделі OSI.

Хакери при необхідності зможуть скористатися таким недоліком у системі, щоб замінити частину вже існуючого драйверу, переписати дані в драйвері завантаження, або навіть написати власну мікропрограму для прошивки. Наприклад таким недоліком можу послугувати BIOS, який по своїм принципам та властивостям слугує для входу – виходу з пристрою на базовому рівні, також UEFI потребуватиме вирішення з аналогічним питанням, що зловмисники не змогли засісти на рівні нижче операційної системи до її розгортання, та

унеможливити будь-які операції з зовні організації. Впровадження додаткових систем сканування вузлів, портів буде немисливо важливим засобом запобігання у разі, коли такі інциденти відбуваються вперше, щоб унеможливити їх у майбутньому. На цьому рівні особливо важко вводити ланки безпеки, так як зловмисне ПЗ на цьому рівні не буде виявлено програмним захистом вузла, виламаной машини – тому такі атаки краще обмежувати, чи об'єднувати машини та впроваджувати додаткові сервіси [3].

Системна прошивка

Зловмисники матимуть змогу зловживати підставною заміною прошивки системи, щоб закріпитися у системі на відповідно низькому рівні. Прикладами системної прошивки слугують:

- BIOS (Basic Input/Output System);
- UEFI (The Unified Extensible Firmware Interface);
- EFI (Extensible Firmware Interface).

Які в свою в свою чергу тісно зв'язані апаратним забезпеченням кожного пристрою, як в принципі. Їх зона відповідальності спрямована на те, щоб слугувати UI (user interface), надавати змогу відображати графічний програмний інтерфейс для користувача між ОС та апаратним забезпеченням комп'ютера.

Системні прошивки слугують базовим рівнем для функціонування будь-якого комп'ютера, і як будь-яке ПЗ матиме точку входу для свого перезапису чи заміщення зловмисниками, якщо не розібрати це питання більш детально. Таким чином системна прошивка може бути використана для виконання несанкціонованих маніпуляції у системі, котра завантажується по даному принципу.

Так як вже було виявлено, то системні прошивки мають можливості перезапису, що може слугувати їжею для хакерів - можливий витік та пошкодження даних пристроїв даної організації. Для цих цілей використовуються зловмисні встановлення сторонніх ПЗ [4].

Компонування прошивки

Сторонні модифікації системної прошивки також є насущною проблемою, якщо зломисники закріпляться в системі, то вони матимуть змогу змінювати код на якому була написана відповідна прошивка у системі. Такі компоненти прошивки наче незахищений мережевий вузол, котрий виходить на вулицю, і манить своїм існування там.

Зломисники при необхідності матимуть змогу використати high-level засоби чи утиліти для компрометації компонентів незахищеного приладу зі встановленням мікропрограм, які надаватимуть змогу виконувати зловмисні дії через код чи інтерфейс. Такі дії важко засікти без додаткових аудитів, тому що вони виконуються поза ОС, ззовні для апарату – на основній системній прошивці. Даний метод з точки зору захисту та безпеки може бути схожим на попередній, котрий напряду пов'язаний із системною прошивкою, але зловживається до зовсім інших системних компонентів на системному рівні пристрою, які вже можуть не мати таких же можливостей відслідковування у системі чи не будуть замічені на тому ж рівні перевірки цілісності.

Компоненти прошивки відтворені як мікропрограми можуть забезпечувати постійний безперешкодний доступ до системи, незважаючи на потенційні типи для такого способу збої у підтримці доступу та можливості заміни образів ОС на жорстких дисках, це також може відпрацьовувати як спосіб обійти програмний захист хоста та перевірки цілісності системи [5].

Bootkit

Буткіт — програмне забезпечення сімейства руткітів, головною функцією якого є зміна програми завантаження ОС, найпоширенішою з яких є завантажувальний сектор жорсткого диска чи Головний завантажувальний запис. Відомими представниками цього сімейства є:

- FinSpy
- TrickBot/TrickBoot
- Backdoor
- Mebroot

Як і інші руткіти, він використовується для отримання прав доступу в ОС. Буткіт може отримати права адміністратора і виконувати будь-які зловмисні дії. Наприклад, зловмисник здатен завантажити в пам'ять динамічні бібліотеки, яких взагалі не існує на жорстких накопичувачах. Це буде важко виявити способами звичайного користувача – засобами антивірусів [6]. Модель загрози (рисунок 1.4):

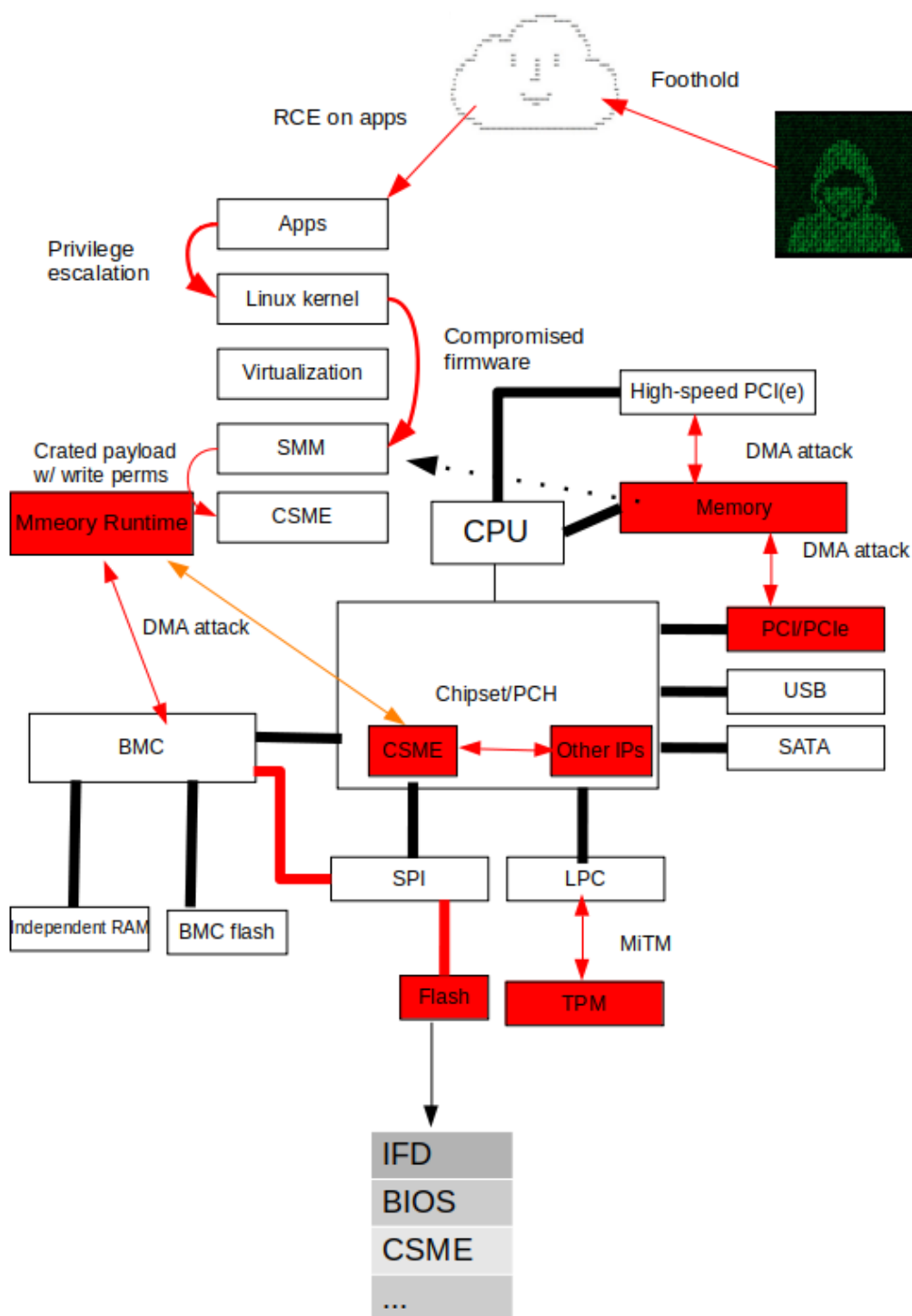


Рисунок 1.4 – Модель загрози буткіта

По аналогії з попередніми методами, теоретично зловмисники при необхідності матимуть змогу використати буткіти для проникнення й подальших маніпуляцій із системою. Буткіти також знаходяться на рівні нижче ніж ОС, їх важко виявити, якщо вони вже є наявними у системі, та нелегким є процес для їх повного відслідкування та виправлення у системи шляхом їх повного очищення. У випадках коли організація не підозрює, що вони використовуються зловмисника у їх конкретній системі, вони не зможуть діяти відповідним чином, щоб виконати міри, які дозволять повністю позбутися буткітів.

Доповнимо, що буткіти – це різновид зловмисного ПО, які націлені на зміну загрузочного сектору жорстких носії, таких як: MBR (головний завантажувальний запис) та VBR (завантажувальний запис тома).

MBR (з англ. master boot record) – це найбільш важливий розділ жорсткого диску, який з’являється у структурі даних на диску, та створюється при розбитті на розділи, у ланцюгу завантаження, підвантажується першим після завершення апаратної ініціалізації в BIOS. Таке положення завантаження хакери використовуються як невідворотній доступ до загрузочних дисків, з можливістю перезаписувати цю область, до якої було отримано доступ. Тим самим перемикання виконуватиметься під час запуску звичайного завантаження на модуль коду зловмисника [7].

VBR (з англ. volume boot record) – це перший сектор апаратного зберігання файлів, котрий не поділений на розділи зберігання даних, або перший сектор розділу апарату зберігання даних, котрий було визначено як розділ диску. Завантажувальні записи як в принципі часто містять комп’ютерний код для запуску процесу завантаження, або для виклику окремої програми чи ОС, котрі установлені на приладі чи на розділі [8].

MBR передає керування процесом завантаження VBR. Як в випадку з MBR, зловмисник при наявності несанкціонованого доступу до процесів загрузочного диску, може перезаписувати VBR, щоб виконувалось перенаправлення чи переадресація на ворожий код під час запуску процесу завантаження [9].

ROMMONkit

Набір інструментів для несанкціонованого доступу в систему машини, до того як вона розпакує та встановить ОС. По своїм принципам та подібностям нагадує руткіт.

Rootkit – це в більшості випадках програма або набір програм, які використовуються для приховування слідів присутності зловмисників або шкідливих несанкціонованих програм у системі. Це спеціальний модуль ядра, який хакер реалізує у системній прошивці, яку було успішно перехоплено, одразу після отримання адмінських доступів. Як правило, цей набір програм та інструментів містить різноманітні утиліти для підробки слідів присутності в системі сніфферів (програма для перехоплення мережевого трафіку), сканерів, кейлоггерів (програмний модуль, що зчитує та записує натиснення клавіш), троянські вірусні програми, замінюючи основні утиліти UNIX, якщо це неядерний руткіт. Дозволяє хакеру закріпитися в зараженій системі та приховати сліди своєї діяльності шляхом приховування файлів, процесів і наявності руткіта в системі.

Руткіт можна встановити в системі декількома способами: за допомогою експлойту (програмний модуль, що використовується здебільше для проведення DoS-атак, але також використовують для захоплення контролю над ПК за допомогою підвищення прав зловмисника по принципу оновлення ACL). Після таких маніпуляцій отримують доступ до оболонки, та в такому випадку на ПК можна завантажити руткіт із віддаленого пристрою за допомогою такого інструменту, як wget або базовий FTP-клієнт), у джерелі програмного продукту код або ресурси [10].

Зловмисники матимуть змогу зловживати монітором ROM, що відноситься до мережевої карти (ROMMON), для підключення до неавторизованої прошивки зі сторонніми засобами коду, або мікропрограм. Таких недолік може бути використаний для забезпечення ними постійного доступу та маніпуляцій з пристроєм та неочікуваним управлінням поведінкою пристрою, яку як вже було

сказано вище доволі важко виявляти без сторонніх засобів та без додаткових аудитів.

ROMMON – це прошивка мережевого пристрою від компанії Cisco, яка відпрацьовує як завантажувач, або завантажуваний образ, або ще як підтримка із допомогою завантаження для ініціалізації апаратного та програмного забезпечення під час запуску чи перезавантаження платформи. Подібно до завантаження по TFTP, зловмисник може замінити образ ROMMON як локально так і віддалено (наприклад, через TFTP) за допомогою шкідливого за своїм характером коду, та зловживаючи маніпуляцією перезапустити пристрій, щоб перезаписати існуючий образ ROMMON. Це надає супротивникам засоби для оновлення ROMMON на підставний, щоб закріпитися у системі на низькому рівні з подальшого зловживання цим [11].

Завантаження через TFTP

TFTP-сервер (з англ. Trivial File Transfer Protocol) – простий протокол, для покрокової синхронної передачі файлів, серверний протокол дозволяє його клієнтам читати або записувати серверні файли. Схожий на службу FTP, все частіше використовується для цілей завантаження мережевої ОС чи завантаження внутрішнього ПЗ на мережеві прилади, що належать до LAN організації, такі як маршрутизатори. TFTP доступний також і на ОС Linux та Windows [12].

Зловмисники можуть зловживати мережевим завантаженням ОС, щоб завантажити неавторизовану ОС мережевого пристрою з сервера по TFTP. Завантаження TFTP (з англ. netbooting – завантаження через локальну мережу) зазвичай використовується адміністраторами локальної мережі для завантаження образів мережевих пристроїв, керування конфігурацією власних ОС на рівні створення системи по-замочуванню, або контролювати централізовано на сервера адміністрування. Мережеве завантаження є одним із варіантів у послідовності завантаження, яке можна використовувати для централізації, управління, керування та корегування образами завантаженими на пристрої.

Зловмисники також можуть маніпулювати конфігурацією мережевого пристрою, вказуючи використання на замінений зловмисний TFTP-сервера, який може використовуватися разом із Modify System Image (зловмисний образ) для завантаження зміненого образу під час запуску пристрою або скидання шляхом перезавантаження чи перевстановлення ОС по мережі. Неавторизований образ або знімок бекапу дозволяє зловмисникам змінювати конфігурацію пристрою, додаванням до нього шкідливий функціонал та можливості, альтернативно вводиться для підтримки контролю над мережевим пристроєм, мінімізуючи виявлення за допомогою стандартної функціональності. Ця техніка схожа на ROMMONkit і може призвести до того, що мережевий пристрій запускає завантаження зміненого образу ОС та його подальше розгортання на апаратному пристрої [13].

1.3 Можливості захищеного мережевого завантаження

Можливості завантаження робочих станцій та пристроїв розумного офісу та автоматизації Raspberry Pi 4.

Можливості серверу завантаження PXE для тонкого клієнта на базі Ubuntu:

- Завантажитися повністю в оперативну пам'ять, не потребує монтування кореневої ФС по мережі.
- На основі Ubuntu можливо знайти майже будь-які пакети ПЗ в доступних репозитаріях або підключитися до зовнішнього, з власнествореним ПЗ. Особливо зручно те, що оновлення безпеки поступають у репозиторії Ubuntu досить швидко.
- Такий тип завантаження ОС може монтувати додаткові надбудови поверх кореневої ФС. Можливість керувати додаванням ПЗ лише для деяких робочих станцій без створення нових образів для цього.
- Використовувати стиснення операційної пам'яті (використовувалось у старих тонких клієнтів з невеликим обсягом оперативної пам'яті). Хоча для сучасних клієнтів це теж непогано та можна знайти цьому застосування.

- Готовий облеглий робочий стіл із RDP-клієнтом (віддалений доступ) реалізовано з коробки. Адреси та параметри серверів RDP просто передаються з сервера PXE у параметрах завантаження.
- При необхідності вести змінити, лише один параметр у конфігурації, і буде створено мінімальну налаштовану консольну систему. Це вдала база для власної збірки ОС.
- При виникненні проблем із завантаження через проблеми з сервером або працездатністю мережі, на короткий час – відпрацює автоматично повідомлення про помилку та розпочнеться спочатку. Зручність полягає у тому, що після усунення проблем - робочі станції запускаються автоматично без втручання вручну [14].

Можливості запланованої автоматизації офісу з ІОТ із використанням Raspberry Pi із власними кластерами. В кластерах розгортаються різного типу функціоналу контейнери, в залежності від потреб. Для підготовки середовища та ініціалізації вузлів кластера для роботи з ними потребується встановлення ОС. Можливість застосувати PXE завантажувати для плат Raspberry Pi 4 вирішує дуже багато питань, як з автоматизацією, так і з налаштуванням подальшого процесу розгортання. Відмова від SD-накопичувачів надає гнучкість цьому процесу:

- Максимально спростити завантаження та обслуговування плат.
- Максимально автоматизуйте їх оновлення.

Завантаження лише від підключення до мережі дозволяє досягати цієї гнучкості. Наприклад, коли завантажування плати чи ПК через мережу, для завантаження плати SD-карта не потрібна, а для робочої станції можна налаштувати процес автоматично, без необхідності бігати з жорстким носієм по офісу, чи слідкувати за кожним етапом встановлення. Як ОС, так і ФС можуть знаходитися на центральному сервері (мережевому сховищі). Більш зручно ініціалізувати ці процеси на центральному сервері, та зрештою автоматизувати ці сценарії [15].

Висновки до першого розділу

В результаті теоретичного аналізу організації мережі інфраструктури підприємства, побудованої з використанням технологій "тонких" клієнтів визначено, що для захищеного завантаження по мережі офісного (корпоративного) простору, необхідно застосувати комбінований спосіб захисту, який дозволить більш ефективно застосувати паралельне завантаження, так і відіграватись на його слабких сторонах за допомогою допоміжних засобів безпеки.

В нашому випадку основна задача полягає в створенні підсистеми захисту мережної інфраструктури підприємства, побудованої з використанням технологій "тонких" клієнтів. Для поставленої задачі ізолювання підмережі завантаження буде більш ніж достатньо. Це захиститись мережу від зовнішніх загроз, тобто надасть можливість правильно завантажуватися робочим станціям локальної мережі та отримувати доступ до Інтернету, забезпечувати правильне функціонування головної мережі з розподіленням трафіку по ній. Все ж таки більш важливою метою нашого завдання є реалізація забезпечення захищеного завантаження ОС по мережі.

РОЗДІЛ 2 ПРОЕКТУВАННЯ МЕРЕЖІ З МОЖЛИВІСТЮ ЗАХИЩЕНОГО ПІДКЛЮЧЕННЯ ДО МЕРЕЖІ ТА ЗАВАНТАЖУВАННЯ "ТОНКИХ" КЛІЄНТІВ

2.1 Принципи проектування мережі

В даній роботі було розглянуто питання забезпечення офісного простору підприємства, побудованого з використанням технологій "тонких" клієнтів. Було визначено підключення до загальної мережі офісу, із розділенням завантаження по VLAN підмережам.

Із міркувань безпеки наведених вище, цілком доцільно обрати варіант мережевого завантаження без монтування кореневої ФС вже існуючих офіційних релізів за допомогою засобів NFS. Зупинимось на варіанті з створенням власного предналаштованого образу ОС Linux Ubuntu 21.04.

З точки зору завантаження мережевою, процес завантаження розділятиметься самостійно, так як файли завантаження на сервері плануються ранзі, щоб вони не перетинались між собою та не заважали функціонуванню мережі. Це означає, що мережеве завантаження PXE не буде утворювати чергу – в паралель зможуть завантажуватись як однакові ОС с одного серверу, так і різнопланові ОС для розумних плат.

Дана підсистема захисту мережної інфраструктури підприємства, побудованої з використанням технологій "тонких" клієнтів складається з:

- Маршрутизаторі та комутаторів в межах офісного простору;
- Головного мережевого приладу, на якому налаштовуються DHCP-сервера, розділення офісу по VLAN підмережам, та немаловажним є налаштування переадресації на наступний сервер – TFTP;
- Хаб-серверу, який виступатиме окрім як місцем збереження усіх офісних даних та логів, буде відповідати у процесі завантаження також і за менеджмент: TFTP, NFS ,SMB;
- Бездискові машини передбачені у офісі, користувацькі ПК та розумні плати.

Обидва типи приладів, що завантажуються в мережі, будуть динамічно отримувати IP-адресу від головного мережевого приладу. Користувачі будуть розділятися на окремі VLAN-підмережі із міркувань захищеності та забезпечення безпеки роботи на просторах офісу.

Розробка топології та параметри адресації наведено в табл. 2.1.

Таблиця 2.1

Параметри адресації мережі

Модель пристрою Mikrotik	Ім'я пристрою	MAC-адреса	CIDR IP-адреса
RB4011iGS+5HacQ2HnD	NB_01_IOF_Ch1	08:55:31:8C:9A:07	10.100.1.1/24
RB2011UiAS-2HnD	NB_02_IWF_Ch1	08:55:31:50:8B:94	10.100.1.2/24
RB960PGS	SW_01_IBR	08:55:31:E9:5C:FA	10.100.1.6/24
CRS318-16P-2S+	SW_02_ILF	2C:C8:1B:48:4B:9C	10.100.1.4/24
CSS610-8G-2S+	SW_03_IOF	2C:C8:1B:48:B6:2A	10.100.1.3/24
CRS112-8P-4S	SW_04_IHF	08:55:31:7D:57:56	10.100.1.5/24
RBwAPG-5HacD2HnD	AP_01_ILL_Ch1	08:55:31:3D:9F:B4	10.100.1.12/24
RBwAPG-5HacD2HnD	AP_02_ILR_CH11	08:55:31:3D:C5:FF	10.100.1.14/24
RBwAPG-5HacD2HnD	AP_03_IHL_Ch1	08:55:31:3C:CD:74	10.100.1.15/24
RBwAPG-5HacD2HnD	AP_04_IOR_Ch6	08:55:31:8C:9A:04	10.100.1.13/24
RBcAPGi-5acD2nD	cAP_01_ILF_Ch6	2C:C8:1B:36:85:E8	10.100.1.10/24
RBcAPGi-5acD2nD	cAP_02_IHF_CH11	2C:C8:1B:36:6C:8A	10.100.1.11/24

Інформація наведена вище враховує особливості налаштування приладів Mikrotik. На приладах використовується безкласова маршрутизація – метод IP-адресації для більш комфортного управління мережею. Завдяки цьому методі комфортніше конфігурувати заплановані VLAN.

У таблиці 2.1 видно, що для більш детального конфігурування мережі у першому стовбці наведена модель мережевого пристрою – це зроблено для розуміння підтримки спільного функціоналу, щоб розуміти їх апаратні та програмні можливості для сумісності роботи в обмеженому офісному просторі.

У вибірці таблиці не описуються прилади, які не є мережевим обладнанням.

У назві приладів використовується сутність призначення обладнання по стандартам мережевої класифікації. Також використовуються в вікні назви кожного пристрою скорочення для розуміння розміщення на локації офісного простору. Присутні помітки у пристроїв, які мають вигляд *_CH* або *_Ch* – це зроблено для адміністрування каналів безпроводної мережі (Wi-Fi), який був реалізований за допомогою програмної утиліти Mikrotik – має назву CAPsMAN. Інструмент для налаштування безшовності безпроводної мережі.

2.2 Логічне проектування мережі

У цій схемі мережеві пристрій підтримуються два підключення до Інтернету. Також ще доступний третій WAN-порт, який поки не використовується через необхідність для резервування Інтернет каналу. Це зроблено для відмовостійкості роботи мережі з виходом в Інтернет та надання працівникам офісу стабільного підключення.

Створено попередній план схеми мережі офісного простору (рисунок 2.1):

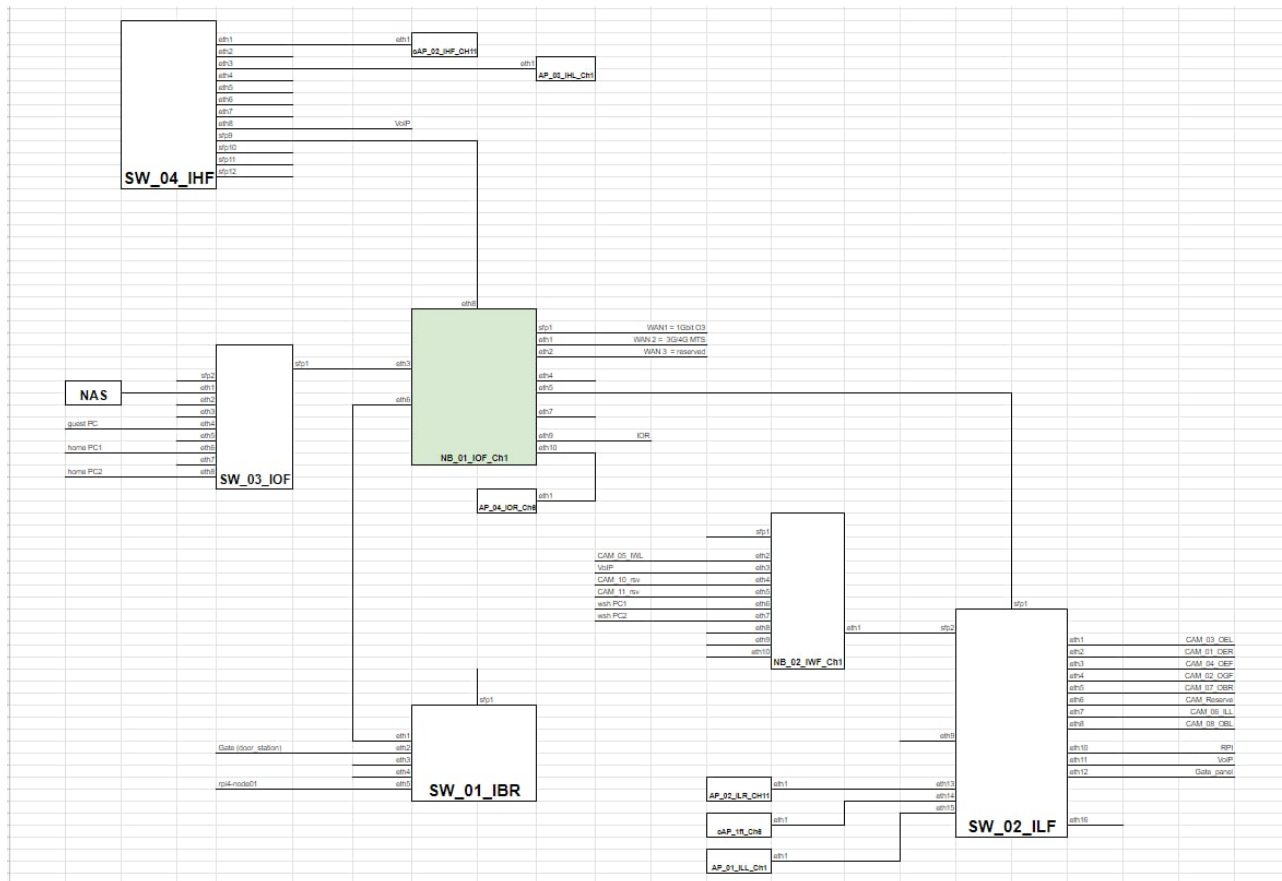


Рисунок 2.1. План схеми мережі побудований у Excel

В даній схемі буде наведений приклад офісного простору корпоративної мережі з використанням технологій "тонких" клієнтів. Питання захисту даного методу завантаження по мережі розглянуті, на практиці реалізуємо підвищення надійності завантаження ОС. Також необхідно забезпечити безпечне завантаження, яке унеможливить доступ до даних, що знаходяться локально на пристрої, що завантажуються. Процес завантаження буде зроблено таким чином, що користувач не повинен слідкувати за завантаженням та встановленням – це буде зроблено автоматично. Підтримка завантаженої ОС буде здійснюватися з хаб-сервера, на якому будуть створюватись користувачі з необхідними на те правами доступу, та кожному буде виділено область даних відповідно до користувача.

У процесі захищеного використання технологій "тонких" клієнтів, буде важливо забезпечити безперешкодне підключення та швидкий доступ до робочого простору, щоб була надана змога виконувати свою роботу не вражаючи локальні медіа, чи будь-які дані кожного з пристроїв, що завантажуються. Таке дослідження вимушує при виборі завантажуючої ОС звернути увагу на ФС цієї системи. Наша організація має містити, що найменше:

- DHCP-сервер (на бріджовому маршрутизаторі);
- TFTP-сервер (для налагодження первинного завантаження ОС та передачі файлів);
- NFS-сервер (на цьому етапі завантаження буде монтувати вже сформовану кореневу ФС, щоб не витратити час на її розгортання);
- "Тонкі" клієнти (завантажуванні ПК);
- Хаб-сервер (дата центр для зберігання файлів кореневої ФС та керування виділенням робочого простору клієнтів);
- Ethernet-підключення (фізичне підключення для такого завантаження необхідне та потребує пряме підключення до локальної мережі).

Головна небезпека при логічному проектуванні полягає в тому, що на практиці до вузла можливе підключення через пристрій та безперешкодне завантаження стороннього пристрої, а в гірших випадках замінити дані

первинного завантаження, тим самим можливий витік та пошкодження даних пристроїв, котрі завантажуються.

Щоб мінімізувати такі ризики, при проектуванні сплануємо розділення на декілька логічний підмереж VLAN мережу організації.

Розробка організування VLAN за логічними групами пристроїв в табл. 2.2.

Таблиця 2.2

Організування VLAN за логічними групами

Призначення віртуальної підмережі	Сутність віртуальної підмережі	Номер віртуальної підмережі
[MNG]	VLAN створений для конфігурування розділення віртуальних мереж між собою	VLAN10
[HOME]	VLAN створений для забезпечення постійної роботи простору керівництва	VLAN20
[OFFICE]	VLAN створений для забезпечення захищеного робочого простору	VLAN30
[MAIN-WIFI]	VLAN створений під вимоги розділення безпроводного покриття для робочого простору	VLAN40
[SECURITY]	VLAN створений для охорони, відеокамер, камера спостереження, панелей спостерігання та дверного дзвінка	VLAN50
[IOT]	VLAN створений для пристроїв Raspbery Pi та інші пристрої автоматизації	VLAN60
[GUEST-WIFI]	VLAN створений під вимоги розділення безпроводного покриття для гостьового простору	VLAN70
[SERVER]	VLAN створений для роботи з хаб-сервером Truenas Scale	VLAN80

Перед усіма налаштуваннями, на основному мережевому пристрої реалізовано організування VLAN за логічними групами пристроїв. Зображення наведено нижче (рисунок 2.2):

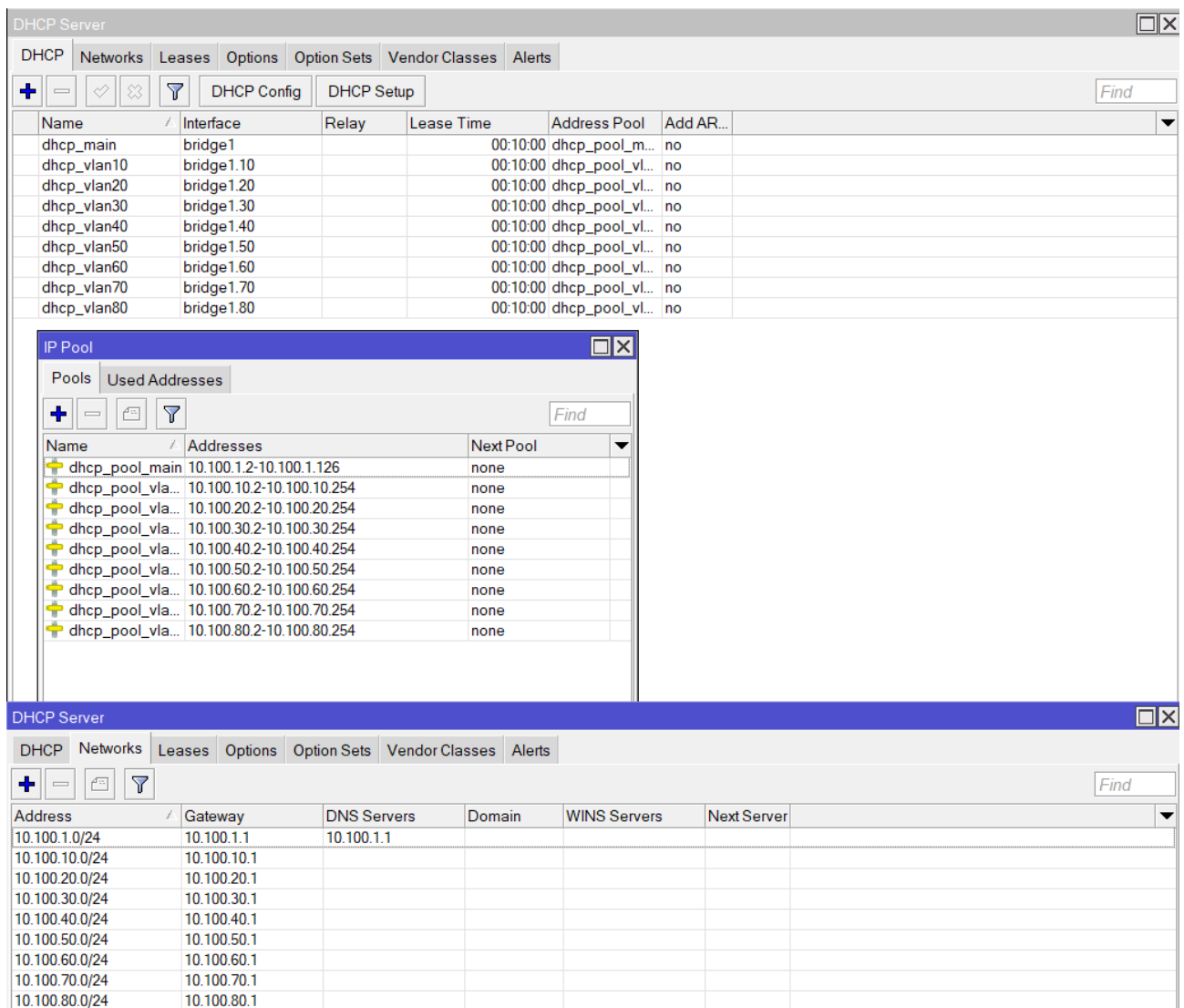


Рисунок 2.2. Організування VLAN за логічними групами пристроїв

2.3 Вибір обладнання та програмного забезпечення для організації роботи мережі

GNS3 (Graphical Network Simulator-3) - це графічний симулятор мереж, який надає можливість віртуально моделювати мережу з використанням маршрутизаторів, комутаторів та віртуальних машин. Безперечно сумісність з нашим вибором ОС – Linux Ubuntu. Можлива побудова комплексних проектів, що складаються з Cisco маршрутизаторів, та обладнанням Mikrotik, а також серверів під управлінням мережевих ОС. При відсутності можливості отримати доступ до проведення необхідних нам налаштувань на реальному обладнанні, GNS3 практично відтворює умови повноцінної лабораторії. Найважливішою з

особливостей GNS3 для нашого завдання є можливість з'єднання проектованої топології з реальною мережею, що надає дійсний доступ до нашого дискового простору через NFS, та можливість обміну з TFTP-сервером для кожного автономного робочого місця організації. Це дає можливість перевірити на практиці будь-який проект, без використання реального обладнання [16]. Нижче зображення програмного забезпечення емуляції GNS3 (рисунок 2.3):



Рисунок 2.3. Емулятор мережевого ПЗ GNS3

При виборі моделі маршрутизатора потрібно враховувати, що він виступатиме у ролі шаблонного майже для усіх шаблонних об'ємів компаній, через дуже специфічну конфігурацію, тому на нього може перепастися великий об'єм трафіку, який проходитиме з усіх користувачів через обмін даними з віртуальною мережею у якій для захищеності буде знаходитися сервер.

Основним критерієм вибору моделі маршрутизатору - була кількість портів, щоб була можливість підключити велику кількість робочих місць не обмежуючи працездатність обладнання, та можливість підтримки захисту на рівні як комутації, так і маршрутизації, не перешкоджаючи швидкості мережі. Оскільки на офісному просторі відкрите питання зі швидкістю роботи мережі для забезпечення відмовостійкості праці та завантажень по мережі, то даний вибір маршрутизатора також враховує швидкість передачі даних по його портам, та наявність декількох WAN-портів для двох або більше ISP. Це послугує налагодженню відмовостійкого Інтернет каналу, шляхом резервування й

автоматичного перемикання на іншого провайдера, якщо перший впав по будь-якій причині.

Наведене нижче зображення відповідного обладнання, котре покриватиме всі наші цілі у мінімумі, та з характеристика у запасі при масштабуванні (рисунок 2.4):



Рисунок 2.4. Маршрутизатор Mikrotik RB4011

Mikrotiks RB4011 використовує дивовижно потужну чотирьохядерну мікросхему Corte A15 від лабораторії Annapurna, таку ж саме, як і в попередника RB1100AHx4. ЦП підтримує апаратне прискорення IPSec, є 1 ГБ ОЗУ, тому цей пристрій буде легко справлятися з будь-якими завданнями, які ви створите для виконання RouterOS. У доповненні до десяти 1 Гбіт/с Ethernet-портів є порт SFP+, який забезпечує додатковий зв'язок 10 Гбіт/с через модуль SFP+ (не входить у комплект), але про можливості розширення не забуваємо. RB4011 може забезпечувати вихід PoE на порт №10 і оснащений компактним і доволі професійно зовні виглядаючим корпусом з твердого металу в матовому чорному кольорі. На 1 порт є підтримка PoE In в режимі Passive PoE з напругою 18-57 В.

Буде в нагоді при використанні точко доступу для продовження сигналу та більшого покриття офісу організації [17].

Ці характеристики залишатимуться актуальними при реалізації IoT компонентів на такому просторі. Дуже сучасне розширення, яке знадобиться для майбутніх плат Raspberry Pi 4.

Майже всі пристрої офісу корпоративної мережі працюватимуть як віртуальні машини, запущені на VirtualBox. Тому перш ніж закупати обладнання, треба перевірити усі заплановані компоненти проекту на сумісність, щоб запобігти неочікуваних та незапланованих проблем із майбутнього. Таке рішення послугує гарним кроком для розподілу фінансів та попереднього перегляду ризиків.

Oracle VM VirtualBox - програмний продукт віртуалізації для ОС, що призначений для створення віртуальної машини з параметрами справжнього комп'ютеру при відсутності можливості запуску на реальному обладнанні. На VB можна запустити будь-яку ОС при наявності образів. ПЗ встановлюється на дійсну ОС, а у середину неї встановлюються гостьові ОС, які запускаються як сеанс системи у системі [18]. Зображення інструменту віртуалізації наведено на рисунку (рисунок 2.5):



Рисунок 2.5. ПЗ віртуалізації VirtualBox

Основне обладнання та інструменти було обрано, тому можемо перейти до ОС, яка слугуватиме основним робочим простором близьким до середовища розробки даної організації, та при масштабуванні залишатиметься універсальною. З урахуванням подальшої гетерогенності мережі було обрано ОС:

Linux Ubuntu – це дистрибутив, розроблений компанією Canonical, на теперішній час він є одним із самих популярних дистрибутивів Linux, зокрема даний реліз вважається самим стабільним для роботи, який було обрано для організації. Його перевагами є простота в освоєнні та роботі з ним. Вважається, що цей дистрибутив є найкращим для новачків, котрі тільки починають працювати з Linux із сімейства Unix-подібних.

Перш ніж почати створювати власний образ, необхідно краще познайомитись із цією ОС. Цей дистрибутив Linux був розроблений на основі ядра Linux, в свою чергу є Unix-подібною системою, створена Лінусом Торвальдсом у 1991 році. Зазвичай дистрибутиви Linux розробляються з відкритим вихідним кодом, та є безкоштовними. Один із ключових критеріїв, через який ця ОС отримала визнання користувачів та залишається досі у фаворитах для розробників ПЗ [19].

Canonical стабільно підтримує свої ОС, та випускають новий реліз кожних 6 місяці – один реліз у квітні та один у жовтні. Стабільно кожних 2 роки компанія переводить один з цих релізів на довготривалу підтримку LTS (Long Term Support). Такі випуски в свою чергу підтримуються протягом 5 років з моменту переведення в LTS, після цього вони вважатимуться більш корпоративним системами для використання у корпоративних сегментах. Інші випуски, котрі не перейшли на довготривалу підтримку від Canonical – призначені для користувачів, яким не так важлива стабільність, а є більш важливим огляд нового функціоналу та інструментів. Перший варіант підходить під наші цілі, плюсом ще послугує те, що корпоративні ОС облежені – цілком підходить для кастомізації та створення власної ОС на базі Ubuntu 21.04.

Під час огляду релізу Ubuntu 21.04 (Hirsute Hippo), справедливим буде виділити їх стабільність, проміжні релізи доволі стабільні в свою чергу. Оскільки проміжні релізи використовуються в якості основних ОС для забезпечення та підтримки роботи ПЗ, то основні функції будуть очікуватися при виході LTS релізу [20]. Зображення робочого простору ОС Linux Ubuntu 21.04 буде наведено на рисунку (рисунок 2.6):

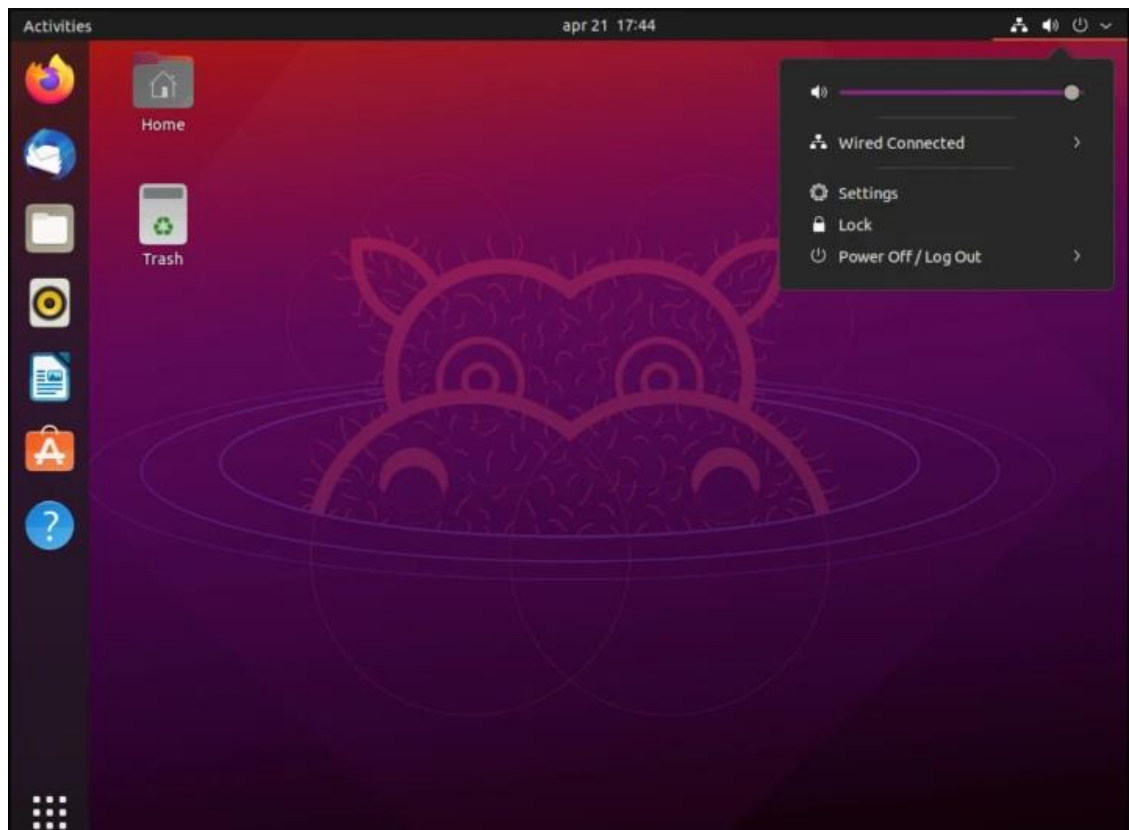


Рисунок 2.6. Робочий простір ОС Linux Ubuntu

Залишилося обрати сервер для офісу та саму стойку для серверу, з можливістю подальшого розширення пам'яті на сервері.

Приглянулося вдале рішення для серверу на базі ПЗ TrueNAS Scale версія релізу 21.08 є досить стабільною. Даний сервер реалізований на базі вдалого консольного середовища Linux Debian. TrueNAS послідовник FreeNAS – вільної ОС для мережевого сховища на базі середовища FreeBSD [21].

Для переміщення файлів завантаження PXE на сервер, на який буде переадресація з DHCP-серверу на TFTP-сервер, можна скористатися утилітою SCP (Secure Copy) або програмою, яка надає можливість зробити це через зручний UI – FileZilla (рисунок 2.7):

					КБ.КВ.Б – 125 – 22– ПЗ	
						35

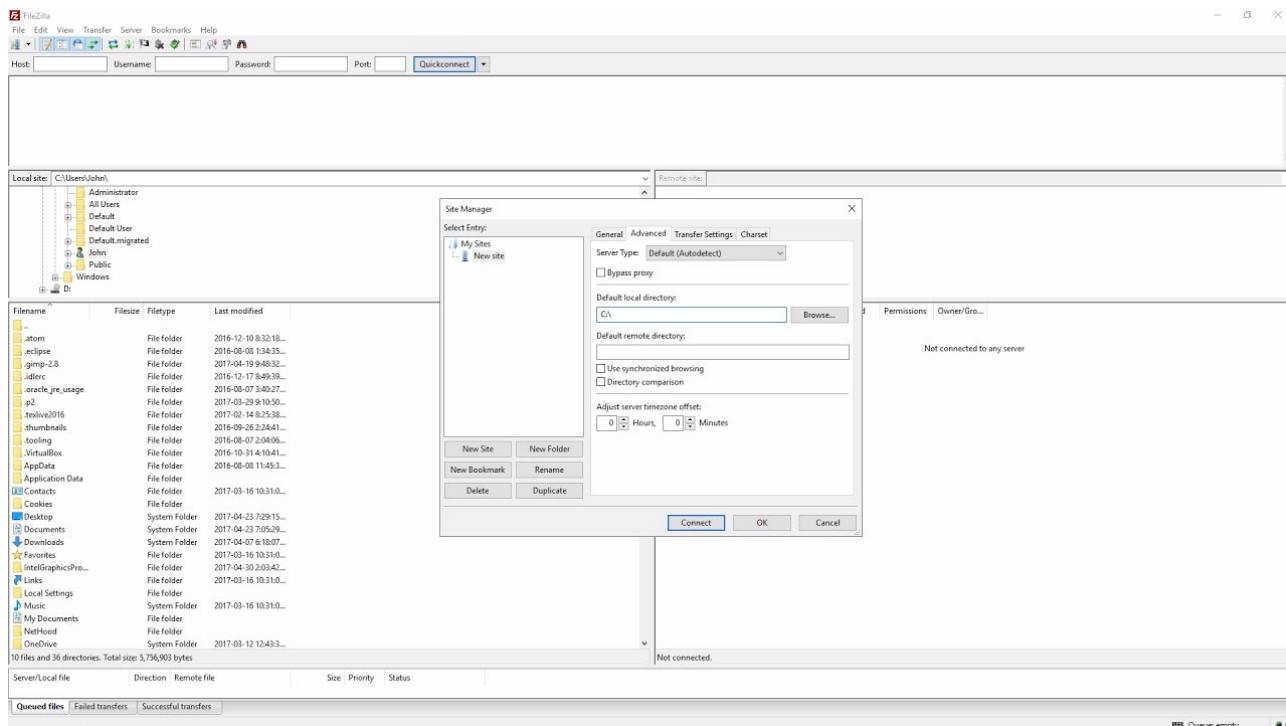


Рисунок 2.7. ПЗ для керування файлами на сервері FileZilla

Висновки до другого розділу

У другому розділі було більш детально спроектовано організацію офісного простору інфраструктури підприємства, побудованої з використанням технологій "тонких" клієнтів, з розташування мережевого обладнання та принципами ізолювання даних та мережевого трафіку від робочих станцій та плат, які плануємо завантажувати через мережу. Сформовано таблиці параметрів адресації до відповідного обладнання та організування VLAN-підмереж за логічними групами.

Обґрунтовано вибір ПЗ та обладнання з функціоналом для реалізації мережевого завантаження ОС у практичній частині. Визначені образи, на базі яких будуть виконуватися завантаження та розподілення відповідно до ПК чи розумних плат. Визначилися зі складом пристроїв інфраструктури підприємства.

Сформоване приблизне уявлення про необхідні налаштування на кожному з пристроїв мережі, які беруть участь у процесі мережевого завантаження ОС. Вибір залежав від того, щоб на всіх пристроях був доступний необхідний функціонал для реалізації поставленого завдання дипломного проекту.

РОЗДІЛ 3 НАЛАГОДЖЕННЯ ПІДСИСТЕМИ ЗАХИСТУ МЕРЕЖНОЇ ІНФРАСТРУКТУРИ ПІДПРИЄМСТВА, ПОБУДОВАНОЇ З ВИКОРИСТАННЯМ ТЕХНОЛОГІЙ "ТОНКИХ" КЛІЄНТІВ

3.1 Налаштування мережі та підготовка інфраструктури підприємства, побудованої з використанням технологій "тонких" клієнтів

Почнемо з того, що повністю збиратимемо схему, яку було спроектовано у попередньому розділі, та будемо поступово налаштовувати кожен з її ділянок, починаючи з базових налаштувань приладів. Фінальне завдання - повністю працездатна мережа згідно вимог до поставленого завдання магістерської дипломної роботи.

Спершу проведено базові налаштування маршрутизатору, який виступатиме у ролі DHCP-серверу та переадресовувати на TFTP-сервер: стандартне ім'я змінимо на відповідне до вимог проекту та організації, інтерфейси сконфігуровані належним чином та видалені IP-адреси по DHCP для інших приладів мережі. Також всіх необхідні налаштування виконано на комутуючих та маршрутизуючих пристроях підприємства.

Скористаємося одним інструментом, а саме **Cubic**, котрий використовувався лише в рамках цього підрозділа роботи. Одним з таких наших інструментів стане саме та предналаштована Linux Ubuntu ОС.

Однією з основних переваг Linux є те, що ви можете налаштувати його так, як вам подобається. Ви можете додати нову функцію, видалити існуючу функцію, змінити зовнішній вигляд дистрибутива Linux і багато іншого. Ви можете налаштувати вже встановлений дистрибутив Linux або створити новий налаштований образ із ISO Linux. У цьому посібнику ми дізнаємося, що таке Cubic і як створити власний живий ISO-образ Ubuntu за допомогою програми Cubic.

Cubic - Custom Ubuntu ISO Creator – являє собою безкоштовну графічну програму з відкритим вихідним кодом для створення налаштованого завантажувального образу Ubuntu Live CD (ISO).

Cubіc дозволяє легко та без зусиль створювати налаштований живий компакт-диск Ubuntu. Він має інтегроване середовище командного рядка chroot, де ви можете виконувати всі налаштування, як зазначено нижче [22]:

- Встановити нові пакети;
- Видалити наявні пакунки;
- Встановіть додаткові ядра;
- Додайте більше фонових шпалер;
- Додати файли та папки;
- Змінити існуючий ISO-образу;
- І багато іншого функціоналу.

Коли у наше середовище встановлено Cubіc, обираємо яка сама версія Linux Ubuntu перекриває усі вимоги для поставленої задачі. В попередньому розділі вже визначились із Ubuntu 21.04.

По-перше, необхідно створювати користувачів відразу у ISO-образі, щоб надалі не виникали питання з безпекою та доступністю до даних на сервері.

По-друге, як раз таки необхідно відразу у ISO-образі вмонтовувати відповідний простір хаб-серверу для роботи до створених користувачів.

Для роботи з ПЗ Cubіc необхідно обрати папку для збереження нового збудованого ISO файлу. Далі перейдемо до вибору офіційного ISO-образу, котрий обрали з офіційного сайту Canonical, так як в ньому буде впевненість, що він не нашкодить та буде безпечним для організації.

Наступний перехід нас перекине на командне середовище, де реалізуються подальші налаштування ОС, утиліти Chroot (change root) – викликає команди відносно неіснуючого досі newroot. Значення всіх наступних стандартних шляхів по директоріям - для “ / ” (косих рисочок) в найменуваннях шляхів змінюється на newroot для команд дочірніх процесів. Після завершення новий початковий робочий каталог – newroot [23].

На останніх етапах обираються системні пакунки та стискається ФС для ISO. Більш детальніше із зображеннями створення користувацької ОС – нижче:

1. Визначаємося з ієрархією папок на сервері, та обираємо каталог проекту для зберігання користувацького живого ISO-образу Ubuntu 21.04 (рисунк 3.1):

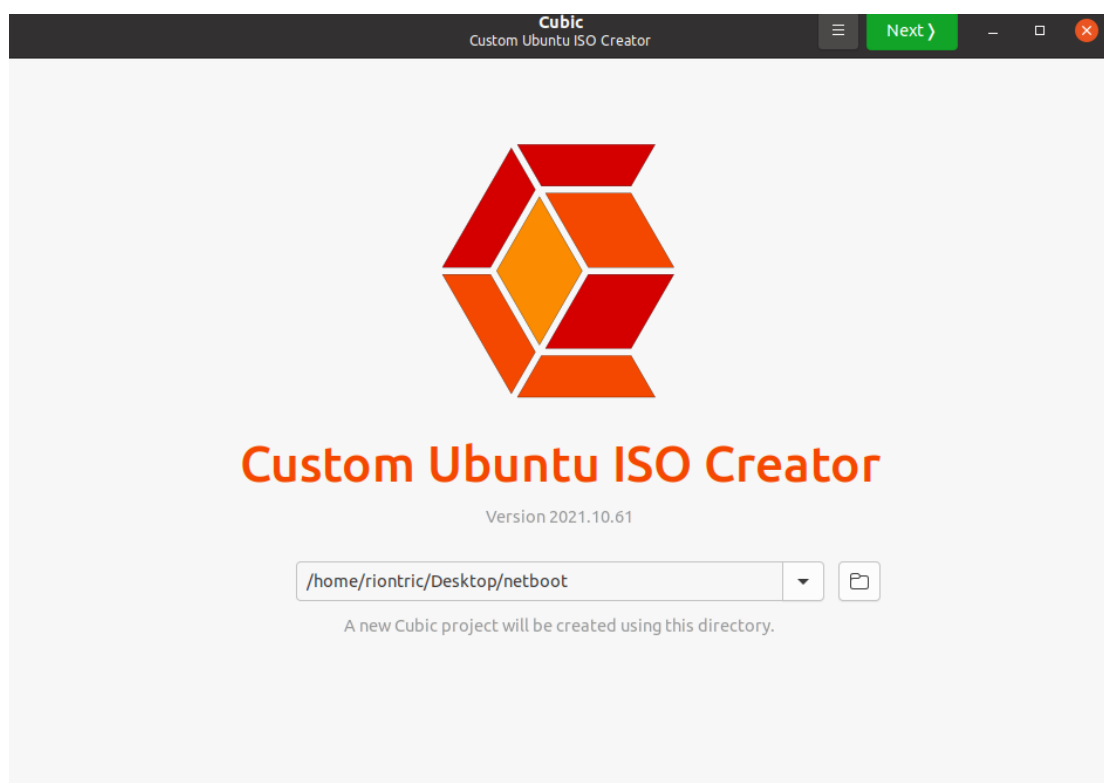


Рисунок 3.1. Точний шлях для створення власної ОС за допомогою ПЗ Cubic

2. Далі обираємо підготовлений ISO-образ Ubuntu 21.04, котрий отримано з офіційного сайту Canonical, для налаштування власної ОС (рисунок 3.2):

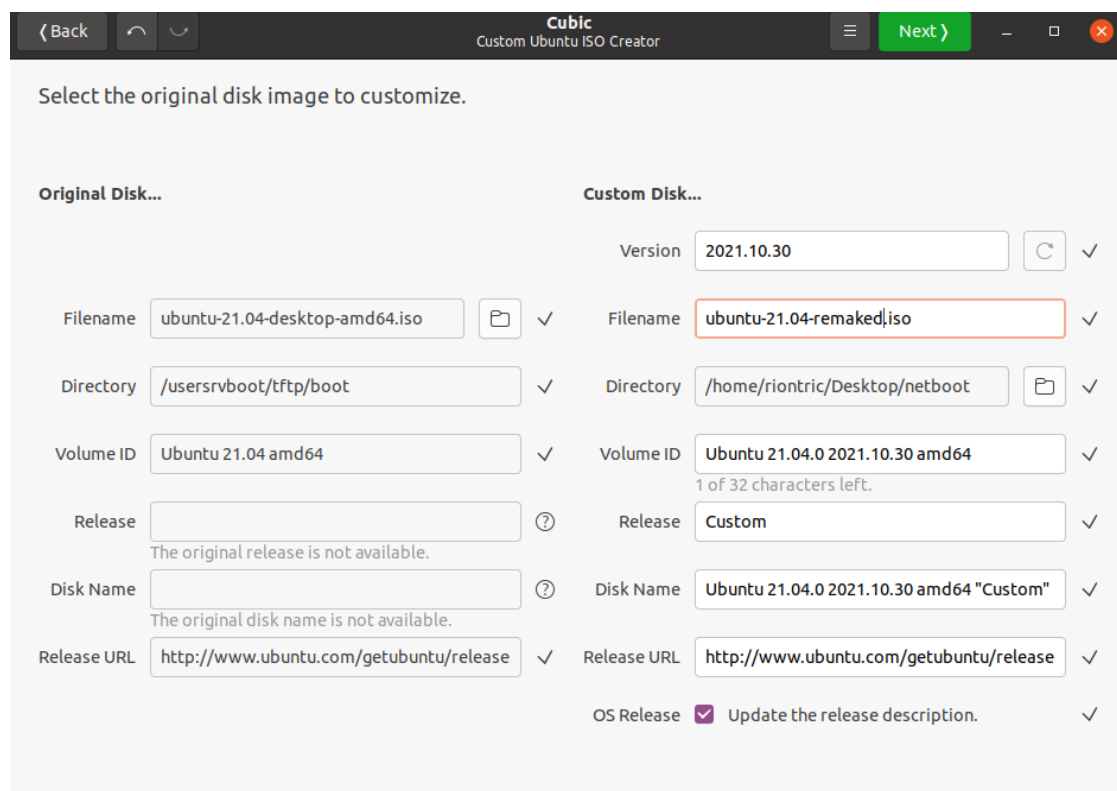


Рисунок 3.2. Обираємо підготовлений образ у ПЗ Cubic

3. Cubic дозволяє легко створювати налаштований живий компакт-диск для офіційних релізів Ubuntu. Він має інтегроване середовище командного рядка Chroot (рисунк 3.3), де виконувались всі наступні налаштування на цьому етапі на зображеннях (рисунк 3.4 – 3.8):

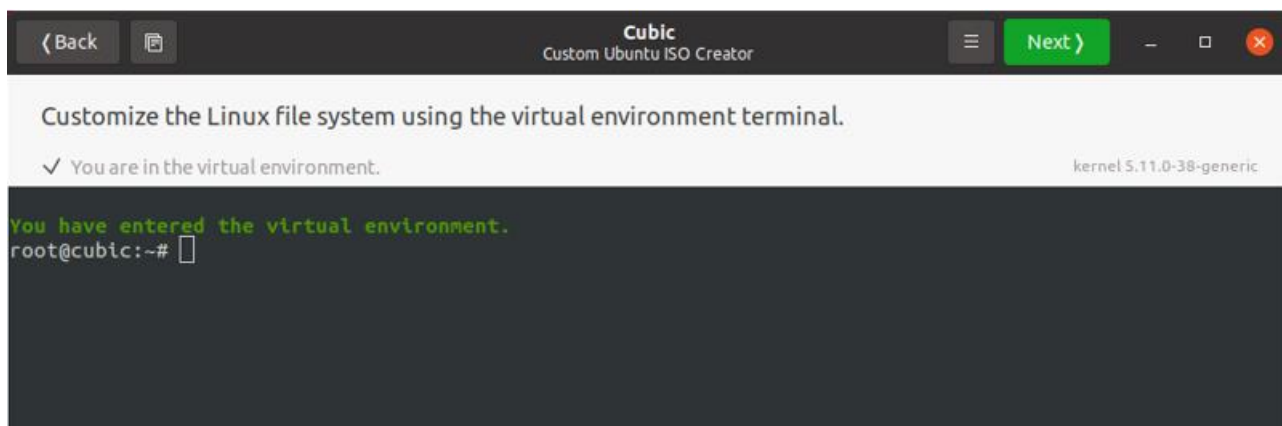


Рисунок 3.3. Середовище Chroot командної строки у ПЗ Cubic

- 3.1. Вимкнення авто-входу та авто-логіну користувачів у системі. Функція автоматичного входу в живій ОС схоже на завантаження зльоту. Після вилучення *initrd.img* необхідно відредагувати скрипт ініціалізації ОС, і слідом відновити файл *initrd.img* замінив при цьому його оригінал, котрий був до того. Щоб відключити для користувача авто-логін – необхідно із скрипту видалити \$USERNAME (ім'я користувача), а на цьому місці залити пусті лапки. Створиться новий файл *initrd.img* без автоматичного входу (рисунк 3.4). Потім можете продовжити майстерінг ОС [24]:

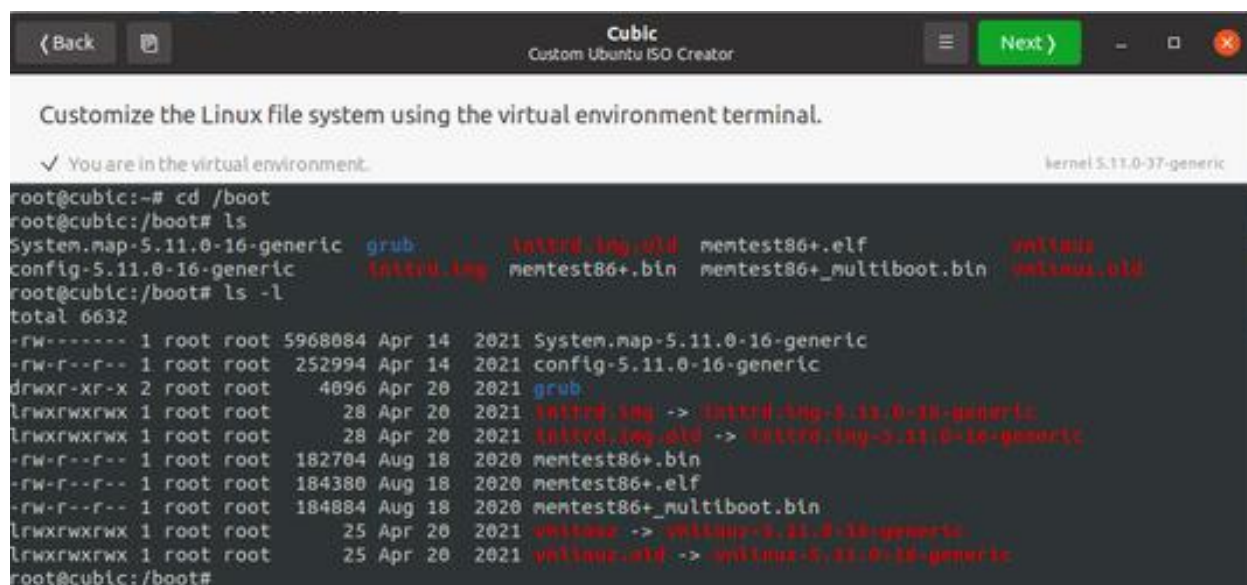


Рисунок 3.4. Вимкнення авто-входу в ОС за допомогою ПЗ Cubic

- 3.2. Далі створюється власний користувач за замовчуванням та відповідні облікові дані (рисунок 3.5):

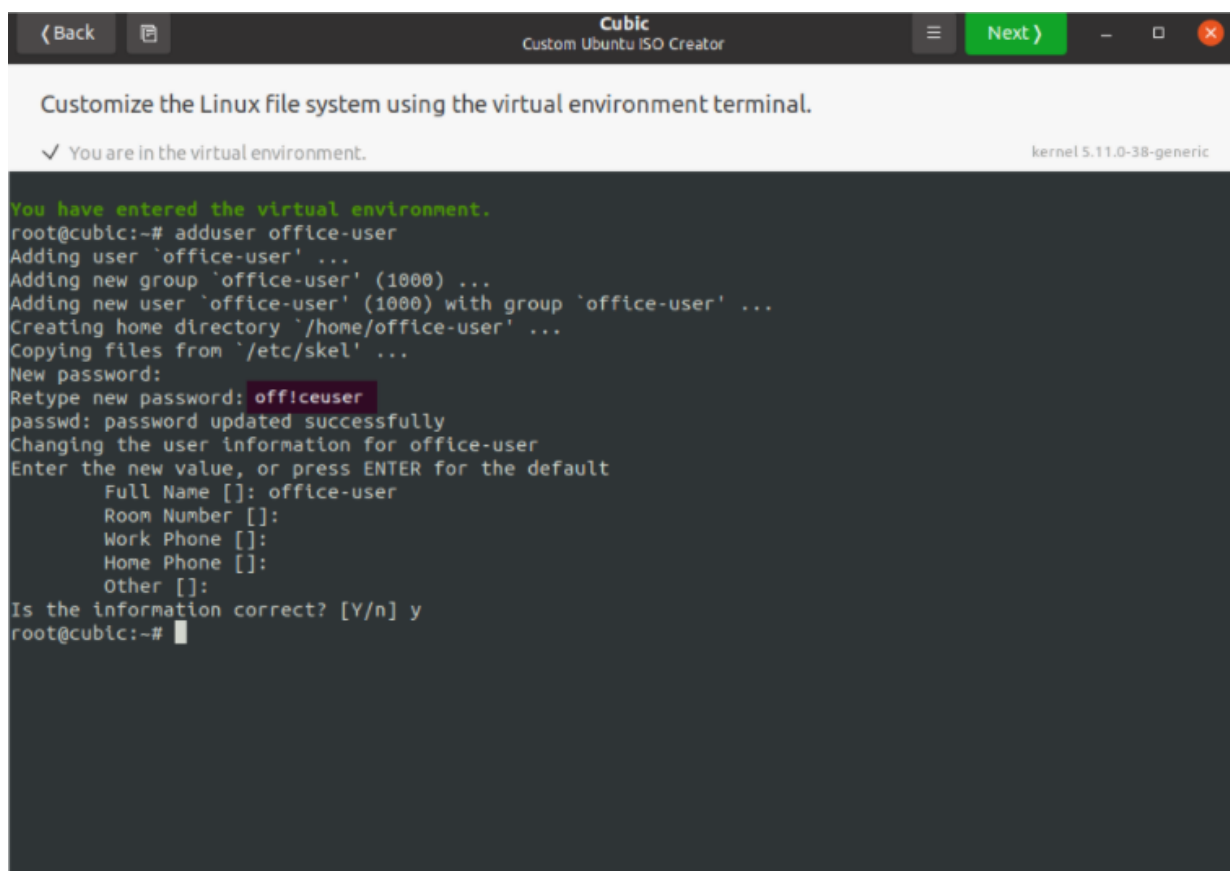


Рисунок 3.5. Створення користувача за замовчуванням у ПЗ Cubic

- 3.3. Наступник кроком даного етапу виконувалось автоматичне монтування SMB-ресурсів для спільного доступу робочого простору всередині командного рядка утиліти Chroot живого компакт-диску ОС Ubuntu. Сховище даних для обміну між робочими станціями об'єднаний хаб-сервером у приміщенні підприємства організації (рисунок 3.6):

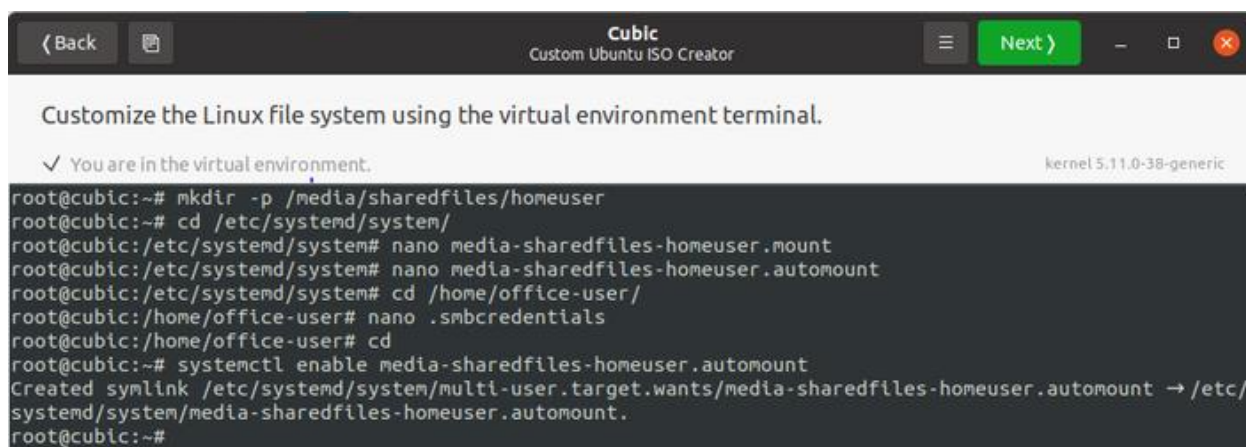
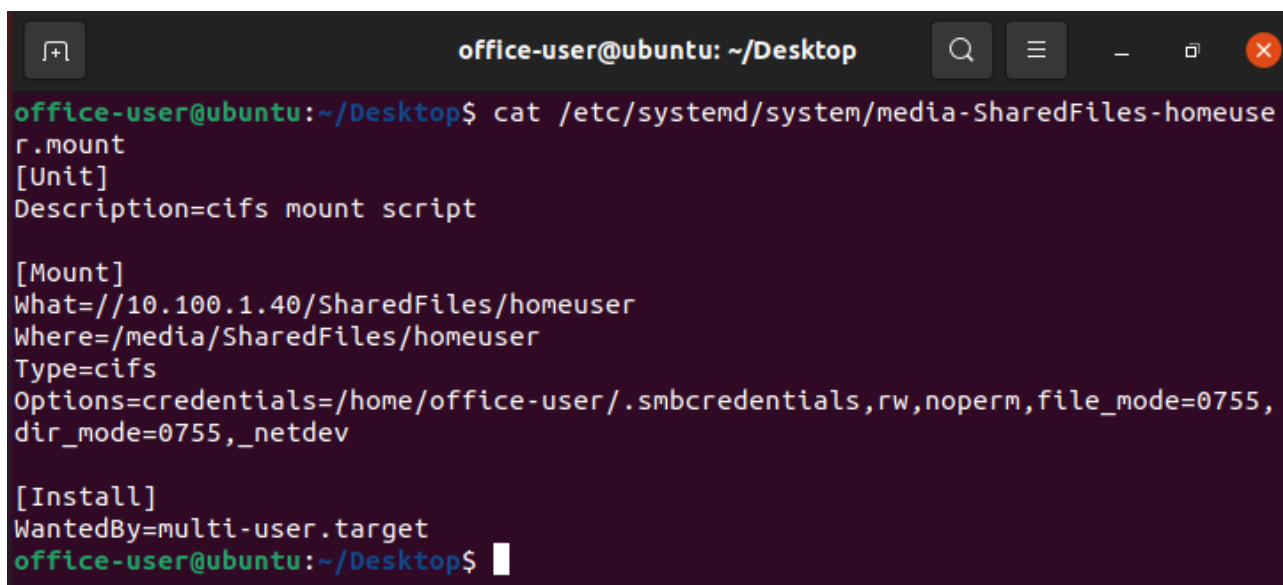


Рисунок 3.6. Монтування SMB у ОС за допомогою ПЗ Cubic

Для робочих ПК довелося провести налаштування автоматичного монтування за допомогою демону *systemd*. Налаштування схожі на *autofs*, але позбавляє від проблем, які виникатимуть при роботі з цим.

На сервері проведені налаштування для спільних папок через ресурси SMB, які вказуватимуть на мережеву папку. До цієї папки нема потреби підключатися кожного разу при роботі в просторі підприємства, доцільніше було б налаштувати один раз автоматичне монтування [25].

Для цього створюється файл "*media-sharedfiles-homeuser.mount*" для процесу ручного монтування (синтаксис: головні риси "/" видаляються, та замінюються на "-") в *"/etc/systemd/system/"*. Виконані налаштування наведені нижче на зображенні (рисунк 3.7):



```
office-user@ubuntu: ~/Desktop
office-user@ubuntu:~/Desktop$ cat /etc/systemd/system/media-sharedfiles-homeuser.mount
[Unit]
Description=cifs mount script

[Mount]
What=//10.100.1.40/SharedFiles/homeuser
Where=/media/SharedFiles/homeuser
Type=cifs
Options=credentials=/home/office-user/.smbcredentials,rw,noperm,file_mode=0755,dir_mode=0755,_netdev

[Install]
WantedBy=multi-user.target
office-user@ubuntu:~/Desktop$
```

Рисунок 3.7 – Монтування мережевих папок на ПК

Далі для зручності було налагоджено процес автоматичного монтування. Для цього також створюється файл в просторі демону *systemd* - створюється файл "*media-sharedfiles-homeuser.automount*" для процесу ручного монтування (з аналогічний до попереднього синтаксисом) в *"/etc/systemd/system/"*. Виконані налаштування також наведені нижче на зображенні (рисунк 3.8):

```

office-user@ubuntu: ~/Desktop
office-user@ubuntu:~/Desktop$ cat /etc/systemd/system/media-SharedFiles-homeuse
r.automount
[Unit]
Description=cifs automount script

[Automount]
Where=/media/SharedFiles/homeuser

[Install]
WantedBy=multi-user.target
office-user@ubuntu:~/Desktop$

```

Рисунок 3.8 – Автоматичне монтування мережових папок на ПК

Після цього залишається тільки запустити останній створений файл монтування через утиліту *systemctl*, щоб кожного разу відпрацьовував автозапуск демону *systemd*.

4. На наступному кроці організації доступна можливість обрати необхідні пакунки та неважливі для робочих ПК, які будуть видалені після типової або мінімальної інсталяції. Із аналізу цілей поставлених організацією підприємства, узгоджений список та обрали ПЗ для встановлення, яке необхідно видалити з користувацького ISO-образу (рисунок 3.9):

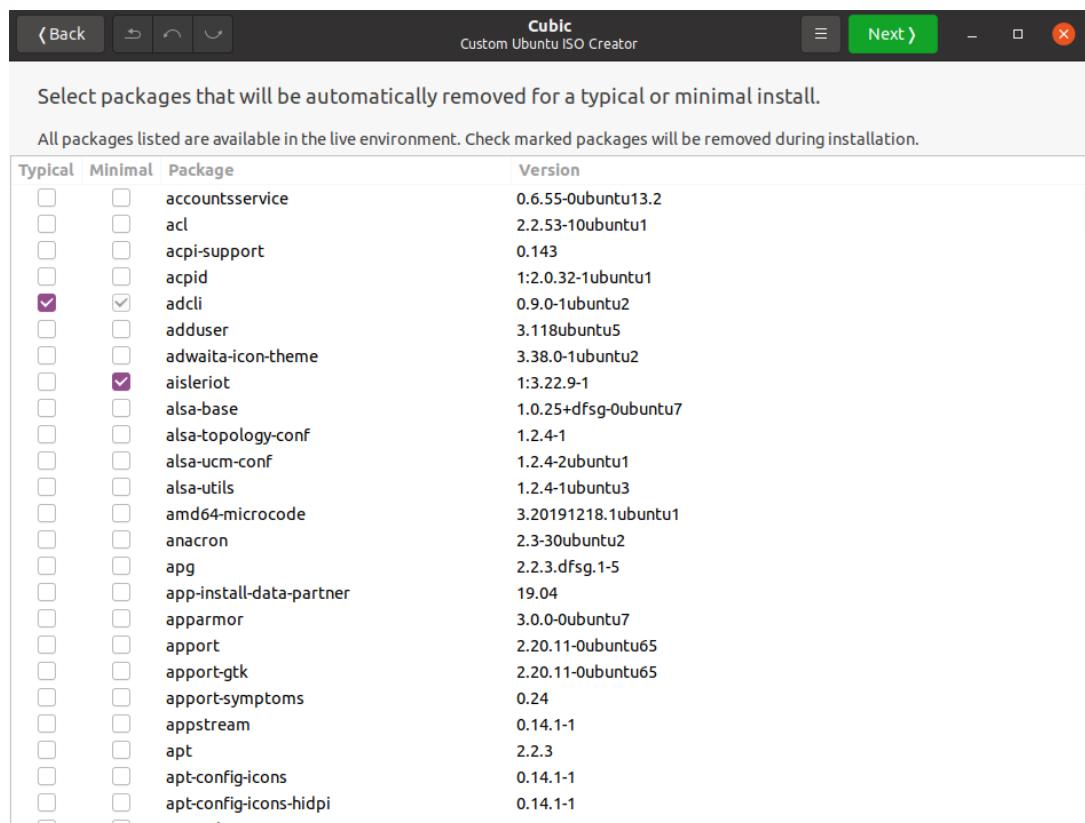


Рисунок 3.9. Урізання розмірів ОС шляхом пакунків у ПЗ Cubic

5. На передостанньому кроці реалізований майстер конфігурування ядра ISO, Preseed (файл попередньої конфігурації) та налаштування ISO завантаження для кастумізації нашого ISO (рисунок 3.10):

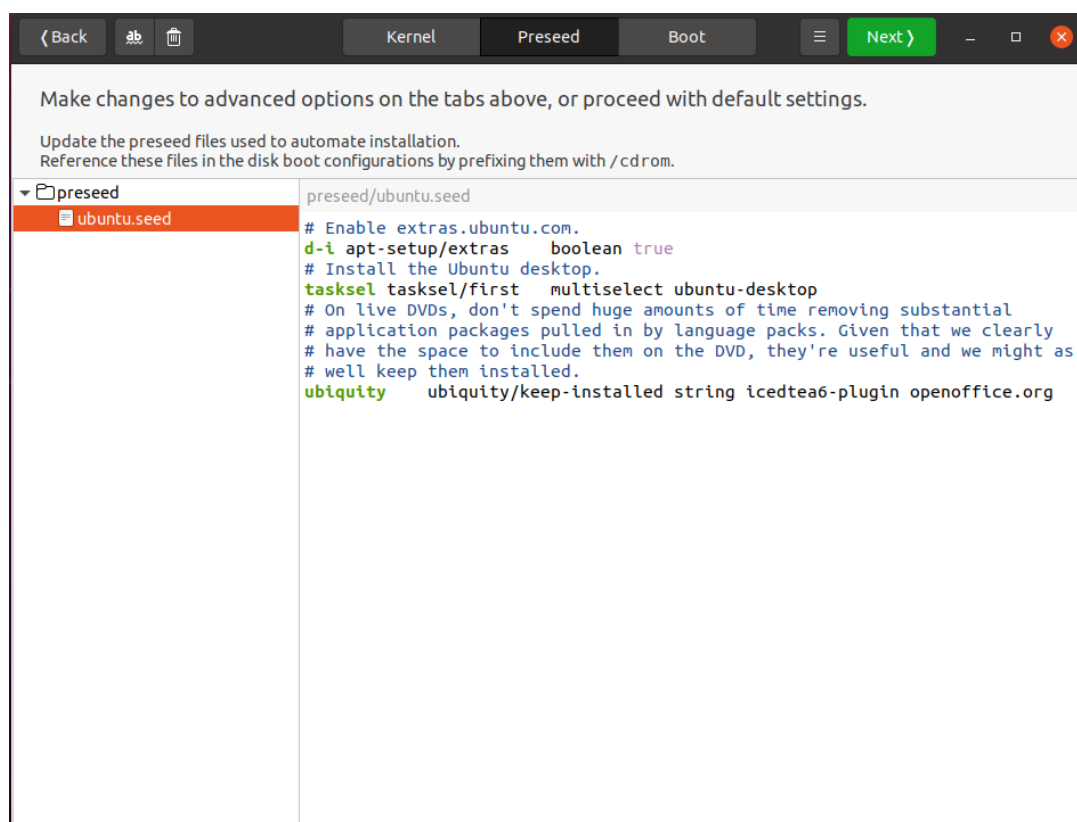


Рисунок 3.10. Попередня конфігурація ядра та завантаження у ПЗ Cubic

Обирається ядро, яке буде використовуватись для завантаження налаштованого Live CD (живого компакт-диску) Ubuntu ISO-образу. Слідом оновлення, додавання, чи видалення Preseed – використовується для автоматизації встановлення ОС.

Якщо використовується попереднє заповнення *initrd*, треба перевірити, що файл з найменуванням *preseed.cfg* наявний у кореневому каталозі файлу *initrd*. Мережеве завантаження автоматично перевірить наявність цього файлу та скачає його при наявності [26].

І на останній вкладці цього меню - завантаження ISO, де реалізована можливість налаштовувати пункти меню завантаження.

6. На останньому кроці обираємо алгоритм стиснення для ФС Linux. За замовчуванням ПЗ Cubic використовує стискання gzip. Додатковими методами стискання є lz4, lzo, zstd, lzma та xz (рисунок 3.11):

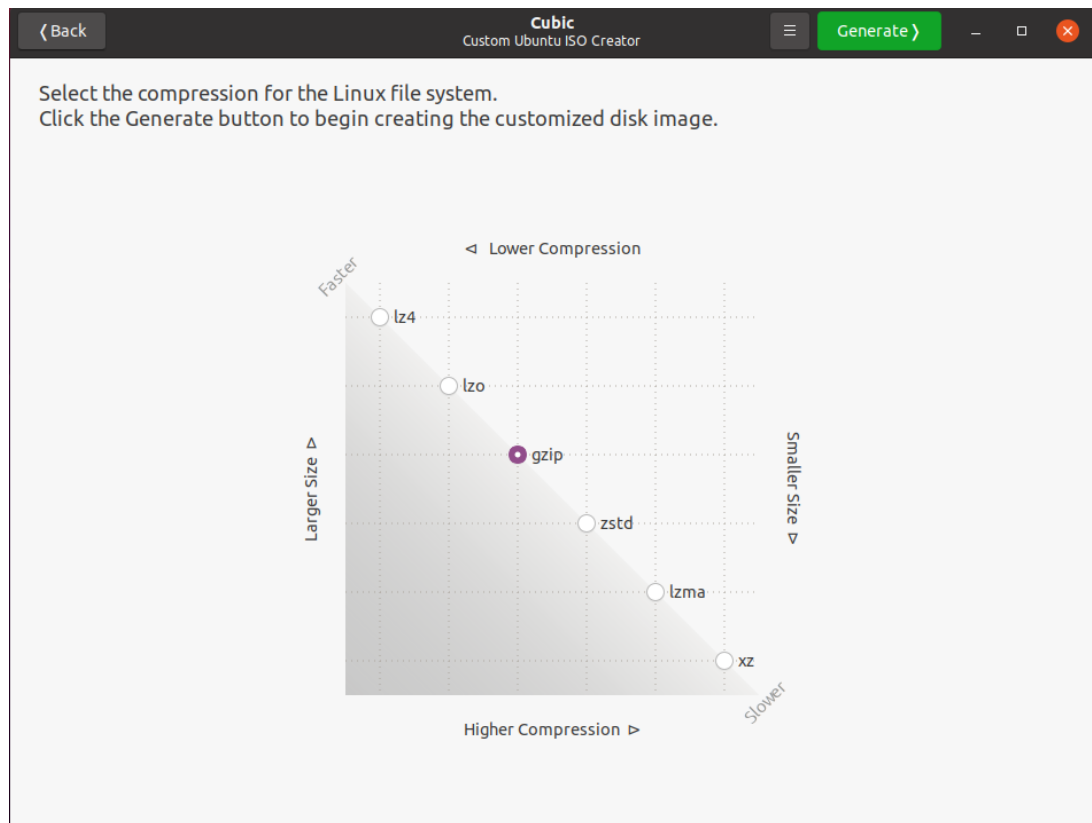


Рисунок 3.11. Стиснення ISO-образу за допомогою ПЗ Cubic

При виборі нижніх алгоритмів стиснення із верхньої частини графіку – процес стиснення буде виконаний швидше, але розмір такого ISO буде більшим.

При виборі більш високих алгоритмів стиснення, що нижче на графіку – створення такого образу займе більше часу, але вже розміри будуть значно менші такого ISO.

По завершенню Cubic виводить деталі новоствореного та налаштованого ISO-образу. Налаштований образ буде збережено до каталогу проекту Cubic.

Скористаємося ще одним інструментом для отримання образу завантаження розумних плат – **Raspberry Pi Imager**.

1. Встановіть Raspberry Pi OS за допомогою Raspberry Pi Imager. Вставте SD-карту, яку ви використовуватимете з Raspberry Pi, у зчитувач і запусіть Raspberry Pi Imager. Наступні кроки цього етапу зображені на рисунках 3.12 – 3.18.

- 1.1. Відкриваємо вікно Raspberry Pi Imager (рисунок 3.12):

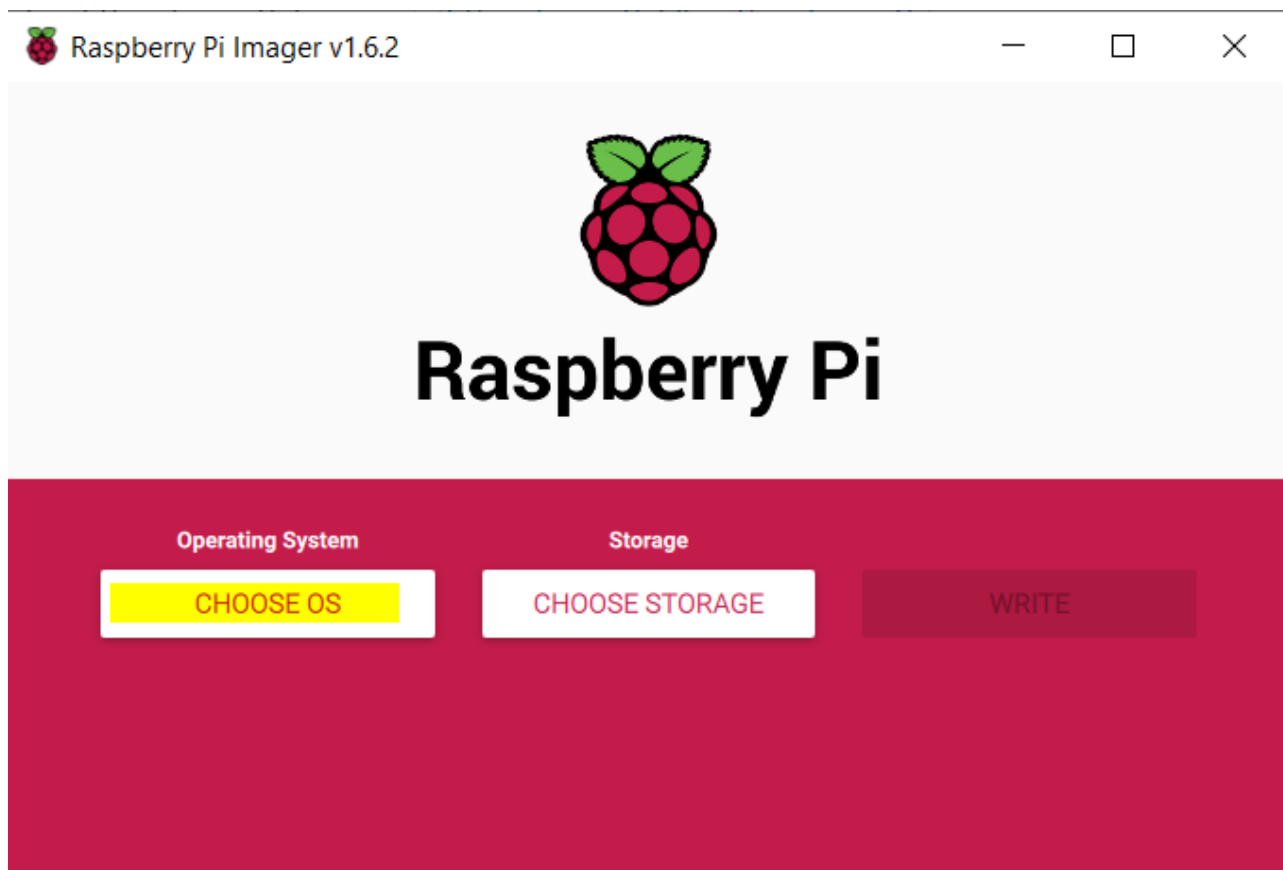


Рисунок 3.12. – Вікно вибору ОС у ПЗ Raspberry Pi Imager
1.2. Вибір ОС Raspberry для завантаження через SD-карту (рисунок 3.13):

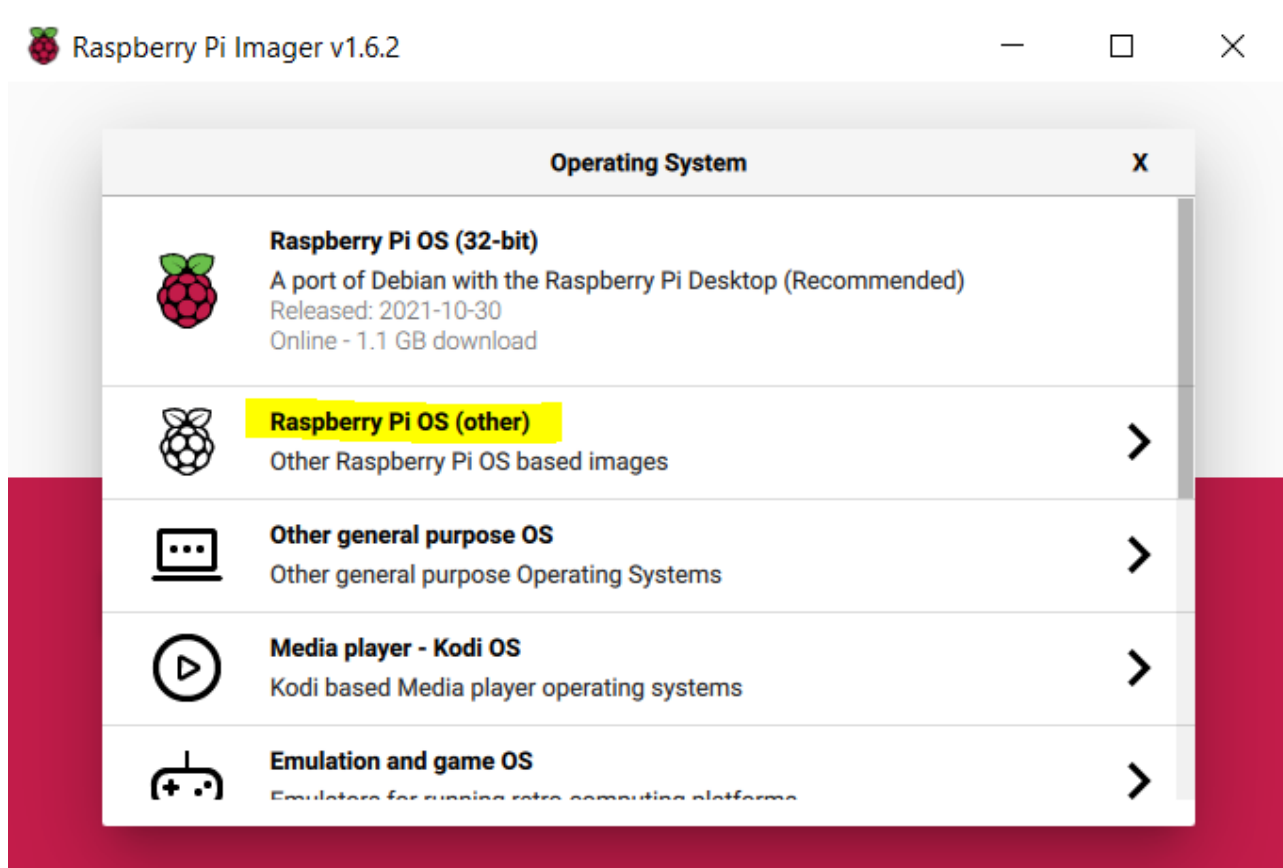


Рисунок 3.13. Розширений вибір ОС у ПЗ Raspberry Pi Imager

1.3. Вибір Raspberry Pi OS Lite (32-bit). Порт Debian без робочого середовища (рисунок 3.14):

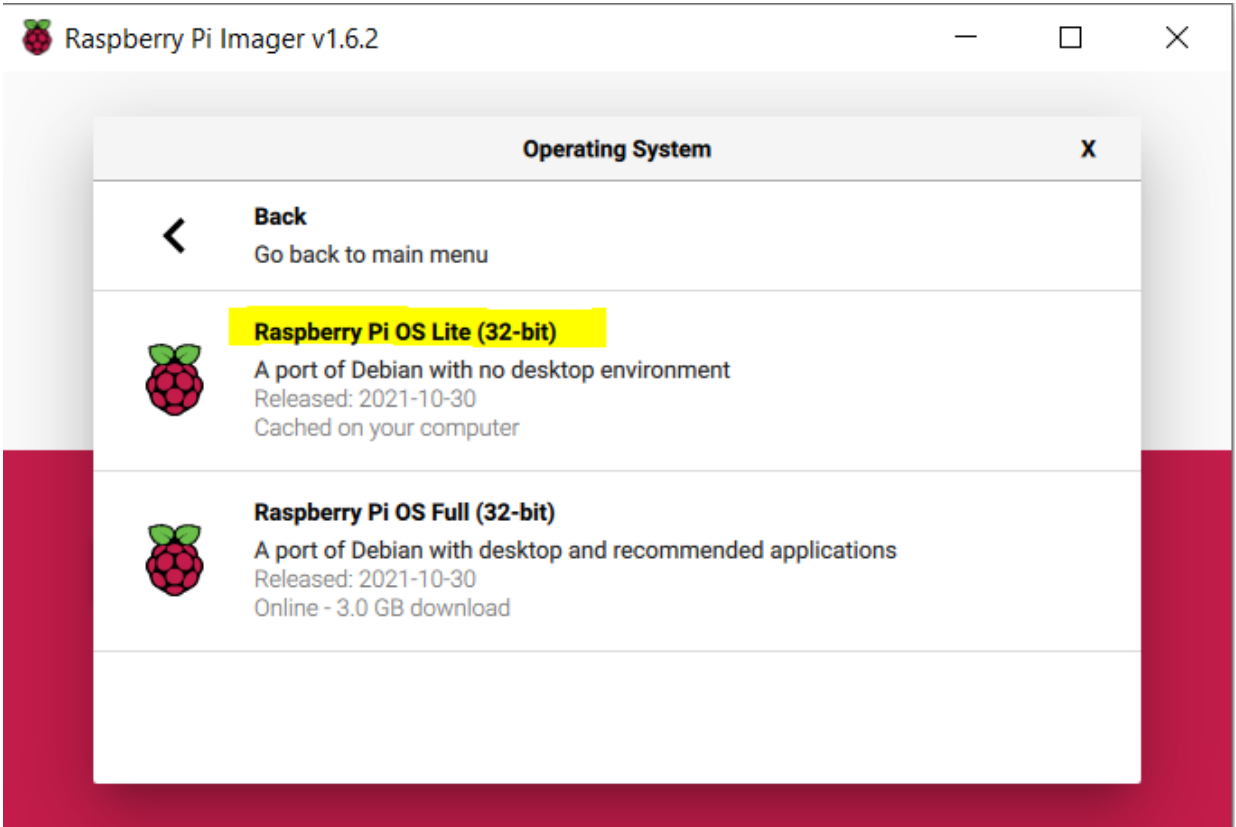


Рисунок 3.14. Остаточний вибір ОС у ПЗ Raspberry Pi Imager

1.4. Вибір сховища (рисунок 3.15):



Рисунок 3.15. Вибір сховища у ПЗ Raspberry Pi Imager

1.5. Вибір SD-карту рекомендовано мінімум 8ГБ (рисунок 3.16):

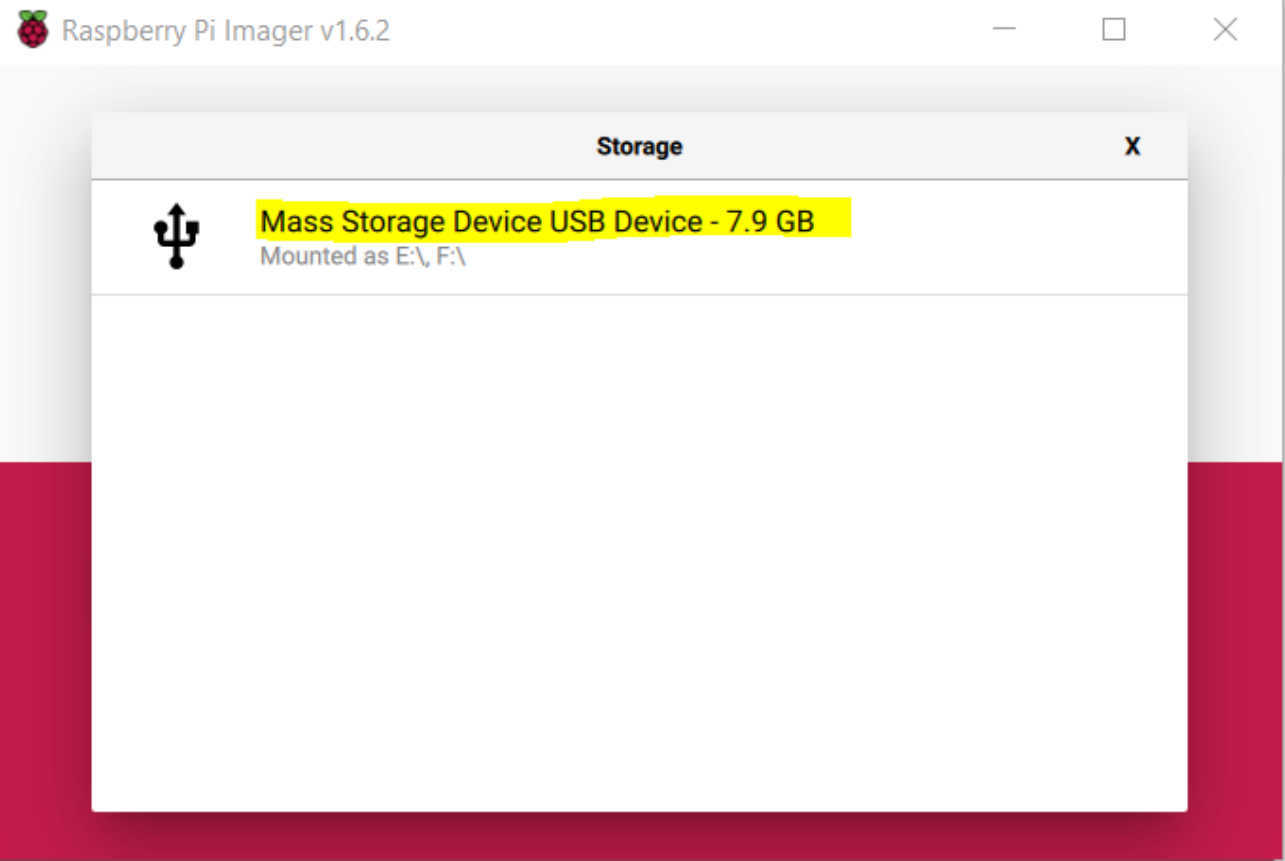


Рисунок 3.16. Вибір носія у ПЗ Raspberry Pi Imager

1.6. Натисніть *CTRL* + *SHIFT* + *X* для ввімкнення SSH (рисунок 3.17):

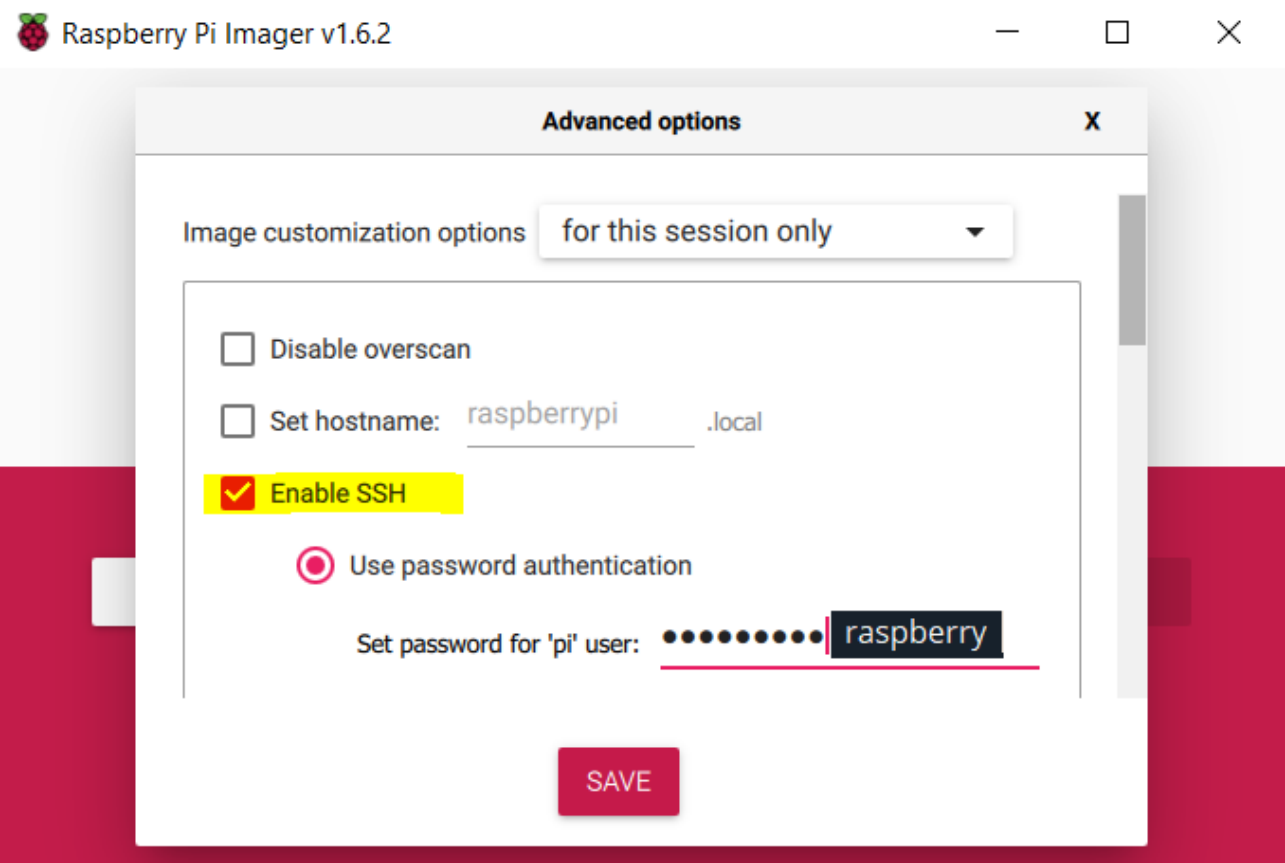


Рисунок 3.17. Відкриття порту 22 для SSH у ПЗ Raspberry Pi Imager

1.7. Запис образу ОС Raspberry Lite на SD-карту (рисунок 3.18):

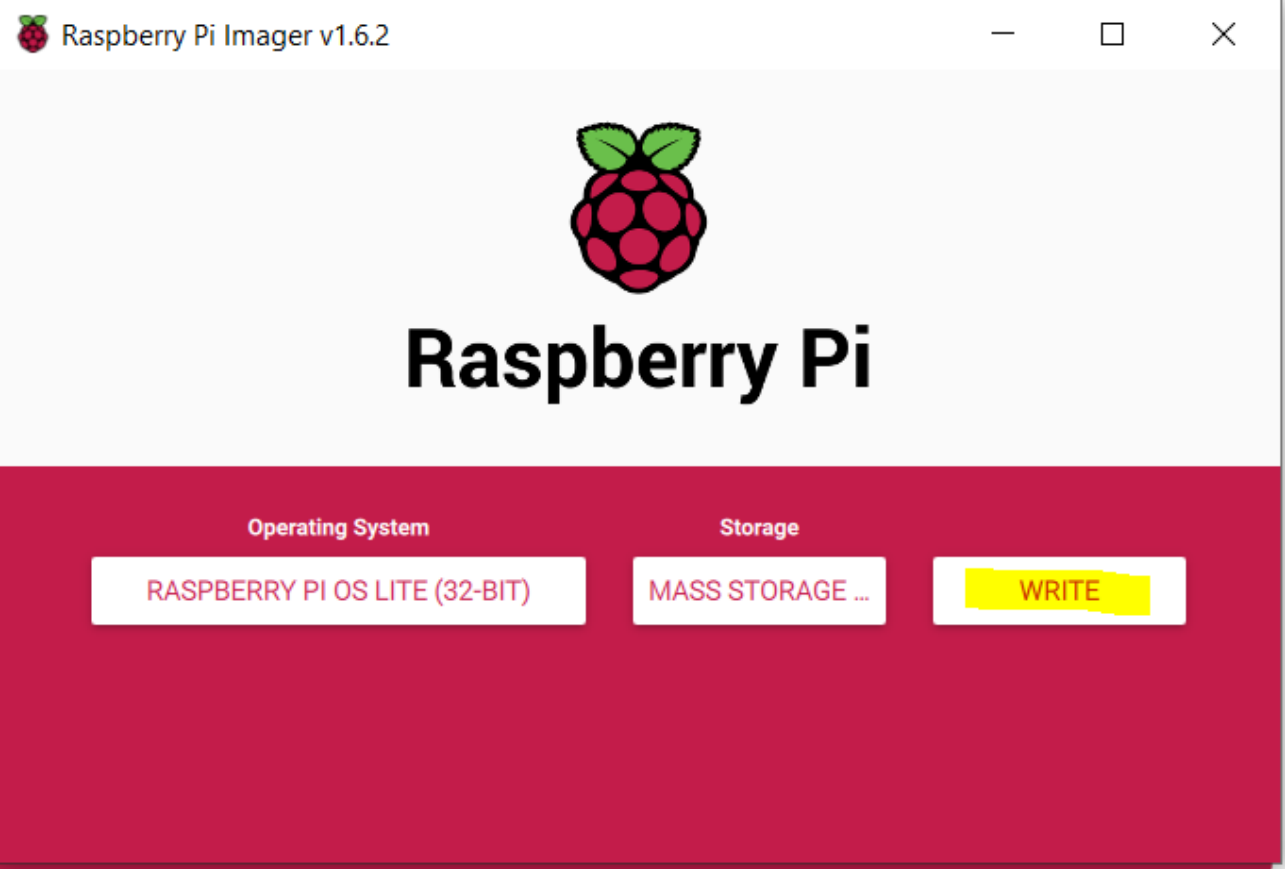


Рисунок 3.18. Запис ОС на носій за допомогою ПЗ Raspberry Pi Imager

2. Завантаження ОС з Raspberry Pi

Вставте SD-карту в клієнтську Raspberry Pi 4 та завантажте його. Використання полегшеної версії Raspbian надає лише текстову консоль взаємодії в ОС. Далі Raspberry Pi підключається до мережі за допомогою кабелю Ethernet.

Перевірка орендованих адреси DHCP-сервера, та чи плата Raspberry проініціалізувалась та отримала IP-адресу (рисунок 3.19):

DHCP Server									
DHCP Networks Leases Options Option Sets Vendor Classes Alerts									
+ - [Icons] Check Status Find									
	Address	MAC Address	Client ID	Server	Active Address	Active MAC Address	Active Host Name	Expires After	Status
D	10.100.40.6	04:B1:67:6A:64:36	1:4:b1:67:6a:64:36	dhcp_vlan40	10.100.40.6	04:B1:67:6A:64:36		00:06:05	bound
D	10.100.40.2	1C:E5:7F:F1:AD:11	1:1:c:e5:7f:f1:ad:11	dhcp_vlan40	10.100.40.2	1C:E5:7F:F1:AD:11	Vasyt-s-A72	00:06:20	bound
	10.100.30.40	04:92:26:58:DA:07		dhcp_vlan30			truenas.local		waiting
	10.100.20.40	04:92:26:58:DA:07		dhcp_vlan20			truenas.local		waiting
D	10.100.20.2	B0:5C:DA:E2:5B:4E	1:b0:5c:da:e2:5b:4e	dhcp_vlan20	10.100.20.2	B0:5C:DA:E2:5B:4E	KGOLUBENKO-N	00:08:59	bound
	10.100.10.40	04:92:26:58:DA:07		dhcp_vlan10			truenas.local		waiting
	10.100.10.20	08:55:31:E8:FF:2C	1:8:55:31:e8:ff:2c	dhcp_vlan10	10.100.10.20	08:55:31:E8:FF:2C	sw_2fl_NAS	00:09:27	bound
	10.100.10.6	08:55:31:E8:FF:2D	1:8:55:31:e8:ff:2d	dhcp_vlan10	10.100.10.6	08:55:31:E8:FF:2D	sw_0fl_2	00:06:08	bound
	10.100.10.5	08:55:31:7D:57:4E	1:8:55:31:7d:57:4e	dhcp_vlan10	10.100.10.5	08:55:31:7D:57:4E	sw_3fl	00:06:51	bound
	10.100.10.4	2C:C8:1B:48:4B:8C	1:2:c8:1b:48:4b:8c	dhcp_vlan10	10.100.10.4	2C:C8:1B:48:4B:8C	sw_1fl_1	00:06:27	bound
	10.100.10.2	08:55:31:50:8B:91	1:8:55:31:50:8b:91	dhcp_vlan10	10.100.10.2	08:55:31:50:8B:91	bridge_wsh_Ch1	00:06:25	bound
	10.100.1.40	04:92:26:58:DA:07		dhcp_main			truenas.local		waiting
D	10.100.1.32	E4:5F:01:11:4F:1F	1:e4:5f:01:11:4f:1f	dhcp_main	10.100.1.32	E4:5F:01:11:4F:1F	raspberrypi	00:09:33	bound

Рисунок 3.19. Таблиця орендованих IP-адрес на маршрутизаторі

3.2 Тестування мережі на базі створеної інфраструктури для завантаження "тонких" клієнтів

PXE – це протокол, що використовується для завантаження ОС по мережі. Цей метод використовується для встановлення будь-якого дистрибутиву Linux на робочий ПК з усіма попередніми конфігураціями ОС.

У цьому підрозділі продемонстровано та пояснюється як налаштувати завантаження ISO-образу по мережі, створеного у попередньому підрозділі. Описано як завантажувати Linux Ubuntu 21.04 в якості мережевого серверу загрузки PXE, щоб його встановлення було реалізовано без використання CD чи DVD приводу, або USB-накопичувача [27].

Налаштовано сервер завантаження (PXE-Boot-Server) в якості серверу завантаження PXE з використанням прошивки iPXE. Сервер завантаження працюватиме як TFTP та NFS сервери. Починається ланцюг завантаження з підключення до мережі, за яку відповідає мережеве обладнання (Router), працюватиме як DHCP-сервер. На зображенні нижче показано принцип, по якому необхідно підключати робочі ПК (рисунок 3.20). IP-адреси інфраструктури вказані як приклад за замочуванням:

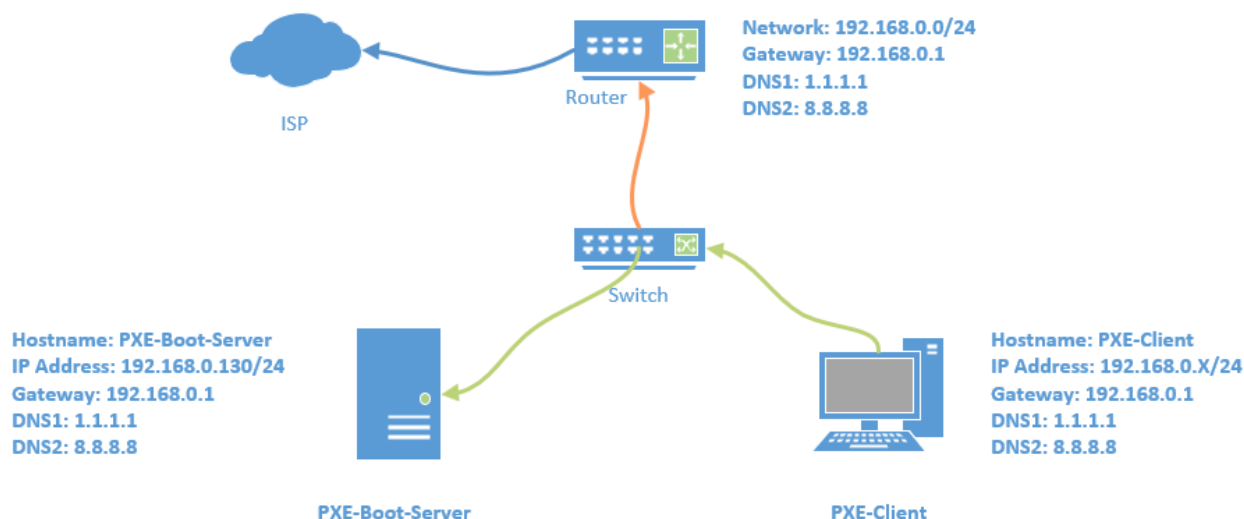


Рисунок 3.20. Інфраструктура мережевого завантаження

Linux Ubuntu 21.04

Файли мережевого завантаження збираються на станції Ubuntu, щоб не виникали конфлікти із робочими середовищами при завантаженні. Створюється ієрархія каталогів для завантаження PXE. Виконуються наступні команди:

```
sudo mkdir -p /pcboot/{nfs,tftp,image}  
sudo mkdir -p /pcboot/tftp/pxelinux.cfg  
sudo mkdir -v /pcboot/{nfs,tftp}
```

Встановлення всіх необхідних файлів завантажувача для завантаження по PXE:

```
sudo apt install -y syslinux pxelinux  
sudo cp -v /usr/lib/PXELINUX/pxelinux.0 /pcboot/tftp/  
sudo cp -v /usr/lib/syslinux/modules/bios/ldlinux.c32 /pcboot/tftp/
```

Копіювання та збереження у каталогах власного налаштованого файлу ISO-образу Ubuntu 21.04. І копіювання вмісту файлу ISO в каталог NFS. І копіювання файлів vmlinuz та initrd до каталогу TFTP.

```
sudo cp -Rfv /srvboot/tftp/boot/ubuntu-21.04-with-systemd.iso  
/pcboot/image/  
sudo mount -o loop ubuntu-21.04-with-systemd.iso /mnt/  
sudo cp -Rfv /mnt/. /pcboot/nfs/ubuntu2104/  
sudo cp -Rfv /pcboot/nfs/ubuntu2104/casper/{vmlinuz,initrd.lz}  
/pcboot/tftp/ubuntu2104/  
sudo umount /mnt
```

Створення файлу конфігурації автоматичного меню завантаження PXE з додаванням до цього файлу параметрів конфігурації мережевого завантаження:

```
cat << EOF | sudo tee '/pcboot/tftp/pxelinux.cfg/default'  
DEFAULT install  
LABEL install  
KERNEL ubuntu2104/vmlinuz  
INITRD ubuntu2104/initrd.lz APPEND root=/dev/ram0 ramdisk_size=250000  
ip=dhcp boot=casper  
netboot=nfs nfsroot=10.100.1.40:/mnt/slow-  
tank/netboot/pcboot/nfs/ubuntu2104 splash toram ---  
EOF
```

Створене дерево каталогів завантаження Live CD ISO, наведено нижче на зображенні та має наступний вигляд (Рисунок 3.21):

					КБ.КВ.Б – 125 – 22– ПЗ	
						51

```

riontric@riontric-VirtualBox:/srvboot$ tree -L 4 -h
├── [4.0K] tftp
│   ├── [4.0K] boot
│   │   ├── [2.6G] ubuntu-21.04-desktop-amd64.iso
│   │   ├── [110M] initrd
│   │   ├── [118K] ldlinux.c32
│   │   ├── [ 42K] pxelinux.0
│   │   ├── [4.0K] pxelinux.cfg
│   │   ├── [ 241] default
│   │   └── [4.0K] ubuntu2104
│   │       ├── [4.0K] boot
│   │       │   ├── [4.0K] grub
│   │       │   └── [178K] memtest86+.bin
│   │       ├── [2.0K] boot.catalog
│   │       ├── [4.0K] casper
│   │       │   ├── [ 56K] filesystem.manifest
│   │       │   ├── [2.8K] filesystem.manifest-minimal-remove
│   │       │   ├── [3.8K] filesystem.manifest-remove
│   │       │   ├── [ 11] filesystem.size
│   │       │   ├── [2.1G] filesystem.squashfs
│   │       │   ├── [ 833] filesystem.squashfs.gpg
│   │       │   ├── [110M] initrd
│   │       │   └── [ 14M] vmlinuz
│   │       ├── [4.0K] dists
│   │       │   ├── [4.0K] hirsute
│   │       │   ├── [  7] stable -> hirsute
│   │       │   └── [  7] unstable -> hirsute
│   │       ├── [4.0K] EFI
│   │       │   ├── [4.0K] boot
│   │       ├── [4.0K] install
│   │       ├── [ 23K] md5sum.txt
│   │       ├── [4.0K] pool
│   │       │   ├── [4.0K] main
│   │       │   └── [4.0K] restricted
│   │       ├── [4.0K] preseed
│   │       │   └── [ 460] ubuntu.seed
│   │       └── [  1] ubuntu -> .
│   └── [ 14M] vmlinuz
└── 19 directories, 18 files

```

Рисунок 3.21 – Ієрархія файлів завантаження PXE

Наступним розглянемо конфігурування мережевого завантаження пристроїв для забезпечення розумного офісного простору – Raspberry Pi 4.

Бонусом буде налагоджено завантаження Raspberry Pi 4 по протоколу PXE для забезпечення обладнання розумного офісу. Це зроблено з метою комфортного адміністрування плат, щоб не потрібно було при кожній похибці чи секундній відсутності доступу до мережі, перезавантажувати руками плати вручну з SD-картки – надається можливість конфігурування потів з маршрутизатору.

Raspberry Pi 4

Виконується вхід через консоль Putty, використовуючи облікові данні за замовчанням **login / password = pi / raspberry**.

Оновлення ОС Raspbian за допомогою утиліти apt-get і встановлення програми rpi-config:

```
sudo apt-get update && sudo apt-get upgrade
cd /lib/firmware/raspberrypi/bootloader/beta/
PI_EEPROM_VERSION=pieeprom-2021-10-27.bin
sudo wget https://github.com/raspberrypi/rpi-
eeprom/raw/master/firmware/beta/pieeprom-2021-10-27.bin
sudo rpi-eeprom-config
/lib/firmware/raspberrypi/bootloader/beta/pieeprom-2021-10-27.bin >
bootconf.txt
sudo nano bootconf.txt
```

Додавання параметрів у файл. Збережіть і закрийте bootconf.txt:

```
[all]
BOOT_UART=0
WAKE_ON_GPIO=1
ENABLE_SELF_UPDATE=1
BOOT_ORDER=0xf12
TFTP_IP=10.100.1.40
TFTP_PREFIX=1
TFTP_PREFIX_STR=f070c3d/ (1st raspbian)
TFTP_PREFIX_STR=d51212c/ (2nd raspbian)
```

Наступним кроком збережіть новий файл *bootconf.txt* в образ прошивки мікропрограми, яку копіювали раніше. Тепер встановіть новий завантажувач для Raspberry Pi 4:

```
rpi-eeprom-config --out netboot-pieeprom.bin --config bootconf.txt
pieeprom-2021-10-27.bin
sudo rpi-eeprom-update -d -f ./netboot-pieeprom.bin
```

Щоб переглянути конфігурацію, яку використовує поточний завантажувач в процесі останнього завантаження, запустіть *rpi-eeprom-config* або *vcgencmd bootloader_config*.

Редагування поточної конфігурації завантажувача:

```
sudo -E rpi-eeprom-config --edit
```

Виведення серійних номерів і MAC-адресів Raspbian плат виконується командами наведеними нижче:

```
cat /proc/cpuinfo | grep Serial | awk -F ':' '{print $2}' | tail -c 8  
ip addr show eth0 | grep ether | awk '{print $2}'
```

Серійний номер і MAC-адреса 1-го Raspbian (рисунок 3.22):

```
pi@raspberrypi:~ $ cat /proc/cpuinfo | grep Serial | awk -F ':' '{print $2}' | tail -c 8  
d51212c  
pi@raspberrypi:~ $ ip addr show eth0 | grep ether | awk '{print $2}'  
e4:5f:01:11:4f:1f  
pi@raspberrypi:~ $
```

Рисунок 3.22. Необхідна інформація для завантаження PXE першої плати

Серійний номер і MAC-адреса 2-го Raspbian (рисунок 3.23):

```
pi@raspberrypi:~ $ cat /proc/cpuinfo | grep Serial | awk -F ':' '{print $2}' | tail -c 8  
f070c3d  
pi@raspberrypi:~ $ ip addr show eth0 | grep ether | awk '{print $2}'  
e4:5f:01:11:4d:de  
pi@raspberrypi:~ $
```

Рисунок 3.23. Необхідна інформація для завантаження PXE першої плати

Вимкнення автоматичного оновлення:

```
sudo systemctl mask rpi-eeprom-update
```

Та перезавантаження:

```
sudo reboot
```

Зміна імені файлу завантаження та конфігурації DHCP-сервері (рисунок 3.24 - 3.25):

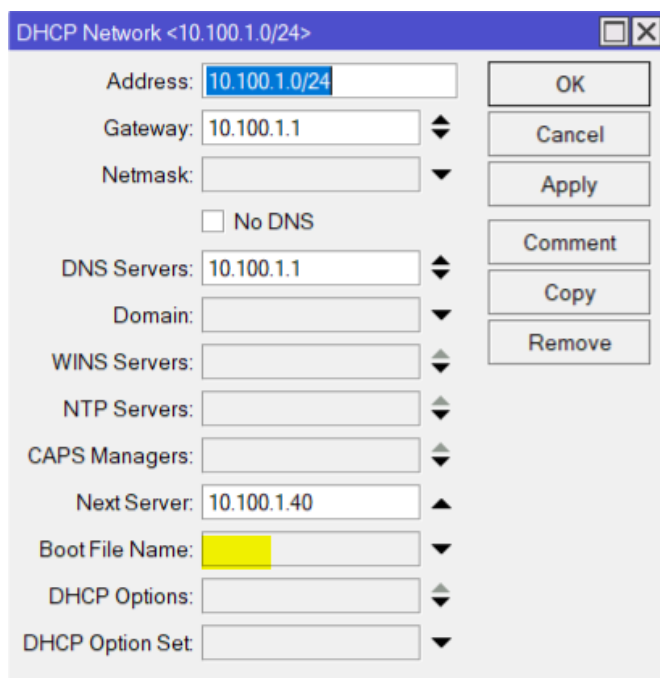


Рисунок 3.24. Видалення назви файлу завантаження для плат на DHCP-сервері

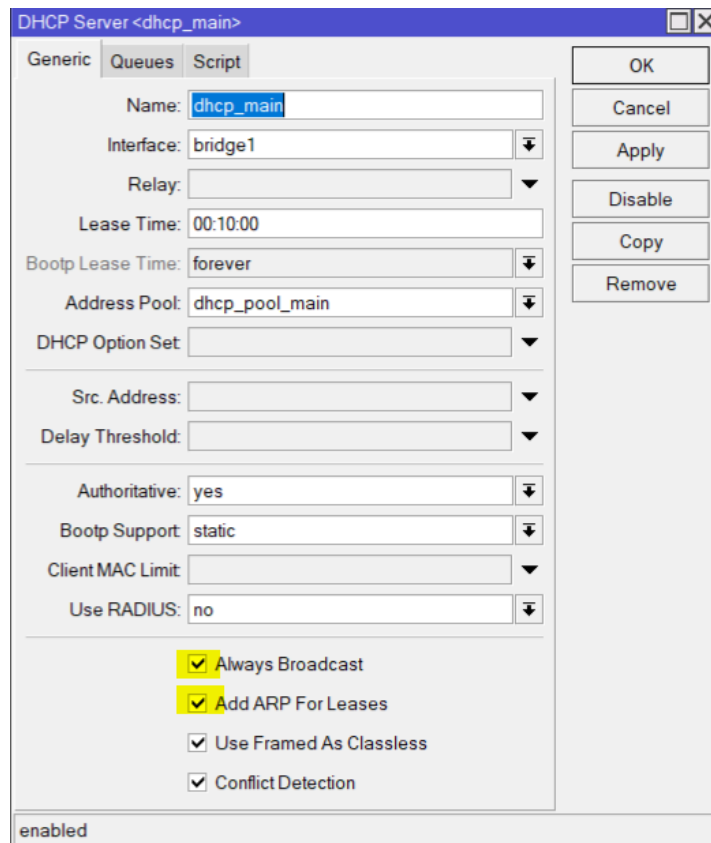


Рисунок 3.25. Зміна конфігурації для завантаження плат на DHCP-сервері

Файли мережевого завантаження збираються на плати Raspbian. Створюється ієрархія каталогів для завантаження PXE. Виконуються наступні команди:

```
#Creating main directories to boot the system:
sudo mkdir -p /raspbrianboot/{nfs,tftp,image}
sudo mkdir -p /raspbrianboot/nfs/rpi4-f070c3d
sudo mkdir -p /raspbrianboot/tftp/f070c3d
sudo mkdir -p /raspbrianboot/nfs/rpi4-d51212c
sudo mkdir -p /raspbrianboot/tftp/d51212c
sudo chmod 777 /raspbrianboot/tftp
#Creating main bootloader and files to boot the system:
cd /raspbrianboot/image
sudo wget https://downloads.raspberrypi.org/raspios_lite_armhf/images/raspios_lite_armhf-2021-11-08/2021-10-30-raspios-bullseye-armhf-lite.zip
sudo unzip 2021-10-30-raspios-bullseye-armhf-lite.zip
sudo kpartx -a -v 2021-10-30-raspios-bullseye-armhf-lite.img
sudo mkdir -p /raspbrianboot/image/{raspbrian_boot,raspbian_root}
sudo mount /dev/mapper/loop13p1 /raspbrianboot/image/raspbian_boot/
sudo mount /dev/mapper/loop13p2 /raspbrianboot/image/raspbian_root/
```

```

sudo cp -a /raspbianboot/image/raspbian_root/* /raspbianboot/nfs/rpi4-
f070c3d/
sudo cp -a /raspbianboot/image/raspbian_boot/* /raspbianboot/nfs/rpi4-
f070c3d/boot
sudo umount /raspbianboot/image/{raspbian_boot,raspbian_root}
sudo kpartx -d 2021-10-30-raspbios-bullseye-armhf-lite.img
cd
#Replacing the default firmware files with the latest version, if needed
it:
sudo rm /raspbianboot/nfs/rpi4-f070c3d/boot/start4.elf
sudo rm /raspbianboot/nfs/rpi4-f070c3d/boot/fixup4.dat
sudo wget https://github.com/Hexxeh/rpi-firmware/raw/master/start4.elf
-P /raspbianboot/nfs/rpi4-f070c3d/boot/
sudo wget https://github.com/Hexxeh/rpi-firmware/raw/master/fixup4.dat
-P /raspbianboot/nfs/rpi4-f070c3d/boot/
sudo cp -a /raspbianboot/nfs/rpi4-f070c3d/* /raspbianboot/nfs/rpi4-
d51212c/
#Cleaning /etc/fstab from boot system, enabling the SSH, telling 1st
raspbian to boot via the network by changing its cmdline.txt:
sudo sed -i /UUID/d /raspbianboot/nfs/rpi4-f070c3d/etc/fstab
echo "console=serial0,115200 console=tty1 root=/dev/nfs
nfsroot=10.100.1.40:/mnt/slow-tank/netboot/raspbianboot/nfs/rpi4-
f070c3d,vers=3 rw ip=dhcp rootwait elevator=deadline" | sudo tee
/raspbianboot/nfs/rpi4-f070c3d/boot/cmdline.txt
sudo touch /raspbianboot/nfs/rpi4-f070c3d/boot/ssh
#Cleaning /etc/fstab from boot system, enabling the SSH, telling 2nd
raspbian to boot via the network by changing its cmdline.txt:
sudo sed -i /UUID/d /raspbianboot/nfs/rpi4-d51212c/etc/fstab
echo "console=serial0,115200 console=tty1 root=/dev/nfs
nfsroot=10.100.1.40:/mnt/slow-tank/netboot/raspbianboot/nfs/rpi4-
d51212c,vers=3 rw ip=dhcp rootwait elevator=deadline" | sudo tee
/raspbianboot/nfs/rpi4-d51212c/boot/cmdline.txt
sudo touch /raspbianboot/nfs/rpi4-d51212c/boot/ssh
#Instead of mounting via a NAS-server:
#sudo cp -Rfv /raspbianboot/nfs/rpi4-f070c3d/boot/*
/raspbianboot/tftp/f070c3d/

```

Надсилання файлів мережевого завантаження PXE на NAS наступними командами:

```

sudo tar -cvf raspbianboot.tar.gz /raspbianboot
sudo tar -xvf raspbianboot.tar.gz
sudo tar -cvf rpi4-d51212c.tar.gz /raspbianboot/nfs/rpi4-d51212c

```

Конфігурування простору NFS для завантаження по мережі PXE на хаб-сервері NAS. Та додатковий простір NFS для завантаження Raspbian (рисунок 3.26):

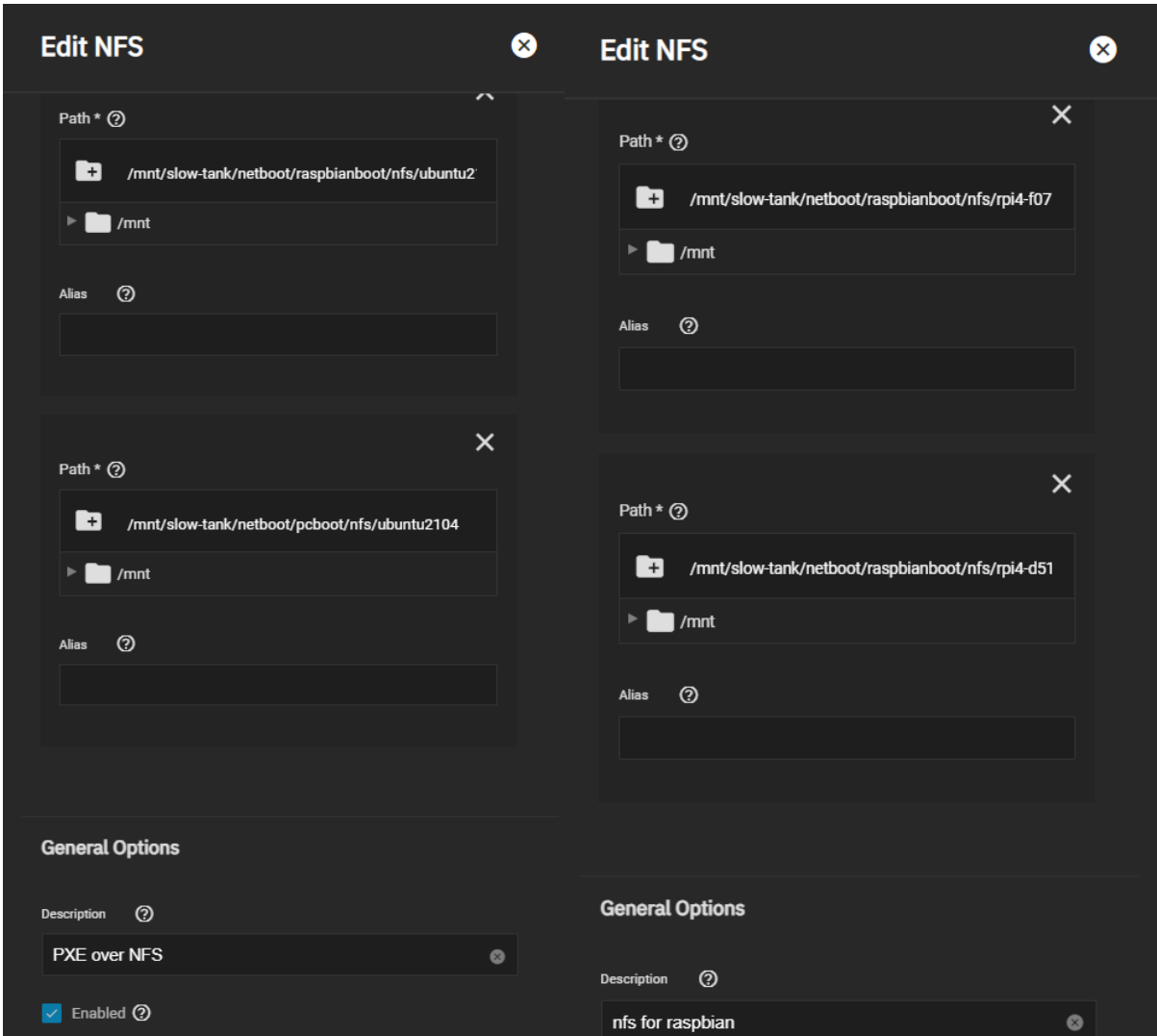


Рисунок 3.26. Додавання спільного ресурсу NFS з Truenas
Налаштування файлів на Truenas */etc/fstab* і монтування прив’язки через NFS за допомогою *mount_script.sh*:

```
cat << EOF | sudo tee /mnt/slow-  
tank/netboot/raspbianboot/mount_script.sh  
echo '/mnt/slow-tank/netboot/raspbianboot/nfs/rpi4-f070c3d/boot  
/mnt/slow-tank/netboot/raspbianboot/tftp/f070c3d none defaults,bind 0  
0' | tee -a /etc/fstab  
echo '/mnt/slow-tank/netboot/raspbianboot/nfs/rpi4-d51212c/boot  
/mnt/slow-tank/netboot/raspbianboot/tftp/d51212c none defaults,bind 0  
0' | tee -a /etc/fstab  
mount /mnt/slow-tank/netboot/raspbianboot/tftp/f070c3d  
mount /mnt/slow-tank/netboot/raspbianboot/tftp/d51212c  
EOF
```

Конфігурування системного файлу на сервері */etc/fstab* для NAS.

Наведено на зображенні (рисунок 3.27):

```
GNU nano 5.4 /etc/fstab *
boot-pool/grub /boot/grub zfs relatime,defaults 0 0
tmpfs /run/lock tmpfs rw,nosuid,nodev,noexec,relatime,size=100m 0 0
/mnt/slow-tank/netboot/raspbianboot/nfs/rpi4-f070c3d/boot /mnt/slow-tank/netboot/raspbianboot/tftp/f070c3d none default,bind 0 0
/mnt/slow-tank/netboot/raspbianboot/nfs/rpi4-d51212c/boot /mnt/slow-tank/netboot/raspbianboot/tftp/d51212c none default,bind 0 0
```

Рисунок 3.27. Налаштування */etc/fstab* на сервері NAS

Комбінація клавіш *Shift+Insert*, щоб вставити скопійований код в текстове середовище оболонки NAS. Виконання скрипту:

```
zsh /mnt/slow-tank/netboot/raspbianboot/mount_script.sh
```

Об'єднання файлів завантаження різних типів приладів з поточною конфігурацією PXE. І редагування завантажувального файлу:

```
cp -Rfv /mnt/slow-tank/netboot/pcboot/tftp/. /mnt/slow-
tank/netboot/raspbianboot/tftp/
cp -Rfv /mnt/slow-tank/netboot/pcboot/nfs/. /mnt/slow-
tank/netboot/raspbianboot/nfs/
cp -Rfv /mnt/slow-tank/netboot/pcboot/image/. /mnt/slow-
tank/netboot/raspbianboot/image/
#####
cat << EOF | sudo tee '/mnt/slow-
tank/netboot/raspbianboot/tftp/pxelinux.cfg/default'
DEFAULT install
LABEL install
KERNEL ubuntu2104/vmlinuz
INITRD ubuntu2104/initrd.lz APPEND root=/dev/ram0 ramdisk_size=250000
ip=dhcp boot=casper
netboot=nfs nfsroot=10.100.1.40:/mnt/slow-
tank/netboot/raspbianboot/nfs/ubuntu2104 splash toram ---
```

3.3 Аналіз результатів тестування мережевого завантаження

Виходячи з результатів тестування підсистеми захисту мережної інфраструктури підприємства побудованої з використанням технологій "тонких" клієнтів, обидва типи пристроїв, що використовують мережеве завантаження отримують динамічну IP-адресу та мають доступ до Інтернету. Таким чином пристрої переадресовуються на TFTP-сервер та можуть продовжувати завантаження по PXE. А це означає, що вдалося імітувати наявність двох різних віртуальних підмереж для завантаження двох різних машин: робочий простір ПК та розумні плати офісу. Важливий момент - щоб дані канали надавалися різними DHCP-серверами та не мають доступу між собою, щоб ізолювати адміністрування ззовні. Та при цьому надається можливість безперервного зв'язку з глобальною мережею Інтернет для обох типів завантажених приладів..

Щоб досягти спільного завантаження пристроїв пов'язуються їх конфігурації та реалізується об'єднання їх конфігурацій на спільному маршрутизаторі та хаб-сервері, із доступом до NFS файлів та розмежованими спільними папками.

Об'єднання та аналізування захищеного мережевого завантаження PXE для Raspberry Pi 4 та робочого простору Linux Ubuntu 21.04:

1. Збереження конфігурації PXE-сервера перед злиттям.
 - 1.1. Конфігурація DHCP-сервера та TFTP-сервера для завантаження PXE Ubuntu перед покращенням його злиття.

IP -> Сервер DHCP -> Мережа -> 10.100.1.0/24 (основна мережа). Наведено на зображенні (рисунок 3.28):

DHCP Network <10.100.1.0/24>

Address: 10.100.1.0/24

Gateway: 10.100.1.1

Netmask:

☐ No DNS

DNS Servers: 10.100.1.1

Domain:

WINS Servers:

NTP Servers:

CAPS Managers:

Next Server: 10.100.1.40

Boot File Name: pxelinux.0

DHCP Options:

DHCP Option Set:

OK

Cancel

Apply

Comment

Copy

Remove

Рисунок 3.28. DHCP-сервер для завантаження Ubuntu

Системні параметри -> Служби -> Налаштувати (поряд з TFTP). Наведено на зображенні (рисунок 3.29):

SERVICES

TFTP

Path

Directory ?

+ /mnt/slow-tank/netboot/srvboot/tftp

▸ /mnt

Connection

Host ?

10.100.1.40

Port ?

69

Username ?

tftp

Access

File Permissions ?

Read Write Execute

User ☒ ☐ ☐

Group ☒ ☐ ☐

☐ Allow New Files ?

Other Options

Auxiliary Parameters ?

Save Cancel

Рисунок 3.29. TFTP-сервер для завантаження Ubuntu

1.2. Конфігурація DHCP-сервера та TFTP-сервера для завантаження PXE Raspbian перед покращенням його злиття.

Параметр рядка *Boot File Name* порожній, тому що Raspberry Pi шукає свій серійний номер у папці TFTP на сервері.

IP -> Сервер DHCP -> Мережа -> 10.100.1.0/24 (основна мережа). Наведено на зображені (рисунок 3.30):

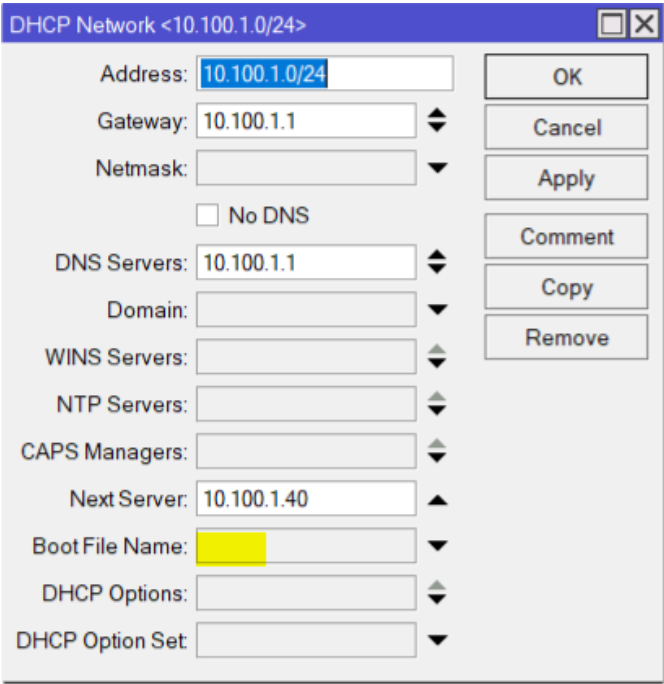


Рисунок 3.30. DHCP-сервер для завантаження Raspbian
Системні параметри -> Служби -> Налаштувати (поряд з TFTP). Наведено на зображені (рисунок 3.31):

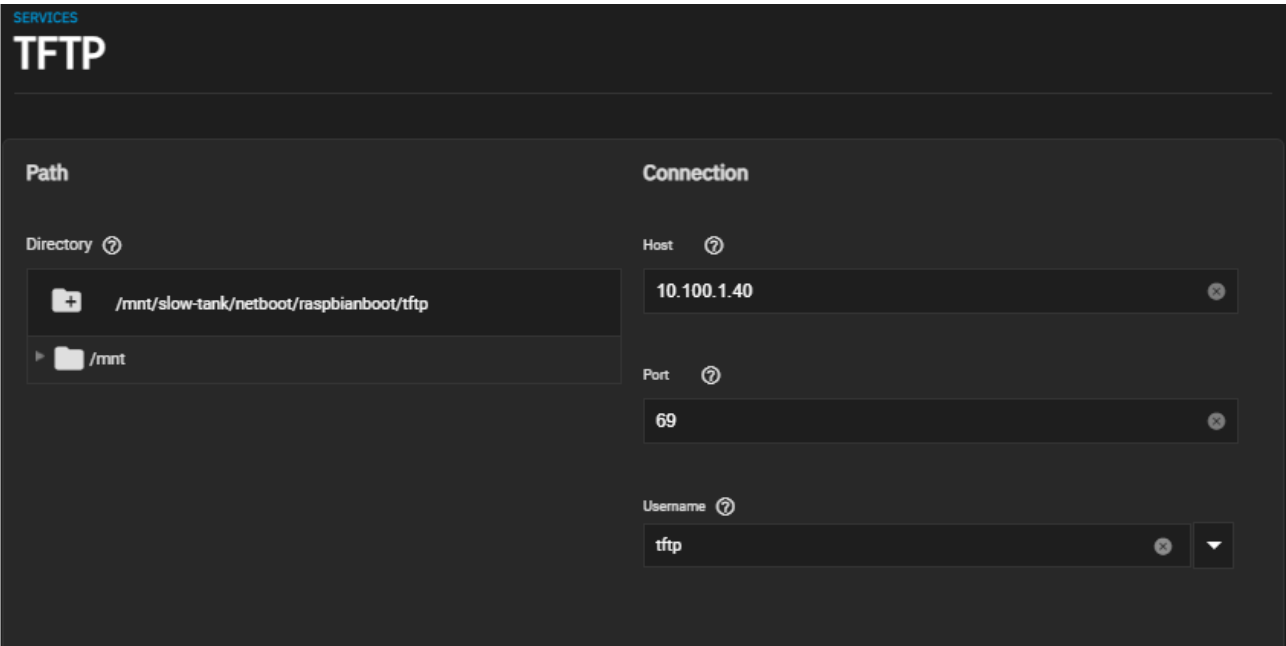


Рисунок 3.31. TFTP-сервер для завантаження Raspbian

2. Зміна структури каталогів для PXE-сервера:

2.1. Створення єдиної папки, куди буде прив'язаний TFTP-сервер.

Шлях `"/mnt/slow-tank/netboot/raspbianboot/tftp/"` буде основним посиланням на TFTP.

Усі інсталяційні файли для робочого простору Ubuntu знаходяться в `"/mnt/slow-tank/netboot/srvboot/tftp"`, командами неведеними нижче копіюємо до основної папки відносно TFTP:

```
cp -Rfv /mnt/slow-tank/netboot/srvboot/tftp/. /mnt/slow-tank/netboot/raspbianboot/tftp
```

2.2. Конфігурація TFTP-сервера та DHCP-сервера для об'єднаного завантаження мережі PXE.

Системні параметри -> Служби -> Налаштувати (поряд з TFTP). Наведено на зображенні (рисунок 3.32):

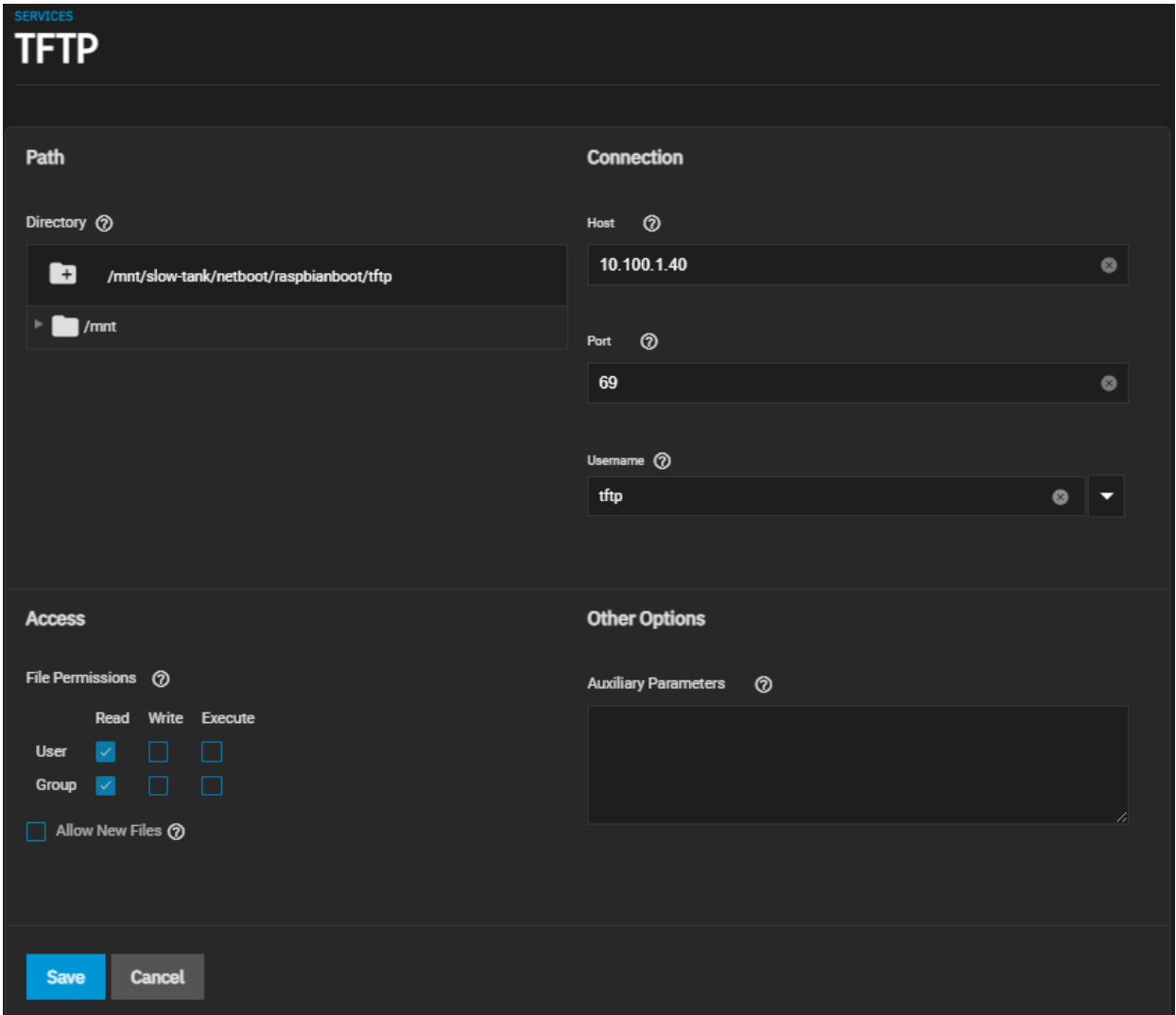


Рисунок 3.32. TFTP-сервер для завантаження Raspbian

За шляхом `"/mnt/slow-tank/netboot/raspbianboot/tftp"` знаходяться папки з серійними номерами Raspbian, слідом параметр рядка *Boot File Name* буде розміщено ім'я завантажувального файлу для робочого простору ОС Ubuntu.

IP -> Сервер DHCP -> Мережа -> 10.100.1.0/24 (основна мережа). Наведено на зображенні (рисунок 3.33):

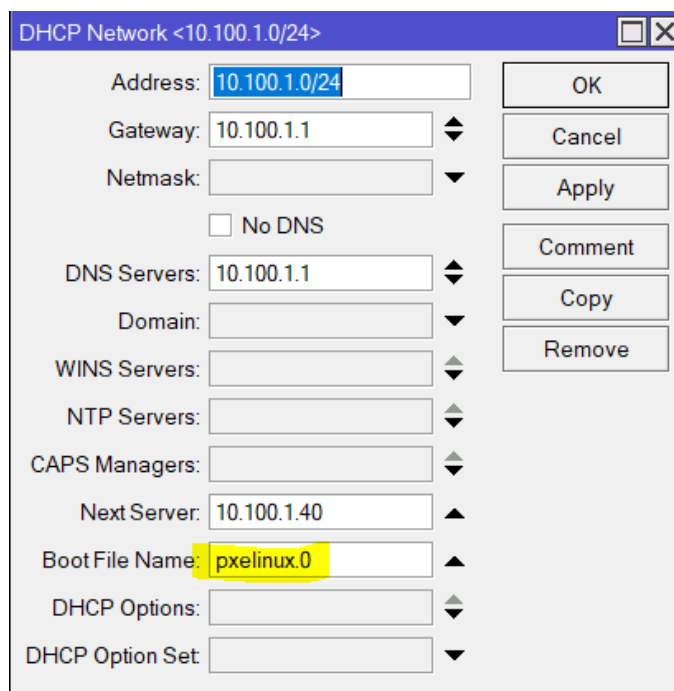


Рисунок 3.33. DHCP-сервер для завантаження Ubuntu

3. Після аналізів результатів тестування мережевого завантаження проведено перевірку процесу завантаження мережі PXE після покращення її злиття для розділення паралельного мережевого завантаження.
- 3.1. Структура каталогів для PXE-сервера після покращення на паралельне завантаження (рисунок 3.34):

```
truenas# cd /mnt/slow-tank/netboot/raspbianboot
truenas# ls
image mount_script.sh nfs tftp
truenas# cd tftp
truenas# ls
boot d51212c f070c3d initrd ldlinux.c32 pxelinux.0 pxelinux.cfg ubuntu2104 vmlinuz
truenas#
```

Рисунок 3.34. Структура каталогів PXE

3.2. Демонстрація паралельного розгортання Raspbian і готового робочого простору Ubuntu. Зображення початку та успішного завантаження обох типів ОС паралельно наведено на рисунках 3.35 – 3.36:



Рисунок 3.35. Початок паралельного мережевого завантаження PXE

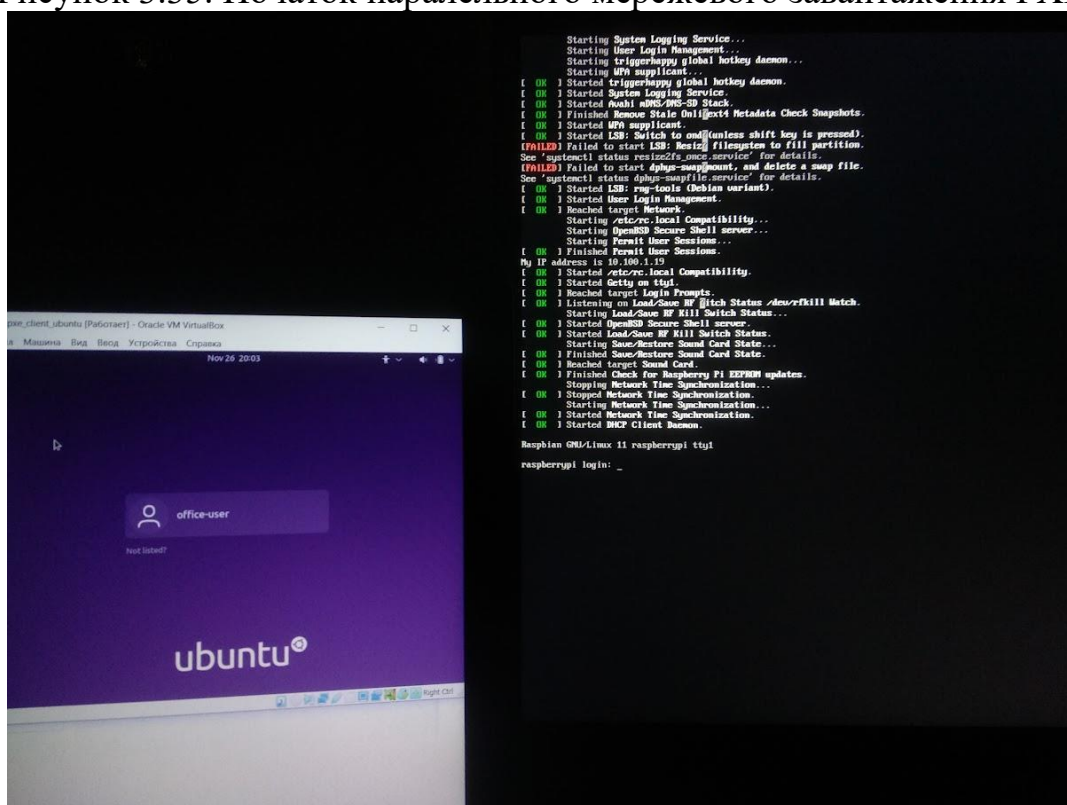


Рисунок 3.36. Успішне завершення паралельного мережевого завантаження PXE

Наявно переконалися, що мережеве виконується коректно для двох типів приладів у паралель. Процес відслідковування стану завантаження може здійснюватися також через маршрутизатор, так як по завершенню в списку орендованих IP-адрес DHCP-серверу з'явиться ім'я хоста машини відповідно до авторизованого користувача під конкретним IP. Якщо відповідну відповідь не отримано, з'являються сумніви щодо успішності завантаження, та надається можливість адміністрування кінцевим вузлом. А поки цей процес перезапущений є час та простір для перевірки конфігурацій завантаження по мережі. Поки використовується процес, варто перевірити захищеність доступних вузлів, які можуть бути підвернуті сторонньому впливу. Коли завантажуванні машини прийшли у доступний стан та їх ім'я відповідатимуть списку зарезервованих, що виділялися для користування під певними обліковими даними – у таблиці виділених IP з'явиться відповідна інформація (без секретних даних, навіть для адміністратора). Стан завантажування по мережевій сітці прийде до норми та IP-адреси, що використовуються перейдуть в таблицю маршрутизації, та отримають необхідні маршрути для їх роботи.

Після було наведено результати перевірки налаштувань шляхом штучного вимикання порту, по якому здійснюється завантаження, після чого видаляється виділений IP та доступ до Інтернету через нього, а завантаження або зависає, або призупиняється навіть без доступів, що було надано на певному етапі завантаження. Так, як були використані обидва завантаження, та їх конфігурації індивідуальні, та майже не стикаються між собою при тестуванні.

Як і планувалось, усі вимоги щодо захищеності успішно мережевого завантаження дотримуються. Переконалися, що після усіх налаштувань проведених з сервером, він працює правильно на просторах підприємства організації для завантаження "тонких" клієнтів, навіть без наявно у машини жорсткого носія, так як розгортання відбувається у операційній пам'яті комп'ютера. А через те, що було створену власну конфігурацію ISO-образу – вона є легкою та без зайвих попередніх інсталяції, та не потребує забагато ресурсів.

З точки зору безпеки, необхідно перевірити відсутність можливості того, що трафік з зовнішніх інтерфейсів організаційного простору буде пропускатися до внутрішньої мережі та передбачення загроз, котрі описано у перших розділах. Перевіряємо налаштування інспекції пакетів з запам'ятовуванням стану. Як результат, на маршрутизаторі бачимо, якщо пакет прийшов з внутрішнього інтерфейсу, який має більш високий рівень довіри (security-level) ніж зовнішній інтерфейс, тоді мережа запам'ятовує цю сесію, робить запис до динамічної таблиці, та наступник кроком – пропустить цей пакет далі по маршруту. Також на мережевому маршрутизаторі бачимо, якщо пакет прийшов з зовнішнього інтерфейсу, який має менший рівень довіри ніж внутрішній інтерфейс, тоді пакет інспектується та перевіряє його на відповідність адміністрований VLAN-підмережам. Якщо ці вимоги не задовольнятимуть мережеве обладнання, то пакет буде відкинуто з метою забезпечення безпеки внутрішніх мереж та ні в якому разі не виконуватиметься завантаження сторонніх ПК.

Наведено результати тестування у попередньому підрозділі, підключення робочої станції офісного простору корпоративної мережі та одержання IP-адреси від DHCP-сервера, у нашому випадку лише для відкритих портів офісного простору, яким був виділений певний рівень довіри. З такими налаштуваннями політики безпеки та інспектуванням пакетів, нашу внутрішню мережу та її користувачів можна вважати захищеними від зовнішніх небезпек, а саме від вихідного трафіку з Інтернету та VLAN-менеджменту. Результати тестування переконали нас у надійності конфігурування мережі.

Як головний результат тестування, ми переконалися у захищеному функціонуванні мережевого завантаження. Фінальним кроком тестування була виконана перевірка з робочої завантажених робочих станції офісу, що знаходиться в офісному просторі корпоративної мережі, на доступність безперервного зв'язку з Інтернетом та обмеженням для цих станій інформації для мережі розрізненням VLAN.

Висновки до третього розділу

В третьому розділі було описане налаштування ОС для двох видів пристрої, на базі власних образів.

Пристрої, що беруть участь у наданні можливостей мережевого завантаження, виконані необхідні для цього налаштування, та поетапно описані.

Маршрутизатор підготовлений до одночасного завантаження двох типів приладів як у паралель, так і окремо один від одного. Це показали результати тестування на даному етапі.

Мережеве сховище також були протестовані на правильність роботи TFTP та NFS серверів. Та було перевірено захищеність даних збережених на хаб-сервері, та детально протестоване унеможливлення влізти у процес завантаження, коли дані з серверу потрапляють на робочу станцію чи плату, як в процесі, так по його завершенню.

Проведено тестування функціонування мережі: образи відповідають потребам, у процесі розробки проекту образи було оптимізовано для більш швидшого розгортання на клієнтські машини.

Проведено аналіз результатів тестування та сформований висновок щодо проведених налаштувань.

ВИСНОВКИ

Метою магістерської наукової дипломної роботи була розробка проекту підсистеми захисту мережної інфраструктури підприємства, побудованої з використанням технологій "тонких" клієнтів.

Розглянута теоретична частина та актуальність теми проекту підсистеми захисту мережної інфраструктури підприємства, побудованої з використанням технологій "тонких" клієнтів.

Більш простими словами, робота над даною темою обумовлює підвищення надійності завантаження ОС. Розглянуто забезпечення безпечного завантаження, яке унеможливить доступ до даних мережі та на сервері через мережевий трафік, що знаходяться локально в офісі на пристрої, що завантажується. Процес завантаження був теоретично зроблено таким чином, що користувач не повинен слідкувати за завантаженням та встановленням – це буде виконано автоматично. Підтримка завантаженої ОС буде здійснюватися з сервера та головного маршрутизатору, на якому будуть заадмініструватися дані щодо користувачів з необхідними на те правами, та кожному буде виділено область даних відповідно до користувача.

Проаналізовано головну небезпеку, котра полягає в тому, що вузли надають підключитися до пристрою та безперешкодно завантажитися, а в гірших випадках замінити дані первинного завантаження, тим самим можливий витік та пошкодження даних пристроїв, що завантажуються після цих дій. Потім, використовуючи ті чи інші уразливості, кіберзлочинці цілком можуть дістатися до ваших файлів і вкрасти якусь конфіденційну інформацію, яку потім можна або продати, або вас же нею і шантажувати тощо.

Тому теоретично розглянуто безпеку користувачів та локальної мережі особливо детально, які будуть завантажувати на свої пристрої нашу налаштовану ОС. Прийнято рішення, що в нашому випадку буде надійніше, видавати права користувачам з підтримкою на самому ISO-образі.

В результаті теоретичного аналізу організації мережі інфраструктури підприємства, побудованої з використанням технологій "тонких" клієнтів

визначено, що для захищеного завантаження по мережі офісного (корпоративного) простору, необхідно застосувати комбінований спосіб захисту, який дозволить більш ефективно застосувати паралельне завантаження, так і відіграватись на його слабких сторонах за допомогою допоміжних засобів безпеки.

В нашому випадку основна задача полягає в створенні підсистеми захисту мережної інфраструктури підприємства, побудованої з використанням технологій "тонких" клієнтів. Для поставленої задачі ізолювання підмережі завантаження буде більш ніж достатньо. Це захиститись мережу від зовнішніх загроз, тобто наддасть можливість правильно завантажуватися робочим станціям локальної мережі та отримувати доступ до Інтернету, забезпечувати правильне функціонування головної мережі з розподіленням трафіку по ній. Все ж таки більш важливою метою нашого завдання є реалізація забезпечення захищеного завантаження ОС по мережі.

У формуванні логічної частини диплому - детально спроектовано організацію офісного простору інфраструктури підприємства, побудованої з використанням технологій "тонких" клієнтів, з розташування мережевого обладнання та принципами ізолювання даних та мережевого трафіку від робочих станцій та плат, які плануємо завантажувати через мережу. Сформовано таблиці параметрів адресації до відповідного обладнання та організування VLAN-підмереж за логічними групами.

Обґрунтовано вибір ПЗ та обладнання з функціоналом для реалізації мережевого завантаження ОС у практичній частині. Визначені образи, на базі яких будуть виконуватися завантаження та розподілення відповідно до ПК чи розумних плат. Визначилися зі складом пристроїв інфраструктури підприємства.

Сформоване приблизне уявлення про необхідні налаштування на кожному з пристроїв мережі, які беруть участь у процесі мережевого завантаження ОС. Вибір залежав від того, щоб на всіх пристроях був доступний необхідний функціонал для реалізації поставленого завдання дипломного проекту.

Останніми етапами впроваджено налаштування ОС для двох видів пристрої, на базі власних образів.

Пристрої, що беруть участь у наданні можливостей мережевого завантаження, виконані необхідні для цього налаштування, та поетапно описані.

Маршрутизатор підготовлений до одночасного завантаження двох типів приладів як у паралель, так і окремо один від одного. Це показали результати тестування на даному етапі.

Мережеве сховище також були протестовані на правильність роботи TFTP та NFS серверів. Та було перевірено захищеність даних збережених на хаб-сервері, та детально протестоване унеможливлення влізти у процес завантаження, коли дані з серверу потрапляють на робочу станцію чи плату, як в процесі, так по його завершенню.

Проведено тестування функціонування мережі: образи відповідають потребам, у процесі розробки проекту образи було оптимізовано для більш швидшого розгортання на клієнтські машини.

Проведено аналіз результатів тестування та сформований висновок щодо проведених налаштувань.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Завантаження бездискових Linux станцій за допомогою PXE (pxe boot linux diskless dhcp tftp) [Електронний ресурс] – Режим доступу до ресурсу: https://www.opennet.ru/base/sys/pxe_diskless.txt.html
2. Кроки встановлення завантажувального сервера PXE у віртуальній машині Ubuntu Server [Електронний ресурс] – Режим доступу до ресурсу: <https://gastack.ru/ubuntu/412574/pxe-boot-server-installation-steps-in-ubuntu-server-vm>
3. Pre-OS Boot [Електронний ресурс] – Режим доступу до ресурсу: <https://attack.mitre.org/techniques/T1542/>
4. Pre-OS Boot: System Firmware [Електронний ресурс] – Режим доступу до ресурсу: <https://attack.mitre.org/techniques/T1542/001/>
5. Pre-OS Boot: Component Firmware [Електронний ресурс] – Режим доступу до ресурсу: <https://attack.mitre.org/techniques/T1542/002/>
6. Bootkit samples [Електронний ресурс] – Режим доступу до ресурсу: <https://github.com/hardenedvault/bootkit-samples>
7. Drive - The Master Boot Record (MBR) [Електронний ресурс] – Режим доступу до ресурсу: <https://datacadamia.com/io/drive/mbr>
8. Volume Boot Record (VBR) [Електронний ресурс] – Режим доступу до ресурсу: <https://www.techopedia.com/definition/25922/volume-boot-record-vbr>
9. Pre-OS Boot: Bootkit [Електронний ресурс] – Режим доступу до ресурсу: <https://attack.mitre.org/techniques/T1542/003/>
10. What Is Rootkit? Definition, Detection, Removal, and Prevention Best Practices for 2022 [Електронний ресурс] – Режим доступу до ресурсу: <https://www.spiceworks.com/it-security/identity-access-management/articles/what-is-rootkit/>
11. Pre-OS Boot: ROMMONkit [Електронний ресурс] – Режим доступу до ресурсу: <https://attack.mitre.org/techniques/T1542/004/>

12. Network Boot Background Information [Електронний ресурс] – Режим доступу до ресурсу: <https://docs.vmware.com/en/VMware-vSphere/7.0/com.vmware.esxi.install.doc/GUID-F594C1E5-E385-4F32-A88E-4BFF81B8AFBD.html>
13. Pre-OS Boot: TFTP Boot [Електронний ресурс] – Режим доступу до ресурсу: <https://attack.mitre.org/techniques/T1542/005/>
14. Ubuntu based thin client made with your own hands [Електронний ресурс] – Режим доступу до ресурсу: <https://selivan.github.io/2018/03/08/ubuntu-based-thin-client.html>
15. Raspberry Pi PXE Boot – Network booting a Pi 4 without an SD card [Електронний ресурс] – Режим доступу до ресурсу: <https://linuxhit.com/raspberry-pi-pxe-boot-netbooting-a-pi-4-without-an-sd-card/>
16. Основи GNS3. Огляд [Електронний ресурс] – Режим доступу до ресурсу: <https://habr.com/ru/post/266503/>
17. Офіційний сайт Mikrotik [Електронний ресурс] – Режим доступу до ресурсу: https://mikrotik.com/product/rb4011igs_rm
18. VirtualBox [Електронний ресурс] – Режим доступу до ресурсу: <https://uk.wikipedia.org/wiki/VirtualBox>
19. What Is Ubuntu? [Електронний ресурс] – Режим доступу до ресурсу: <https://www.howtogeek.com/763775/what-is-ubuntu/>
20. What's New in Ubuntu 21.04 'Hirsute Hippo' [Електронний ресурс] – Режим доступу до ресурсу: <https://www.howtogeek.com/723039/whats-new-in-ubuntu-21.04-hirsute-hippo/>
21. FreeNAS vs TrueNAS: What Are the Differences and How to Upgrade? [MiniTool Tips] [Електронний ресурс] – Режим доступу до ресурсу: <https://www.minitool.com/backup-tips/freenas-vs-truenas.html>
22. How To Create A Custom Ubuntu Live ISO Image With Cubic [Електронний ресурс] – Режим доступу до ресурсу: <https://ostechnix.com/how-to-create-a-custom-ubuntu-live-iso-image-with-cubic/>
23. ORACLE man pages section 8: System Administration Commands

[Электронный ресурс] – Режим доступа до ресурсу:

https://docs.oracle.com/cd/E88353_01/html/E72487/chroot-8.html

24. LiveCDCustomization. Advanced Customizations. [Электронный ресурс] –

Режим доступа до ресурсу:

[https://help.ubuntu.com/community/LiveCDCustomization#Removing the .28Casper.29_Autologin](https://help.ubuntu.com/community/LiveCDCustomization#Removing_the_.28Casper.29_Autologin)

25. Automatic mounts using systemd [Электронный ресурс] – Режим доступа до ресурсу: <https://anteru.net/blog/2019/automatic-mounts-using-systemd/>

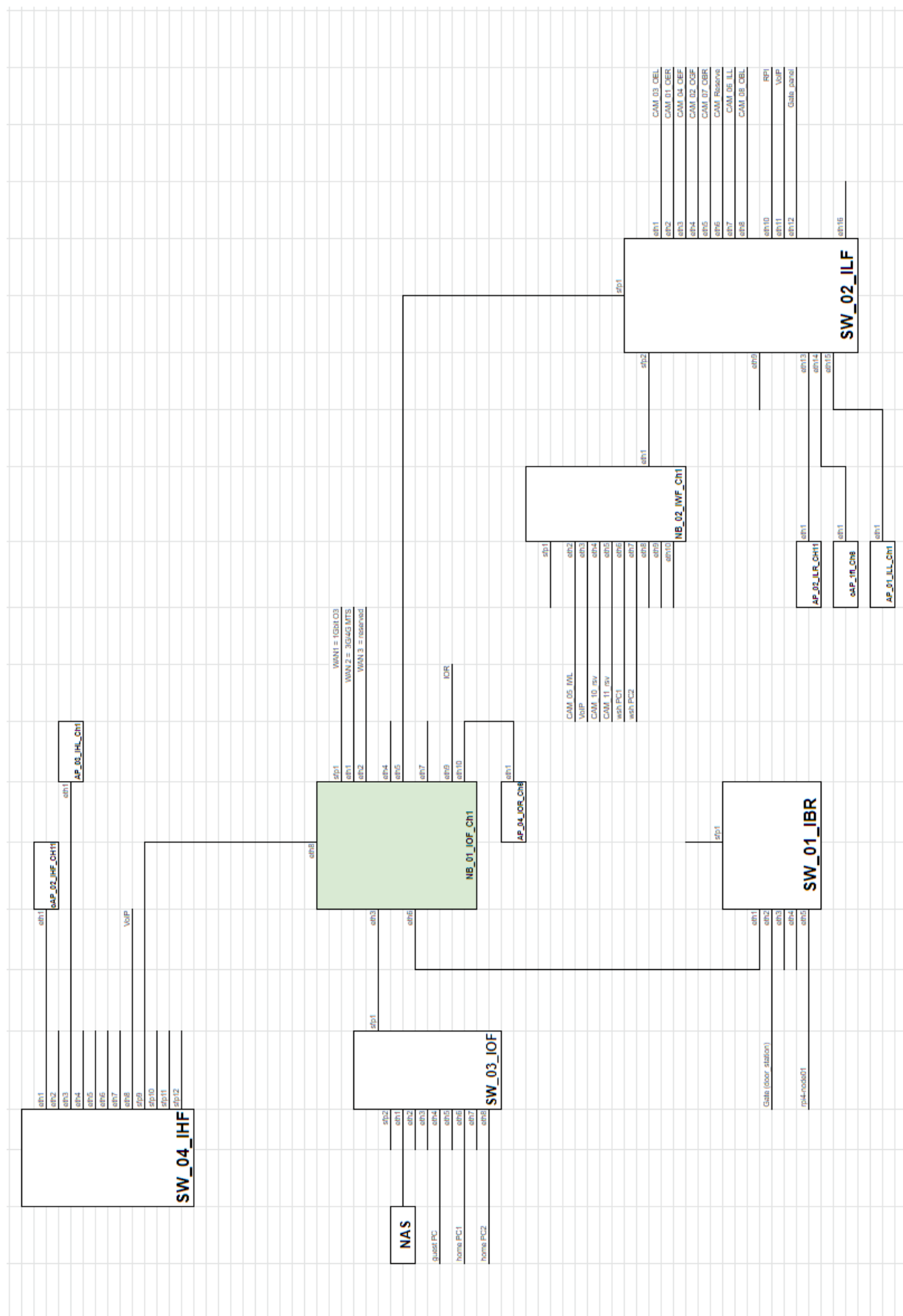
26. B.2. Using preseeding [Электронный ресурс] – Режим доступа до ресурсу: <https://www.debian.org/releases/jessie/amd64/apbs02.html.en>

27. Configuring PXE Network Boot Server on Ubuntu 22.04 LTS [Электронный ресурс] – Режим доступа до ресурсу: https://linuxhint.com/pxe_boot_ubuntu_server/

ДОДАТКИ

					КБ.КВ.Б – 125 – 22– ПЗ	
						74

Детальна функціонуюча схема побудованої мережі



Додаток Б Лістинги конфігурацій:

Cubic:

```
sudo apt-add-repository ppa:cubic-
wizard/release
sudo apt-key adv --keyserver
keyserver.ubuntu.com --recv-keys
B7579F80E494ED3406A59DF9081525E2B4F1283B
sudo apt update
sudo apt install cubic
!
mkdir extract-cd
cd extract-cd
mkdir casper
cd casper
cp -Rfv /boot/vmlinuz-5.11.0-16-generic
~/extract-cd/casper/vmlinuz
cp -Rfv /boot/initrd.img-5.11.0-16-
generic ~/extract-cd/casper/initrd.gz
mkdir temp
cd temp
gunzip -dc ../initrd.gz | cpio -imvd --
no-absolute-filenames
cp scripts/casper-
bottom/25configure_init scripts/casper-
bottom/25configure_init.orig
nano scripts/casper-
bottom/25configure_init
cp ../initrd.gz ../initrd.gz.orig
find . | cpio -o -H newc | gzip -9 >
../initrd.gz!
!
21:log_begin_msg "$DESCRIPTION"
22:
23:# Arrange for shells on virtual
consoles, rather than login prompts
24:
25:if [ -n "" ]; then
!
adduser office-user!
!
```

```
root@cubic:~# mkdir -p
/media/sharedfiles/homeuser
root@cubic:~# cd /etc/systemd/system/
root@cubic: /etc/systemd/system# nano
media-sharedfiles-homeuser.mount
root@cubic: /etc/systemd/system# nano
media-sharedfiles-homeuser.automount
root@cubic: /etc/systemd/system# cd
/home/office-user/
root@cubic: /home/office-user# nano
.smbcredentials
root@cubic: /home/office-user# cd
root@cubic:~# systemctl enable media-
sharedfiles-homeuser.automount
!
mkdir -p /media/sharedfiles/homeuser
cat << EOF | sudo tee
/etc/systemd/system/media-sharedfiles-
homeuser.mount
[Unit]
Description=cifs mount script

[Mount]
What=//10.100.1.40/sharedfiles/homeuser
Where=/media/sharedfiles/homeuser
Type=cifs
Options=credentials=/home/riontric/.smbc
redentials,rw,noperm,file_mode=0755,
dir_mode=0755,_netdev

[Install]
WantedBy=multi-user.target
EOF
systemctl enable media-sharedfiles-
homeuser.automount
!
Ubuntu:
sudo mkdir -p /pcboot/{nfs,tftp,image}
sudo mkdir -p /pcboot/tftp/pxelinux.cfg
sudo mkdir -v /pcboot/{nfs,tftp}
```

```

!
sudo apt install -y syslinux
pxelinuxsudo
sudo cp -v /usr/lib/PXELINUX/pxelinux.0
/pcboot/tftp/
sudo cp -v
/usr/lib/syslinux/modules/bios/ldlinux.c
32 /pcboot/tftp/
!
sudo cp -Rfv /srvboot/tftp/boot/ubuntu-
21.04-with-systemd.iso /pcboot/image/
sudo mount -o loop ubuntu-21.04-with-
systemd.iso /mnt/
sudo cp -Rfv /mnt/.
/pcboot/nfs/ubuntu2104/
sudo cp -Rfv
/pcboot/nfs/ubuntu2104/casper/{vmlinuz,i
nitrd.lz} /pcboot/tftp/ubuntu2104/
sudo umount /mnt
!
cat << EOF | sudo tee
'/pcboot/tftp/pxelinux.cfg/default'
DEFAULT install
LABEL install
KERNEL ubuntu2104/vmlinuz
INITRD ubuntu2104/initrd.lz APPEND
root=/dev/ram0 ramdisk_size=250000
ip=dhcp boot=casper
netboot=nfs
nfsroot=10.100.1.40:/mnt/slow-
tank/netboot/pcboot/nfs/ubuntu2104
splash toram ---
EOF
!
Raspbian:
sudo apt-get update && sudo apt-get
upgrade
cd
/lib/firmware/raspberrypi/bootloader/bet
a/

```

```

PI_EEPROM_VERSION=pieeprom-2021-10-
27.bin
sudo wget
https://github.com/raspberrypi/rpi-
eeprom/raw/master/firmware/beta/pieeprom
-2021-10-27.bin
sudo rpi-eeprom-config
/lib/firmware/raspberrypi/bootloader/bet
a/pieeprom-2021-10-27.bin > bootconf.txt
sudo nano bootconf.txt
!
[all]
BOOT_UART=0
WAKE_ON_GPIO=1
ENABLE_SELF_UPDATE=1
BOOT_ORDER=0xf12
TFTP_IP=10.100.1.40
TFTP_PREFIX=1
TFTP_PREFIX_STR=f070c3d/ (1st raspbian)
TFTP_PREFIX_STR=d51212c/ (2nd raspbian)
!
rpi-eeprom-config --out netboot-
pieeprom.bin --config bootconf.txt
pieeprom-2021-10-27.bin
sudo rpi-eeprom-update -d -f ./netboot-
pieeprom.bin!
!
sudo -E rpi-eeprom-config --edit
!
cat /proc/cpuinfo | grep Serial | awk -F
': ' '{print $2}' | tail -c 8
ip addr show eth0 | grep ether | awk
'{print $2}'!
!
sudo systemctl mask rpi-eeprom-update
!
sudo reboot
!
#Creating main directories to boot the
system:

```

```

sudo mkdir -p
/raspbianboot/{nfs,tftp,image}
sudo mkdir -p /raspbianboot/nfs/rpi4-
f070c3d
sudo mkdir -p /raspbianboot/tftp/f070c3d
sudo mkdir -p /raspbianboot/nfs/rpi4-
d51212c
sudo mkdir -p /raspbianboot/tftp/d51212c
sudo chmod 777 /raspbianboot/tftp
#Creating main bootloader and files to
boot the system:
cd /raspbianboot/image
sudo wget
https://downloads.raspberrypi.org/raspio
s_lite_armhf/images/raspbios_lite_armhf-
2021-11-08/2021-10-30-raspbios-bullseye-
armhf-lite.zip
sudo unzip 2021-10-30-raspbios-bullseye-
armhf-lite.zip
sudo kpartx -a -v 2021-10-30-raspbios-
bullseye-armhf-lite.img
sudo mkdir -p
/raspbianboot/image/{raspbian_boot,raspb
ian_root}
sudo mount /dev/mapper/loop13p1
/raspbianboot/image/raspbian_boot/
sudo mount /dev/mapper/loop13p2
/raspbianboot/image/raspbian_root/
sudo cp -a
/raspbianboot/image/raspbian_root/*
/raspbianboot/nfs/rpi4-f070c3d
sudo cp -a
/raspbianboot/image/raspbian_boot/*
/raspbianboot/nfs/rpi4-f070c3d/boot
sudo umount
/raspbianboot/image/{raspbian_boot,raspb
ian_root}
sudo kpartx -d 2021-10-30-raspbios-
bullseye-armhf-lite.img
cd

```

```

#Replacing the default firmware files
with the latest version, if needed it:
sudo rm /raspbianboot/nfs/rpi4-
f070c3d/boot/start4.elf
sudo rm /raspbianboot/nfs/rpi4-
f070c3d/boot/fixup4.dat
sudo wget https://github.com/Hexxeh/rpi-
firmware/raw/master/start4.elf -P
/raspbianboot/nfs/rpi4-f070c3d/boot/
sudo wget https://github.com/Hexxeh/rpi-
firmware/raw/master/fixup4.dat -P
/raspbianboot/nfs/rpi4-f070c3d/boot/
sudo cp -a /raspbianboot/nfs/rpi4-
f070c3d/* /raspbianboot/nfs/rpi4-
d51212c/
#Cleaning /etc/fstab from boot system,
enabling the SSH, telling 1st raspbian
to boot via the network by changing its
cmdline.txt:
sudo sed -i /UUID/d
/raspbianboot/nfs/rpi4-f070c3d/etc/fstab
echo "console=serial0,115200
console=tty1 root=/dev/nfs
nfsroot=10.100.1.40:/mnt/slow-
tank/netboot/raspbianboot/nfs/rpi4-
f070c3d,vers=3 rw ip=dhcp rootwait
elevator=deadline" | sudo tee
/raspbianboot/nfs/rpi4-
f070c3d/boot/cmdline.txt
sudo touch /raspbianboot/nfs/rpi4-
f070c3d/boot/ssh
#Cleaning /etc/fstab from boot system,
enabling the SSH, telling 2nd raspbian
to boot via the network by changing its
cmdline.txt:
sudo sed -i /UUID/d
/raspbianboot/nfs/rpi4-d51212c/etc/fstab
echo "console=serial0,115200
console=tty1 root=/dev/nfs
nfsroot=10.100.1.40:/mnt/slow-
tank/netboot/raspbianboot/nfs/rpi4-

```

```

d51212c,vers=3 rw ip=dhcp rootwait
elevator=deadline" | sudo tee
/raspbianboot/nfs/rpi4-
d51212c/boot/cmdline.txt
sudo touch /raspbianboot/nfs/rpi4-
d51212c/boot/ssh
#Instead of mounting via a NAS-server:
#sudo cp -Rfv /raspbianboot/nfs/rpi4-
f070c3d/boot/*
/raspbianboot/tftp/f070c3d/
!
sudo tar -cvf raspbianboot.tar.gz
/raspbianboot
sudo tar -xvf raspbianboot.tar.gz
sudo tar -cvf rpi4-d51212c.tar.gz
/raspbianboot/nfs/rpi4-d51212c
!
cat << EOF | sudo tee /mnt/slow-
tank/netboot/raspbianboot/mount_script.s
h
echo '/mnt/slow-
tank/netboot/raspbianboot/nfs/rpi4-
f070c3d/boot /mnt/slow-
tank/netboot/raspbianboot/tftp/f070c3d
none defaults,bind 0 0' | tee -a
/etc/fstab
echo '/mnt/slow-
tank/netboot/raspbianboot/nfs/rpi4-
d51212c/boot /mnt/slow-
tank/netboot/raspbianboot/tftp/d51212c
none defaults,bind 0 0' | tee -a
/etc/fstab
mount /mnt/slow-
tank/netboot/raspbianboot/tftp/f070c3d
mount /mnt/slow-
tank/netboot/raspbianboot/tftp/d51212c
EOF
!
zsh /mnt/slow-
tank/netboot/raspbianboot/mount_script.s
h

```

```

!
cp -Rfv /mnt/slow-
tank/netboot/pcboot/tftp/. /mnt/slow-
tank/netboot/raspbianboot/tftp/
cp -Rfv /mnt/slow-
tank/netboot/pcboot/nfs/. /mnt/slow-
tank/netboot/raspbianboot/nfs/
cp -Rfv /mnt/slow-
tank/netboot/pcboot/image/. /mnt/slow-
tank/netboot/raspbianboot/image/
#####
cat << EOF | sudo tee '/mnt/slow-
tank/netboot/raspbianboot/tftp/pxelinux.
cfg/default'
DEFAULT install
LABEL install
KERNEL ubuntu2104/vmlinuz
INITRD ubuntu2104/initrd.lz APPEND
root=/dev/ram0 ramdisk_size=250000
ip=dhcp boot=casper
netboot=nfs
nfsroot=10.100.1.40:/mnt/slow-
tank/netboot/raspbianboot/nfs/ubuntu2104
splash toram ---
!
cp -Rfv /mnt/slow-
tank/netboot/srvboot/tftp/. /mnt/slow-
tank/netboot/raspbianboot/tftp

```