

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 бакалавра
на тему: «Проект розширеного управління груповими
політиками з використанням PowerShell»

Виконав: студент 4-го курсу групи КІ-21-2
спеціальності 123 «Комп'ютерна інженерія»

 Анна МАКАРИК

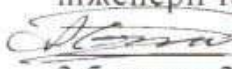
Керівник: старший викладач кафедри КІ та КБ

 Ігор ФАЛЬКОВСЬКИЙ

Рецензент: завідувач кафедри КІ та КБ, кандидат техніч-
них наук, доцент

 Андрій ЄФІМЕНКО

Державний університет «Житомирська політехніка»
Факультет інформаційно-комп'ютерних технологій
Кафедра комп'ютерної інженерії та кібербезпеки
Спеціальність 125 «Кібербезпека»
Освітня програма «Кібербезпека»

ЗАТВЕРДЖУЮ
Завідувач
кафедри комп'ютерної
інженерії та кібербезпеки
 Андрій Єфіменко
3 березня 2025 р.

ЗАВДАННЯ

на випускню кваліфікаційну роботу бакалавра

Студент: Макарик Анна Сергіївна

Тема роботи: «Проект розширеного управління груповими політиками з використанням PowerShell»

затверджена Наказом університету від 3 березня 2025 р. №97

Термін здачі студентом закінченої роботи 9 червня 2025 р.

Вихідні дані роботи: Засоби та методи автоматизації керування груповими політиками з використанням PowerShell та опис процесу їх управління в корпоративному середовищі Windows.

Обладнання та програмне забезпечення:

1. Гіпервізор VirtualBox
2. Інтегроване середовище сценаріїв PowerShell ISE
3. Образ ОС Windows Server 2019 Standart
4. Active Directory

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

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв
1	Фальковський І.Г	01.04.2025 	01.04.2025 
2	Фальковський І.Г	26.05.2025 	26.05.2025 
3	Фальковський І.Г	11.06.2025 	11.06.2025 

Студент

Анна МАКАРИК

Керівник

Ігор ФАЛЬКОВСЬКИЙ

Календарний план

№ з/п	Назви етапів випускної кваліфікаційної роботи	Термін виконання етапів роботи	Примітка
1	Постановка завдання	05.03.2025	
2	Ознайомлення з задачею	05.03.2025-10.03.2025	
3	Вивчення тематичних завдань	10.03.2025-21.03.2025	
4	Опрацювання літературних джерел	10.03.2025-21.04.2025	
5	Проектування мережі	27.03.2025-02.04.2025	
6	Налаштування системи	02.04.2025-07.04.2025	
7	Розробка скриптів GPO	07.04.2025-15.05.2025	
8	Розробка скриптів адміністрування	12.05.2025-06.06.2025	
9	Тестування проєкту	20.05.2025-09.06.2025	
10	Оформлення звіту	28.05.2025-11.06.2025	
11			

Студент  Анна МАКАРИК

Керівник  Ігор ФАЛЬКОВСЬКИЙ

АНОТАЦІЯ

Кваліфікаційна робота присвячена розробці розширеного управління груповими політиками з використанням PowerShell.

Пояснювальна записка на тему “Проект розширеного управління груповими політиками з використанням PowerShell” складається зі вступу, чотирьох розділів, головного висновку, списку використаних джерел та додатків.

Пояснювальна записка налічує 148 сторінок, з яких 31 сторінка – додатки. Основна частина охоплює 118 сторінок і містить 142 рисунки, 21 таблиці та 20 джерел, що розміщені на трьох сторінках.

В першому розділі відбулося ознайомлення з теоретичними відомостями про командну оболонку PowerShell, її переваги та недоліки у використанні для автоматизації налаштування групових політик.

У другому розділі наведено комплексну підготовку для реалізації проекту з розгортання мережі, забезпечення налаштування й управління груповими політиками.

У третьому розділі проведено застосування групових політик на домен та впровадження методів їх розширеного адміністрування за допомогою PowerShell.

У четвертому розділі проведено тестування налаштованої системи та порівняння з традиційним графічним інтерфейсом.

Висновок містить в собі результати розробки проекту розширеного управління груповими політиками з використанням PowerShell.

Ключові слова: POWERSHELL, GROUP POLICY OBJECT, ACTIVE DIRECTORY, ORGANIZATIONAL UNIT.

					ККІКБ.КР.Б – 123 – 15 – ПЗ			
Ім'я	Арк.	№ докум.	Пісочис	Дата	Проект розширеного управління гру- повими політиками з використанням PowerShell	Літ.	Арк.	Архивів
Розроб.		Макарик А.С.	16.06.25					
Керівник		Фатковський І.І.	16.06.25				4	148
Реценз.		Сфіменко А.А.	16.06.25			ФІКТ, Гр. КІ-21-2		
Н. контр.		Голован О.С.	16.06.25					
Затверд.		Сфіменко А.А.	16.06.25					

ANNOTATION

The qualification thesis is dedicated to the development of extended group policy management using PowerShell.

The explanatory note titled “Project of Extended Group Policy Management Using PowerShell” consists of an introduction, four chapters, a main conclusion, a list of references, and appendices.

The explanatory note consists of 148 pages, including 31 pages of appendices. The main part spans 118 pages and contains 142 figures, 21 tables, and 20 sources listed across three pages.

The first chapter provides an overview of the theoretical background of the PowerShell command shell, highlighting its advantages and disadvantages in automating Group Policy configuration.

The second chapter describes the comprehensive preparation for the project implementation, including network deployment and configuration and management of group policies.

The third chapter outlines the application of group policies within a domain and the implementation of extended administration methods using PowerShell.

The fourth chapter presents the testing of the configured system and compares it with the traditional graphical interface.

The conclusion summarizes the outcomes of the project on extended group policy management using PowerShell.

Keywords: POWERSHELL, GROUP POLICY OBJECT, ACTIVE DIRECTORY, ORGANIZATIONAL UNIT.

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				5
Змн.	Арк.	№ докум.	Підпис	Дата		

Зміст

ВСТУП.....	9
РОЗДІЛ 1 ТЕОРЕТИЧНІ АСПЕКТИ УПРАВЛІННЯ ГРУПОВИМИ ПОЛІТИКАМИ ЗА ДОПОМОГОЮ POWERSHELL..... 11	
1.1 Середовище реалізації автоматизованого управління GPO	11
1.1.1 Загальна характеристика групових політик	11
1.1.1 Загальні відомості про PowerShell	14
1.1.3 Огляд Windows PowerShell ISE	15
1.2 Використання PowerShell у системному адмініструванні.....	17
1.3 Основи управління груповими політиками у графічному інтерфейсі	18
1.3.1 Консоль керування груповими політиками (GMPC)	18
1.3.2 Порівняння керування GPO через графічний інтерфейс та PowerShell.....	19
РОЗДІЛ 2 ПРОЄКТУВАННЯ ТА ОРГАНІЗАЦІЯ КОРПОРАТИВНОГО СЕРЕДОВИЩА..... 22	
2.1 Вибір схеми реалізації проєкту	22
2.1.1 Модель адміністрування домену.....	22
2.1.2 Архітектура серверів та їх ролі	22
2.1.3 Визначення ролей користувачів.....	25
2.2 Розгортання інфраструктури віртуального середовища.....	29
2.2.1 Налаштування VirtualBox та створення віртуальних машин	29
2.2.2 Розгортання контролерів домену	34
2.2.3 Конфігурація основних служб (DNS, DHCP).....	38
2.3 Проєктування групових політик у корпоративному середовищі	44
2.3.1 Структуризація GPO за рівнями.....	44
2.3.2 Планування та категоризація GPO.....	46
2.3.3. Механізми успадкування та виключення політик.....	51
РОЗДІЛ 3. РЕАЛІЗАЦІЯ АВТОМАТИЗОВАНОГО УПРАВЛІННЯ GPO..... 56	
3.1 Підготовка інфраструктури для автоматизації керування груповими політиками.....	56
3.1.1 Впровадження модулів PowerShell для роботи з GPO.....	56
3.1.2 Налаштування групових політик у доменному середовищі.....	64
3.2 Автоматизація адміністрування через PowerShell.....	68
3.2.1 Створення PowerShell-скриптів для управління GPO.....	68
3.2.2 Реалізація скриптів для масового внесення зміни у політики.....	69

3.2.3	Проектування засобів управління дозволами та безпекою GPO.....	73
3.3	Розробка сценаріїв моніторингу та підтримки GPO	77
3.3.1	Налаштування планових задач для автоматизації менеджменту GPO	77
3.3.2	Розробка скриптів для генерація звітів про застосування політик	80
РОЗДІЛ 4. ТЕСТУВАННЯ ТА АНАЛІЗ ЕФЕКТИВНОСТІ АВТОМАТИЗАЦІЇ..		85
4.1	Тестування реалізованої системи	85
4.1.1	Перевірка коректності роботи запроваджених GPO	85
4.1.2	Тестування PowerShell-скриптів для адміністрування	92
4.2	Оцінка ефективності автоматизованого управління GPO	108
4.2.1	Порівняння швидкості ручного та автоматизованого адміністрування ...	108
4.2.2	Аналіз переваг та обмежень використання PowerShell.....	109
4.3	Висновки щодо впровадження автоматизованого управління GPO	111
ВИСНОВКИ		113
СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ		114
ДОДАТКИ		117

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

ОС – операційна система

ПЗ – програмне забезпечення

ВМ – віртуальна машина

AD - Active Directory

GPO – об'єкт групової політики

OU – організаційні одиниці

GI – графічний інтерфейс

GPMC - Group Policy Management Console

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				8
Змн.	Арк.	№ докум.	Підпис	Дата		

ВСТУП

Актуальність теми. Усе більш актуальним стає питання забезпечення належного рівня управління конфігурацією користувачів і комп'ютерів як одного з основних чинників ефективного функціонування ІТ-інфраструктури. Зокрема, в умовах адміністрування мережі на базі серверних операційних систем сімейства Windows доцільним є використання служби каталогів Active Directory, яка забезпечує ієрархічну модель керування обліковими записами, ресурсами та політиками безпеки.

Одним із інструментів у складі Active Directory є механізм групових політик (Group Policy Object, GPO), що дозволяє налаштовувати широкий спектр параметрів – від прав доступу користувачів до ресурсів до конфігурацій системної безпеки. Застосування GPO дає змогу значно спростити процес адміністрування, особливо в масштабних організаціях із великою кількістю користувачів та пристроїв, шляхом уніфікації.

Разом з тим, постає проблема раціонального використання адміністративних ресурсів, оскільки традиційне конфігурування групових політик може бути тривалим і трудомістким. У зв'язку з цим у даній кваліфікаційній роботі досліджуються шляхи оптимізації процесів управління GPO, зокрема за рахунок впровадження автоматизованих скриптів на базі PowerShell, що дозволяють підвищити ефективність адміністрування та скоротити час виконання рутинних завдань.

Метою кваліфікаційної роботи є обґрунтування методів реалізації розширеного управління груповими політиками з використанням PowerShell.

Визначена мета обумовлює наступні завдання:

- Створення групових політик за допомогою скриптів PowerShell
- Адміністрування групових політик з використанням сценаріїв PowerShell
- Порівняння використання традиційного графічного інтерфейсу для встановлення GPO з PowerShell
- Створення лабораторного стенду для демонстрації виконаної роботи

Об'єктом дослідження є процес управління груповими політиками в корпоративному середовищі Windows.

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				9
Змн.	Арк.	№ докум.	Підпис	Дата		

Предметом дослідження є засоби та методи автоматизації керування груповими політиками з використанням PowerShell.

У процесі виконання кваліфікаційної роботи було розроблено проєкт розширеного управління GPO, що передбачає створення, налаштування, моніторинг політик за допомогою мови сценаріїв PowerShell та також демонстрація тестування створених скриптів на лабораторному стенді.

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				
Змн.	Арк.	№ докум.	Підпис	Дата		10

РОЗДІЛ 1 ТЕОРЕТИЧНІ АСПЕКТИ УПРАВЛІННЯ ГРУПОВИМИ ПОЛІТИКАМИ ЗА ДОПОМОГОЮ POWERSHELL

1.1 Середовище реалізації автоматизованого управління GPO

1.1.1 Загальна характеристика групових політик

Групові політики (Group Policy) – Один з елементів керування в Active Directory, що дозволяє адміністратору керувати стандартизованими політиками на декількох комп'ютерах та користувачах одночасно. Автоматизована система розгортання, заснована на цій можливості, усуває тисячі завдань ручної конфігурації, впроваджуючи стандартизовані вимоги безпеки з мінімальним втручанням людини.

Групові політики в операційній системі Windows формуються у вигляді ієрархічної моделі. Політики поширюються від рівня контролера домену до визначених організаційних одиниць (OU). Доступ користувачів до ресурсів операційної системи здійснюється через службу Active Directory[1], яка формує перелік активних політик відповідно до ідентифікаційних даних користувача та характеристик його кінцевого хосту. Після цього параметри системи пристрою застосовуються автоматично, відповідно до заданих параметрів GPO.

Групові політики можуть бути встановлені для широкого спектру налаштувань: від зміни зовнішнього виду інтерфейсу шляхом, наприклад, заборони змінювати одне задане адміністратором фонове зображення до ключових аспектів безпеки, включаючи політику паролів, блокування портів фаєрволу, контролю запуску програмного забезпечення та управління доступу до мережевих ресурсів.

Впровадження групових політик надає організаціям дві критично важливі переваги – підвищення рівня безпеки, а також оптимізацію операційної діяльності. Підрозділи, відповідальні за кіберзахист, використовують політики для централізованого впровадження стандартів безпеки, тим самим уніфікуючі захист кінцевих пристроїв. Адміністратори отримують інструмент для автоматизованого розгортання програм, налаштування служб і керування робочим середовищем без потреби у втручанні користувачів.

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				
Змн.	Арк.	№ докум.	Підпис	Дата		11

Наявні кілька видів групових політик залежно від їх призначення та застосування:

- Локальні групові політики – функціонують як автономний засіб керування параметрами системи на рівні окремого комп'ютера, не потребуючи наявності інфраструктури Active Directory.
- Групові політики домену – є головним інструментом усіма пристроями, що були інтегровані у домен.
- Групові політики організаційних одиниць – дозволяють отримати детальний контроль над окремими відділами або операційними функціями в організації.
- Групові політики на рівні сайту – спрямовує налаштування на сайти Active Directory - логічні об'єднання IP-підмереж, які відображають фізичне розташування підрозділів організації.
- Політики безпеки – відіграють значну роль у забезпеченні єдиної системи захисту інформаційних ресурсів на рівні всієї організації.
- Розширене керування груповою політикою (Advanced Group Policy Management, AGPM) – функціонує як комплексна платформа адміністрування, яка впроваджує контроль версій, відстеження змін і механізми затвердження змін для управління груповими політиками.

Різноманіття типів групових політик – доменних, локальних, політик рівня організаційних одиниць та сайтів – відображає гнучкість і масштабованість механізмів управління у середовищі Windows. В основі реалізації кожного з цих типів політик лежить базова структурна одиниця – об'єкт групової політики він являє собою організований набір параметрів, який встановлює правила, щодо поведінки користувачів та кінцевих пристроїв у домені Active Directory.

Політики GPO можуть бути різними в залежності від об'єкта до якого вони застосовуються.

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				
Змн.	Арк.	№ докум.	Підпис	Дата		12

Комп'ютерні політики визначають, як працює операційна система на комп'ютерах у мережі, не залежно від того хто користуватиметься цим пристроєм. Наприклад, до них входять налаштування безпеки, програм, параметрів мережі, запуску додатків або скриптів під час завантаження комп'ютера.

Користувацькі політики застосовуються до окремих користувачів, активні лише при заходженні даного користувача у систему і включають налаштування їхнього робочого середовища, призначених папок, доступу до програм та сценаріїв для входу і виходу з системи.

Групові політики активується в визначені моменти життєвого циклу системи – під час увімкнення комп'ютера або авторизації користувача. Даний режим застосування політики має назву foreground application. У випадку комп'ютера конфігураційні параметри впроваджуються ще до завершення завантаження операційної системи, що гарантує її відповідність корпоративним стандартам перед початком роботи. Хоча ще є спосіб примусово оновити політики без необхідності перезавантаження пристрою чи виходу користувача з системи.

Процес застосування GPO може виконуватися у двох режимах: синхронному або асинхронному. У першому випадку система зупиняє подальше завантаження, доки всі політики не буде повністю застосовано, забезпечивши повну готовність середовища до використання. В асинхронному режимі система дозволяє завершити вхід або запуск незалежно від повного впровадження налаштувань, які тоді застосовуються у фоновому режимі. Це дозволяє зменшити час очікування для користувача, водночас забезпечуючи поступове оновлення параметрів.

Фонове застосування політик дає можливість системі регулярно перевіряти відповідність поточних налаштувань до актуальних параметрів, заданих у GPO. У результаті зміни можуть поширюватися по мережі без необхідності перезавантаження клієнтських пристроїв, що значно підвищує ефективність та динамічність централізованого адміністрування.

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				13
Змн.	Арк.	№ докум.	Підпис	Дата		

1.1.1 Загальні відомості про PowerShell

PowerShell - це оболонка командного рядка, а також потужна мова сценаріїв, розроблена компанією Microsoft спеціально для автоматизації управління системами та їх адміністрування. Побудована на базі платформи .NET Framework і загальномовному виконуючому середовищі .NET Framework (CLR) тому мова скриптів забезпечена тісною інтеграцією з іншими компонентами Windows.

На відміну від традиційних командних оболонок, які працюють виключно з текстовими рядками, PowerShell оперує об'єктами .NET[2]. Кожен об'єкт має властивості та методи, які можна виконати над ним. Такий підхід значно спрощує обробку даних, оскільки немає необхідності парсити текстовий вивід – достатньо працювати з об'єктною моделлю. Це дозволяє легко передавати вихід однієї команди як вхід до іншої, реалізуючи концепцію конвеєра об'єктів.

PowerShell підтримує чотири типи команд, а саме: командлети, сценарії, функції та методи.

- Командлети (cmdlets) – це однокрокові команди, призначені для виконання конкретних операцій у межах сценаріїв або безпосередньо через інтерфейс командного рядка й становлять основу функціональності PowerShell. Незважаючи на простоту окремих cmdlets, у комбінації створюють потужний інструмент для взаємодії з системою.
- Функції – користувацький послідовний код написаний мовою PowerShell. Як у будь-якій іншій мові програмування, функції можуть приймати параметри і повертати необхідні значення. Існують два типи функцій: базові та розширені. Базові функції мають лише набір фігурних дужок та тіло, на противагу їм – розширені містять базу попередників та підтримують додаткову функціональність, як наприклад підтримку атрибутів і обробку параметрів у стилі командлет.
- Сценарії або скрипти – звичайний текстовий файл з розширенням .ps1 та містять у собі одну або ж кілька команд PowerShell. Вони дозволяють автоматизувати щоденні завдання, включаючи адміністрування мереж, створення звітів або керування службами.

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				
Змн.	Арк.	№ докум.	Підпис	Дата		14

- Методи - дії або функціональні можливості, які пов'язані з об'єктами й дозволяють виконувати певні операції над цими об'єктами. Вони є складовою об'єктно-орієнтованої моделі PowerShell і, на відміну від командлетів або функцій, зазвичай реалізовані на інших мовах програмування (наприклад, C#) як частина бібліотек .NET.

Таблиця – 1.1.2.1

Перевага	Опис	Недолік	Опис
Об'єктна модель	Працює з об'єктами .NET, а не з текстом	Високий поріг входу	Складний синтаксис, незвичний для початківців
Інтеграція з Windows	Доступ до WMI, реєстру, служб, AD	Низька продуктивність	Неоптимальний при обробці великих даних
Розширюваність	Можливість створення модулів, cmdlet-ів	Залежність від .NET	Обмеження в системах без підтримки .NET
Повноцінна мова сценаріїв	Підтримка циклів, умов, обробки помилок	Обмеження на Linux (частково усунуто)	Деякі модулі працюють тільки в Windows
Автоматизація DevOps	Використовується в CI/CD, Azure, Docker	Відсутність GUI	Вимагає роботи з командним рядком

1.1.3 Огляд Windows PowerShell ISE

Ознайомившись з архітектурними особливостями PowerShell як об'єктно-орієнтованої оболонки командного рядка та мови сценаріїв, доцільним є перехід до розгляду засобів, які полегшують розробку та налагодження скриптів. Одним із таких інструментів є Windows PowerShell ISE (Integrated Scripting Environment) – візуальне середовище, створене для підвищення ефективності роботи з PowerShell-скриптами.

Windows PowerShell ISE – це інтегроване графічне середовище, яке слугує хост-додатком для PowerShell та забезпечує користувача функціональністю, що виходить за межі класичної командної консолі[3]. Головною метою використання PowerShell ISE є спрощення процесу написання, виконання, тестування та налагодження сценаріїв. Вперше середовище було представлено у версії PowerShell 2.0, а

в PowerShell 3.0 зазнало суттєвої модернізації як з функціональної, так і з інтерфейсної точки зору.

PowerShell ISE має ряд переваг, що робить його використання більш доцільним у задачах пов'язаних з адмініструванням системи:

- Багаторядкове редагування коду, котре дозволяє зручно працювати з великими фрагментами коду.
- Синтаксичне підсвічування – покращує читабельність написаних скриптів та допомагає швидше знайти помилки.
- Наявне автозавершення для командлетів, параметрів, змінних і методів.
- Контекстна довідка – довідка про доступні у використанні командлети.
- Встановлення breakpoints та виконання коду частинами, що полегшує діагностику та пошук помилок під час розробки.

Загалом PowerShell ISE і PowerShell пропонують користувачу практично однакові можливості написання скриптів для середовища Windows, але все ж, як ми могли побачити вище, є значне відмінність – зручність. PowerShell є простішим, хоч і дещо обмеженішим за функціональністю для використання, в то й же час як PowerShell ISE надає гучкіший функціонал редагування і функціонування.

Незважаючи на численні переваги PowerShell ISE більше не знаходиться в активній розробці. Microsoft припинила додавання нових функцій до середовища після випуску Windows PowerShell версії 5.1. Проте воно залишається частиною операційної системи Windows і все ще офіційно підтримується для виправлень безпеки та критичних оновлень.

На заміну Microsoft рекомендують використовувати середовища розробки як наприклад VS Code з встановленим додатковим розширенням PowerShell Extension, у котрому наявна кросплатформенна підтримка, взаємодія з Git та розширений функціонал PowerShell Core s PowerShell 7+.

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				
Змн.	Арк.	№ докум.	Підпис	Дата		16

1.2 Використання PowerShell у системному адмініструванні

PowerShell, а особливо його покращена версія PowerShell ISE є важливим інструментом у плані системного адміністрування, надаючи користувачу високий рівень автоматизації та зручний контроль на операційною системою Windows. Завдяки роботі з об'єктами .NET Framework[4] і тісній інтеграції з компонентами ОС, він надає змогу ефективно виконувати широкий спектр завдань – від управління користувачами у Active Directory до налаштування мережевих параметрів домену.

Напевно головною перевагою PowerShell є можливість масового налаштування. Наприклад, адміністратор може за допомогою кількох рядків коду та командлет отримати інформацію про усі комп'ютери у домені, перевірити наявність оновлень та змінити налаштування або запустити необхідні служби. Тому використання PowerShell значно знижує час необхідний для виконання рутинних завдань, зменшивши при цьому можливість виникнення помилок при ручному втручанні.

Також його активно використовують для створення скриптів, що відповідають за автоматизацію процесів, як створення резервних копій, налаштування групових політик, моніторингу продуктивності системи, обробці журналів подій і тому подібного. Додатковим бонусом є можливість підключити зовнішні бібліотеки для взаємодії з сервісами як з Microsoft 365, Exchange, Azure та іншими.

У питанні безпеки PowerShell надає зручні інструменти для контролю прав доступу, перевірки цілісності системи та віддаленого адміністрування, включно з можливістю запуску скриптів через PowerShell Remoting. Це дозволяє централізовано керувати численними пристроями без фізичного доступу до кожного з них.

Тому PowerShell можна вважати, незамінною складовою інфраструктури, навіть попри відсутність оновлень від Microsoft. Адміністратору який прагне підвищити продуктивність, забезпечити стабільність роботи систем і мінімізувати людський фактор при виконанні повсякденних завдань варто замислитися про використання вже вбудованого у ОС середовища розробки.

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				
Змн.	Арк.	№ докум.	Підпис	Дата		17

1.3 Основи управління груповими політиками у графічному інтерфейсі

1.3.1 Консоль керування груповими політиками (GPMC)

Консоль керування груповими політиками (GPMC Group Policy Management Console) – це комплексний інструмент для адміністрування групових політик. Адміністратори використовують GPMC для виконання всіх завдань з керування груповими політиками, за винятком налаштування окремих параметрів у самих об'єктах групової політики (GPO), яке виконується за допомогою редактора локальної групової політики. Ви можете встановити GPMC через Диспетчер серверів на сервері наступним чином. [\[5\]](#)

GPMC пропонує кілька основних функцій для управління GPO:

- **Централізоване управління** – централізоване керування GPO у мережі дозволяє створювати, редагувати, прив'язувати та видаляти політики з єдиного застосунку без необхідності бути підключеним до окремих контролерів домену, що економить час і ресурси.
- **Зручний інтерфейс** – Простий для сприйняття користувачем інтерфейс полегшує роботу користувача за допомогою організації об'єктів у ієрархічну структуру, що дає змогу швидко знаходити потрібні політики для роботи. Якщо ж виникають труднощі, GPMC може запропонувати вбудовану систему пошуку та фільтрації.
- **Звітність і аналіз** – Консоль надає додаткові можливості з генерації розширених звітів, котрі міститимуть в собі інформацію про групові політики, пристрої чи облікові записи користувачів на яких вони встановлені та налаштовані параметри, допомагаючи оцінити вплив GPO на систему та завчасно передбачити потенційні помилки.
- **Копіювання GPO між доменами** – Імпортування та копіювання GPO між доменами та лісами за використовуючи таблиці міграції, котрі дозволяють зіставляти посилання на користувачів, групи й комп'ютери з вихідного GPO з новим значенням в цільовому.

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				18
Змн.	Арк.	№ докум.	Підпис	Дата		

- Прототипування групової політики – Використання змодельованих даних підсумкового набору політик (RSoP), щоб протестувати групову політику перед впровадженням її у робоче середовище.

Таким чином, GPMC є універсальним рішенням, яке поєднує простоту використання із високим рівнем функціональності, забезпечуючи ефективне управління груповими політиками у середовищі Active Directory.

1.3.2 Порівняння керування GPO через графічний інтерфейс та PowerShell

Управління груповими політиками є одним з головних завдань для адміністраторів Active Directory, і для цього існує декілька підходів та інструментів, кожен із яких має свої переваги та недоліки. Найпоширеніші варіанти включають використання графічного інтерфейсу, таких як редактор групової політики, і автоматизованих засобів, наприклад PowerShell.

Таблиця – 1.3.2.1

Переваги та недоліки у GPMC

Графічний інтерфейс (GPMC)	
Переваги	Недоліки
GPMC надає зручний інтерфейс, що дозволяє легко виконувати базові завдання, такі як створення, редагування та зв'язування GPO	Обмежена автоматизація. Виконання повторюваних завдань, таких як масове редагування чи створення політик, є трудомістким.
Мінімальний поріг входження. Для роботи з GPMC не потрібні глибокі технічні знання або програмування	Залежність від ручних дій. Велика кількість рутинної роботи підвищує ймовірність помилок.
GPMC дозволяє переглядати ієрархічну структуру політик і легко зрозуміти, як вони застосовуються в Active Directory.	Відсутність підтримки сценаріїв. Неможливість автоматизувати складні сценарії або інтегрувати з іншими інструментами

Переваги та недоліки у PowerShell

PowerShell	
Переваги	Недоліки
Гнучкість. PowerShell дозволяє виконувати широкий спектр завдань — від створення політик до їхнього розгортання та аудиту.	Ризик помилок. Неправильно написаний сценарій може призвести до масштабних помилок, наприклад, до неправильного налаштування політик у всій організації.
Завдяки командлетам, таким як New-GPO, Set-GPPermission і Get-GPOReport, можна автоматизувати рутинні завдання, що значно підвищує продуктивність.	Відсутність графічного інтерфейсу. Усе виконується через командний рядок або сценарії, що може бути незручним для новачків
PowerShell легко інтегрується з іншими системами та сервісами, такими як Microsoft Graph або сторонні інструменти моніторингу.	

Нижче наведено порівняльну таблицю, яка висвітлює основні відмінності між використанням GPMC та PowerShell або ж PowerShell ISE у процесах адміністрування групових політик. Представлено особливості кожного підходу, щоб усвідомлено обрати найбільш ефективний інструмент відповідно до завдань, масштабів мережі та рівня автоматизації.

Таблиця 1.3.2.2

Порівняння GPMC з PowerShell у адмініструванні GPO

Характеристика	GPMC	PowerShell
Створення та редагування політик	Зручний для разових завдань і простих налаштувань.	Переважає при необхідності масового створення або редагування політик.
Аудит та звіти	Дозволяє переглядати результати політик у графічному вигляді.	Дозволяє створювати докладні звіти, наприклад, за допомогою командлета Get-GPOReport.
Моніторинг змін	Має базові інструменти для відстеження змін.	Дає змогу розробляти сценарії для відстеження змін і надсилання сповіщень у реальному часі.
Автоматизація складних сценаріїв	Не підтримує автоматизацію.	Ідеально підходить для виконання таких завдань завдяки підтримці циклів, умовних конструкцій та інтеграції з іншими інструментами.

Уважно проаналізувавши таблиці 1.3.2.1 та 1.3.2.2 ми можемо зробити висновок, що для невеликих організацій та простих завдань з невеликою кількістю GPO краще підійде GPMC через зручність у графічному інтерфейсі та загальній простоті використання. В цьому і криється його суттєвий недолік – неможливість масштабованого та автоматизованого налаштування GPO. На противагу, PowerShell виправданий інструмент для управління груповими політиками у великих корпоративних середовищах з розгалуженою структурою організаційних одиниць, оскільки він суттєво оптимізує адміністративні завдання та мінімізує ризик виникнення помилок.

Висновки до першого розділу

У ході написання цього розділу кваліфікаційної роботи було розглянуто теоретичні відомості про використання PowerShell для розширеного управління груповими політиками.

Було вивчено основи роботи з PowerShell, включно з його інтегрованим середовищем PowerShell ISE, яке забезпечує зручність написання, тестування та налагодження сценаріїв. Наостанок у цьому розділі розглянуто теоретичні аспекти групових політик, їхню структуру, а також методи їх застосування та налаштування.

При порівнянні кількох інструментів та підходів до управління груповими політиками, PowerShell виявився одним із найгнучкіших і найефективніших засобів.

Отримані теоретичні знання лягли в основу проектування автоматизованого рішення в наступних розділах.

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				
Змн.	Арк.	№ докум.	Підпис	Дата		21

РОЗДІЛ 2 ПРОЄКТУВАННЯ ТА ОРГАНІЗАЦІЯ КОРПОРАТИВНОГО СЕРЕДОВИЩА

2.1 Вибір схеми реалізації проєкту

2.1.1 Модель адміністрування домену

Вибір моделі адміністрування домену тісно пов'язаний із архітектурою доменної структури, оскільки саме вона визначає, як організовується управління обліковими записами користувачів та адмінів, ресурсами та політиками безпеки в межах мережі. У проєктованій системі передбачено використання моделі з одним доменом, яка є найпростішою у впровадженні й підтримці, через ресурсні обмеження хост-системи. Створений домен з назвою “makaryk.net” є кореневим доменом лісу.

Модель лісу з одним доменом зменшує складність адміністрування, надаючи наступні переваги[6]:

- Всі контролери домену можуть виконати автентифікацію, будь-якого користувача.
- Всі контролери домену можуть слугувати глобальними каталогами, щоб користувачам не довелося розміщувати додатковий сервер глобальних каталогів.

У межах побудови моделі на основі одного домену реалізується багаторівнева ієрархія адміністративного доступу, котра дозволяє охопити різні ролі тим самим делегуючи деякі обов'язки головного адміністратора на інших користувачів домену. Це у свою чергу позитивно відображається загальній ефективності роботи організації, зберігаючи помірно високий рівень контролю і безпеки.

2.1.2 Архітектура серверів та їх ролі

Напевно головною задачею адміністратора є побудова надійної доменної мережі. У межах даного проєкту з урахуванням обмежених ресурсів, було реалізовано базову, проте цілком ефективну, а головне відмовостійку архітектуру серверної інфраструктури, що відповідає усім потребам мережевого середовища для організації середнього і нижчого масштабу.

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				
Змн.	Арк.	№ докум.	Підпис	Дата		22

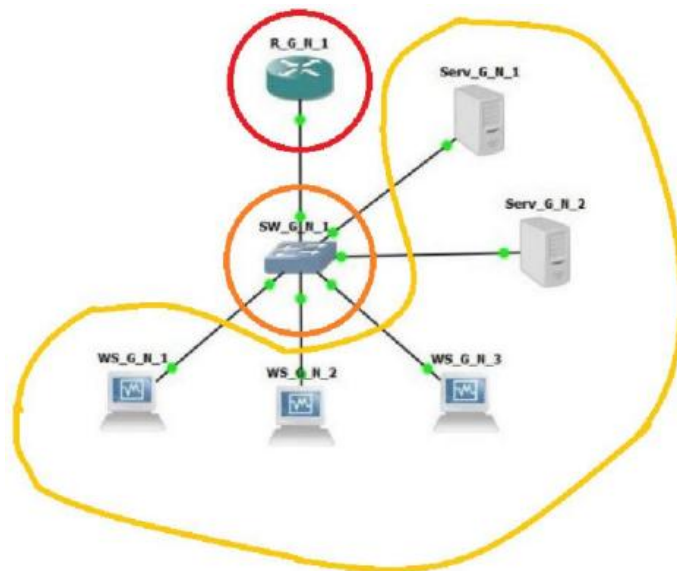


Рисунок 2.1.2.1 – Умовна топологія мережі

Детальніше розглянемо топологію корпоративної мережі, щоб визначитися з наявними елементами. Як можна зрозуміти по наведеному вище (рис. 2.1.2.1) основну архітектуру становлять два сервери та три кінцевих хости. Сервери у домені можуть приймати одну чи кілька ролей[8] (табл. 2.1.2.1).

Таблиця 2.1.2.1

Основні ролі серверів у домені

Роль	Характеристика
Контролер домену	Сервер з якого здійснюється головне керування доменом
Сервер друку	Має доступ до мережевих принтерів та керує їх роботою
Сервер додатків	Хостить та виконує корпоративні або веб-додатки
Сервер терміналів	Відповідає за віддалений доступ до робочих столів
Сервер віддаленого доступу	Дозволяє підключитися до внутрішньої мережі
VPN-сервер	Створює VPN з'єднання між користувачем та мережею
Вузол кластера серверів	Група серверів для забезпечення підвищеної надійності
Файловий сервер	Надає доступ до збережених файлів домену
DHCP-сервер	Відповідає за реалізацію протоколу DHCP
DNS-сервер	Перетворює доменні імена на IP-адреси
Резервний контролер	Дублює основний контролер для відмовостійкості

У нашому випадку є один контролер домену WinServ_ki212_mas, який виконує усі основні ролі в інфраструктурі, та резервний контролер WinServ_res_ki212_mas, котрий відповідальний за функції дублювання й відмовостійкості, щоб у разі критичної помилки доступ домену не втратився. Обидва сервери об'єднані в єдиний домен та взаємодіють між собою через служби Active Directory.

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				
Змн.	Арк.	№ докум.	Підпис	Дата		23

До головних ролей головного контролера домену належать:

- Active Directory Domain Services - служба каталогів, яка керує обліковими записами користувачів, комп'ютерами та іншими ресурсами в мережі домену Windows.
- Domain Name System – інтегровано у AD DNS-сервер відповідає за надійну і швидку резолюцію імен.
- Group Policy Management – контролер домену відповідає за створення та менеджмент GPO.
- DHCP-сервер - У невеликих мережах доцільно поєднувати функціональність роздачі IP-адрес безпосередньо на одному з серверів.

Резервний сервер виконує роль додаткового контролера домену (Additional Domain Controller) і забезпечує надійність та безперервність роботи мережі. Його головні завдання включають:

- Реплікація Active Directory – AD DS автоматично реплікується між основним і резервним контролерами для забезпечення синхронізації та цілісності даних.
- Резервний DNS-сервер - Працює як дублюючий вузол, що підвищує надійність і відмовостійкість при обробці DNS-запитів.
- Аутентифікації – Якщо головний доменний контролер вийде з ладу, WinServ_res_ki212_mas перейме на себе повноваження з аутентифікації користувачів та обслуговування мережевих ресурсів.
- Функції контролю доступу і відновлення - На сервері реалізовані процедури регулярного резервного копіювання критичних конфігурацій і служб, що дозволяє швидко відновити роботу системи у разі збою.

Обидва сервери розміщено в одному логічному домені з налаштованою автоматичною реплікацією служб каталогу. Такий підхід дозволяє:

- забезпечити відмовостійкість критично важливих сервісів;
- спростити адміністрування;

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				24
Змн.	Арк.	№ докум.	Підпис	Дата		

- зберегти актуальність і цілісність інформації про користувачів та пристрої;
- уникнути повного виходу мережі з ладу в разі аварії одного з серверів.

У випадку подальшого розширення, шляхом додавання до домену деякої кількості серверів та користувачів дана архітектура дозволить гнучко масштабуватися, оскільки закладено основу для зручного делегування ролей та розподілу навантаження між серверами, тому збільшення кількості контролерів, впровадження спеціалізованих серверів і розширення організаційної структури в Active Directory відбуватиметься без потреби у суттєвих змінах готової мережі інфраструктури.

2.1.3 Визначення ролей користувачів

Локальні облікові записи користувачів за замовчуванням - це вбудовані облікові записи, які створюються автоматично під час встановлення контролера домену Windows Server і створення домену [8]. Користувачі у середовищі Active Directory є об'єктами, котрі репрезентують фізичних осіб з доступом до ресурсів домену. Облікові записи забезпечують ідентифікацію шляхом використання механізму авторизації та автентифікації. Крім того вони можуть слугувати як інструмент контролю доступу та обмеження впливу користувачів на організацію мережі.

Active Directory ці обмеження визначаються за допомогою низки вбудованих облікових записів, кожен з яких має специфічні повноваження та сценарії використання. Основні ролі включають:

- Administrator – обліковий запис із найвищим рівнем привілеїв у домені. Використовується для виконання всіх адміністративних задач, як створення користувачів, керування групами, конфігурація політик безпеки, розгортання служб тощо. Даний обліковий запис не рекомендується використовувати для повсякденної роботи з міркувань безпеки.
- Guest – обмежений обліковий запис, призначений для тимчасового доступу до системи без надання можливості змінювати конфігурацію або встановлювати програмне забезпечення. Має мінімальні дозволи і, як правило, відключений за замовчуванням.

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				
Змн.	Арк.	№ докум.	Підпис	Дата		25

- HelpAssistant – обліковий запис, що використовується службою «Віддалена допомога Windows» для надання тимчасового доступу до комп'ютера. Цей обліковий запис активується лише під час сесій дистанційної підтримки і не повинен використовуватись для регулярного адміністрування.

Варто загадати існування спеціалізованого облікового запису KRBtgt. Він створюється автоматично під час розгортання домену в AD та не призначений для використання адміністраторами чи звичайними користувачами. Справа в тому, що KRBtgt використовується для підписування й шифрування квитків Kerberos, тому з міркувань безпеки даний обліковий запис мусить залишатися деактивованим для входу та не повинен бути використаним для будь-яких цілей.

Отже давайте детальніше розглянемо користувачів створених у мережі домену makaryk.net (рис. 2.1.3.1). Вони були розміщені у відповідних організаційних одиницях, що відображають умовну ієрархію організації. Тож як можемо побачити наявного усього 12 облікових записів.

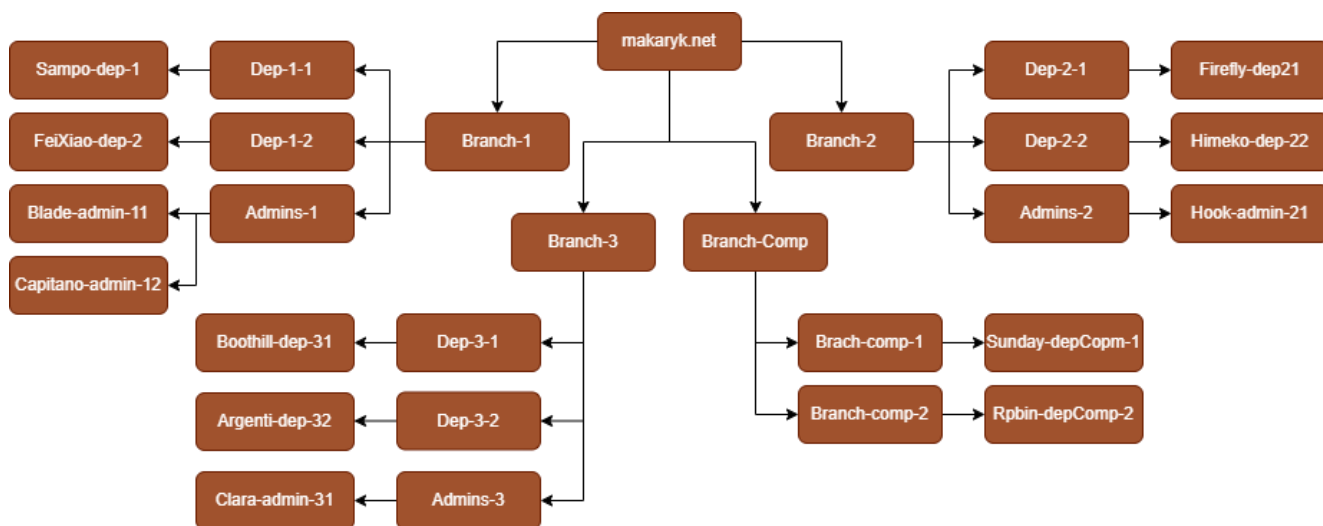


Рисунок 2.1.3.1 – Ієрархічна організація домену з OU та користувачами

Щоб створити будь-якого користувача, необхідно скористатися службою Active Directory Users and Computers. У ній надається можливість задати основні параметри облікового запису, як ім'я користувача, унікальний логін, початковий пароль, приналежність до організаційної одиниці, відразу призначити членство в групах безпеки (рис. 2.1.3.2). Крім того, можна налаштувати додаткові атрибути,

наприклад контактна інформація, посада, місце розташування та інші властивості, які використовуються для організаційної структури.

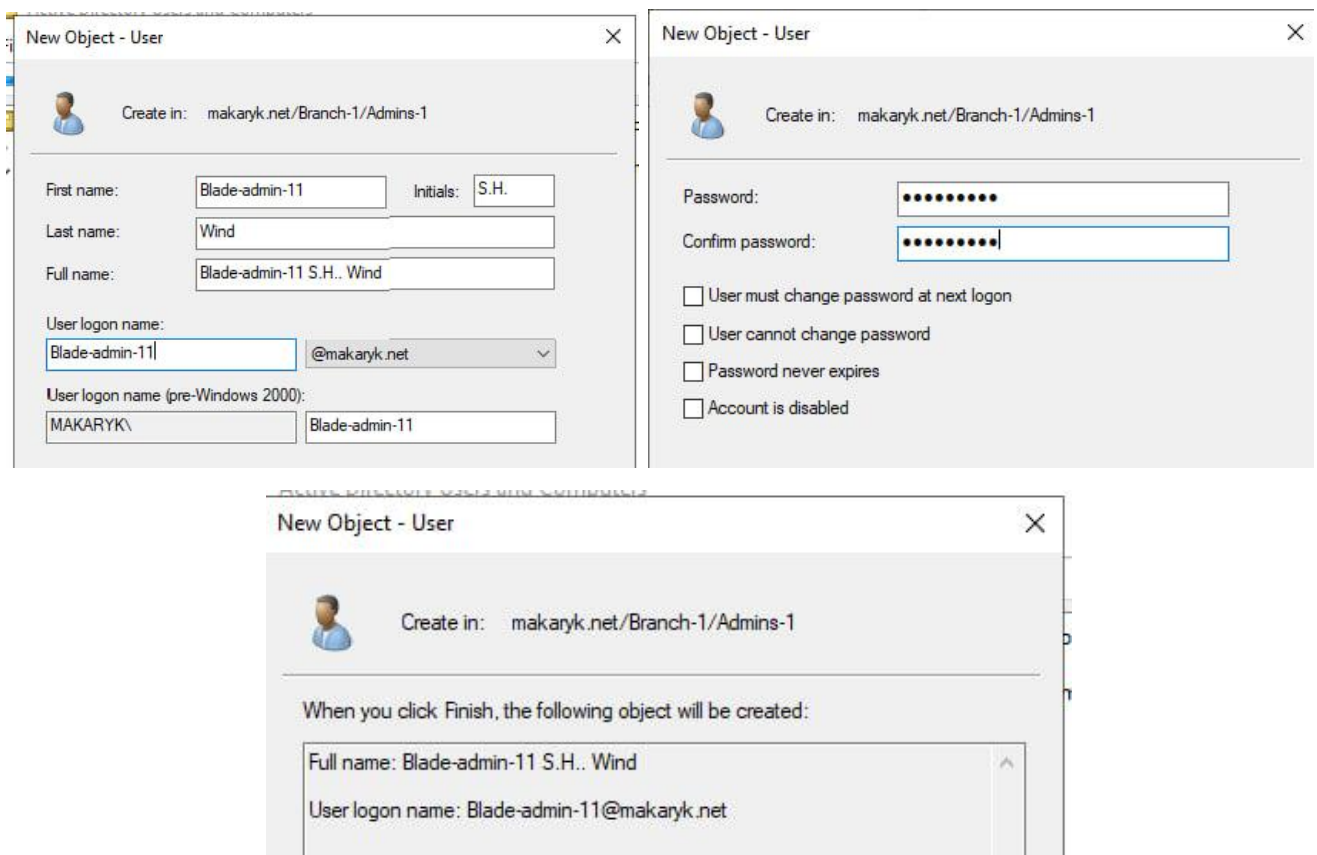


Рисунок 2.1.3.2 – Створення користувача у відповідній OU

Після створення нового користувача доцільно задати йому членство у групі безпеки залежно від його ролі у організації. Самі групи безпеки є способом об'єднання облікових записів користувачів та комп'ютерів та інших груп у керовані одиниці. Вони використовуються для призначення прав доступу до ресурсів домену, і є основним інструментом у політиках безпеки та контролі доступом. Це дозволяє централізовано керувати доступом до ресурсів та спрощує підтримку структури прав. У разі зміни посадових обов'язків користувача достатньо змінити його членство в групах, не переналаштовуючи права вручну.

Таблиця – 2.1.3.1

Налаштовані групи безпеки у домені

Група	Призначення
Administrators	Повний контроль над доменом і всіма об'єктами. Використовується для основних адмінів.

Група	Призначення
Domain Admins	Автоматично має права адміністратора на всіх комп'ютерах домену. Основна привілейована група.
Users	Базова група для всіх користувачів. Задає стандартні привілеї.
DNSAdmin	Група для адміністрування DNS
Domain Users	Група, до якої автоматично додається кожен новий користувач. Зазвичай використовується як базова.
Guests	Група з мінімальними правами.
Group Policy Creator Owners	Дозволяє створювати та відповідно редагувати, прив'язувати і видаляти GPO.

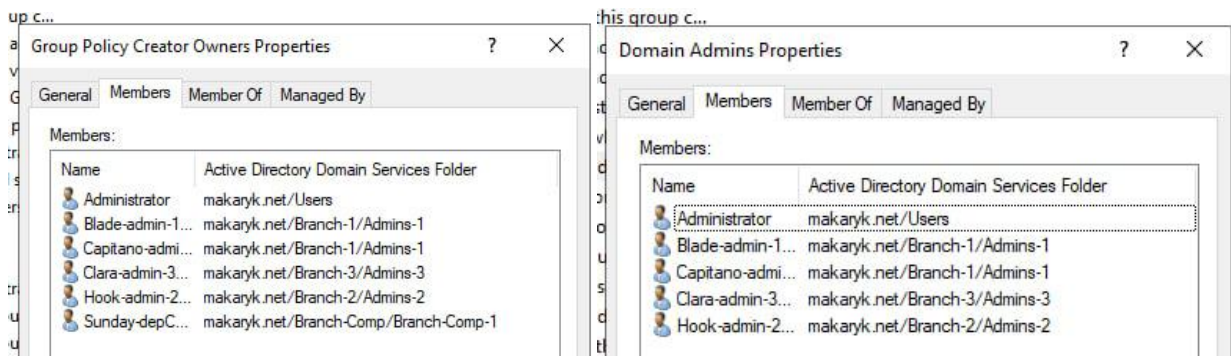


Рисунок 2.1.3.3 – Члени групи Group Policy та Domain Admin

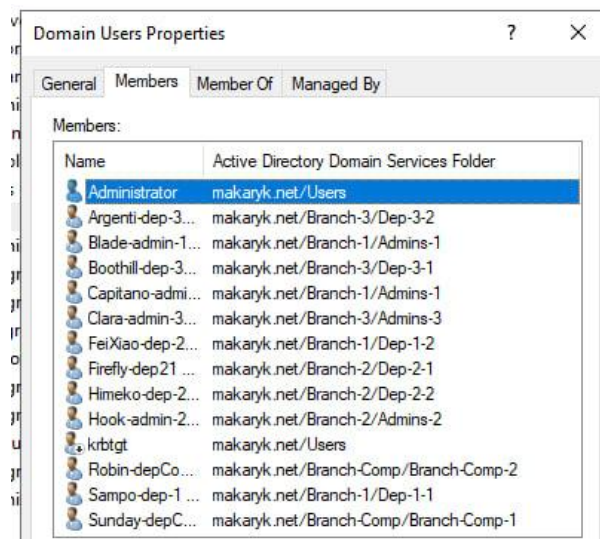


Рисунок 2.1.3.4 – Члени групи Domain User

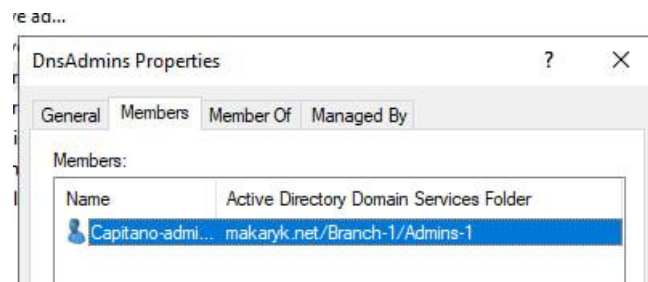


Рисунок 2.1.3.5 – Члени групи DnsAdmins

2.2 Розгортання інфраструктури віртуального середовища

2.2.1 Налаштування VirtualBox та створення віртуальних машин

Перед створенням образів віртуальних машин давайте налаштуємо мережу у якій вони працюватимуть, для цього за даними таблиці, що демонструє адресації для мережі, включаючи основні діапазони адрес і їх призначення, що допомагає організувати мережу з урахуванням оптимальних практик управління IP-адресами (табл. 2.2.1.1).

Таблиця 2.2.1.1

Мережа	Інт	ІР	Маска	/24
Мережа А	х	192.168.6.0	255.255.255.0	24
NAT-Makaryk	х	192.168.6.1	255.255.255.0	24
WinServ-ki212-mas	Мер-ад	192.168.10	255.255.255.0	24
	Шлюз	192.168.6.1	х	х
WinServ-res-ki212-mas	Мер-ад	192.168.20	255.255.255.0	24
	Шлюз	192.168.10	х	х
WS-Win10	Мер-ад	DCHP	х	24
	Шлюз	х	х	х
Win11-23H2	Мер-ад	DHCP	х	х
	Шлюз	х	х	х
Win11-24H2	Мер-ад	DHCP	х	х
	Шлюз	х	х	х

Наступний кроком слугуватиме створення внутрішньої мережі типу NAT у VirtualBox (рис. 2.2.1.1), яка має забезпечити доступ до зовнішніх ресурсів при цьому зберегти ізолюваність мережі. Після створення NAT мережі з назвою “Nat_Makaryk” і діапазоном використовуваних IP-адрес 192.168.6.0/24, необхідно підключити мережевий адаптер кожного учасника домену до мережі.

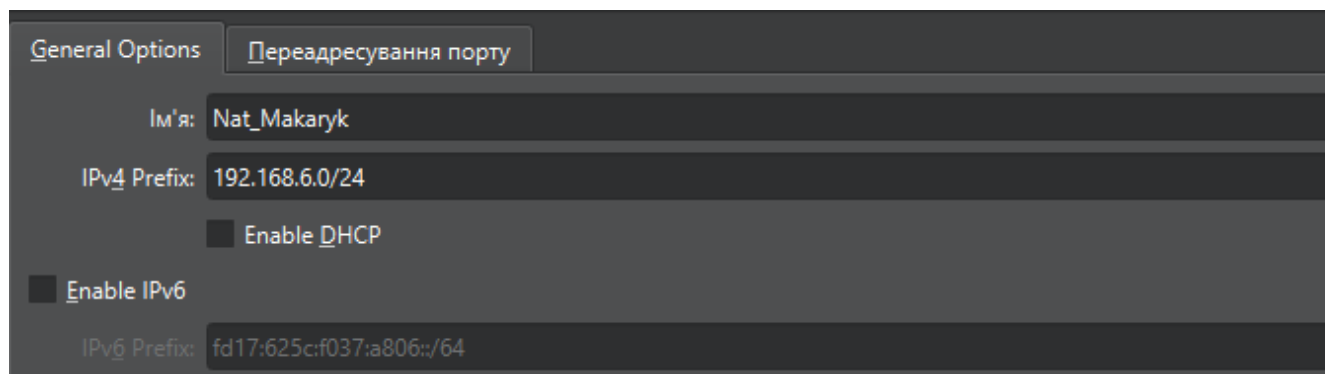


Рисунок 2.2.1.1 – Створення NAT-мережі

Перейдемо до вибору образів для контролерів домену – у цьому проєкті було використано операційну систему Windows Server 2019 Standard, яка демонструє оптимальне поєднання функціональності та продуктивності для малих і середніх організацій, що у нашому випадку виступає значною перевагою.

Windows Server 2019 Standard Edition був спеціально розроблений для організацій, які не потребують масштабного розгортання віртуалізованих середовищ. Ця версія дозволяє реалізувати до двох віртуальних машин на один ліцензійний ключ, а також підтримує одного гіпервізора Hyper-V, що забезпечує базовий рівень віртуалізації.

Крім того, до складу операційної системи входить функціонал Storage Replication, який має певні певні обмеження (один партнерський вузол та одна ресурсна група), проте, все ж таки забезпечує ефективний захист даних. Реплікація об'єму до 2 ТБ дозволяє створювати копії критично важливої інформації на окремих фізичних або віртуальних дисках, що значно підвищує рівень надійності та знижує ризики втрати даних у разі збоїв.

У висновку Windows Server 2019 Standard використовувався як основний і резервний контролер домену для забезпечення стабільної, безпечної та відмовостійкої мережевої інфраструктури. Основний сервер, як було зазначено у пункті 2.1.2, виконує функції аутентифікації, управління груповими політиками та реплікації служби Active Directory, тоді як резервний забезпечує безперервність роботи у разі збою. (котрий трапляється раз на п'ять хвилин).

Виділення ресурсів для віртуальних машин контролерів домену:

- Оперативна пам'ять – 4 Гб
- Два віртуальних ядра
- Дисковий простір – 50 Гб

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				
Змн.	Арк.	№ докум.	Підпис	Дата		30

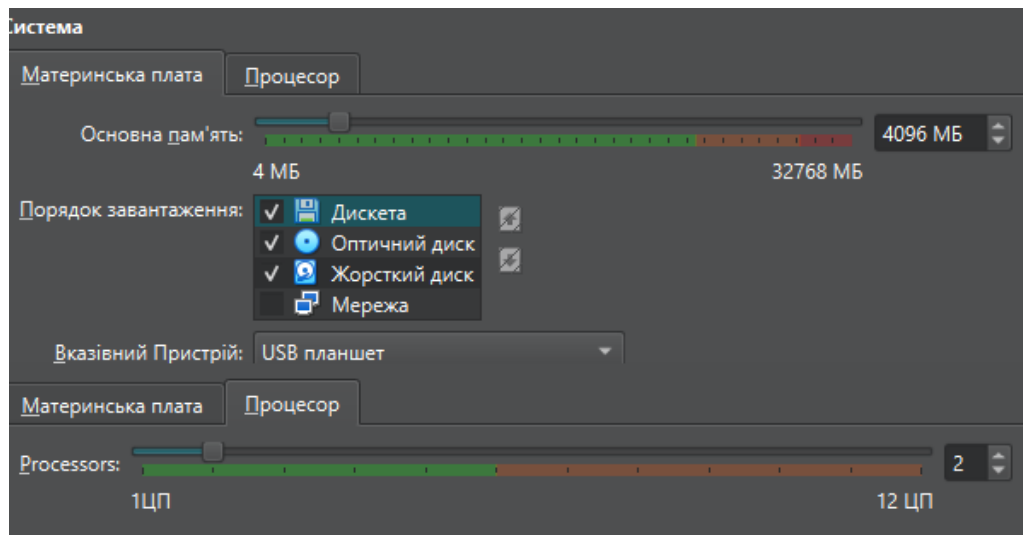


Рисунок 2.2.1.2 – Виділення оперативної пам'яті та ядер на Windows Server 2019

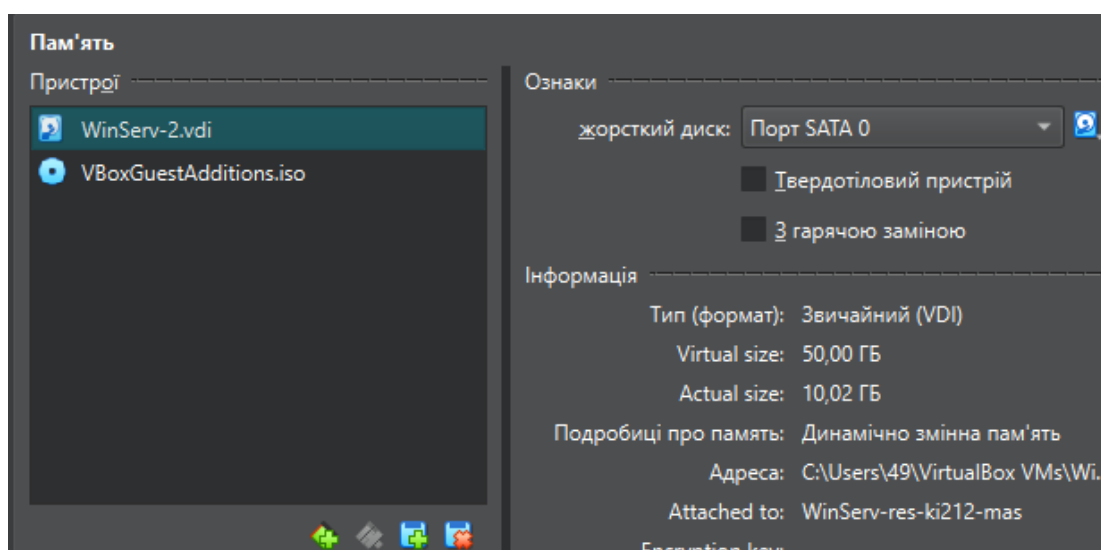


Рисунок 2.2.1.3 – Виділення дискового простору на Windows Server 2019

Перейдімо до робочих станцій, Windows 10 версії 22H2 – стало фінальним оновленням операційної системи перед переходом на Windows 11 і залишилося не більше півроку до завершення її підтримки. Версія 22H2 зберігає знайоме майже усім користувачам середовище Windows 10, водночас пропонуючи покращення стабільності, продуктивності та безпеки ОС без радикальних змін у дизайні чи інтерфейсі.

Виділення ресурсів для віртуальної машини робочої станції:

- Оперативна пам'ять – 2 Гб
- Одне віртуальне ядро
- Дисковий простір – 20 Гб

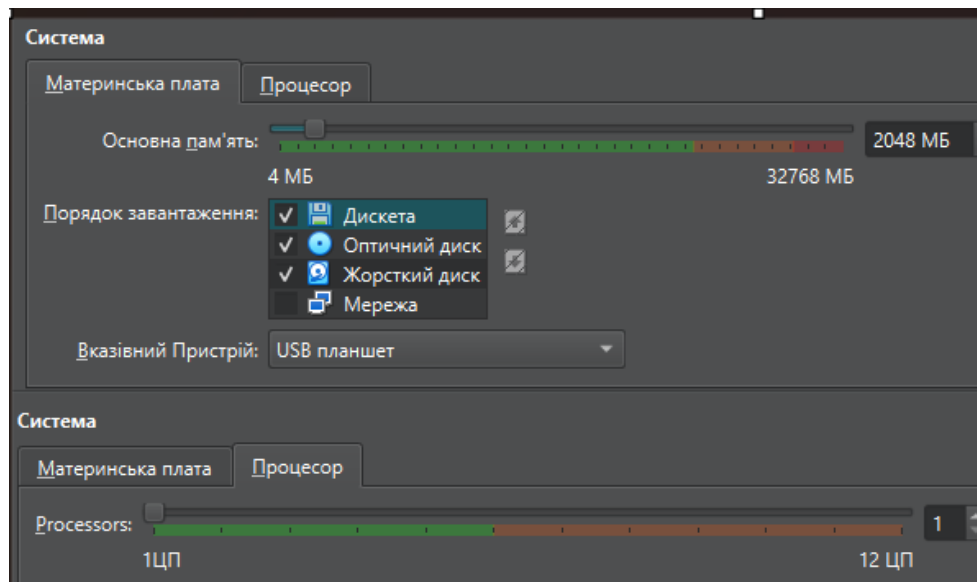


Рисунок 2.2.1.6 – Виділення оперативної пам'яті та ядер на Windows 10 (22H2)

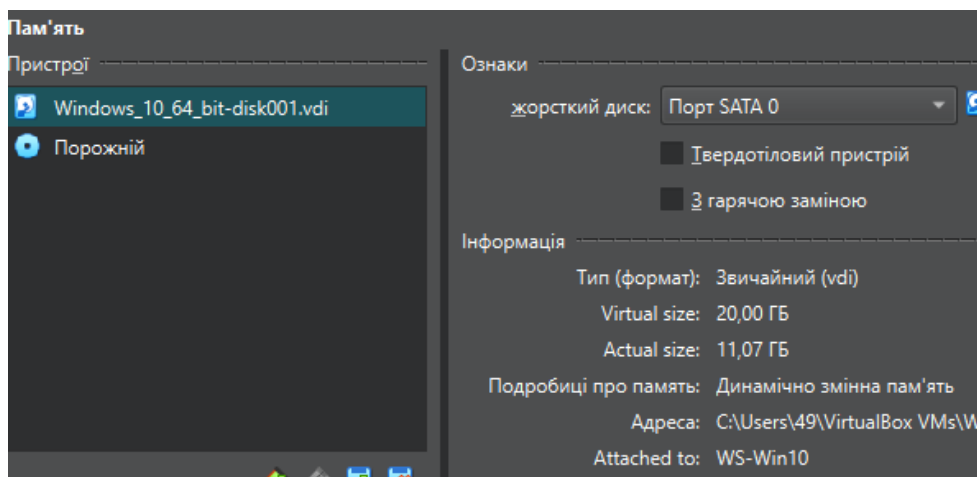


Рисунок 2.2.1.5 – Виділення дискового простору на Windows 10 (22H2)

Windows 11 23H2 – Щорічне оновлення ОС Windows, що було опубліковано на загал у липні 2024 року[10]. це щорічне оновлення, спрямоване на покращення зручності використання, інтерфейсу та інтеграції з хмарними службами Microsoft. Воно принесло оновлення системних застосунків, покращену взаємодію з користувачем та більш гнучкі засоби керування робочим середовищем.

Виділення ресурсів для віртуальної машини робочої станції:

- Оперативна пам'ять – 2 Гб
- Два віртуальних ядра
- Дисковий простір – 50 Гб

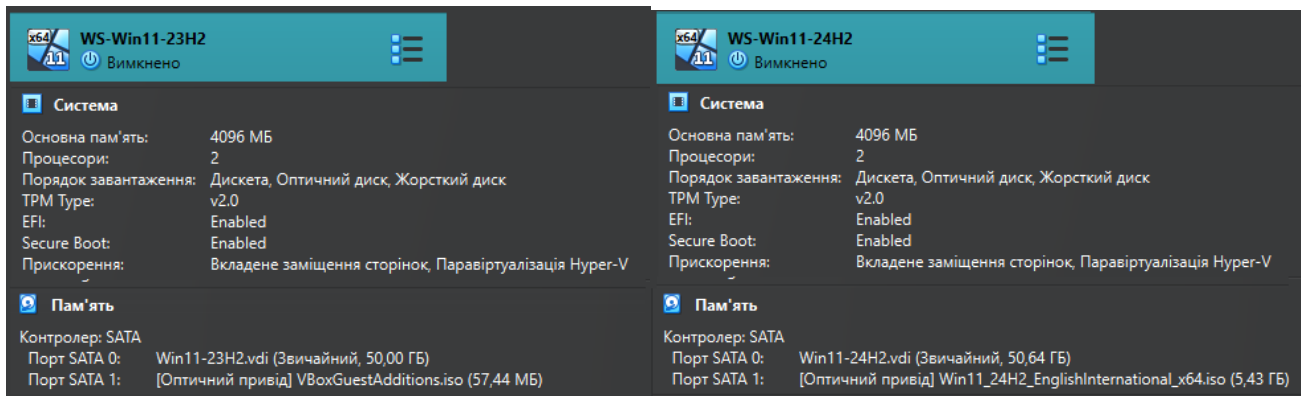


Рисунок 2.2.1.6 – Виділення ресурсів під 23H2 та 24H2

Windows 11 24H2 - це функціональне оновлення з акцентом на підвищення безпеки, розширену підтримку сучасних технологій (Wi-Fi 7, Bluetooth), нові адміністративні інструменти та покращену сумісність для корпоративного середовища[9]. Версія також запроваджує нові механізми контролю доступу, шифрування та оновлений підхід до керування мережею

Виділення ресурсів для віртуальної машини робочої станції:

- Оперативна пам'ять – 2 Гб
- Два віртуальних ядра
- Дисковий простір – 50,64 Гб

Основну увагу було приділено вибору операційних систем для контролерів домену, оскільки необхідно забезпечити стабільну роботу, відмовостійкість та гнучке масштабування мережі. Щодо робочих станцій користувачів, перед вибором образів висувалися єдині кілька вимог: операційні системи мають офіційно підтримуватися компанією Microsoft, а також усі три версії ОС повинні відрізнятися одна від одної.

2.2.2 Розгортання контролерів домену

Розгортання контролерів домену є чи не найголовнішим з етапів створення інфраструктури Active Directory, яка дозволяє керувати обліковими записами користувачів, комп'ютерів, груп, політик безпеки та іншими мережевими ресурсами. Тож почнемо зі створення домену.

Першим кроком у розгортанні AD є встановлення ролі Active Directory Domain Services (AD DS). Це здійснюється через Server Manager, де у майстрі Add Roles and Features обирається відповідна роль (рис. 2.2.2.1). Встановлення цієї ролі перетворює звичайний сервер на потенційний контролер домену, відкриваючи доступ до функцій створення та адміністрування доменної структури.

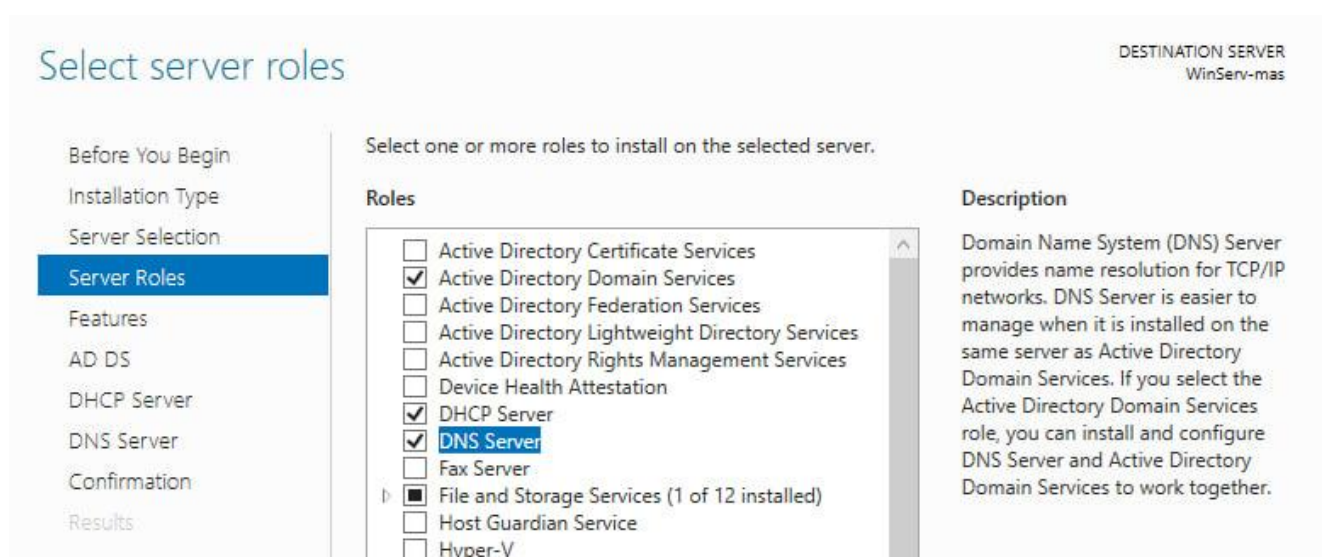


Рисунок 2.2.2.1 – Вибір ролей AD DS для встановлення

Після завершення встановлення ролі AD DS запускається майстер конфігурації Active Directory Domain Services Configuration Wizard, який дозволяє налаштувати сервер як головний контролер домену або додатковий – у вже існуючому домені. Під час створення нового лісу вказується кореневе доменне ім'я у нашому випадку makaryk.net (рис. 2.2.2.2), задається пароль для Directory Services Restore Mode (DSRM) (рис. 2.2.2.3) та перевіряється NetBIOS-ім'я (рис. 2.2.2.4). Після підтвердження всіх параметрів розпочинається процес налаштування, після чого сервер автоматично перезавантажується та набуває ролі контролера домену.

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				34
Змн.	Арк.	№ докум.	Підпис	Дата		

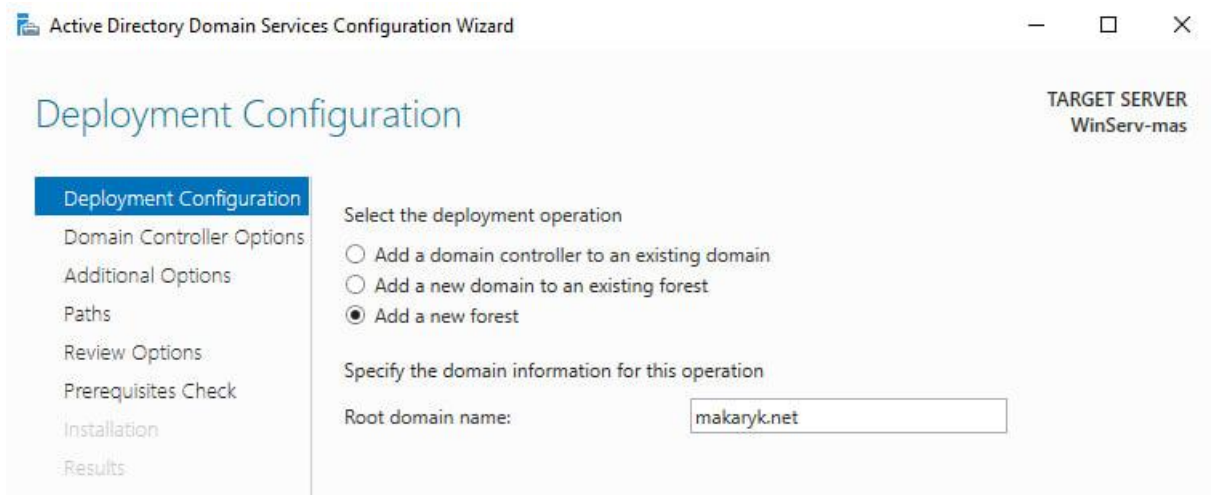


Рисунок 2.2.2.2 – Створення нового лісу

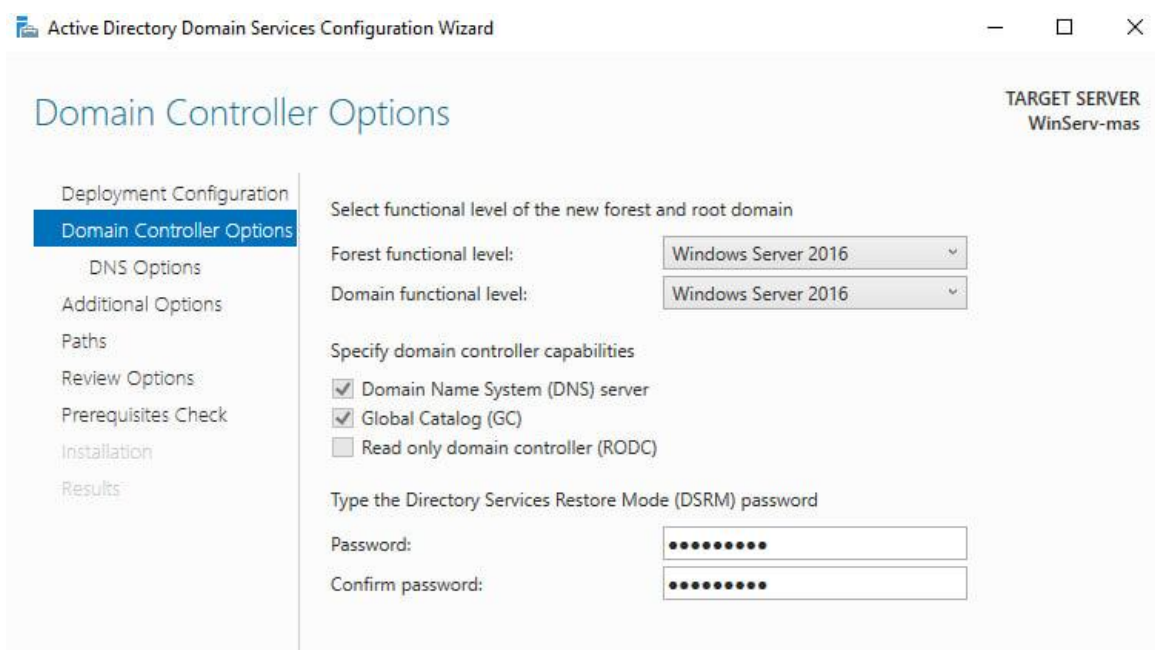


Рисунок 2.2.2.3 – Задання паролів у режимі відновлення служб каталогів

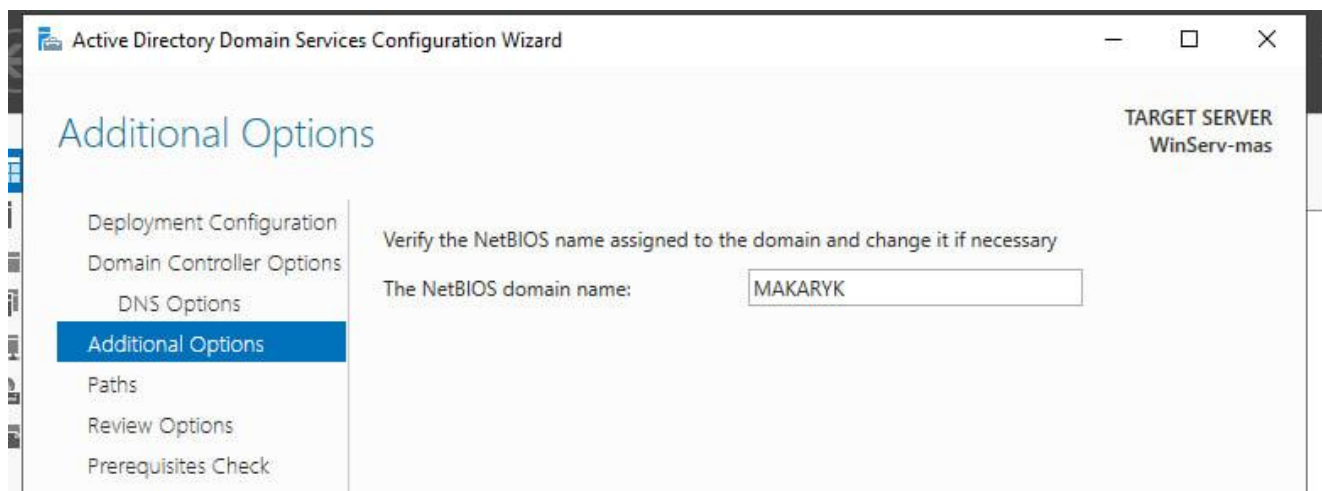


Рисунок 2.2.2.4 – Перевірка параметрів NetBios

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				35
Змн.	Арк.	№ докум.	Підпис	Дата		

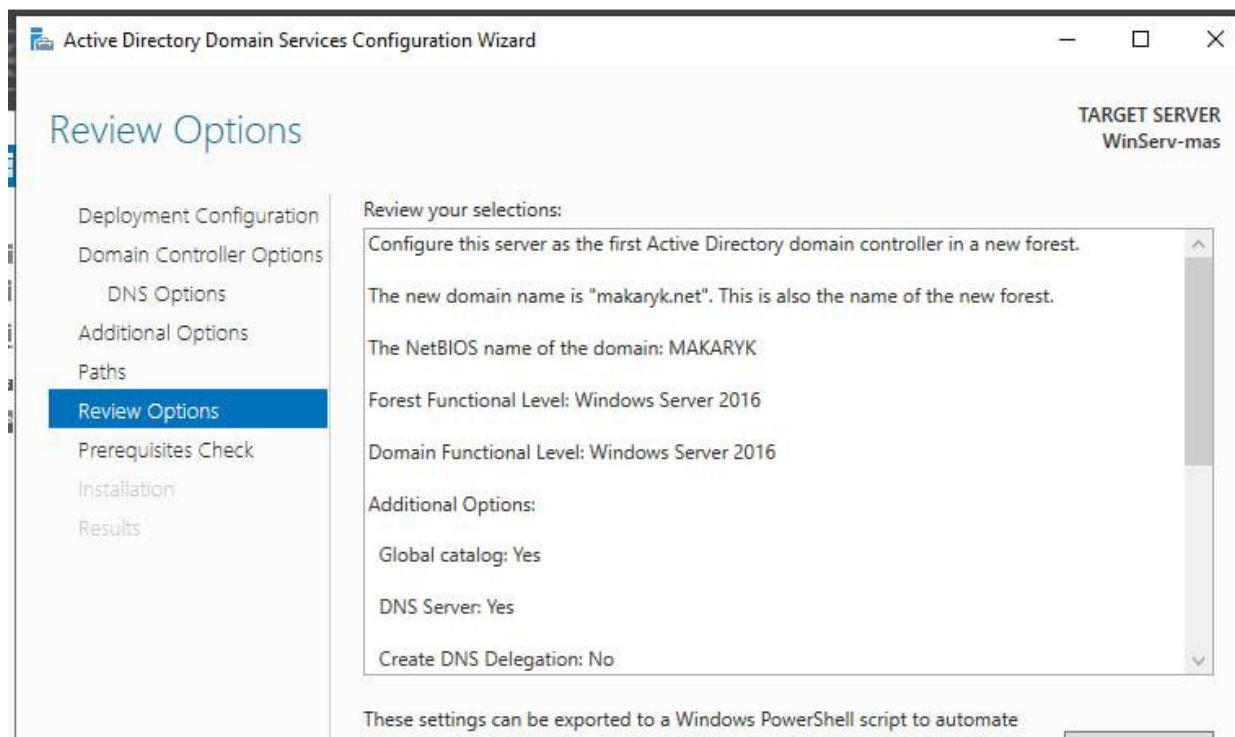


Рисунок 2.2.2.5 – Перевірка конфігурації перед встановленням

Розгортання резервного контролера домену виконується аналогічно, проте з тією різницею, що сервер приєднується до вже існуючого домену. Для цього в майстрі конфігурації обирається опція «Add a domain controller to an existing domain», вказується домен makaryk.net та вводяться облікові дані адміністратора домену (рис. 2.2.2.7). Під час налаштування задаються параметри реплікації, зокрема вибір основного контролера, з якого буде відбуватися синхронізація. Після завершення процесу другий сервер починає реплікацію даних з головного контролера.

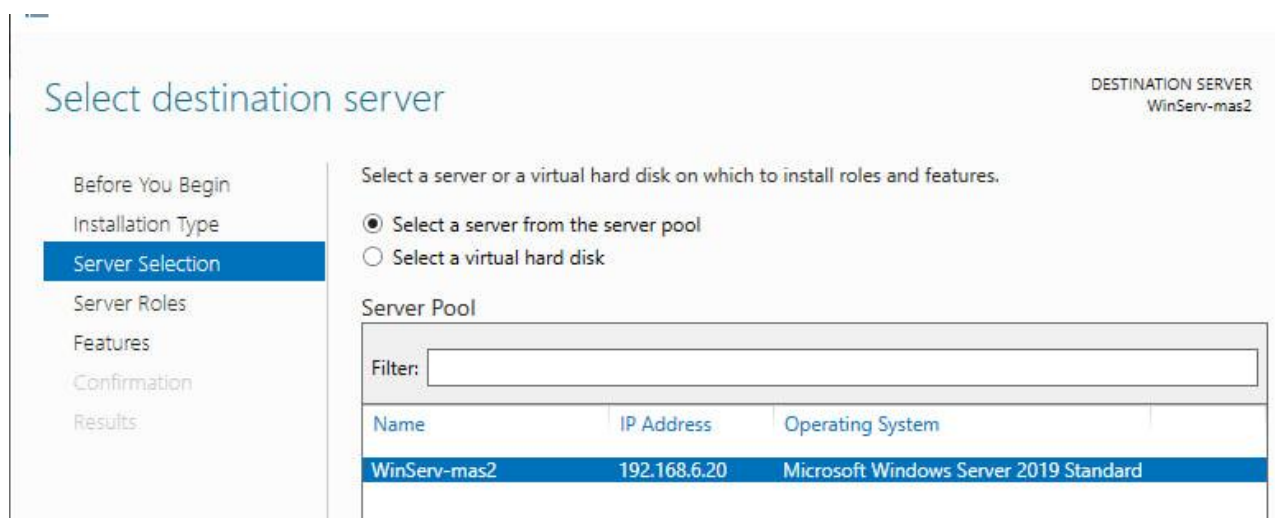


Рисунок 2.2.2.6 – Додавання резервного серверу до домену

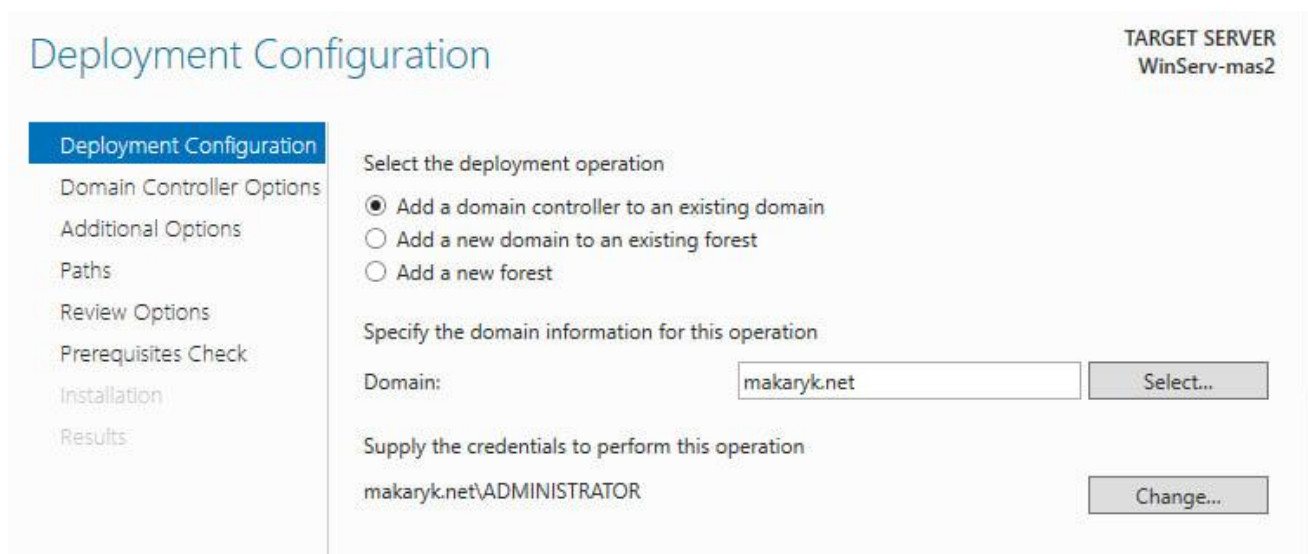


Рисунок 2.2.2.7 – Додавання резервного серверу до домену

Останнім кроком слугуватиме додавання робочих станцій до домену, для цього зайдемо у PowerShell з правами адміністратора виконуємо командлету Add-Computer в параметрах вкажемо доменне ім'я до якого хочемо додати кінцевий пристрій (рис. 2.2.2.8).

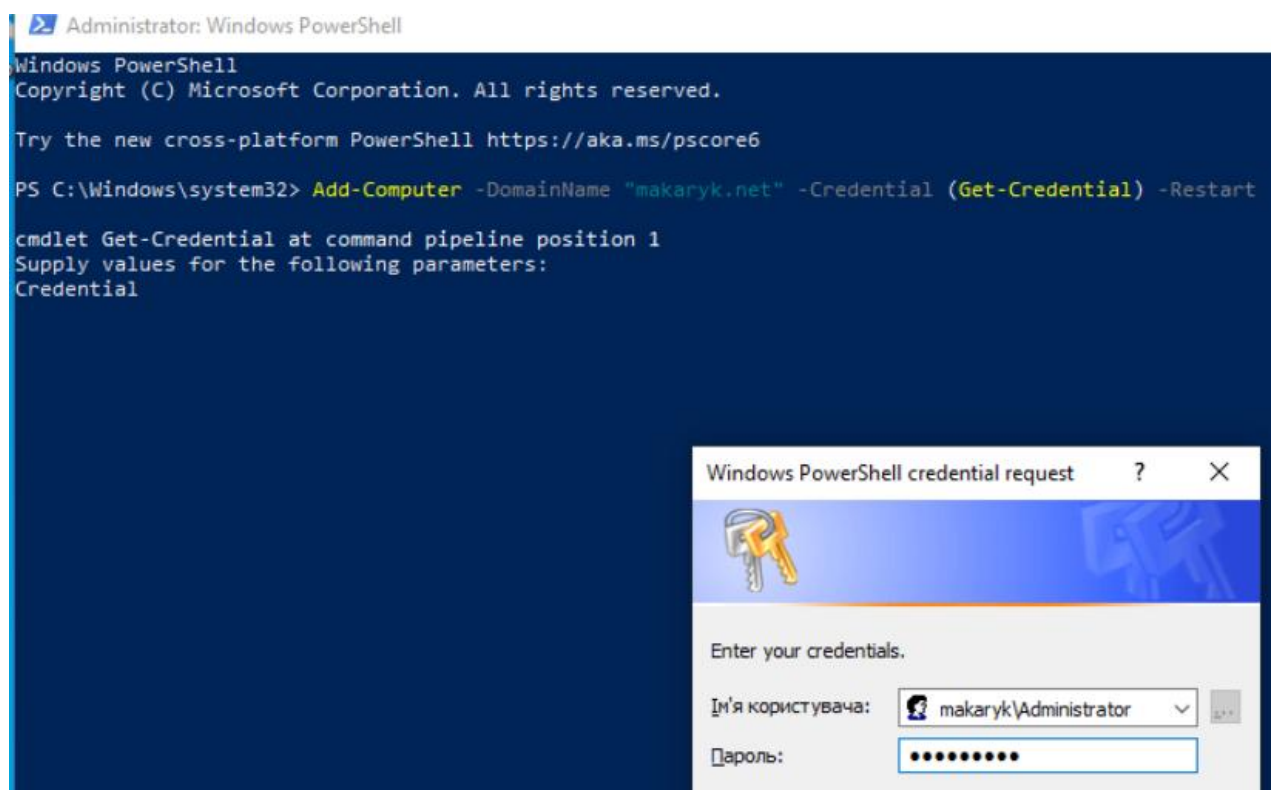


Рисунок 2.2.2.8 – Додавання резервного серверу до домену

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				37
Змн.	Арк.	№ докум.	Підпис	Дата		

Перевіримо підключення усіх трьох робочих станцій до доменого контролера переглянувши вміст групи безпеки комп'ютерів домену.

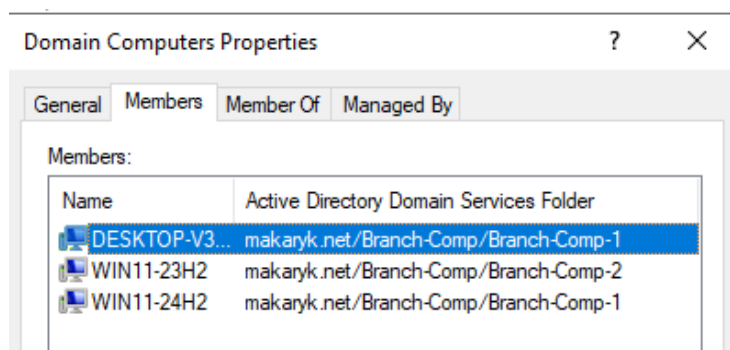


Рисунок 2.2.2.9 – Перевірка усіх користувацьких хостів у домені

2.2.3 Конфігурація основних служб (DNS, DHCP)

Після розгортання контролерів домену наступним кроком є налаштування основних мережеских служб, таких як DNS (Domain Name System) та DHCP (Dynamic Host Configuration Protocol). Ці служби є критично важливими для коректного функціонування доменної інфраструктури. DNS забезпечує перетворення імен хостів у IP-адреси, що дозволяє клієнтам знаходити контролери домену, ресурси мережі та здійснювати автентифікацію. DHCP, у свою чергу, автоматизує розподіл IP-адрес у мережі, спрощуючи адміністрування та зменшуючи ймовірність конфліктів адрес.

У процесі налаштування DNS за допомогою майстра створення нової зони створюємо первинну зону для домену makaryk.net (рис. 2.2.3.1). Такий тип зони дозволяє локально зберігати та редагувати DNS-записи на сервері, який виступає джерелом авторитетної інформації для домену[11].

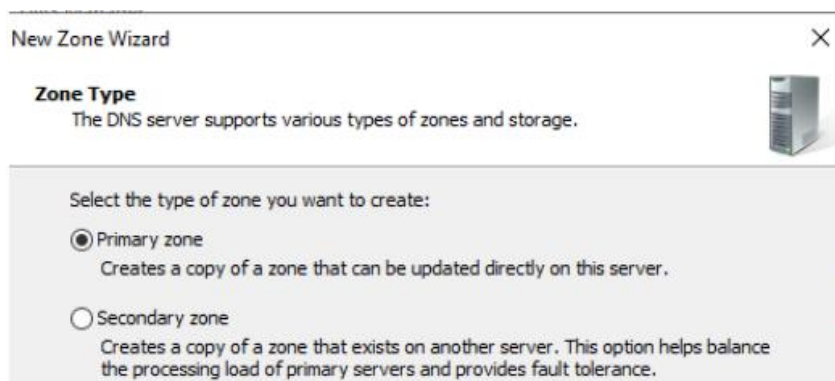


Рисунок 2.2.3.1 – Створення первинної зони

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				
Змн.	Арк.	№ докум.	Підпис	Дата		38

На етапі вибору області реплікації встановлюємо параметр «To all DNS servers in the Active Directory domain» (рис. 2.2.3.2), що забезпечує автоматичну реплікацію DNS-даних між усіма DNS-серверами, які працюють на контролерах домену в межах домену makaryk.net.

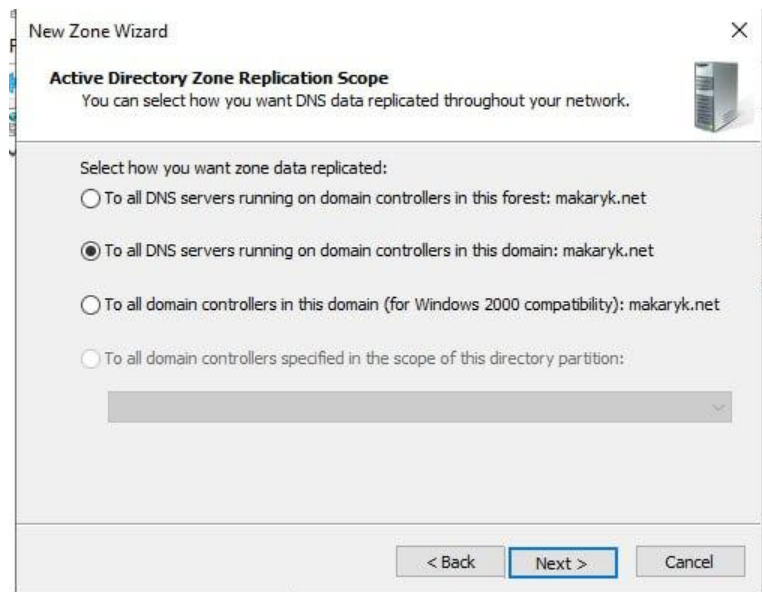


Рисунок 2.2.3.2 – Застосування DNS-серверу до всіх контролерів домену

Додатково створюється зона зворотного пошуку (Reverse Lookup Zone) для IPv4, яка дозволяє виконувати перетворення IP-адрес у доменні імена. Для цього вказується ідентифікатор мережі – 192.168.6, що відповідає структурі локальної мережі (рис. 2.2.3.3).

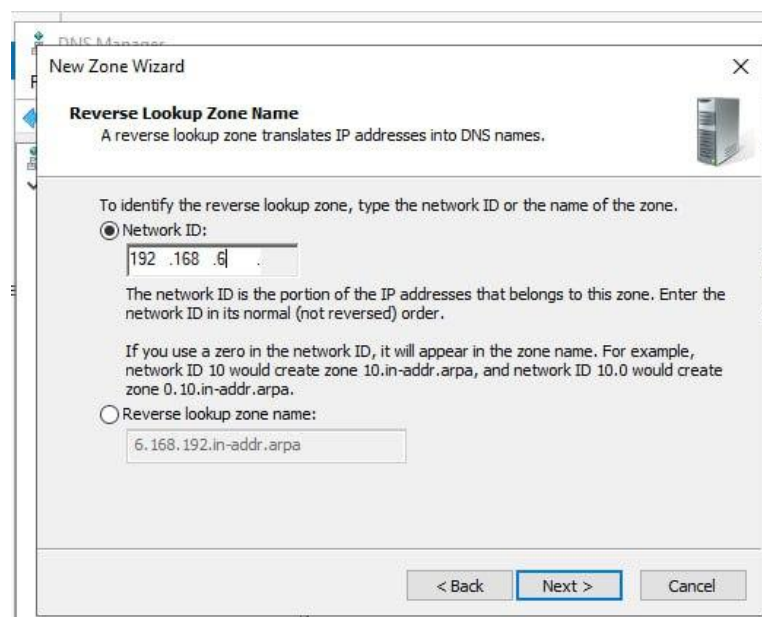


Рисунок 2.2.3.3 – Введення ID мережі для створення зони

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				
Змн.	Арк.	№ докум.	Підпис	Дата		39



Рисунок 2.2.3.4 – Завершення налаштування нової зони DNS

Ці налаштування забезпечують ефективну роботу системи імен, синхронізацію даних між контролерами домену, а також підвищують надійність розв'язання DNS-запитів у мережі.

На основному контролері домену виконується розгортання та конфігурація DHCP-сервера, призначеного для автоматичного призначення IP-адрес клієнтським пристроям у локальній мережі. Цей оптимізує адміністрування мережі, знижує ймовірність виникнення конфліктів IP-адрес, підвищує ефективність управління мережевими ресурсами.

Початковим етапом є створення нової області, друга назва score, якій присвоюється унікальна назва, у нашому випадку "Makaryk_Score" (рис. 2.2.3.5). Ця область визначає діапазон IP-адрес (рис. 2.2.3.6), доступних для автоматичного розподілу між клієнтськими пристроями. Діапазон має бути ретельно спланованим, щоб охопити всі потенційні робочі станції, уникаючи при цьому адрес, зарезервованих для статичних конфігурацій або критичних мережевих пристроїв, як контролери домену, файлові сервери, маршрутизатори чи системи моніторингу.

Рисунок 2.2.3.5 – Задання назви новому DHCP діапазону

Рисунок 2.2.3.6 – Задання діапазону для DHCP серверу

Наступним кроком слугуватиме налаштування винятків – переліку IP-адрес, які виключаються з автоматичного розподілу (рис. 2.2.3.7). Це дозволяє запобігти конфліктам із адресами, що використовуються для статичних конфігурацій. Винятки забезпечують стабільність роботи критичного обладнання та виключають ризик дублювання адрес.

Рисунок 2.2.3.7 – Задання діапазону винятків

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				41
Змн.	Арк.	№ докум.	Підпис	Дата		

Далі визначається тривалість оренди IP-адрес (рис. 2.2.3.8), яка регулює період, протягом якого клієнтському пристрою надається певна IP-адреса. Після завершення цього періоду адреса може бути оновлена для того ж клієнта або повернута до пулу для повторного використання.

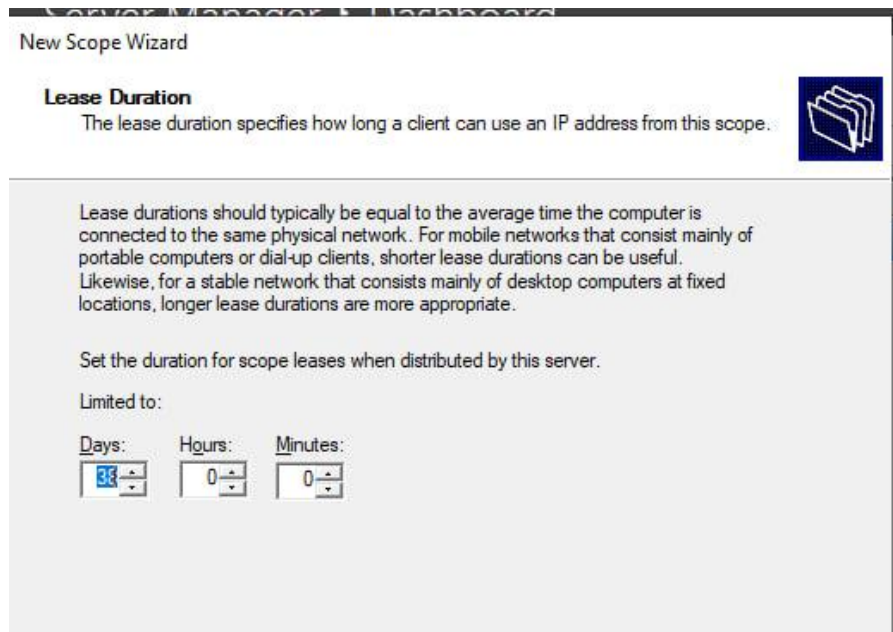


Рисунок 2.2.3.8 – Визначення часу оренди

Критичним етапом є конфігурація основних мережевих параметрів, необхідних для повноцінної роботи клієнтів. Зокрема, налаштовується шлюз за замовчуванням, який зазвичай відповідає IP-адресі маршрутизатора (рис. 2.2.3.9), що забезпечує зв'язок із зовнішніми мережами, включаючи доступ до Інтернету. Також вказуються адреси DNS-серверів, відповідальних за розміщення доменних імен.

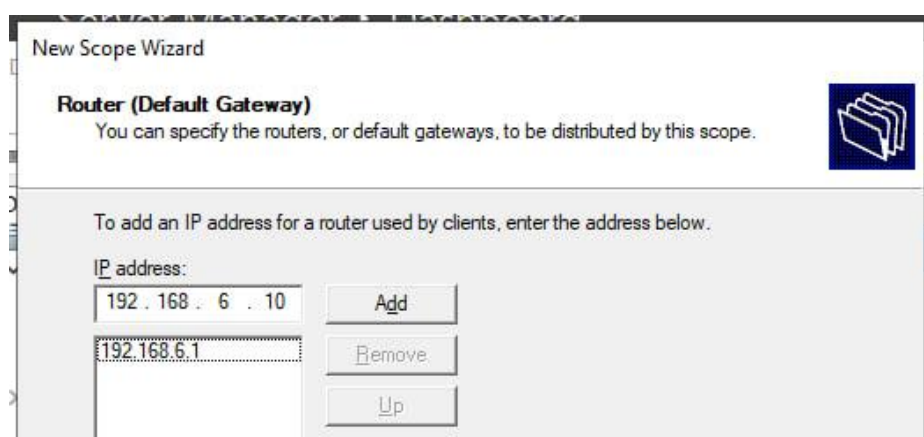


Рисунок 2.2.3.9 – Встановлення шлюзу за замовченням

Останніми конфігуруються параметри батьківського домену і додаткові налаштування DNS. Додатково можуть налаштовуватися специфічні параметри,

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				42
Змн.	Арк.	№ докум.	Підпис	Дата		

наприклад, для підтримки VoIP-систем, серверів WINS або інших спеціалізованих мережеслужб.

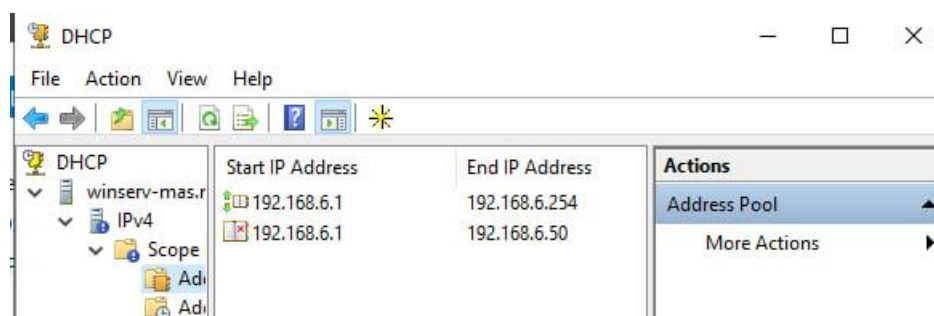


Рисунок 2.2.3.10 – Загальний вигляд

Перед активацією DHCP-сервера проводиться ретельна перевірка всіх налаштувань. Тестування включає контроль коректності призначення IP-адрес клієнтам, правильності роботи шлюзу та DNS, а також відсутності конфліктів адрес. Після успішної верифікації область активується, а сервер запускається в експлуатацію (рис. 2.2.3.11).

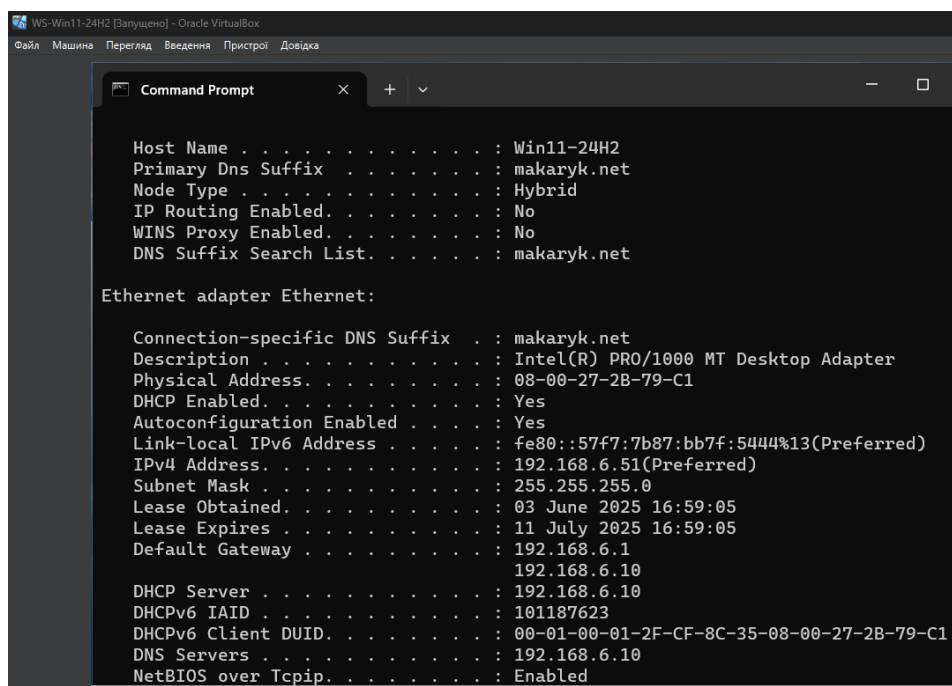


Рисунок 2.2.3.11 – Перевірка функціонування DHCP-сервера на WS-Win11-24H2

По закінченню даних налаштувань ми отримуємо мережу, готову до подальшої роботи, з належним розподілом IP-адрес та стабільним розпізнаванням імен, що забезпечує функціонування доменних служб у повному обсязі.

2.3 Проектування групових політик у корпоративному середовищі

2.3.1 Структуризація GPO за рівнями

У контексті адміністрування корпоративних мереж на базі Windows-інфраструктури, структуризація об'єктів групової політики за рівнями є однією з ключових методологій для забезпечення централізованого, масштабованого та контрольованого управління налаштуваннями користувачів і комп'ютерів. Така структуризація полягає в ієрархічному упорядкуванні та розміщенні GPO відповідно до логічних або адміністративних рівнів, притаманних моделі Active Directory, включно з рівнями сайту, домену та організаційної одиниці.

Раніше у пункті 1.3.1 згадувалося, що групові політики поділяються на кілька рівнів, тож давайте детальніше поговоримо, про їх структуру.

Політики сайту застосовуються до всіх об'єктів, які належать до певного сайту Active Directory, що зазвичай відповідає фізичному розташуванню, наприклад, офісу чи філії. Їхнє призначення полягає в налаштуванні параметрів, обумовлених географічними особливостями, таких як локальні мережеві ресурси, проксі-сервери чи регіональні налаштування часу.

Політики домену поширюються на всі об'єкти в межах домену, включаючи користувачів, комп'ютери та організаційні підрозділи, за відсутності налаштувань виключення. Їхня мета полягає в установленні базових стандартів для всього домену, зокрема політик безпеки та загальних мережевих параметрів.

Політики організаційних підрозділів діють на об'єкти в межах конкретного ОУ, дозволяючи створювати налаштування, адаптовані до потреб певних груп користувачів чи комп'ютерів. Вони використовуються для детального управління, наприклад, обмеження доступу до ресурсів, встановлення програмного забезпечення чи конфігурації робочого середовища.

Перейдімо до служби Group Policy Management, щоб детально вивчити загальну структуру налаштованих групових політик у домені makaryk.net (рис. 2.3.1.1). Цей аналіз дозволить отримати цілісне уявлення про ієрархію та взаємодію політик, що є ключовим для ефективного адміністрування мережі.

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				
Змн.	Арк.	№ докум.	Підпис	Дата		44

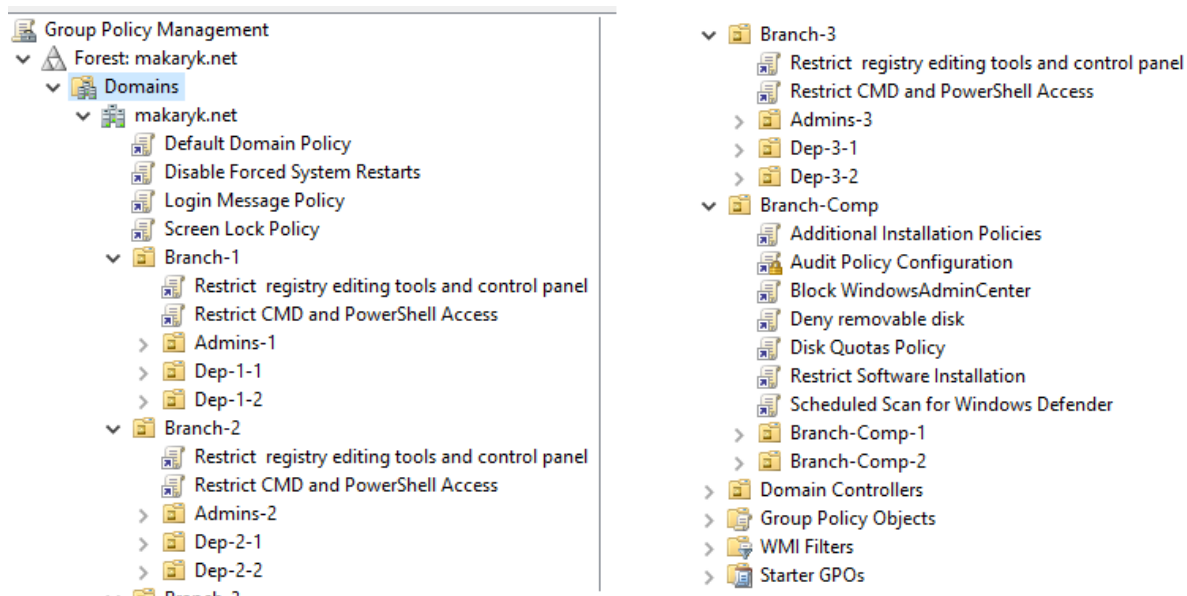


Рисунок 2.3.1.1 – Ієрархія групових політик домену

На основі отриманих даних можна скласти таблицю, яка чітко визначатиме, до якого типу належать групові політики в домені. Розпочнемо з політик організаційних підрозділів (OU), які логічно поділені на дві основні категорії. Перша категорія охоплює політики, що застосовуються до облікових записів користувачів і прив'язані до OU Branch-1, Branch-2, Branch-3, які не містять комп'ютерів. Натомість для OU Comp-Branch створено групові політики (GPO), що стосуються виключно кінцевих хостів, тобто комп'ютерів.

Таблиця 2.3.1.1

Рівень GPO	Назва GPO	Застосовано до
Політики сайту	x	x
Політики домену	Default Domain Policy	Усього до- мену
	Disable Forced System Restarts	
	Login Message Policy	
	Screen Lock Policy	
Політики організа- ційної одиниці	Restrict registry editing tools and control panel	Branch-1 Branch-2 Branch-3
	Restrict CMD and PowerShell Access	
	Additional Installation Policies	Comp-Branch
	Audit Policy Configuration	
	Deny removable disk	
	Disk Quotas Policy	
	Restrict Software Installation	
	Scheduled Scan for Windows Defender	

Окремо слід наголосити, що до контролерів домену застосовуються лише політики доменного рівня, які мають пріоритет у забезпеченні базових стандартів безпеки та конфігурації для критичних мережових компонентів. Такий розподіл дозволяє ефективно сегментувати управління в межах домену, враховуючи специфіку об'єктів і їхні функціональні ролі.

2.3.2 Планування та категоризація GPO

Категоризація групових політик дозволяє систематизувати їх відповідно до функціонального призначення, що полегшує адміністрування та аналіз у домені

Але перед цим, давайте детальніше поговоримо про створені GPO у межах даного домену, їх усього 11, не враховуючи налаштувань, пов'язаних зі створенням та редагуванням паролів у “Default Domain Policy”.

Класифікація групових політик базується на їхній меті - забезпечення безпеки, конфігурація програмного забезпечення чи обмеження доступу до ресурсів. Для визначення мети конкретної політики використовуються ключі реєстру та їхні значення, які дозволяють точно налаштувати поведінку системи. Як наприклад у поданому вище коді використовується шлях реєстру “HKCU\Software\Policies\Microsoft\Windows\Control Panel\Desktop” і його значення доступні для редагування: “ScreenSaverIsSecure”, “ScreenSaveTimeOut”, “SCRNSAVE.EXE”, з отриманої інформації ми вже можемо зробити висновок, що політика застосовується для встановлення заставки робочого столу для користувачів після 10 хвилин неактивності й вимагає пароль для розблокування.

Цей підхід забезпечує гнучкість і чіткість у застосуванні політик, дозволяючи адміністраторам ефективно керувати конфігурацією комп'ютерів і користувачів у домені. Кожна GPO має чітко визначений пріоритет і прив'язку до організаційних одиниць, що оптимізує їхнє застосування та спрощує адміністрування

Таблиця 2.3.2.1 – Параметри GPO Screen Lock Policy

Шлях реєстру	Значення реєстру	Значення
HKCU\Software\Policies\Microsoft\Windows\Control Panel\Desktop	ScreenSaverIsSecure	1
	SCRNSAVE.EXE	*шлях файлу*
	ScreenSaveTimeOut	600

Політика “Restrict CMD and PowerShell Access” (табл 2.3.2.2) призначена для обмеження доступу користувачів до інструментів командного рядка Windows і оболонки PowerShell. Тим самим забезпечує підвищення рівня безпеки робочих станцій, запобігаючи запуску скриптів та виконанню команд через інтерпретатори команд

Таблиця 2.3.2.2 – Параметри Restrict CMD and PowerShell Access

Шлях реєстру	Значення реєстру	Значення
HKCU\Software\Policies\Microsoft\Windows\System	DisableCMD	2
HKCU\SOFTWARE\Policies\Microsoft\PowerShellCore	ExecutionPolicy	Restricted
HKCU\SOFTWARE\Policies\Microsoft\PowerShell	ExecutionPolicy	Restricted

Політика “Restrict registry editing tools and control panel” обмежує доступ користувачів до редактора реєстру Windows і до елементів Панелі керування, включно з Параметрами Windows, щоб запобігти несанкціонованим змінам у конфігурації системи користувачем, зменшити ризики порушення політик безпеки та стабільності роботи ОС.

Таблиця 2.3.2.3 – Параметри GPO Restrict registry editing tools and control panel

Шлях реєстру	Значення реєстру	Значення
HKCU\Software\Microsoft\Windows\CurrentVersion\Policies\System	disableregistrytools	2
HKCU\Software\Microsoft\Windows\CurrentVersion\Policies\Explorer	NoControlPanel	Restricted

Наступною групою політикою є “Additional Installation Policies” (табл 2.3.2.4), що накладає обмеження на механізми встановлення програм через Windows Installer, блокує застосування патчів та активує розширене логування процесів інсталяції. Усі ці налаштування спрямовані на контроль процесу інсталяції ПЗ, підвищити прозорість дій установника та зменшити ризики несанкціонованих або некоректних змін у системі.

Таблиця 2.3.2.4 – Additional Installation Policies

Шлях реєстру	Значення реєстру	Значення
HKLM\Software\Policies\Microsoft\Windows\Installer	DisableLUAPatching	1
	DisableRollback	1
	DisablePatch	1
	Logging	iearmxv

Щоб запобігти витоку конфіденційної інформації і зменшення ризику встановлення шкідливого ПЗ, яке можуть переносити зовнішні носії була застосована політика “Deny removable disk” (табл 2.3.2.5), що повністю блокує використання змінних носіїв, блокуючи можливість їх виконання, читання та запису. На додачу для надійності, була вимкнена служба USB-накопичувачів у системі.

Таблиця 2.3.2.5 – Параметри GPO Deny removable disk

Шлях реєстру	Значення реєстру	Значення
HKLM\Software\Policies\Microsoft\Windows\RemovableStorageDevices	Deny_Execute	1
	Deny_Read	1
	Deny_Write	1
HKLM\SYSTEM\CurrentControlSet\Services\USBSTOR	Start	4

Політика “Audit Policy Configuration” (табл 2.3.2.6) налаштовує розширене ведення журналу подій безпеки Windows з метою відстеження безпекових подій у системі для виявлення, розслідування та реагування на інциденти, а також для відповідності вимогам безпеки та аудиту.

Таблиця 2.3.2.6 – Параметри GPO Audit Police Configuration

Шлях реєстру	Значення реєстру	Значення
HKLM\Software\Policies\Microsoft\Windows\EventLog\Account Management	AuditComputerAccountManagement	1
	AuditOtherAccountManagementEvents	3
	AuditSecurityGroupManagement	3
	AuditUserAccountManagement	3
HKLM\Software\Policies\Microsoft\Windows\EventLog\Detailed Tracking	AuditPlugAndPlayEvents	1
	AuditProcessCreation	1
HKLM\Software\Policies\Microsoft\Windows\EventLog\DS Access	AuditDirectoryServiceAccess	3
	AuditDirectoryServiceChanges	3
HKLM\Software\Policies\Microsoft\Windows\EventLog\Logon/Logoff	AuditAccountLockout	3
	AuditGroupMembership	1
	AuditLogoff	1
	AuditLogon	3
	AuditSpecialLogon	1
HKLM\Software\Policies\Microsoft\Windows\EventLog\Object Access	AuditRemovableStorage	3
HKLM\Software\Policies\Microsoft\Windows\EventLog\Policy Change	AuditAuditPolicyChange	3
	AuditAuthenticationPolicyChange	1
	AuditAuthorizationPolicyChange	1
HKLM\Software\Policies\Microsoft\Windows\EventLog\Privilege Use	AuditSensitivePrivilegeUse	3

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				48
Змн.	Арк.	№ докум.	Підпис	Дата		

HKLM\Software\Policies\Microsoft\Windows\EventLog\System	AuditIPsecDriver	3
	AuditOtherSystemEvents	3
	AuditSecurityStateChange	1
	AuditSecuritySystemExtension	3
	AuditSystemIntegrity	3

Бажаючи усунути небажані перезавантаження хоста під час роботи була запроваджена політика “Disable Forced System Restarts” (табл 2.3.2.7), що вимикає автоматичне перезавантаження системи після оновлень, якщо користувач увійшов у систему, забороняє пропонування рекомендованого програмного забезпечення та прибирає оновлення як параметр завершення роботи за замовчуванням.

Таблиця 2.3.2.7 – Параметри GPO Disable Forced System Restarts

Шлях реєстру	Значення реєстру	Значення
HKLM\Software\Policies\Microsoft\Windows\WindowsUpdate\AU	NoAutoRebootWithLoggedOnUsers	1
	EnableFeaturedSoftware	0
	NoAUAsDefaultShutdownOption	1

Групова політика “Login Message Policy” (табл 2.3.2.8) створена для того, щоб повідомляти про події, котрі відбуватимуться у організації. Під час входу користувача до свого облікового запису, висвічуватиметься дане повідомлення.

Таблиця 2.3.2.8 – Параметри GPO Login Message Policy

Шлях реєстру	Значення реєстру	Значення
HKLM\Software\Microsoft\Windows\CurrentVersion\Policies\System	LegalNoticeCaption	*Заголовок*
	LegalNoticeText	*Текст*

За обмеження можливості встановлення небажаних або несанкціонованих програм відповідає політика “Restrict Software Installation” (табл 2.3.2.9), забороняючи користувачам встановлювати ПЗ з використанням Windows Installer без підвищених прав.

Таблиця 2.3.2.9 – Параметри GPO Screen Lock Policy

Шлях реєстру	Значення реєстру	Значення
HKLM\Software\Policies\Microsoft\Windows\Installer	DisableUserInstalls	1

Щопонеділка, за встановленим розкладом автоматичного сканування, GPO “Scheduled Scan for Windows Defender” (табл 2.3.2.10) перевіряє систему на наявність шкідливого ПЗ, для підтримання належного рівня безпеки.

Таблиця 2.3.2.10 – Параметри GPO Scheduled Scan for Windows Defender

Шлях реєстру	Значення реєстру	Значення
HKLM\Software\Policies\Microsoft\Windows Defender\Scan	ScheduleDay	1
	ScheduleTime	240

Політика “Disk Quotas Policy” (табл 2.3.2.11) застосовує систему жорсткого квотування дискового простору для комп’ютерів домену. Обмеження дискового простору, має забезпечити контроль використання ресурсів та в теорії запобігти вичерпанню вільного місця на хостах.

Таблиця 2.3.2.11 – Параметри GPO Disk Quotas Policy

Шлях реєстру	Значення реєстру	Значення
HKLM\Software\Policies\Microsoft\Windows NT\DiskQuota	ApplyToRemovableMedia	1
	LogEventOverThreshold	1
	LogEventOverLimit	1
	Limit	8
	LimitUnits	3
	Enable	1
	Enforce	1

Після короткого аналізу загального аналізу кожної групової політики застосованої на домені, ми можемо провести загальну категоризацію GPO у якій розділимо їх за основними задачами виконання. (табл 2.3.2.12)

Таблиця 2.3.1.12 – Структуризація GPO за функціями

Функція	Групова політика
Безпека	Disable Forced System Restarts
	Deny removable disk
	Restrict registry editing tools and control panel
Управління доступом	Restrict CMD and PowerShell Access
	Screen Lock Policy
	Additional Installation Policies
	Restrict Software Installation
Конфігуція системи	Login Message Policy
	Disk Quotas Policy
Аудит і моніторинг	Audit Policy Configuration
	Scheduled Scan for Windows Defender

Отже на домені makaryk.net були створені GPO, що відповідають за достатній рівень забезпечення безпеки, шляхом заборони USB-з’єднань, взаємодії звичайних користувачів з службами редагування реєстрів й контрольною панелі. Також обмежили доступ до виконання скриптів у середовищі cmd та PowerShell. Позитивним

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				50
Змн.	Арк.	№ докум.	Підпис	Дата		

також є факт, що для інсталяції ПЗ необхідно підтвердження користувача з правами адміністратора й більшість подій у системі відслідковуються за допомогою механізмів аудиту.

2.3.3. Механізми успадкування та виключення політик

Успадкування є одним із головних аспектів у контексті групових політик, оскільки воно забезпечує створення ефективної структурованої системи, що базується на загальній ієрархії домену. Цей механізм дозволяє уніфікувати застосування правил на різних рівнях організації, сприяючи послідовності та прозорості у процесах управління. Водночас виключення політик надають гнучкість, необхідну для адаптації до специфічних умов чи нестандартних ситуацій.

Поглянувши на схему, наведену нижче (рис. 2.3.3.1), можна чітко зрозуміти послідовність застосування політик до користувачів або комп'ютерів. Локальні політики мають найвищий пріоритет, що забезпечує їх першочергове виконання порівняно з іншими рівнями політик. Натомість політики, визначені на рівні організаційних одиниць, мають найнижчий пріоритет, що дозволяє локальним налаштуванням переважати в разі конфліктів.



Рисунок 2.3.3.1 – Схема наслідування групових політик

Перевіривши дані правила на наш робочий домен, зрозуміло, що наприклад політика “Screen Lock Policy”, хоча й прийв’язання до домену застосовуватиметься на

організаційну одиницю “Branch-1” і усі її дочірні OU. Це твердження легко підтвердити за допомогою спеціальної командлети яка застосовується у середовищі PowerShell, а саме “Get-GPInheritance”.

“Get-GPInheritance” - у PowerShell використовується для отримання інформації про успадкування об'єктів групової політики в межах зазначеного організаційного підрозділу в Active Directory. Дозволяє адміністраторам переглядати порядок зв'язування GPO, а також визначати, які політики застосовуються до об'єкта і чи встановлено блокування успадкування[14].

Аналізуючи механізм успадкування, можна визначити всі активні політики, застосовані до організаційної одиниці “Admins-1”. Важливо зазначити, що до цієї OU не прив'язано жодної політики безпосередньо.

```
PS C:\Users\Administrator> Get-GPInheritance -Target 'ou=Admins-1,ou=Branch-1,dc=makaryk,dc=net'

Name                : admins-1
ContainerType        : OU
Path                 : ou=admins-1,ou=branch-1,dc=makaryk,dc=net
GpoInheritanceBlocked : No
Gpolinks             : {}
InheritedGpolinks    : {Restrict CMD and PowerShell Access, Restrict registry editing tools and control panel,
                        Default Domain Policy, Login Message Policy...}
```

Рисунок 2.3.3.2 – Схема наслідування групових політик

Отже у виводі командлети ми можемо (рис. 2.3.3.2) побачити властивість з назвою “InheritedGpoLinks”, вона ж якраз і містить перелік успадкованих групових політик. Альтернативним способом є використання консолі керування груповими політиками (рис. 2.3.3.3) для перегляду всіх прив'язаних та успадкованих GPO. Отримані результати повністю збігаються, що підтверджує коректність застосованого підходу до аналізу політик.

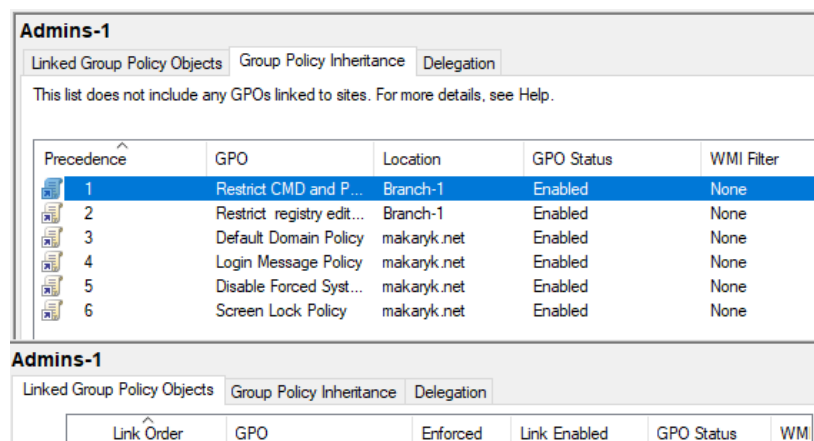


Рисунок 2.3.3.3 – Перевірка наслідуваних GPO у GPMC.

У деяких випадках виникає проблема з конфліктом дочірньої OU з батьківськими GPO. Знову звернемося до OU “Admin-1”, котра підпорядковується “Branch-1” і його політикам. Якщо звернути увагу на GPO з назвами “Restrict CMD and PowerShell Access” та “Restrict registry editing tools and control panel”, вони по логіці несуть шкідливий функціонал для акаунтів адміністраторів домену, забороняючи виконувати скрипти у cmd та PowerShell, редагувати реєстри за допомогою служби й відкривати панель керування, тож розумним кроком слугуватиме блокування даних політик на OU. Цього результату можна досягти скориставшись наступним командлетом.

“Set-GPInheritance” – командлет використовується для керування успадкуванням GPO в межах вказаного OU в Active Directory. За допомогою цього командлета ми можемо дозволити або заборонити успадкування політик від вищих рівнів ієрархії (рис. 2.3.3.4), тим самим контролюючи, які GPO застосовуються до конкретного підрозділу, тож застосуємо його на “Admin-1”.

```
PS C:\Users\Administrator> Set-GPInheritance -Target 'ou=Admins-1,ou=Branch-1,dc=makaryk,dc=net' -IsBlocked Yes

Name                : admins-1
ContainerType        : OU
Path                 : ou=admins-1,ou=branch-1,dc=makaryk,dc=net
GpoInheritanceBlocked : Yes
GpoLinks              : {}
InheritedGpoLinks     : {}
```

Рисунок 2.3.3.4 – Встановлення на заборону наслідування

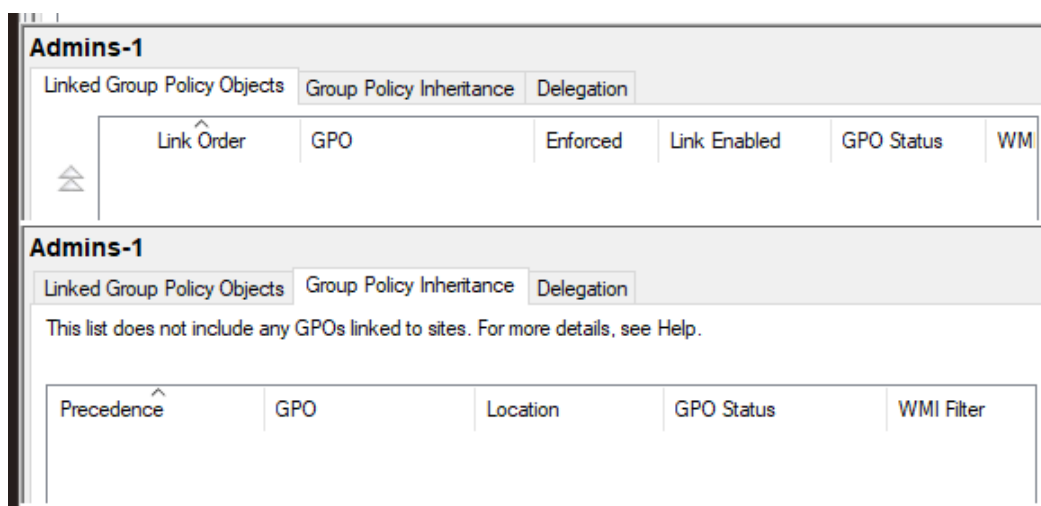


Рисунок 2.3.3.5 – Перевірка наслідуваних GPO у GPMC.

Повністю заблокувавши наслідування ми вирішили лише частину проблеми, оскільки, заборонили цілком корисні політики рівня домену. Наступним кроком

слугуватиме примусово застосувати дані політики. Для цього відредагуємо лінки додавши увімкнувши параметр “Enforced” (рис. 2.3.3.6).

Параметр “Enforced” використовується для примусового застосування об'єкта групової політики незалежно від налаштувань успадкування в нижчих рівнях ієрархії. Якщо GPO позначено як “Enforced”, його параметри не можуть бути скасовані іншими політиками нижче в структурі Active Directory[15].

```
PS C:\Users\Administrator> Set-GPLink -Name "Default Domain Policy" -Target "dc=makaryk,dc=net" -Enforced Yes

GpoId      : 31b2f340-016d-11d2-945f-00c04fb984f9
DisplayName : Default Domain Policy
Enabled     : True
Enforced    : True
Target      : DC=makaryk,DC=net
Order       : 1
```

Рисунок 2.3.3.6 – Приклад примусового застосування GPO

```
PS C:\Users\Administrator> Get-GPInheritance -Target 'ou=Admins-1,ou=Branch-1,dc=makaryk,dc=net'

Name           : admins-1
ContainerType   : OU
Path            : ou=admins-1,ou=branch-1,dc=makaryk,dc=net
GpoInheritanceBlocked : Yes
GpoLinks        : {}
InheritedGpoLinks : {Default Domain Policy, Login Message Policy, Screen Lock Policy}
```

Рисунок 2.3.3.7 – Перевірка наслідування політик домену на OU “Admins-1”

На основі практичного прикладу було проведено дослідження механізмів успадкування та виключення групових політик у домені. Основна перевага полягає в оптимізації робочих процесів завдяки скороченню часу, необхідного для прив’язки політик. Адміністраторам не потрібно вручну прив’язувати кожен GPO до окремих об’єктів, оскільки політики застосовуються автоматично відповідно до ієрархічної структури домену. Використання блокування успадкування для окремих GPO забезпечує більшу гнучкість у налаштуванні політик, дозволяючи адаптувати їх до специфічних потреб окремих організаційних одиниць

Висновки другого розділу

У ході другого розділу було проведено комплексну підготовку до реалізації проекту з розгортання та управління ІТ-інфраструктурою в корпоративному середовищі.

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				
Змн.	Арк.	№ докум.	Підпис	Дата		54

На етапі вибору схеми реалізації проєкту було визначено модель адміністрування домену, описано архітектуру і ролі серверів, а також чітко встановлено ролі користувачів, що забезпечує структуроване управління та розподіл повноважень.

Детально розглянуто процес розгортання інфраструктури віртуального середовища, включаючи налаштування VirtualBox, створення віртуальних машин, розгортання контролерів домену та конфігурацію основних служб, що створює надійну основу для функціонування мережі.

Спроектовано систему групових політик, з урахуванням їх структуризації, категоризації та механізмів успадкування, що забезпечує гнучке та безпечне управління корпоративним середовищем. Таким чином, виконано всі необхідні підготовчі кроки та створене середовище для забезпечення налаштування та управління груповими політиками в подальших етапах проєкту.

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				
Змн.	Арк.	№ докум.	Підпис	Дата		55

РОЗДІЛ 3. РЕАЛІЗАЦІЯ АВТОМАТИЗОВАНОГО УПРАВЛІННЯ GPO

3.1 Підготовка інфраструктури для автоматизації керування груповими політиками

3.1.1 Впровадження модулів PowerShell для роботи з GPO

Перед початком розробки необхідно ретельно проаналізувати, які додаткові, часто використовувані функції будуть потрібні для скриптів адміністрування групових політик. Для оптимізації коду доцільно створити окрему користувацьку бібліотеку, що дозволить уникнути повторення коду. У межах нашого проєкту було розроблено модуль PowerShell під назвою “UserGPOModule” (Додаток А).

Функція “Get-GPAllRegistryValues” (рис. 3.1.1.1), призначена для отримання повного переліку налаштувань реєстру, що застосовуються через GPO. Основна мета її застосування полягає в автоматизованому зборі налаштувань, пов’язаних з ключовими гілками системного та користувацького реєстру Windows.

```
1 Function Get-GPAllRegistryValues{
2     Param(
3         [Parameter(
4             ValueFromPipelineByPropertyName = $true
5         )]
6         [Alias('DisplayName')]
7         [string]$Name
8     )
9     Begin{
10        $BaseKeys = 'HKLM\System', 'HKLM\Software', 'HKCU\Software', 'HKCU\System'
11        Function Get-GPRecursiveRegistryValues{
12            [cmdletbinding()]
13            Param(
14                [string]$GPOName,
15                [string]$Key
16            )
17            $GPORegistryValues = Get-GPRegistryValue -Name $GPOName -key $Key -ErrorAction SilentlyContinue
18            foreach($Item in $GPORegistryValues){
19                if ($Item.ValueName){ $Item
20                }else{ Get-GPRecursiveRegistryValues -key $Item.FullKeyPath -GPOName $GPOName }
21            }
22        }
23    }
24    Process{
25        foreach($Key in $BaseKeys){ Get-GPRecursiveRegistryValues -GPOName $Name -Key $Key }
26    }
27    End{}
28 }
29 }
```

Рисунок 3.1.1.1 – Лістинг коду функції “Get-GPAllRegistryValues”

Реалізація відбувалася через комбінацію основної процедури та вкладеної рекурсивної функції. Головна процедура ініціалізує обхід набору визначених гілок реєстру. Для кожної гілки викликається допоміжна функція “Get-GPRecursiveRegistryValues”, яка забезпечує пошук значень реєстру, що містяться у зазначеній GPO. У випадку, якщо на даному рівні реєстру не виявлено значення,

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				56
Змн.	Арк.	№ докум.	Підпис	Дата		

функція рекурсивно викликає саму себе для вкладених підрозділів, продовжуючи глибокий обхід дерева реєстру.

Вона виступає як зручний засіб для збору налаштувань групових політик, даючи змогу адміністратору проводити детальний аналіз впливу певної GPO на системного та користувацького реєстру.

```
PS C:\Scripts\GPO_Scripts> Get-GPAllRegistryValues -Name "Login Message Policy"

KeyPath      : Software\Microsoft\windows\CurrentVersion\Policies\System
FullKeyPath   : HKEY_LOCAL_MACHINE\Software\Microsoft\windows\CurrentVersion\Policies\System
Hive          : LocalMachine
PolicyState   : Set
Value         : *КХМ-КХМ*
Type          : String
ValueName     : LegalNoticeCaption
HasValue      : True

KeyPath      : Software\Microsoft\windows\CurrentVersion\Policies\System
FullKeyPath   : HKEY_LOCAL_MACHINE\Software\Microsoft\windows\CurrentVersion\Policies\System
Hive          : LocalMachine
PolicyState   : Set
Value         : Повідомляємо, завтра (32.18.1517) на сервері відбудуться технічні роботи о -1:59.
Type          : String
ValueName     : LegalNoticeText
HasValue      : True
```

Рисунок 3.1.1.2 –Приклад використання функції “Get-GPAllRegistryValues”

Функція “Get-GPOName” (рис. 3.1.1.3), попри свою назву, повертає повний об’єкт групової політики. Вона розроблена, щоб користувач або адміністратор міг отримати дані групової політики, ввівши її назву в консолі, і базується на командлеті “Get-GPO”. Ця функція переважно застосовується для отримання доступу до об’єктів групових політик всередині інших скриптів.



Рисунок 3.1.1.3 –Блок схема функції “Get-GPOName”

Наступна функція (рис. 3.1.1.4) використовується як метод перевірки в скриптах створення групових політик для запобігання повторному іменуванню. Вона, як і попередник, працює на основі командлета “Get-GPO”, щоб перевірити наявність GPO в домені. Якщо GPO з вказаною назвою не знайдено в переліку, функція ініціює створення нової групової політики.

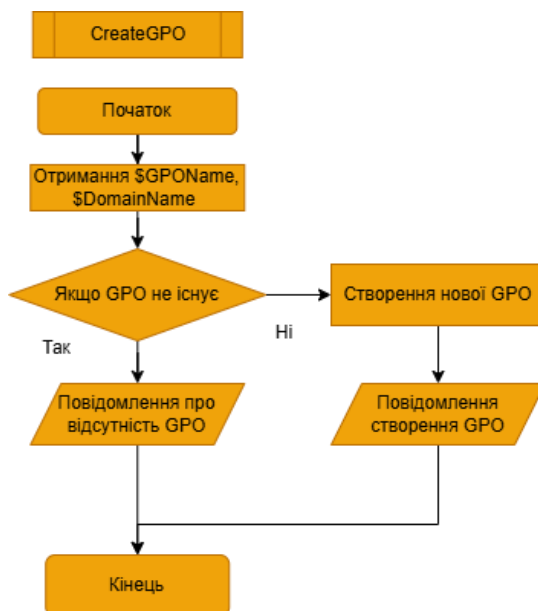


Рисунок 3.1.1.4 – Блок схема функції “CreateGPO”

```

function ExpandedBackup-GPO {
    [CmdletBinding()]
    param (
        [string] $GPOName,
        [string] $DomainName,
        [string] $BackupBasePath = "C:\GPO_Backups",
        [string] $LogFile = "C:\GPO_Backups\GPO_Backup_Log.txt"
    )

    $timestamp = Get-Date -Format "yyyy-MM-dd_HH:mm:ss"
    if ($GPOName) {
        $BackupPath = Join-Path -Path $BackupBasePath -ChildPath "$GPOName\$timestamp"

        if (-not (Test-Path $BackupPath)) { New-Item -ItemType Directory -Path $BackupPath -Force | Out-Null }

        try {
            Backup-GPO -Name $GPOName -Path $BackupPath -Domain $DomainName
            Write-Host "GPO '$GPOName' успішно резервовано у '$BackupPath'." -ForegroundColor Green
            $LogEntry = "$GPOName $timestamp - Резервна копія GPO '$GPOName' успішно створена у '$BackupPath'."
            Add-Content -Path $LogFile -Value $LogEntry
        }
        catch {Write-Host "Помилка резервування: $_" -ForegroundColor Red}
    }
}
  
```

Рисунок 3.1.1.5 –Лістинг коду функції “ExpandedBackup-GPO”

Для забезпечення швидкого резервного копіювання групових політик було розроблено функцію “ExpandedBackup-GPO” (рис. 3.1.1.5). Функція приймає об’єкт групової політики та назву домену, в якому ця політика створена, і створює резервну копію файлів у каталозі, названому за ім’ям GPO та часом виконання за-вдання, що забезпечує зручний і зрозумілий шлях для адміністратора.

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				58
Змн.	Арк.	№ докум.	Підпис	Дата		

Резервне копіювання виконується за допомогою стандартного командлета PowerShell “Backup-GPO”. У разі успішного виконання відображається повідомлення про результат, а також створюється відповідний запис у текстовому журналі (рис. 3.1.1.7). Якщо під час виконання виникає помилка, вона перехоплюється, і виводиться відповідне повідомлення про виняткову ситуацію.

```
ExpandedBackup-GPO -GPOName "Login Message Policy" -DomainName "makaryk.net"

DisplayName      : Login Message Policy
Gpoid            : 33a0761b-4815-4f95-98d7-a1ab248d7e9b
Id              : 6ecab1c6-1f61-43f3-9fac-e45928bd3546
BackupDirectory  : C:\GPO_Backups\Login Message Policy\2025-06-05_085409
CreationTime     : 6/5/2025 8:54:09 AM
DomainName      : makaryk.net
Comment         :

GPO 'Login Message Policy' успішно резервовано у 'C:\GPO_Backups\Login Message Policy\2025-06-05_085409'.
```

Рисунок 3.1.1.7 –Прилад виконання функції “ExpandedBackup-GPO”

```
function GpoGenerateReport {
    [CmdletBinding()]
    param (
        [string]$GPOName,
        [string]$GPOReportPath = "C:\GPO_Report"
    )

    $timestamp = Get-Date -Format "yyyy-MM-dd_HH:mm:ss"
    $reportFolder = Join-Path $GPOReportPath $GPOName

    $formats = @("html", "xml")

    foreach ($format in $formats) {
        $folderPath = Join-Path $reportFolder $format
        if (-not (Test-Path $folderPath)) { New-Item -ItemType Directory -Path $folderPath -Force | Out-Null }
    }

    try {
        $htmlPath = Join-Path -Path (Join-Path $reportFolder "html") -ChildPath "$GPOName\_timestamp.html"
        $xmlPath = Join-Path -Path (Join-Path $reportFolder "xml") -ChildPath "$GPOName\_timestamp.xml"

        Get-GPOReport -Name $GPOName -ReportType Html -Path $htmlPath
        Get-GPOReport -Name $GPOName -ReportType Xml -Path $xmlPath

        Write-Host "Створений звіт для групової політики '$GPOName'" -ForegroundColor Green
    } catch {
        Write-Host "Помилка генерації звіту для групової політики '$GPOName': $_" -ForegroundColor Red
    }
}
```

Рисунок 3.1.1.8 –Лістинг коду функції “GpoGenerateReport”

За автоматизоване створення звітів у форматах XLM та HTML і збереження їх результатів у визначеному каталозі відповідає функція “GpoGenerateReport” (рис. 3.1.1.8). Вона приймає ім’я GPO та шлях до каталогу, де зберігатимуться звіти. Під час виконання функція створює відповідні підкаталоги для кожного формату (рис. 3.1.1.10), генерує унікальні імена файлів за часовою міткою та зберігає туди результати команди Get-GPOReport[16].

```
PS C:\Scripts\GPO_Scripts> Import-Module C:\Scripts\UserGPOModule.psm1 -Force -warningAction SilentlyContinue
GpoGenerateReport -GPOName "Login Message Policy"
Створений звіт для групової політики 'Login Message Policy'
```

Рисунок 3.1.1.9 –Приклад виконання “GpoGenerateReport”

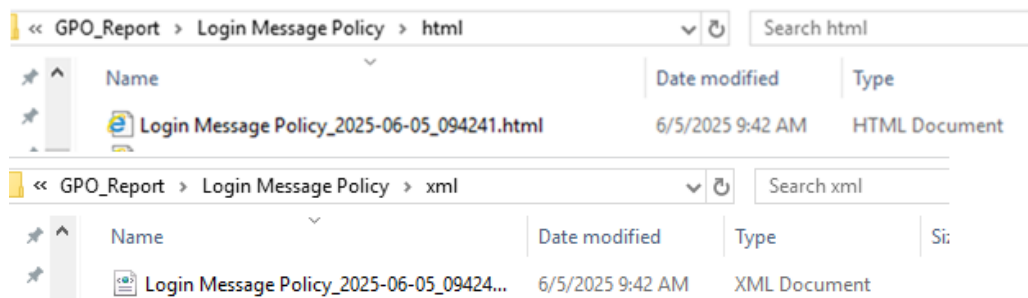


Рисунок 3.1.1.10 –Перевірка виконання “GpoGenerateReport”

```
function Input-Index {
    param (
        [Parameter(Mandatory=$true)]
        [Object[]]$ImportantList
    )
    do {
        $validInput = $true
        $TrueList = @()
        $ListIndexes = Read-Host "Введіть потрібні номери (через кому)"

        foreach ($index in $ListIndexes -split ',') {
            if (-not ($index -match '^d+$')) {
                Write-Host "Помилка - '$index' повинен бути числом." -ForegroundColor Red
                $validInput = $false
                break
            }
            elseif ([int]$index -ge $ImportantList.Count -or [int]$index -lt 0) {
                Write-Host "Помилка - '$index' виходить за межі списку" -ForegroundColor Red
                $validInput = $false
                break
            }
            $TrueList += $ImportantList[$index]
        }
    } while ($validInput -ne $true)
    return $TrueList
}
```

Рисунок 3.1.1.11 –Перевірка виконання “Input-Index”

У деяких скриптах нам знадобиться механізм ручного вибору елементів введених за індексами, тож “Input-Index” (рис. 3.1.1.11) створена для вирішення даної проблеми. Функція приймає параметр у вигляді списку об’єктів, виводить його з пронумерованими елементами та дозволяє користувачу ввести номер потрібного об’єкта. Для обробки некоректного введення передбачено дві перевірки: перша використовує регулярний вираз, що приймає лише числові значення, друга перевіряє, чи не перевищує введений номер межі списку. У разі успішного проходження перевірок функція повертає певну вибрану кількість значень об’єктів.

Остання функція “Write-GPOEventLog” (рис. 3.1.1.12) реалізує базову систему логування, основним і єдиним завданням якої є запис подій у текстовий файл залежно від значення параметра модифікації (рис. 3.1.1.13). Ця функція забезпечує надійне збереження інформації про дії, виконані в процесі адміністрування групових політик.

```

Function Write-GPOEventLog {
    param (
        [Parameter(Mandatory)]
        [string]$GPOName,

        [Parameter(Mandatory)]
        [ValidateSet("Edit", "Delete", "Restore", "WMI")]
        [string]$Mode
    )

    $logFilePath = "C:\Logs\GeneralLog.txt"
    $timestamp = Get-Date -Format "yyyy-MM-dd HH:mm:ss"

    try {
        switch ($Mode) {
            "Edit" { $action = "Edit GPO" }
            "Delete" { $action = "Delete GPO" }
            "Restore" { $action = "Restore GPO" }
            "WMI" { $action = "Set WMI filter on GPO" }
        }

        $logEntry = "[timestamp] $action - $GPOName"
        if (-not (Test-Path $logFilePath)) { New-Item -ItemType File -Path $logFilePath -Force | Out-Null }
        Add-Content -Path $logFilePath -Value $logEntry
    } catch {Write-Host "Сталася помилка при приписці події : $_" -ForegroundColor Red}
}

```

Рисунок 3.1.1.12 – Лістинг коду функції “Write-GPOEventLog”

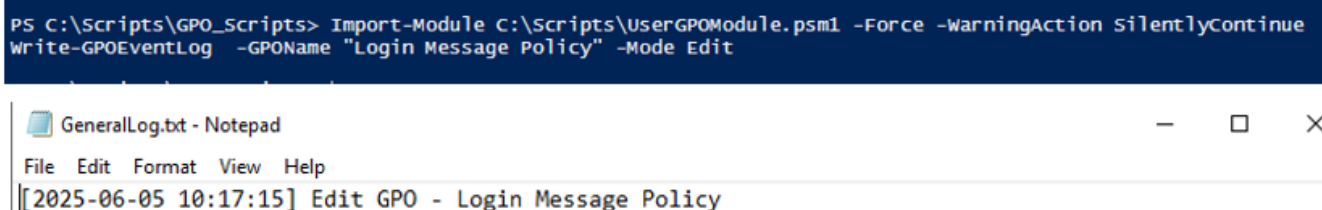


Рисунок 3.1.1.13 – Приклад використання функції “Write-GPOEventLog”

Оскільки в середовищі PowerShell не передбачено командлети, що відповідають за взаємодію WMI-фільтрів разом із груповими політиками, довелося розробити кілька базових функцій для зручної інтеграції цих компонентів через консоль, що усуває потребу в зверненні до графічного інтерфейсу GPMC. Для цього створено окремий користувацький модуль “WmiFilter”, котрий має забезпечити ефективне управління фільтрами в рамках адміністрування GPO.

За створення нового WMI-фільтру у Active Directory відповідає функція “New-WmiFilter” (Додаток Е) (рис. 3.1.1.14). Приймаючи назву фільтра, деяку кількість WMI-запитів, що не завжди обмежується одним значенням й опис за бажанням адміністратора. Залишається лише заповнити атрибути котрі потрібні для коректної роботи майбутнього об’єкта AD: згенерувати унікальний ідентифікатор фільтра, отримати поточний час, отримати корінь інформаційного дерева домену, формує спеціальний рядок WMI-запитів у форматі який підтримує служба GPO. У кінці створюється об’єкти типу msWMI-Som у вказаній гілці каталогу.

```

function New-WmiFilter {
    [CmdletBinding()]
    param (
        [Parameter(Mandatory = $true)]
        [string]$Name,
        [Parameter(Mandatory = $true)]
        [string[]]$Filter,
        [string]$Description = '',
        [string]$Server = $env:USERDNSDOMAIN
    )

    try {
        $RootDSE = Get-ADRootDSE -Server $Server
        $NamingContext = $RootDSE.DefaultNamingContext
        $wmiGuid = "{${[guid]::NewGuid()}"
        $creationDate = (Get-Date).ToUniversalTime().ToString("yyyyMMddHHmmss.ffffff-000")
        $filterString = "{0};" -f $Filter.Count
        foreach ($query in $Filter) {
            $filterString += "3;10;{0};WQL;root\CIMv2;{1};" -f $query.Length, $query
        }
        $attributes = @{
            "msWMI-Name" = $Name
            "msWMI-Param1" = $Description
            "msWMI-Param2" = $filterString
            "msWMI-Author" = "$env:USERNAME@$env:USERDNSDOMAIN"
            "msWMI-ID" = $wmiGuid
            "instanceType" = 4
            "msWMI-ChangeDate" = $creationDate
            "msWMI-CreationDate" = $creationDate
            "showInAdvancedViewOnly" = "TRUE"
            "distinguishedName" = "CN=$wmiGuid,CN=SOM,CN=WMIPolicy,CN=System,$NamingContext"
        }
        New-ADObject -Name $wmiGuid `
            -Type "msWMI-Som" `
            -Path "CN=SOM,CN=WMIPolicy,CN=System,$NamingContext" `
            -OtherAttributes $attributes `
            -Server $Server

        Write-Host "WMI-фільтр '$Name' створено успішно" -ForegroundColor Green
    } catch {
        Write-Error "Помилка створення WMI-фільтра: $_" -ForegroundColor Red
    }
}

```

Рисунок 3.1.1.14 – Лістинг скрипту функції “New-WmiFilter”

Для пошуку WMI-фільтру створено функцію “FindWmiFilter”. Вона може функціонувати у двох режимах, залежно від значення параметру “Mode” – через GPMC або ж безпосередньо в AD.

За обраного режиму “GPMC” функція використовуватиме COM-об’єкти бібліотеки Group Policy Management Console для пошуку всіх WMI-фільтрів у домені. Відфільтрувавши їх за назвою повертає відповідний об’єкт, якщо він знайдений. Якщо ні – виводиться попередження.

Кори адміністратор використає режим “AD” - підключиться до каталогу Active Directory і використає пошук об’єктів типу “msWMI-Som”. Якщо вказане ім’я фільтру як “AllWMI”, то повертає список усіх WMI-фільтрів створених у домені. Якщо ж задано конкретну назву, то виконується пошук лише одного фільтра за атрибутом “msWMI-Name”.

Щоб прив’язати WMI-фільтр до групової політики реалізовано “Set-GPOWmiFilter” (рис. 3.1.1.15). Звичайно, спочатку потрібно вказати два параметри, а саме ім’я GPO та фільтру. У процесі виконання відбувається пошук

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				62
Змн.	Арк.	№ докум.	Підпис	Дата		

об'єктів AD, за допомогою “FindWmiFilter” та командлети “Get-GPO”. Якщо обидва об'єкти знайдено, функція намагається встановити властивість “WmiFilter” для GPO, тобто прив'язати знайдений WMI-фільтр до цієї політики. У разі помилки виводиться відповідне повідомлення.

```
function Set-GPOWmiFilter
{
    [CmdletBinding()]
    Param(
        [string]$WmiFilterName,
        [string]$GPOName
    )
    BEGIN
    {
        $f = $MyInvocation.InvocationName
    }
    PROCESS
    {
        $WmiFilter = FindWmiFilter -WmiFilterName $WmiFilterName -Mode GPMC
        $GPOObject = $null

        if($GPOName)
        {
            $GPOObject = Get-GPO -Name $GPOName
            if(-not $GPOObject) { Write-Host "Не знайдено GPO за назвою '$GPOName'" -ErrorAction Stop -ForegroundColor Red }
        }

        Write-Host "$f - Застосовано WMI-фільтрацію '$($WmiFilter.Name)' до GPO '$($GPOObject.DisplayName)'" -ForegroundColor Green

        try { $GPOObject.WmiFilter = $WmiFilter }
        catch { Write-Error "Помилка при прив'язці WMI-фільтра до GPO: $_" }
    }
}
```

Рисунок 3.1.1.15 – Лістинг скрипту функції “Set-GPOWmiFilter”

Можливість видалити WMI-фільтр є важливим аспектом адміністрування, тож функція “Remove-WmiFilter” (рис. 3.1.1.16), чудово справляється з поставленою задачею. Після отримання вказаних параметрів, вона шукає вказаний фільтр у каталозі AD і знайшовши, функція перевіряє чи дозволено виконання операції, вивівши маленьке допоміжне віконечко з підтвердженням дії. Після погодження, використавши командлет “Remove-ADObject”, що видає об'єкт AD.

```
function Remove-WmiFilter {
    [CmdletBinding(SupportsShouldProcess = $true, ConfirmImpact = 'High')]
    param (
        [Parameter(Mandatory = $true)]
        [string]$WmiFilterName,
        [string]$Server = $env:USERDNSDOMAIN
    )
    try {
        $adFilter = FindWmiFilter -WmiFilterName $WmiFilterName -Mode AD

        if ($PSCmdlet.ShouldProcess($adFilter.DistinguishedName, "Видалити WMI-фільтр")) {
            Remove-ADObject -Identity $adFilter.DistinguishedName -Confirm:$false -Server $Server
            Write-Host "Видалено: $WmiFilterName" -ForegroundColor Green
        }
    } catch { Write-Error "Помилка: $_" }
}
```

Рисунок 3.1.1.16 – Лістинг скрипту функції “Remove-WmiFilter”

Залишилося лише продумати функцію котра відповідає за редагування раніше створеного фільтра з цією метою реалізована “Set-WmiFilter” (рис. 3.1.1.17). Щоб її викликати адміністратору доведеться обов'язково ввести назву WMI-

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				
Змн.	Арк.	№ докум.	Підпис	Дата		63

фільтру та WMI-запит. Далі відбудеться пошук даного фільтру у AD, якщо його було знайдено формується новий набір параметрів для оновлення. Заміняється значення запиту на нове, а також за наявності змінюється ще й опис.

```
function Set-WmiFilter {
    [CmdletBinding()]
    param (
        [Parameter(Mandatory = $true)]
        [string]$WmiFilterName,
        [string[]]$Filter,
        [string]$Description = '',
        [string]$Server = $env:USERDNSDOMAIN
    )

    try{
        $adFilter = FindWmiFilter -WmiFilterName $WmiFilterName -Mode AD
        $newParams = @{}
        if ($Filter) {
            $filterString = "{0};" -f $Filter.Count
            foreach ($query in $Filter) {
                $filterString += "3;10;{0};WQL;root\CIMv2;{1};" -f $query.Length, $query
            }
            $newParams["msWMI-Parm2"] = $filterString
        }
        # Якщо задано Description - оновлюємо msWMI-Parm1
        if ($Description) {
            $newParams["msWMI-Parm1"] = $Description
        }
        if ($newParams.Count -eq 0) {
            Write-Warning "Нічого не задано для оновлення."
            return
        }
        if ($PSCmdlet.ShouldProcess($adFilter.DistinguishedName, "Оновити WMI-фільтр")) {
            Set-ADObject -Identity $adFilter.DistinguishedName -Replace $newParams -Server $Server
            Write-Host "Оновлено фільтр '$Name'" -ForegroundColor Green
        }
    }
    catch { Write-Error "Помилка: $_" }
```

Рисунок 3.1.1.17 – Лістинг скрипту функції “Set-WmiFilter”

3.1.2 Налаштування групових політик у доменному середовищі

Перед налаштуванням адміністрування групових політик (GPO) розглянемо загальний принцип їх створення. Для цілісності викладу припустимо, що попередня частина дипломного проєкту не стосувалася аналізу домену з повністю налаштованими груповими політиками.

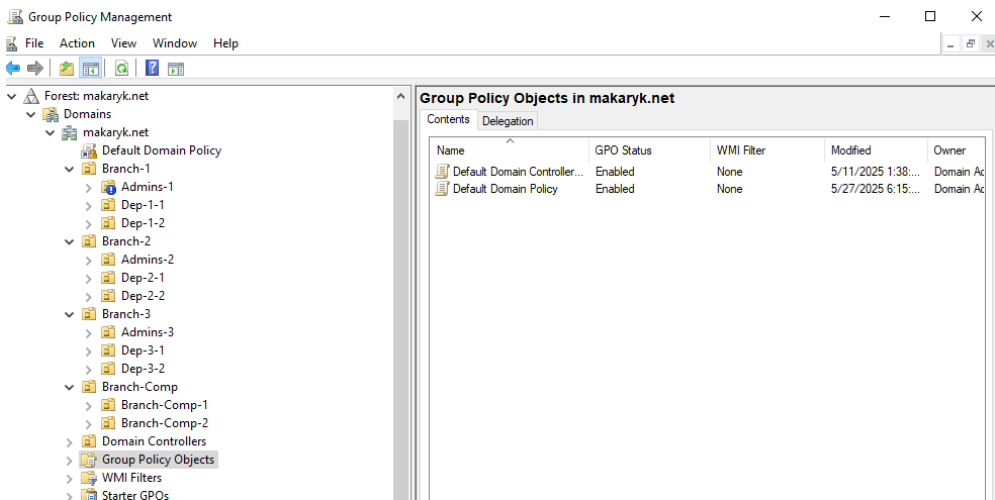


Рис. 3.1.2.1 – Відсутність налаштованих користувацьких GPO

Усі GPO розроблено на основі концептуальної блок-схеми (рис. 3.1.2.2), яка чітко визначає етапи їх створення та впровадження.

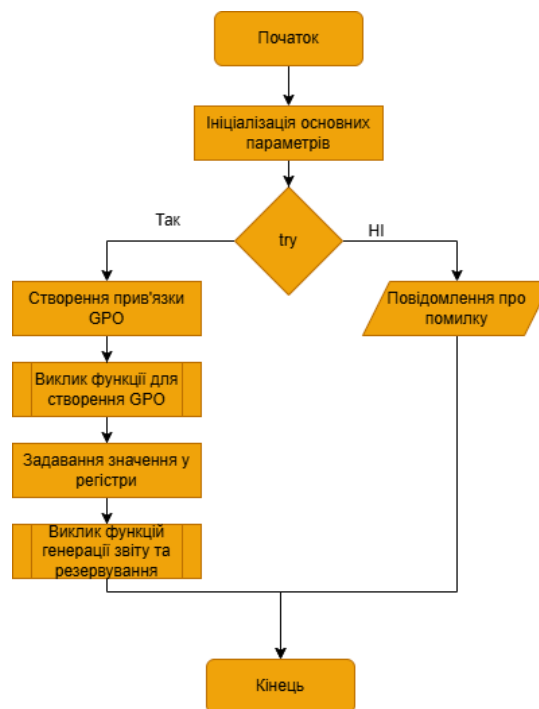


Рисунок 3.1.2.2 – Відсутність налаштованих користувачьких GPO
Приклад коду створення групової політики за блок-схемою (Додаток Б):

```

Import-Module GroupPolicy
Import-Module C:\Scripts\UserGPOModule.psm1 -Force -WarningAction SilentlyContinue

$GPOName = "Screen Lock Policy"
$DomainName = "makaryk.net"
$KeyPath = "HKCU\Software\Policies\Microsoft\Windows\Control Panel\Desktop"

try {
    CreateGPO -GPOName $GPOName -DomainName $DomainName

    $ScreenSaverPath = "C:\Windows\System32\scrnsave.scr"
    Set-GPRegistryValue -Name $GPOName -Key $KeyPath -ValueName "SCRNSAVE.EXE" -Type String -Value $ScreenSaverPath
    Set-GPRegistryValue -Name $GPOName -Key $KeyPath -ValueName "ScreenSaveTimeout" -Type String -Value "600"
    Set-GPRegistryValue -Name $GPOName -Key $KeyPath -ValueName "ScreenSaverIsSecure" -Type String -Value "1"

    New-GPLink -Name $GpoName -Target "DC=makaryk,DC=net"

    ExpandedBackup-GPO -GPOName $GpoName -DomainName $DomainName
    GpoGenerateReport -GPOName $GpoName
    Write-GPOEventLog -GPOName $GpoName -Mode Create
} catch { Write-Host "Сталася помилка при створенні GPO '$GPOName': $_" -ForegroundColor Red }
  
```

Рисунок 3.1.2.3 – Лістинг коду створення групової політики “Screen Lock Policy”

Основні командлети, що застосовуються для створення та налаштування групової політики у даному скрипті це:

- “New-GPO” – створює новий об’єкт групової політики у заданому домені. Командлет дозволяє визначити ім’я та контейнер зберігання політики під час створення.

- “Set-GPRegistryValue” – задає або змінює значення реєстру в об’єкті групової політики (GPO). Командлет застосовується для конфігурації системних і користувачьких параметрів через політику, вказуючи шлях до ключа, тип, ім’я та значення параметра.
- “New-GPLink” – створює зв’язок між GPO та об’єктом Active Directory, таким як сайт, домен або OU. Командлет дозволяє вказати порядок застосування та режим примусовості.

На основі отриманих даних базовий скрипт створює новий об’єкт групової політики. Для нього задаються параметри реєстру, які визначають конфігурацію новоствореної політики. Після цього здійснюється прив’язка об’єкта до домену або організаційної одиниці. На завершальному етапі виконуються функції резервного копіювання, генерації звіту та логування нової групової політики, що забезпечує її повну інтеграцію в систему адміністрування (рис. 3.1.2.4).

```
PS C:\Scripts\GPO_Scripts> C:\Scripts\GPO_Scripts\Screen Lock Policy.ps1
Створено новий GPO: Screen Lock Policy

DisplayName      : Screen Lock Policy
DomainName       : makaryk.net
Owner            : MAKARYK\Domain Admins
Id               : d0623687-68eb-4add-92a3-6acbda6c6e5b
GpoStatus        : AllSettingsEnabled
Description       :
CreationTime     : 6/5/2025 11:40:36 AM
ModificationTime : 6/5/2025 11:40:36 AM
UserVersion      : AD Version: 1, SysVol Version: 1
ComputerVersion  : AD Version: 0, SysVol Version: 0
WmiFilter        :

DisplayName      : Screen Lock Policy
GpoId            : d0623687-68eb-4add-92a3-6acbda6c6e5b
Id               : cd2af8a0-644a-47c7-a2e5-377824a860cf
BackupDirectory  : C:\GPO_Backups\Screen Lock Policy\2025-06-05_114037
CreationTime     : 6/5/2025 11:40:37 AM
DomainName       : makaryk.net
Comment         :

GPO 'Screen Lock Policy' успішно резервовано у 'C:\GPO_Backups\Screen Lock Policy\2025-06-05_114037'.
Створений звіт для групової політики 'Screen Lock Policy'
```

Рисунок 3.1.2.4 – Неповний вивід створення GPO “Screen Lock Policy”

Крім створення абсолютно нових GPO Active Directory надає можливість відредагувати наявну “Default Domain Policy” за допомогою командлети “Set-ADDefaultDomainPasswordPolicy” (рис. 3.1.2.5). Наприклад скрипт політики під назвою “PasswordPolicy” встановлює політику паролів для домену, згідно якої надається збереження до трьох паролів користувачів, максимальний термін активності паролів 30, після чого їх попросять змінити з мінімальною довжиною у 8 символів із певного рівня складності (рис. 3.1.2.6).

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				
Змн.	Арк.	№ докум.	Підпис	Дата		66

```

Import-Module GroupPolicy
Import-Module C:\Scripts\UserGPOModule.psm1 -Force -WarningAction SilentlyContinue

$DomainName = "makaryk.net"

try {
    Set-ADDefaultDomainPasswordPolicy -Identity $DomainName -PasswordHistoryCount 3
    Set-ADDefaultDomainPasswordPolicy -Identity $DomainName -MaxPasswordAge (New-TimeSpan -Days 30)
    Set-ADDefaultDomainPasswordPolicy -Identity $DomainName -MinPasswordLength 8
    Set-ADDefaultDomainPasswordPolicy -Identity $DomainName -ComplexityEnabled $true

    Write-Host "Успішне оновлення політик паролів" -ForegroundColor Green

    Get-ADDefaultDomainPasswordPolicy
    ExpandedBackup-GPO -GPOName $GpoName -DomainName $DomainName
    GpoGenerateReport -GPOName $GpoName
    Write-GPOEventLog -GPOName $GpoName -Mode Create
}
catch { Write-Host "Сталася помилка при створенні GPO '$GPOName': $_" -ForegroundColor Red }

```

Рисунок 3.1.2.5 – Лістинг коду “PasswordPolicy”

```

PS C:\Scripts\GPO_Scripts> C:\Scripts\GPO_Scripts\PasswordPolicy.ps1
Успішне оновлення політик паролів

ComplexityEnabled           : True
DistinguishedName           : DC=makaryk,DC=net
LockoutDuration              : 00:30:00
LockoutObservationWindow     : 00:30:00
LockoutThreshold             : 0
MaxPasswordAge               : 30.00:00:00
MinPasswordAge               : 1.00:00:00
MinPasswordLength            : 8
objectClass                  : {domainDNS}
objectGuid                   : ed2e44b4-e1ed-4b30-a183-4aa2976f1ba3
PasswordHistoryCount         : 3
ReversibleEncryptionEnabled  : False

DisplayName                  : Screen Lock Policy
GpoId                        : d0623687-68eb-4add-92a3-6acbdafc6e5b
Id                            : 7347fbac-a407-438d-a01b-7608ce0780d1
BackupDirectory              : C:\GPO_Backups\Screen Lock Policy\2025-06-05_122251
CreationTime                  : 6/5/2025 12:22:51 PM
DomainName                   : makaryk.net
Comment                      :

GPO 'Screen Lock Policy' успішно резервовано у 'C:\GPO_Backups\Screen Lock Policy\2025-06-05_122251'.
Створений звіт для групової політики 'Screen Lock Policy'

```

Рисунок 3.1.2.6 – Виконання скрипту “PasswordPolicy”

Для реалізації масового адміністрування групових політик виконано відповідні скрипти, функціонал яких базується на раніше описаних можливостях GPO. Результати виконання відображено за допомогою командлета Get-GPO –All (рис. 3.1.2.7). На цьому етапі система підготовлена до комплексного управління груповими політиками.

```

PS C:\Scripts\GPO_Scripts> Get-GPO -All | Select-Object -Property DisplayName | Format-Table -AutoSize

DisplayName
-----
Restrict CMD and PowerShell Access
Disk Quotas Policy
Default Domain Policy
Additional Installation Policies
Default Domain Controllers Policy
Scheduled Scan for Windows Defender
Disable Forced System Restarts
Audit Policy Configuration
Login Message Policy
Deny removable disk
Restrict Software Installation
Screen Lock Policy
Restrict registry editing tools and control panel

```

Рисунок 3.1.2.7 – Виведення усіх налаштованих GPO на домені

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				67
Змн.	Арк.	№ докум.	Підпис	Дата		

3.2 Автоматизація адміністрування через PowerShell

3.2.1 Створення PowerShell-скриптів для управління GPO

PowerShell скрипт з назвою “Delete-Backup” (Додаток В) розроблений з ціллю оптимізувати аспект адміністрування, що тісно пов’язаний з резервуванням та відновлення групових політик. Перший чином користувачу виведеться меню з вибором можливих дій та закінченням програми, залежно від вводу запусяться три наступні функції.

“Backup-GPOs” (рис. 3.2.1.1) розроблена спеціально для резервування GPO у домені. Формує перелік усіх групових політик та відображає їх із номерами для зручного вибору. За допомогою функції Input-Index користувач може обрати одну або кілька політик для створення резервних копій. Для кожної обраної політики автоматично створюється окрема папка з назвою, що поєднує ім’я GPO та дату з часом, куди зберігається резервна копія. Усі дії записуються до текстового журналу, що забезпечує відстеження історії резервного копіювання.

```
function Backup-GPOs {  
    $AllGPOs = Get-GPO -All | Sort-Object DisplayName  
    for ($i = 0; $i -lt $AllGPOs.Count; $i++) {  
        Write-Host "$i. $($AllGPOs[$i].DisplayName)"  
    }  
  
    $SelectedGPOs = Input-Index -ImportantList $AllGPOs  
    $Date = Get-Date -Format "yyyyMMdd_HH:mm:ss"  
  
    foreach ($GPO in $SelectedGPOs) {  
        $GPOName = $GPO.DisplayName  
        $BackupPath = "$BackupBasePath\$GPOName\$Date"  
  
        if (-not (Test-Path $BackupPath)) {  
            New-Item -ItemType Directory -Path $BackupPath -Force | Out-Null  
        }  
  
        Backup-GPO -Name $GPOName -Path $BackupPath -Domain "makaryk.net"  
        Write-Host "GPO $GPOName успішно резервовано у $BackupPath." -ForegroundColor Green  
        Add-Content -Path $LogFile -Value "$(Get-Date): Резервна копія GPO '$GPOName' створена у '$BackupPath'. "  
    }  
}
```

Рисунок 3.2.1.1 – Лістинг коду функції “Backup-GPOs”

Наступною у черзі йде функція видалення групової політики зі списку активних на домені, для цього знов ж таки виводиться пронумерований список застосованих GPO, після чого користувачу дають можливість обрати які саме політики будуть видалені за допомогою командлети “Remove-GPO” (рис. 3.2.1.2).

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				
Змн.	Арк.	№ докум.	Підпис	Дата		68

```
function Remove-GPOs {
    $AllGPOs = Get-GPO -All | Sort-Object DisplayName
    for ($i = 0; $i -lt $AllGPOs.Count; $i++) {
        Write-Host "$i. $($AllGPOs[$i].DisplayName)"
    }

    $SelectedGPOs = Input-Index -ImportantList $AllGPOs
    foreach ($GPO in $SelectedGPOs) {
        Remove-GPO -Name $GPO.DisplayName -Confirm:$false
        Write-Host "GPO '$($GPO.DisplayName)' успішно видалено." -ForegroundColor Green
        Write-GPOEventLog -GPOName $GPO -Mode Delete
    }
}
```

Рисунок 3.2.1.2 – Лістинг коду функції “Remove-GPOs”

Остання функція забезпечує можливість відновлення однієї групової політики з наявних резервних копій (рис. 3.2.1.3). Спочатку вона сканує основну директорію, в якій знаходяться резервні копії. Далі користувач обирає політику для відновлення. Після цього відображається список доступних резервних копій для обраної політики, відсортований за датами, з якого користувач робить свій вибір. Скрипт відновлює політику в Active Directory.

```
function Restore-GPOFromBackup {
    $GPOFolders = Get-ChildItem -Path $BackupBasePath -Directory | Sort-Object Name
    if ($GPOFolders.Count -eq 0) {
        Write-Host "Немає жодної резервної копії для відновлення." -ForegroundColor Yellow
        return
    }

    Write-Host "Оберіть політику для відновлення:"
    for ($i = 0; $i -lt $GPOFolders.Count; $i++) {
        Write-Host "$i. $($GPOFolders[$i].Name)"
    }

    $SelectedFolder = (Input-Index -ImportantList $GPOFolders)[0]
    $GPOName = $SelectedFolder.Name
    $FullPath = Join-Path $BackupBasePath $GPOName

    $Backups = Get-ChildItem -Path $FullPath -Directory | Sort-Object Name
    if ($Backups.Count -eq 0) {
        Write-Host "Для '$GPOName' не знайдено жодної резервної копії." -ForegroundColor Red
        return
    }

    Write-Host "Оберіть резервну копію для відновлення:"
    for ($i = 0; $i -lt $Backups.Count; $i++) {
        Write-Host "$i. $($Backups[$i].Name)"
    }

    $SelectedBackup = (Input-Index -ImportantList $Backups)[0]
    $BackupPath = $SelectedBackup.FullName
    Restore-GPO -Name $GPOName -Domain "makaryk.net" -Path $BackupPath
    Write-Host "GPO $GPOName успішно відновлено з резервної копії: '$BackupPath'." -ForegroundColor Green
    Write-GPOEventLog -GPOName $GPOName -Mode Restore
}
```

Рисунок 3.2.1.3 – Лістинг коду функції “Restore-GPOs”

3.2.2 Реалізація скриптів для масового внесення зміни у політики

У цьому пункті розглянуто скрипти для внесення змін до групових політик, які забезпечують гнучке адміністрування їхньої конфігурації. Розроблено два скрипти зі схожим, але різним за застосуванням функціоналом. Скрипт “EditGPO”

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				
Змн.	Арк.	№ докум.	Підпис	Дата		69

(Додаток Г) призначений для редагування ключів реєстру групових політик, пропонуючи кілька методів модифікації параметрів залежно від потреб адміністратора. “EditLinkGPO” (Додаток Г) відповідає за адміністрування зв’язків GPO з організаційними одиницями або доменом, забезпечуючи точне налаштування їхньої прив’язки. Обидва коди оптимізують процес управління груповими політиками, дозволяючи швидко вносити зміни та забезпечуючи їх коректну інтеграцію в доменну структуру

Отже почнемо з “EditGPO” у якому реалізовано можливість редагування як через скрипти запуску, так і за допомогою безпосередніх змін в реєстрових ключах, які пов’язані з конкретними груповими політиками.

Сенс функції “EditScriptInISE” (рис. 3.2.2.1) полягає в тому, щоб дати адміністратору можливість ввести та відкрити у середовищі PowerShell ISE потрібний скрипт. Цим самим забезпечується гнучкість у модифікації та вдосконаленні групових політик, дозволяючи користувачу ефективно адаптувати їх до специфічних потреб адміністрування.

```
function EditScriptInISE {
    param (
        [string]$ScriptsFolder
    )

    $scripts = Get-ChildItem -Path $ScriptsFolder -Filter *.ps1 | Select-Object -ExpandProperty Name

    if ($scripts.Count -eq 0) {
        Write-Host "У вибраній папці немає скриптів .ps1." -ForegroundColor Yellow
        return
    }

    $scripts | ForEach-Object { Write-Host "$($_)" }
    $scriptName = Read-Host "Введіть назву скрипту для редагування (включаючи розширення .ps1)"

    $scriptPath = Join-Path -Path $ScriptsFolder -ChildPath $scriptName

    if (Test-Path $scriptPath) {
        ise $scriptPath
    } else {
        Write-Host "Скрипт не знайдено за вказаною назвою: $scriptName" -ForegroundColor Red
    }
    PostEditMenu -scriptPath $scriptPath
}
```

Рисунок 3.2.2.1 – Лістинг коду функції “EditScriptInISE”

Паралельно до відкриття скрипту у середовищі PowerShell ISE активується функція “PostEditMenu” (рис. 3.2.2.2) у якій пропонується вибір між повернення до головного меню та видалення групової політики і застосування відредагованого коду. Важливо зазначити, що адміністратор повинен власноруч ввести назву GPO

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				70
Змн.	Арк.	№ докум.	Підпис	Дата		

для видалення, що є слабким моментом даної функції, оскільки завжди є ризик помилки пов'язаний з людським фактором, що призведе до випадкового видалення іншої політики. Застосування оновленого коду супроводжується викликом розглянутих раніше функцій у пункті 3.1.1, а саме: “ExpandedBackup-GPO”, “GpoGenerateReport” та “Write-GPOEventLog”. Крім того для моніторингу змін додана “CompareLastAndNewGPO”, мета якої – порівняти минулу і відредаговану версію скрипта, записавши відмінності у окремий текстовий файл.

```
function PostEditMenu {
    param (
        [string]$ScriptPath
    )

    Write-Host "( — — ) ~> Якщо скрипт з GPO було відредаговано " -ForegroundColor Cyan
    Write-Host "1. Видалити минулу GPO"
    Write-Host "2. Виконати оновлений скрипт"
    Write-Host "0. Вийти в головне меню"
    $choice = Read-Host "Ваш вибір"

    switch ($choice) {
        "1" {
            $GPOName = Get-GPOName -Prompt "Введіть ім'я GPO для видалення"
            if ($GPOName) {
                Remove-GPO -Name $GPOName -Confirm:$false
                Write-Host "GPO '$GPOName' успішно видалено." -ForegroundColor Green
            }
            PostEditMenu -scriptPath $scriptPath
        }
        "2" {
            if (Test-Path $ScriptPath) {
                Write-Host "Виконання скрипта: $ScriptPath" -ForegroundColor Cyan
                & $ScriptPath
                ExpandedBackup-GPO -GPOName $GPOName -DomainName "makaryk.net"
                GpoGenerateReport -GPOName $GPOName
                CompareLastAndNewGPO
                Write-Host "Скрипт виконано успішно." -ForegroundColor Green
            } else {
                Write-Host "Скрипт не знайдено за вказаним шляхом: $ScriptPath" -ForegroundColor Red
            }
        }
        "0" { Main }
        default { Write-Host "Неправильний вибір. Спробуйте ще раз" -ForegroundColor Red }
    }
}
```

Рисунок 3.2.2.2 – Лістинг коду функції “PostEditMenu”

Повернімося до другого методу редагування, що включає у себе взаємодію з значенням реєстру за допомогою командлет “Set-GPRegistryValue”. Цей процес умовно складається з трьох взаємопов'язаних функцій, які забезпечують зручний та контрольований підхід до зміни політик. Спочатку адміністратор звертається до функції “FindGPOSettings” яка виводить перелік усіх політик налаштованих у домені, пропонує вибрати одну для подальшої роботи, з метою виклику “Get-GPAllRegistryValues”, котра поверне усі налаштовані значення реєстру вказаної GPO, після чого автоматично викликає функцію “EditGPO”. Остання просить ввести шлях до відповідного ключа реєстру, назву параметра й нове значення, після чого намагається зчитати поточні властивості параметра з локального реєстру,

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				71
Змн.	Арк.	№ докум.	Підпис	Дата		

визначаючи його тип, щоб правильно передати це у командлет зміни політики. Далі як і в першому варіанті відбувається резервування, журналювання, порівняння звітів та логування.

```
function EditGPO {
    param (
        [string]$GPOName
    )
    Write-Host "$GPOName - в функції редагування" -ForegroundColor Cyan
    $GPOFullKey = Read-Host "Введіть повний шлях ключа реєстру для редагування"
    $GPOValueName = Read-Host "Введіть ім'я вартості(?)"
    $GPOValue = Read-Host "Введіть нове значення ГП"

    $RegistryPath = NormalizeRegistryPath -RegistryPath $GPOFullKey

    try {
        $registryKey = Get-Item -Path $RegistryPath -ErrorAction Stop

        if ($registryKey.GetValueNames() -contains $GPOValueName) {
            $type = $registryKey.GetValueKind($GPOValueName)
        }
        else { Write-Host "Значення '$GPOValueName' не знайдено в ключі" -ForegroundColor Red }
    } catch { Write-Host "Ключ реєстру не знайдено: $RegistryPath" -ForegroundColor Red }

    if (-not (Set-GPRegistryValue -Name $GPOName -Key $GPOFullKey -ValueName $GPOValueName -Value $GPOValue -Type `
    $type -ErrorAction SilentlyContinue)) {
        Write-Host "Не вдалося оновити групову політику" -ForegroundColor Red
    } else {
        Write-Host "Групову політику успішно оновлено" -ForegroundColor Green
        Get-GPO -Name $GPOName
        ExpandedBackup-GPO -GPOName $GPOName -DomainName "makaryk.net"
        GpoGenerateReport -GPOName $GPOName
        CompareLastAndNewGPO
    }
}
Main
```

Рисунок 3.2.2.3 – Лістинг коду функції “EditGPO”

Розширений скрипт PowerShell “EditGPOLink” реалізує зручний консольний інструмент для адміністрування прив’язок групових політик у Active Directory. Його призначення у спробі спрощення керування зв’язками GPO з доменом та організаційними одиницями шляхом створення, редагування та видалення.

Якщо не брати до уваги головну функцію “Main” єдина задача якої надати адміністратору вибір між тим які дії над прив’язуванням виконуються, то залишається розглянути основні методи керування посиланнями.

Створення зв’язків виконує функція “MassCreate-GpoLinks”, яка запитує в адміністратора параметри, такі як увімкнення політики та примусове застосування. Підтримує два режими: масове створення зв’язків із однаковими параметрами для всіх OU і GPO або ж покрокове налаштування для кожної комбінації. Після створення зв’язку активуються резервне копіювання “ExpandedBackup-GPO” та генерація звіту “GpoGenerateReport”.

Функція Get-AllGpoTargets створює уніфікований список об’єктів, до яких можна застосовувати GPO, включаючи домен і всі OU, із зазначенням їхніх імен і

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				
Змн.	Арк.	№ докум.	Підпис	Дата		72

DistinguishedName для роботи з командлетами GPO. Це забезпечує чітке визначення цільових об'єктів для прив'язки.

За модифікацію наявних посилань відповідає "Edit-GPOLinks". Вона визначає об'єкти, до яких прив'язана вибрана політика, і, якщо таких об'єктів кілька, інформує користувача для уточнення. У разі єдиного об'єкта параметри змінюються через командлет "Set-GPLink".

"Remove-GpoLinks" викликається для видалення зв'язків GPO за двома сценаріями: за конкретною політикою або за доменом чи OU. У першому випадку користувачу пропонується вибір серед пов'язаних GPO, і для кожної політики перевіряються всі можливі цільові об'єкти на наявність зв'язку. У другому сценарії видаляються вибрані GPO зі списку зв'язків обраного об'єкта.

Інтерактивність забезпечують функції "Print-Select" і "Input-Index". Перша відображає нумерований перелік елементів (наприклад, GPO або OU) за їхньою властивістю "Name" чи "DisplayName", а друга дозволяє користувачеві вводити індекси вибраних об'єктів. "Input-Index" перевіряє, чи є введені значення цілими числами в межах допустимого діапазону, і в разі помилки пропонує повторити введення, що підвищує надійність взаємодії. Як приємний бонус існує функція з пошуку GPO без активних прив'язок у домені чи OU – "Get-UnlinkedGPOs".

3.2.3 Проектування засобів управління дозволами та безпекою GPO

У даному пункті будуть розглянуті скрипти для управління дозволами та безпеки, розширюючи можливості адміністрування групових політик у межах домену. Розроблено PowerShell скрипт, що дозволяє керувати доступом користувачів до окремих GPO, а також другий скрипт який тісно пов'язаний з інтегрованим користувачьким модулем у якому реалізуються функції з менеджменту WMI-фільтрування через консольний інтерфейс. Ці інструменти забезпечують точне налаштування прав доступу та ефективне застосування фільтрів для оптимізації роботи з GPO.

Щоб захистити політики від несанкціонованих змін та обмежити вплив певної групи осіб використовується "GPPermissionEdit" (Додаток Д), що дозволяє ад-

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				
Змн.	Арк.	№ докум.	Підпис	Дата		73

міністратору оперативно задати деякій кількості користувачів чи усіх облікових записів організаційних одиниць, за допомогою точкового вибору через функцію “Input-Index”. Сформований список даних передається до “EditPermissionMenu” (рис. 3.2.3.1), яка приймає у параметрах інформацію про користувачів та групових політик до яких змінюватиметься рівень доступу. Адміністратору дається можливість вибрати який саме рівень прав застосується, після чого командлет “Set-GPPermission” застосовує зміни. Як і в попередніх кодах, виконується створення резервної копії та генерації звіту.

```

4 function EditPermissionMenu {
5     param (
6         [Parameter(Mandatory=$true)]
7         [Object[]]$UserOU,
8         [Parameter(Mandatory=$true)]
9         [Object[]]$UsedGPO
10    )
11
12    Write-Host "o(=▽=) ---Оберіть тип доступу---" -ForegroundColor Cyan
13    Write-Host "1. Лише перегляд GPO "
14    Write-Host "2. Редагування GPO"
15    Write-Host "3. Редагування, видалення, зміна безпеки доступу"
16    Write-Host "4. Видалити доступ"
17    $PerMenu = @{
18        "1" = "GpoRead"
19        "2" = "GpoEdit"
20        "3" = "GpoEditDeleteModifySecurity"
21        "4" = "None"
22    }
23
24    do {
25        $validChoice = $true
26        $choice = Read-Host "Введіть ваш вибір (1-4)"
27        if ($PerMenu.ContainsKey($choice)) {
28            $Permission = $PerMenu[$choice]
29            foreach ($GPOName in $UsedGPO) {
30                foreach ($User in $UserOU) {
31                    try {
32                        Set-GPPermission -Name $GPOName.DisplayName -TargetName $User.SamAccountName -TargetType User -PermissionLevel $Permission
33                        Write-Host "Встановлено дозвіл на $Permission на $GPOName для $($User.SamAccountName)" -ForegroundColor Green
34                        ExpandedBackup-GPO -GPOName $GPOName -DomainName "makaryk.net"
35                        GpoGenerateReport -GPOName $GPOName
36                    } catch { Write-Host "Помилка встановлення дозволу для '$($User.SamAccountName)' на '$GPOName': $_" -ForegroundColor Red }
37                }
38            }
39        } else {
40            Write-Host "Неправильний вибір. Спробуйте ще раз." -ForegroundColor Red
41            $validChoice = $false
42        }
43    } while (-not $validChoice)
44 }
45
46
47

```

Рисунок 3.2.3.1 – Лістинг коду функції “EditPermissionMenu”

Щоб уникнути надмірної кількості групових політик і складного керування груповим членством, у багатьох випадках доцільно використати WMI-фільтрування. Воно дозволяє задати чіткі умови, за яких конкретна політика буде застосована лише до відповідних кінцевих пристроїв, не важливо контролер чи звичайний комп’ютер користувача - наприклад, до хоста з певною версією Windows, обсягом оперативної пам’яті чи у деяких випадках в окремі дні тижня. Фільтрування значно спрощує структуру GPO, зменшує кількість необхідних груп без шкоди для точності налаштувань. Саме з метою був створений скрипт “WorkWMI”.

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				74
Змн.	Арк.	№ докум.	Підпис	Дата		

Головною функцією програми слугує “Main”, що дає користувачі можливість обрати відповідні дії стосовно WMI-фільтрів, як наприклад: створення нового фільтру, прив’язка до групової політики, його редагування та видалення. Як стало відомо з пункту 3.1.1, у середовищі PowerShell не передбачені засоби, щоб на пряму менеджментити, як GPO, тож раніше були описані функції які дозволяють зручно виконувати поставлені завдання. Уважно проаналізувавши блок-схему (рис 3.2.3.2) можна зрозуміти основний принцип роботи.

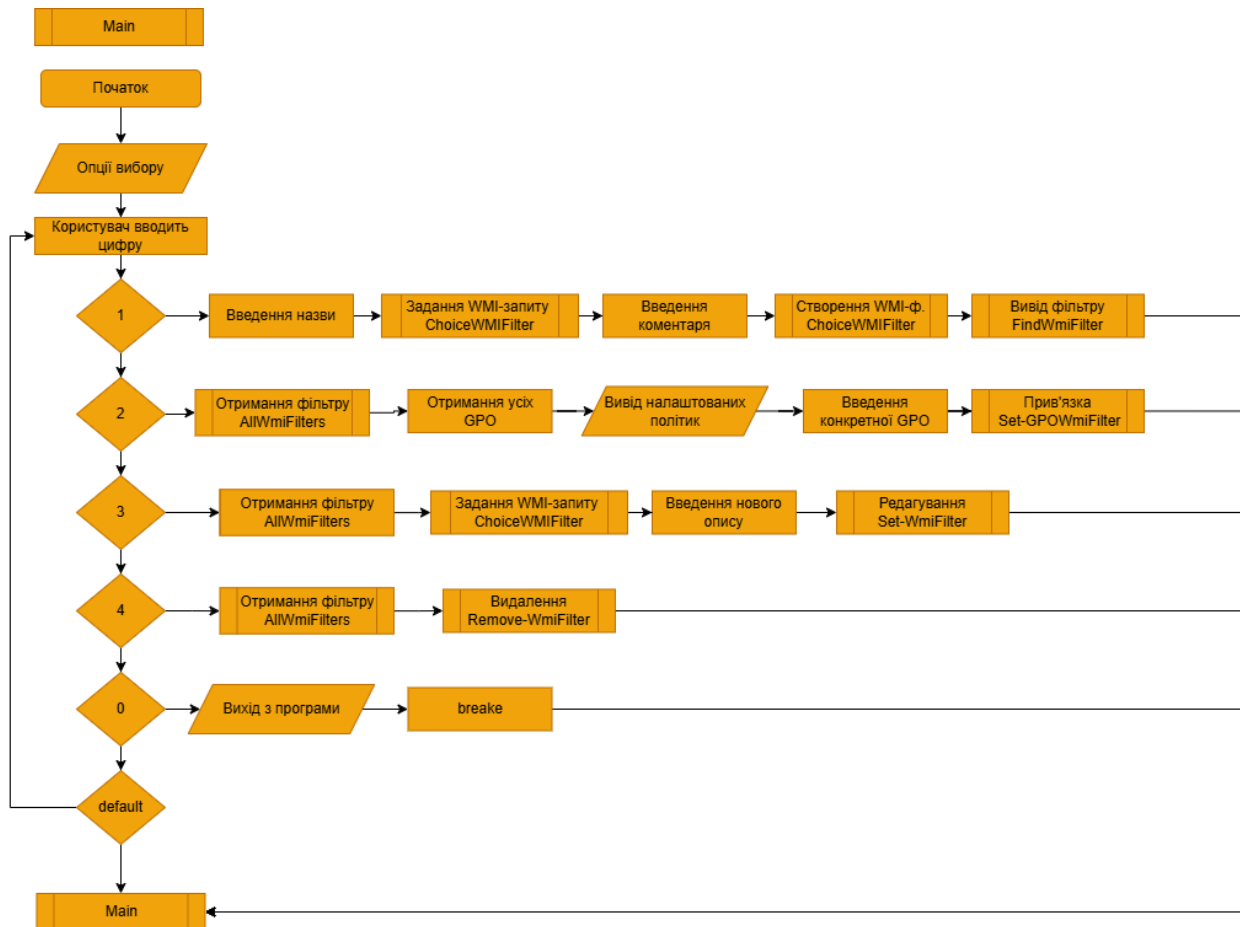


Рисунок 3.2.3.2 – Блок-схема функції “Main”

Створення WMI-фільтру відбувається шляхом введення адміністратором назви фільтру, далі активується функція для вибору методу заповнення WMI-запиту “ChoiceWmiFilter”, перший “WmiRegular” дозволяє ввести довільний запит за умови його коректного запису, що відповідає базовій структурі WQL-запиту з використанням ключових слів SELECT, FROM і WHERE, перевіряється за допомогою регулярного виразу[17].

Другий “WmiFiterNoteList” (рис. 3.2.3.3) призначений для інтерактивного вибору WMI-запиту з попереднього підготовлених текстових файлах (рис. 3.2.3.4-5). Окремо зчитавши назви фільтрів та їх запити, перевіряє відповідність за кількістю, після чого пропонує вибрати необхідний варіант за номером. У разі правильного вводу повертається відповідний WMI-запит. Останнім кроком перед викликом функції створення WMI-фільтру є задання довільного коментаря.

```
function WmiFiterNoteList {
    $WmiTitles = Get-Content "C:\Scripts\Scripts\WMIFilter-list\OS-list.txt"
    $WmiFilters = Get-Content "C:\Scripts\Scripts\WMIFilter-list\Filter-list.txt"

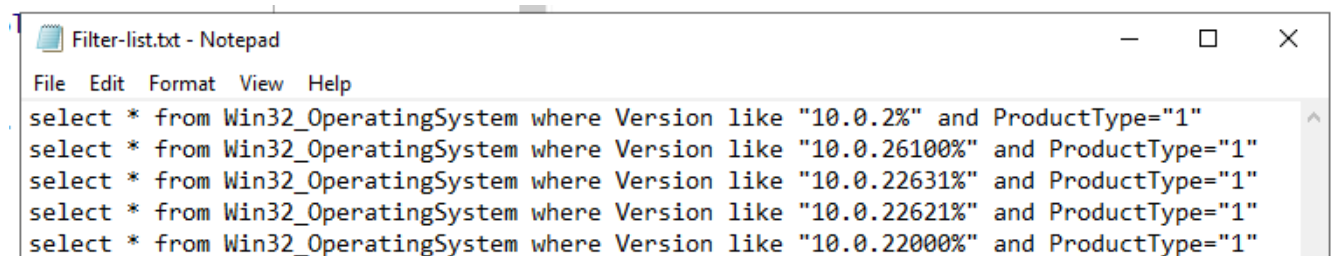
    if ($WmiTitles.Count -ne $WmiFilters.Count) {
        Write-Error "Кількість заголовків і фільтрів не збігається." -ForegroundColor Red
        return $null
    }

    Write-Host "_(з)_ Виберіть WMI-фільтр для застосування:`n" -ForegroundColor Cyan
    for ($i = 0; $i -lt $WmiTitles.Count; $i++) {
        Write-Host "$($i + 1)) $($WmiTitles[$i])"
    }

    $choice = Read-Host "`nВведіть номер варіанту"

    [int]$iChoice = 0
    if ([int]::TryParse($choice, [ref]$iChoice) -and $iChoice -ge 1 -and $iChoice -le $WmiTitles.Count) {
        $selectedFilter = $WmiFilters[$iChoice - 1]
        Write-Host "Вибраний фільтр: $selectedFilter" -ForegroundColor Cyan
        return $selectedFilter
    } else {
        Write-Host "Невірний вибір." -ForegroundColor Red
        return $null
    }
}
```

Рисунок 3.2.3.3 – Лістинг коду функції “WmiFiterNoteList”



```
select * from Win32_OperatingSystem where Version like "10.0.2%" and ProductType="1"
select * from Win32_OperatingSystem where Version like "10.0.26100%" and ProductType="1"
select * from Win32_OperatingSystem where Version like "10.0.22631%" and ProductType="1"
select * from Win32_OperatingSystem where Version like "10.0.22621%" and ProductType="1"
select * from Win32_OperatingSystem where Version like "10.0.22000%" and ProductType="1"
```

Рисунок 3.2.3.4 – Вміст файлу з WMI-запитами



```
Windows 11 (All Versions)
Windows 11 (24H2)
Windows 11 (23H2)
Windows 11 (22H2)
Windows 11 (21H2)
```

Рисунок 3.2.3.5 – Формат їх виводу користувачу

Прив’язка фільтру до групової політики розроблено механізм, який відображає на екрані усіх об’єктів типу “msWMI-Som” у Acvite Directory, надаючи корис-

тувачу змогу обрати потрібний варіант. Аналогічна підхід застосовується до вибору GPO. Отримані параметри передаються до функції “Set-GPOWmiFilter”, що відповідає за встановлення зв’язку між фільтром та політикою.

Щоб провести редагування WMI-фільтру, спочатку обирається об’єкт для модифікування, за допомогою “AllWmiFilters” далі у змінні записуються новий WMI-запит та коментар. Ці дані використовує “Set-WmiFilter”, щоб оновити відомості фільтру.

Функція Remove-WmiFilter забезпечує видалення обраного WMI-фільтра з Active Directory. Вона дає змогу адміністратору прибрати непотрібні об’єкти, підтримуючи порядок у структурі фільтрів.

3.3 Розробка сценаріїв моніторингу та підтримки GPO

3.3.1 Налаштування планових задач для автоматизації менеджменту GPO

З метою поліпшення рівня моніторингу різноманітних станів у домені, доцільно використати інструменти, що дозволяють налаштовувати періодичне виконання задач. Одним з подібних рішень є Task Scheduler – планувальник завдань у операційній системі Windows, котрий забезпечує автоматичний запуск скриптів чи завдань у визначений час.

Даний метод дозволяє зменшити залежність від зовнішнього впливу адміністратора на систему, що в теорії позитивно відобразить на її стабільності та безпеці, оскільки мінімізується ризик людського фактору.

Бонусом автоматизоване планування дозволяє формувати лог-файли з результатами перевірок на основі яких адміністратор базуватиме свої подальші дії.

Створимо планове завдання, що складатиметься з трьох взаємопов’язаних частин (Додаток 3). Перша й основна функціональна частина стосовно взаємодією з груповими політиками “Backup_TS” (рис. 3.3.1.1), відповідає за процес резервного копіювання. Визначивши базову директорію для збереження за її відсутності створює далі збирає інформацію про усі наявні GPO на домені й резервує їх.

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				77
Змн.	Арк.	№ докум.	Підпис	Дата		

Другий скрипт “ScriptsLog” (рис. 3.3.1.2) призначений для журналювання. Його ключова функція Write-Log, яка записує всі події в лог-файли з позначками часу часу, дозволяючи відстежувати виконання скрипта: успішний запуск, завершення, та можливе виникнення помилок. Крім того, скрипт запускає допоміжний скрипт, у нашому випадку посилається на попередній код.

```
$BackupBasePath = "C:\Scripts\GPO_Backups"
$Date = Get-Date -Format "yyyyMMdd_HH:mm:ss"
$LogFile = "$BackupBasePath\BackupLog.txt"

if (-not (Test-Path $BackupBasePath)) {
    New-Item -ItemType Directory -Path $BackupBasePath | Out-Null
}

$GPOList = Get-GPO -All | Select-Object -ExpandProperty DisplayName

foreach ($GPOName in $GPOList) {
    $BackupPath = "$BackupBasePath\$GPOName\$Date"

    if (-not (Test-Path $BackupPath)) {
        New-Item -ItemType Directory -Path $BackupPath | Out-Null
    }
    Backup-GPO -Name $GPOName -Path $BackupPath -Domain "makaryk.net"
}
```

Рисунок 3.3.1.1 – Лістинг коду “Backup_TS”

```
$logFolder = "C:\Scripts\Logs"
$logFile = Join-Path -Path $logFolder -ChildPath "BackupLog.txt"

function Write-Log {
    param (
        [string]$message
    )
    $timestamp = Get-Date -Format "yyyy-MM-dd HH:mm:ss"
    $logMessage = "$timestamp - $message"
    Add-Content -Path $logFile -Value $logMessage
}

Write-Log "start"
try {
    $ScriptPath = "C:\Scripts\Backup_TS.ps1"
    Start-Process -FilePath "powershell.exe" -ArgumentList "-NoProfile -ExecutionPolicy Bypass -File `"$ScriptPath`"" -Wait
    Write-Log "Run'$ScriptPath'."
} catch { Write-Log "Error: $_" }

Write-Log "end"
```

Рисунок 3.3.1.2 – Лістинг коду “ScriptsLog”

Третій скрипт “ScheduledTask_Backup” (рис. 3.3.1.3) відповідає за автоматизацію регулярного виконання процесу. Він створює заплановане завдання через командлет Register-ScheduledTask, яке запускає скрипт із резервним копіюванням кожні 4 години протягом доби. Завдання налаштоване на виконання від імені системного облікового запису з найвищим рівнем привілеїв, гарантуючи безперервну роботу навіть без втручання адміністратора.

```
$scriptPath = "C:\Scripts\ScriptLog.ps1"

if (Test-Path $scriptPath) {
    $action = New-ScheduledTaskAction -Execute "powershell.exe" -Argument "-NoProfile -ExecutionPolicy Bypass -File '$scriptPath'"
    $trigger = New-ScheduledTaskTrigger -Once -At (Get-Date) -RepetitionInterval (New-TimeSpan -Hours 4) -RepetitionDuration (New-TimeSpan -Days 1)
    $taskName = "BackupAndLogs"
    try {
        Register-ScheduledTask -TaskName $taskName -Action $action -Trigger $trigger -User "SYSTEM" -RunLevel Highest
        Write-Host "Задача '$taskName' створена та виконуватиметься кожні 4 години." -ForegroundColor Green
    } catch { Write-Host "Не вдалося створити задачу: $_" -ForegroundColor Red }
} else { Write-Host "Файл скрипта не знайдено за шляхом: $scriptPath" -ForegroundColor Red }
```

Рисунок 3.3.1.3 – Лістинг коду “ScheduledTask_Backup”

У домені наявні ще пара запланованих завдань, що відповідаються за діагностування групових політик, та їх загальну оптимізацію. Наприклад “Check-UnlinkedGPO” (рис. 3.3.1.4) активується кожного дня о шостій годині ранку й виконує перевірку всіх GPO на прив’язаність до будь-якої організаційної одиниці чи хоча б домену (Додаток I).

Після знаходження неприв’язаних політик скрипт переходить до логування де у текстовий файл записується лише імена знайдених відірваних політик.

```
function Get-UnlinkedGPOs {
    $AllGPOs = Get-GPO -All
    $LinkedGPOs = @()
    $Targets = Get-AllGpoTargets

    foreach ($target in $Targets) {
        try {
            $Inheritance = Get-GPInheritance -Target $target.DistinguishedName
            foreach ($Link in $Inheritance.GpoLinks) {
                if ($Link.DisplayName -and $LinkedGPOs -notcontains $Link.DisplayName) {
                    $LinkedGPOs += $Link.DisplayName
                }
            }
        } catch { Write-Warning "Помилка при обробці $($target.DistinguishedName): $_" }
    }
    $UnlinkedGPOs = $AllGPOs | Where-Object { $LinkedGPOs -notcontains $_.DisplayName }
    return $UnlinkedGPOs
}

$LogFolder = "C:\Scripts\Logs"
$LogPath = "$LogFolder\UnlinkedGPOs.txt"
if (-not (Test-Path $LogFolder)) { New-Item -ItemType Directory -Path $LogFolder | Out-Null }

$UnlinkedGPOs = Get-UnlinkedGPOs
$UnlinkedGPOs | Select-Object DisplayName | Out-File -FilePath $LogPath -Encoding UTF8
```

Рисунок 3.3.1.4 – Лістинг коду “Check-UnlinkedGPO”

Наступна задача з використанням Task Scheduler (Додаток II) відповідає за пошук дублікатів реєстрових налаштувань у GPO домену (рис. 3.3.1.5). В основній частині ініціалізується змінна у котру записуватимуться усі унікальні комбінації ключа реєстру та його значення з кожної політики. Для кожного параметра створюється унікальний ключ, який використовується для агрегації GPO, у яких цей параметр зустрічається. Тому, якщо декілька GPO змінюють один і той самий параметр, ця інформація накопичується.

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				79
Змн.	Арк.	№ докум.	Підпис	Дата		

Далі використовується фільтрація, вибираючи записи, де значення містить більше одного елементу, що означатиме присутність однакового налаштування в кількох політиках. У разі відсутності співпадінь, у лог файл за допомогою відповідної функції “Log” записується повідомлення, в іншому випадку в текстовий файл запишеться шлях ключа реєстру його назва, значення та список політик з повторенням.

```
try {
    $RegistryMap = @{}
    $AllGPOs = Get-GPO -All

    foreach ($GPO in $AllGPOs) {
        Log "Processing GPO: $($GPO.DisplayName)"
        try {
            $regValues = Get-GPAllRegistryValues -Name $GPO.DisplayName
            foreach ($val in $regValues) {
                $key = "$($val.KeyPath)| $($val.ValueName)"
                if (-not $RegistryMap.ContainsKey($key)) {
                    $RegistryMap[$key] = @()
                }
                $RegistryMap[$key] += $GPO.DisplayName
            }
        } catch { Log "Error processing GPO '$($GPO.DisplayName)': $_" }

        $duplicates = $RegistryMap.GetEnumerator() | Where-Object { $_.Value.Count -gt 1 }

        if ($duplicates.Count -eq 0) {
            Log "No duplicate registry entries found."
        } else {
            foreach ($entry in $duplicates) {
                $keySplit = $entry.Key -split '|'
                $keyPath = $keySplit[0]
                $valueName = $keySplit[1]
                $gpoList = $entry.Value -join ', '
                Log "Duplicate found: $keyPath -> $valueName | GPOs: $gpoList"
            }
        }
    }
} catch { Log "Unexpected error: $_" }
```

Рисунок 3.3.1.5 – Лістинг коду “CreateTask_CompareGPO”

3.3.2 Розробка скриптів для генерація звітів про застосування політик

Важливим аспектом адміністрування корпоративного середовища є забезпечення постійного контролю за застосуванням та налаштуваннями групових політик. За відсутності належного журналювання у певний момент адміністратору стане важко відстежити зміни у їх структурі чи виявити причини конфліктів та збоїв коли вони стануться по вині некоректно налаштованих групових політик. Саме з цією метою доречно час від часу зберігати звіти GPO, забезпечивши їм зручну структуру доступу за зрозумілим найменуванням, щоб зменшити час на пошуки у разі виникнення потреби у оперативному втручанні.

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				80
Змн.	Арк.	№ докум.	Підпис	Дата		

Для цього було реалізовано PowerShell-скрипт (Додаток Ж), який відповідає за створення звітів та надає базові інструменти для аналізу ефективності застосування у вигляді журналів наслідування групових політик, порівняння різних версій та RSOP-звітів.

Розглянемо блок-схему (рис. 3.3.2.1), яка покаже нам принцип роботи даного скрипта. Адміністратору дається кілька дій на вибір, а саме: генерація звітів з усіх групових політик, звіт для певної GPO, створити RSOP звіт для користувача і комп'ютера, звіт про наслідування політик організаційної одиниці та порівняння двох GPO.

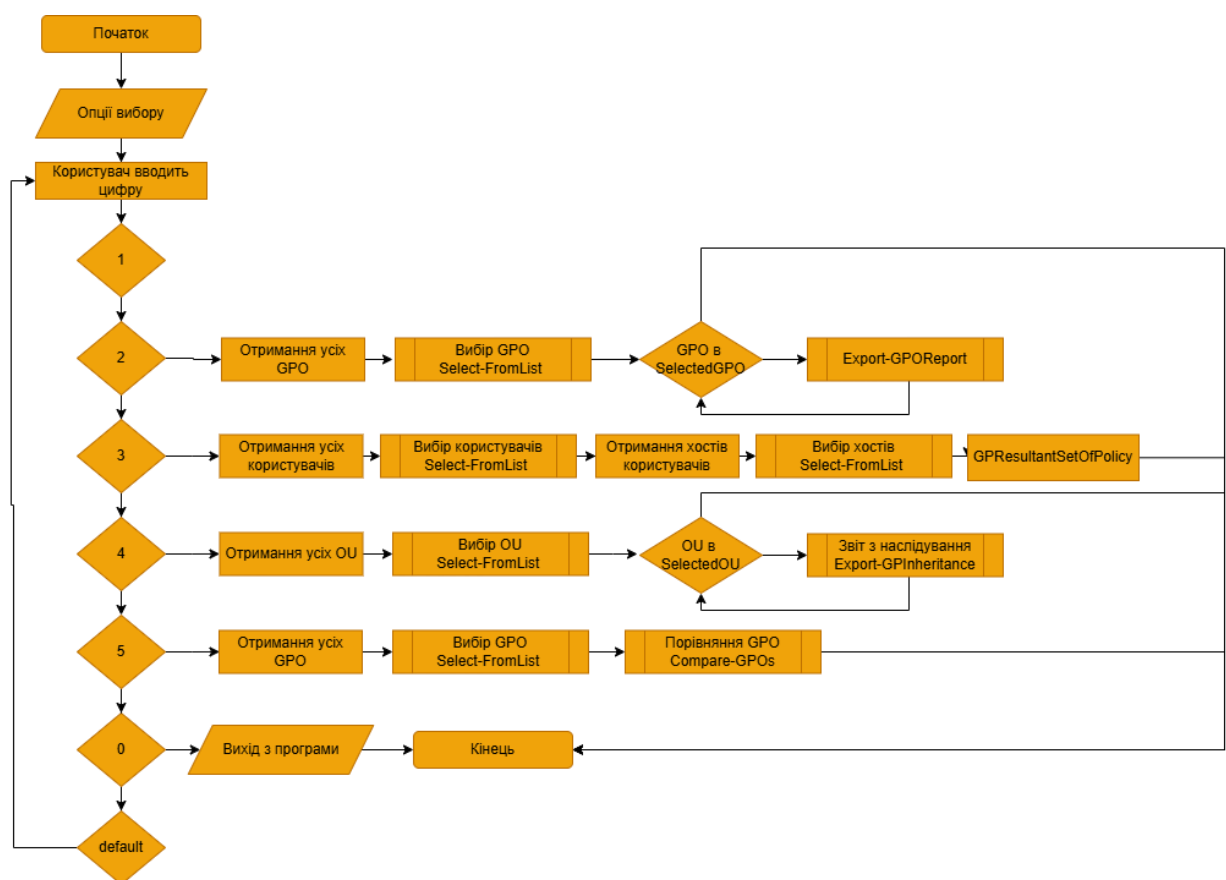


Рисунок 3.3.2.1 – Блок-Схема “Report”

Почнемо з основної функції для генерації звітів з одної чи кількох групових політик “Export-GPOReport” (рис. 3.3.2.2). Вона викликає допоміжну функцію “GpoGenerateReport”, яка створює для кожної політики окрему папку з підкаталогами html та xml, у які зберігаються відповідні звіти. Кожен файл доповнюється міткою часу, що забезпечує збереження історії змін політик.

```
function Export-GPOReport {
    param (
        [string]$GPOName = '',
        [string]$ReportPath = "C:\GPO_Report"
    )

    if (-not (Test-Path $ReportPath)) { New-Item -ItemType Directory -Force -Path $ReportPath | Out-Null }
    $GPOs = if ($GPOName) {
        Get-GPO -Name $GPOName
    } else { Get-GPO -All }

    foreach ($GPO in $GPOs) {
        try { GpoGenerateReport -GPOName $GPO.DisplayName -GPOReportPath $ReportPath }
        catch { Write-Host "Помилка створення звіту для $($GPO.DisplayName): $_" -ForegroundColor Red }
    }
}
```

Рисунок 3.3.2.2 – Лістинг коду функції “Export-GPOReport”

```
function GpoGenerateReport {
    [CmdletBinding()]
    param (
        [string]$GPOName,
        [string]$GPOReportPath = "C:\GPO_Report"
    )

    $Timestamp = Get-Date -Format "yyyy-MM-dd_HH:mm:ss"
    $ReportFolder = Join-Path $GPOReportPath $GPOName
    $formats = @("html", "xml")
    foreach ($format in $formats) {
        $FolderPath = Join-Path $ReportFolder $format
        if (-not (Test-Path $FolderPath)) { New-Item -ItemType Directory -Path $FolderPath -Force | Out-Null }
    }
    try {
        $htmlPath = Join-Path -Path (Join-Path $ReportFolder "html") -ChildPath "$GPOName_$Timestamp.html"
        $xmlPath = Join-Path -Path (Join-Path $ReportFolder "xml") -ChildPath "$GPOName_$Timestamp.xml"
        Get-GPOReport -Name $GPOName -ReportType Html -Path $htmlPath
        Get-GPOReport -Name $GPOName -ReportType Xml -Path $xmlPath
        Write-Host "Створений звіт для $GPOName" -ForegroundColor Green
    } catch { Write-Host "Помилка генерації звіту $GPOName $_" -ForegroundColor Red }
}
```

Рисунок 3.3.2.3 – Лістинг коду функції “GpoGenerateReport”

Для запису усіх наслідуваних групових політик на рівні організаційних одиниць використовується “Export-GPInheritance” (рис. 3.3.2.4). Отримавши список політик за допомогою командлети “Get-GPInheritance” зберігає звіт у відповідній папці.

```
function Export-GPInheritance {
    param (
        [string]$TargetOU
    )
    try {
        $Inheritance = Get-GPInheritance -Target $TargetOU
        $InheritanceFilePath = "C:\GPO_Report\GP_Inheritance_$(Get-Date -Format 'yyyy-MM-dd_HH:mm:ss').txt"
        $Inheritance | Out-File -FilePath $InheritanceFilePath
        Write-Host "Звіт про наслідування політик збережено в $InheritanceFilePath" -ForegroundColor Green
    } catch { Write-Host "Помилка отримання наслідування: $_" -ForegroundColor Red }
}
```

Рисунок 3.3.2.4 – Лістинг коду функції “Export-GPInheritance”

Як тільки функція порівняння двох звітів однієї політики “Compare-GPOs” (рис. 3.3.2.5), отримує той самий об’єкт GPO, то починає шукати збережені файли формату XML у відповідній теці. Знайшовши виводить їх загальний список надавши користувачу довільно одрати два файли, які використовуються у командлеті “Compare-Object”.

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				82
Змн.	Арк.	№ докум.	Підпис	Дата		

```

function Compare-GPOs {
    param (
        [Parameter(Mandatory=$true)]
        [Object[]]$UsedGPO
    )
    $xmlFolder = Join-Path -Path "C:\GPO_Report\$($UsedGPO.DisplayName)" -ChildPath "xml"
    if (-not (Test-Path $xmlFolder)) {
        Write-Host "Папка не знайдена: $xmlFolder" -ForegroundColor Red
        return
    }
    $files = Get-ChildItem -Path $xmlFolder -Filter *.xml
    $SelectedFiles = Select-FromList -Items $files -PropertyToShow "Name"

    if ($SelectedFiles.Count -eq 2) {
        $Report = @(
            $SelectedFiles[0].FullName
            $SelectedFiles[1].FullName
        )
    } else { Write-Host "Потрібно вибрати рівно 2 файли!" -ForegroundColor Red }

    try {
        $Comparison = Compare-Object -ReferenceObject $Report[0] -DifferenceObject $Report[1]
        $ComparisonFilePath = "C:\GPO_Report\Comparison_$(Get-Date -Format 'yyyy-MM-dd_HH:mm:ss').txt"
        $Comparison | Out-File -FilePath $ComparisonFilePath
        Write-Host "Результат порівняння збережено в $ComparisonFilePath" -ForegroundColor Green
    } catch { Write-Host "Помилка при порівнянні: $_" -ForegroundColor Red }
}

```

Рисунок 3.3.2.5 – Лістинг коду функції “Compare-GPOs”

Функція “GPResultantSetOfPolicy” (рис. 3.3.2.6) створює RSOP-звіт, який відображає реальне застосування політик до конкретного користувача чи комп’ютера. Скрипт дозволяє обрати одного або кількох облікових записів та пристроїв домену зі списку, після чого для кожної пари генерується окремий звіт у форматі HTML.

```

function GPResultantSetOfPolicy {
    param (
        [Parameter(Mandatory=$true)]
        [Object[]]$UsedUser,
        [Parameter(Mandatory=$true)]
        [Object[]]$UsedComp
    )
    foreach ($OU in $UsedUser) {
        foreach ($Comp in $UsedComp) {
            try {
                $Path = "C:\GPO_Report\RSOP_${OU}_${Comp}_$(Get-Date -Format 'yyyy-MM-dd_HH:mm:ss').html"
                Get-GPResultantSetOfPolicy -User $OU -Computer $Comp -ReportType HTML -Path $Path
                Write-Host "Звіт збережено: $Path" -ForegroundColor Green
            } catch { Write-Host "Помилка RSOP для $OU@$Comp $_" -ForegroundColor Red }
        }
    }
}

```

Рисунок 3.3.2.6 – Лістинг коду функції “GPResultantSetOfPolicy”

У підсумку, даний скрипт виходить зручним та досить універсальним інструментом для спрощення й мінімізації витраченого часу на рутинні завдання пов’язані з відстеженням змін у групових політиках та їх взаємодії з іншими об’єктами Active Directory.

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				
Змн.	Арк.	№ докум.	Підпис	Дата		83

Висновки до третього розділу

У ході цього розділу була проведена розробка PowerShell-скриптів групових політик та методів їх автоматизованого адміністрування. Забезпечено масове внесення змін до GPO за допомогою редагування скриптів першоджерел та використання командлета “Set-GPRegistryValue”. Додано скрипти, які відповідають за прив’язку політик до OU, зміни рівня доступу для певних користувачів. Для спрощення роботи створено користувацькі PowerShell-модулі, які включають найпоширеніші функції. Забезпечено резервування, журналювання, діагностику через планові завдання і видалення GPO.

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				
Змн.	Арк.	№ докум.	Підпис	Дата		84

РОЗДІЛ 4. ТЕСТУВАННЯ ТА АНАЛІЗ ЕФЕКТИВНОСТІ АВТОМАТИЗАЦІЇ

4.1 Тестування реалізованої системи

4.1.1 Перевірка коректності роботи запроваджених GPO

Тестування є невід’ємною частиною перед повним розгортанням засобів адміністрування, оскільки дозволяє на практиці перевірити роботу системи. У разі помилок чи відхилень від запланованого функціоналу тестування дає змогу оперативно виправити проблемні фрагменти. Тож у цьому пункті ми побачимо роботу застосованих у домені групових політик.

Почнемо з політики жорсткого квотування з назвою “Disk Quotas Policy” перевіriamo її застосування на домені, отримавши значення налаштованих реєстрів та їх значень (рис. 4.1.1.2). Ця політика впливає на сам пристрій, а не користувачів, тому ми можемо зайти на будь-який обліковий запис організаційних одиниць. Підключаємося до робочої станції WS-Win10, після чого відкриємо каталог і у властивостях локального диска С перейдемо до вкладки «Квоти» де ввівши дані адміністратора, зможемо побачити (рис. 4.1.1.1) застосоване жостке квотування та його параметри.

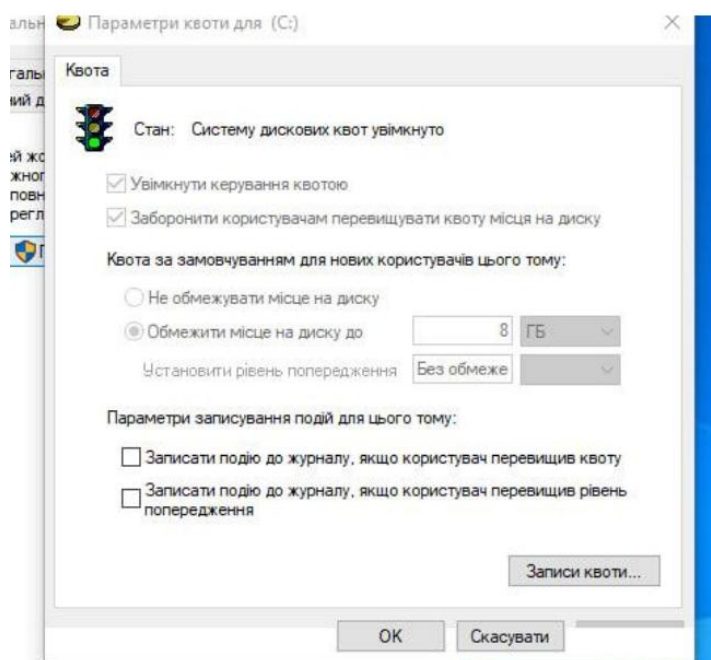


Рисунок 4.1.1.1 – Параметри квоти на диску С на WS-Win10

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				
Змн.	Арк.	№ докум.	Підпис	Дата		85

```
Get-GPAllRegistryValues -Name "Disk Quotas Policy" | Select-Object FullKeyPath, Value, ValueName
```

FullKeyPath	Value	ValueName
HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows NT\DiskQuota	1	ApplyToRemovableMedia
HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows NT\DiskQuota	1	LogEventOverThreshold
HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows NT\DiskQuota	1	LogEventOverLimit
HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows NT\DiskQuota	8	Limit
HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows NT\DiskQuota	3	LimitUnits
HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows NT\DiskQuota	1	Enable
HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows NT\DiskQuota	1	Enforce

Рисунок 4.1.1.2 – Параметри реєстрів на “Disk Quotas Policy”

"Screen Lock Policy" автоматично активує заставку комп'ютера після 10 хвилин бездіяльності користувача й потребує введення паролю для авторизації. Щоб перевірити наявність політики на хості зайдемо з акаунту адміністратора, бо перегляд змінених параметрів реєстру доступний лише для користувачів організаційної одиниці “Admins-1”.

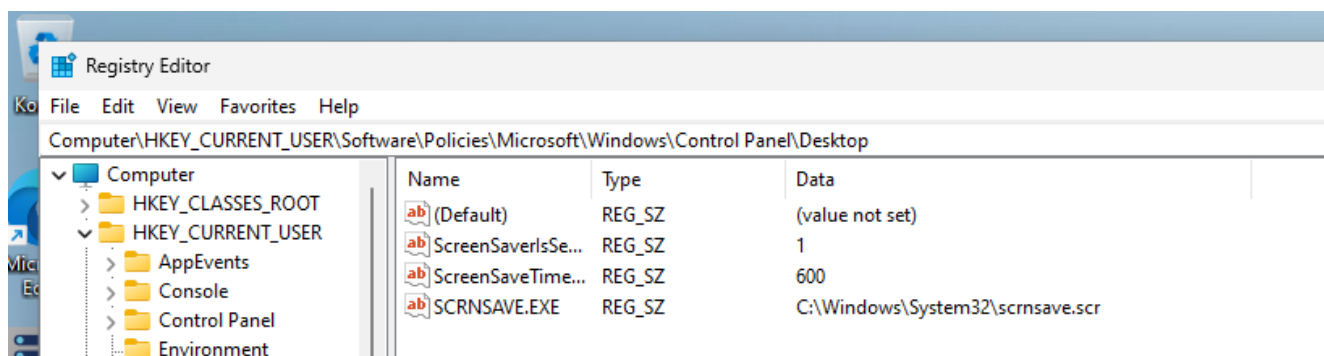


Рисунок 4.1.1.3 – Параметри реєстрів на “Disk Quotas Policy” на WS-Win11-32H2

```
Get-GPAllRegistryValues -Name "Screen Lock Policy" | Select-Object FullKeyPath, Value, ValueName
```

FullKeyPath	Value	ValueName
HKEY_CURRENT_USER\Software\Policies\Microsoft\Windows\Control Panel\Desktop	C:\Windows\System32\scrnsave.scr	SCRNSAVE.EXE
HKEY_CURRENT_USER\Software\Policies\Microsoft\Windows\Control Panel\Desktop	600	ScreenSaveTimeout
HKEY_CURRENT_USER\Software\Policies\Microsoft\Windows\Control Panel\Desktop	1	ScreenSaverIsSecure

Рисунок 4.1.1.4 – Параметри реєстрів на “Disk Quotas Policy”

Щоб протестувати GPO "Restrict CMD and PowerShell Access", котра обмежує використання середовища командного рядка та PowerShell у системі. Заборонивши доступ усіх користувачів до cmd (рис. 4.1.1.6) й забороняє виконання будь-який скриптів (рис. 4.1.1.7). Використаємо обліковий запис “Sampro-dep-1”. Для цього просто відкриємо командний рядок і додатково створимо пустий скрипт спробувавши його запустити.

```
#Set-GPLink -Name "Screen Lock Policy" -Target "dc=makaryk,dc=net" -Enforced Yes
```

FullKeyPath	Value	ValueName
HKEY_CURRENT_USER\Software\Policies\Microsoft\PowerShell	Restricted	ExecutionPolicy
HKEY_CURRENT_USER\Software\Policies\Microsoft\PowerShellCore	Restricted	ExecutionPolicy
HKEY_CURRENT_USER\Software\Policies\Microsoft\Windows\System	2	DisableCMD

Рисунок 4.1.1.5 – Параметри реєстрів на “Restrict CMD and PowerShell Access”

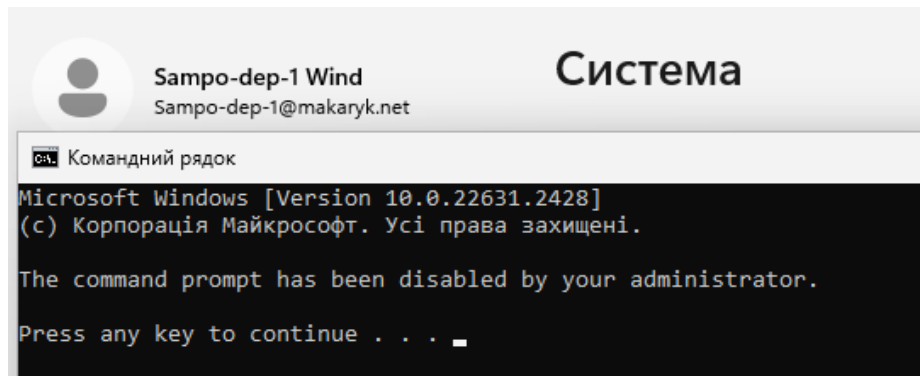


Рисунок 4.1.1.6 – Заборона використовувати командний рядок

```
PS C:\Users\Sampo-dep-1> C:\Users\Sampo-dep-1\teest.ps1
File C:\Users\Sampo-dep-1\teest.ps1 cannot be loaded because running scripts is disabled on this system.
For more information, see about_Execution_Policies at https://go.microsoft.com/fwlink/?LinkID=135170.
+ CategoryInfo          : SecurityError (:) [], ParentContainsErrorRecordException
+ FullyQualifiedErrorId : UnauthorizedAccess
```

Рисунок 4.1.1.7 – Заборона запускати PowerShell-скрипти

“Additional Installation Policies” працює як заборона оновлення програмного забезпечення в умовах, коли користувач не має повного адміністративного доступу та вимкнення можливості сканування змін під час встановлення ПЗ. Додатково встановлено логуювання подій і заборона на інсталяцію патчів (рис. 4.4.1.9). Тому для тестування зайдемо у Windows Installer.

```
#Set-GPLink -Name "Screen Lock Policy" -Target "dc=makaryk,dc=net" -Enforced Yes
```

FullKeyPath	Value	ValueName
HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows\Installer	1	DisableLUAPatching
HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows\Installer	1	DisableRollback
HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows\Installer	1	DisablePatch
HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows\Installer	iearmxv	Logging

Рисунок 4.1.1.8 – Параметри реєстрів на “Additional Installation Policies”

View configured update policies

Wondering why you're seeing 'Some settings are managed by your organization'?

This text is typically displayed on Windows Update after installation and delivery policies are configured.

Examples include:

- Your organization has set some policies to manage updates
- You have opted in for the Windows Insider Program

Policies set on your device

Disable automatic updates
Source: Administrator
Type: Group Policy

Get updates for other Microsoft products
Source: Administrator
Type: Group Policy

Exclude drivers from Windows quality updates
Source: Administrator
Type: Group Policy

Set Automatic Update options
Source: Administrator
Type: Group Policy

Рисунок 4.1.1.9 – Тестування групової політики “Additional Installation Policies”

Як стало відомо раніше, “Scheduled Scan for Windows Defender” використовують для налаштування розкладу планового запуску сканування Windows Defender, тому перевіримо використавши командлет “Get-MpPreference” (рис. 4.1.1.10).

```
PS C:\Users\Sampo-dep-1> Get-MpPreference | Select-Object ScanScheduleDay, ScanScheduleTime

ScanScheduleDay ScanScheduleTime
-----
1 04:00:00
```

Рисунок 4.1.1.10 – Вивід командлету “Get-MpPreference”

```
#Set-GPLink -Name "Screen Lock Policy" -Target "dc=makaryk,dc=net" -Enforced Yes

FullKeyPath Value ValueName
-----
HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows Defender\Scan 1 ScheduleDay
HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows Defender\Scan 240 ScheduleTime
```

Рисунок 4.1.1.11 – Параметри реєстрів на “Scheduled Scan for Windows Defender”

У системі налаштовані одна GPO “Login Message Policy”, яка виводить повідомлення, що відображається користувачам при спробі входу на пристрій.

```
#Set-GPLink -Name "Screen Lock Policy" -Target "dc=makaryk,dc=net" -Enforced Yes

FullKeyPath Value ValueName
-----
HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Policies\System *Км-кхм* LegalNoticeCaption
HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Policies\System Повідомляємо, завтра (32.18.1517) на сервері відбудуться технічні роботи о -1:59. LegalNoticeText
```

Рис. 4.1.1.12 – Параметри реєстрів на “Login Message Policy”

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				88
Змн.	Арк.	№ докум.	Підпис	Дата		

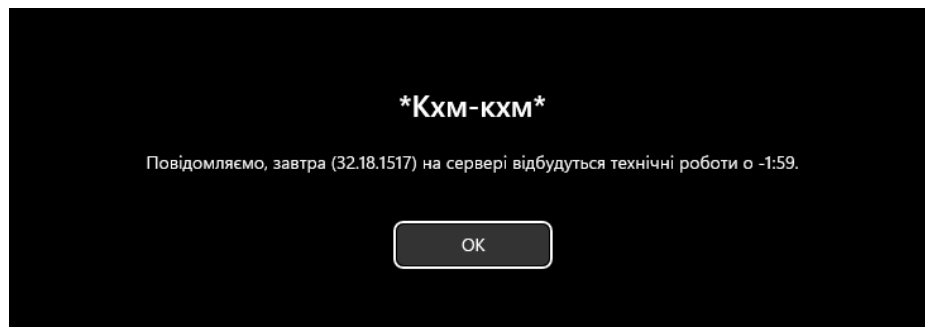


Рисунок 4.1.1.13 – Повідомлення, котре виводиться

Даний об’єкт групової політики запобігає автоматичному перезавантаженню, усуває вплив непотрібного ПЗ в Windows Update шляхом вимкнення “Рекомендованого програмного забезпечення” та зменшує незручності для користувачів під час завершення роботи комп’ютера за допомогою видалення опції встановлення оновлень.

FullKeyPath	Value	ValueName
HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\windows\windowsUpdate\AU	1	NoAutoRebootWithLoggedOnUsers
HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\windows\windowsUpdate\AU	0	EnableFeaturedSoftware
HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\windows\windowsUpdate\AU	1	NoAUAstDefaultShutdownOption

Рисунок 4.1.1.14 – Параметри реєстрів на “Disable Forced System Restarts”

GPO “Audit Policy Configuration” налаштовує аудит безпеки Windows, увімкнувши детальне відстеження ключових подій у системі. Вона реєструє дії, пов’язані з управлінням обліковими записами, доступом до директорій, подіями входу/виходу, використанням змінних носіїв та змінами в політиках. Щоб протестувати її зайдемо до “Event Viewer” (рис. 4.1.1.16) на головному доменному контролері.

FullKeyPath	Value	ValueName
HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows\EventLog\Account Management	1	AuditComputerAccountManagement
HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows\EventLog\Account Management	3	AuditOtherAccountManagementEvents
HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows\EventLog\Account Management	3	AuditSecurityGroupManagement
HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows\EventLog\Account Management	3	AuditUserAccountManagement
HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows\EventLog\Detailed Tracking	1	AuditPlugAndPlayEvents
HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows\EventLog\Detailed Tracking	1	AuditProcessCreation
HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows\EventLog\DS Access	3	AuditDirectoryServiceAccess
HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows\EventLog\DS Access	3	AuditDirectoryServiceChanges
HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows\EventLog\Logon/Logoff	3	AuditAccountLockout
HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows\EventLog\Logon/Logoff	1	AuditGroupMembership
HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows\EventLog\Logon/Logoff	1	AuditLogoff
HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows\EventLog\Logon/Logoff	3	AuditLogon
HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows\EventLog\Logon/Logoff	1	AuditSpecialLogon
HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows\EventLog\Object Access	3	AuditRemovableStorage
HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows\EventLog\Policy Change	3	AuditAuditPolicyChange
HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows\EventLog\Policy Change	1	AuditAuthenticationPolicyChange
HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows\EventLog\Policy Change	1	AuditAuthorizationPolicyChange
HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows\EventLog\Privilege Use	3	AuditSensitivePrivilegeUse
HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows\EventLog\System	3	AuditIPsecDriver
HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows\EventLog\System	3	AuditOtherSystemEvents
HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows\EventLog\System	1	AuditSecurityStateChange
HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows\EventLog\System	3	AuditSecuritySystemExtension
HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows\EventLog\System	3	AuditSystemIntegrity

Рисунок 4.1.1.15 – Параметри реєстрів на “Audit Policy Configuration”

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				89
Змн.	Арк.	№ докум.	Підпис	Дата		

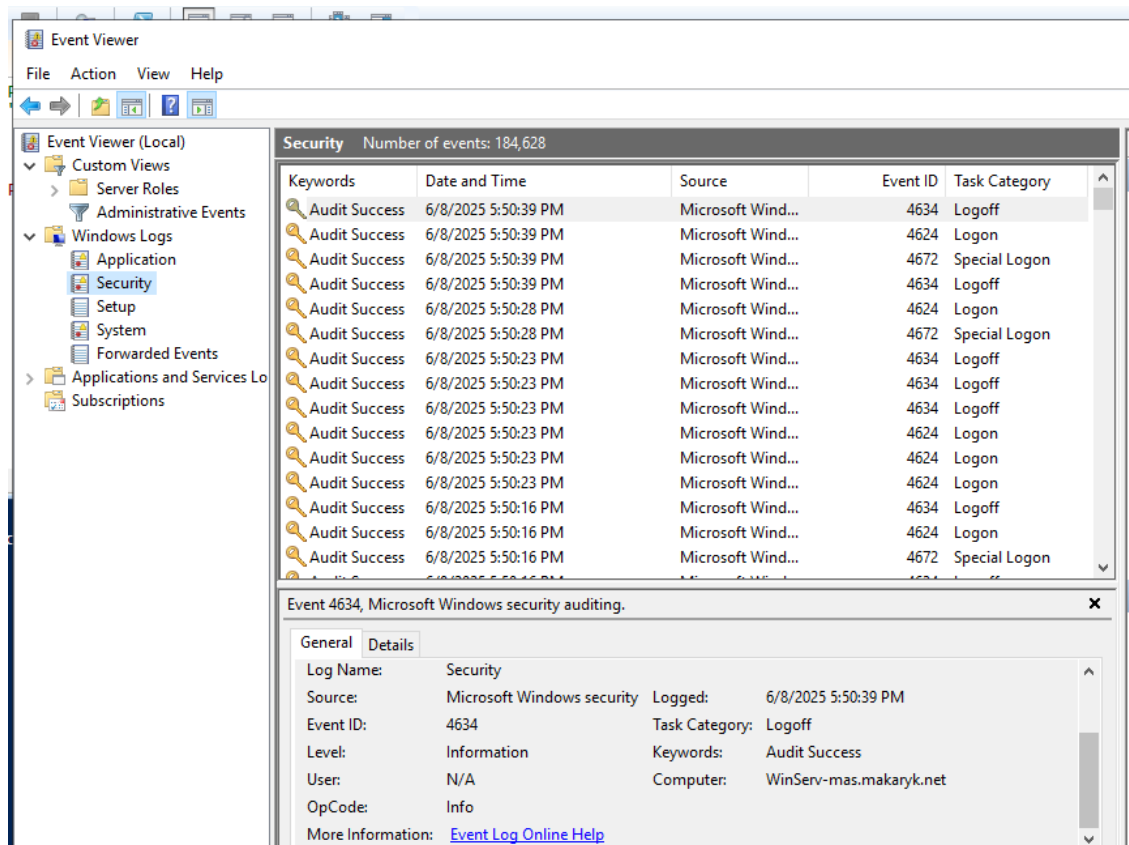


Рисунок 4.1.1.16 – Моніторинг подій у “Audit Policy Configuration”

Дана групова політика “Deny removable disk” намагається забезпечити захист від випадкового занесення у систему шкідливого програмного забезпечення через USB-накопичувач. Вимкнення USB-накопичувача, читання, запуску та редагування записаних у нього файлів перевіряється підключенням флешки до віртуальної машини. Спочатку відбувся перегляд незастосованої політики на контролері (рис. 4.1.1.17), а пізніше другим (рис. 4.1.1.18) зображено активований захист

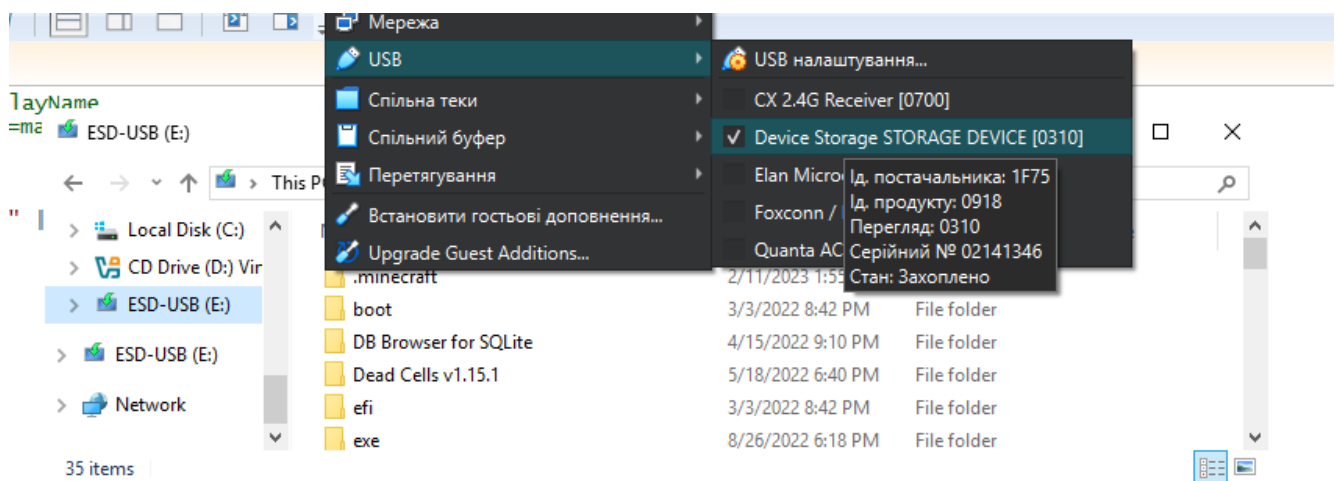


Рисунок 4.1.1.17 – Зчитування USB-накописувача на контролері

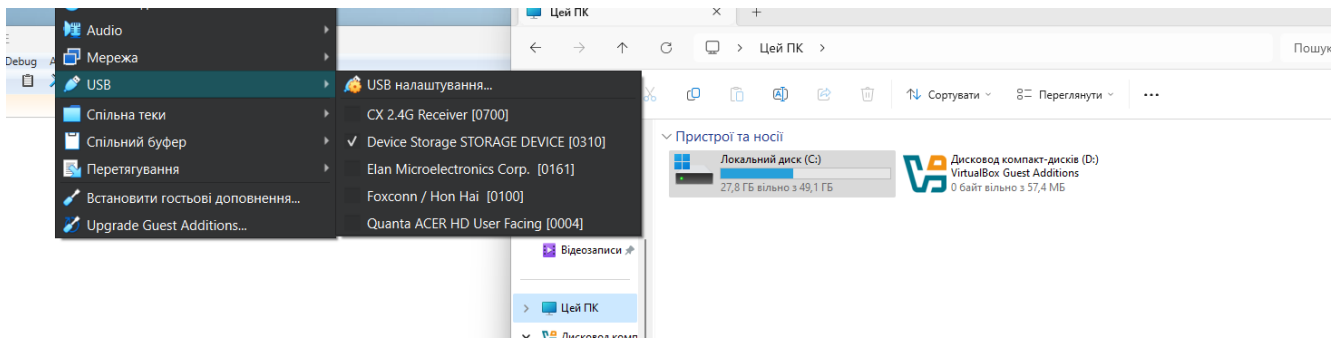


Рисунок 4.1.1.18 – Невдала спроба вимкнути USB-накопичувач на хості

FullKeyPath	Value	ValueName
HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\USBSTOR	4	Start
HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows\RemovableStorageDevices	1	Deny_Execute
HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows\RemovableStorageDevices	1	Deny_Read
HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows\RemovableStorageDevices	1	Deny_Write

Рисунок 4.1.1.19 – Параметри реєстрів на “Deny removable disk”

“Restrict Software Installation” - політика обмежує користувачам можливість встановлювати програмне забезпечення на їхніх комп'ютерах. Встановлення значення DisableUserInstalls на 1 вимикає можливість для користувачів виконувати інсталяцію програм без адміністративних прав. Тому спробуємо встановити довільний файл (рис. 4.1.1.21).

FullKeyPath	Value	ValueName
HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows\Installer	1	DisableUserInstalls

Рисунок 4.1.1.20 – Параметри реєстрів на “Restrict Software Installation”



Рисунок 4.1.1.21 – Запитування прав адміністратора

Як впливає з назви “Restrict registry editing tools and control panel” – обмежує доступ користувачів до панелі керування та редактору реєстрів. Що може бути корисно для запобігання внесенню небажаних змін до системних налаштувань. Отже, перевіримо запустивши обліковий запис користувача на якому встановлено цю заборону (рис. 4.1.1.22).

FullKeyPath	Value	ValueName
HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Policies\Explorer	Restricted	NoControlPanel
HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Policies\System	2	disableregistrytools

Рисунок 4.1.1.21 – Параметри реєстрів на “Restrict registry editing tools and control panel”

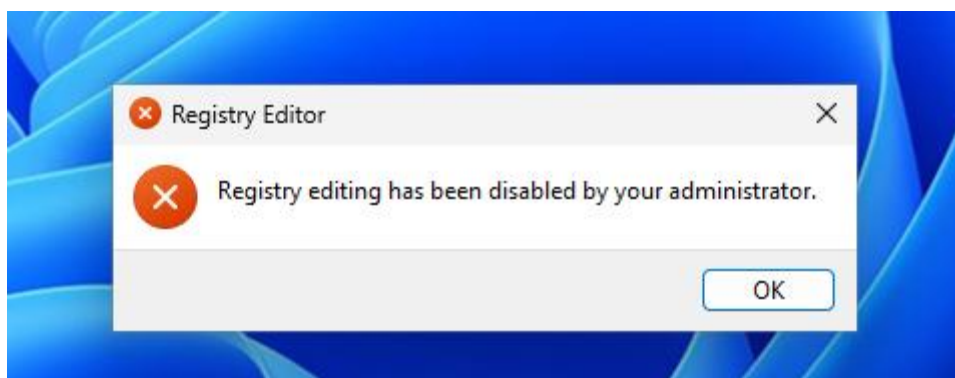


Рисунок 4.1.1.22 – Заборона запуску редактору реєстрів

4.1.2 Тестування PowerShell-скриптів для адміністрування

Після ознайомлення з принципом роботи групових політики у домені, настав час протестувати основні методи їх адміністрування написані через PowerShell-скрипти. Головна ціль перевірки – дізнатися, чи відповідають скрипти своїй головним задачам, а саме: бути інтуїтивними у використанні, оптимізувати роботу адміністратора за рахунок можливості одночасної зміни у кількох політиках, мати функціонал максимально наближений до GI версії (GPMC) та коректно працювати.

Першим на черзі стане скрипт, що відповідає за менеджмент резервування, видалення та відновлення GPO “Delete-Backup”. На запуску перед користувачем висвічується просте та зрозуміле меню з доступними функціями програми (рис. 4.1.2.1). При виборі резервування виведеться перелік усіх застосованих політик на домені (рис. 4.1.2.2), де ввівши необхідний номер GPO відбудеться резервування (рис. 4.1.2.4) у папку зі шляхом “C:\GPO_Backups\назва_GPO\поточний час”. Для

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				92
Змн.	Арк.	№ докум.	Підпис	Дата		

перевірки коректності вводу застосована функція з користувацького модуля “Input-Index” (рис. 4.1.2.3), що у разі помилки попросить повторити дію вказавши на джерело проблеми.

```
PS C:\Scripts> C:\Scripts\Delete-Backup.ps1
Σ(⌢ °Д °;)⌢--- Меню ---
1. Резервувати GPO
2. Видалити GPO
3. Відновити GPO
4. Вийти
Введіть номер дії: |
```

Рисунок 4.1.2.1 – Виведення головного меню “Delete-Backup”

```
4. Вийти
Введіть номер дії: 1
0. Additional Installation Policies
1. Audit Policy Configuration
2. Default Domain Controllers Policy
3. Default Domain Policy
4. Deny removable disk
5. Disable Forced System Restarts
6. Disk Quotas Policy
7. Login Message Policy
8. Restrict CMD and PowerShell Access
9. Restrict registry editing tools and control panel
10. Restrict Software Installation
11. Scheduled Scan for Windows Defender
12. Screen Lock Policy
Введіть потрібні номери (через кому): 0,1,5,6
```

Рисунок 4.1.2.2 – Виведення застосованих політик на домені

```
11. Scheduled Scan for Windows Defender
12. Screen Lock Policy
Введіть потрібні номери (через кому): 99
Помилка - '99' виходить за межі списку
Введіть потрібні номери (через кому): fdfd
Помилка - 'fdfd' повинен бути числом.
Введіть потрібні номери (через кому): 1.2.3
Помилка - '1.2.3' повинен бути числом.
Введіть потрібні номери (через кому):
```

Рисунок 4.1.2.3 – Перевірка коректності вводу

```
DisplayName      : Additional Installation Policies
GpoId            : 6706adf9-a746-4698-9e84-2acc46335488
Id              : d1bf44ef-0ab7-41bc-a477-1469513f2775
BackupDirectory : C:\GPO_Backups\Additional Installation Policies\20250610_045002
CreationTime    : 6/10/2025 4:50:02 AM
DomainName      : makaryk.net
Comment         :

GPO Additional Installation Policies успішно резервовано у C:\GPO_Backups\Additional Installation Policies\20250610_045002.

GPO Disable Forced System Restarts успішно резервовано у C:\GPO_Backups\Disable Forced System Restarts\20250610_045002.
DisplayName      : Disk Quotas Policy
GpoId            : 1021906e-adb9-4825-a38e-68bdf7cdf9c
Id              : 85d8f781-619a-40a5-84b8-f4400612bd97
BackupDirectory : C:\GPO_Backups\Disk Quotas Policy\20250610_045002
CreationTime    : 6/10/2025 4:50:03 AM
DomainName      : makaryk.net
Comment         :

GPO Disk Quotas Policy успішно резервовано у C:\GPO_Backups\Disk Quotas Policy\20250610_045002.
```

Рисунок 4.1.2.4 – Виведення про резервування першої та останньої GPO переліку

Вибравши другу опцію, а точніше “Видалити GPO”, описані вище кроки для вибору об’єктів групової політики повторяться. Після виконання видалення GPO з домену виведеться повідомлення про успішність виконання (рис. 4.1.2.5). Для перевірки дайсності даного повідомлення скористаємося командлетом “Get-GPO”, щоб вивести усі імена залишившихся GPO (рис. 4.1.2.6).

```
5. Disable Forced System Restarts
6. Disk Quotas Policy
7. Login Message Policy
8. Restrict CMD and PowerShell Access
9. Restrict registry editing tools and control panel
10. Restrict Software Installation
11. Scheduled Scan for Windows Defender
12. Screen Lock Policy
Введіть потрібні номери (через кому): 7,6
GPO 'Login Message Policy' успішно видалено.
GPO 'Disk Quotas Policy' успішно видалено.
Σ(⌂ °Д °;)⌂---- Меню ----
```

Рисунок 4.1.2.5 – Видалення GPO з домену

```
#Get-GPAPIRegistryValues -Name 'Restrict Registry editing tools'
Get-GPO -All | Select-Object -ExpandProperty DisplayName

Default Domain Policy
Screen Lock Policy
Restrict CMD and PowerShell Access
Additional Installation Policies
Default Domain Controllers Policy
Scheduled Scan for Windows Defender
Disable Forced System Restarts
Audit Policy Configuration
Deny removable disk
Restrict Software Installation
Restrict registry editing tools and control panel
```

Рисунок 4.1.2.6 – Вивід усіх застосованих GPO у домені

Останньою опцією є відновлення групової політики (GPO), або так званий відкат до попередньої стабільної версії, що особливо корисно у разі виникнення помилок після внесення змін до налаштувань. Після того, як користувач обирає політику для відновлення, відображається список доступних резервних копій, створених раніше, що дозволяє швидко повернути стабільний стан політики. (рис. 4.1.2.7).

```
14. Screen Lock Policy TEST
15. Windows Remote Management Policy
Введіть потрібні номери (через кому): 0
Оберіть резервну копію для відновлення:
0. 20241224_061216
1. 20241224_061523
2. 20241224_061927
3. 20241224_141349
4. 20241224_181144
5. 2025-05-27_181430
6. 2025-06-05_122840
7. 20250610_045002
Введіть потрібні номери (через кому): 7

Id : 6706adf9-a746-4698-9e84-2acc46335488
DisplayName : Additional Installation Policies
Path : cn={6706ADF9-A746-4698-9E84-2ACC46335488},cn=policies,cn=system,DC=makaryk,DC=net
Owner : MAKARYK\Domain Admins
DomainName : makaryk.net
CreationTime : 6/5/2025 12:28:40 PM
ModificationTime : 6/10/2025 5:06:28 AM
User : Microsoft.GroupPolicy.UserConfiguration
Computer : Microsoft.GroupPolicy.ComputerConfiguration
GpoStatus : AllSettingsEnabled
WmiFilter :
Description :

GPO Additional Installation Policies успішно відновлено з резервної копії: 'C:\GPO_Backups\Additional Installation Policies\20250610_045002'.
Σ(⌂ °Д °;)⌂---- Меню ----
```

Рисунок 4.1.2.7 – Реалізація відновлення GPO

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				94
Змн.	Арк.	№ докум.	Підпис	Дата		

Для редагування значень реєстрів групових політик скористаймося скриптом “EditGPO” (рис. 4.1.2.8). Є два варіанти якими може скористатися адміністратор, перший – запустити основний скрипт за яким застосовувалося GPO, це розглядається як розширений варіант втручання, оскільки є доступ до повного редагування файлу, а не лише зміни кількох вже існуючих реєстрів. Користувачу потрібно власноруч ввести назву скрипту, бо подекуди назви файлів та політик відрізняються. Під час відкриття файлу GPO у PowerShell ISE у консолі виведеться додаткове меню (рис. 4.1.2.10) у якому після збереження змін, видалиться існуюча GPO і застосується відредагований скрипт. Також для забезпечення журналювання реалізовано резервування та порівняння минулої та нової версії групової політики (рис. 4.1.2.11).

```
Введіть ваш вибір.
PS C:\Scripts> C:\Scripts\EditGPO.ps1
o(" ▽ ")o Меню редагування групових політик
1. Відкрити скрипт політики в PowerShell ISE
2. Переглянути і редагувати параметри GPO
0. Вийти
Введіть ваш вибір: 1
Additional Installation Policies.ps1
```

Рисунок 4.1.2.8 – Головне меню “EditGPO”

```
Soft Disk Quotas Policy.ps1
Введіть назву скрипту для редагування (включаючи розширення .ps1): Not.ps1
Скрипт не знайдено за вказаною назвою: Not.ps1
( → ) ~> Якщо скрипт з GPO було відредаговано
1. Видалити минулу GPO і виконати оновлений скрипт
0. Вийти в головне меню
Ваш вибір: |
```

Рисунок 4.1.2.9 – Помилкове вказування скрипта

```
1 $GPOName = "Screen Lock Policy"
2 $DomainName = "makaryk.net"
3 $KeyPath = "HKCU\Software\Policies\Microsoft\Windows\Control Panel\Desktop"
4
5
6
7
Soft Disk Quotas Policy.ps1
Введіть назву скрипту для редагування (включаючи розширення .ps1): Screen Lock Policy.ps1
( → ) ~> Якщо скрипт з GPO було відредаговано
1. Видалити минулу GPO і виконати оновлений скрипт
0. Вийти в головне меню
Ваш вибір: 1
Введіть ім'я GPO для видалення: Screen Lock Policy
GPO 'Screen Lock Policy' успішно видалено.
Виконання скрипта: C:\Scripts\GPO_Scripts\Screen Lock Policy.ps1
Створено новий GPO: Screen Lock Policy
```

Рис. 4.1.2.10 – Редагування GPO у власному файлі

```
GPO 'Screen Lock Policy' успішно резервовано у 'C:\GPO_Backups\Screen Lock Policy\2025-06-10_052413'.
Створений звіт для групової політики 'Screen Lock Policy'
Результати порівняння збережено в C:\GPO_Report\Comparison\Screen Lock Policy_2025-06-10_052413.txt
Скрипт виконано успішно.
```

Рис. 4.1.2.11 – Застосування додаткових функцій з моніторингу

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				
Змн.	Арк.	№ докум.	Підпис	Дата		95

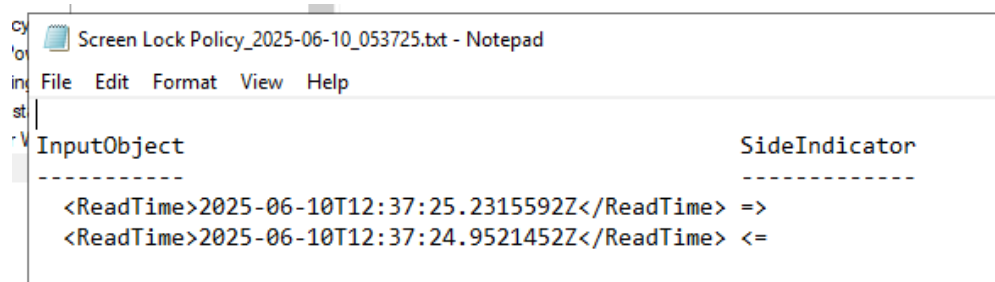


Рисунок 4.1.2.12 – Перевірка функції “CompareLastAndNewGPO”

Другий спосіб передбачає швидку базову зміну реєстру у консольному середовищі PowerShell. Користувач вручну вказує назву політики, після чого скрипт відображає поточні значення її ключів реєстру (рис. 4.1.2.13). Тому покроково вводячи нові значення, котрі вимагає скрипт політика успішно оновиться (рис. 4.1.2.14).

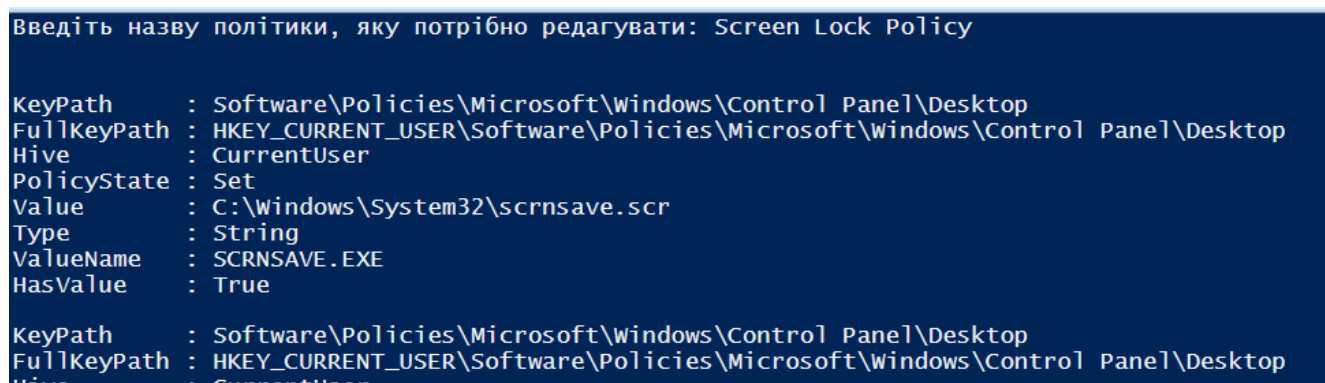


Рисунок 4.1.2.13 – Виведення інформацію про реєстри GPO

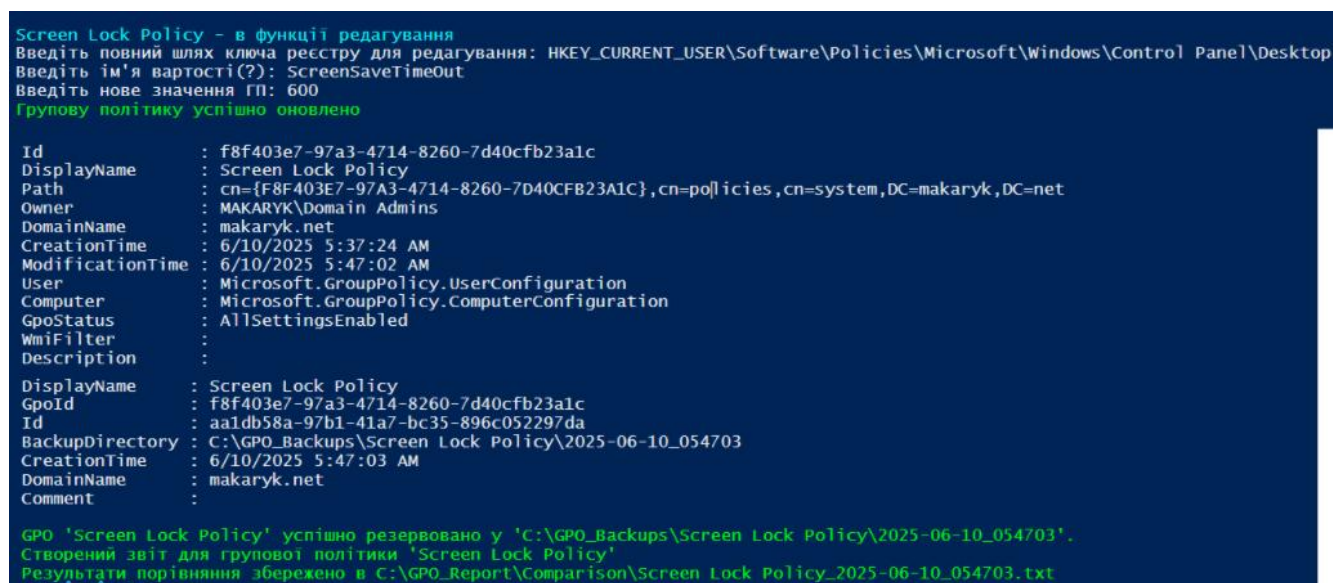


Рисунок 4.1.2.14 – Успішне редагування реєстрів GPO

Наступним розглянемо адміністрування прив’язок групових політик до організаційних одиниць чи домену. На початку, користувач бачить компактне меню

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				
Змн.	Арк.	№ докум.	Підпис	Дата		96

(рис. 4.1.2.15) у якому доступне три опції: створення нових зв'язків для політик, зміна параметрів застосування посилань та видалення існуючих прив'язок. За вибором першої опції (рис. 4.1.2.16) адміністратор отримає список усіх неприв'язаних GPO у домені і OU доступних для взаємодії. Далі можна вибрати чи задати однакові значення для усіх OU (рис. 4.1.2.17), чи підібрати кожній окремо(рис. 4.1.2.18).

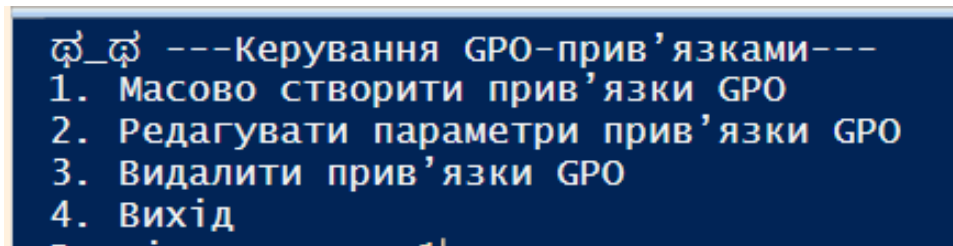


Рисунок 4.1.2.15 – Головне меню “EditLinkGPO”

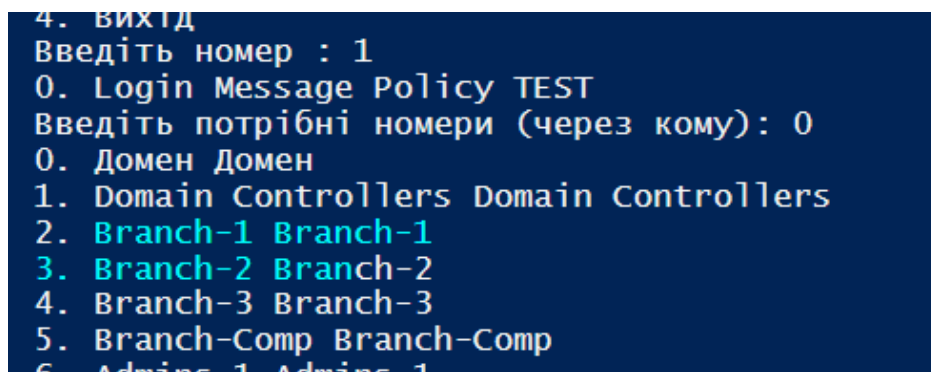


Рисунок 4.1.2.16 – Вибір неприв'язаних політик і OU домену

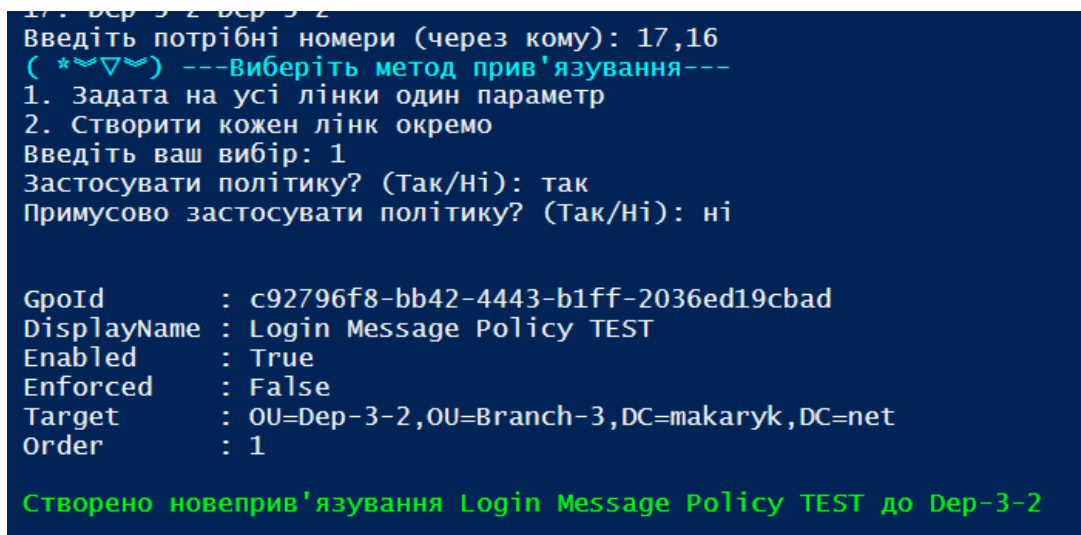


Рисунок 4.1.2.17 – Застосування до OU однакових параметрів прив'язки

```

( *▽*) ---Виберіть метод прив'язування---
1. Задана на усі лінки один параметр
2. Створити кожен лінк окремо
Введіть ваш вибір: 2
Застосувати політику 'Login Message Policy TEST' на 'Dep-3-1'? (Так/Ні): на
Некоректне значення. Спробуйте ще раз.
Застосувати політику 'Login Message Policy TEST' на 'Dep-3-1'? (Так/Ні): ні
Примусово застосувати політику 'Login Message Policy TEST' на 'Dep-3-1'? (Так/Ні): ні

GpoId      : 627fca56-3a1a-4e1c-8ac5-b99775946627
DisplayName : Login Message Policy TEST
Enabled     : False
Enforced    : False
Target      : OU=Dep-3-1,OU=Branch-3,DC=makaryk,DC=net
Order       : 1

Створено новеприв'язування Login Message Policy TEST до Dep-3-1

```

Рисунок 4.1.2.18 – Застосування до OU відміних параметрів прив'язки

```

GPO 'Login Message Policy TEST' успішно резервовано у 'C:\GPO_Backups\Login Message Policy TEST\2025-06-10_070036'.
Створений звіт для групової політики 'Login Message Policy TEST'
Застосувати політику 'Login Message Policy TEST' на 'Dep-3-2'? (Так/Ні): так
Примусово застосувати політику 'Login Message Policy TEST' на 'Dep-3-2'? (Так/Ні): ні
GpoId      : 627fca56-3a1a-4e1c-8ac5-b99775946627
DisplayName : Login Message Policy TEST
Enabled     : True
Enforced    : False
Target      : OU=Dep-3-2,OU=Branch-3,DC=makaryk,DC=net
Order       : 1

Створено новеприв'язування Login Message Policy TEST до Dep-3-2

```

Рисунок 4.1.2.19 – Застосування до OU відміних параметрів прив'язки

Щоб редагувати посилання адміністратор обирає одну чи кілька необхідних для роботи GPO. Наступним кроком обов'язково уточнює на потребу застосувати дане посилання у примусовому режимі відносно інших організаційних одиниць.

```

12. restrict registry editing tools and control panel
13. Screen Lock Policy
Введіть потрібні номери (через кому): 13
Опрацювання GPO: Screen Lock Policy
GPO 'Screen Lock Policy' пов'язано з: DC=makaryk,DC=net
Застосувати політику 'Screen Lock Policy' на 'DC=makaryk,DC=net'? (Так/Ні): так
Примусово застосувати політику 'Screen Lock Policy' на 'DC=makaryk,DC=net'? (Так/Ні): так
GpoId      : f8f403e7-97a3-4714-8260-7d40cfb23a1c
DisplayName : Screen Lock Policy
Enabled     : True
Enforced    : True
Target      : DC=makaryk,DC=net
Order       : 3

Посилання оновлено.

```

Рисунок 4.1.2.20 – Оновлення посилання

Для спрощення та прискорення видалення зв'язків було розроблено два варіанти реалізації цього процесу рис. (4.1.2.21). Перший варіант (рис. 4.1.2.22), “Видалення зв'язків за на назвою GPO” – повністю видаляє усі посилання на вказану політку, натомість “Видалення за ім'ям OU чи домену” (рис. 4.1.2.23) дає більш точновий підхід, дозволяючи користувачу вибрати конкретну GPO для від'язки від OU чи домену. Обидва варіанти супроводжуються журналюванням та резервуванням.

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				
Змн.	Арк.	№ докум.	Підпис	Дата		98

```

Введіть номер : 3
(°□°) J ^ _ _ _Видалення прив'язування GPO---
1. Видалити за GPO
2. Видалити за OU/доменом
Введіть ваш вибір: 1
0. Disk Quotas Policy
1. Default Domain Policy
2. Login Message Policy
3. Login Message Policy TEST

```

Рисунок 4.1.2.21 – Меню видалення посилання

```

Видалено лінк Login Message Policy TEST з Dep-3-2

DisplayName      : Login Message Policy TEST
GpoId             : 627fca56-3a1a-4e1c-8ac5-b99775946627
Id               : 258b3b4a-3c26-4ee9-add6-8b3fcd916a34
BackupDirectory  : C:\GPO_Backups\Login Message Policy TEST\2025-06-10_070311
CreationTime     : 6/10/2025 7:03:11 AM
DomainName       : makaryk.net
Comment          :

GPO 'Login Message Policy TEST' успішно резервовано у 'C:\GPO_Backups\Login Message Policy TEST\2025-06-10_070311'.
Створений звіт для групової політики 'Login Message Policy TEST'

```

Рисунок 4.1.2.22 – Видалення посилання за назвою GPO

```

15. Dep-2-2 Dep-2-2
16. Dep-3-1 Dep-3-1
17. Dep-3-2 Dep-3-2
Введіть потрібні номери (через кому): 16
0. Login Message Policy TEST
Введіть потрібні номери (через кому): 0

Id              : 627fca56-3a1a-4e1c-8ac5-b99775946627
DisplayName     : Login Message Policy TEST
Path            : cn={627FCA56-3A1A-4E1C-8AC5-B99775946627},cn=policies,cn=system,DC=makaryk,DC=net
Owner           : MAKARYK\Domain Admins
DomainName      : makaryk.net
CreationTime    : 6/10/2025 7:00:05 AM
ModificationTime : 6/10/2025 7:00:04 AM
User            : Microsoft.GroupPolicy.UserConfiguration
Computer        : Microsoft.GroupPolicy.ComputerConfiguration
GpoStatus       : AllSettingsEnabled
WmiFilter       :
Description     :

Видалено Login Message Policy TEST з Dep-3-1

```

Рисунок 4.1.2.23 – Видалення посилання на одній OU

Протестуємо роботу скрипту, що слугує інструментом зміни дозволів до рівня взаємодії облікового запису користувача та груповою політикою. На відміну від звичного початку попередніх скриптів “GPPermissionEdit” одразу виводить доступні GPO (рис. 4.4.2.24). Щойно користувач введе номери політик то виведеться вибір на спосіб за яким визначиться список акаунтів рівень доступу яких оновиться.

```

Введіть потрібні номери (через кому): 3,2,13,5
Задати до вибраних користувачів чи користувачів OU
1. Вибрані користувачі
2. Усі користувачі OU
Введіть ваш вибір: 1
0. Administrator
1. Guest
2. krbtgt
3. Blade-admin-11

```

Рисунок 4.1.2.24 – Вибір режиму задання дозволів

Виберемо кілька довільних користувачів, котрі матимуть змогу тільки переглядати в GPMC чи скориставшись командлетами скриптів, для цього введемо одиницю у меню з вибором типу доступу (рис. 4.1.2.25).

```
14. Robin-depComp-2
Введіть потрібні номери (через кому): 5,6,7
o(*^▽^*) ---Оберіть тип доступу---
1. Лише перегляд GPO
2. Редагування GPO
3. Редагування, видалення, зміна безпеки доступу
4. Видалити доступ
Введіть ваш вибір (1-4): 11,13
Неправильний вибір. Спробуйте ще раз.
Введіть ваш вибір (1-4): 1

DisplayName      : Login Message Policy TEST
DomainName       : makaryk.net
Owner            : MAKARYK\Domain Admins
Id               : 627fca56-3a1a-4e1c-8ac5-b99775946627
```

Рисунок 4.1.2.25 – Вибір режиму задання дозволів

```
Встановлено дозвіл на GpoRead на Microsoft.GroupPolicy.Gpo для FeiXiao-dep-2
DisplayName      : Additional Installation Policies
DomainName       : makaryk.net
Owner            : MAKARYK\Domain Admins
Id               : 6706adf9-a746-4698-9e84-2acc46335488
GpoStatus        : AllSettingsEnabled
Description      :
CreationTime     : 6/5/2025 12:28:40 PM
ModificationTime : 6/10/2025 9:06:00 AM
UserVersion      : AD Version: 1, SysVol Version: 1
ComputerVersion  : AD Version: 5, SysVol Version: 5
WmiFilter        :

Встановлено дозвіл на GpoRead на Microsoft.GroupPolicy.Gpo для Hook-admin-21
```

Рисунок 4.1.2.26 – Приклад успішної зміни дозволів

Для економії часу або в ситуаціях, коли у домені адміністратори не використовували ієрархічне іменування акаунтів, зручно застосувати спосіб призначення дозволів для усіх користувачів організаційної одиниці (рис. 4.1.2.27). Наприклад, у нашому випадку ми можемо надати повний доступ для OU адмінів, які в списку під номерами 5-7 (рис. 4.1.2.28).

```
Задати до вибраних користувачів чи користувачів OU
1. Вибрані користувачі
2. Усі користувачі OU
Введіть ваш вибір: 2
0. Domain Controllers
1. Branch-1
2. Branch-2
3. Branch-3
```

Рисунок 4.1.2.27 – Вибір режиму задання дозволів для усіх користувачів OU

```

16. Dep-3-2
Введіть потрібні номери (через кому): 5,6,7
o(*_▽_*)J ---Оберіть тип доступу---
1. Лише перегляд GPO
2. Редагування GPO
3. Редагування, видалення, зміна безпеки доступу
4. Видалити доступ
Введіть ваш вибір (1-4): 3

```

Рисунок 4.1.2.28 – Вибір режиму задання дозволів

```

Встановлено дозвіл на GpoEditDeleteModifySecurity на Microsoft.GroupPolicy.Gpo для FeiXiao-dep-2
DisplayName      : Deny removable disk
DomainName       : makaryk.net
Owner            : MAKARYK\Domain Admins
Id               : 98443c25-f07e-4cab-afbe-fddc715fb16c
GpoStatus        : AllSettingsEnabled
Description      :
CreationTime     : 6/5/2025 12:29:03 PM
ModificationTime : 6/10/2025 9:09:40 AM
UserVersion      : AD Version: 0, SysVol Version: 0
ComputerVersion  : AD Version: 4, SysVol Version: 4
WmiFilter        :
Встановлено дозвіл на GpoEditDeleteModifySecurity на Microsoft.GroupPolicy.Gpo для Hook-admin-21

```

Рисунок 4.1.2.29 – Приклад успішної зміни дозволів

WMI-фільтрування є невід'ємною частиною роботи з груповими політиками тож варто перевірити коректність роботи скрипту пов'язаного з адмініструванням даних фільтрів. Почнемо зі створення, на початку адміністратор вводить ім'я майбутнього фільтра (рис. 4.1.2.30) й вибирає значення WMI-запиту, оскільки ми побажали обрати запит з наявного списку, виводиться 13 найвживаніших варіантів запиту (рис. 4.1.2.31), а бажанням додається опис. У кінці виводиться повідомлення про створення WMI-фільтру з його головними параметрами (рис. 4.1.2.32).

```

Скрипт менеджменту WMI-фільтрів^(_w_o)
1. Створення нового WMI-фільтру
2. Прив'язання WMI-фільтру до GPO
3. Редагування WMI-фільтру
4. Видалення WMI-фільтру
0. Вийти
Введіть ваш вибір: 1
Введіть назву нового WMI-фільтра: "Deny access for cmd win10"
1: Задати власний фільтр
2: Вибрати зі списку (не повний)
Введіть ваш вибір: 2

```

Рисунок 4.1.2.30 – Меню та початок створення нового WMI-фільтру

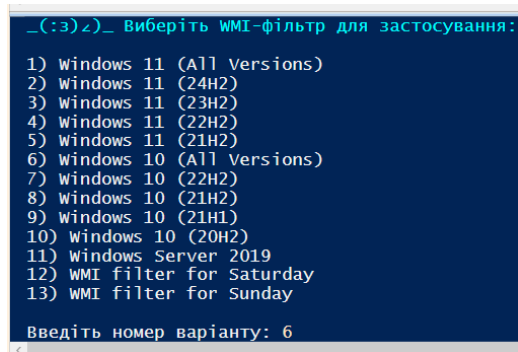


Рисунок 4.1.2.31 – Вибір доступних WMI-запитів

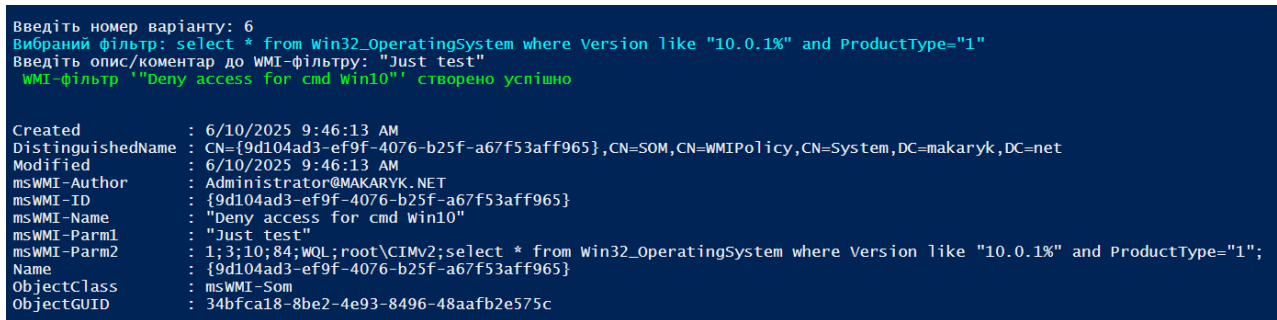


Рисунок 4.1.2.32 – Успішне створення WMI-фільтру

Перед прив'язкою WMI-фільтру до групової політики, перевіримо роботу GPO “Restrict CMD and PowerShell Access” на робочих машинах двох різних версій (рис. 4.1.2.33-4.1.2.34). Як було вказано у попередніх скрінах, запит встановлений на фільтрі дозволяє застосовувати політики лише на версіях операційної системи Windows 10.

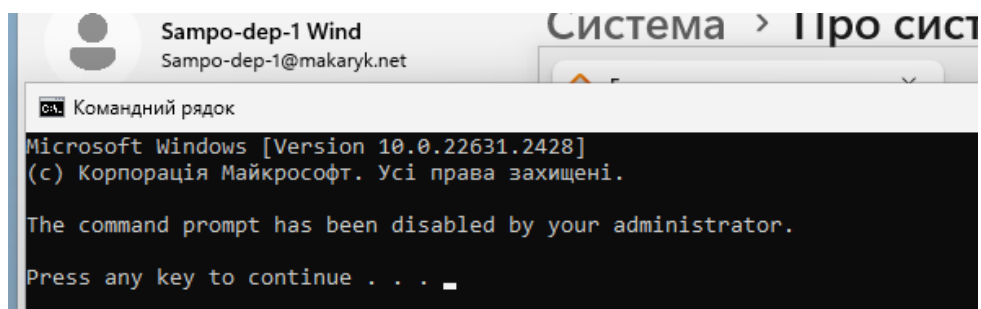


Рисунок 4.1.2.33 – Тестування роботи GPO до прив'язування на WS-Win11-22H3

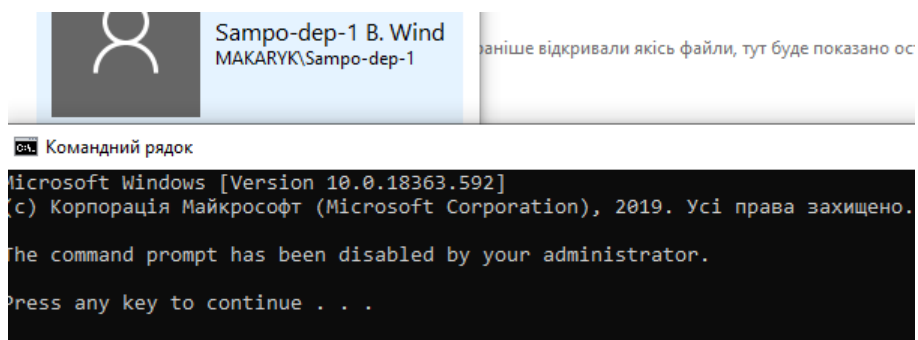


Рисунок 4.1.2.34 – Тестування роботи політики до прив'язування на WS-Win10

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				102
Змн.	Арк.	№ докум.	Підпис	Дата		

Настав час пов'язати GPO з WMI-фільтром (рис. 4.1.2.35). Результати роботи консольного застосунку після прив'язання підтверджують коректність функції: на Windows 11 (рис. 4.1.2.36) команда cmd працює у звичайному режимі, тоді як на Windows 10 (рис. 4.1.2.37) заборона зберігається, що відповідає налаштуванням фільтра.

```
Screen Lock Policy
Введіть назву WMI-фільтра для застосування: Restrict CMD and PowerShell Access
Set-GPOWmiFilter - Застосовано WMI-фільтрацію '"Deny access for cmd Win10"' до GPO 'Restrict CMD and PowerShell Access'
DisplayName      : Restrict CMD and PowerShell Access
GpoId            : 6682ebd7-b550-4d65-bd5a-b0b3e14f52e4
Id              : 7d4eb172-00ad-439a-ab42-5964b163761a
BackupDirectory  : C:\GPO_Backups\Restrict CMD and PowerShell Access\2025-06-10_095622
CreationTime     : 6/10/2025 9:56:22 AM
DomainName      : makaryk.net
Comment         :
GPO 'Restrict CMD and PowerShell Access' успішно резервовано у 'C:\GPO_Backups\Restrict CMD and PowerShell Access\2025-06-10_095622'.
Створений звіт для групової політики 'Restrict CMD and PowerShell Access'
```

Рисунок 4.1.2.35 – Прив'язування WMI-фільтра до політики

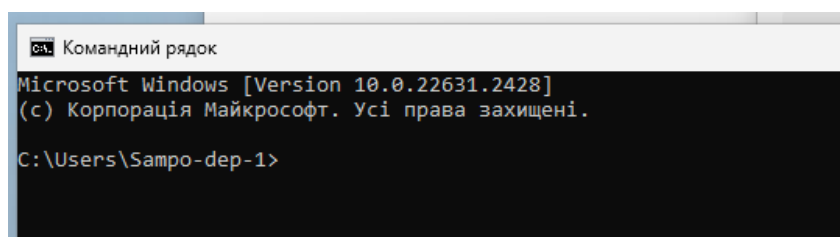


Рисунок 4.1.2.36 – Тестування роботи GPO після застосування на WS-Win11-22H3

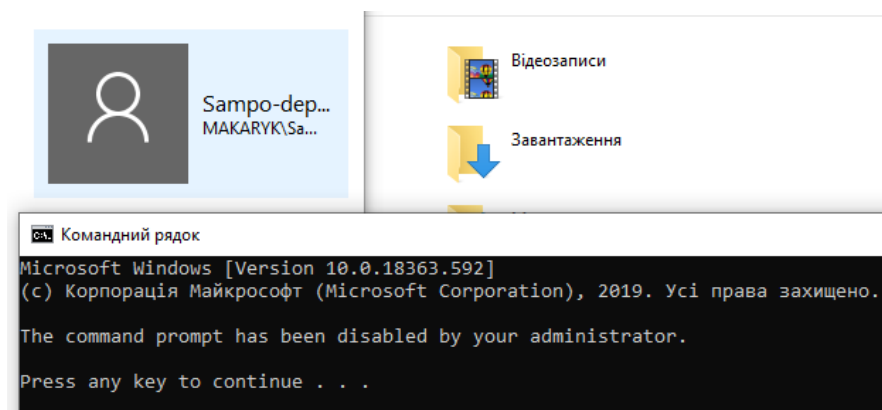


Рисунок 4.1.2.37 – Тестування роботи політики після застосування на WS-Win10

Редагування відбувається шляхом вводу назви вже існуючого фільтра, у нашому випадку “Only Windows 11”, та зміни його коментаря з запитом. Цього разу скористаємося опцією самостійного введення WMI-запиту. Задавши довільний коментар, скрипт виводить повідомлення про успішні зміни у фільтрі (рис. 4.1.2.38).

```
Only Windows 11
Введіть назву WMI-фільтра для застосування: Only Windows 11
1: Задати власний фільтр
2: Вибрати зі списку (не повний)
Введіть ваш вибір: 1
Введіть новий WMI-фільтр або 0 для виходу: select * from Win32_OperatingSystem where Version like "10.0.1%" and ProductType="1"
Введіть НОВИЙ опис/коментар до WMI-фільтра: ds
Оновлено фільтр 'Only Windows 11'
```

Рисунок 4.1.2.38 – Оновлення політики через ручне задавання WMI-запиту

Щоб видалити WMI-фільтр, оберемо відповідний варіант у меню, а потім вручну введемо його назву. З'явиться спливаюче вікно від PowerShell ISE (рис. 4.1.2.39) з попередженням, мета якого – базовий захист від помилкового введення. Підтвердивши намір, ми побачимо повідомлення про видалення (рис. 4.1.2.40). Щоб перевірити коректність видалення, переглянемо існуючі фільтри у графічному інтерфейсі GPMC (рис. 4.1.2.41).

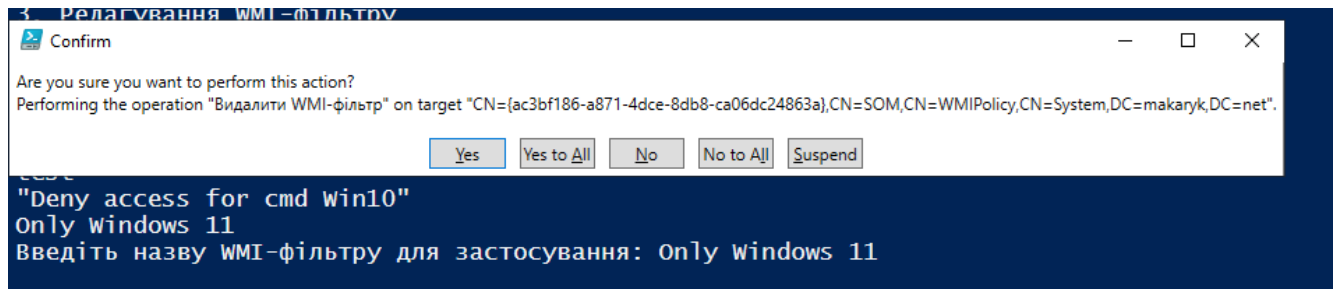


Рисунок 4.1.2.39 – Видалення WMI-фільтру

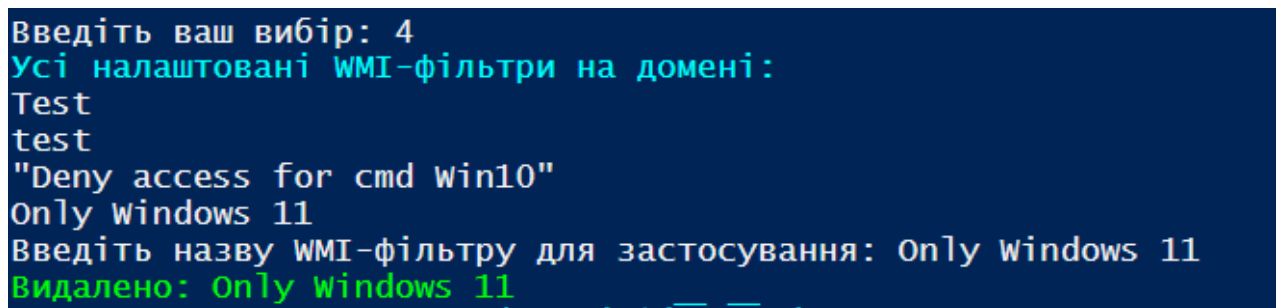


Рисунок 4.1.2.40 – Видалення WMI-фільтру

WMI Filters in makaryk.net					
Contents		Delegation			
Name	Description	Linked GPO	Author	Created	Modified
"Deny access for cmd..."	"Just test"	Restrict CMD and Pow...	Administrator@...	6/10/2025 9:46:...	6/10/2025 9:46:1...
Test	fdfd	<none>	Administrator@...	5/26/2025 3:57:...	5/26/2025 3:57:4...
test	test	<none>	Administrator@...	5/22/2025 11:45:...	5/22/2025 11:45:...

Рисунок 4.1.2.41 – Перевірка видалення фільтру через GPMC

Проведемо тестування основної системи журналювання у домені. Після активації скрипта, нас зустрічає меню з усіма доступними для реалізації функціями (рис. 4.1.2.42), розпочнемо з генерування звітів у форматі XML та HTML для кожної активної політики (рис. 4.1.2.43). GPO має записати звіти у теці зі шляхом "C:\GPO_Report\Назва_GPO\HTML/XML\Поточний_час", забезпечивши тим самим звучний пошук. Також доступно сформувати репорт для кількох певних політик (рис. 4.1.2.46).

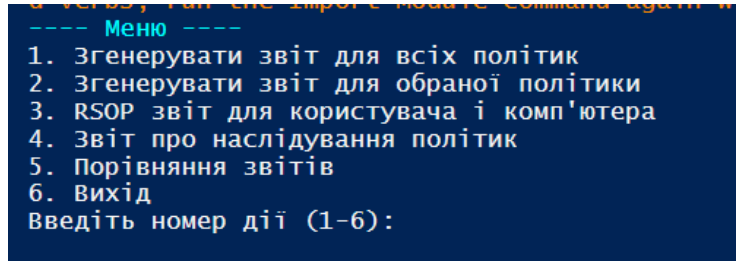


Рисунок 4.1.2.42 – Основне меню скрипта “ReportGeneration”

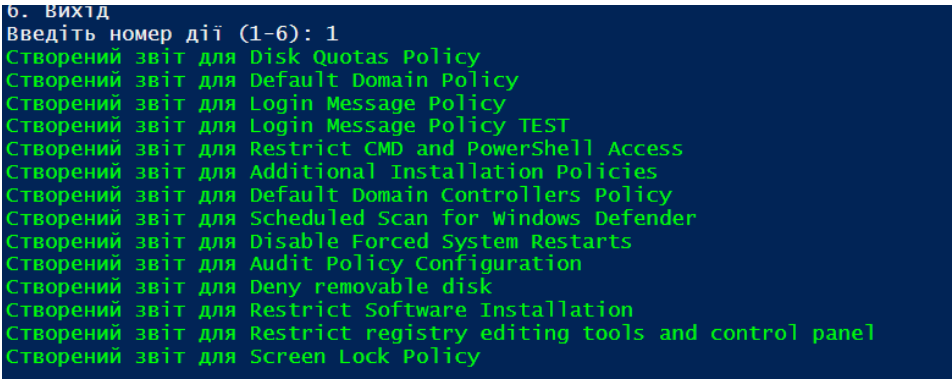


Рисунок 4.1.2.43 – Повідомлення про створення звітів

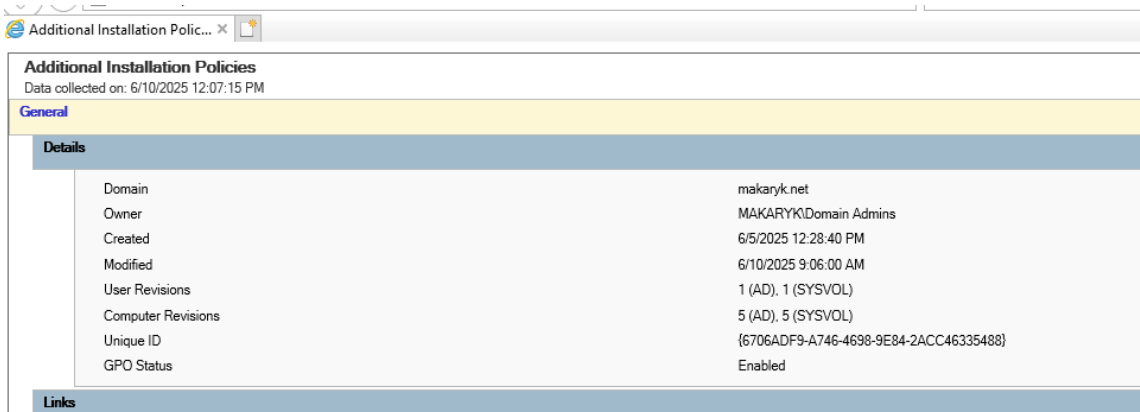


Рисунок 4.1.2.44 – Звіт у форматі HTML

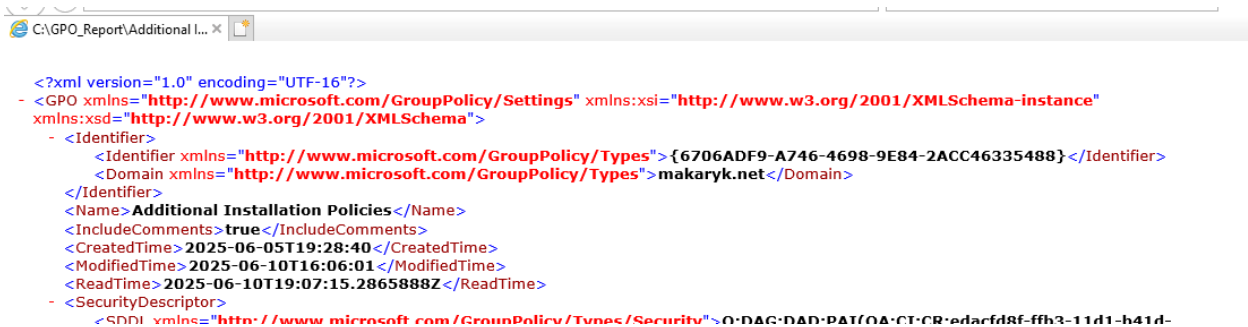


Рисунок 4.1.2.45 – Звіт у форматі XML

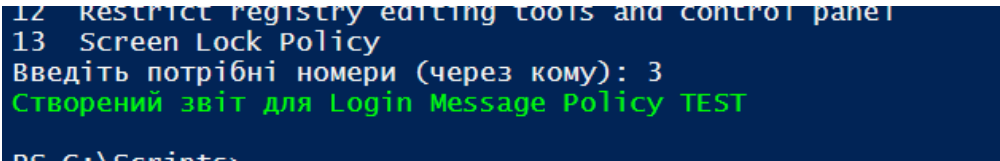


Рисунок 4.1.2.46 – Генерація звіту для однієї політики

Для отримання інформації про те, які групові політики наслідуються організаційними одиницями, необхідно вибрати відповідний пункт у меню та вказати потрібні OU (рис. 4.1.2.47). Після цього автоматично створюється звіт у форматі текстового документа, який містить детальні дані про наслідування політик. (рис. 4.1.2.48)

```

4. Звіт про наслідування політик
5. Порівняння звітів
6. Вихід
Введіть номер дії (1-6): 4
0 OU=Domain Controllers,DC=makaryk,DC=net
1 OU=Branch-1,DC=makaryk,DC=net
2 OU=Branch-2,DC=makaryk,DC=net
3 OU=Branch-3,DC=makaryk,DC=net
4 OU=Branch-Comp,DC=makaryk,DC=net
5 OU=Admins-1,OU=Branch-1,DC=makaryk,DC=net
6 OU=Admins-2,OU=Branch-2,DC=makaryk,DC=net
7 OU=Admins-3,OU=Branch-3,DC=makaryk,DC=net
8 OU=Branch-Comp-1,OU=Branch-Comp,DC=makaryk,DC=net
9 OU=Branch-Comp-2,OU=Branch-Comp,DC=makaryk,DC=net
10 OU=Branch-Comp-2,OU=Branch-Comp,DC=makaryk,DC=net
11 OU=Dep-1-1,OU=Branch-1,DC=makaryk,DC=net
12 OU=Dep-1-2,OU=Branch-1,DC=makaryk,DC=net
13 OU=Dep-2-1,OU=Branch-2,DC=makaryk,DC=net
14 OU=Dep-2-2,OU=Branch-2,DC=makaryk,DC=net
15 OU=Dep-3-1,OU=Branch-3,DC=makaryk,DC=net
16 OU=Dep-3-2,OU=Branch-3,DC=makaryk,DC=net
Введіть потрібні номери (через кому): 0,4
Звіт про наслідування політик збережено в C:\GPO_Report\GP_Inheritance_OU=Domain Controllers,DC=makaryk,DC=net_2025-06-10_121109.txt
Звіт про наслідування політик збережено в C:\GPO_Report\GP_Inheritance_OU=Branch-Comp,DC=makaryk,DC=net_2025-06-10_121109.txt

```

Рисунок 4.1.2.47 – Генерація звітів про наслідування GPO

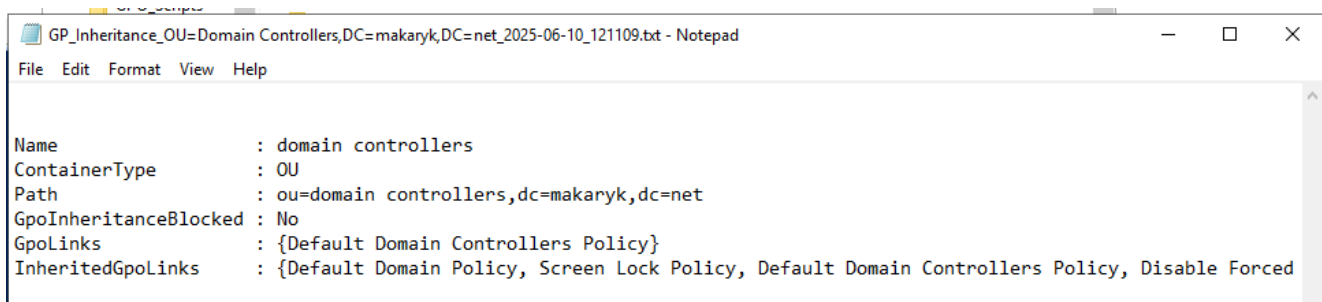


Рисунок 4.1.2.48 – Приклад звіту

Додатково доступно порівняння двох звітів однієї групової політики у скрипті передбачено механізм, який допомагає відстежувати зміни в налаштуваннях. Обравши відповідну опцію в меню та у консоль виводиться список звітів у форматі XML (рис. 4.1.2.49). Після цього скрипт аналізує вміст файлів і формує звіт про відмінності, який зберігається в текстовому файлі в директорії (рис. 4.1.2.50).

```

4. Звіт про наслідування політик
5. Порівняння звітів
6. Вихід
Введіть номер дії (1-6): 5
12 Restrict registry editing tools and control panel
13 Screen Lock Policy
Введіть потрібні номери (через кому): 13
19 Screen Lock Policy_2025-06-10_053548.xml
20 Screen Lock Policy_2025-06-10_053724.xml
21 Screen Lock Policy_2025-06-10_053725.xml
22 Screen Lock Policy_2025-06-10_054703.xml
23 Screen Lock Policy_2025-06-10_070211.xml
Введіть потрібні номери (через кому): 19,23
Результат порівняння збережено в C:\GPO_Report\Comparison_2025-06-10_121447.txt

```

Рисунок 4.1.2.49 – Генерація порівняння між двома звітами однієї GPO

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				106
Змн.	Арк.	№ докум.	Підпис	Дата		



Рисунок 4.1.2.50 – Генерація звіту між двома звітами однієї GPO

У попередньому розділі було зазначено, що на контролері домену налаштовано три заплановані завдання в Task Scheduler, які автоматизують адміністрування групових політик[18].

Name	Status	Triggers	Next Run Time	Last Run Time	Last
BackupAnd...	Ready	At 5:16 PM on 5/27/2025 - After triggered, repeat every 04:00:00 for a duration of 1 day.		5/28/2025 1:16:09 PM	Th
CheckUnlink...	Ready	At 6:00 AM every day	6/11/2025 6:00:00 AM	6/10/2025 6:14:39 AM	Th
CompareGP...	Ready	At 9:09 AM on 5/28/2025 - After triggered, repeat every 02:00:00 for a duration of 30.00:00:00.	6/10/2025 3:09:39 PM	6/10/2025 1:09:39 PM	Th
CompareGP...	Ready	At 6:35 PM on 5/27/2025 - After triggered, repeat every 02:00:00 for a duration of 30.00:00:00.	6/10/2025 2:35:03 PM	6/10/2025 12:49:41 PM	Th
CreateExplor...	Running	When the task is created or modified		6/9/2025 6:18:45 PM	Th
User_Feed_S...	Ready	At 4:45 PM every day - Trigger expires at 6/10/2035 4:45:54 PM.	6/10/2025 4:45:54 PM	6/10/2025 10:10:39 AM	Th

Рисунок 4.1.2.51 – Усі налаштовані планові завдання у Task Scheduler

Перше завдання відповідає за періодичне створення резервних копій GPO, друге виявляє неприв’язані політики та записує їх до лог-файлу, а третє аналізує GPO для пошуку випадкових повторень у налаштуваннях реєстру.

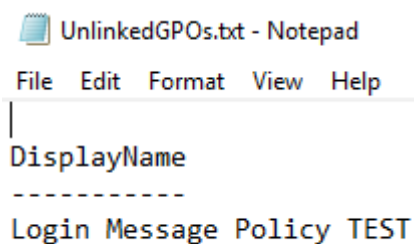


Рисунок 4.1.2.52 – Лог планового завдання “CheckUnlinkedGPOs”

Timestamp	Status	GPO Name	Backup Path
06/10/2025 04:50:02	????????	GPO 'Additional Installation Policies'	'C:\GPO_Backups\Additional Inst
06/10/2025 04:50:02	????????	GPO 'Audit Policy Configuration'	'C:\GPO_Backups\Audit Policy Configur
06/10/2025 04:50:03	????????	GPO 'Disable Forced System Restarts'	'C:\GPO_Backups\Disable Forced Sy
06/10/2025 04:50:03	????????	GPO 'Disk Quotas Policy'	'C:\GPO_Backups\Disk Quotas Policy\20250610_0

Рисунок 4.1.2.53 – Лог планового завдання “BackupAndLogs”

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				
Змн.	Арк.	№ докум.	Підпис	Дата		107

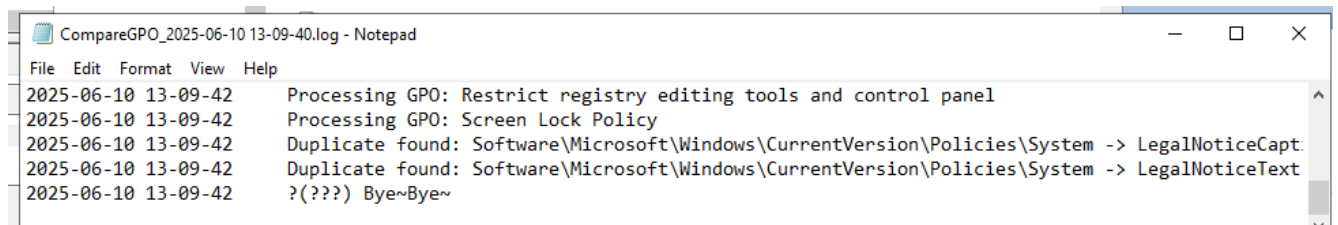


Рисунок 4.1.2.54 – Лог планового завдання “CompareGPO_Duplicates”

4.2 Оцінка ефективності автоматизованого управління GPO

4.2.1 Порівняння швидкості ручного та автоматизованого адміністрування

У межах практичного дослідження було проведено серію вимірювань часу, необхідного для виконання типових дій спрямованих на адміністрування групових політик у домені. Skorиставшись середовищем Active Directory, а саме його графічним інтерфейсом GPMC та створених раніше PowerShell-скриптів, перевіримо який метод ефективніше використовувати.

Для додаткової точності під час вимірювання тривалості виконання сценаріїв було використано командлет “Get-Date” [19] (рис. 4.2.1.1). Він дозволив зафіксувати час початку та завершення операцій, що у поєднанні з розрахунком різниці забезпечило точні результати, незалежні від зовнішніх чинників чи суб’єктивної оцінки.

```

L$
$start = Get-Date
Main
$end = Get-Date
$duration = $end - $start
Write-Host "Скрипт виконувався: $($duration.TotalSeconds) секунд"

```

Рисунок 4.2.1.1 – Приклад використання командлети у “EditGPO”

GPO 'Login Message Policy TimeTest' успішно резервовано у 'C:\GPO_Backups\Login Message Policy TimeTest\2025-06-10_161138'.
Створений звіт для групової політики 'Login Message Policy TimeTest'
Скрипт виконувався: 0.4690308 секунд

Рисунок 4.2.1.2 – Результат роботи конмадлети

Натомість у графічному інтерфейсі подібні засоби для автоматичного фіксування часу виконання дій не передбачені. Тому вимірювання довелося проводити вручну за допомогою вбудованого в операційну систему секундоміра чи . Такий метод супроводжується значною людською похибкою, оскільки потребує реакції оператора, що унеможлиблює мілісекундну точність і знижує репрезентативність результатів.

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				
Змн.	Арк.	№ докум.	Підпис	Дата		108

Таблиця 4.2.1.1 – Порівняння PowerShell та GPMC

Функція	PowerShell	GPMC
Створення GPO	0,46	120
Редагування GPO	35,1	53
Видалення	7,9	25
Відновлення	16,3	23
Делегування	19,035	15
Створення WMI-фільтру	18,34	32
Прив'язка фільтру	25,68	9
Резервування	3,8	10
Журналювання	5,9	22

Проаналізувавши отримані результати з таблиці, можна дійти висновку, що використання мови сценаріїв PowerShell є значно ефективнішим, ніж застосування графічного інтерфейсу. Хоча деякі функції у GPMC виконуються швидше – наприклад, прив'язка WMI-фільтру чи делегування прав – це пояснюється лише зручністю контекстного меню, що полегшує виконання одноразових дій. Проте в умовах реального адміністрування, особливо у масштабних доменах, ключову роль відіграє саме автоматизація.

На мою думку, GPMC не є оптимальним середовищем для створення нових політик через перевантажену ієрархічну структуру тек, у яких зберігаються параметри налаштувань. Замість цього значно простіше знайти на сторонньому сайті [\[20\]](#) повний шлях реєстру, його значення для налаштування політики та швидко застосувати їх за допомогою PowerShell.

4.2.2 Аналіз переваг та обмежень використання PowerShell

Давайте повернемося до раніше розглянутої таблиці з першого розділу, у якій були описані основні переваги та недоліки використання PowerShell. Таблиця дозволила зробити загальне уявлення про функціональні можливості PowerShell у порівнянні з графічним інтерфейсом. Однак для повноти аналізу доцільно більш детально розглянути сильні та слабкі сторони PowerShell в застосуванні для адміністрування GPO.

Переваги використання PowerShell:

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				
Змн.	Арк.	№ докум.	Підпис	Дата		109

- Автоматизація – можливість максимально автоматизувати рутинні операції, зменшивши потребу у ручному вводі, що значно економить час і знижує ризик помилок
- Масштабованість – на практичному досвіді тестуванні кількох скриптів, наприклад зміни рівня доступу, ми з'ясували, що код легко адаптувати під використання для роботи з десятками й сотнями об'єктами AD.
- Швидкість виконання завдань – у порівнянні з GPMC, PowerShell виконує ті самі операції значно швидше, особливо коли йдеться про пакетну обробку.
- Гнучкість та точність – дає досягти функціоналу який не передбачено у GI, як автоматичне порівняння при оновленні GPO.
- Створення звітів - За допомогою таких командлетів, як Get-GPOReport, можна автоматично створювати звіти в різних форматах для подальшого аналізу чи документації.

Обмеження у використанні консольної оболонки:

- Неінтуїтивний інтерфейс – практична відсутність графічного інтерфейсу, тому для початківців використання командлетів може видатися складним без попереднього досвіду.
- Вища вимога до знань – адміністратор повинен добре орієнтуватися у структурі та об'єктах Active Directory.
- Менше інтерактивності - на відміну від GPMC, у PowerShell не видно змін в реальному часі, тому важливо додавати в скрипти перевірки, логування та обробку помилок.
- Не всі функції реалізовані в командлетах - деякі дії, як-то менеджмент WMI-фільтрів, можуть бути реалізовані лише через GPMC або сторонні утиліти, а не через стандартні командлети, тому довелося створювати примітивні функції вручну.

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				110
Змн.	Арк.	№ докум.	Підпис	Дата		

Отже, PowerShell майже ідеально підходить для автоматизованого адміністрування GPO, особливо в середовищах із великою кількістю політик. Однак для максимальної ефективності вимагає від адміністратора певної технічної підготовки та досвіду. У поєднанні з GPMC PowerShell дозволяє створити зручну систему керування політиками.

4.3 Висновки щодо впровадження автоматизованого управління GPO

Проведений аналіз показав, що впровадження автоматизованого управління груповими політиками за допомогою PowerShell, а особливо PowerShell ISE є обґрунтовано ефективним рішенням для адміністрування домену мережі. За умови використання великої кількості GPO, складної ієрархії організаційних одиниць і частих змін у налаштуваннях.

Автоматизація дозволяє значно зменшити витрати часу на рутинні адміністративні задачі, мінімізує ймовірність помилок, пов'язаних з людським фактором, і водночас забезпечує повторюваність операцій. Наприклад, автоматичне виявлення неактивних або не зв'язаних GPO, генерація звітів у форматі XML чи HTML, контроль за змінами політик – усе це можливо реалізувати за допомогою скриптів PowerShell, інтегрованих у планувальник завдань Windows.

Додатково це спрощує аудит і документування змін у політиках без потреби в ручному втручанні для генерації звітів. Крім того налаштування додаткового логування для кожного етапу роботи з груповими політиками спрощує відстеження помилок та підвищує прозорість управління.

У висновку – перехід до автоматизованого управління груповими політиками є виправданим та бажаним кроком для підвищення ефективності адміністрування, оптимізації часових ресурсів і забезпечення стабільної контрольованої системи.

Висновки до четвертого розділу.

У ході написання четвертого розділу було протестовано роботу лабораторного стенду методом дослідження дії запроваджених групових політик на організаційні одиниці та домен. Основна увага приділялася методам реалізації адміністрування

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				
Змн.	Арк.	№ докум.	Підпис	Дата		111

GPO за допомогою середовища PowerShell – детально проаналізовано принципи їхньої роботи та проведено практичну перевірку.

Проведено дослідження порівняння швидкості виконання певних адміністративних функцій у Active Directory, зокрема за допомогою графічного інтерфейсу GPMC та PowerShell. Визначено основні переваги та недоліки використання мови сценаріїв для менеджменту групових політик.

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				
Змн.	Арк.	№ докум.	Підпис	Дата		112

ВИСНОВКИ

У результаті виконання кваліфікаційної роботи було реалізовано проєкт з розширеного управління груповими політиками за допомогою PowerShell.

У межах проєкту досліджено можливості PowerShell і його інтегрованого середовища PowerShell ISE як інструментів для створення, модифікації й адміністрування GPO. Практична частина включає розробку низки скриптів, які автоматизують ключові адміністративні процеси: створення, редагування, резервне копіювання GPO, прив'язку до OU, делегування прав, керування WMI-фільтрами. Особливу увагу приділено інтеграції з планувальником завдань Windows для організації періодичного запуску сценаріїв.

Проведено дослідження ефективності PowerShell порівняно з традиційним графічним інтерфейсом GPMC. Тестування засвідчило перевагу PowerShell у швидкості масових операцій, масштабованості та розширюваності функціоналу.

У цілому, реалізоване рішення підтвердило доцільність використання PowerShell для автоматизації управління груповими політиками. Використання PowerShell дозволяє істотно скоротити ручну працю адміністратора, забезпечити повторюваність налаштувань безпеки та підвищити ефективність обслуговування інформаційної інфраструктури.

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				
Змн.	Арк.	№ докум.	Підпис	Дата		113

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Group Policy Management Console. Microsoft Learn: Build skills that open doors in your career. [Електронний ресурс] URL: [https://learn.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2012-r2-and-2012/dn265969\(v=ws.11\)](https://learn.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2012-r2-and-2012/dn265969(v=ws.11))
2. Tom (2023). PowerShell Command Line vs. PowerShell ISE: Understanding the Differences. Medium. [Електронний ресурс] URL: <https://medium.com/tomtalkspowershell/powershell-command-line-vs-powershell-ise-understanding-the-differences-be18c3820f0f>
3. What is .NET Framework? A software development framework | .NET. Microsoft. [Електронний ресурс] URL: <https://dotnet.microsoft.com/en-us/learn/dotnet/what-is-dotnet-framework>
4. Bigelow S. J. (2022) What is Windows PowerShell ISE (Integrated Scripting Environment)?. SearchWindows Server. [Електронний ресурс] URL: <https://www.techtarget.com/searchwindowsserver/definition/Microsoft-Windows-PowerShell-Integrated-Scripting-Environment-ISE>
5. Marshall S. What is Group Policy? Its Types and Uses in Active Directory. Lepide Blog: A Guide to IT Security, Compliance and IT Operations. [Електронний ресурс] URL: <https://www.lepide.com/blog/what-is-group-policy/>
6. Group Policy Management Console. Microsoft Learn: Build skills that open doors in your career. [Електронний ресурс] URL: [https://learn.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2012-r2-and-2012/dn265969\(v=ws.11\)](https://learn.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2012-r2-and-2012/dn265969(v=ws.11))
7. Reviewing the Domain Models. Microsoft Learn: Build skills that open doors in your career. [Електронний ресурс] URL: <https://learn.microsoft.com/en-us/windows-server/identity/ad-ds/plan/reviewing-the-domain-models>
8. Roles, Role Services, and Features included in Windows Server - Server Core. Microsoft Learn: Build skills that open doors in your career. [Електронний ресурс] URL: <https://learn.microsoft.com/en-us/windows-server/administration/server-core/server-core-roles-and-services?tabs=roles>.

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				114
Змн.	Арк.	№ докум.	Підпис	Дата		

9. Active Directory User and Account Types. IS Decisions. [Електронний ресурс] URL: <https://www.isdecisions.com/en/glossary/active-directory-user-and-account-types>
10. What's new in Windows 11, version 23H2 for IT pros. Microsoft Learn: Build skills that open doors in your career. [Електронний ресурс] URL: <https://learn.microsoft.com/en-us/windows/whats-new/whats-new-windows-11-version-23h2>
11. What's new in Windows 11, version 24H2 for IT pros. Microsoft Learn: Build skills that open doors in your career. [Електронний ресурс] URL: <https://learn.microsoft.com/en-us/windows/whats-new/whats-new-windows-11-version-24h2>
12. Kawushika B. Active Directory, DNS and DHCP Configuration on Windows Server. LinkedIn: Log In or Sign Up. [Електронний ресурс] URL: <https://www.linkedin.com/pulse/active-directory-dns-dhcp-configuration-windows-server-kawushika-v0k4c>
13. Get-GPInheritance (GroupPolicy). Microsoft Learn: Build skills that open doors in your career. [Електронний ресурс] URL: <https://learn.microsoft.com/en-us/powershell/module/grouppolicy/get-gpinheritance?view=windowsserver2025-ps>
14. Set-GPLink (GroupPolicy). Microsoft Learn: Build skills that open doors in your career. [Електронний ресурс] URL: <https://learn.microsoft.com/en-us/powershell/module/grouppolicy/set-gplink?view=windowsserver2025-ps>
15. Get-GPOReport (GroupPolicy). Microsoft Learn: Build skills that open doors in your career. [Електронний ресурс] URL: <https://learn.microsoft.com/en-us/powershell/module/grouppolicy/get-gporeport?view=windowsserver2025-ps>
16. WMI Filter Cheat Sheet - Danny Moran. Danny Moran. [Електронний ресурс] URL: <https://www.dannymoran.com/wmi-filter-cheat-sheet/>

- 17.Windows Task Scheduler | NXLog Documentation. Redirect Notice.
[Электронный ресурс] URL: <https://docs.nxlog.co/integrate/windows-task-scheduler.html>
- 18.Get-Date (Microsoft.PowerShell.Utility) - PowerShell. Microsoft Learn: Build skills that open doors in your career. [Электронный ресурс] URL: <https://learn.microsoft.com/en-us/powershell/module/microsoft.powershell.utility/get-date?view=powershell-7.5>
- 19.How to create, manage, and delete users and groups in Active Directory. Kamatera. [Электронный ресурс] URL: <https://www.kamatera.com/knowledgebase/how-to-manage-and-delete-users-in-active-directory/>
- 20.Group Policy Search. Group Policy Search. [Электронный ресурс] URL: <https://gpsearch.azurewebsites.net/#176>

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				116
Змн.	Арк.	№ докум.	Підпис	Дата		

ДОДАТКИ

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				117
Змн.	Арк.	№ докум.	Підпис	Дата		

“UserGPOModule”

```

Function Get-GPAllRegistryValues{
    Param(
        [Parameter(
            ValueFromPipelineByPropertyName = $true
        )]
        [Alias('DisplayName')]
        [string]$Name
    )
    Begin{
        $BaseKeys = 'HKLM\System', 'HKLM\Software', 'HKCU\Software', 'HKCU\System'
        Function Get-GPRecursiveRegistryValues{
            [cmdletbinding()]
            Param(
                [string]$GPOName,
                [string]$Key
            )
            $GPORegistryValues = Get-GPRegistryValue -Name $GPOName -key $Key -
ErrorAction SilentlyContinue
            foreach($item in $GPORegistryValues){
                if ($item.ValueName){ $item
            }else{ Get-GPRecursiveRegistryValues -key $item.FullKeyPath -
GPOName $GPOName }
            }
        }
        Process{
            foreach($Key in $BaseKeys){ Get-GPRecursiveRegistryValues -GPOName $Name -key
$Key }
        }
        End{}
    }

    function Get-GPOName {
        [cmdletbinding()]
        param (
            [string]$Prompt
        )
        $GPOName = Read-Host -Prompt $Prompt
        if (-not (Get-GPO -Name $GPOName -ErrorAction SilentlyContinue)) {
            Write-Host "GPO з ім'ям '$GPOName' не знайдено!" -ForegroundColor Red
            return $null
        }
        return $GPOName
    }

    function CreateGPO {
        [CmdletBinding()]
        param (
            [string]$GPOName,
            [string]$DomainName
        )
        if (Get-GPO -Name $GPOName -ErrorAction SilentlyContinue) {
            Write-Host "GPO з ім'ям '$GPOName' вже існує!" -ForegroundColor Red
            return
        } else {
            $NewGPO = New-GPO -Name $GPOName -Domain $DomainName
            Write-Host "Створено новий GPO: $($NewGPO.DisplayName)" -ForegroundColor Green
        }
    }

    function ExpandedBackup-GPO {
        [CmdletBinding()]
        param (
            [string]$GPOName,
            [string]$DomainName,
            [string]$BackupBasePath = "C:\GPO_Backups",
            [string]$LogFile = "C:\GPO_Backups\GPO_Backup_Log.txt"
        )
        $timestamp = Get-Date -Format "yyyy-MM-dd_HH:mm:ss"
    }

```

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				118
Змн.	Арк.	№ докум.	Підпис	Дата		

```

if ($GPOName) {
    $BackupPath = Join-Path -Path $BackupBasePath -ChildPath "$GPOName\$timestamp"
    if (-not (Test-Path $BackupPath)) { New-Item -ItemType Directory -Path $BackupPath -
Force | Out-Null }
    try {
        Backup-GPO -Name $GPOName -Path $BackupPath -Domain $DomainName
        Write-Host "GPO '$GPOName' успішно резервовано у '$BackupPath'." -ForegroundColor
Green
        $logEntry = "$GPOName $timestamp - Резервна копія GPO '$GPOName' успішно створена у
'$BackupPath'."
        Add-Content -Path $LogFile -Value $logEntry
    }
    catch {Write-Host "Помилка резервування: $_" -ForegroundColor Red}
}

function GpoGenerateReport {
    [CmdletBinding()]
    param (
        [string]$GPOName,
        [string]$GPOReportPath = "C:\GPO_Report"
    )

    $timestamp = Get-Date -Format "yyyy-MM-dd_HH:mm:ss"
    $ReportFolder = Join-Path $GPOReportPath $GPOName

    $formats = @("html", "xml")

    foreach ($format in $formats) {
        $folderPath = Join-Path $ReportFolder $format
        if (-not (Test-Path $folderPath)) { New-Item -ItemType Directory -Path $folderPath
-Force | Out-Null }
    }

    try {
        $htmlPath = Join-Path -Path (Join-Path $ReportFolder "html") -ChildPath
"$GPOName\_timestamp.html"
        $xmlPath = Join-Path -Path (Join-Path $ReportFolder "xml") -ChildPath
"$GPOName\_timestamp.xml"

        Get-GPOReport -Name $GPOName -ReportType Html -Path $htmlPath
        Get-GPOReport -Name $GPOName -ReportType Xml -Path $xmlPath

        Write-Host "Створений звіт для групової політики '$GPOName'" -ForegroundColor Green
    }
    catch { Write-Host "Помилка генерації звіту для групової політики '$GPOName': $_" -
ForegroundColor Red }
}

function Input-Index {
    param (
        [Parameter(Mandatory=$true)]
        [Object[]]$ImportantList
    )
    do {
        $validInput = $true
        $TrueList = @()
        $ListIndexes = Read-Host "Введіть потрібні номери (через кому)"

        foreach ($index in $ListIndexes -split ',') {
            if (-not ($index -match '^\\d+$')) {
                Write-Host "Помилка - '$index' повинен бути числом." -ForegroundColor Red
                $validInput = $false
                break
            }
            elseif ([int]$index -ge $ImportantList.Count -or [int]$index -lt 0) {
                Write-Host "Помилка - '$index' виходить за межі списку" -ForegroundColor Red
                $validInput = $false
                break
            }
            $TrueList += $ImportantList[$index]
        }
    }
}

```

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				119
Змн.	Арк.	№ докум.	Підпис	Дата		

```

    } while ($validInput -ne $true)
    return $TrueList
}

function Write-GPOEventLog {
    param (
        [Parameter(Mandatory)]
        [string]$GPOName,

        [Parameter(Mandatory)]
        [ValidateSet("Create", "Edit", "Delete", "Restore", "WMI")]
        [string]$Mode
    )

    $logFilePath = "C:\Logs\GeneralLog.txt"
    $timestamp = Get-Date -Format "yyyy-MM-dd HH:mm:ss"

    try {
        switch ($Mode) {
            "Edit"      { $action = "Edit GPO" }
            "Delete"    { $action = "Delete GPO" }
            "Restore"    { $action = "Restore GPO" }
            "WMI"       { $action = "Set WMI filter on GPO" }
            "Create"    { $action = "Create GPO" }
        }

        $logEntry = "[ $timestamp] $action - $GPOName"
        if (-not (Test-Path $logFilePath)) { New-Item -ItemType File -Path
$logFilePath -Force | Out-Null }
        Add-Content -Path $logFilePath -Value $logEntry
    }
    catch {Write-Host "Сталася помилка при приписі події : $_" -ForegroundColor
Red}
}

```

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				120
Змн.	Арк.	№ докум.	Підпис	Дата		

“Additional Installation Policies”

```

Import-Module GroupPolicy
Import-Module C:\Scripts\UserGPOModule.psm1 -Force -WarningAction SilentlyContinue

$DomainName = «makaryk.net»
$GpoName = «Additional Installation Policies»
$UserOU = «OU=Branch-Comp,DC=makaryk,DC=net»
$ScriptPath = «C:\Scripts\CreateGPO.ps1»
$RegistryPath = «HKKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows\Installer»

try {
    CreateGPO -GPOName $GPOName -DomainName $DomainName

    Set-GPRegistryValue -Name $GpoName -Key $RegistryPath -ValueName «DisableLUAPatching» -
Type Dword -Value 1
    Set-GPRegistryValue -Name $GpoName -Key $RegistryPath -ValueName «DisableRollback» -
Type Dword -Value 1
    Set-GPRegistryValue -Name $GpoName -Key $RegistryPath -ValueName «DisablePatch» -Type
Dword -Value 1
    Set-GPRegistryValue -Name $GpoName -Key $RegistryPath -ValueName «Logging» -Type String
-Value «iearmxv»

    New-GPLink -Name $GpoName -Target $UserOU
    Write-Host «GPO '$GpoName' успішно створено до $UserOU!» -ForegroundColor Green

    ExpandedBackup-GPO -GPOName $GpoName -DomainName $DomainName
    GpoGenerateReport -GPOName $GpoName
    Write-GPOEventLog -GPOName $GpoName -Mode Create
}
catch { Write-Host «Сталася помилка при створенні GPO '$GPOName': $_» -ForegroundColor Red
}

```

“Audit Policy Configuration”

```

Import-Module GroupPolicy
Import-Module C:\Scripts\UserGPOModule.psm1 -Force -WarningAction SilentlyContinue

$GpoName = "Audit Policy Configuration"
$gpoPath = "OU=Branch-Comp,DC=makaryk,DC=net"
$gpoKeyAccMan = "HKLM\Software\Policies\Microsoft\Windows\EventLog\Account Management"
$DomainName = "makaryk.net"

try {
    CreateGPO -GPOName $GPOName -DomainName $DomainName

    # Account Management
    Set-GPRegistryValue -Name $GpoName -Key $gpoKeyAccMan -ValueName
"AuditComputerAccountManagement" -Type DWORD -Value 1
    Set-GPRegistryValue -Name $GpoName -Key $gpoKeyAccMan -ValueName
"AuditOtherAccountManagementEvents" -Type DWORD -Value 3
    Set-GPRegistryValue -Name $GpoName -Key $gpoKeyAccMan -ValueName
"AuditSecurityGroupManagement" -Type DWORD -Value 3
    Set-GPRegistryValue -Name $GpoName -Key $gpoKeyAccMan -ValueName
"AuditUserAccountManagement" -Type DWORD -Value 3

    $gpoKeyDetailedTrack = "HKLM\Software\Policies\Microsoft\Windows\EventLog\Detailed
Tracking"
    # Detailed Tracking
    Set-GPRegistryValue -Name $GpoName -Key $gpoKeyDetailedTrack -ValueName
"AuditPlugAndPlayEvents" -Type DWORD -Value 1
    Set-GPRegistryValue -Name $GpoName -Key $gpoKeyDetailedTrack -ValueName
"AuditProcessCreation" -Type DWORD -Value 1

    $gpoKeyDSAccess = "HKLM\Software\Policies\Microsoft\Windows\EventLog\DS Access"
    # DS Access
    Set-GPRegistryValue -Name $GpoName -Key $gpoKeyDSAccess -ValueName
"AuditDirectoryServiceAccess" -Type DWORD -Value 3
    Set-GPRegistryValue -Name $GpoName -Key $gpoKeyDSAccess -ValueName
"AuditDirectoryServiceChanges" -Type DWORD -Value 3

    $gpoKeyEventLog = "HKLM\Software\Policies\Microsoft\Windows\EventLog\Logon/Logoff"

```

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				121
Змн.	Арк.	№ докум.	Підпис	Дата		

```

# Logon/Logoff
Set-GPRegistryValue -Name $GpoName -Key $gpoKeyEventLog -ValueName
"AuditAccountLockout" -Type DWORD -Value 3
Set-GPRegistryValue -Name $GpoName -Key $gpoKeyEventLog -ValueName
"AuditGroupMembership" -Type DWORD -Value 1
Set-GPRegistryValue -Name $GpoName -Key $gpoKeyEventLog -ValueName "AuditLogoff" -Type
DWORD -Value 1
Set-GPRegistryValue -Name $GpoName -Key $gpoKeyEventLog -ValueName "AuditLogon" -Type
DWORD -Value 3
Set-GPRegistryValue -Name $GpoName -Key $gpoKeyEventLog -ValueName "AuditSpecialLogon"
-Type DWORD -Value 1

# Object Access
Set-GPRegistryValue -Name $GpoName -Key
"HKLM\Software\Policies\Microsoft\windows\EventLog\Object Access" -ValueName
"AuditRemovableStorage" -Type DWORD -Value 3

$gpoKeyEventPolicyChange = "HKLM\Software\Policies\Microsoft\windows\EventLog\Policy
Change"
# Policy Change
Set-GPRegistryValue -Name $GpoName -Key $gpoKeyEventPolicyChange -ValueName
"AuditAuditPolicyChange" -Type DWORD -Value 3
Set-GPRegistryValue -Name $GpoName -Key $gpoKeyEventPolicyChange -ValueName
"AuditAuthenticationPolicyChange" -Type DWORD -Value 1
Set-GPRegistryValue -Name $GpoName -Key $gpoKeyEventPolicyChange -ValueName
"AuditAuthorizationPolicyChange" -Type DWORD -Value 1

# Privilege Use
Set-GPRegistryValue -Name $GpoName -Key
"HKLM\Software\Policies\Microsoft\windows\EventLog\Privilege Use" -ValueName
"AuditSensitivePrivilegeUse" -Type DWORD -Value 3

$gpoKeyESystem = "HKLM\Software\Policies\Microsoft\windows\EventLog\System"
# System
Set-GPRegistryValue -Name $GpoName -Key $gpoKeyESystem -ValueName "AuditIPsecDriver" -
Type DWORD -Value 3
Set-GPRegistryValue -Name $GpoName -Key $gpoKeyESystem -ValueName
"AuditOtherSystemEvents" -Type DWORD -Value 3
Set-GPRegistryValue -Name $GpoName -Key $gpoKeyESystem -ValueName
"AuditSecurityStateChange" -Type DWORD -Value 1
Set-GPRegistryValue -Name $GpoName -Key $gpoKeyESystem -ValueName
"AuditSecuritySystemExtension" -Type DWORD -Value 3
Set-GPRegistryValue -Name $GpoName -Key $gpoKeyESystem -ValueName
"AuditSystemIntegrity" -Type DWORD -Value 3

New-GPLink -Name $GpoName -Target $gpoPath -Enforced Yes
Write-Host "GPO '$GpoName' успішно прив'язано до '$gpoPath'." -ForegroundColor Green

ExpandedBackup-GPO -GPOName $GpoName -DomainName $DomainName
GpoGenerateReport -GPOName $GpoName
Write-GPOEventLog -GPOName $GpoName -Mode Create
}
catch { Write-Host "Сталася помилка при створенні GPO '$GPOName': $_" -ForegroundColor Red
}

```

“Deny removable disk”

```

Import-Module GroupPolicy
Import-Module C:\Scripts\UserGPOModule.psm1 -Force -warningAction SilentlyContinue

$GPOName = "Deny removable disk"
$DomainName = "makaryk.net"
$Key = "HKLM\Software\Policies\Microsoft\windows\RemovableStorageDevices"

try {
    CreateGPO -GPOName $GPOName -DomainName $DomainName

    New-Item -Path "HKLM:\Software\Policies\Microsoft\windows\RemovableStorageDevices" -
Force
    Set-ItemProperty -Path
"HKLM:\Software\Policies\Microsoft\windows\RemovableStorageDevices" -Name "Deny_Read" -
Value 1
    Set-ItemProperty -Path
"HKLM:\Software\Policies\Microsoft\windows\RemovableStorageDevices" -Name "Deny_Write" -
Value 1
    Set-ItemProperty -Path
"HKLM:\Software\Policies\Microsoft\windows\RemovableStorageDevices" -Name "Deny_Execute" -
Value 1
}

```

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				122
Змн.	Арк.	№ докум.	Підпис	Дата		

```

Set-GPRegistryValue -Name $GPOName -Key
"HKLM\SYSTEM\CurrentControlSet\Services\USBSTOR" -ValueName "Start" -Type Dword -Value 4
Set-GPRegistryValue -Name $GPOName -Key $Key -ValueName "Deny_Execute" -Type Dword -
Value 1
Set-GPRegistryValue -Name $GPOName -Key $Key -ValueName "Deny_Read" -Type Dword -Value
1
Set-GPRegistryValue -Name $GPOName -Key $Key -ValueName "Deny_Write" -Type Dword -Value
1

New-GPLink -Name $GpoName -Target "OU=Branch-Comp,DC=makaryk,DC=net"

ExpandedBackup-GPO -GPOName $GpoName -DomainName $DomainName
GpoGenerateReport -GPOName $GpoName
}
catch { Write-Host "Сталася помилка при створенні GPO '$GPOName': $_" -ForegroundColor Red
}

```

“Disable Forced System Restarts”

```

Import-Module GroupPolicy
Import-Module C:\Scripts\UserGPOModule.psm1 -Force -WarningAction SilentlyContinue

$GPOName = "Disable Forced System Restarts"
$gpoPath = "DC=makaryk,DC=net"
$PolicyPath = "HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows\WindowsUpdate\AU"
$DomainName = "makaryk.net"

try {
    CreateGPO -GPOName $GPOName -DomainName $DomainName

    $GPO = Get-GPO -Name $GPOName -ErrorAction SilentlyContinue

    Set-GPRegistryValue -Name $GPOName -Key $PolicyPath -ValueName
    "NoAutoRebootWithLoggedOnUsers" -Type Dword -Value 1
    Set-GPRegistryValue -Name $GPOName -Key $PolicyPath -ValueName "EnableFeaturedSoftware"
    -Type Dword -Value 0
    Set-GPRegistryValue -Name $GPOName -Key $PolicyPath -ValueName
    "NoAUASDefaultShutdownOption" -Type Dword -Value 1

    Write-Host "Політика заборони примусових перезавантажень налаштована." -ForegroundColor
    Green
    New-GPLink -Name $GPOName -Target $gpoPath -LinkEnabled Yes

    ExpandedBackup-GPO -GPOName $GpoName -DomainName $DomainName
    GpoGenerateReport -GPOName $GpoName
    Write-GPOEventLog -GPOName $GpoName -Mode Create
}
catch { Write-Host "Сталася помилка при створенні GPO '$GPOName': $_" -ForegroundColor Red
}

```

“Disk Quotas Policy”

```

Import-Module GroupPolicy
Import-Module C:\Scripts\UserGPOModule.psm1 -Force

$DomainName = "makaryk.net"
$GpoName = "Disk Quotas Policy"
$UserOU = "OU=Branch-Comp,DC=makaryk,DC=net"
$ScriptPath = "C:\Scripts\CreateGPO.ps1"
$RegistryPath = "HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows NT\DiskQuota"

try {
    CreateGPO -GPOName $GPOName -DomainName $DomainName

    Set-GPRegistryValue -Name $GpoName -Key $RegistryPath -ValueName
    "ApplyToRemovableMedia" -Type Dword -Value 1
    Set-GPRegistryValue -Name $GpoName -Key $RegistryPath -ValueName
    "LogEventOverThreshold" -Type Dword -Value 1
    Set-GPRegistryValue -Name $GpoName -Key $RegistryPath -ValueName "LogEventOverLimit" -
    Type Dword -Value 1
    Set-GPRegistryValue -Name $GpoName -Key $RegistryPath -ValueName "Limit" -Type Dword -
    Value 8
    Set-GPRegistryValue -Name $GpoName -Key $RegistryPath -ValueName "LimitUnits" -Type
    Dword -Value 3
}

```

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				
Змн.	Арк.	№ докум.	Підпис	Дата		123

```

Set-GPRegistryValue -Name $GpoName -Key $RegistryPath -ValueName "Enable" -Type Dword -Value 1
Set-GPRegistryValue -Name $GpoName -Key $RegistryPath -ValueName "Enforce" -Type Dword -Value 1

New-GPLink -Name $GpoName -Target $UserOU
Write-Host "GPO '$GpoName' успішно створено до $UserOU!" -ForegroundColor Green
ExpandedBackup-GPO -GPOName $GpoName -DomainName $DomainName
GpoGenerateReport -GPOName $GpoName
}
catch { Write-Host "Сталася помилка при створенні GPO '$GPOName': $_" -ForegroundColor Red
}

```

“Login Message Policy”

```
Import-Module C:\Scripts\UserGPOModule.psm1 -Force
```

```

$GpoName = "Login Message Policy"
$RegistryPath = "HKLM\Software\Microsoft\windows\CurrentVersion\Policies\System"
$UserOU = "DC=makaryk,DC=net"
$ScriptPath = "C:\Scripts\CreateGPO.ps1"
$DomainName = "makaryk.net"

try {
    CreateGPO -GPOName $GPOName -DomainName $DomainName

    $Title = "*КХМ-КХМ*"
    $Message = "Повідомляємо, завтра (32.18.1517) на сервері відбудуться технічні роботи о 1:59."

    Set-GPRegistryValue -Name $GpoName -Key $RegistryPath -ValueName "LegalNoticeCaption" -Type String -Value $Title
    Set-GPRegistryValue -Name $GpoName -Key $RegistryPath -ValueName "LegalNoticeText" -Type String -Value $Message

    New-GPLink -Name $GpoName -Target $UserOU

    ExpandedBackup-GPO -GPOName $GpoName -DomainName $DomainName
    GpoGenerateReport -GPOName $GpoName
}
catch { Write-Host "Сталася помилка при створенні GPO '$GPOName': $_" -ForegroundColor Red
}

```

“PasswordPolicy”

```

Import-Module GroupPolicy
Import-Module C:\Scripts\UserGPOModule.psm1 -Force -WarningAction SilentlyContinue

$DomainName = "makaryk.net"

try {
    Set-ADDefaultDomainPasswordPolicy -Identity $DomainName -PasswordHistoryCount 3
    Set-ADDefaultDomainPasswordPolicy -Identity $DomainName -MaxPasswordAge (New-TimeSpan -Days 30)
    Set-ADDefaultDomainPasswordPolicy -Identity $DomainName -MinPasswordLength 8
    Set-ADDefaultDomainPasswordPolicy -Identity $DomainName -ComplexityEnabled $true

    Write-Host "Успішне оновлення політик паролів" -ForegroundColor Green

    Get-ADDefaultDomainPasswordPolicy
    ExpandedBackup-GPO -GPOName $GpoName -DomainName $DomainName
    GpoGenerateReport -GPOName $GpoName
    Write-GPOEventLog -GPOName $GpoName -Mode Create
}
catch { Write-Host "Сталася помилка при створенні GPO '$GPOName': $_" -ForegroundColor Red
}

```

“Restrict CMD and PowerShell Access”

```

Import-Module GroupPolicy
Import-Module C:\Scripts\UserGPOModule.psm1 -Force -WarningAction SilentlyContinue

$DomainName = "makaryk.net"
$GpoName = "Restrict CMD and PowerShell Access"
$UsersOUS = @(
    "OU=Branch-1,DC=makaryk,DC=net",

```

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				
Змн.	Арк.	№ докум.	Підпис	Дата		124

```

        "OU=Branch-2,DC=makaryk,DC=net",
        "OU=Branch-3,DC=makaryk,DC=net"
    )

    try {
        CreateGPO -GPOName $GPOName -DomainName $DomainName
        New-GPO -Name $GpoName -ErrorAction SilentlyContinue | Out-Null

        Set-GPRegistryValue -Name $GpoName -Key
        "HKCU\Software\Policies\Microsoft\Windows\System" `
        -ValueName "DisableCMD" -Type Dword -Value 2
        Set-GPRegistryValue -Name $GpoName -Key
        "HKCU\SOFTWARE\Policies\Microsoft\PowerShellCore" `
        -ValueName "ExecutionPolicy" -Type String -Value "Restricted"
        Set-GPRegistryValue -Name $GpoName -Key "HKCU\SOFTWARE\Policies\Microsoft\PowerShell" `
        -ValueName "ExecutionPolicy" -Type String -Value "Restricted"

        foreach ($OU in $UsersOUs) {
            New-GPLink -Name $GpoName -Target $OU
            Write-Host "GPO '$GpoName' успішно створено та прив'язано до домену!" -
            ForegroundColor Green
        }
        ExpandedBackup-GPO -GPOName $GpoName -DomainName $DomainName
        GpoGenerateReport -GPOName $GpoName
        Write-GPOEventLog -GPOName $GpoName -Mode Create
    }
    catch { Write-Host "Сталася помилка при створенні GPO '$GPOName': $_" -ForegroundColor Red
    }
}

```

“Restrict registry editing tools and control panel”

```

Import-Module GroupPolicy
Import-Module C:\Scripts\UserGPOModule.psm1 -Force

$DomainName = "makaryk.net"
$GpoName = "Restrict registry editing tools and control panel"
$UsersOUs = @(
    "OU=Branch-1,DC=makaryk,DC=net",
    "OU=Branch-2,DC=makaryk,DC=net",
    "OU=Branch-3,DC=makaryk,DC=net"
)

try {
    CreateGPO -GPOName $GPOName -DomainName $DomainName
    New-GPO -Name $GpoName -ErrorAction SilentlyContinue | Out-Null

    Set-GPRegistryValue -Name $GpoName -Key
    "HKCU\Software\Microsoft\Windows\CurrentVersion\Policies\System" `
    -ValueName "disableregistrytools" -Type Dword -Value 2

    Set-GPRegistryValue -Name $GpoName -Key
    "HKCU\Software\Microsoft\Windows\CurrentVersion\Policies\Explorer" `
    -ValueName "NoControlPanel" -Type String -Value "Restricted"

    foreach ($OU in $UsersOUs) {
        New-GPLink -Name $GpoName -Target $OU
        Write-Host "GPO '$GpoName' успішно створено та прив'язано до домену!" -
        ForegroundColor Green
    }
    ExpandedBackup-GPO -GPOName $GpoName -DomainName $DomainName
    GpoGenerateReport -GPOName $GpoName
    Write-GPOEventLog -GPOName $GpoName -Mode Create
}
catch { Write-Host "Сталася помилка при створенні GPO '$GPOName': $_" -ForegroundColor Red
}

```

“Restrict Software Installation”

```

Import-Module GroupPolicy
Import-Module C:\Scripts\UserGPOModule.psm1 -Force -warningAction SilentlyContinue

$DomainName = "makaryk.net"
$GpoName = "Restrict Software Installation"
$UserOU = "OU=Branch-Comp,DC=makaryk,DC=net"
$ScriptPath = "C:\Scripts\CreateGPO.ps1"

```

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				125
Змн.	Арк.	№ докум.	Підпис	Дата		

```

$RegistryPath = "HKLM\Software\Policies\Microsoft\Windows\Installer"
try {
    CreateGPO -GPOName $GPOName -DomainName $DomainName
    Set-GPRegistryValue -Name $GpoName -key $RegistryPath -ValueName "DisableUserInstalls"
-Type Dword -Value 1
    New-GPLink -Name $GpoName -Target $UserOU
    Write-Host "GPO '$GpoName' успішно створено до $UserOU!" -ForegroundColor Green
    ExpandedBackup-GPO -GPOName $GpoName -DomainName $DomainName
    GpoGenerateReport -GPOName $GpoName
    Write-GPOEventLog -GPOName $GpoName -Mode Create
}
catch { Write-Host "Сталася помилка при створенні GPO '$GPOName': $_" -ForegroundColor Red
}

```

“Scheduled Scan for Windows Defender”

```

Import-Module GroupPolicy
Import-Module C:\Scripts\UserGPOModule.psm1 -Force -warningAction silentlyContinue
$OU = "OU=Branch-Comp,DC=makaryk,DC=net"
$DomainName = "makaryk.net"
$GPOName = "Scheduled Scan for windows Defender"
$RegistryPath = "HKLM\Software\Policies\Microsoft\Windows Defender\Scan"
try {
    CreateGPO -GPOName $GPOName -DomainName $DomainName
    Set-GPRegistryValue -Name $GPOName -key $RegistryPath -ValueName "ScheduleDay" -Type
Dword -Value 1
    Set-GPRegistryValue -Name $GPOName -key $RegistryPath -ValueName "ScheduleTime" -Type
Dword -Value 240
    New-GPLink -Name $GPOName -Target $OU
    Write-Host "GPO '$GPOName' успішно прив'язано до $OU." -ForegroundColor Green
    ExpandedBackup-GPO -GPOName $GpoName -DomainName $DomainName
    GpoGenerateReport -GPOName $GpoName
    Write-GPOEventLog -GPOName $GpoName -Mode Create
}
catch { Write-Host "Сталася помилка при створенні GPO '$GPOName': $_" -ForegroundColor Red
}

```

“Screen Lock Policy”

```

Import-Module GroupPolicy
Import-Module C:\Scripts\UserGPOModule.psm1 -Force -warningAction silentlyContinue
$GPOName = "Screen Lock Policy"
$DomainName = "makaryk.net"
$KeyPath = "HKCU\Software\Policies\Microsoft\Windows\Control Panel\Desktop"
try {
    CreateGPO -GPOName $GPOName -DomainName $DomainName
    $ScreenSaverPath = "C:\Windows\System32\scrnsave.scr"
    Set-GPRegistryValue -Name $GPOName -key $KeyPath -ValueName "SCRNSAVE.EXE" -Type String
-Value $ScreenSaverPath
    Set-GPRegistryValue -Name $GPOName -key $KeyPath -ValueName "ScreenSaveTimeout" -Type
String -Value "60"
    Set-GPRegistryValue -Name $GPOName -key $KeyPath -ValueName "ScreenSaverIsSecure" -Type
String -Value "1"
    New-GPLink -Name $GpoName -Target "DC=makaryk,DC=net"
    ExpandedBackup-GPO -GPOName $GpoName -DomainName $DomainName
    GpoGenerateReport -GPOName $GpoName
    Write-GPOEventLog -GPOName $GpoName -Mode Create
}
catch { Write-Host "Сталася помилка при створенні GPO '$GPOName': $_" -ForegroundColor Red
}

```

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				126
Змн.	Арк.	№ докум.	Підпис	Дата		

“Delete-Backup”

```

Import-Module GroupPolicy
Import-Module C:\Scripts\UserGPOModule.psm1 -Force -WarningAction SilentlyContinue

$BackupBasePath = "C:\GPO_Backups"
$LogFile = "$BackupBasePath\BackupLog.txt"

if (-not (Test-Path $BackupBasePath)) {
    New-Item -ItemType Directory -Path $BackupBasePath | Out-Null
}

function Show-Menu {
    Write-Host "Σ(つ °Д °;)つ--- Меню ---" -ForegroundColor Cyan
    Write-Host "1. Резервувати GPO" -ForegroundColor Cyan
    Write-Host "2. Видалити GPO" -ForegroundColor Cyan
    Write-Host "3. Відновити GPO" -ForegroundColor Cyan
    Write-Host "4. Вийти" -ForegroundColor Cyan
}

function Backup-GPOs {
    $AllGPOs = Get-GPO -All | Sort-Object DisplayName
    for ($i = 0; $i -lt $AllGPOs.Count; $i++) {
        Write-Host "$i. $($AllGPOs[$i].DisplayName)"
    }

    $SelectedGPOs = Input-Index -ImportantList $AllGPOs
    $Date = Get-Date -Format "yyyyMMdd_HH:mm:ss"

    foreach ($GPO in $SelectedGPOs) {
        $GPOName = $GPO.DisplayName
        $BackupPath = "$BackupBasePath\$GPOName\$Date"

        if (-not (Test-Path $BackupPath)) {
            New-Item -ItemType Directory -Path $BackupPath -Force | Out-Null
        }

        Backup-GPO -Name $GPOName -Path $BackupPath -Domain "makaryk.net"
        Write-Host "GPO $GPOName успішно резервовано у $BackupPath." -ForegroundColor Green
        Add-Content -Path $LogFile -Value "$($Get-Date): Резервна копія GPO '$GPOName' створена у '$BackupPath'."
    }
}

function Remove-GPOs {
    $AllGPOs = Get-GPO -All | Sort-Object DisplayName
    for ($i = 0; $i -lt $AllGPOs.Count; $i++) {
        Write-Host "$i. $($AllGPOs[$i].DisplayName)"
    }

    $SelectedGPOs = Input-Index -ImportantList $AllGPOs
    foreach ($GPO in $SelectedGPOs) {
        Remove-GPO -Name $GPO.DisplayName -Confirm:$false
        Write-Host "GPO '$($GPO.DisplayName)' успішно видалено." -ForegroundColor Green
        Write-GPOEventLog -GPOName $GPO -Mode Delete
    }
}

function Restore-GPOFromBackup {
    $GPOFolders = Get-ChildItem -Path $BackupBasePath -Directory | Sort-Object Name
    if ($GPOFolders.Count -eq 0) {
        Write-Host "Немає жодної резервної копії для відновлення." -ForegroundColor Yellow
        return
    }

    Write-Host "Оберіть політику для відновлення:"
    for ($i = 0; $i -lt $GPOFolders.Count; $i++) {
        Write-Host "$i. $($GPOFolders[$i].Name)"
    }

    $SelectedFolder = (Input-Index -ImportantList $GPOFolders)[0]
    $GPOName = $SelectedFolder.Name
    $FullPath = Join-Path $BackupBasePath $GPOName

    $Backups = Get-ChildItem -Path $FullPath -Directory | Sort-Object Name
    if ($Backups.Count -eq 0) {

```

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				
Змн.	Арк.	№ докум.	Підпис	Дата		127

```

        Write-Host "Для '$GPOName' не знайдено жодної резервної копії." -ForegroundColor
Red
        return
    }

    Write-Host "Оберіть резервну копію для відновлення:"
    for ($i = 0; $i -lt $Backups.Count; $i++) { Write-Host "$i. $($Backups[$i].Name)" }

    $SelectedBackup = (Input-Index -ImportantList $Backups)[0]
    $BackupPath = $SelectedBackup.FullName
    Restore-GPO -Name $GPOName -Domain "makaryk.net" -Path $BackupPath
    Write-Host "GPO $GPOName успішно відновлено з резервної копії: '$BackupPath'." -
ForegroundColor Green
    Write-GPOEventLog -GPOName $GPOName -Mode Restore
}

while ($true) {
    Show-Menu
    $choice = Read-Host "Введіть номер дії"

    switch ($choice) {
        "1" { Backup-GPOs }
        "2" { Remove-GPOs }
        "3" { Restore-GPOFromBackup }
        "4" {
            Write-Host "Вихід із програми." -ForegroundColor Cyan
            break
        }
        default { Write-Host "Невірний вибір, спробуйте ще раз." -ForegroundColor Red }
    }
}

```

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				
Змн.	Арк.	№ докум.	Підпис	Дата		128

“EditGPO”

```

Import-Module C:\Scripts\UserGPOModule.psm1 -Force -DisableNameChecking
Import-Module GroupPolicy

# Функція для відкриття скрипту в PowerShell ISE
function EditScriptInISE {
    param (
        [string]$ScriptsFolder
    )

    $scripts = Get-ChildItem -Path $ScriptsFolder -Filter *.ps1 | Select-Object -
ExpandProperty Name

    if ($scripts.Count -eq 0) {
        Write-Host "У вибраній папці немає скриптів .ps1." -ForegroundColor Yellow
        return
    }

    $scripts | ForEach-Object { Write-Host "$($_)" }
    $scriptName = Read-Host "Введіть назву скрипту для редагування (включаючи розширення
.ps1)"

    $scriptPath = Join-Path -Path $ScriptsFolder -ChildPath $scriptName

    if (Test-Path $scriptPath) {
        ise $scriptPath
    } else {
        Write-Host "Скрипт не знайдено за вказаною назвою: $scriptName" -ForegroundColor
Red
    }
    PostEditMenu -scriptPath $scriptPath
}

function PostEditMenu {
    param (
        [string]$ScriptPath
    )

    Write-Host "( ↵ ↵ ) ~> Якщо скрипт з GPO було відредаговано " -ForegroundColor Cyan
    Write-Host "1. Видалити минулу GPO і виконати оновлений скрипт"
    Write-Host "0. Вийти в головне меню"
    $choice = Read-Host "Ваш вибір"

    switch ($choice) {
        "1" {
            $GPOName = Get-GPOName -Prompt "Введіть ім'я GPO для видалення"
            if ($GPOName) {
                Remove-GPO -Name $GPOName -Confirm:$false
                Write-Host "GPO '$GPOName' успішно видалено." -ForegroundColor Green
            }
            if (Test-Path $ScriptPath) {
                Write-Host "Виконання скрипта: $ScriptPath" -ForegroundColor Cyan
                & $ScriptPath
                ExpandedBackup-GPO -GPOName $GPOName -DomainName "makaryk.net"
                GpoGenerateReport -GPOName $GPOName
                CompareLastAndNewGPO -GPOName $GPOName
                Write-GPOEventLog -GPOName $GPOName -Mode Edit
                Write-Host "Скрипт виконано успішно." -ForegroundColor Green
            } else {
                Write-Host "Скрипт не знайдено за вказаним шляхом: $ScriptPath" -
ForegroundColor Red
            }
        }
        "0" { Main }
        default { Write-Host "неправильний вибір. Спробуйте ще раз" -ForegroundColor Red }
    }
}

function FindGPOSettings {
    $AllGPOinDomain = Get-GPO -All -Domain "makaryk.net"
    Write-Host "Усі налаштовані політики на домені:" -ForegroundColor Cyan
    $AllGPOinDomain | ForEach-Object { Write-Host "$($_.DisplayName)" }
}

```

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				129
Змн.	Арк.	№ докум.	Підпис	Дата		

```

$GPOName = Get-GPOName -Prompt "Введіть назву політики, яку потрібно редагувати"
if (-not $GPOName) { return Main }

Get-GPAllRegistryValues -Name $GPOName
EditGPO -GPOName $GPOName
Write-Host "$GPOName - в функції знаходження" -ForegroundColor Cyan
}

function NormalizeRegistryPath {
    param (
        [string]$RegistryPath
    )
    $normalizedPath = $RegistryPath -replace '^HKKEY_CURRENT_USER\\', 'HKCU:\' `
        -replace '^HKCU\\', 'HKCU:\' `
        -replace '^HKKEY_LOCAL_MACHINE\\', 'HKLM:\' `
        -replace '^HKLM\\', 'HKLM:\'

    return $normalizedPath
}

function EditGPO {
    param (
        [string]$GPOName
    )
    Write-Host "$GPOName - в функції редагування" -ForegroundColor Cyan
    $GPOFullKey = Read-Host "Введіть повний шлях ключа реєстру для редагування"
    $GPOValueName = Read-Host "Введіть ім'я вартості(?)"
    $GPOValue = Read-Host "Введіть нове значення ГП"

    $registryPath = NormalizeRegistryPath -RegistryPath $GPOFullKey

    try {
        $registryKey = Get-Item -Path $registryPath -ErrorAction Stop

        if ($registryKey.GetValueNames() -contains $GPOValueName) {
            $type = $registryKey.GetValueKind($GPOValueName)
        }
        else { Write-Host "Значення '$GPOValueName' не знайдено в ключі" -ForegroundColor Red }

    } catch { Write-Host "Ключ реєстру не знайдено: $registryPath" -ForegroundColor Red }

    if (-not (Set-GPRegistryValue -Name $GPOName -Key $GPOFullKey -ValueName $GPOValueName
    -Value $GPOValue -Type $type -ErrorAction SilentlyContinue)) {
        Write-Host "Не вдалося оновити групову політику" -ForegroundColor Red

    } else {
        Write-Host "Групову політику успішно оновлено" -ForegroundColor Green
        Get-GPO -Name $GPOName
        ExpandedBackup-GPO -GPOName $GPOName -DomainName "makaryk.net"
        GpoGenerateReport -GPOName $GPOName
        CompareLastAndNewGPO -GPOName $GPOName
        Write-GPOEventLog -GPOName $GPOName -Mode Edit
    }
    Main
}

function CompareLastAndNewGPO {
    param (
        [string]$GPOName
    )
    $xmlFolder = Join-Path -Path "C:\GPO_Report\$GPOName" -ChildPath "xml"
    $timestamp = Get-Date -Format "yyyy-MM-dd_HH:mm:ss"

    if (-not (Test-Path $xmlFolder)) {
        Write-Host "Каталог $xmlFolder не існує." -ForegroundColor Red
        return
    }

    $lastTwoXmlReports = Get-ChildItem -Path $xmlFolder -Filter *.xml |
        Sort-Object LastWriteTime -Descending |
        Select-Object -First 2

    if ($lastTwoXmlReports.Count -lt 2) {
        Write-Host "Недостатньо XML-файлів для порівняння." -ForegroundColor Yellow
        return
    }
}

```

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				
Змн.	Арк.	№ докум.	Підпис	Дата		130

```

    $Comparison = Compare-Object -ReferenceObject (Get-Content
$lastTwoXmlReports[1].FullName) -DifferenceObject (Get-Content
$lastTwoXmlReports[0].FullName)

    $comparisonFolder = "C:\GPO_Report\Comparison"
    if (-not (Test-Path $comparisonFolder)) { New-Item -Path $comparisonFolder -ItemType
Directory | Out-Null }

    $ComparisonFilePath = Join-Path -Path $comparisonFolder -ChildPath
"$GPOName\_$_timestamp.txt"
    $Comparison | Out-File -FilePath $ComparisonFilePath -Encoding UTF8
    Write-Host "Результати порівняння збережено в $ComparisonFilePath" -ForegroundColor
Green
}

function Main {
    Write-Host "о( " ^ ▽ ^ " )о Меню редагування групових політик " -ForegroundColor Cyan
    Write-Host "1. Відкрити скрипт політики в PowerShell ISE"
    Write-Host "2. Переглянути і редагувати параметри GPO"
    Write-Host "0. Вийти"

    $choice = Read-Host "Введіть ваш вибір"

    switch ($choice) {
        "1" {
            $scriptsFolder = "C:\Scripts\GPO_Scripts"
            EditScriptInISE -ScriptsFolder $scriptsFolder
        }
        "2" { FindGPOSettings }
        "0" {
            Write-Host "Вихід із програми" -ForegroundColor Green
            break
        }
        default { Write-Host "неправильний вибір. Спробуйте ще раз" -ForegroundColor Red }
    }
}

Main

```

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				
Змн.	Арк.	№ докум.	Підпис	Дата		131

“EditLinkGPO”

```

Import-Module C:\Scripts\UserGPOModule.psm1 -Force

function Get-UnlinkedGPOS {
    $AllGPOS = Get-GPO -All
    $LinkedGPOS = @()
    $Targets = Get-AllGpoTargets

    foreach ($target in $Targets) {
        try {
            $Inheritance = Get-GPInheritance -Target $target.DistinguishedName
            foreach ($Link in $Inheritance.GpoLinks) {
                if ($Link.DisplayName -and $LinkedGPOS -notcontains $Link.DisplayName) {
                    $LinkedGPOS += $Link.DisplayName
                }
            }
        } catch { Write-Warning "Помилка при обробці $($target.DistinguishedName): $_" }
    }
    $UnlinkedGPOS = $AllGPOS | where-Object { $LinkedGPOS -notcontains $_.DisplayName }
    return $UnlinkedGPOS
}

function Get-AllGpoTargets {
    $domainDN = (Get-ADDomain).DistinguishedName
    $Targets = @(
        [PSCustomObject]@{ Name = "Домен"; DistinguishedName = $domainDN }
    )
    $Targets += Get-ADOrganizationalUnit -Filter * | select-Object Name, DistinguishedName
    return $Targets
}

function Get-YesNoInput {
    param (
        [string]$Prompt = "Введіть 'Так' або 'Ні'"
    )

    while ($true) {
        $inputValue = Read-Host $Prompt

        switch ($inputValue.ToLower()) {
            "так" { return "Yes" }
            "ні" { return "No" }
            default { Write-Host "Некоректне значення. Спробуйте ще раз." -ForegroundColor Yellow }
        }
    }
}

function MassCreate-GpoLinks {
    param (
        [Parameter(Mandatory = $true)]
        [Microsoft.GroupPolicy.GPO[]]$GPOList,

        [Parameter(Mandatory = $true)]
        [Object[]]$OUList
    )

    Write-Host "( *▽▽) ---Виберіть метод прив'язування---" -ForegroundColor Cyan
    Write-Host "1. Задана на усі лінки один параметр"
    Write-Host "2. Створити кожен лінк окремо"
    $choice = Read-Host "Введіть ваш вибір"

    switch ($choice) {
        "1" {
            $linkEnabled = Get-YesNoInput -Prompt "Застосувати політику? (Так/Ні)"
            $enforced = Get-YesNoInput -Prompt "Примусово застосувати політику? (Так/Ні)"
            foreach ($GPO in $GPOList) {
                foreach ($OU in $OUList) {
                    try {
                        New-GPLink -Name $GPO.DisplayName -Target $OU.DistinguishedName -
                        LinkEnabled $linkEnabled -Enforced $enforced
                        Write-Host "Створено нове прив'язування $($GPO.DisplayName) до
                        $($OU.Name)" -ForegroundColor Green
                    }
                }
            }
        }
    }
}

```

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				132
Змн.	Арк.	№ докум.	Підпис	Дата		

```

"makaryk.net" ExpandedBackup-GPO -GPOName $($GPO.DisplayName) -DomainName
                GpoGenerateReport -GPOName $($GPO.DisplayName)
            }
            catch { Write-Host "Помилка при створенні прив'язування
$($GPO.DisplayName) до $($OU.Name) : $_" -ForegroundColor Red}
        }
    }
    "2" {
        foreach ($GPO in $GPOList) {
            foreach ($OU in $OUList) {
                try{
                    $linkEnabled = Get-YesNoInput -Prompt "Застосувати політику
'$($GPO.DisplayName)' на '$($OU.Name)'? (Так/Ні)"
                    $enforced = Get-YesNoInput -Prompt "Примусово застосувати полі-
тику '$($GPO.DisplayName)' на '$($OU.Name)'? (Так/Ні)"
                    New-GPLink -Name $GPO.DisplayName -Target $OU.DistinguishedName
                    -LinkEnabled $linkEnabled -Enforced $enforced
                    Write-Host "Створено нове прив'язування $($GPO.DisplayName) до
$($OU.Name)" -ForegroundColor Green
                    ExpandedBackup-GPO -GPOName $($GPO.DisplayName) -DomainName
                    "makaryk.net"
                    GpoGenerateReport -GPOName $($GPO.DisplayName)
                }
                catch { Write-Host "Помилка при створенні прив'язування
'$($GPO.DisplayName)' до '$($OU.Name)' : $_" -ForegroundColor Red}
            }
        }
        default { Write-Host "Неправильний вибір. Спробуйте ще раз" -ForegroundColor Red }
    }
}

function Print-Select {
    param (
        [Parameter(Mandatory=$true)]
        [Object[]]$PrintAll
    )
    for ($i = 0; $i -lt $PrintAll.Count; $i++) {
        $item = $PrintAll[$i]
        $name = if ($item.PSObject.Properties["DisplayName"]) {
            $item.DisplayName
        } elseif ($item.PSObject.Properties["Name"]) { $item.Name
        } else { "Невідомий об'єкт" }
        Write-Host "$i. $name"
    }

    $selected = Input-Index -ImportantList $PrintAll
    return $selected
}

function Edit-GPOLinks {
    param (
        [Parameter(Mandatory = $true)]
        [Microsoft.GroupPolicy.GPO[]]$GPOList
    )

    $allTargets = Get-AllGpoTargets

    foreach ($GPO in $GPOList) {
        $GPOName = $GPO.DisplayName
        Write-Host "Оброблення GPO: $GPOName" -ForegroundColor Cyan

        $linkedTargets = @()

        foreach ($target in $allTargets) {
            $gpoLinks = Get-GPINheritance -Target $target.DistinguishedName | Select-Object
-ExpandProperty GpoLinks
            foreach ($link in $gpoLinks) {
                if ($link.DisplayName -eq $GPOName) {
                    $linkedTargets += $target
                    break
                }
            }
        }

        if ($linkedTargets.Count -eq 0) {

```

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				
Змн.	Арк.	№ докум.	Підпис	Дата		133

```

        Write-Host "GPO '$GPOName' не прив'язано ні до одного OU чи домену." -
ForegroundColor Yellow
        continue
    }
    elseif ($linkedTargets.Count -gt 1) {
        Write-Host "GPO '$GPOName' прив'язано до кількох об'єктів. Виберіть вручну." -
ForegroundColor Yellow
        $linkedTargets | ForEach-Object {
            Write-Host "- $($_.Name): $($_.DistinguishedName)"
        }
        continue
    }

    $targetDN = $linkedTargets[0].DistinguishedName
    Write-Host "GPO '$GPOName' пов'язано з: $targetDN" -ForegroundColor Green

    $linkEnabled = Get-YesNoInput -Prompt "Застосувати політику '$GPOName' на
'$targetDN'? (Так/Ні)"
    $enforced = Get-YesNoInput -Prompt "Примусово застосувати політику '$GPOName' на
'$targetDN'? (Так/Ні)"

    try {
        Set-GPLink -Name $GPOName -Target $targetDN -LinkEnabled $linkEnabled -Enforced
$enforced
        Write-Host "Посилання оновлено." -ForegroundColor Green
        ExpandedBackup-GPO -GPOName $GPOName -DomainName "makaryk.net"
        GpoGenerateReport -GPOName $GPOName
    }
    catch { Write-Host "Помилка при зміні посилання: $_" -ForegroundColor Red }
}

function Remove-GpoLinks {
    Write-Host "(◡ ◡) ◡ ---Видалення прив'язування GPO---" -ForegroundColor Cyan
    Write-Host "1. Видалити за GPO"
    Write-Host "2. Видалити за OU/доменом"
    $choice = Read-Host "Введіть ваш вибір"

    $allTargets = Get-AllGpoTargets

    switch ($choice) {
        "1" {
            $linkedGPOs = Get-GPO -All | where-Object { $_.DisplayName -notin (Get-
UnlinkedGPOs).DisplayName }

            if ($linkedGPOs.Count -eq 0) {
                Write-Host "Немає GPO, які прив'язані до OU або домену." -ForegroundColor
Yellow
                return
            }

            $selectedGPOs = Print-Select -PrintAll $linkedGPOs

            foreach ($GPO in $selectedGPOs) {
                foreach ($target in $allTargets) {
                    try {
                        $inheritance = Get-GPInheritance -Target $target.DistinguishedName
                        foreach ($link in $inheritance.GpoLinks) {
                            if ($link.DisplayName -eq $($GPO.DisplayName)) {
                                Remove-GPLink -Name $($GPO.DisplayName) -Target
$target.DistinguishedName
                                Write-Host "Видалено лінк $($GPO.DisplayName) з
$($target.Name)" -ForegroundColor Green
                                ExpandedBackup-GPO -GPOName $($GPO.DisplayName) -DomainName
"makaryk.net"
                                GpoGenerateReport -GPOName $($GPO.DisplayName)
                            }
                        }
                    } catch { Write-Warning "Помилка при перевірці
$($target.DistinguishedName): $_" }
                }
            }
        }
        "2" {
            $selectedTargets = Print-Select -PrintAll $allTargets

            foreach ($target in $selectedTargets) {
                try {

```

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				
Змн.	Арк.	№ докум.	Підпис	Дата		134

```

        $inheritance = Get-GPInheritance -Target $target.DistinguishedName
        if ($inheritance.GpoLinks.Count -eq 0) {
            Write-Host "$($target.Name) не має жодного лінку." -ForegroundColor Yellow
            continue
        }

        $selectedLinks = $inheritance.GpoLinks | Select-Object DisplayName
        $selectedGPOs = Print-Select -PrintAll $selectedLinks

        foreach ($GPO in $selectedGPOs) {
            Remove-GPLink -Name $GPO.DisplayName -Target
            Write-Host "Видалено $($GPO.DisplayName) з $($target.Name)" -
            ForegroundColor Green
            ExpandedBackup-GPO -GPOName $($GPO.DisplayName) -DomainName
            "makaryk.net"
            GpoGenerateReport -GPOName $($GPO.DisplayName)
        }
    } catch { Write-Warning "Помилка при обробці $($target.DistinguishedName):
    $_" }
}
}
default { Write-Host "неправильний вибір. Спробуйте ще раз." -ForegroundColor Red }
}

function Main {
    while ($true) {
        Write-Host "👤 ---Керування GPO-прив'язками---" -ForegroundColor Cyan
        Write-Host "1. Масово створити прив'язки GPO"
        Write-Host "2. Редагувати параметри прив'язки GPO"
        Write-Host "3. Видалити прив'язки GPO"
        Write-Host "4. Вихід"
        $choice = Read-Host "Введіть номер "

        switch ($choice) {
            "1" {
                $GPOList = Print-Select -PrintAll (Get-UnlinkedGPOs)
                $OUList = Print-Select -PrintAll (Get-AllGpoTargets)
                MassCreate-GpoLinks -GPOList $GPOList -OUList $OUList
            }
            "2" {
                $linkedGPOs = Get-GPO -All | Where-Object { $_.DisplayName -notin (Get-
                UnlinkedGPOs).DisplayName }
                if ($linkedGPOs.Count -eq 0) {
                    Write-Host "Немає GPO для редагування." -ForegroundColor Yellow
                } else {
                    $selected = Print-Select -PrintAll $linkedGPOs
                    Edit-GPOLinks -GPOList $selected
                }
            }
            "3" { Remove-GpoLinks }
            "4" {
                Write-Host "Вихід"
                break
            }
            default { Write-Host "неправильний вибір. Спробуйте ще раз." -ForegroundColor Red }
        }
    }
}

Main

```

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				
Змн.	Арк.	№ докум.	Підпис	Дата		135

“GPPermissionEdit”

```
Import-Module C:\Scripts\UserGPOModule.psm1 -Force
```

```
function EditPermissionMenu {
    param (
        [Parameter(Mandatory=$true)]
        [Object[]]$UserOU,
        [Parameter(Mandatory=$true)]
        [Object[]]$UsedGPO
    )

    Write-Host "о(*▽*)づ ---Оберіть тип доступу---" -ForegroundColor Cyan
    Write-Host "1. Лише перегляд GPO "
    Write-Host "2. Редагування GPO"
    Write-Host "3. Редагування, видалення, зміна безпеки доступу"
    Write-Host "4. Видалити доступ"
    $PerMenu = @{
        "1" = "GpoRead"
        "2" = "GpoEdit"
        "3" = "GpoEditDeleteModifySecurity"
        "4" = "None"
    }
    do {
        $validChoice = $true
        $choice = Read-Host "Введіть ваш вибір (1-4)"
        if ($PerMenu.ContainsKey($choice)) {
            $Permission = $PerMenu[$choice]
            foreach ($GPOName in $UsedGPO) {
                foreach ($User in $UserOU) {
                    try {
                        Set-GPPermission -Name $GPOName.DisplayName -TargetName
                        $User.SamAccountName -TargetType User -PermissionLevel $Permission
                        Write-Host "Встановлено дозвіл на $Permission на $GPOName для
                        $($User.SamAccountName)" -ForegroundColor Green
                    }
                    catch { Write-Host "Помилка встановлення дозволу для
                    '$($User.SamAccountName)' на '$GPOName': $_" -ForegroundColor Red }
                }
            }
        }
        else {
            Write-Host "Неправильний вибір. Спробуйте ще раз." -ForegroundColor Red
            $validChoice = $false
        }
    } while (-not $validChoice)

    Write-Host "Список доступних GPO" -ForegroundColor Cyan
    $AllGPO = Get-GPO -All

    for ($i = 0; $i -lt $AllGPO.Count; $i++) {
        Write-Host "$i. $($AllGPO[$i].DisplayName)"
    }

    $UsedGPO = Input-Index -ImportantList $AllGPO

    Write-Host "Задати до вибраних користувачів чи користувачів OU" -ForegroundColor Cyan
    Write-Host "1. Вибрані користувачі"
    Write-Host "2. Усі користувачі OU"

    $choice = Read-Host "Введіть ваш вибір"

    switch ($choice) {
        "1" {
            $AllUsers = Get-ADUser -Filter *
            for ($i = 0; $i -lt $AllUsers.Count; $i++) { Write-Host "$i.
            $($AllUsers[$i].SamAccountName)" }
            $UsedUsers = Input-Index -ImportantList $AllUsers
        }
        "2" {
            $AllOU = Get-ADOrganizationalUnit -Filter 'Name -like "*"'
            for ($i = 0; $i -lt $AllOU.Count; $i++) { Write-Host "$i. $($AllOU[$i].Name)" }
            $UsedAllOU = Input-Index -ImportantList $AllOU
        }
    }
}
```

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				136
Змн.	Арк.	№ докум.	Підпис	Дата		

```

$AllUsers = foreach ($OU in $UsedOU) {
$UsedUsers = Get-ADUser -Filter * -SearchBase $OU.DistinguishedName
}
default { write-host "Неправильний вибір. Спробуйте ще раз" -ForegroundColor Red }
}
EditPermissionMenu -UserOU $UsedUsers -UsedGPO $UsedGPO

```

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				
Змн.	Арк.	№ докум.	Підпис	Дата		137

“WmiFilter”

```

Add-Type -AssemblyName 'Microsoft.GroupPolicy.Management'

function New-WmiFilter {
    [CmdletBinding()]
    param (
        [Parameter(Mandatory = $true)]
        [string]$Name,
        [Parameter(Mandatory = $true)]
        [string[]]$Filter,
        [string]$Description = '',
        [string]$Server = $env:USERDNSDOMAIN
    )

    try {
        $RootDSE = Get-ADRootDSE -Server $Server
        $NamingContext = $RootDSE.DefaultNamingContext
        $wmiGuid = "{$([guid]::NewGuid())}"
        $creationDate = (Get-Date).ToUniversalTime().ToString("yyyyMMddHHmmss.ffffff-000")
        $filterString = "{0};" -f $Filter.Count
        foreach ($query in $Filter) {
            $filterString += "3;10;{0};WQL;root\CIMv2;{1};" -f $query.Length, $query
        }
        $attributes = @{
            "msWMI-Name"           = $Name
            "msWMI-Parm1"          = $Description
            "msWMI-Parm2"          = $filterString
            "msWMI-Author"         = "$env:USERNAME@$env:USERDNSDOMAIN"
            "msWMI-ID"             = $wmiGuid
            "instanceType"         = 4
            "msWMI-ChangeDate"     = $creationDate
            "msWMI-CreationDate"   = $creationDate
            "showInAdvancedViewOnly" = "TRUE"
            "distinguishedName"    =
            "CN=$wmiGuid,CN=SOM,CN=WMIPolicy,CN=System,$NamingContext"
        }
        New-ADObject -Name $wmiGuid `
            -Type "msWMI-Som" `
            -Path "CN=SOM,CN=WMIPolicy,CN=System,$NamingContext" `
            -OtherAttributes $attributes
            -Server $Server

        Write-Host " WMI-фільтр '$Name' створено успішно" -ForegroundColor Green
    }
    catch { Write-Error " Помилка створення WMI-фільтра: $_" -ForegroundColor Red }
}

function FindWmiFilter {
    [CmdletBinding()]
    Param(
        [Parameter(Mandatory = $true)]
        [string]$WmiFilterName,
        [string]$Server = $env:USERDNSDOMAIN,
        [ValidateSet("AD", "GPMC")]
        [string]$Mode = "GPMC"
    )

    try {
        if ($Mode -eq "GPMC") {
            $GPDomain = New-Object Microsoft.GroupPolicy.GPDomain
            $SearchFilter = New-Object Microsoft.GroupPolicy.GPSearchCriteria
            $allWmiFilters = $GPDomain.SearchWmiFilters($SearchFilter)
            $wmiFilter = $allWmiFilters | Where-Object Name -eq $WmiFilterName

            if ($wmiFilter) { return $wmiFilter }
            else { Write-Warning "WMI-фільтр '$WmiFilterName' не знайдено в GPMC" }
        }

        if ($Mode -eq "AD") {
            $root = Get-ADRootDSE -Server $Server
            $Properties = @(
                'msWMI-Author'
                'msWMI-ID'
                'msWMI-Name'
            )
        }
    }
}

```

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				138
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        'msWMI-Parm1'
        'msWMI-Parm2'
        'Created'
        'Modified'
    )

    if ($WMIFilterName -eq "AllWMI")
    {
        $adFilters = Get-ADObject -Filter "ObjectClass -eq 'msWMI-Som'" `
            -SearchBase "CN=SOM,CN=WMIPolicy,CN=System,($root.DefaultNamingContext)" `
            -SearchScope OneLevel `
            -Properties $Properties

        $adFilters | Select-Object `
            @{Name = 'Name'; Expression = { $_.'msWMI-Name' } },
            @{Name = 'Description'; Expression = { $_.'msWMI-Parm2' } },
            @{Name = 'Query'; Expression = { $_.'msWMI-Parm1' } },
            @{Name = 'Author'; Expression = { $_.'msWMI-Author' } },
            @{Name = 'Created'; Expression = { $_.'Created' } },
            @{Name = 'Modified'; Expression = { $_.'Modified' } },
            DistinguishedName
        return
    }

    $adFilter = Get-ADObject -Filter "ObjectClass -eq 'msWMI-Som' -and msWMI-Name -eq '$WMIFilterName'" `
        -SearchBase "CN=SOM,CN=WMIPolicy,CN=System,($root.DefaultNamingContext)" `
        -SearchScope OneLevel `
        -Properties $Properties

    if ($adFilter) { return $adFilter }
    else { Write-Warning "WMI-фільтр '$WMIFilterName' не знайдено в AD" }
}

} catch { Write-Error "Помилка при пошуку WMI-фільтра ($Mode): $_" }

function Set-GPOWmiFilter
{
    [cmdletbinding()]
    Param(
        [string]$WMIFilterName,
        [string]$GPOName
    )

    BEGIN
    {
        $f = $MyInvocation.InvocationName
    }
    PROCESS
    {
        $WMIfilter = FindWmiFilter -WMIFilterName $WMIFilterName -Mode GPMC
        $GPOObject = $null

        if($GPOName)
        {
            $GPOObject = Get-GPO -Name $GPOName
            if(-not $GPOObject) { Write-Host "Не знайдено GPO за назвою '$GPOName'" -ErrorAction Stop -ForegroundColor Red }
        }

        Write-Host "$f - Застосовано WMI-фільтрацію '$($WMIfilter.Name)' до GPO '$($GPOObject.DisplayName)'" -ForegroundColor Green

        try { $GPOObject.WmiFilter = $WMIfilter }
        catch { Write-Error "Помилка при прив'язці WMI-фільтра до GPO: $_" }
    }
}

function Remove-WmiFilter {
    [CmdletBinding(SupportsShouldProcess = $true, ConfirmImpact = 'High')]
    param (
        [Parameter(Mandatory = $true)]
        [string]$WMIFilterName,
        [string]$Server = $env:USERDNSDOMAIN
    )
}

```

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				139
Змн.	Арк.	№ докум.	Підпис	Дата		

```

    )
    try {
        $adFilter = FindWmiFilter -WmiFilterName $WmiFilterName -Mode AD
        if ($PSCmdlet.ShouldProcess($adFilter.DistinguishedName, "Видалити WMI-фільтр")) {
            Remove-ADObject -Identity $adFilter.DistinguishedName -Confirm:$false -Server
$Server
            Write-Host "Видалено: $WmiFilterName" -ForegroundColor Green
        }
    } catch { Write-Error "Помилка: $_" }
}

function Set-WmiFilter {
    [CmdletBinding()]
    param (
        [Parameter(Mandatory = $true)]
        [string]$WmiFilterName,
        [string]$Filter,
        [string]$Description = '',
        [string]$Server = $env:USERDNSDOMAIN
    )
    try{
        $adFilter = FindWmiFilter -WmiFilterName $WmiFilterName -Mode AD
        $newParams = @{}
        if ($Filter) {
            $filterString = "{0};" -f $Filter.Count
            foreach ($query in $Filter) {
                $filterString += "3;10;{0};WQL;root\CIMv2;{1};" -f $query.Length, $query
            }
            $newParams["msWMI-Parm2"] = $filterString
        }
        # Якщо задано Description – оновлюємо msWMI-Parm1
        if ($Description) {
            $newParams["msWMI-Parm1"] = $Description
        }
        if ($newParams.Count -eq 0) {
            Write-Warning "Нічого не задано для оновлення."
            return
        }
        if ($PSCmdlet.ShouldProcess($adFilter.DistinguishedName, "Оновити WMI-фільтр")) {
            Set-ADObject -Identity $adFilter.DistinguishedName -Replace $newParams -Server
$Server
            Write-Host "Оновлено фільтр '$WmiFilterName'" -ForegroundColor Green
        }
    }
    catch { Write-Error "Помилка: $_" }
}

```

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				
Змн.	Арк.	№ докум.	Підпис	Дата		140

“WorkWMI”

```

Add-Type -AssemblyName 'Microsoft.GroupPolicy.Management'

Import-Module C:\Scripts\WmiFilter.psm1 -Force
Import-Module C:\Scripts\UserGPOModule.psm1

function WmiFilterNoteList {
    $WmiTitles = Get-Content "C:\Scripts\Scripts\WMIFiltes-list\OS-list.txt"
    $WmiFilters = Get-Content "C:\Scripts\Scripts\WMIFiltes-list\Filter-list.txt"

    if ($WmiTitles.Count -ne $WmiFilters.Count) {
        Write-Error "Кількість заголовків і фільтрів не збігається." -ForegroundColor Red
        return $null
    }

    Write-Host "_(з)⌵_ Виберіть WMI-фільтр для застосування:\n" -ForegroundColor Cyan
    for ($i = 0; $i -lt $WmiTitles.Count; $i++) {
        Write-Host "$($i + 1)) $($WmiTitles[$i])"
    }

    $choice = Read-Host "`nВведіть номер варіанту"

    [int]$iChoice = 0
    if ([int]::TryParse($choice, [ref]$iChoice) -and $iChoice -ge 1 -and $iChoice -le
    $WmiTitles.Count) {
        $selectedFilter = $WmiFilters[$iChoice - 1]
        Write-Host "Вибраний фільтр: $selectedFilter" -ForegroundColor Cyan
        return $selectedFilter
    } else {
        Write-Host "Невірний вибір." -ForegroundColor Red
        return $null
    }
}

function WmiRegular {
    while ($true) {
        $WmiFilter = Read-Host "Введіть новий WMI-фільтр або 0 для виходу"
        if ($WmiFilter -eq '0'){
            Write-Host "Вихід із створення фільтру"
        }
        if ($WmiFilter -match '^s*select\s+.\s+\s+from\s+.\s+\s+where\s+\w+') {
            return $WmiFilter
            break
        } else { Write-Host "Помилка у записі WMI-запиту"}
    }
}

function AllWmiFilters {
    $AllWmiFilters = FindWmiFilter -WmiFilterName 'AllWMI' -Mode AD
    Write-Host "Усі налаштовані WMI-фільтри на домені:" -ForegroundColor Cyan
    $AllWmiFilters | ForEach-Object { Write-Host "$($_.Name)" }
    $WmiFilterName = Read-Host "Введіть назву WMI-фільтру для застосування"
    return $WmiFilterName
}

function ChoiceWmiFilter {
    Write-Host "1: Задати власний фільтр"
    Write-Host "2: Вибрати зі списку (не повний)"
    $choice = Read-Host "Введіть ваш вибір"
    switch ($choice) {
        "1" { $WmiFilter = WmiRegular }
        "2" { $WmiFilter = WmiFilterNoteList }
        default { Write-Host "Неправильний вибір. Спробуйте ще раз" -ForegroundColor Red }
    }
    return $WmiFilter
}

function Main {
    Write-Host "Скрипт менеджменту WMI-фільтрів (ωο)" -ForegroundColor Cyan

```

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				141
Змн.	Арк.	№ докум.	Підпис	Дата		

```

Write-Host "1. Створення нового WMI-фільтру"
Write-Host "2. Прив'язання WMI-фільтру до GPO"
Write-Host "3. Редагування WMI-фільтру"
Write-Host "4. Видалення WMI-фільтру"
Write-Host "0. Вийти"

$choice = Read-Host "Введіть ваш вибір"

switch ($choice) {
    "1" {
        $WMIFilterName = Read-Host "Введіть назву нового WMI-фільтра"
        $WMIFilter = ChoiceWMIFilter
        $WMIDescription = Read-Host "Введіть опис/коментар до WMI-фільтру"
        New-WmiFilter -Name $WMIFilterName -Filter $WMIFilter -Description
$WMIDescription
        FindWmiFilter -WMIFilterName $WMIFilterName -Mode AD
    }
    "2" {
        $WMIFilterName = AllWmiFilters
        $AllGPOinDomain = Get-GPO -All -Domain "makaryk.net"
        Write-Host "Усі налаштовані політики на домені:" -ForegroundColor Cyan
        $AllGPOinDomain | ForEach-Object { Write-Host "$($_.DisplayName)" }
        $GPOName = Read-Host "Введіть назву WMI-фільтру для застосування"
        Set-GPOWmiFilter -WMIFilterName $WMIFilterName -GPOName $GPOName
        ExpandedBackup-GPO -GPOName $GPOName -DomainName "makaryk.net"
        GpoGenerateReport -GPOName $GPOName
    }
    "3" {
        $WMIFilterName = AllWmiFilters
        $WMIFilter = ChoiceWMIFilter
        if ($WMIFilter -eq $null) {
            Write-Host "Фільтр не вибрано. Спробуйте ще раз." -ForegroundColor Red
            return
        }
        $WMIDescription = Read-Host "Введіть НОВИЙ опис/коментар до WMI-фільтру"
        Set-WmiFilter -WMIFilterName $WMIFilterName -Filter $WMIFilter -Description
$WMIDescription
    }
    "4" {
        $WMIFilterName = AllWmiFilters
        Remove-WmiFilter -WMIFilterName $WMIFilterName
    }
    "0" {
        Write-Host "Вихід із програми" -ForegroundColor Green
        break
    }
    default {
        Write-Host "Неправильний вибір. Спробуйте ще раз" -ForegroundColor Red
    }
}
Main
}

```

Main

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				
Змн.	Арк.	№ докум.	Підпис	Дата		142

“Report Generation”

```

Import-Module C:\Scripts\UserGPOModule.psm1 -Force

function Select-FromList {
    param (
        [Parameter(Mandatory = $true)]
        [Array]$Items,
        [string]$Property
    )

    $Items | ForEach-Object -Begin { $i = 0 } -Process {
        if ($Property) { Write-Host "$i $($_. $Property)"
        } else { Write-Host "$i $_" }
        $i++
    }
    return Input-Index -ImportantList $Items
}

function Export-GPOReport {
    param (
        [string]$GPOName = '',
        [string]$ReportPath = "C:\GPO_Report"
    )

    if (-not (Test-Path $ReportPath)) {
        New-Item -ItemType Directory -Force -Path $ReportPath | Out-Null
    }

    $GPOs = if ($GPOName) { Get-GPO -Name $GPOName }
    else { Get-GPO -All }

    foreach ($GPO in $GPOs) {
        try { GpoGenerateReport -GPOName $GPO.DisplayName -GPOReportPath $ReportPath
        } catch { Write-Host "Помилка створення звіту для $($GPO.DisplayName): $_" -
        ForegroundColor Red }
    }
}

function GpoGenerateReport {
    [CmdletBinding()]
    param (
        [string]$GPOName,
        [string]$GPOReportPath = "C:\GPO_Report"
    )

    $timestamp = Get-Date -Format "yyyy-MM-dd_HH:mm:ss"
    $ReportFolder = Join-Path $GPOReportPath $GPOName
    $formats = @("html", "xml")

    foreach ($format in $formats) {
        $folderPath = Join-Path $ReportFolder $format
        if (-not (Test-Path $folderPath)) {
            New-Item -ItemType Directory -Path $folderPath -Force | Out-Null
        }
    }

    try {
        $htmlPath = Join-Path -Path (Join-Path $ReportFolder "html") -ChildPath
        "$GPOName_$timestamp.html"
        $xmlPath = Join-Path -Path (Join-Path $ReportFolder "xml") -ChildPath
        "$GPOName_$timestamp.xml"
        Get-GPOReport -Name $GPOName -ReportType Html -Path $htmlPath
        Get-GPOReport -Name $GPOName -ReportType Xml -Path $xmlPath
        Write-Host "Створений звіт для $GPOName" -ForegroundColor Green
    } catch { Write-Host "Помилка генерації звіту $GPOName $_" -ForegroundColor Red }
}

function Export-GPInheritance {
    param (
        [string]$TargetOU
    )
    try {
        $Inheritance = Get-GPInheritance -Target $TargetOU
        $InheritanceFilePath = "C:\GPO_Report\GP_Inheritance_$(Get-Date -
        Format 'yyyy-MM-dd_HH:mm:ss').txt"
        $Inheritance | Out-File -FilePath $InheritanceFilePath
    }
}

```

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				143
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        Write-Host "Звіт про наслідування політик збережено в $InheritanceFilePath" -
ForegroundColor Green
    } catch { Write-Host "Помилка отримання наслідування: $_" -ForegroundColor Red }
}

function Compare-GPOs {
    param (
        [Parameter(Mandatory = $true)]
        [Object[]]$UsedGPO
    )

    $xmlFolder = Join-Path -Path "C:\GPO_Report\$($UsedGPO.DisplayName)" -ChildPath "xml"
    if (-not (Test-Path $xmlFolder)) {
        Write-Host "Папка не знайдена: $xmlFolder" -ForegroundColor Red
        return
    }

    $files = Get-ChildItem -Path $xmlFolder -Filter *.xml
    $SelectedFiles = Select-FromList -Items $files -Property "Name"

    if ($SelectedFiles.Count -eq 2) {
        $Report = @(
            $SelectedFiles[0].FullName
            $SelectedFiles[1].FullName
        )

        try {
            $Comparison = Compare-Object -ReferenceObject (Get-Content $Report[0]) -
DifferenceObject (Get-Content $Report[1])
            $ComparisonFilePath = "C:\GPO_Report\Comparison_$(Get-Date -Format 'yyyy-MM-
dd_HH:mm:ss').txt"
            $Comparison | Out-File -FilePath $ComparisonFilePath
            Write-Host "Результат порівняння збережено в $ComparisonFilePath" -
ForegroundColor Green
        } catch { Write-Host "Помилка при порівнянні: $_" -ForegroundColor Red }
    } else { Write-Host "Потрібно вибрати рівно 2 файли!" -ForegroundColor Red }
}

function GPResultantSetOfPolicy {
    param (
        [Parameter(Mandatory = $true)]
        [Object[]]$UsedUser,
        [Parameter(Mandatory = $true)]
        [Object[]]$UsedComp
    )

    foreach ($OU in $UsedUser) {
        foreach ($Comp in $UsedComp) {
            try {
                $Path = "C:\GPO_Report\RSOP_${OU}_${Comp}_$(Get-Date -Format 'yyyy-MM-
dd_HH:mm:ss').html"
                Get-GPResultantSetOfPolicy -User $OU -Computer $Comp -ReportType HTML -Path
$Path
                Write-Host "Звіт збережено: $Path" -ForegroundColor Green
            } catch { Write-Host "Помилка RSOP для $OU@$Comp $_" -ForegroundColor Red }
        }
    }
}

Write-Host "---- Меню ----" -ForegroundColor Cyan
Write-Host "1. Згенерувати звіт для всіх політик"
Write-Host "2. Згенерувати звіт для обраної політики"
Write-Host "3. RSOP звіт для користувача і комп'ютера"
Write-Host "4. Звіт про наслідування політик"
Write-Host "5. Порівняння звітів"
Write-Host "6. Вихід"

$choice = Read-Host "Введіть номер дії (1-6)"
switch ($choice) {
    1 { Export-GPOReport }
    2 {
        $AllGPO = Get-GPO -All
        $SelectedGPO = Select-FromList -Items $AllGPO -Property "DisplayName" | Select-
Object -First 1
        Export-GPOReport -GPOName $SelectedGPO.DisplayName
    }
    3 {
        try {
            $AllUsers = Get-ADUser -Filter *
            $SelectedUser = Select-FromList -Items $AllUsers -Property "SamAccountName"
            $AllComps = Get-ADComputer -Filter *

```

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				144
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        $SelectedComputer = Select-FromList -Items $AllComps -Property "Name"
        GPResultantSetOfPolicy -UsedUser $SelectedUser -UsedComp $SelectedComputer
    } catch { Write-Host "Помилка отримання AD-об'єктів: $_" -ForegroundColor Red }
}
4 {
    try {
        $AllOUs = Get-ADOrganizationalUnit -Filter *
        $SelectedOU = Select-FromList -Items $AllOUs -Property "DistinguishedName"
        foreach ($OU in $SelectedOU) {
            Export-GPInheritance -TargetOU $OU
        }
    } catch { Write-Host "Помилка отримання OU: $_" -ForegroundColor Red }
}
5 {
    $AllGPO = Get-GPO -All
    $SelectedGPO = Select-FromList -Items $AllGPO -Property "DisplayName" | Select-
Object -First 1
    Compare-GPOs -UsedGPO $SelectedGPO
}
6 { Write-Host "Вихід" }
default { Write-Host "Невірний вибір!" -ForegroundColor Red }
}

```

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				
Змн.	Арк.	№ докум.	Підпис	Дата		145

“Backup_TS”

```

$BackupBasePath = "C:\Scripts\GPO_Backups"
$Date = Get-Date -Format "yyyyMMdd_HH:mm:ss"
$LogFile = "$BackupBasePath\BackupLog.txt"

if (-not (Test-Path $BackupBasePath)) {
    New-Item -ItemType Directory -Path $BackupBasePath | Out-Null
}

$GPOList = Get-GPO -All | Select-Object -ExpandProperty DisplayName

foreach ($GPOName in $GPOList) {
    $BackupPath = "$BackupBasePath\$GPOName\$Date"

    if (-not (Test-Path $BackupPath)) {
        New-Item -ItemType Directory -Path $BackupPath | Out-Null
    }
    Backup-GPO -Name $GPOName -Path $BackupPath -Domain "makaryk.net"
}

```

“ScheduledTask_Backup”

```

$scriptPath = "C:\Scripts\ScriptLog.ps1"

if (Test-Path $scriptPath) {
    $action = New-ScheduledTaskAction -Execute "powershell.exe" -Argument "-NoProfile -
ExecutionPolicy Bypass -File `"$scriptPath`""
    $trigger = New-ScheduledTaskTrigger -Once -At (Get-Date) -RepetitionInterval (New-
TimeSpan -Hours 4) -RepetitionDuration (New-TimeSpan -Days 1)
    $taskName = "BackupAndLogs"
    try {
        Register-ScheduledTask -TaskName $taskName -Action $action -Trigger $trigger -User
"SYSTEM" -RunLevel Highest
        Write-Host "Задача '$taskName' створена та виконуватиметься кожні 4 години." -
ForegroundColor Green
    } catch { Write-Host "Не вдалося створити задачу: $_" -ForegroundColor Red }
} else { Write-Host "Файл скрипта не знайдено за шляхом: $scriptPath" -ForegroundColor Red }

```

“ScriptLog”

```

$logFolder = "C:\Scripts\Logs"
$logFile = Join-Path -Path $logFolder -ChildPath "BackupLog.txt"

function Write-Log {
    param (
        [string]$message
    )
    $timestamp = Get-Date -Format "yyyy-MM-dd HH:mm:ss"
    $logMessage = "$timestamp - $message"
    Add-Content -Path $logFile -Value $logMessage
}

Write-Log "start"
try {
    $ScriptPath = "C:\Scripts\Backup_TS.ps1"
    Start-Process -FilePath "powershell.exe" -ArgumentList "-NoProfile -ExecutionPolicy
Bypass -File `"$ScriptPath`"" -Wait
    Write-Log "Run'$ScriptPath'."
} catch { Write-Log "Error: $_" }

Write-Log "end"

```

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				146
Змн.	Арк.	№ докум.	Підпис	Дата		

“CreateTask_CompareGPOL”

```

$scriptPath = "C:\Scripts\Compare-GPODuplicates.ps1"
$taskName = "CompareGPO_Duplicates"

if (Test-Path $scriptPath) {
    $action = New-ScheduledTaskAction -Execute "powershell.exe" -Argument "-NoProfile -
ExecutionPolicy Bypass -File \"$scriptPath\""
    $trigger = New-ScheduledTaskTrigger -Once -At (Get-Date) -RepetitionInterval (New-
TimeSpan -Hours 2) -RepetitionDuration (New-TimeSpan -Days 30)

    try {
        Register-ScheduledTask -TaskName $taskName -Action $action -Trigger $trigger -User
"SYSTEM" -RunLevel Highest
        Write-Host "Задача '$taskName' створена. Виконуватиметься кожні 2 години протягом
30 днів." -ForegroundColor Green
    }
    catch { Write-Host "Помилка при створенні задачі: $_" -ForegroundColor Red }
} else { Write-Host "Скрипт не знайдено: $scriptPath" -ForegroundColor Red }

```

“Compare-GPODuplicates”

```

Import-Module GroupPolicy
Import-Module C:\Scripts\UserGPOModule.psm1 -Force

$logDir = "C:\Logs\CompareGPO"
$logFile = Join-Path $logDir "CompareGPO_$(Get-Date -Format 'yyyy-MM-dd HH-mm-ss').log"

if (-not (Test-Path $logDir)) {
    New-Item -ItemType Directory -Path $logDir | Out-Null
}
Function Log {
    param([string]$Message)
    $timestamp = Get-Date -Format "yyyy-MM-dd HH-mm-ss"
    $line = "$timestamp`t$Message"
    Add-Content -Path $logFile -Value $line
}
Log "( ① ω ① *)"

try {
    $RegistryMap = @{}
    $AllGPOs = Get-GPO -All

    foreach ($GPO in $AllGPOs) {
        Log "Processing GPO: $($GPO.DisplayName)"
        try {
            $regValues = Get-GPAllRegistryValues -Name $GPO.DisplayName
            foreach ($val in $regValues) {
                $key = "$($val.KeyPath)| $($val.ValueName)"
                if (-not $RegistryMap.ContainsKey($key)) {
                    $RegistryMap[$key] = @()
                }
                $RegistryMap[$key] += $GPO.DisplayName
            }
        } catch { Log "Error processing GPO '$($GPO.DisplayName)': $_" }
    }
    $duplicates = $RegistryMap.GetEnumerator() | Where-Object { $_.Value.Count -gt 1 }

    if ($duplicates.Count -eq 0) {
        Log "No duplicate registry entries found."
    } else {
        foreach ($entry in $duplicates) {
            $keySplit = $entry.Key -split '\|'
            $keyPath = $keySplit[0]
            $valueName = $keySplit[1]
            $gpoList = $entry.Value -join ', '
            Log "Duplicate found: $keyPath -> $valueName | GPOs: $gpoList"
        }
    }
} catch { Log "Unexpected error: $_" }

Log "☺(¯▽¯) Bye~Bye~"

```

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				147
Змн.	Арк.	№ докум.	Підпис	Дата		

“Check-UnlinkedGPOs”

```
function Get-AllGpoTargets {
    $OUS = Get-ADOrganizationalUnit -Filter * | Select-Object -ExpandProperty DistinguishedName
    $domainObject = Get-ADDomain
    $DomainDN = $domainObject.DistinguishedName

    return @(
        foreach ($DN in $OUS + $DomainDN) { [PSCustomObject]@{ DistinguishedName = $DN } }
    )
}

function Get-UnlinkedGPOs {
    $AllGPOs = Get-GPO -All
    $LinkedGPOs = @()
    $Targets = Get-AllGpoTargets

    foreach ($target in $Targets) {
        try {
            $Inheritance = Get-GPInheritance -Target $target.DistinguishedName
            foreach ($Link in $Inheritance.GpoLinks) {
                if ($Link.DisplayName -and $LinkedGPOs -notcontains $Link.DisplayName) {
                    $LinkedGPOs += $Link.DisplayName
                }
            }
        } catch { Write-Warning "Помилка при обробці $($target.DistinguishedName): $_" }
    }
    $UnlinkedGPOs = $AllGPOs | Where-Object { $LinkedGPOs -notcontains $_.DisplayName }
    return $UnlinkedGPOs
}

$LogFolder = "C:\Scripts\Logs"
$LogPath = "$LogFolder\UnlinkedGPOs.txt"
if (-not (Test-Path $LogFolder)) { New-Item -ItemType Directory -Path $LogFolder | Out-Null }

$UnlinkedGPOs = Get-UnlinkedGPOs
$UnlinkedGPOs | Select-Object DisplayName | Out-File -FilePath $LogPath -Encoding UTF8
```

“Create-ScheduledTask_UnlinkedGPOs”

```
$scriptPath = "C:\Scripts\Check-UnlinkedGPOs.ps1"
$taskName = "CheckUnlinkedGPOs"

if (Test-Path $scriptPath) {
    $action = New-ScheduledTaskAction -Execute "powershell.exe" -Argument "-NoProfile -ExecutionPolicy Bypass -File \"$scriptPath\""
    $trigger = New-ScheduledTaskTrigger -Daily -At 6am
    try {
        Register-ScheduledTask -TaskName $taskName -Action $action -Trigger $trigger -User "SYSTEM" -RunLevel Highest
        Write-Host "Задача '$taskName' створена." -ForegroundColor Green
    } catch { Write-Host "Помилка при створенні задачі: $_" -ForegroundColor Red }
} else { Write-Host "Скрипт не знайдено за шляхом: $scriptPath" -ForegroundColor Red }
```

		Макарик А.С.			ККІКБ.КР.Б – 123 – 15 – ПЗ	Арк.
		Фальковський І.Г.				148
Змн.	Арк.	№ докум.	Підпис	Дата		