

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет комп'ютерних наук та технологій
Кафедра комп'ютерних систем та мереж

Пояснювальна записка

до дипломного проекту (роботи)

бакалавра

(ступінь вищої освіти)

на тему Розробка системи віддаленого керування об'єктами
критичної інфраструктури

Виконав: студент 4 курсу, групи КНТ-519

Спеціальності 123 “Комп'ютерна інженерія”
(код і найменування спеціальності)

Освітня програма (спеціалізація) «Комп'ютерна інженерія»

ПЕСТОВ О. Д.
(прізвище та ініціали)

Керівник КИРИЧЕК. Г. Г.
(прізвище та ініціали)

Рецензент КУЛЯБА-ХАРИТОНОВА Т. І.
(прізвище та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»
(повне найменування закладу вищої освіти)

Інститут, факультет Комп'ютерних наук і технологій
 Кафедра «Комп'ютерні системи та мережі»
 Ступінь вищої освіти бакалаврський
 Спеціальність 123 Комп'ютерна інженерія
(код і найменування)
 Освітня програма (спеціалізація) Комп'ютерна інженерія
(назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮЗавідувач кафедри КУДЕРМЕТОВ Р.К.

Ку
 « » 2023 року

З А В Д А Н Н Я
НА ДИПЛОМНИЙ ПРОЕКТ (РОБОТУ) СТУДЕНТА

ПЕСТОВА Олексія Дмитровича(прізвище, ім'я, по батькові)

1. Тема проекту (роботи) Розробка системи віддаленого керування об'єктами критичної інфраструктури

керівник проекту (роботи) к. т. н., доцент, КИРИЧЕК Галина Григорівна,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом закладу вищої освіти від «24» березня 2023 року № 71

2. Строк подання студентом проекту (роботи) 10 травня 2023 року

3. Вихідні дані до проекту (роботи) наявні методи та засоби віддаленого керування, вихідна система керованого об'єкту

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1) Аналіз технічного завдання та наявних засобів віддаленого керування
2) Розробка конфігураційної утиліти YggVPN;
3) Проектування системи на конкретному прикладі

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)

Структурна схема вихідної системи

Структурна схема готової системи

Структурна схема логічних з'єднань готової системи

Графіки доступності обраних публічних вузлів мережі Yggdrasil

6. Консультанти розділів проекту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-3	КИРИЧЕК Г. Г., к. т. н., доцент		
нормоконтроль	ЩЕРБАК Н.В., ст. викл.		

7. Дата видачі завдання « 23 » лютого 2023 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проекту (роботи)	Строк виконання етапів проекту (роботи)	Примітка
1	Аналіз технічного завдання та наявних засобів віддаленого керування	25.03.2023 р.	
2	Визначення технологій, необхідних для реалізації системи	05.04.2023 р.	
3	Збір даних про доступність обраних публічних вузлів мережі Yggdrasil	28.04.2023 р.	
4	Розробка конфігураційної утиліти YggVPN	30.04.2023 р.	
5	Проектування системи на конкретному прикладі	05.05.2023 р.	
6	Оформлення графічного матеріалу	10.05.2023 р.	
7	Оформлення пояснювальної записки	15.05.2023 р.	

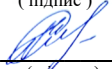
Студент(ка)


(підпис)

Олексій ПЕСТОВ

(прізвище та ініціали)

Керівник проекту (роботи)


(підпис)

Галина КИРИЧЕК

(прізвище та ініціали)

РЕФЕРАТ

ПЗ: 92 с; 24 рис; 1 табл; 22 джерела; 1 додаток.

ВІДДАЛЕНЕ КЕРУВАННЯ, ВІДДАЛЕНИЙ ДОСТУП, P2P, VNC, IPV6, ОВЕРЛЕЙНА МЕРЕЖА, УТИЛІТА

Об'єктом дослідження є організація віддаленого керування комп'ютеризованих систем через оверлейну P2P-мережу Yggdrasil.

Предметом дослідження є методи і засоби забезпечення віддаленого керування комп'ютеризованих систем.

Метою даної дипломної роботи є проектування системи віддаленого керування об'єктами критичної інфраструктури в умовах активних бойових дій та аварійних відключень електроенергії.

У роботі проаналізовано наявні готові рішення віддаленого керування, виявлено їх недоліки та запропоновано варіант його реалізації через мережу Yggdrasil при використанні мобільних телефонів з підтримкою протоколу RNDIS та мереж мобільних операторів зв'язку, а також певного клієнтського програмного забезпечення.

Для полегшення процесу коректного встановлення та налаштування клієнту Yggdrasil на кінцевих пристроях під управлінням операційних систем сімейства Windows було розроблено конфігураційну утиліту YggVPN мовою C.

Запропонований варіант системи було реалізовано на конкретному прикладі з використанням клієнтського та серверного програмного забезпечення TigerVNC для робочих місць та додатку droidVNC-NG для віддаленого керування телефонами.

ЗМІСТ

Вступ.....	7
1 Аналіз технічного завдання та наявних засобів віддаленого керування.....	8
1.1 Основні поняття та визначення.....	8
1.2 Аналіз наявних рішень у сфері віддаленого керування.....	9
1.3 Аналіз наявних рішень у сфері віддаленого доступу.....	11
1.4 Постановка завдань та загальна структура системи віддаленого керування....	13
2 Розробка конфігураційної утиліти YggVPN.....	15
2.1 Процедура встановлення та налаштування клієнту Yggdrasil.....	15
2.2 Головна функція та меню.....	28
2.3 Виведення списків публічних вузлів та хостів.....	32
2.4 Редагування списків публічних вузлів та хостів.....	33
2.5 Процедура видалення.....	36
2.6 Маніфест додатку та компіляція.....	36
2.7 Вигляд готової утиліти.....	38
3 Проектування системи на конкретному прикладі.....	43
3.1 Аналіз структури наявної системи об'єкту.....	43
3.2 Організація виходу у Інтернет.....	44
3.3 Обирання публічних вузлів Yggdrasil.....	46
3.4 Налаштування локальних робочих місць.....	50
3.4.1 Встановлення серверу TigerVNC.....	50
3.4.2 Встановлення і налаштування клієнту Yggdrasil.....	51
3.4.3 Перенаправлення портів.....	53
3.4.4 Налаштування віддаленого керування телефонами.....	54
3.4.5 Додавання правил брандмауєру.....	55
3.5 Налаштування віддалених робочих місць.....	57
3.6 Структура готової системи та перевірка її роботи.....	57

Висновки.....	62
Перелік джерел посилання.....	64
Додаток А.....	66
Перелік графічних матеріалів:	
Креслення 1 — Структурна схема вихідної системи	
Креслення 2 — Структурна схема готової системи	
Креслення 3 — Структурна схема логічних з'єднань готової системи	
Плакат 1 — Графіки доступності обраних публічних вузлів мережі Yggdrasil	

ВСТУП

Об'єктами критичної інфраструктури є об'єкти інфраструктури, системи, їх частини та їх сукупність, які є важливими для економіки, національної безпеки та оборони, порушення функціонування яких може завдати шкоди життєво важливим національним інтересам [1]. До таких об'єктів можна віднести електричні станції та підстанції, аеродроми, залізничні вузли, військові об'єкти, тощо.

У сучасних реаліях і умовах активних бойових дій об'єкти критичної інфраструктури часто піддаються ракетним ударам, атакам за допомогою баражуючих боєприпасів та безпілотних літальних апаратів (БПЛА) [2], тому з міркувань безпеки персоналу особисто знаходитись на таких об'єктах цілодобово не рекомендується. У той же час, особиста присутність персоналу необхідна для здійснення чергування, спостереження та підтримки дієздатності об'єкту. Таким чином, виникає проблема підтримки дієздатності об'єкту з мінімальними ризиками для життя та здоров'я персоналу.

На щастя, більшість підприємств комп'ютеризують свої системи для спрощення обслуговування та спостереження, що відкриває певні шляхи здійснення віддаленого керування такими об'єктами зі значно меншими ризиками для персоналу. Цей дипломний проект присвячено розгляданню та вирішенню проблем впровадження надійного віддаленого керування комп'ютеризованими об'єктами критичної інфраструктури на конкретному прикладі.

1 АНАЛІЗ ТЕХНІЧНОГО ЗАВДАННЯ ТА НАЯВНИХ ЗАСОБІВ ВІДДАЛЕНОГО КЕРУВАННЯ

1.1 Основні поняття та визначення

Важливо розрізняти засоби віддаленого доступу та віддаленого керування. Під віддаленим доступом до об'єкту мається на увазі лише організація зв'язку між ним та віддаленим робочим місцем, тоді як віддалене керування є процесом спостереження та впливу оператора на об'єкт з віддаленого робочого місця за допомогою певних програмних та апаратних засобів. У більшості випадків для здійснення віддаленого керування об'єктом необхідно мати віддалений доступ до нього.

Логічно припустити, що віддалене керування можливо лише для комп'ютеризованих систем. Комп'ютеризованим об'єктом інфраструктури будемо вважати такий, експлуатація якого здійснюється за допомогою комп'ютеру (робочого місця) під управлінням операційної системи сімейств Windows, Unix, тощо з відповідним програмним забезпеченням. Такі робочі місця поділяються на так звані товсті та тонкі клієнти.

Товсті клієнти у даному випадку є пристроями, до яких безпосередньо підключаються усі контрольовані підсистеми. Програмне забезпечення на таких клієнтах частіше за все є специфічним і створеним спеціально для конкретної системи. На товсті клієнти також не завжди є можливість встановити стороннє програмне забезпечення, що ускладнює задачу віддаленого керування та змушує вдаватися до застосування зовнішніх апаратних засобів, таких як KVM-over-IP комутатор.

Тонкі клієнти є малопотужними комп'ютерами, які лише надають доступ до серверу через локальну мережу об'єкту [3], який в свою чергу і контролює підключені до нього підсистеми. Доступ при цьому здійснюється засобами мережевого програмного забезпечення, встановленого на клієнті та сервері

відповідно. У такому випадку не важливо, яким саме є це програмне забезпечення, оскільки впровадження віддаленого керування зводиться до встановлення цього ж програмного забезпечення на віддалене робоче місце та організації зв'язку між ним та сервером.

1.2 Аналіз наявних рішень у сфері віддаленого керування

На даний момент існує велика кількість готових повноцінних програмних рішень, що забезпечують як віддалений доступ, так і віддалене керування комп'ютерами, зокрема AnyDesk, Team Viewer, RealVNC VNC Connect, GoToMyPC та ін. Хоча вони прості у налаштуванні і вимагають здебільшого лише доступу у Інтернет, їм характерні наступні недоліки:

- централізованість;
- закритість вихідного коду;
- платність для комерційного використання та/або обмеження безкоштовної версії.

Централізованість означає, що при використанні рішення доводиться покладатися на певну компанію та її сервери. У такому разі вона є єдиною точкою відмови, а також може у будь-який момент припинити надавання своїх послуг. Закритість вихідного коду, в свою чергу, практично унеможлиблює переконання у конфіденційності зв'язку з об'єктом та відсутності можливості втручання сторонніх осіб, що ставить під сумнів безпеку об'єкту.

Альтернативою переліченим рішенням є система MeshCentral, яка є повноцінним платформонезалежним рішенням віддаленого керування з відкритим вихідним кодом як серверної, так і клієнтської частини [4]. MeshCentral є безкоштовним до використання і дозволяє розгорнути серверну частину на власному сервері, до якого як клієнти підключаються керовані комп'ютери та

віддалені робочі місця. При цьому керування підконтрольним комп'ютером може здійснюватись з веб-інтерфейсу сервера (віддалений робочий стіл, емулятор терміналу, доступ до файлів, тощо), або засобами сторонніх утиліт, зв'язок для яких організується за допомогою перенаправлення портів утилітою MeshRouter. Увесь трафік є зашифрованим за замовчанням. Користувачі реєструються адміністратором серверу, автентифікація здійснюється за логіном та паролем з можливістю підключення додаткової двохфакторної автентифікації та апаратних ключів [5].

Використання MeshCentral вирішує проблеми закритості вихідного коду, але водночас додає необхідність власного сервера з публічно-доступною IP-адресою або оренди віртуального приватного сервера (VPS) для розгортання системи, що зазвичай є платними послугами. До того ж, залишається проблема централізованості – сервер є єдиною точкою відмови системи, при цьому ситуація погіршується, якщо його розміщено у межах країни і він піддається періодичним аварійним відключенням електроенергії.

Оскільки існуючі готові рішення є затратними або недостатньо надійними, задача організації віддаленого керування може бути розбита на підзадачі керування та доступу, що вирішуються окремими засобами.

Вирішення задачі керування віддаленою системою залежить від структури керованої системи. Якщо у її основі лежить тонкий клієнт, то, скоріш за все, у системі вже використовується певний протокол віддаленого керування – SSH, VNC, RDP, NX або деякий специфічний протокол обміну даними з сервером. Таким чином, залишається лише встановити відповідне програмне забезпечення тонкого клієнта на віддалене робоче місце і вирішити задачу доступу.

У випадку товстого клієнта з можливістю встановлення стороннього програмного забезпечення керування забезпечується встановленням на нього відповідного серверного додатку з підтримкою одного з перелічених вище протоколів, наприклад, TigerVNC, який є платформонезалежним VNC-сервером та клієнтом з відкритим вихідним кодом і забезпечує контроль над середовищем

робочого стола сервера [6]. Для товстого клієнта без можливістю встановлення стороннього програмного забезпечення єдиним рішенням є застосування зовнішнього апаратного комутатора KVM-over-IP, який емулює пристрої введення та виведення, і водночас є сервером з підтримкою певних протоколів віддаленого керування. Так задача зводиться до організації доступу віддаленого робочого місця до сервера.

1.3 Аналіз наявних рішень у сфері віддаленого доступу

Задача доступу може бути вирішена багатьма способами, хоча більшість з них є досить затратними. Одним з таких є виділена лінія зв'язку між об'єктом та віддаленим робочим місцем, що є досить надійним рішенням, але вимагає значних фінансових і часових інвестицій та обмежує мобільність віддаленого оператора. До того ж, не всі провайдери надають такі послуги.

Для забезпечення мобільності віддалених операторів об'єкту ідеальним варіантом був би доступ до нього через Інтернет, але його реалізація не є такого прямолінійною. Навіть якщо у об'єкту вже наявний вихід у глобальну мережу, скоріше за все, він знаходиться за NAT відповідного провайдера.

NAT (Network Address Translation, перетворення мережевих адрес) є механізмом у мережах TCP/IP, призначеним для економії глобальних IP-адрес за рахунок підміни маршрутизатором адреси відправника у вихідних пакетах та адреси призначення у вхідних. Таким чином, пристрої з приватними локальними адресами можуть створювати вихідні з'єднання з хостами у глобальній мережі, але зворотне для них неможливо — пристрій за NAT не має унікальної глобальної адреси, за якою до нього можна було б звернутися. Зазвичай, це вирішується отриманням статичної глобальної IP-адреси, яка є платною послугою у більшості провайдерів.

Для обходу NAT можливо застосувати власний виділений VPN-сервер, проте такий сервер сам по собі вимагає статичної глобальної IP-адреси або VPS-хостингу, а також йому притаманна вже обговорена проблема централізованості.

Альтернативою VPN-серверу може бути оверлейна P2P-мережа Yggdrasil, яка побудована поверх протоколу IP. У Yggdrasil кожен вузол мережі може звертатися до будь-якого іншого за його внутрішньомережевою IPv6-адресою, а трафік між ними проходить через інші вузли доти, доки не досягне адресата. Мережі властиві самоконфігурація та децентралізованість — кожним вузлом адреса отримується як частина хеш-суми відкритого ключа випадково згенерованої асиметричної пари ключів. Маршрутизація пакетів можлива засобами координат дерева мережі, розподіленої хеш-таблиці (DHT) та тимчасових таблиць маршрутизації на кожному вузлі. Весь трафік у мережі шифрується відправником перед надсиланням та може бути розшифрований лише отримувачем, гарантуючи конфіденційність зв'язку.

Вузли у одній локальній мережі знаходять один одного автоматично за допомогою multicast-розсилки. Вузли, що знаходяться за NAT, підключаються до мережі через одного або декількох так званих публічних вузлів, тому з точки зору NAT з'єднання завжди є вихідним [7]. Саме цю властивість Yggdrasil можна використати для обходу NAT та зв'язку з об'єктом, при цьому зв'язок буде можливий доти, доки буде доступний хоча б один публічний вузол, що підвищує надійність системи порівняно з централізованими рішеннями.

Клієнт Yggdrasil, написаний мовою програмування Go, є платформонезалежним та має відкритий вихідний код [8]. До недоліків мережі відносять незручність використання довгих IPv6-адрес та відсутність вбудованого фаєрволу для обмеження доступу до вузла. Довжина адрес може бути компенсована використанням файлу hosts, який наявний у майже будь-якій операційній системі і дозволяє поставити у відповідність адресам зручніші символічні імена. Для фільтрації трафіку рекомендується використовувати сторонні засоби, наприклад фаєрвол iptables у дистрибутивах Linux або політики

безпеки IPsec у Windows [7]. До того ж, оскільки адреси отримуються криптографічно, фільтрацію трафіку за адресою джерела можна використовувати у якості міри безпеки разом з парольним захистом.

Наявність лише одного каналу зв'язку робить його єдиною точкою відмови, погіршуючи надійність системи. Для зберігання доступу у разі його відмови необхідно ввести резервні виходи у Інтернет. Такими можуть виступати мережі мобільних операторів зв'язку, підключення до яких можливе за допомогою локальних робочих місць об'єкту, до яких підключені USB-модеми або звичайні мобільні телефони з підтримкою протоколу RNDIS, який дозволяє пристрою емулювати мережевий інтерфейс Ethernet поверх інтерфейсу USB [9].

1.4 Постановка завдань та загальна структура системи віддаленого керування

Виходячи з аналізу наявних засобів віддаленого керування та доступу, розроблена система віддаленого керування об'єктом повинна:

- мати відкритий вихідний код, що стосується усіх залучених програмних компонентів;
- бути децентралізованою або надійно резервованою;
- забезпечувати конфіденційність зв'язку з об'єктом;
- виключати несанкціонований доступ до об'єкту;
- не обмежувати мобільність віддалених операторів;
- бути відносно простою у експлуатації для віддалених операторів;
- підтримувати будь-які резервні канали зв'язку з Інтернет.

Згідно з цими вимогами, можемо навести загальну структуру системи віддаленого керування об'єктом.

Як засіб керування об'єктом використовуються:

а) для систем з тонкими клієнтами — наявне програмне забезпечення, встановлене на віддалені робочі місця;

б) для систем з товстим клієнтом:

1) з можливістю встановлення стороннього програмного забезпечення – сервер TigerVNC на керованому комп'ютері, клієнт TigerVNC на віддалених робочих місцях;

2) без можливості встановлення стороннього програмного забезпечення – комутатор KVM-over-IP з підтримкою протоколу VNC, клієнт TigerVNC на віддалених робочих місцях.

Як засіб доступу до об'єкту використовуються:

а) наявні виходи у Інтернет, а також мережі мобільних операторів через USB-модеми або мобільні телефони з підтримкою протоколу RNDIS, підключені до локальних робочих місць;

б) клієнт Yggdrasil на локальних та віддалених робочих місцях;

в) файл hosts на віддалених робочих місцях.

Як засіб безпеки і обмеження доступу до об'єкту використовуються фаєрвол або політики безпеки IPsec на локальних та віддалених робочих місцях.

2 РОЗРОБКА КОНФІГУРАЦІЙНОЇ УТИЛІТИ YGGVPN

Оскільки на комп'ютерах під управлінням операційних систем сімейства Windows налаштування клієнту Yggdrasil, файлу hosts та в особливості політик безпеки IPsec є дещо громіздким, було розроблено спеціальну конфігураційну утиліту YggVPN. Для розробки було обрано мову програмування C, оскільки саме нею реалізовані необхідні бібліотеки та системні виклики Windows. Розроблена утиліта є консольним додатком з текстовим інтерфейсом.

2.1 Процедура встановлення та налаштування клієнту Yggdrasil

При першому запуску програма автоматично завантажує та встановлює останню версію клієнта Yggdrasil необхідною архітектури з репозиторію GitHub. Підтримуються як 32-бітні, так і 64-бітні версії Windows починаючи з Windows 7. Остання версія клієнту визначається за допомогою API GitHub та бібліотеки json-c для зчитування завантажуваного JSON-файлу. Завантаження виконується за допомогою функції URLDownloadToFile() бібліотеки Windows Urlmon. Системний виклик GetNativeSystemInfo() використовується для отримання архітектури системи (x86 або x64). Далі з номеру останньої версії клієнту та архітектури будується URL для завантаження. Завантажений файл зберігається у директорії Temp під ім'ям "yggdrasil.msi". Встановлення виконується за допомогою стандартної утиліти Windows msexec, процес якої створюється системним викликом CreateProcessA() з відповідними аргументами командного рядку. Після встановлення файл переноситься до директорії C:\ProgramData\YggVPN.

Фрагмент програмного коду функції `install()`, який відповідає за завантаження та встановлення клієнту Yggdrasil, наведено у лістингу А.1.

Далі виконується редагування файлу `C:\ProgramData\Yggdrasil\yggdrasil.conf`, який містить налаштування клієнту Yggdrasil у форматі JSON або HJSON. HJSON є більш орієнтованим на редагування людиною варіантом JSON і застосовується за замовчанням [10]. Оскільки бібліотека `json-c` не підтримує HJSON, перед редагуванням файл нормалізується у стандартний JSON за допомогою виконуваного файлу `C:\Program Files\Yggdrasil\yggdrasil.exe` з відповідними аргументами командного рядку.

Редагування файлу складається з:

- встановлення опції `NodeInfoPrivacy` у значення `true`, що вимикає поширення такої інформації про вузол, як, наприклад використовувана операційна система;
- заповнення секції `Peers`, що містить URL публічних вузлів, які обираються користувачем.

Введення URL публічних вузлів користувачем реалізовано у функції `add_peers()`. Функція отримує рядок від користувача за допомогою функції `scanf()` і перевіряє його збіг з рядком `"f"`, який означає завершення введення та повернення з функції (лише якщо наявний хоча б один публічний вузол). Далі перевіряється формат рядку, оскільки URL публічних вузлів мають починатися з `"tcp://"`, `"tls://"` або `"socks://"` у випадку з'єднання через SOCKS-проксі. Також перевіряється наявність URL у списку публічних вузлів для виключення дубльованих записів. Функція зчитує та редагує цей список та кількість публічних вузлів у структурі `config` через покажчик `PCONF`, які буде розглянуто пізніше. Фактичне оновлення конфігурації Yggdrasil здійснюється функцією `update_yggdrasil_config()`. Функція використовує бібліотеку `json-c` для редагування секції `Peers` файлу `C:\ProgramData\Yggdrasil\yggdrasil.conf` відповідно до вмісту структури `config`.

Програмний код функції `add_peers()` наведено у лістингу 2.1. Програмний код функції `update_yggdrasil_config()` наведено у лістингу 2.2. Фрагмент програмного коду функції `install()`, який відповідає за налаштування клієнту Yggdrasil, наведено у лістингу 2.3.

Лістинг 2.1 — Програмний код функції `add_peers()`

```
void add_peers()
{
    char* buf = malloc(196*sizeof(char));
    for(;;)
    {
        printf("Peer URL (f to finish): ");
        scanf("%195s", buf);
        clear();
        if(strlen(buf) == 1 && !strcmp(buf, "f"))
        {
            if(PCONF->peer_count > 0)
                break;
            puts("You must add at least one peer.");
            continue;
        }
        if(strstr(buf, "tcp://") != buf && strstr(buf, "tls://") !=
= buf && strstr(buf, "socks://") != buf)
        {
            puts("Invalid peer URL.");
            continue;
        }
        if(find(PCONF->peers, PCONF->peer_count, buf) != -1)
            puts("Peer URL already present.");
        else
        {
            //Write the peer URL to config
            append(&PCONF->peers, PCONF->peer_count, strdup(buf));
            PCONF->peer_count++;
        }
    }
    free(buf);
}
```

Лістинг 2.2 — Програмний код функції `update_yggdrasil_config()`

```
void update_yggdrasil_config()
{
    json_object* root =
json_object_from_file(YGGDRASIL_CONFIG_PATH);
    json_object* obj_arr = json_object_new_array();
    json_object* obj_str;
    for(int i = 0; i < PCONF->peer_count; i++)
    {
        obj_str = json_object_new_string(PCONF->peers[i]);
        json_object_array_add(obj_arr, obj_str);
    }
    json_object_object_add(root, "Peers", obj_arr);
    json_object_to_file_ext(YGGDRASIL_CONFIG_PATH, root,
JSON_C_TO_STRING_PRETTY | JSON_C_TO_STRING_SPACED);
    json_object_put(root);
}
```

Лістинг 2.3 — Фрагмент програмного коду, який відповідає за налаштування клієнту Yggdrasil

```
root = json_object_from_file(YGGDRASIL_CONFIG_PATH);
//Normalise Yggdrasil config
if(root == NULL)
{
    system("\"C:\\Program Files\\Yggdrasil\\yggdrasil.exe\" -
useconffile \"YGGDRASIL_CONFIG_PATH\" -normaliseconf -json >
\"YGGDRASIL_CONFIG_PATH\");
    root = json_object_from_file(YGGDRASIL_CONFIG_PATH);
}
json_object* obj_b = json_object_new_boolean(TRUE);
json_object_object_add(root, "NodeInfoPrivacy", obj_b);
json_object_to_file_ext(YGGDRASIL_CONFIG_PATH, root,
JSON_C_TO_STRING_PRETTY | JSON_C_TO_STRING_SPACED);
json_object_put(root);
if(!EXPORTED_CONF)
{
    puts("\nPlease enter URLs of public peers you wish to use.
Choose the closest ones to you from
https://publicpeers.neilalexander.dev/, for example.");
    add_peers();
}
update_yggdrasil_config();
```

Наступний етап — налаштування служби Yggdrasil, яке виконується за допомогою API Windows. Воно полягає у встановленні функцією `set_yggdrasil_service_params()` параметрів служби, які відповідають за її перезапуск разі аварійного завершення з будь-якої причини. Це виконується за допомогою системного виклику `ChangeServiceConfig2A()` та заповнених структур `SERVICE_FAILURE_ACTIONS` і `SC_ACTION`, що задають перезапуск служби з затримками у 1, 5 та 10 секунд. Додатково виставляється флаг запуску цих дій при самостійному завершенні служби без помилок.

Для запуску, зупинки та очікування служби визначено функції `start_yggdrasil_service()`, `stop_yggdrasil_service()` та `wait_for_yggdrasil_service()` відповідно. Ці функції запитують статус служби за допомогою системного виклику `QueryServiceStatusEx()` і відповідно до нього виконують необхідні дії системними викликами `StartServiceA()` та `ControlService()` з аргументом `SERVICE_CONTROL_STOP` відповідно.

Перед використанням будь-яких функцій керування службою Yggdrasil необхідно викликати функцію `SCM_init()`, яка отримує дескриптори менеджера контролю служб та самої служби. Так само, після завершення роботи зі службою викликається функція `SCM_cleanup()`, яка закриває попередньо відкриті дескриптори.

Програмний код необхідних для керування службою функцій наведено у лістингу А.2. Фрагмент програмного коду функції `install()`, який відповідає за налаштування служби Yggdrasil наведено у лістингу 2.4

Далі програма виконує налаштування фаєрволу як політики безпеки IPsec за допомогою стандартної утиліти Windows `netsh`, яка призначена для налаштування мережевих служб та компонентів системи [11]. Необхідні для цього команди викликаються за допомогою системного виклику `system()`.

Лістинг 2.4 — Фрагмент програмного коду, який відповідає за налаштування служби Yggdrasil

```
printf("Stopping Yggdrasil service . . . ");
SCM_init();
if(!stop_yggdrasil_service()){
    printf("ERROR: Could not stop Yggdrasil service (%d).\n",
GetLastError());
    SCM_cleanup();
    system("pause");
    return -1;
}
puts("Done.");
...
printf("Configuring Yggdrasil service . . . ");
if(!set_yggdrasil_service_params()){
    printf("ERROR: Could not configure Yggdrasil service (%d).\n",
GetLastError());
    SCM_cleanup();
    system("pause");
    return -1;
}
puts("Done.");
printf("Starting Yggdrasil service . . . ");
if(!start_yggdrasil_service()){
    printf("ERROR: Could not start Yggdrasil service (%d).\n",
GetLastError());
    SCM_cleanup();
    system("pause");
    return -1;
}
puts("Done.");
SCM_cleanup();
```

Ініціалізація фаєрволу виконується функцією `firewall_init()` і складається зі скидання старих параметрів за їх наявності, створення політики `Yggdrasil_Firewall`, додавання білого та чорного списків фільтрів і створення для останнього фільтра, який блокує увесь трафік, адресований мережевому інтерфейсу `Yggdrasil`. До білого списку при цьому додається фільтр з власною

адресою. Списки фільтрів з'єднуються з політикою створенням відповідних правил.

Оскільки для створення фільтрів потрібна власна адреса вузла у мережі, вона отримується функцією `get_own_yggdrasil_address()` за допомогою API Yggdrasil. Ця функція з'єднується з локальним адміністративним сокетом клієнту Yggdrasil за портом 9001, використовуючи для цього бібліотеку Windows Sockets API (Winsock). Після встановлення з'єднання функція надсилає у сокет запит інформації про клієнт у форматі JSON, після чого отримує відповідь у тому ж форматі. З отриманого JSON за допомогою бібліотеки `json-c` виділяється адреса вузла.

Після ініціалізації фаєрвол вмикається функцією `firewall_enable()`, яка призначає створену політику безпеки відповідною командою. Так само, функція `firewall_disable()` виконує зворотнє у разі необхідності.

Програмний код функцій налаштування фаєрволу наведено у лістингу А.3. Фрагмент програмного коду функції `install()`, який відповідає за налаштування фаєрволу, наведено у лістингу 2.5. Програмний код функції `get_own_yggdrasil_address()` наведено у лістингу 2.6.

Лістинг 2.5 — Фрагмент програмного коду, який відповідає за налаштування фаєрволу

```
char* own_address = get_own_yggdrasil_address();
printf("Setting up the firewall . . . ");
firewall_init(own_address);
firewall_enable();
puts("Done.");
printf("\nYour Yggdrasil IPv6 address: %s\n", own_address);
puts("Share your IPv6 address with other hosts you want to have
access to this machine.\n");
```

Лістинг 2.6 — Програмний код функції `get_own_yggdrasil_address()`

```

char* get_own_yggdrasil_address() {
    WSADATA wsadata;
    WSASStartup(MAKEWORD(2, 2), &wsadata);
    SOCKET s = socket(AF_INET, SOCK_STREAM, IPPROTO_TCP);
    struct sockaddr_in yggdrasil_client_API;
    yggdrasil_client_API.sin_family = AF_INET;
    yggdrasil_client_API.sin_addr.s_addr = inet_addr("127.0.0.1");
    yggdrasil_client_API.sin_port = htons(9001);
    connect(s, (SOCKADDR*)&yggdrasil_client_API,
sizeof(yggdrasil_client_API));
    char* buf = "{\"request\":\"getSelf\"}";
    send(s, buf, (strlen(buf) + 1)*sizeof(char), 0);
    shutdown(s, SD_SEND);
    const int buf_len = 128;
    buf = malloc(buf_len);
    char* response;
    int response_len = 0;
    int received;
    for(;;){
        received = recv(s, buf, buf_len, 0);
        if(received <= 0)
            break;
        if(response_len > 0)
            extend((void**)&response, response_len, buf, received);
        else{
            response = malloc(received);
            memcpy(response, buf, received);
        }
        response_len += received;
    }
    free(buf);
    closesocket(s); WSACleanup();
    json_object* root = json_tokener_parse(response);
    free(response);
    json_object* obj;
    json_object_object_get_ex(root, "response", &obj);
    json_object* str_obj;
    json_object_object_get_ex(obj, "address", &str_obj);
    char* address = strdup(json_object_get_string(str_obj));
    json_object_put(str_obj); json_object_put(obj);
    json_object_put(root);
    return address;
}

```

Далі за необхідності до дозволених хостів фаєрволу додаються адреси та відповідні їм символьні імена, що вводяться користувачем, а також задається власне символьне ім'я хосту. Для цього використовується функція `add_hosts()`, аналогічна функції `add_peers()`, наведеній у лістингу 2.1. Окрім введення адреси хосту, вона також дає можливість ввести його символьне ім'я, перевіряючи при цьому його наявність у поточній конфігурації. При введенні імен хостів виконується перевірка на порожній рядок — у такому випадку замість імені хосту заноситься “ ”.

Додавання хосту до білого списку фаєрволу здійснюється за допомогою функції `firewall_add_host()`, наведеної у лістингу А.3. Функція будує командний рядок з адреси хосту та виконує його системним викликом `ropen()`, який використовується тут замість `system()` для перевірки результату виконання на помилку шляхом читання з отриманого каналу, оскільки у разі помилки `netsh` повертає рядок, що містить “IPsec”.

Програмний код функції `add_hosts()` наведено у лістингу 2.7. Фрагмент програмного коду, який відповідає за додавання хостів та задавання власного імені хосту, наведено у лістингу А.4.

На цьому первинне налаштування клієнту Yggdrasil завершено. Залишається лише записати зміни у файл `hosts` та конфіг-файл `C:\ProgramData\YggVPN\yggvpn.conf`, у якому зберігається поточний стан налаштувань хосту. Потім цей же файл звичайним копіюванням може бути експортований у `yggvpn_export.conf`, який у свою чергу використовується для автоматичної конфігурації — при розміщенні його у поточній робочій директорії та запуску функції `install()` дані про публічні вузли та хости беруться з нього замість введення їх користувачем.

Запис налаштувань у конфіг-файл `C:\ProgramData\YggVPN\yggvpn.conf` виконується функцією `update_yggvpn_config()`. Функція записує у файл поточний вміст структури `config` за допомогою функції форматованого виведення `fprintf()`.

Лістинг 2.7 — Програмний код функції add_hosts()

```

void add_hosts() {
    char* buf = malloc(40*sizeof(char));
    for(;;){
        printf("Host (f to finish): ");
        scanf("%39s", buf);
        if(strlen(buf) == 1 && strcmp(buf, "f") == 0)
            break;
        if(find(PCONF->hosts, PCONF->host_count, buf) != -1){
            puts("Host already present.");
            continue;
        }
        if(!firewall_add_host(buf)){
            puts("Invalid host.");
            continue;
        }
        append(&PCONF->hosts, PCONF->host_count, strdup(buf));
        char input; char* hostname; char* buf2;
        clear();
        for(;;){
            printf("Hostname (optional): ");
            input = getchar();
            if(input != '\n'){
                ungetc(input, stdin);
                buf2 = malloc(64*sizeof(char));
                scanf("%63s", buf2);
                clear();
                if(find(PCONF->hostnames, PCONF->host_count, buf2) != -1){
                    puts("Hostname already present.");
                    free(buf2);
                    continue;
                }
                hostname = strdup(buf2);
                free(buf2);
            }
            else
                hostname = strdup(" ");
            break;
        }
        append(&PCONF->hostnames, PCONF->host_count, hostname);
        PCONF->host_count++;
    }
    free(buf);
}

```

Для запису хостів у файл `hosts` призначена функція `update_hosts_file()`. На початку виконання вона зчитує вміст файлу у пам'ять, після чого перевіряє наявність попередніх записів утиліти на його початку, секція з якими позначається рядками-маркерами `"# YGGVPN"` до та після. Якщо попередні записи наявні, вони затираються переміщенням покажчику на початок змісту файлу у пам'яті. Після цього файл `hosts` перезаписується новими маркерами, записами хостів з іменами зі структури `config` та попереднім вмістом.

Програмний код функцій оновлення файлів `C:\ProgramData\YggVPN\yggvpn.conf` та `hosts` наведено у лістингах 2.8 та 2.9 відповідно.

Лістинг 2.8 — Програмний код функції `update_yggvpn_config()`

```
void update_yggvpn_config()
{
    FILE* fp = fopen(YGGVPN_CONFIG_DIR "\\yggvpn.conf", "w");
    int i;
    fprintf(fp, "[peers]\n");
    fprintf(fp, "peers = ");
    for(i = 0; i < PCONF->peer_count - 1; i++)
        fprintf(fp, "%s,", PCONF->peers[i]);
    fprintf(fp, "%s\n\n", PCONF->peers[i]);
    fprintf(fp, "[hosts]\n");
    fprintf(fp, "hosts = ");
    for(i = 0; i < PCONF->host_count - 1; i++)
        fprintf(fp, "%s,", PCONF->hosts[i]);
    fprintf(fp, "%s\n", PCONF->hosts[i]);
    fprintf(fp, "hostnames = ");
    for(i = 0; i < PCONF->host_count - 1; i++)
        fprintf(fp, "%s,", PCONF->hostnames[i]);
    fprintf(fp, "%s\n", PCONF->hostnames[i]);
    fclose(fp);
}
```

Лістинг 2.9 — Програмний код функції update_hosts_file()

```

int update_hosts_file()
{
    FILE* fp = fopen("C:\\Windows\\System32\\drivers\\etc\\hosts",
"r+");
    if(fp == NULL)
    {
        puts("ERROR: Could not open C:\\Windows\\System32\\
drivers\\etc\\hosts.");
        return -1;
    }
    fseek(fp, 0, SEEK_END);
    int file_size = ftell(fp);
    rewind(fp);
    char* file_content = malloc(file_size + sizeof(char*));
    fread(file_content, 1, file_size, fp);
    file_content[file_size] = '\\0'; // For fputs()
    fp = freopen("C:\\Windows\\System32\\drivers\\etc\\hosts", "w",
fp); //Erase old contents
    fputs("# DO NOT EDIT THIS SECTION OR ADD ANYTHING ABOVE IT\\
n"HOSTS_FILE_SECTION_MARKER"\\n", fp);
    for(int i = 0; i < PCONF->host_count; i++)
    {
        if(!strcmp(PCONF->hostnames[i], " "))
            continue;
        fprintf(fp, "%s\\t%s\\n", PCONF->hosts[i], PCONF->hostnames[i]);
    }
    fputs(HOSTS_FILE_SECTION_MARKER"\\n\\n", fp);
    char* offset = strstr(file_content, HOSTS_FILE_SECTION_MARKER);
    if(offset != NULL)
    {
        offset = strstr(offset + 1, HOSTS_FILE_SECTION_MARKER) +
strlen(HOSTS_FILE_SECTION_MARKER) + 2; // Find the second occurrence
including two \\n characters
        fputs(offset, fp); //Write back everything except old
hosts info
    }
    else
        fputs(file_content, fp);
    free(file_content);
    fclose(fp);
    system("ipconfig /flushdns > nul"); // Update DNS cache
    return 0;
}

```

Протягом процесу встановлення програма читає та редагує структуру config через покажчик PCONF. Ця структура містить поточний стан налаштувань хосту, а саме:

- кількість публічних вузлів;
- масив рядків URL публічних вузлів;
- кількість хостів;
- масив рядків з адресами хостів;
- масив рядків з символьними іменами хостів.

Для роботи з масивами визначено наступні функції:

- find() - виконує пошук рядку у масиві рядків, повертає його індекс або -1 якщо переданого рядку не існує;
- append() - додає рядок до масиву рядків, переміщуючи масив у пам'яті;
- pop() - видаляє рядок за переданим індексом з масиву рядків, переміщуючи масив у пам'яті;
- extend() - розширює один масив іншим, переміщуючи його у пам'яті.

Опис структури config наведено у лістингу 2.10, а програмний код функцій роботи з масивами — у лістингу A.5.

Лістинг 2.10 — Опис структури config

```
typedef struct
{
    int peer_count;
    char** peers;
    int host_count;
    char** hosts;
    char** hostnames;
} config;

static config* PCONF = NULL;
```

2.2 Головна функція та меню

На початку виконання програма перевіряє наявність прав адміністратора, оскільки більшість необхідних дій без них неможлива. Це виконується за допомогою системного виклику `CheckTokenMembership()` шляхом перевірки належності користувача до групи адміністраторів, SID якої отримується системним викликом `AllocateAndInitializeSid()`.

Програмний код функції `is_user_admin()` наведено у лістингу 2.11.

Лістинг 2.11 — Програмний код функції `is_user_admin()`

```

BOOL is_user_admin()
{
    BOOL b;
    SID_IDENTIFIER_AUTHORITY NtAuthority = SECURITY_NT_AUTHORITY;
    PSID AdministratorsGroup;
    b = AllocateAndInitializeSid(&NtAuthority, 2,
SECURITY_BUILTIN_DOMAIN_RID, DOMAIN_ALIAS_RID_ADMINS, 0, 0, 0, 0, 0,
0, &AdministratorsGroup);
    if (b)
    {
        if (!CheckTokenMembership(NULL, AdministratorsGroup, &b))
        {
            b = FALSE;
        }
        FreeSid(AdministratorsGroup);
    }
    return (b);
}

```

Далі ініціалізується структура `config` та перевіряється наявність файлу `C:\ProgramData\YggVPN\yggvpn.conf`. Якщо його не існує, утиліта запускається вперше і виконується процедура встановлення. При цьому, за наявності файлу `yggvpn_export.conf` у поточній робочій директорії, структура `config` заповнюється даними з нього за допомогою бібліотеки `inih` та callback-функції `handler()`, змінна

EXPORTED_CONF встановлюється у значення TRUE і встановлення проходить за даними структури [12]. Функція викликається бібліотекою для кожного запису у файлі зі знаком "=", при цьому їй передаються імена секції та опції. Відповідно до них, функція зчитує значення та заповнює відповідні поля структури config. Для цього отримані рядки необхідно розділити на підрядки, оскільки у файлі URL публічних вузлів, адреси та імена хостів розділені комами. Для визначення кількості таких підрядків призначена функція strcnt(), програмний код якої наведено у лістингу А.5. Безпосередньо розділення рядків на підрядки виконується функцією strtok() бібліотеки string.h. У випадку зчитування імен хостів додатково вводяться перевірки порожніх значень — у такому разі до масиву імен хостів заносяться рядки " ".

Якщо файл C:\ProgramData\YggVPN\yggvpn.conf існує, Yggdrasil вже встановлено та налаштовано. Тоді структура config аналогічно заповнюється даними з цього файлу та викликається функція меню.

Програмний код функції handler() наведено у лістингу А.6. Програмний код головної функції наведено у лістингу 2.12.

Функція меню виводить список доступних команд та запрошення, після чого оброблює команди, які вводять користувач, викликаючи необхідні функції. Команди складаються з одного або двох символів. При введенні невірної команди виводиться список доступних команд.

Після виклику функцій редагування списків публічних вузлів та хостів функція меню оновлює файли C:\ProgramData\Yggdrasil\yggdrasil.conf, hosts та C:\ProgramData\YggVPN\yggvpn.conf за допомогою відповідних функцій, при чому це виконується лише після введення користувачем команди "f", яка завершує роботу функції редагування. У випадку редагування списку публічних вузлів додатково виконується перезапуск служби Yggdrasil.

Список доступних команд утиліти наведено у табл. 2.1. Програмний код функції виведення доступних команд наведено у лістингу 2.13, а функції меню — у лістингу А.7.

Лістинг 2.12 — Програмний код функції main()

```

int main() {
    if(!is_user_admin()){
        puts("ERROR: Administrative permissions required.\n");
        system("pause");
        return -1;
    }
    puts("YggVPN Installation and Configuration Utility\n");
    PCONF = malloc(sizeof(config));
    PCONF->peer_count = 0;
    PCONF->host_count = 0;
    if(!PathFileExistsA(YGGVPN_CONFIG_DIR"\\yggvpn.conf")){
        puts("YggVPN not installed, initializing installation . . . ");
        CreateDirectoryExA("C:\\ProgramData\\Microsoft",
YGGVPN_CONFIG_DIR, NULL);
        if(PathFileExistsA(".\\yggvpn_export.conf")){
            printf("Exported config found, importing it . . . ");
            if(ini_parse(".\\yggvpn_export.conf", handler, PCONF) < 0)
            {
                free(PCONF);
                puts("ERROR: Invalid config.");
                system("pause");
                return -1;
            }
            EXPORTED_CONF = TRUE;
            puts("Done.");
        }
        if(install()){
            free(PCONF);
            return -1;
        }
    }
    else if(ini_parse(YGGVPN_CONFIG_DIR"\\yggvpn.conf", handler, PCONF) < 0)
    {
        free(PCONF);
        return -1;
    }
    menu();
    config_reset(PCONF);
    free(PCONF);
    return 0;
}

```

Таблиця 2.1 — Список доступних команд утиліти

Команда	Опис
i	Встановлення/Оновлення YggVPN
pa	Виведення власної Yggdrasil IPv6-адреси
ah	Додавання хостів
ph	Виведення хостів
eh	Редагування символьних імен хостів
dh	Видалення хостів
ap	Додавання публічних вузлів
pp	Виведення публічних вузлів
dp	Видалення публічних вузлів
ec	Експорт конфігурації
h	Виведення списку доступних команд
u	Видалення YggVPN
q	Вихід з утиліти

Лістинг 2.13 — Програмний код функції print_help()

```

void print_help()
{
    puts("\n  i\tInstall/Reinstall/Update YggVPN");
    puts("  pa\tPrint own address");
    puts("  ah\tAdd hosts");
    puts("  ph\tPrint hosts");
    puts("  eh\tEdit hostnames");
    puts("  dh\tDelete hosts");
    puts("  ap\tAdd public peers");
    puts("  pp\tPrint public peers");
    puts("  dp\tDelete public peers");
    puts("  ec\tExport config");
    puts("  h\tPrint this help");
    puts("  u\tUninstall YggVPN");
    puts("  q\tQuit\n");
}

```

2.3 Виведення списків публічних вузлів та хостів

Для виведення списків налаштованих публічних вузлів та хостів визначено функції `print_hosts()` та `print_peers()`, які наведено у лістингах 2.14 та 2.15 відповідно. Функції перебирають масиви рядків URL публічних вузлів, адрес та символічних імен хостів відповідно. При виведенні списку хостів власні адреса та ім'я мають позначку “(you)”.

Лістинг 2.14 — Програмний код функції `print_hosts()`

```
void print_hosts()
{
    puts("\nHosts:\n");
    int own_hostname_index = find(PCONF->hosts, PCONF->host_count,
get_own_yggdrasil_address());
    for(int i = 0; i < PCONF->host_count; i++)
    {
        printf("%d  %s\t", i + 1, PCONF->hosts[i]);
        if(strlen(PCONF->hosts[i]) < 37)
            printf("\t");
        printf("%s", PCONF->hostnames[i]);
        if(i == own_hostname_index)
            puts(" (you)");
        else
            puts("");
    }
    puts("");
}
```

Лістинг 2.15 — Програмний код функції `print_peers()`

```
void print_peers()
{
    puts("\nPeers:\n");
    for(int i = 0; i < PCONF->peer_count; i++)
        printf("%d  %s\n", i + 1, PCONF->peers[i]);
    puts("");
}
```

2.4 Редагування списків публічних вузлів та хостів

Публічні вузли та хости додаються тими ж самими функціями, які використовувалися для цього у процедурі встановлення. Для редагування символьних імен хостів призначена функція `edit_hostnames()`, програмний код якої наведено у лістингу А.8. Функція виводить список хостів викликом функції `print_hosts()`, після чого за номером у списку користувач обирає необхідний запис то вводить нове ім'я. Виконується перевірка входження номеру у діапазон номерів хостів, після чого він переводиться у індекс відніманням від нього одиниці. Далі виконується перевірка наявності введенного імені у масиві імен хостів за допомогою функції `find()` та заміна рядку за індексом. Якщо користувачем у якості нового імені було введено порожній рядок, рядок за індексом замінюється на “ ”. Якщо замість номеру хосту користувач вводить рядок “f”, функція завершує свою роботу.

Видалення публічних вузлів та хостів можливе за допомогою функцій `delete_peers()` та `delete_hosts()`, програмний код яких наведено у лістингах 2.16 та 2.17 відповідно. Аналогічно функції `edit_hostnames()`, вони отримують номер від користувача, переводять його у індекс масиву та видаляють відповідний рядок з масиву функцією `pop()` і декрементують кількість публічних вузлів або хостів відповідно.

У випадку видалення хостів для видалення їх з білого списку фаєрволу застосовується функція `firewall_delete_host()`, програмний код якої наведено у лістингу А.3. Як можна побачити, ця функція повністю видаляє білий список хостів з політики безпеки, після чого додає його заново з усіма хостами, окрім видаленого. Така реалізація пов'язана з виявленням багу в утиліті `netsh`, через який неможливо видалити окремий фільтр зі списку при використанні IPv6-адрес.

Лістинг 2.16 — Програмний код функції delete_peers()

```

void delete_peers()
{
    char input;
    int index;
    for(;;)
    {
        print_peers();
        printf("Peer index to delete (f to finish): ");
        input = getchar();
        if(input == '\n')
            continue;
        if(input == 'f')
        {
            clear();
            break;
        }
        else
        {
            ungetc(input, stdin);
            scanf("%d", &index);
            clear();
            if(PCONF->peer_count < 2)
            {
                puts("You must leave at least one peer.");
                continue;
            }
            if(index < 1 || index > PCONF->peer_count)
            {
                puts("Invalid index.");
                continue;
            }
            index--;
            pop(&PCONF->peers, PCONF->peer_count, index);
            PCONF->peer_count--;
        }
    }
}

```

Лістинг 2.17 — Програмний код функції delete_hosts()

```

void delete_hosts()
{
    char input;
    int index;
    int own_hostname_index;
    for(;;)
    {
        print_hosts();
        printf("Host index to delete (f to finish): ");
        input = getchar();
        if(input == '\n')
            continue;
        if(input == 'f')
        {
            clear();
            break;
        }
        else
        {
            ungetc(input, stdin);
            own_hostname_index = find(PCONF->hosts, PCONF-
>host_count, get_own_yggdrasil_address());
            scanf("%d", &index);
            clear();
            if(index < 1 || index > PCONF->host_count)
            {
                puts("Invalid index.");
                continue;
            }
            index--;
            if(index == own_hostname_index)
            {
                puts("Cannot delete self.");
                continue;
            }
            firewall_delete_host(PCONF->hosts, PCONF->host_count, index);
            pop(&PCONF->hosts, PCONF->host_count, index);
            pop(&PCONF->hostnames, PCONF->host_count, index);
            PCONF->host_count--;
        }
    }
}

```

2.5 Процедура видалення

Для видалення клієнту Yggdrasil та відповідних налаштувань визначено функцію `uninstall()`, програмний код якої наведено у лістингу А.9. Функція зупиняє службу і видаляє клієнт Yggdrasil за допомогою файлу `C:\ProgramData\YggVPN\yggdrasil.msi`, збереженого після встановлення, та утиліти `msiexec`. Після цього системним викликом `DeleteFileA()` видаляється файл `C:\ProgramData\YggVPN\yggvpn.conf`. Налаштування фаєрволу видаляються за допомогою функції `firewall_remove()`, програмний код якої наведено у лістингу А.3. Ця функція послідовно видаляє попередньо визначені правила, списки і дії фільтрів та саму політику безпеки.

Прописані у файлі `hosts` записи хостів видаляються аналогічно функції `update_hosts_file()`, тобто шляхом зчитування вмісту файлу у пам'ять, зміщення покажчику початку рядку поза позначену секцію та перезапису файлу.

2.6 Маніфест додатку та компіляція

У маніфесті формату XML вказуються метадані про додаток, зокрема рівень виконання [13]. Зміст файлів `yggvpn.exe.manifest` та `yggvpn.rc` наведено у лістингах 2.18 та 2.19 відповідно. Запис “`<requestedExecutionLevel level="requireAdministrator" uiAccess="false"/>`” змушує додаток виконуватися з правами адміністратора на системах з увімкненим контролем облікових записів користувачів (UAC) (рис 2.1).

Лістинг 2.18 — Вміст файлу yggvpn.exe.manifest

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<assembly xmlns="urn:schemas-microsoft-com:asm.v1"
manifestVersion="1.0">
    <assemblyIdentity version="1.0.0.0" processorArchitecture="X86"
name="YggVPN Installation and Configuration Utility" type="win32"/>
    <description>YggVPN</description>
    <trustInfo xmlns="urn:schemas-microsoft-com:asm.v2">
        <security>
            <requestedPrivileges>
                <requestedExecutionLevel
level="requireAdministrator" uiAccess="false"/>
            </requestedPrivileges>
        </security>
    </trustInfo>
</assembly>
```

Лістинг 2.19 — Вміст файлу yggvpn.rc

```
1 24 "yggvpn.exe.manifest"
```

Компіляцію додатку виконано за допомогою набору інструментів i686-w64-mingw32 у середовищі Cygwin [14]. Для цього використано команди, наведені у лістингу 2.20. Перша команда збирає ресурси (файл маніфесту) у об'єктний файл yggvpn.o, друга виконує безпосередньо компіляцію вихідного коду і статичне компонування цього файлу, коду та бібліотек у виконуваний файл yggvpn.exe.

Лістинг 2.20 — Вміст файлу compile.bat

```
i686-w64-mingw32-windres src/yggvpn.rc build/yggvpn.o
i686-w64-mingw32-gcc -m32 src/yggvpn.c build/yggvpn.o -o build/yggvp
n.exe-static -L lib/ -lurlmon -lws2_32 -lshlwapi -ljson-c
```

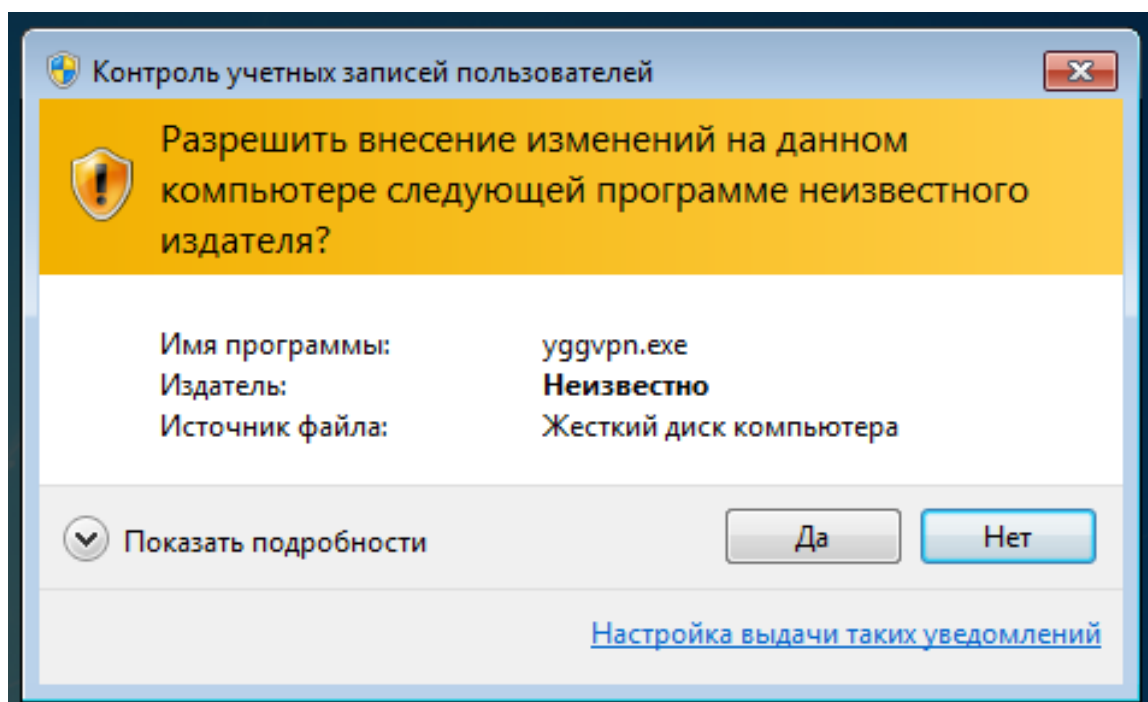


Рисунок 2.1 — Вікно підтвердження виконання програми з правами адміністратора

2.7 Вигляд готової утиліти

На рис. 2.2 наведено процес встановлення YggVPN без експортованого конфіг-файлу, виведення списків вузлів та хостів і експорт конфіг-файлу. У процесі встановлення користувачем вводяться URL двох публічних вузлів, підтвердження додавання хостів “у”, адреса другого хосту та його символічне ім’я “vm.ygg” і власне ім’я хосту “me.ygg”. Далі командою “pp” виводиться список публічних вузлів, командою “ph” — список хостів. Команда “ec” виконує експорт конфігурації хосту у файл yggvpn_export.conf.

```

YggVPN not installed, initializing installation . . .
Checking latest Yggdrasil version . . . 0.4.7
Downloading Yggdrasil v0.4.7 installer for x64 architecture . . . Done.
Installing Yggdrasil . . . Done.
Stopping Yggdrasil service . . . Done.
Configuring Yggdrasil . . .
Please enter URLs of public peers you wish to use. Choose the closest ones to you
from https://publicpeers.neilalexander.dev/, for example.
Peer URL (f to finish): tls://185.165.169.234:8443
Peer URL (f to finish): tcp://193.107.20.230:7743
Peer URL (f to finish): f
Done.
Configuring Yggdrasil service . . . Done.
Starting Yggdrasil service . . . Done.
Setting up the firewall . . . Done.

Your Yggdrasil IPv6 address: 200:826:a249:6679:9b11:5430:ac5e:6c2e
Share your IPv6 address with other hosts you want to have access to this machine.

Add hosts? (y/N): y
Host (f to finish): 200:49b3:f17:9832:3c4c:8bd2:ba2f:4202
Hostname (optional): vm.ygg
Host (f to finish): f

Please enter your desired hostname (optional): me.ygg
Installation complete.

  i    Install/Reinstall/Update YggVPN
 pa    Print own address
 ah    Add hosts
 ph    Print hosts
 eh    Edit hostnames
 dh    Delete hosts
 ap    Add public peers
 pp    Print public peers
 dp    Delete public peers
 ec    Export config
 h     Print this help
 u     Uninstall YggVPN
 q     Quit

(h for help)> pp

Peers:

1  tls://185.165.169.234:8443
2  tcp://193.107.20.230:7743

(h for help)> ph

Hosts:

1  200:49b3:f17:9832:3c4c:8bd2:ba2f:4202      vm.ygg
2  200:826:a249:6679:9b11:5430:ac5e:6c2e      me.ygg (you)

(h for help)> ec
Config exported to yggvpn_export.conf
(h for help)>

```

Рисунок 2.2— Встановлення без експортованого конфіг-файлу, виведення списків вузлів та хостів, експорт конфіг-файлу

На рис. 2.3 наведено процес видалення YggVPN. Для цього необхідно ввести команду “u” та згоду підтвердження видалення “y”. Після видалення утиліта очікує введення та закривається при натисканні будь-якої клавіші.

```
(h for help)> u
Are you sure you want to uninstall YggVPN? (y/N): y
Stopping Yggdrasil service . . . Done.
Uninstalling Yggdrasil . . . Done.
Deleting YggVPN config . . . Done.
Removing the firewall . . . Done.
Cleaning up hosts entries . . . Done.
YggVPN uninstalled successfully.
Для продовження натисніть будь-яку клавішу . . .
```

Рисунок 2.3 — Видалення YggVPN

На рис. 2.4 наведено процес встановлення з попередньо експортованим конфіг-файлом. Процес ідентичний наведеному на рис. 2.2, але від користувача потребується ввести лише власне ім'я хосту, або, у даному випадку, підтвердження застосування вказаного у конфіг-файлі власного імені хосту (порожній рядок). Усі інші налаштування беруться з експортованої конфігурації.

На рис. 2.5 продемонстровано процес додавання та видалення публічного вузла. Для додавання вузла вводиться команда “ap”, його URL, та команда “f” для повернення до меню. Для видалення вводиться команда “dp”, номер вузла зі списку та команда “f” для повернення до меню.

На рис. 2.6 продемонстровано процес додавання, редагування та видалення хосту. Хост додається введенням команди “ah”, адреси та імені хосту, що додається, та команди “f” для повернення до меню. Редагування хосту виконується введенням команди “eh”, номеру редагованого хосту зі списку, нового імені та команди “f” для повернення до меню. Аналогічно редагуванню виконується видалення.

```

YggVPN Installation and Configuration Utility

YggVPN not installed, initializing installation . . .
Exported config found, importing it . . . Done.
Checking latest Yggdrasil version . . . 0.4.7
Downloading Yggdrasil v0.4.7 installer for x64 architecture . . . Done.
Installing Yggdrasil . . . Done.
Stopping Yggdrasil service . . . Done.
Configuring Yggdrasil . . . Done.
Configuring Yggdrasil service . . . Done.
Starting Yggdrasil service . . . Done.
Setting up the firewall . . . Done.

Your Yggdrasil IPv6 address: 200:826:a249:6679:9b11:5430:ac5e:6c2e
Share your IPv6 address with other hosts you want to have access to this machine.

Adding host 200:49b3:f17:9832:3c4c:8bd2:ba2f:4202 . . . Done.

Your hostname is set as me.ygg. Enter a new one to change it or nothing to leave as is:
Installation complete.

i      Install/Reinstall/Update YggVPN
pa     Print own address
ah     Add hosts
ph     Print hosts
eh     Edit hostnames
dh     Delete hosts
ap     Add public peers
pp     Print public peers
dp     Delete public peers
ec     Export config
h      Print this help
u      Uninstall YggVPN
q      Quit

(h for help)> pp

Peers:

1  tls://185.165.169.234:8443
2  tcp://193.107.20.230:7743

(h for help)> ph

Hosts:

1  200:49b3:f17:9832:3c4c:8bd2:ba2f:4202      vm.ygg
2  200:826:a249:6679:9b11:5430:ac5e:6c2e      me.ygg (you)

(h for help)>

```

Рисунок 2.4 — Встановлення з експортованим конфіг-файлом

```
(h for help)> pp
Peers:
1  tls://185.165.169.234:8443
2  tcp://193.107.20.230:7743

(h for help)> ap
Peer URL (f to finish):  tls://193.111.114.28:1443
Peer URL (f to finish): f
Restarting Yggdrasil service . . . Done.
(h for help)> dp
Peers:
1  tls://185.165.169.234:8443
2  tcp://193.107.20.230:7743
3  tls://193.111.114.28:1443

Peer index to delete (f to finish): 3
Peers:
1  tls://185.165.169.234:8443
2  tcp://193.107.20.230:7743

Peer index to delete (f to finish): f
Restarting Yggdrasil service . . . Done.
(h for help)>
```

Рисунок 2.5 — Додавання та видалення публічного вузлу

```
(h for help)> ah
Host (f to finish): 222:a8e4:50cd:55c:788e:b0a5:4e2f:a92c
Hostname (optional):
Host (f to finish): f
(h for help)> eh
Hosts:
1  200:49b3:f17:9832:3c4c:8bd2:ba2f:4202      vm.ygg
2  200:826:a249:6679:9b11:5430:ac5e:6c2e      me.ygg (you)
3  222:a8e4:50cd:55c:788e:b0a5:4e2f:a92c

Host index (f to finish): 3
New hostname (Enter to leave empty): howto.ygg
Hosts:
1  200:49b3:f17:9832:3c4c:8bd2:ba2f:4202      vm.ygg
2  200:826:a249:6679:9b11:5430:ac5e:6c2e      me.ygg (you)
3  222:a8e4:50cd:55c:788e:b0a5:4e2f:a92c      howto.ygg

Host index (f to finish): f
(h for help)> dh
Hosts:
1  200:49b3:f17:9832:3c4c:8bd2:ba2f:4202      vm.ygg
2  200:826:a249:6679:9b11:5430:ac5e:6c2e      me.ygg (you)
3  222:a8e4:50cd:55c:788e:b0a5:4e2f:a92c      howto.ygg

Host index to delete (f to finish): 3
Hosts:
1  200:49b3:f17:9832:3c4c:8bd2:ba2f:4202      vm.ygg
2  200:826:a249:6679:9b11:5430:ac5e:6c2e      me.ygg (you)

Host index to delete (f to finish): f
(h for help)>
```

Рисунок 2.6 — Додавання, редагування та видалення хосту

3 ПРОЕКТУВАННЯ СИСТЕМИ НА КОНКРЕТНОМУ ПРИКЛАДІ

3.1 Аналіз структури наявної системи об'єкту

Структуру наявної системи об'єкту наведено на рис. 3.1. Керування підсистемами об'єкту здійснюється комп'ютером PC0 з робочих місць PC1, PC2 та PC3, які являють собою тонкі клієнти для керування PC0. Для цього застосовується KVM-over-IP комутатор, підключений до PC0 інтерфейсами VGA для отримання зображення та USB для емуляції миші і клавіатури. KVM-over-IP комутатор та робочі місця PC1-3 з'єднані у мережу за допомогою FastEthernet комутатора S1.

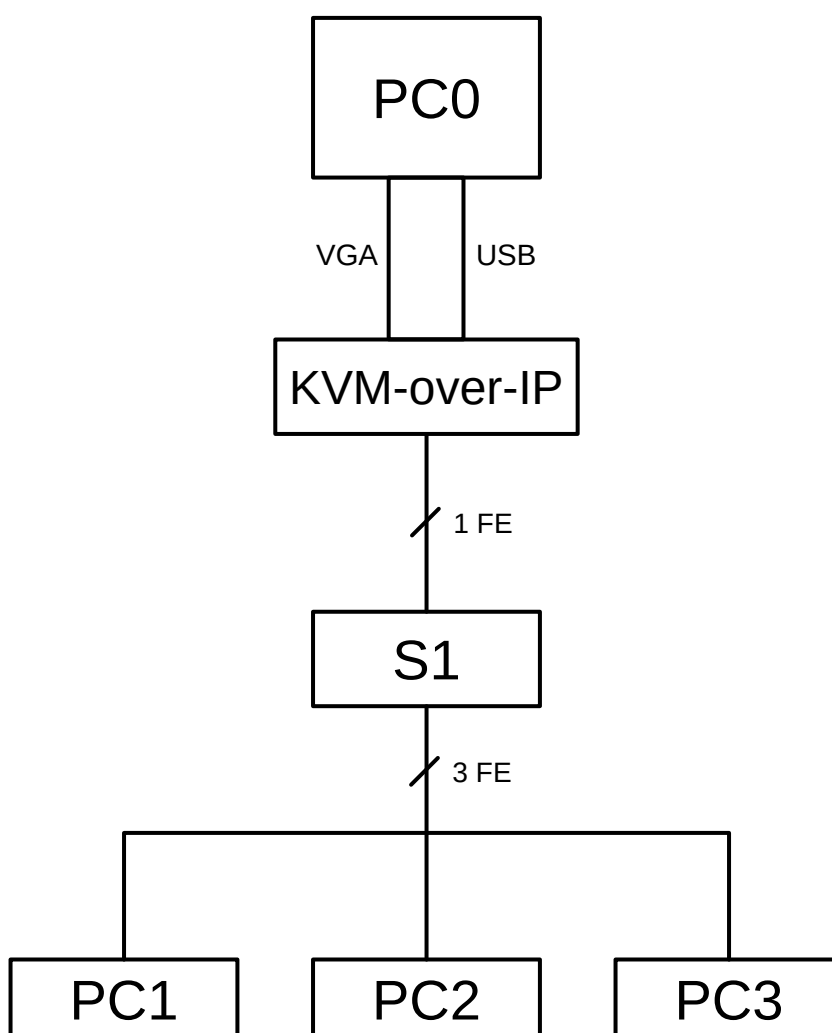


Рисунок 3.1 — Структура наявної системи

Пристрої мають наступні IP-адреси:

- PC1 — 192.168.1.11/24;
- PC2 — 192.168.1.22/24;
- PC3 — 192.168.1.33/24;
- KVM-over-IP — 192.168.1.100/24.

Мережа є ізольованою і не має жодних з'єднань з Інтернет.

На робочих місцях встановлені наступні операційні системи:

- PC1 — Windows 7 32bit;
- PC2 — Windows 7 64bit;
- PC3 — Windows 10 64bit;

Комутатор KVM-over-IP працює як VNC-сервер, робочі місця підключаються до нього за допомогою встановленого клієнту TigerVNC.

3.2 Організація виходу у Інтернет

Оскільки мережа об'єкту є ізольованою від глобальної мережі, для віддаленого доступу першочергово необхідно забезпечити виходи у Інтернет. Для цього застосуємо три мобільні телефони під управлінням Android. Для більшої надійності системи телефони використовують різних мобільних операторів зв'язку (Vodafone, Kyivstar, lifecell).

Телефони підключаються до робочих місць за допомогою USB-to-microUSB кабелів. У налаштуваннях Android необхідно увімкнути режим модему (USB tethering) (рис. 3.2). Після цього на робочих місцях з'явиться новий мережевий адаптер “Ethernet” з описом “Remote NDIS based Internet Sharing Device” (рис. 3.3) [15].

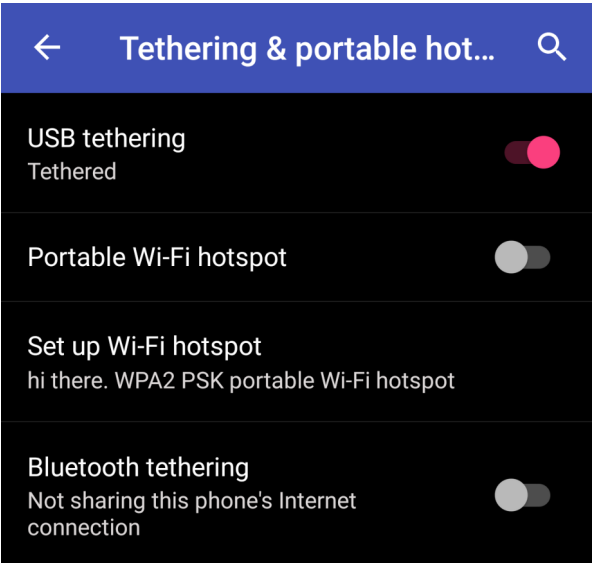


Рисунок 3.2 — Увімкнення режиму модема

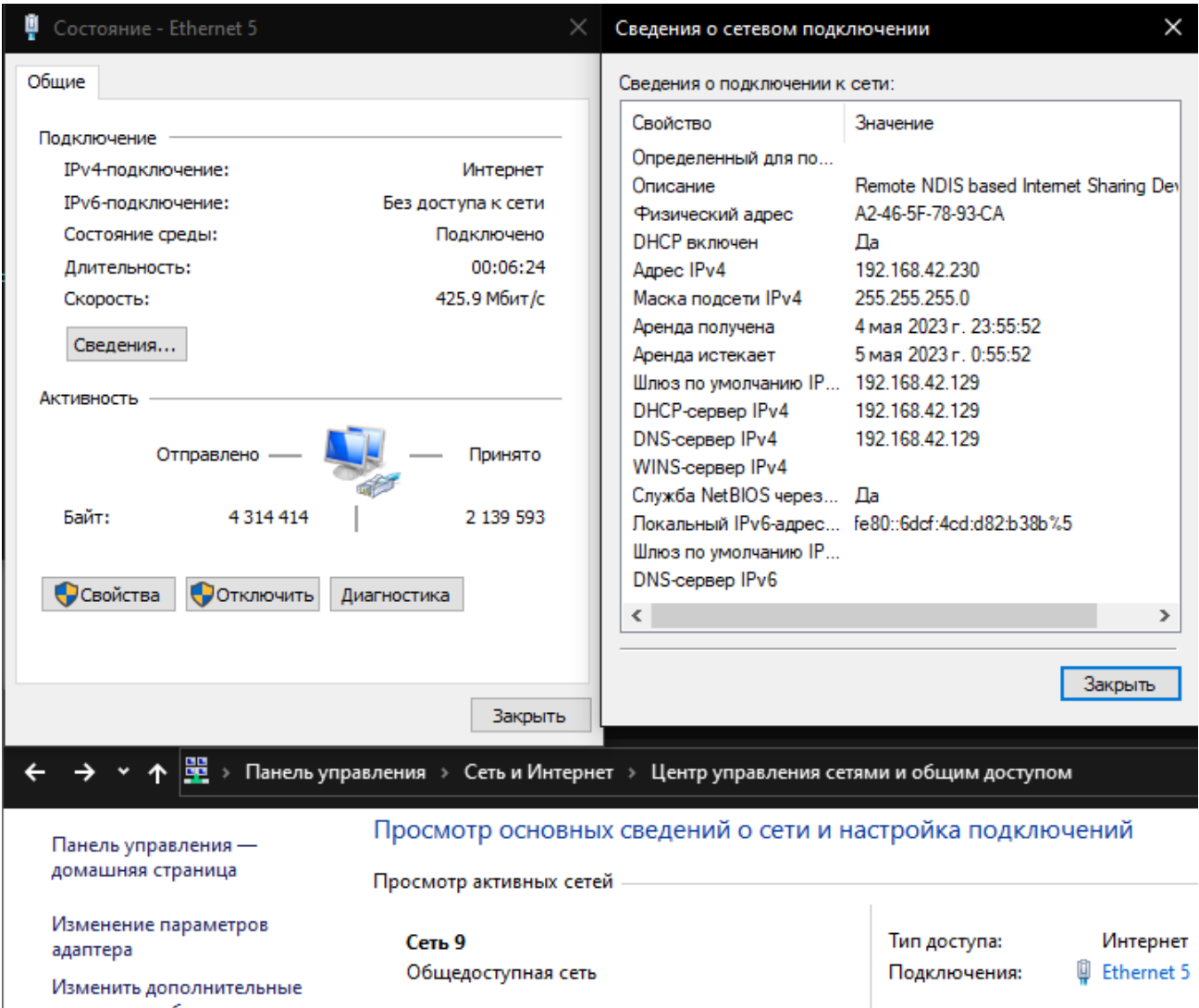


Рисунок 3.3 — Нове мережеве з’єднання RNDIS

3.3 Обирання публічних вузлів Yggdrasil

Орієнтовний список публічних вузлів мережі Yggdrasil ведеться на сайті розробника — <https://publicpeers.neilalexander.dev/> (рис. 3.4). Хоча список містить лише ті вузли, які виявили бажання бути таким чином анонсованими, його зручно використовувати для підключення до мережі завдяки іншій корисній інформації про вузли, а саме:

- географічне розташування вузла (країна);
- необхідну версію клієнта;
- статус вузла (онлайн/офлайн);
- орієнтовну надійність.

Статус вузлів оновлюється щогодини.

poland

<code>tls://54.37.137.221:11129</code>	$\geq 0.4.0$	online	reliable
<code>tls://[2001:41d0:601:1100::cf2]:11129</code>	$\geq 0.4.0$	online	average
<code>tls://pl1.servers.devices.cwinfo.net:11129</code>	$\geq 0.4.0$	online	very unreliable

Рисунок 3.4 — Публічні вузли мережі Yggdrasil (Польща)

Для системи було обрано наступні чотири публічних вузла, відмічені як “reliable”:

- `tls://193.111.114.28:1443` (Україна);
- `tls://54.37.137.221:11129` (Польща);
- `tcp://193.107.20.230:7743` (Німеччина);
- `tls://185.165.169.234:8443` (Румунія).

Обрані вузли географічно віддалені один від одного, і в той же час є найближчими до об’єкту з міркувань зменшення вірогідності їх одночасної відмови та мінімізації затримки. Для перевірки надійності вузлів було створено bash-скрипт, наведений у лістингу 3.1.

Лістинг 3.1 — Програмний код скрипту longping.sh

```
#!/bin/bash

HOST=$1
if [ -z "$HOST" ]
then
    echo "Usage: ./longping.sh <host> [interval]"
    echo "    host      Host to ping"
    echo "    interval  Time interval between pings in seconds
(default - 30)"
    exit
fi
INTERVAL=$2
if [ -z "$INTERVAL" ]
then
    INTERVAL=30
fi
TX=0
RX=0
TIME=0
CSV="$HOST"_"$(date +%FT%T).csv"
echo "time,state" > $CSV
trap break SIGINT
while true
do
    ((TX++))
    if (ping -c 1 $HOST | grep "1 received" > /dev/null)
    then
        ((RX++))
        TIMESTAMP=$(date +%FT%T)
        echo "$TIMESTAMP,1" >> $CSV
        echo "[$TIMESTAMP] Response from $HOST"
    else
        TIMESTAMP=$(date +%FT%T)
        echo "$TIMESTAMP,0" >> $CSV
        echo "[$TIMESTAMP] No response from $HOST"
    fi

    echo "$TX transmitted, $RX received" > "$HOST"_ping_stats
    sleep $INTERVAL
done
trap SIGINT
echo "$HOST: $TX transmitted, $RX received"
```

Скрипт раз на 30 секунд перевіряє з'єднання зі вказаним вузлом, надсилаючи йому ICMP-пакет. Якщо відповідь не була отримана впродовж наступних 20 секунд, фіксується відсутність зв'язку. Час та статус вузла записуються у файл формату CSV, який потім можливо використати для побудови графіків доступності. Додатково ведеться файл зі статистикою перевірки зв'язку, а саме кількістю надісланих та отриманих пакетів. Робота продовжується до зупинки виконання користувачем.

Зібрану статистику доступності обраних публічних вузлів протягом двох тижнів наведено на рис. 3.5. Моменти зникнення зв'язку з вузлом позначено вертикальними чорними лініями.



Рисунок 3.5 — Графіки доступності обраних публічних вузлів

Як видно з рис. 3.5, єдиний випадок близький до одночасної відмови усіх чотирьох вузлів відбувся під кінець 17 квітня. Цей сегмент графіків детальніше наведено на рис. 3.6.

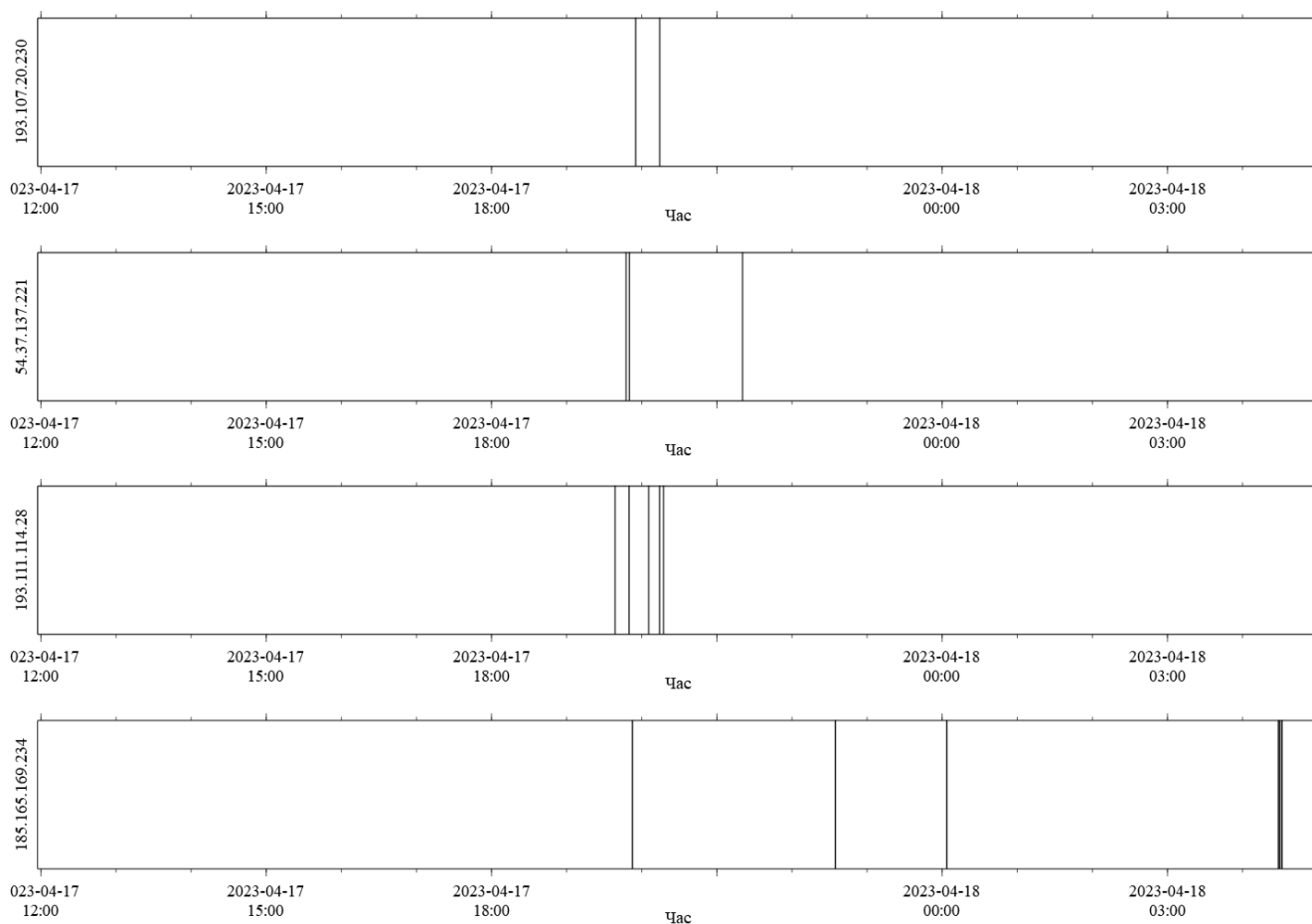


Рисунок 3.6 — Сегмент графіків доступності з 12:00 17.04.2023 по 5:00 18.04.2023

Як видно з наведених графіків доступності, протягом двох тижнів не відбулося жодної одночасної відмови усіх чотирьох обраних публічних вузлів, з чого можна зробити висновок про їх достатню надійність.

На рис. 3.7 наведено структуру логічних з'єднань системи (можливих маршрутів пересилання даних), де:

- PP1-4 — обрані публічні вузли;
- RPC1-3 — віддалені робочі місця.

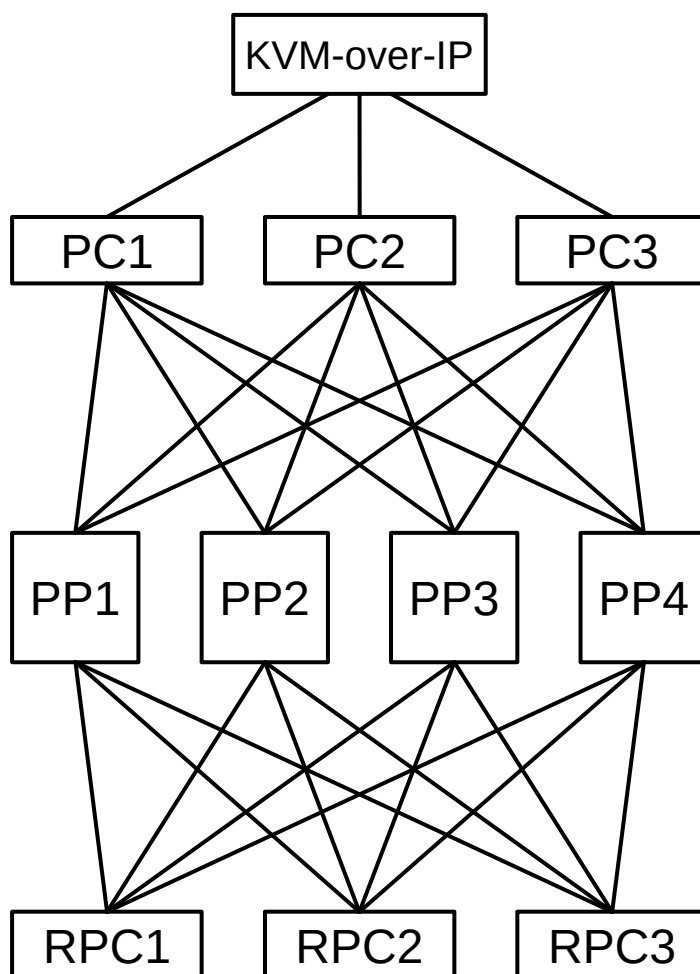


Рисунок 3.7 — Структура логічних з'єднань системи

3.4 Налаштування локальних робочих місць

3.4.1 Встановлення серверу TigerVNC

З наявним доступом у Інтернет на локальні робочі місця встановлюється сервер TigerVNC. Це робиться для можливості віддаленого керування самими робочими місцями при необхідності. У контрольній панелі TigerVNC налаштовується пароль для підключення до робочого місця. Також у вкладці “Inputs” необхідно увімкнути опцію “Send raw keyboard events to applications”, (рис. 3.8). Шифрування з'єднання не потрібне, оскільки усі з'єднання через мережу Yggdrasil шифровані за замовчанням.

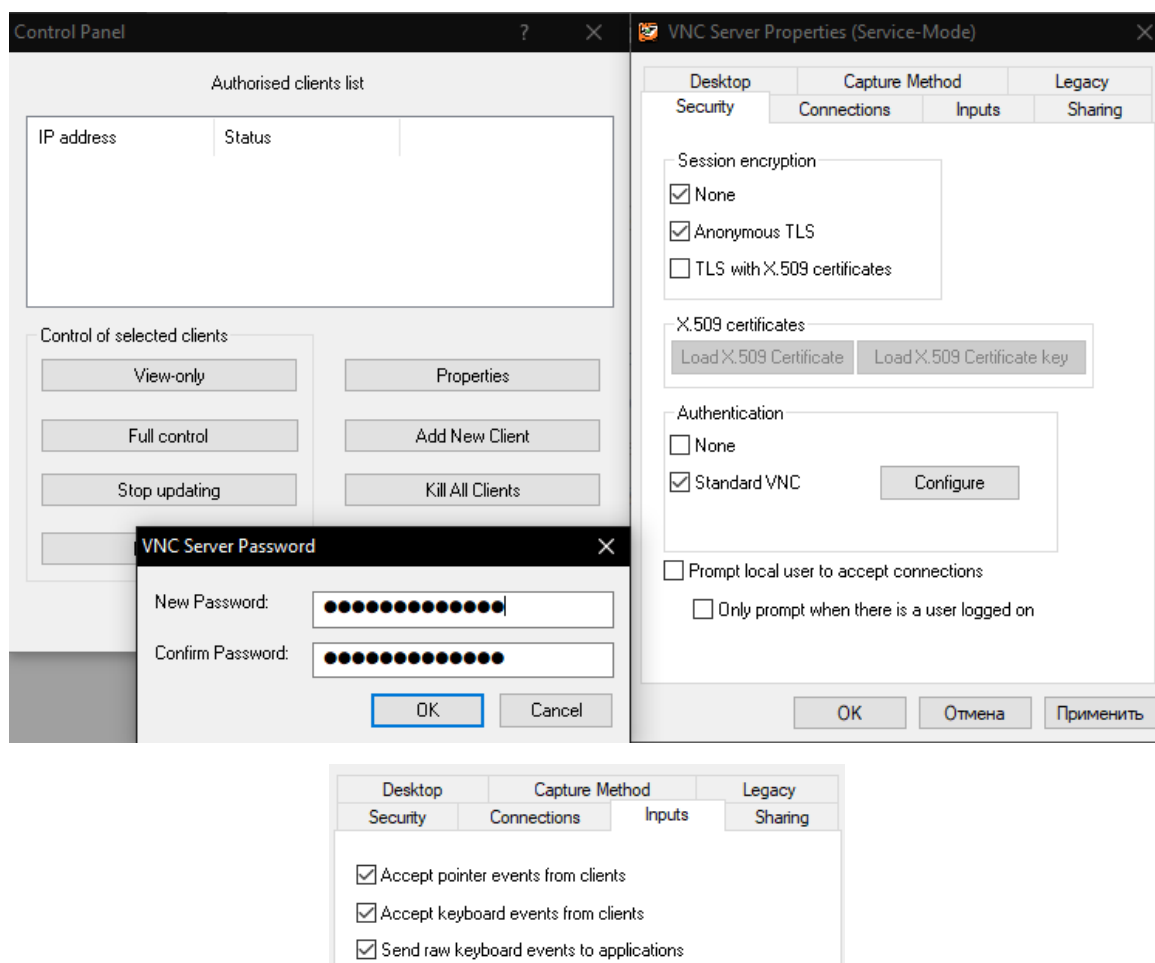


Рисунок 3.8 — Налаштування серверу TigerVNC

3.4.2 Встановлення і налаштування клієнту Yggdrasil

Для встановлення і налаштування клієнту Yggdrasil використовується розроблена утиліта YggVPN. Деяким антивірусним програмним забезпеченням (включаючи стандартний Windows Defender) вона помилково сприймається як загроза, тому перед запуском її потрібно додати до винятків.

На рис. 3.9 продемонстровано виконання YggVPN на робочому місці PC1. При виконанні вводяться URL обраних публічних вузлів та ім'я хосту — pc1.ygg. Після цього налаштування експортуються у yggvpn_export.conf командою “ec”.

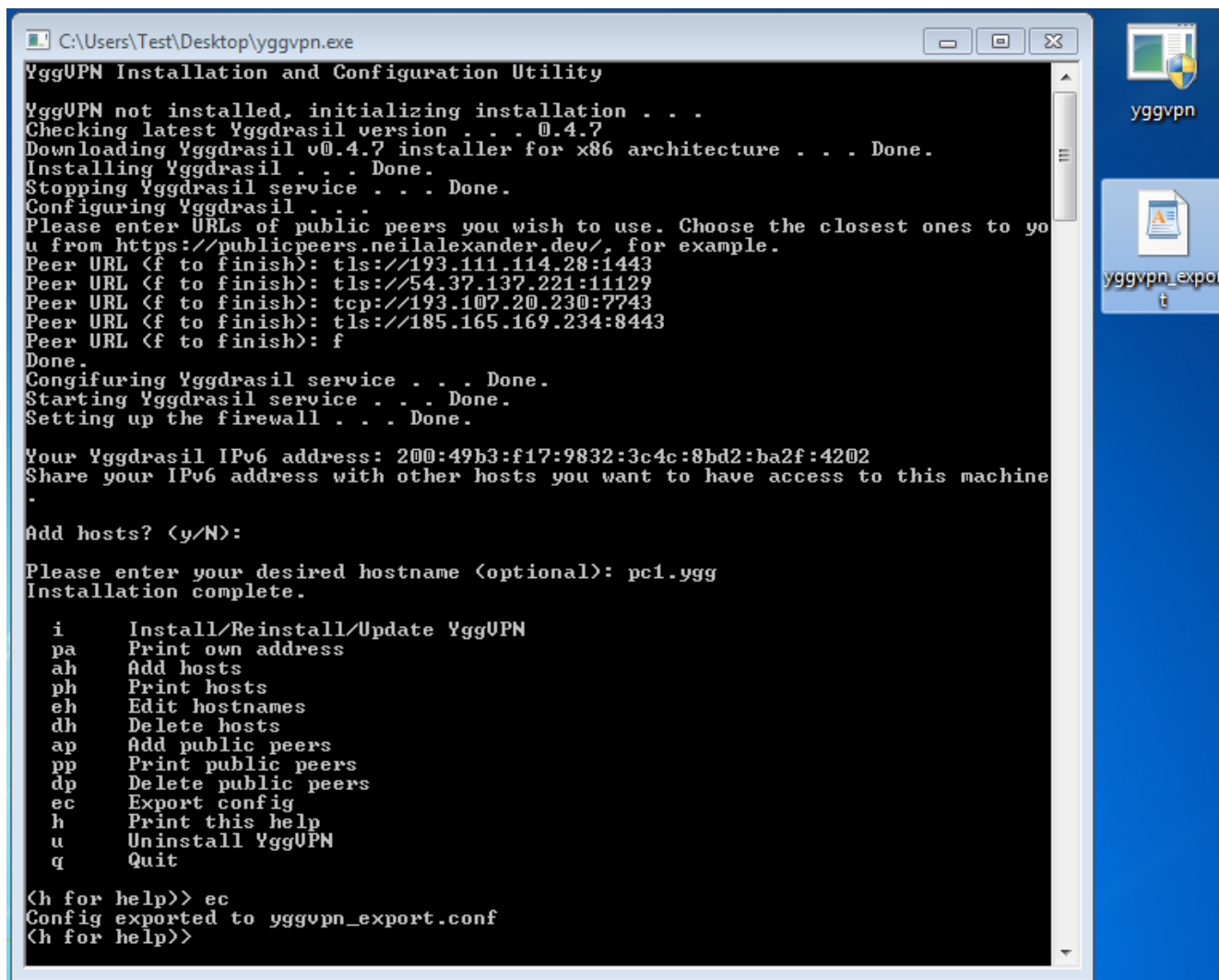


Рисунок 3.9 — Виконання YggVPN на робочому місці PC1

Після цього утиліту можна закрити командою “q”. Налаштування робочого місця PC2 виконується аналогічно, але з конфіг-файлом, експортованим з PC1 (рис. 3.10). Завдяки йому не потрібно знову вводити URL публічних вузлів, лише ім’я хосту pc2.ygg та команду експорту налаштувань.

Аналогічним чином за допомогою yggvpn_export.conf з PC2 налаштовується і PC3. Експортований з нього конфіг далі буде використаний для налаштування віддалених робочих місць.

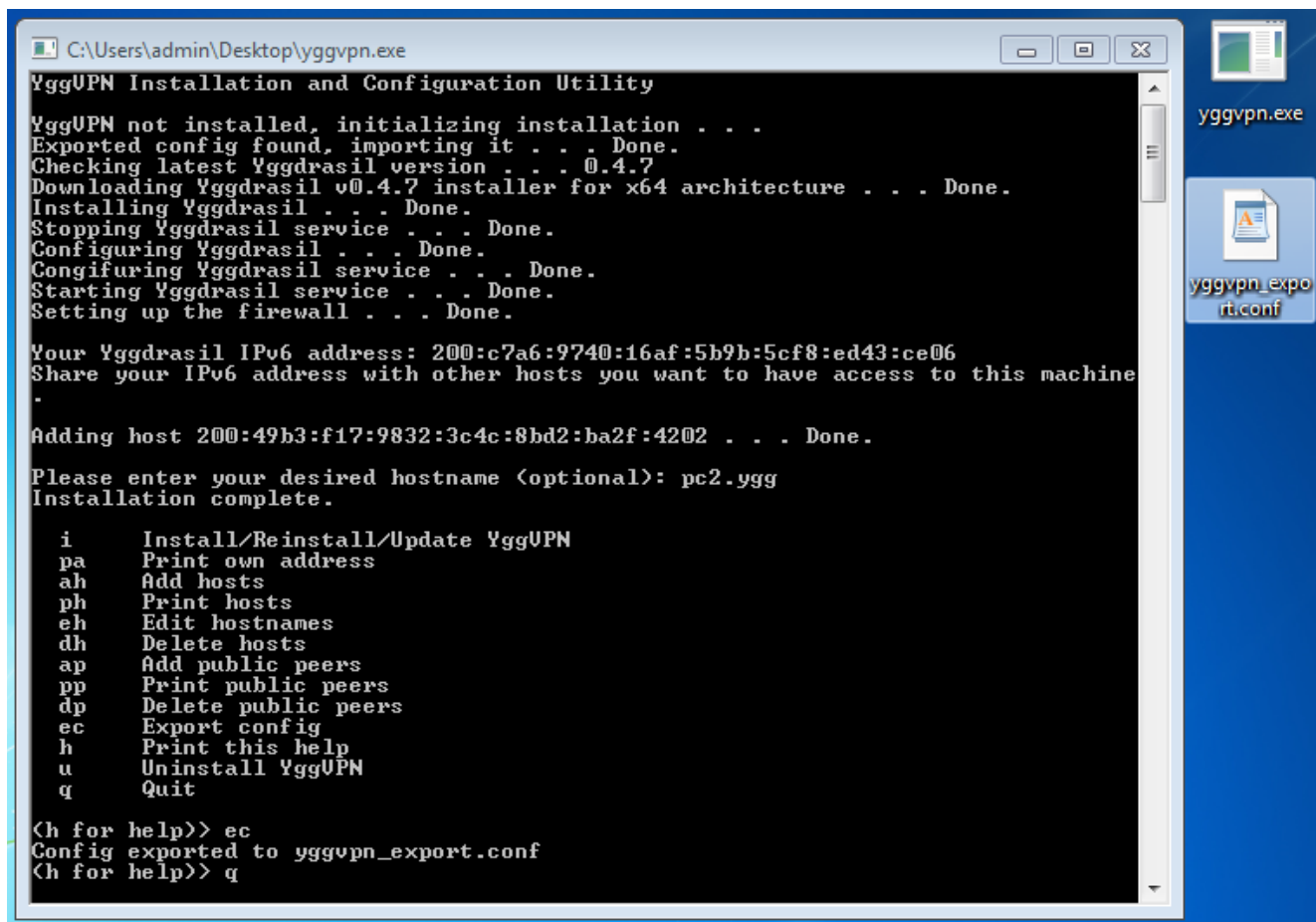


Рисунок 3.10 — Виконання YggVPN на робочому місці PC2

3.4.3 Перенаправлення портів

Для можливості з'єднання з комутатором KVM-over-IP через локальні робочі місця на них необхідно налаштувати перенаправлення вхідних IPv6-з'єднань з локального порту 5990 у порт 5900 (стандартний порт VNC) комутатора (192.168.1.100). Для цього застосовується системна утиліта `nesth` і команда, наведена у лістингу 3.2. Команду слід виконувати від імені адміністратора.

Лістинг 3.2 — Перенаправлення порту 5990 у 192.168.1.100:5900

```
netsh      interface      portproxy      add      v6tov4      listenport=5990
connectaddress=192.168.1.100 connectport=5900
```

3.4.4 Налаштування віддаленого керування телефонами

Згідно з рис. 3.3., у режимі модему телефон отримує адресу 192.168.42.129 і прописується як шлюз за замовчанням. Це значення постійне і однакове для усіх версій Android, що відкриває можливість звернення до телефону за цією адресою. Цю можливість можна використати для віддаленого керування телефоном у разі такої необхідності, для цього достатньо встановити на телефон додаток droidVNC-NG, який являє собою реалізацію VNC-сервера для Android версії 7 та новіше [16], а також виконати перенаправлення портів за допомогою утиліти netsh.

На рис. 3.11 наведено інтерфейс додатку droidVNC-NG.

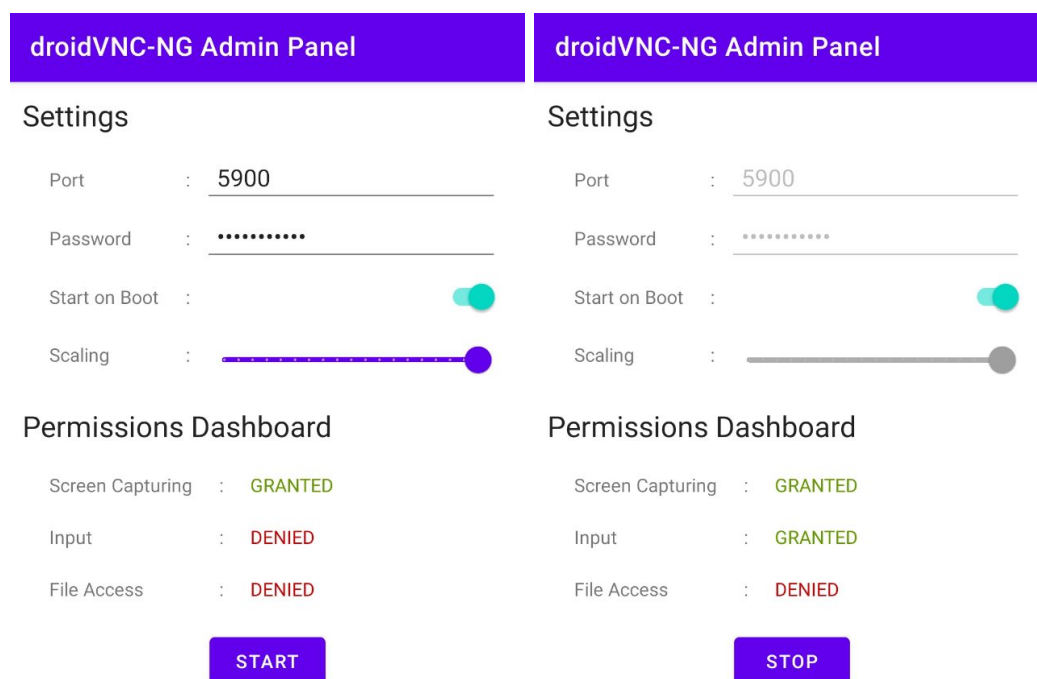


Рисунок 3.11 — Інтерфейс додатку droidVNC-NG

У додатку необхідно задати пароль для доступу, увімкнути запуск при завантаженні системи та запустити службу. При першому запуску необхідно також надати додатку дозволи додатків спеціальних можливостей (Accessibility), що продемонстровано на рис. 3.12.

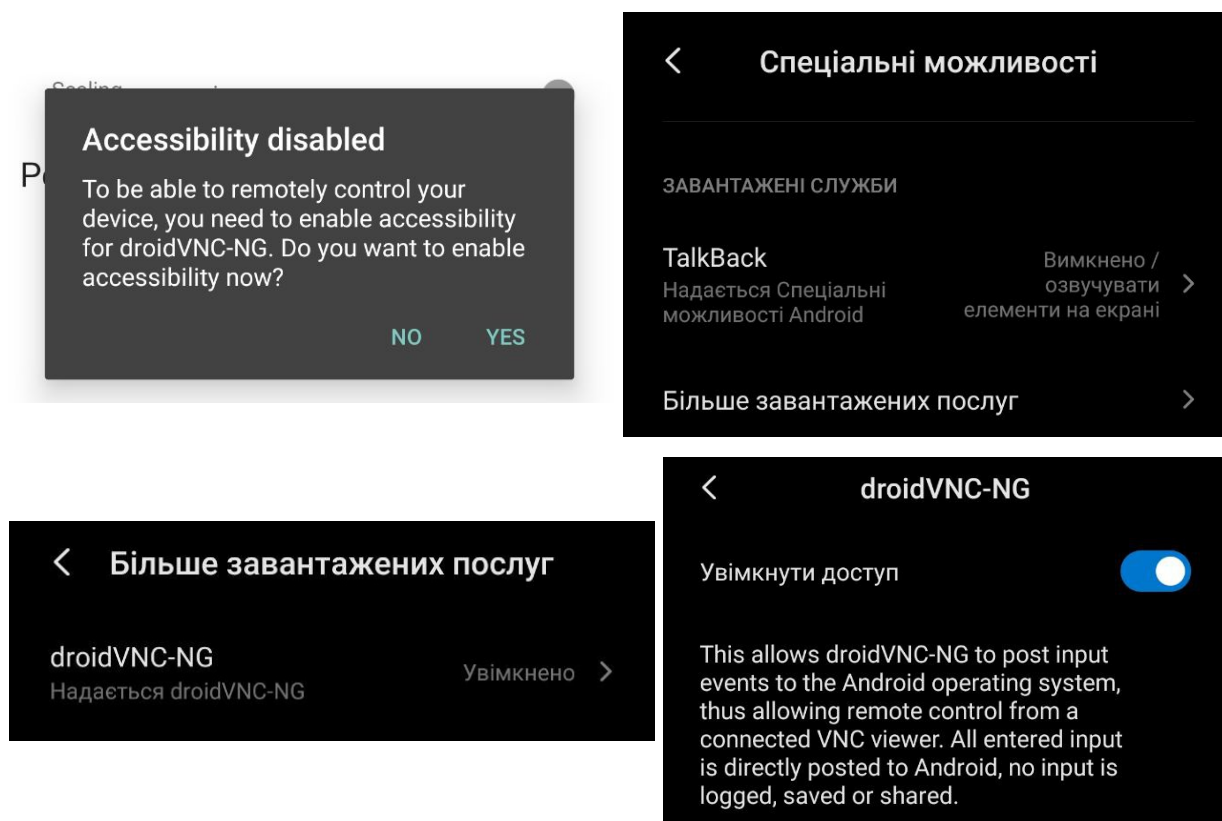


Рисунок 3.12 — Інтерфейс додатку droidVNC-NG

У лістингу 3.3 наведено команду для перенаправлення будь яких IPv6-з'єднань з портом 5909 комп'ютера на порт 5900 підключеного телефона (192.168.42.129).

Лістинг 3.3 — Перенаправлення порту 5909 у 192.168.1.42.129:5900

```
netsh interface portproxy add v6tov4 listenport=5909
connectaddress=192.168.42.129 connectport=5900
```

3.4.5 Додавання правил брандмауеру

За замовчанням майже усі порти у Windows закриті. Для можливості встановлення вхідних з'єднань з локальними робочими місцями необхідно додати відповідні правила брандмауеру. Для цього відкривається стандартна утиліта “Монитор брандмауєра Захитника Windows в режимі підвищеної безпеки” та створюється нове правило для вхідних з'єднань (рис. 3.13). У вікні майстра створення правила послідовно вводяться значення “Для порта”,

номер порта, “Разрешить подключение” та ім’я правила. Таким чином додаються три правила:

- 5900 — VNC;
- 5990 — VNC KVM;
- 5909 — VNC mobile.

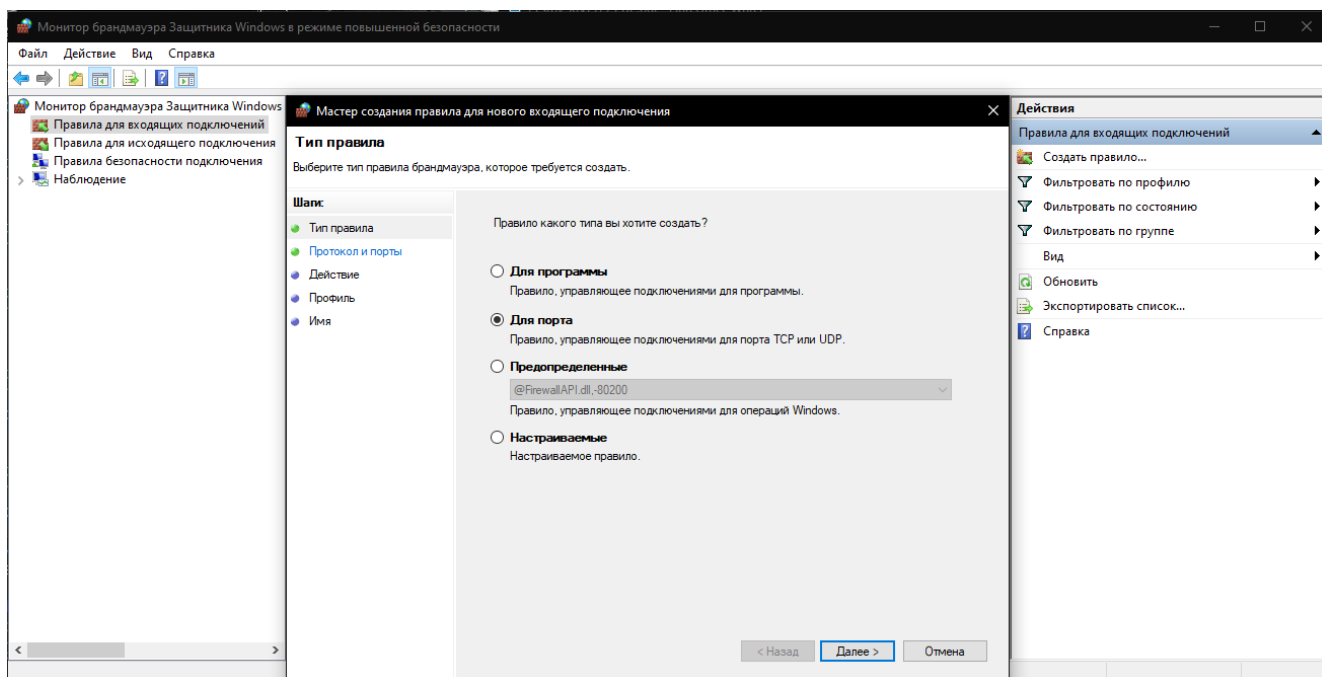


Рисунок 3.13 — Додавання правил брандмауеру

На деяких операційних системах локальні робочі місця за замовченням не можуть встановлювати пірингові з’єднання Yggdrasil між собою. Для виправлення цього додатково створюється правило “Yggdrasil”, що дозволяє вхідні з’єднання програмі C:\Program Files\Yggdrasil\yggdrasil.exe. Це робиться аналогічним чином за винятком обирання опції “Для программы” на першій сторінці майстра та введення шляху до виконуваного файлу.

3.5 Налаштування віддалених робочих місць

Віддалене керування об'єктом здійснюється з трьох віддалених робочих місць під управлінням Windows 7 або 10. Клієнт Yggdrasil на них встановлюється та налаштовуються аналогічним чином до пункту 3.4.2 з використанням останнього експортованого конфіг-файлу, про цьому вказувати ім'я хосту для віддалених робочих місць необов'язково. При додаванні робочого місця до системи його Yggdrasil IPv6-адресу необхідно додати до списку хостів на кожному з локальних робочих місць за допомогою утиліти YggVPN (команда "ah"). Після цього достатньо встановити клієнт TigerVNC та перевірити з'єднання з об'єктом за налаштованими іменами локальних робочих місць та номерами портів.

3.6 Структура готової системи та перевірка її роботи

Фізичну структуру налаштованої системи наведено на рис. 3.14, де:

- P1-3 — мобільні телефони;
- CBS — Cellular Base Station, базова станція сотового зв'язку.

Крапками позначено бездротове сотове з'єднання, тире — з'єднання будь-якого типу.

Для перевірки роботи системи виконуємо підключення до робочого столу PC1 з одного з віддалених робочих місць. Для цього у клієнті TigerVNC вводимо адресу "pc1.ygg:5900" та обираємо "Connect". Далі вводиться попередньо встановлений пароль. Процес підключення наведено на рис. 3.15.

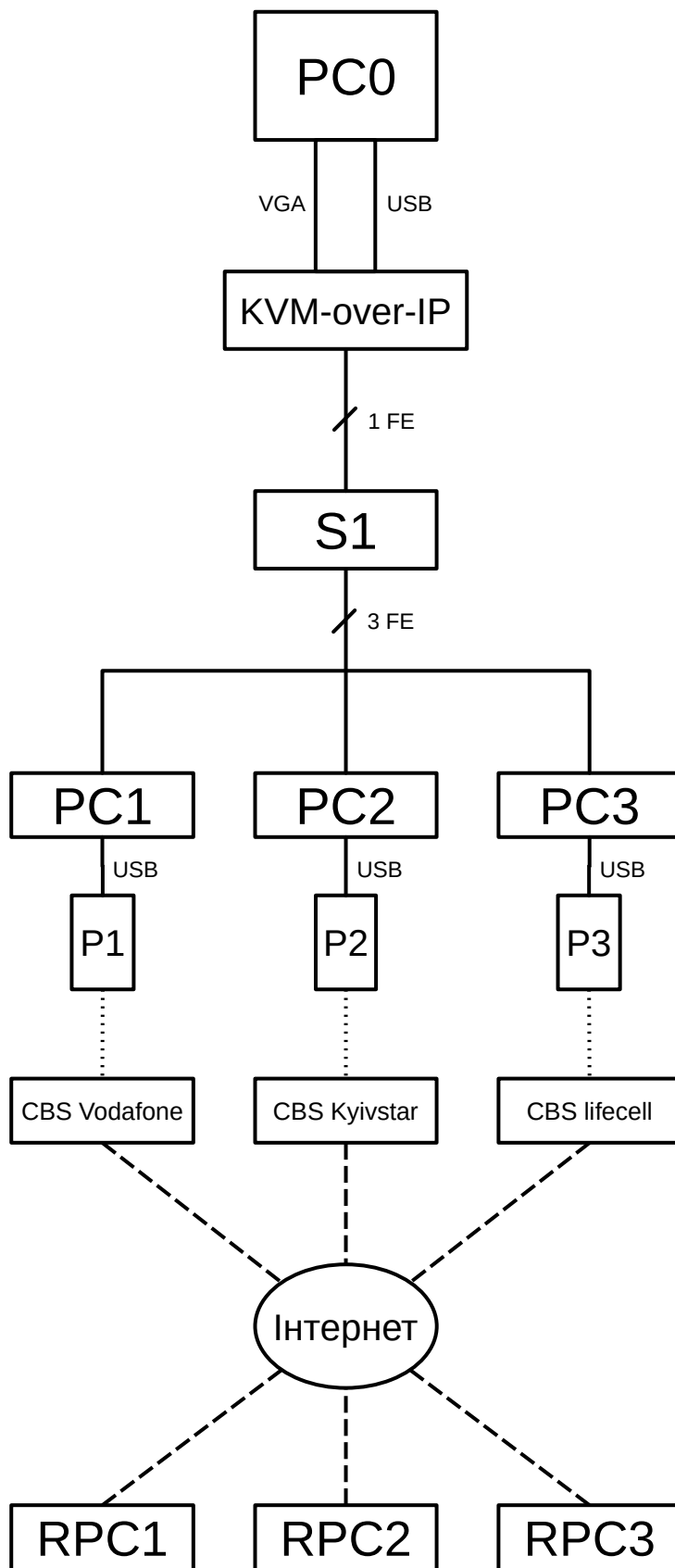


Рисунок 3.14 — Фізична структура налаштованої системи

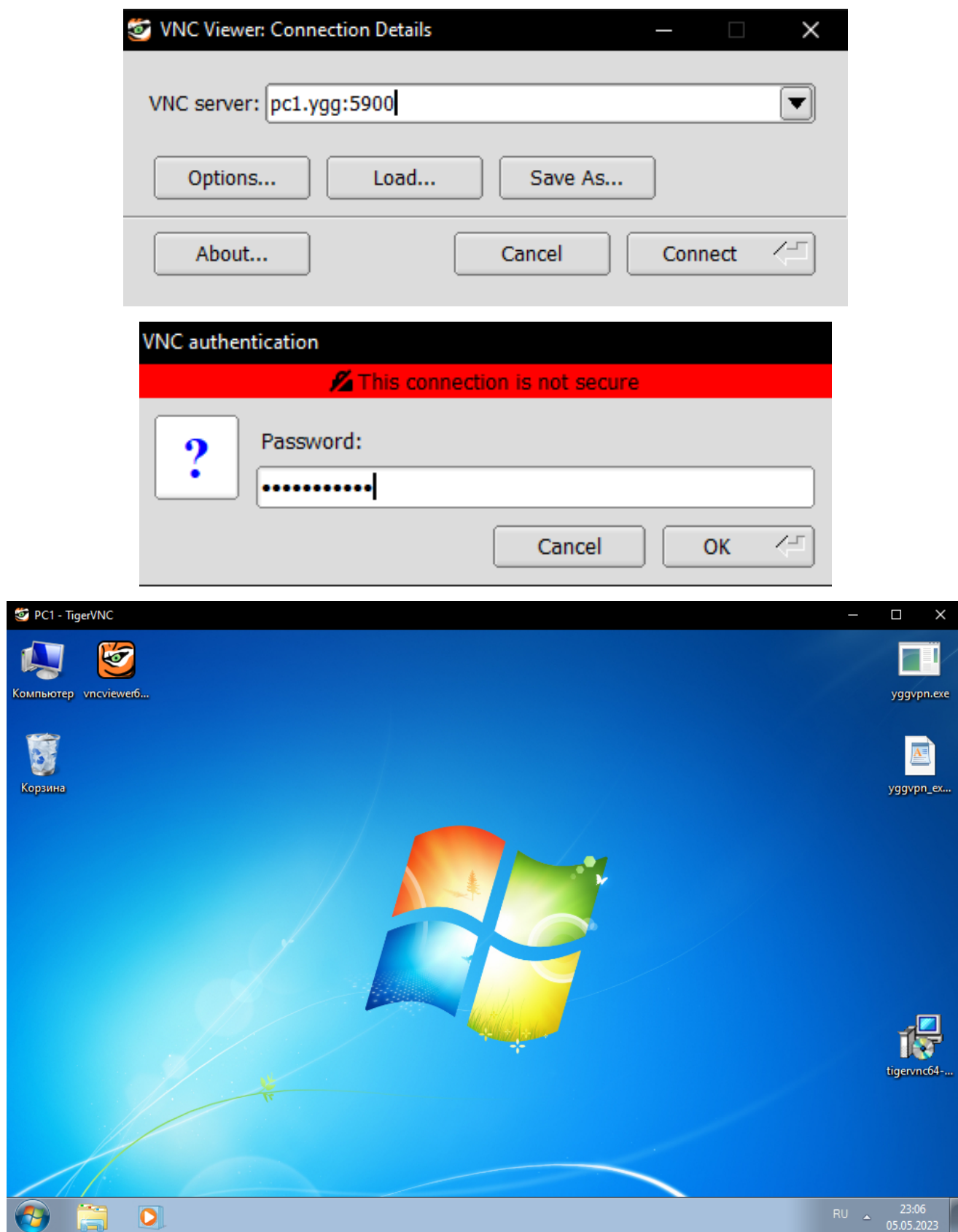


Рисунок 3.15 — Підключення до PC1

За портом 5990 аналогічно отримуємо доступ безпосередньо до керованого об'єкту (PC0 через KVM-over-IP) (рис. 3.16).

Примітка: наведений інтерфейс є лише прикладом інтерфейсу керованого об'єкту і не має відношення до реальної системи [17].

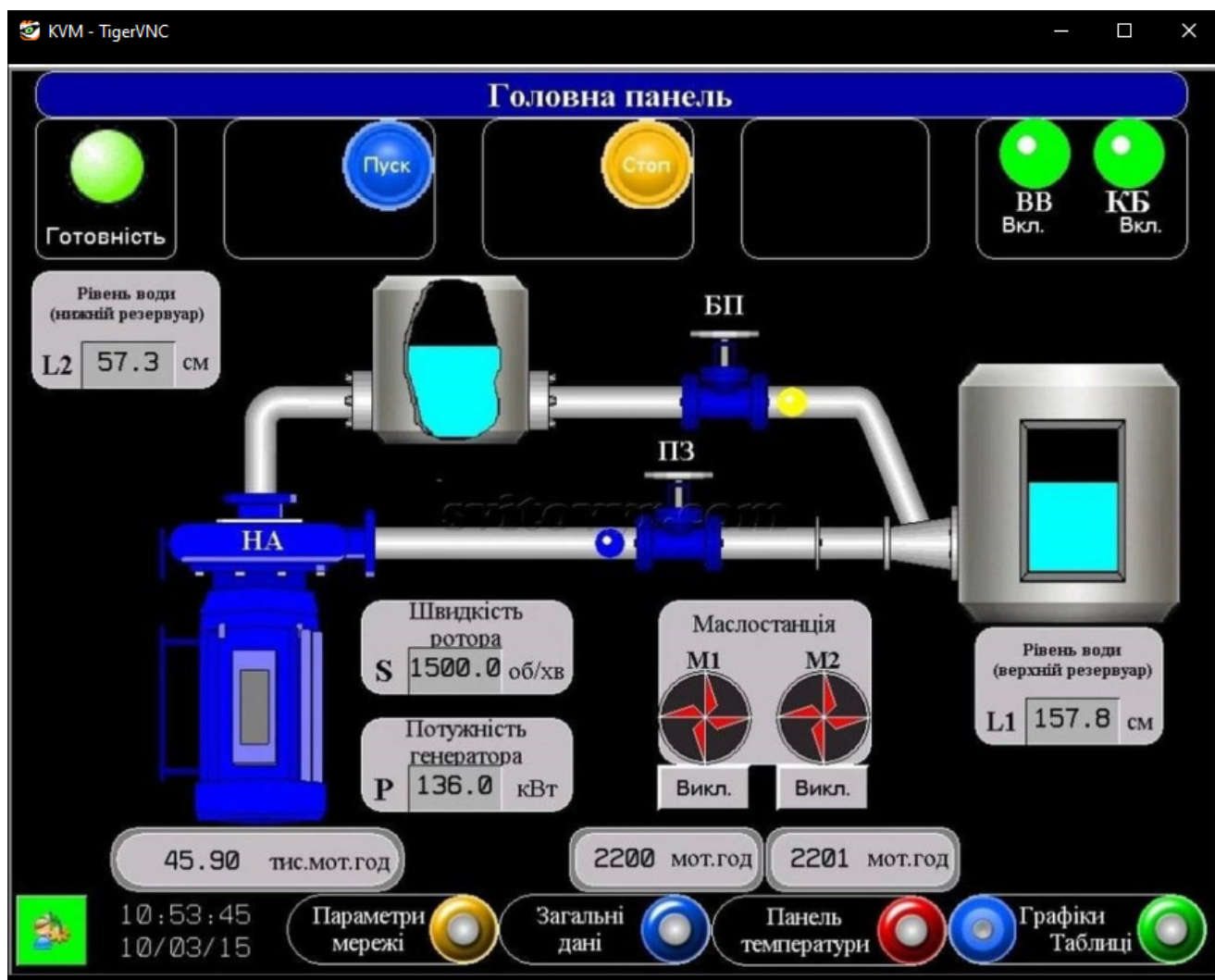


Рисунок 3.16 — Підключення до керованого об'єкту

За портом 5909 аналогічним чином можемо отримати доступ до телефону P1 (рис. 3.17).

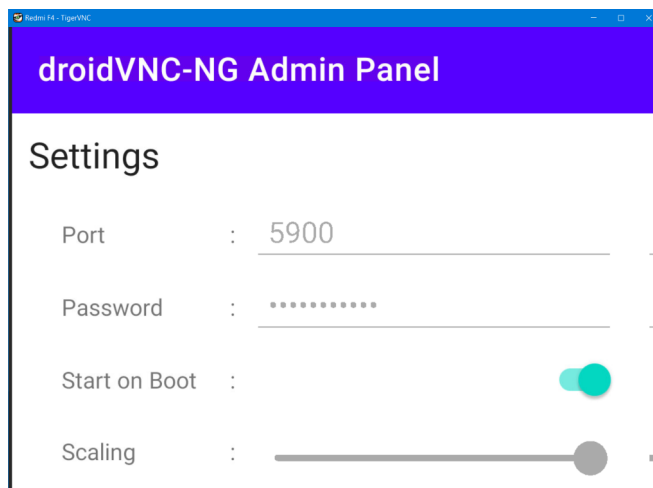


Рисунок 3.17 — Підключення до P1

У випадку повільного зв'язку клієнт TigerVNC дозволяє власноруч налаштувати рівень компресії зображення (рис. 3.18).

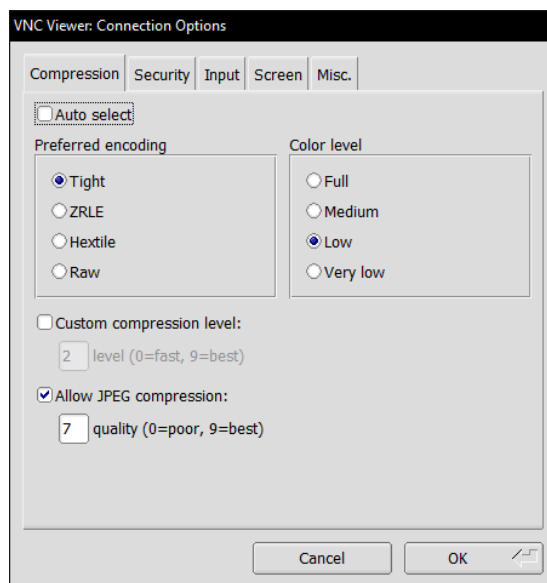


Рисунок 3.18 — Налаштування компресії зображення у TigerVNC

Для інших віддалених та локальних робочих місць результати перевірки аналогічні. При цьому, навіть при зникненні з'єднання з Інтернет у двох з трьох локальних робочих місць, вони все ще будуть доступні завдяки автоматичній побудові клієнтами Yggdrasil маршруту через третє робоче місце.

Система функціональна та готова до роботи.

ВИСНОВКИ

В процесі виконання роботи було проаналізовано наявні готові рішення віддаленого керування, виявлено їх недоліки при використанні для віддаленого керування об'єктом критичної інфраструктури. Було визначено вимоги до розроблюваної системи віддаленого керування об'єктом та запропоновано варіант її реалізації через оверлейну децентралізовану мережу Yggdrasil при використанні мобільних телефонів з підтримкою протоколу RNDIS та мереж мобільних операторів зв'язку, а також певного клієнтського програмного забезпечення.

Для полегшення процесу коректного встановлення та налаштування клієнту Yggdrasil на кінцевих пристроях під управлінням операційних систем сімейства Windows було розроблено конфігураційну утиліту YggVPN мовою C. Розроблена утиліта працює на 32/64-бітних операційних системах від Windows 7 та новіше, а також підходить для використання поза сценарієм віддаленого керування об'єктом.

Запропонований варіант системи було реалізовано на конкретному прикладі з використанням клієнтського та серверного програмного забезпечення TigerVNC для робочих місць та додатку droidVNC-NG для віддаленого керування телефонами при необхідності. Для зв'язку віддалених робочих місць з об'єктом обрано чотири публічні вузли мережі Yggdrasil. Надійність обрані вузлів була протестована шляхом постійної перевірки зв'язку з ними протягом двох тижнів.

Усі залучені програмні компоненти мають відкритий вихідний код. Розроблена система є децентралізованою — для віддаленого керування об'єктом достатньо по одному доступному віддаленому і локальному робочому місцю та публічному вузлу. Засобами мережі Yggdrasil та паролем захистом VNC-серверів забезпечується конфіденційність зв'язку та виключається несанкціонований доступ до об'єкту. Зв'язок з об'єктом здійснюється через Інтернет, тим самим забезпечуючи мобільність віддалених операторів та робочих

місць, при цьому система підтримує будь-які додаткові канали зв'язку об'єкту з глобальною мережею.

Після налаштування експлуатація системи віддаленими операторами зводиться до використання TigerVNC-клієнту та звернення до віддалених робочих місць за налаштованими іменами і номерами портів, а також редагування списків хостів і публічних вузлів робочих місць за допомогою розробленої утиліти. Таким чином, система відносно проста у використанні.

Розроблена система задовольняє усім попередньо висунутим вимогам.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Конституція України: Закон України “Про критичну інфраструктуру” від 05.12.2022 №1882-IX // База даних «Законодавство України»/Верховна Рада України. URL: <https://zakon.rada.gov.ua/laws/show/1882-20> (дата звернення: 11.05.2023).

2. Скільки об’єктів критичної інфраструктури постраждали через атаки РФ // Інформаційне агентство УНІАН. URL: <https://www.unian.ua/war/skilki-ob-yektiv-kritichnoji-infrastrukturi-postrazhdali-cherez-ataki-rf-12092421.html> (дата звернення: 11.05.2023).

3. Zero vs Thin vs Thick Clients: What's Right for Your Business? // BizTech. URL: <https://biztechmagazine.com/article/2018/10/thin-vs-thick-vs-zero-client-whats-right-fit-your-business-perfcon> (дата звернення: 11.05.2023).

4. GitHub - Ylianst/MeshCentral: A complete web-based remote monitoring and management web site. // GitHub. URL: <https://github.com/Ylianst/MeshCentral>

5. MeshCentral Security Features // MeshCentral. URL: <https://meshcentral.com/info/docs/securityfeatures.pdf> (дата звернення: 11.05.2023).

6. TigerVNC // TigerVNC. URL: <https://tigervnc.org/>

7. FAQ | Yggdrasil Network // Yggdrasil Network. URL: <https://yggdrasil-network.github.io/faq.html> (дата звернення: 11.05.2023).

8. GitHub - yggdrasil-network/yggdrasil-go: An experiment in scalable routing as an encrypted IPv6 overlay network // GitHub. URL: <https://github.com/yggdrasil-network/yggdrasil-go> (дата звернення: 11.05.2023).

9. Overview of Remote NDIS (RNDIS) // Microsoft Learn. URL: <https://learn.microsoft.com/en-us/windows-hardware/drivers/network/overview-of-remote-ndis--rndis-> (дата звернення: 11.05.2023).

10. Hjson, a user interface for JSON // Hjson. URL: <https://hjson.github.io/> (дата звернення: 11.05.2023).

11. Network shell (netsh) // Microsoft Learn. URL: <https://learn.microsoft.com/en-us/windows-server/networking/technologies/netsh/netsh> (дата звернення: 11.05.2023).
12. GitHub - benhoyt/inih: Simple .INI file parser in C, good for embedded systems // GitHub. URL: <https://github.com/benhoyt/inih> (дата звернення: 11.05.2023).
13. Application manifests // Microsoft Learn. URL: <https://learn.microsoft.com/en-us/windows/win32/sbscs/application-manifests> (дата звернення: 11.05.2023).
14. Cygwin // Cygwin URL: <https://cygwin.com/> (дата звернення: 11.05.2023).
15. Regupathy R. Unboxing Android USB: A hands on approach with real world examples. – Apress, 2014.
16. GitHub - bk138/droidVNC-NG: VNC server app for Android that does not require root privileges. // GitHub. URL: <https://github.com/bk138/droidVNC-NG> (дата звернення: 11.05.2023).
17. Гідроелектростанції. Реалізовані проекти // Svitovyr — URL: https://svitovyr.ua/HPP_projects/ (дата звернення: 11.05.2023).
18. Браян В. Керніган. Мова програмування С. / Браян В. Керніган, Деніс М. Річі. 2-е видання.: Пер. з англ. URL: <http://programming.in.ua/programming/c-language/227-book-programming-c-kernighan.html> (дата звернення: 11.05.2023).
19. Шпак З. Я. Програмування мовою С / З. Я. Шпак. – Львів : “Оріяна-Нова”, 2006. – 431 с.
20. Галісеєв Г. В. Системне програмування : навч. посіб. / Г. В. Галісеєв. – Київ : УН-Т «Україна», 2019. – 113 с.
21. Kurose J., Ross K. Computer Networking: A Top-Down Approach, Global Edition. – Pearson Education; 2018.
22. Tanenbaum, Andrew S., Wetherall, David J. Computer networks – 5th ed. – Pearson Education; 2021.

ДОДАТОК А

Лістинг А.1 — Фрагмент програмного коду, що відповідає за завантаження та встановлення клієнту Yggdrasil

```
//Check the latest Yggdrasil version
printf("Checking latest Yggdrasil version . . . ");
char version[10];
char* latest_json_path = malloc(64*sizeof(char));
//Expand %Temp%
ExpandEnvironmentStringsA("%Temp%\\latest.json", latest_json_path,
63);
HRESULT res = URLDownloadToFile(NULL, LATEST_RELEASE_URL,
latest_json_path, 0, NULL);
if(res != S_OK)
{
    printf("ERROR: ");
    if(res == E_OUTOFMEMORY)
        printf("The buffer length is invalid, or there is
insufficient memory to complete the operation. ");
    else if(res == INET_E_DOWNLOAD_FAILURE)
        printf("The specified resource or callback interface was
invalid. ");
    else
        printf("Unknown error. ");
    puts("Download failed.");
    system("pause");
    return -1;
}
root = json_object_from_file(latest_json_path);
if(root == NULL)
{
    puts("ERROR: Could not retrieve version information.");
    json_object_put(root);
    system("pause");
    return -1;
}
json_object_object_get_ex(root, "tag_name", &obj);
strcpy(version, json_object_get_string(obj) + 1); //Get rid of "v"
puts(version);
//Cleanup
DeleteFileA(latest_json_path);
```

Продовження лістингу A.1

```

free(latest_json_path);
json_object_put(obj);
json_object_put(root);
//Check architecture
char arch[4] = "x86";
SYSTEM_INFO sysinfo;
GetNativeSystemInfo(&sysinfo);
if(sysinfo.wProcessorArchitecture == PROCESSOR_ARCHITECTURE_AMD64)
    strcpy(arch, "x64");
//Build download URL
char* URL = malloc(100*sizeof(char));
strcpy(URL,
"https://github.com/yggdrasil-network/yggdrasil-go/releases/download
/v");
strcat(URL, version);
strcat(URL, "/yggdrasil-");
strcat(URL, version);
strcat(URL, "-");
strcat(URL, arch);
strcat(URL, ".msi");
//Download Yggdrasil installer
printf("Downloading Yggdrasil v%s installer for %s
architecture . . . ", version, arch);
res = URLDownloadToFile(NULL, URL, installer_path, 0, NULL);
free(URL);
if(res != S_OK)
{
    printf("ERROR: ");
    if(res == E_OUTOFMEMORY)
        printf("The buffer length is invalid, or there is
insufficient memory to complete the operation. ");
    else if(res == INET_E_DOWNLOAD_FAILURE)
        printf("The specified resource or callback interface was
invalid. ");
    else
        printf("Unknown error. ");
    puts("Download failed.");
    system("pause");
    return -1;
}
puts("Done.");
}
else

```

Продовження лістингу A.1

```

printf("Installer already downloaded. ");
//Launch the installer
printf("Installing Yggdrasil . . . ");
STARTUPINFO si;
PROCESS_INFORMATION pi;
ZeroMemory(&si, sizeof(si));
si.cb = sizeof(si);
ZeroMemory(&pi, sizeof(pi));
// Construct command line: msixec.exe /i %Temp%\yggdrasil.msi /qn
char* cmdline = malloc((strlen(installer_path) + 20)*sizeof(char));
strcpy(cmdline, "msiexec.exe /i ");
strcat(cmdline, installer_path);
strcat(cmdline, " /qn");
if(!CreateProcessA(NULL, cmdline, NULL, NULL, FALSE, 0, NULL, NULL,
&si, &pi))
{
printf("ERROR: Installation failed (%d).\n", GetLastError());
DeleteFileA(installer_path);
system("pause");
return -1;
}
free(cmdline);
WaitForSingleObject(pi.hProcess, INFINITE);
CloseHandle(pi.hProcess);
CloseHandle(pi.hThread);
//Copy the MSI file to YGGVPN_CONFIG_DIR for uninstall()
if(!CopyFile(installer_path, YGGVPN_CONFIG_DIR"\yggdrasil.msi",
FALSE))
{
printf("ERROR: Failed to copy the installer to \"YGGVPN_CONFIG_DIR\"
(%d).\n", GetLastError());
DeleteFileA(installer_path);
system("pause");
return -1;
}
DeleteFileA(installer_path);
free(installer_path);
puts("Done.");
...

```

Лістинг А.2 — Програмний код функцій керування службою Yggdrasil
(файл yggdrasil_service.c)

```
#include <windows.h>

static SC_HANDLE SCM = NULL;
static SC_HANDLE YGGDRASIL_SERVICE = NULL;

BOOL SCM_init()
{
    SC_HANDLE SCM = OpenSCManagerA(NULL, NULL, GENERIC_EXECUTE);
    if(SCM == NULL)
        return FALSE;
    YGGDRASIL_SERVICE = OpenServiceA(SCM, "Yggdrasil", GENERIC_READ
| GENERIC_WRITE | GENERIC_EXECUTE);
    if(YGGDRASIL_SERVICE == NULL)
        return FALSE;
    return TRUE;
}

BOOL SCM_cleanup()
{
    CloseHandle(YGGDRASIL_SERVICE);
    CloseHandle(SCM);
}

BOOL wait_for_yggdrasil_service()
{
    SERVICE_STATUS_PROCESS ssp;
    DWORD dwBytesNeeded;
    DWORD dwTimeout = 10000; // 10-second time-out
    DWORD dwWaitTime;
    DWORD dwCurrTime;
    if(ssp.dwCurrentState != SERVICE_STOPPED &&
ssp.dwCurrentState != SERVICE_STOP_PENDING)
        return TRUE;
    dwCurrTime = GetTickCount();
    while(ssp.dwCurrentState != SERVICE_RUNNING)
    {
        dwWaitTime = ssp.dwWaitHint/10;
        if(dwWaitTime < 1000)
            dwWaitTime = 1000;
        else if(dwWaitTime > 10000)
            dwWaitTime = 10000;
    }
}
```

Продовження лістингу A.2

```

        Sleep(dwWaitTime);
        if(!QueryServiceStatusEx(YGGDRASIL_SERVICE,
SC_STATUS_PROCESS_INFO, (LPBYTE)&ssp,
sizeof(SERVICE_STATUS_PROCESS), &dwBytesNeeded))
            return FALSE;
        else if(GetTickCount() - dwCurrTime > dwTimeout)
            return FALSE;
    }
    return TRUE;
}

BOOL start_yggdrasil_service()
{
    SERVICE_STATUS_PROCESS ssp;
    DWORD dwStartTime;
    DWORD dwBytesNeeded;
    DWORD dwTimeout = 30000; // 30-second time-out
    DWORD dwWaitTime;
    if(!QueryServiceStatusEx(YGGDRASIL_SERVICE,
SC_STATUS_PROCESS_INFO, (LPBYTE)&ssp,
sizeof(SERVICE_STATUS_PROCESS), &dwBytesNeeded))
        return FALSE;
    if(ssp.dwCurrentState != SERVICE_STOPPED &&
ssp.dwCurrentState != SERVICE_STOP_PENDING)
        return TRUE;
    dwStartTime = GetTickCount();
    // Wait for the service to stop before attempting to start it
    while(ssp.dwCurrentState == SERVICE_STOP_PENDING)
    {
        dwWaitTime = ssp.dwWaitHint/10;
        if(dwWaitTime < 1000)
            dwWaitTime = 1000;
        else if(dwWaitTime > 10000)
            dwWaitTime = 10000;
        Sleep(dwWaitTime);
        if(!QueryServiceStatusEx(YGGDRASIL_SERVICE,
SC_STATUS_PROCESS_INFO, (LPBYTE)&ssp,
sizeof(SERVICE_STATUS_PROCESS), &dwBytesNeeded))
            return FALSE;
        if(GetTickCount() - dwStartTime > dwTimeout)
            return FALSE;
    }
    if(!StartServiceA(YGGDRASIL_SERVICE, 0, NULL))

```

Продовження лістингу A.2

```

        return FALSE;
    if(!QueryServiceStatusEx(YGGDRASIL_SERVICE,
SC_STATUS_PROCESS_INFO, (LPBYTE)&ssp,
sizeof(SERVICE_STATUS_PROCESS), &dwBytesNeeded))
        return FALSE;
    while(ssp.dwCurrentState == SERVICE_START_PENDING)
    {
        dwWaitTime = ssp.dwWaitHint/10;
        if(dwWaitTime < 1000)
            dwWaitTime = 1000;
        else if(dwWaitTime > 10000)
            dwWaitTime = 10000;
        Sleep(dwWaitTime);
        if(!QueryServiceStatusEx(YGGDRASIL_SERVICE,
SC_STATUS_PROCESS_INFO, (LPBYTE)&ssp,
sizeof(SERVICE_STATUS_PROCESS), &dwBytesNeeded))
            return FALSE;
        else if(GetTickCount() - dwStartTime > dwTimeout)
            return FALSE;
    }
    return TRUE;
}

BOOL stop_yggdrasil_service()
{
    SERVICE_STATUS_PROCESS ssp;
    DWORD dwStartTime = GetTickCount();
    DWORD dwBytesNeeded;
    DWORD dwTimeout = 30000; // 30-second time-out
    DWORD dwWaitTime;
    if(!QueryServiceStatusEx(YGGDRASIL_SERVICE,
SC_STATUS_PROCESS_INFO, (LPBYTE)&ssp,
sizeof(SERVICE_STATUS_PROCESS), &dwBytesNeeded))
        return FALSE;
    if(ssp.dwCurrentState == SERVICE_STOPPED)
        return TRUE;
    // If a stop is pending, wait for it
    while(ssp.dwCurrentState == SERVICE_STOP_PENDING)
    {
        dwWaitTime = ssp.dwWaitHint/10;
        if(dwWaitTime < 1000)
            dwWaitTime = 1000;
        else if(dwWaitTime > 10000)

```

Продовження лістингу A.2

```

        dwWaitTime = 10000;
        Sleep(dwWaitTime);
        if(!QueryServiceStatusEx(YGGDRASIL_SERVICE,
SC_STATUS_PROCESS_INFO, (LPBYTE)&ssp,
sizeof(SERVICE_STATUS_PROCESS), &dwBytesNeeded))
            return FALSE;
        if(ssp.dwCurrentState == SERVICE_STOPPED)
            return TRUE;
        if(GetTickCount() - dwStartTime > dwTimeout)
            return FALSE;
    }
    if(!ControlService(YGGDRASIL_SERVICE, SERVICE_CONTROL_STOP,
(LPSERVICE_STATUS)&ssp))
        return FALSE;
    // Wait for the service to stop
    while(ssp.dwCurrentState != SERVICE_STOPPED)
    {
        Sleep(ssp.dwWaitHint);
        if(!QueryServiceStatusEx(YGGDRASIL_SERVICE,
SC_STATUS_PROCESS_INFO, (LPBYTE)&ssp,
sizeof(SERVICE_STATUS_PROCESS), &dwBytesNeeded))
            return FALSE;
        if(GetTickCount() - dwStartTime > dwTimeout)
            return FALSE;
    }
    return TRUE;
}

BOOL set_yggdrasil_service_params()
{
    SERVICE_FAILURE_ACTIONSA sfa;
    sfa.dwResetPeriod = 3;
    sfa.lpRebootMsg = NULL;
    sfa.lpCommand = NULL;
    sfa.cActions = 3;
    SC_ACTION sa[3];
    sa[0].Type = SC_ACTION_RESTART;
    sa[0].Delay = 1000;
    sa[1].Type = SC_ACTION_RESTART;
    sa[1].Delay = 5000;
    sa[2].Type = SC_ACTION_RESTART;
    sa[2].Delay = 10000;
    sfa.lpsaActions = (LPSC_ACTION)&sa;

```

Продовження лістингу A.2

```

    if(!ChangeServiceConfig2A(YGGDRASIL_SERVICE,
SERVICE_CONFIG_FAILURE_ACTIONS, &sfa))
        return FALSE;
    SERVICE_FAILURE_ACTIONS_FLAG sfaf;
    sfaf.fFailureActionsOnNonCrashFailures = TRUE;
    if(!ChangeServiceConfig2A(YGGDRASIL_SERVICE,
SERVICE_CONFIG_FAILURE_ACTIONS_FLAG, &sfaf));
}

```

Лістинг A.3 — Програмний код функцій налаштування фаєрволу (файл firewall.c)

```

#include <windows.h>
#include <string.h>
#include <stdio.h>

BOOL firewall_remove()
{
    system("netsh ipsec static delete rule name=Yggdrasil_Deny
policy=Yggdrasil_Firewall > nul");
    system("netsh ipsec static delete filterlist
name=Yggdrasil_Deny > nul");
    system("netsh ipsec static delete rule name=Yggdrasil_Whitelist
policy=Yggdrasil_Firewall > nul");
    system("netsh ipsec static delete filterlist
name=Yggdrasil_Whitelist > nul");
    system("netsh ipsec static delete policy
name=Yggdrasil_Firewall > nul");
    system("netsh ipsec static delete filteraction name=Deny >
nul");
    system("netsh ipsec static delete filteraction name=Permit >
nul");
}

BOOL firewall_add_host(char* host)
{
    char* buf = malloc((93 + strlen(host) + 1)*sizeof(char));
    strcpy(buf, "netsh ipsec static add filter
filterlist=Yggdrasil_Whitelist mirrored=yes dstaddr=me srcaddr=");
    strcat(buf, host);
    FILE* ppipe = popen(buf, "r");
    fgets(buf, 93 + strlen(host) + 1, ppipe);
}

```

Продовження лістингу A.3

```

    if(strstr(buf, "IPsec") != NULL) //Check for error
        return FALSE;
    free(buf);
    return TRUE;
}

BOOL firewall_delete_host(char** hosts, int host_count, int index)
{
    system("netsh ipsec static delete rule name=Yggdrasil_Whitelist
policy=Yggdrasil_Firewall > nul");
    system("netsh ipsec static delete filterlist
name=Yggdrasil_Whitelist > nul");
    system("netsh ipsec static add filterlist
name=Yggdrasil_Whitelist > nul");
    int i;
    for(i = 0; i < index; i++)
        firewall_add_host(hosts[i]);
    //Skip the deleted host
    for(i++; i < host_count; i++)
        firewall_add_host(hosts[i]);
    system("netsh ipsec static add rule name=Yggdrasil_Whitelist
policy=Yggdrasil_Firewall filterlist=Yggdrasil_Whitelist
filteraction=Permit > nul");
    return TRUE;
}

BOOL firewall_init(char* own_address)
{
    int own_address_size = (strlen(own_address) + 1)*sizeof(char);
    system("netsh ipsec static add filteraction name=Deny
action=block > nul");
    system("netsh ipsec static add filteraction name=Permit
action=permit > nul");

    //In case our Yggdrasil address changed
    system("netsh ipsec static delete rule name=Yggdrasil_Deny
policy=Yggdrasil_Firewall > nul");
    system("netsh ipsec static delete filterlist
name=Yggdrasil_Deny > nul");
    system("netsh ipsec static delete rule name=Yggdrasil_Whitelist
policy=Yggdrasil_Firewall > nul");
    system("netsh ipsec static delete filterlist
name=Yggdrasil_Whitelist > nul");

```

Продовження лістингу А.3

```

system("netsh ipsec static add filterlist name=Yggdrasil_Deny >
nul");
char* buf;
buf = malloc(95*sizeof(char) + own_address_size);
strcpy(buf, "netsh ipsec static add filter
filterlist=Yggdrasil_Deny mirrored=yes srcaddr=any dstaddr=");
strcat(buf, own_address);
strcat(buf, " > nul");
system(buf);
free(buf);
system("netsh ipsec static add filterlist
name=Yggdrasil_Whitelist > nul");
firewall_add_host(own_address);
system("netsh ipsec static add policy name=Yggdrasil_Firewall
assign=no > nul");
system("netsh ipsec static add rule name=Yggdrasil_Whitelist
policy=Yggdrasil_Firewall filterlist=Yggdrasil_Whitelist
filteraction=Permit > nul");
system("netsh ipsec static add rule name=Yggdrasil_Deny
policy=Yggdrasil_Firewall filterlist=Yggdrasil_Deny
filteraction=Deny > nul");
return TRUE;
}

BOOL firewall_enable()
{
    system("netsh ipsec static set policy name=Yggdrasil_Firewall
assign=yes > nul");
    return TRUE;
}

BOOL firewall_disable()
{
    system("netsh ipsec static set policy name=Yggdrasil_Firewall
assign=no > nul");
    return TRUE;
}

```

Лістинг А.4 — Фрагмент програмного коду, який відповідає за додавання хостів та задавання власного імені хосту

```

if(EXPORTED_CONF)
{
    for(int i = 0; i < PCONF->host_count; i++)
    {
        if(!strcmp(PCONF->hosts[i], own_address))
            continue;
        printf("Adding host %s . . . ", PCONF->hosts[i]);
        if(firewall_add_host(PCONF->hosts[i]))
            puts("Done.");
        else
            puts("Invalid host.");
    }
}
else
{
    printf("Add hosts? (y/N): ");
    char input = 'n';
    input = getchar();
    if(input == 'y')
        add_hosts();
    if(input != '\n')
        clear();
}
int index = find(PCONF->hosts, PCONF->host_count, own_address);
if(index < 0)
{
    append(&PCONF->hosts, PCONF->host_count, strdup(own_address));
    printf("\nPlease enter your desired hostname (optional): ");
}
else
{
    if(!strcmp(PCONF->hostnames[index], " "))
        printf("\nPlease enter your desired hostname (optional): ");
    else
        printf("\nYour hostname is set as %s. Enter a new one to change it or nothing to leave as is: ", PCONF->hostnames[index]);
}
//Set own hostname
char input = getchar();
char* hostname;

```

Продовження лістингу A.4

```

if(input != '\n')
{
    ungetc(input, stdin);
    char* buf = malloc(64*sizeof(char));
    scanf("%63s", buf);
    hostname = strdup(buf);
    free(buf);
    clear();
}
else
    hostname = strdup(" ");
if(index < 0)
{
    append(&PCONF->hostnames, PCONF->host_count, hostname);
    PCONF->host_count++;
}
else if(input != '\n')
{
    free(PCONF->hostnames[index]);
    PCONF->hostnames[index] = hostname;
}

```

Лістинг A.5 — Програмний код допоміжних функцій (файл util.c)

```

#include <string.h>
#include <windows.h>
/**
Clears stdin.
*/
void clear()
{
    int c;
    while ((c = getchar()) != '\n' && c != EOF);
}

/**
Return the index of val in arr or -1 if it's not present
*/
int find(char** arr, int size, char* val)
{
    int ind = -1;
    for(int i = 0; i < size; i++)
        if(!strcmp(arr[i], val))

```

Продовження лістингу A.5

```

        {
            ind = i;
            break;
        }
    return ind;
}

/**
Extends arr1 with arr2, relocating it in memory.
*/
void extend(void** arr1, const int size1, const void* arr2, const
int size2)
{
    void* buf = malloc(size1 + size2);
    memcpy(buf, *arr1, size1);
    memcpy(buf + size1, arr2, size2);
    free(*arr1);
    *arr1 = buf;
}

/**
Appends arr with str, relocating it in memory.
*/
void append(char*** arr, const int sizea, char* str)
{
    char** buf = malloc((sizea + 1)*sizeof(char*));
    memcpy(buf, *arr, sizea*sizeof(char*));
    buf[sizea] = str;
    free(*arr);
    *arr = buf;
}

/**
Removes indexed string from arr, relocating it in memory.
*/
void pop(char*** arr, const int size, const int index)
{
    char** buf = malloc((size - 1)*sizeof(char*));
    int i;
    for(i = 0; i < index; i++)
        buf[i] = (*arr)[i];
    free((*arr)[i]);
    for(i++; i < size; i++)

```

Продовження лістингу A.5

```

        buf[i - 1] = (*arr)[i];
    free(*arr);
    *arr = buf;
}

/**
Returns the number of substrings in 'strings' that are divided with
(the first character of) 'delimiter'.
*/
int strcnt(const char* strings, char* delimiter)
{
    int string_count = 1;
    char d = delimiter[0];
    char* c = (char*)strings;
    while(*c != '\0')
    {
        if(*c == d)
            string_count++;
        c++;
    }
    return string_count;
}

```

Лістинг A.6 — Програмний код функції handler()

```

static int handler(void* conf, const char* section, const char*
parameter, const char* value)
{
    config* pconfig = (config*)conf;
    #define MATCH(s, n) strcmp(section, s) == 0 &&
strcmp(parameter, n) == 0
    if(MATCH("peers", "peers"))
    {
        pconfig->peer_count = strcnt(value,
CONFIG_STRING_DELIMITER);
        pconfig->peers = malloc(pconfig-
>peer_count*sizeof(char*));
        char* peers = strtok(strdup(value),
CONFIG_STRING_DELIMITER);
        pconfig->peers[0] = strdup(peers);
        for(int i = 1; i < pconfig->peer_count; i++)
            pconfig->peers[i] = strdup(strtok(NULL,
CONFIG_STRING_DELIMITER));
    }
}

```

Продовження лістингу А.6

```

        free(peers);
    }
    else if(MATCH("hosts", "hosts"))
    {
        pconfig->host_count = strcnt(value,
CONFIG_STRING_DELIMITER);
        pconfig->hosts = malloc(pconfig-
>host_count*sizeof(char*));
        char* hosts = strtok(strdup(value),
CONFIG_STRING_DELIMITER);
        pconfig->hosts[0] = strdup(hosts);
        for(int i = 1; i < pconfig->host_count; i++)
            pconfig->hosts[i] = strdup(strtok(NULL,
CONFIG_STRING_DELIMITER));
        free(hosts);
    }
    else if(MATCH("hosts", "hostnames"))
    {
        pconfig->host_count = strcnt(value,
CONFIG_STRING_DELIMITER);
        pconfig->hostnames = malloc(pconfig-
>host_count*sizeof(char*));
        char* hostnames = strdup(value);
        //First value empty handling
        int i = 0;
        if(hostnames[0] == CONFIG_STRING_DELIMITER_CHAR)
        {
            pconfig->hostnames[0] = strdup(" ");
            i++;
        }
        hostnames = strtok(hostnames, CONFIG_STRING_DELIMITER);
        pconfig->hostnames[i] = strdup(hostnames);
        i++;
        char* tmp;
        for(; i < pconfig->host_count; i++)
        {
            tmp = strtok(NULL, CONFIG_STRING_DELIMITER);
            //Last value empty handling
            if(tmp != NULL)
                pconfig->hostnames[i] = strdup(tmp);
            else
                pconfig->hostnames[i] = strdup(" ");
        }
    }

```

Продовження лістингу A.6

```

        free(hostnames);
    }
    else
    {
        printf("WARNING: Invalid config parameters (section %s
parameter %s)\n", section, parameter);
        return 0; // unknown section/parameter, error
    }
    return 1;
}

```

Лістинг A.7 — Програмний код функції menu()

```

int menu()
{
    SCM_init();
    if(!start_yggdrasil_service())
    {
        printf("ERROR: Could not start Yggdrasil service (%d).\n",
GetLastError());
        SCM_cleanup();
        system("pause");
        return -1;
    }
    SCM_cleanup();
    print_help();
    char input;
    printf("(h for help)> ");
    for(;;)
    {
        input = getchar();
        switch(input)
        {
            case 'i':
                clear(); //clear stdin
                config_reset(PCONF);
                EXPORTED_CONF = FALSE;
                if(PathFileExistsA(".\\yggvpn_export.conf"))
                {
                    printf("Exported config found, do you wish to use it? (y/N): ");
                    input = 'n';
                    input = getchar();
                    if(input == 'y')

```

Продовження лістингу A.7

```

        {
            if(ini_parse(".\\yggvpn_export.conf", handler, PCONF) < 0)
            {
                free(PCONF);
                puts("ERROR: Invalid config.");
                system("pause");
                return -1;
            }
            EXPORTED_CONF = TRUE;
        }
        if(input != '\n')
            clear();
    }
    install();
    break;
case 'p':
    input = getchar();
    switch(input)
    {
        case 'a':
            clear();
            printf("Your Yggdrasil IPv6 address:
%s\n", get_own_yggdrasil_address());
            break;
        case 'p':
            clear();
            print_peers();
            break;
        case 'h':
            clear();
            print_hosts();
            break;
        default:
            print_help();
    }
    break;
case 'a':
    input = getchar();
    switch(input)
    {
        case 'h':
            clear();
            add_hosts();

```

Продовження лістингу A.7

```

        update_hosts_file();
        update_yggvpn_config();
        clear();
        break;
    case 'p':
        clear();
        add_peers();
        printf("Restarting Yggdrasil service . . . ");
        SCM_init();
        if(!stop_yggdrasil_service())
        {
            printf("ERROR: Could not stop
Yggdrasil service (%d).\n", GetLastError());
            SCM_cleanup();
            system("pause");
            return -1;
        }
        update_yggdrasil_config();
        if(!start_yggdrasil_service())
        {
            printf("ERROR: Could not start
Yggdrasil service (%d).\n", GetLastError());
            SCM_cleanup();
            system("pause");
            return -1;
        }
        SCM_cleanup();
        update_yggvpn_config();
        puts("Done.");
        break;
    default:
        print_help();
    }
    break;
case 'd':
    input = getchar();
    switch(input)
    {
        case 'p':
            clear();
            delete_peers();
            printf("Restarting Yggdrasil service . . . ");
            SCM_init();

```

Продовження лістингу A.7

```

        if(!stop_yggdrasil_service())
        {
            printf("ERROR: Could not stop
Yggdrasil service (%d).\n", GetLastError());
            SCM_cleanup();
            system("pause");
            return -1;
        }
        update_yggdrasil_config();
        if(!start_yggdrasil_service())
        {
            printf("ERROR: Could not start
Yggdrasil service (%d).\n", GetLastError());
            SCM_cleanup();
            system("pause");
            return -1;
        }
        SCM_cleanup();
        puts("Done.");
        update_yggvpn_config();
        break;
    case 'h':
        clear();
        delete_hosts();
        update_hosts_file();
        update_yggvpn_config();
        break;
    default:
        print_help();
    }
    break;
case 'e':
    input = getchar();
    switch(input)
    {
        case 'h':
            clear();
            edit_hostnames();
            update_hosts_file();
            update_yggvpn_config();
            break;
        case 'c':
            clear();

```

Продовження лістингу A.7

```

                                export_config();
                                break;
                        default:
                                print_help();
                }
                break;
        case 'u':
                clear();
                uninstall();
                break;
        case 'q':
                break;
        case '\n':
                break;
        default:
                clear();
                print_help();
    }
    if(input == 'q')
        break;
    printf("(h for help)> ");
}
return 0;
}

```

Лістинг A.8 — Програмний код функції edit_hostnames()

```

void edit_hostnames()
{
    char input;
    int index;
    char* buf;
    char* hostname;
    for(;;)
    {
        print_hosts();
        printf("Host index (f to finish): ");
        input = getchar();
        if(input == '\n')
            continue;
        if(input == 'f')
        {
            clear();

```

Продовження лістингу A.8

```

        break;
    }
    else
    {
        ungetc(input, stdin);
        scanf("%d", &index);
        clear();
        if(index < 1 || index > PCONF->host_count)
        {
            puts("Invalid index.");
            continue;
        }
        index--;
        for(;;)
        {
            printf("New hostname (Enter to leave empty): ");
            input = getchar();
            if(input != '\n')
            {
                ungetc(input, stdin);
                buf = malloc(64*sizeof(char));
                scanf("%63s", buf);
                clear();
                if(find(PCONF->hostnames, PCONF-
>host_count, buf) != -1)
                {
                    puts("Hostname already present.");
                    free(buf);
                    continue;
                }
                hostname = strdup(buf);
                free(buf);
            }
            else
                hostname = strdup(" ");
            break;
        }
        free(PCONF->hostnames[index]);
        PCONF->hostnames[index] = hostname;
    }
}

```

Лістинг А.9 — Програмний код функції uninstall()

```

void uninstall()
{
    printf("Are you sure you want to uninstall YggVPN? (y/N): ");
    char input = 'n';
    input = getchar();
    if(input != '\n')
        clear();
    if(input != 'y')
        return;
    printf("Stopping Yggdrasil service . . . ");
    SCM_init();
    if(!stop_yggdrasil_service())
    {
        printf("ERROR: Could not stop Yggdrasil service (%d).\n",
GetLastError());
        SCM_cleanup();
        system("pause");
        return;
    }
    puts("Done.");
    SCM_cleanup();
    printf("Uninstalling Yggdrasil . . . ");
    system("msiexec.exe /x "YGGVPN_CONFIG_DIR"\yggdrasil.msi /qn");
    puts("Done.");
    printf("Deleting YggVPN config . . . ");
    if(!DeleteFileA(YGGVPN_CONFIG_DIR"\yggvpn.conf"))
    {
        puts("ERROR: Could not delete "YGGVPN_CONFIG_DIR"\
yggvpn.conf.\n");
        return;
    }
    puts("Done.");
    printf("Removing the firewall . . . ");
    firewall_remove();
    puts("Done.");
    DeleteFileA(YGGVPN_CONFIG_DIR"\yggdrasil.msi");
    printf("Cleaning up hosts entries . . . ");
    FILE* fp = fopen("C:\\Windows\\System32\\drivers\\etc\\hosts",
"r+");
    if(fp == NULL)
    {
        puts("ERROR: Could not open C:\\Windows\\System32\\
drivers\\etc\\hosts.");
    }
}

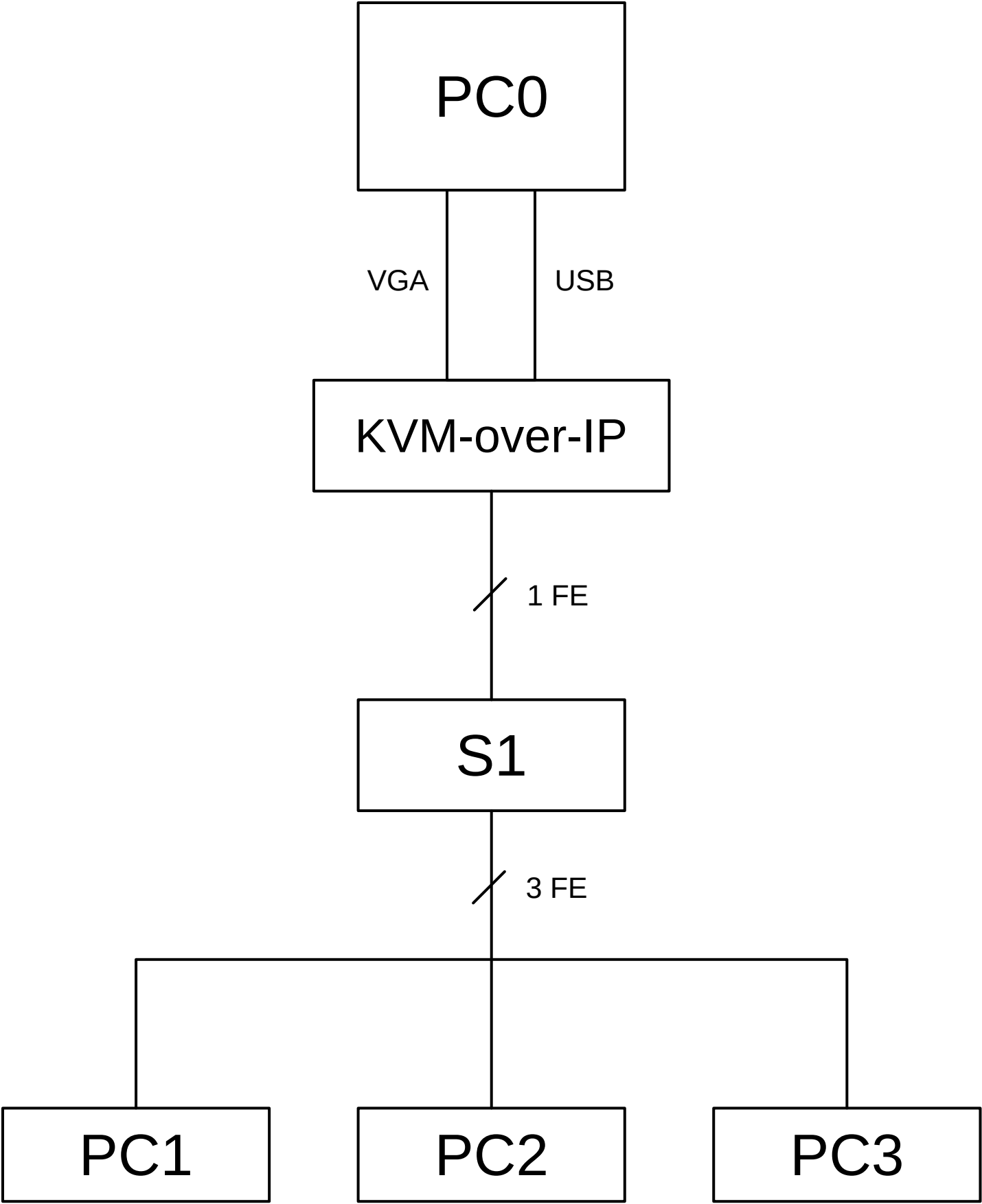
```

Продовження лістингу А.9

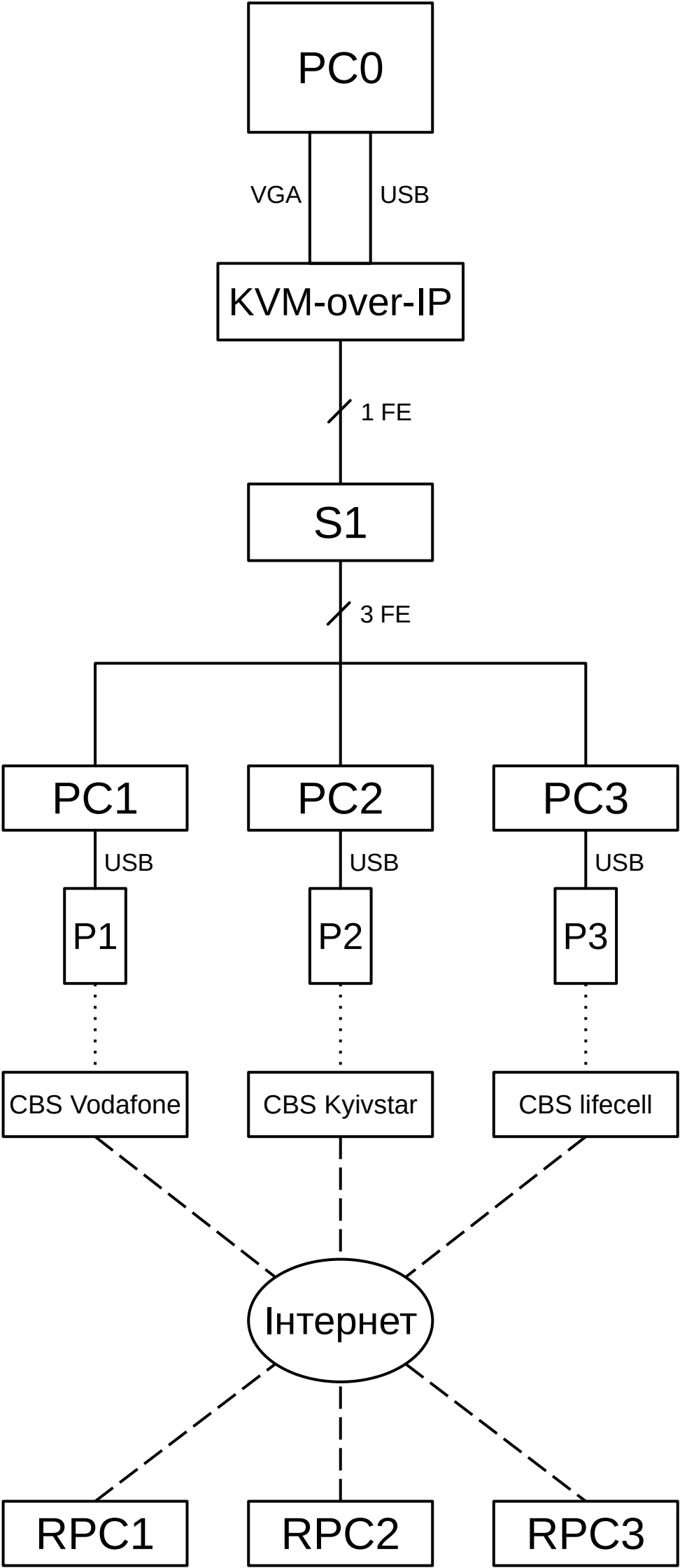
```

        return;
    }
    fseek(fp, 0, SEEK_END);
    int file_size = ftell(fp);
    rewind(fp);
    char* file_content = malloc(file_size);
    fread(file_content, 1, file_size, fp);
    char* offset = strstr(file_content, HOSTS_FILE_SECTION_MARKER);
    if(offset != NULL)
    {
        fp = freopen("C:\\Windows\\System32\\drivers\\etc\\hosts",
"w", fp); //Erase old contents
        offset = strstr(offset + 1, HOSTS_FILE_SECTION_MARKER) +
strlen(HOSTS_FILE_SECTION_MARKER) + 2;
        fputs(offset, fp);
    }
    free(file_content);
    fclose(fp);
    puts("Done.");
    config_reset(PCONF);
    free(PCONF);
    puts("YggVPN uninstalled successfully.");
    system("pause");
    exit(0);
}

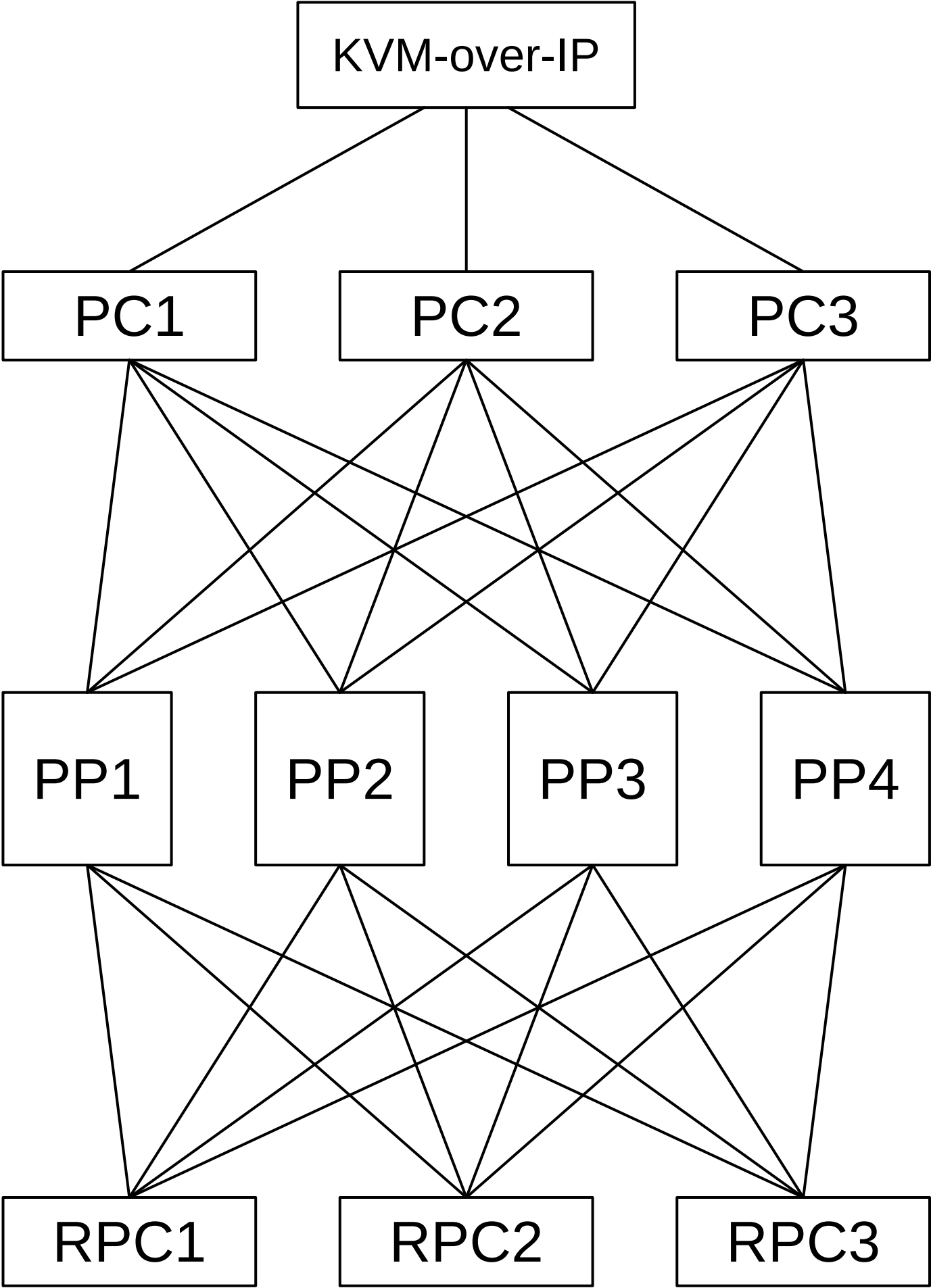
```



					13.02070849.51914 E1				
					Розробка системи віддаленого керування об'єктами критичної інфраструктури	Літ.		Маса	Масштаб
Зм.	Лист.	№ докум.	Підп.	Дата					
Розробив	Пестов О.Д.								
Перевірив	Киричек Г.Г.								
Т. контр.									
					Структурна схема вихідної системи	Лист. 1		Листів 1	
Н. контр.	Щербак Н.В.								
Затв.	Кудерметов Р.К.					НУ «Запорізька політехніка» КНТ-519			



					13.02070849.51914 E1				
					Розробка системи віддаленого керування об'єктами критичної інфраструктури	Літ.		Маса	Масштаб
Зм.	Лист.	№ докум.	Підп.	Дата					
Розробив	Пестов О.Д.								
Перевірив	Киричек Г.Г.								
Т. контр.									
					Структурна схема готової системи	Лист. 1		Листів 1	
Н. контр.	Щербак Н.В.								
Затв.	Кудерметов Р.К.					НУ «Запорізька політехніка» КНТ-519			



					13.02070849.51914 E1			
					Розробка системи віддаленого керування об'єктами критичної інфраструктури	Літ.	Маса	Масштаб
Зм.	Лист.	№ докум.	Підп.	Дата				
Розробив	Пестов О.Д.				Структурна схема логічних з'єднань готової системи			
Перевірив	Киричек Г.Г.							
Т. контр.						Лист. 1	Листів 1	
Н. контр.	Щербак Н.В.					НУ «Запорізька політехніка»		
Затв.	Кудерметов Р.К.					КНТ-519		

185.165.169.234 193.111.114.28 54.37.137.221 193.107.20.230

The Gantt chart displays the schedule for the 2023-2024 season, organized into four horizontal tracks. The x-axis represents dates from 2023-04-15 to 2023-04-28, and the y-axis represents time slots from 08:00 to 20:00. The chart shows various activities, including 'Lecture', 'Workshop', 'Seminar', and 'Guest Lecture', with their respective durations and locations.

Activity	Location	Date	Time Slot	Duration
Lecture	Room 101	2023-04-15	08:00 - 10:00	2h
Workshop	Room 102	2023-04-15	10:00 - 12:00	2h
Seminar	Room 103	2023-04-15	12:00 - 14:00	2h
Guest Lecture	Room 104	2023-04-15	14:00 - 16:00	2h
Lecture	Room 101	2023-04-16	08:00 - 10:00	2h
Workshop	Room 102	2023-04-16	10:00 - 12:00	2h
Seminar	Room 103	2023-04-16	12:00 - 14:00	2h
Guest Lecture	Room 104	2023-04-16	14:00 - 16:00	2h
Lecture	Room 101	2023-04-17	08:00 - 10:00	2h
Workshop	Room 102	2023-04-17	10:00 - 12:00	2h
Seminar	Room 103	2023-04-17	12:00 - 14:00	2h
Guest Lecture	Room 104	2023-04-17	14:00 - 16:00	2h
Lecture	Room 101	2023-04-18	08:00 - 10:00	2h
Workshop	Room 102	2023-04-18	10:00 - 12:00	2h
Seminar	Room 103	2023-04-18	12:00 - 14:00	2h
Guest Lecture	Room 104	2023-04-18	14:00 - 16:00	2h
Lecture	Room 101	2023-04-19	08:00 - 10:00	2h
Workshop	Room 102	2023-04-19	10:00 - 12:00	2h
Seminar	Room 103	2023-04-19	12:00 - 14:00	2h
Guest Lecture	Room 104	2023-04-19	14:00 - 16:00	2h
Lecture	Room 101	2023-04-20	08:00 - 10:00	2h
Workshop	Room 102	2023-04-20	10:00 - 12:00	2h
Seminar	Room 103	2023-04-20	12:00 - 14:00	2h
Guest Lecture	Room 104	2023-04-20	14:00 - 16:00	2h
Lecture	Room 101	2023-04-21	08:00 - 10:00	2h
Workshop	Room 102	2023-04-21	10:00 - 12:00	2h
Seminar	Room 103	2023-04-21	12:00 - 14:00	2h
Guest Lecture	Room 104	2023-04-21	14:00 - 16:00	2h
Lecture	Room 101	2023-04-22	08:00 - 10:00	2h
Workshop	Room 102	2023-04-22	10:00 - 12:00	2h
Seminar	Room 103	2023-04-22	12:00 - 14:00	2h
Guest Lecture	Room 104	2023-04-22	14:00 - 16:00	2h
Lecture	Room 101	2023-04-23	08:00 - 10:00	2h
Workshop	Room 102	2023-04-23	10:00 - 12:00	2h
Seminar	Room 103	2023-04-23	12:00 - 14:00	2h
Guest Lecture	Room 104	2023-04-23	14:00 - 16:00	2h
Lecture	Room 101	2023-04-24	08:00 - 10:00	2h
Workshop	Room 102	2023-04-24	10:00 - 12:00	2h
Seminar	Room 103	2023-04-24	12:00 - 14:00	2h
Guest Lecture	Room 104	2023-04-24	14:00 - 16:00	2h
Lecture	Room 101	2023-04-25	08:00 - 10:00	2h
Workshop	Room 102	2023-04-25	10:00 - 12:00	2h
Seminar	Room 103	2023-04-25	12:00 - 14:00	2h
Guest Lecture	Room 104	2023-04-25	14:00 - 16:00	2h
Lecture	Room 101	2023-04-26	08:00 - 10:00	2h
Workshop	Room 102	2023-04-26	10:00 - 12:00	2h
Seminar	Room 103	2023-04-26	12:00 - 14:00	2h
Guest Lecture	Room 104	2023-04-26	14:00 - 16:00	2h
Lecture	Room 101	2023-04-27	08:00 - 10:00	2h
Workshop	Room 102	2023-04-27	10:00 - 12:00	2h
Seminar	Room 103	2023-04-27	12:00 - 14:00	2h
Guest Lecture	Room 104	2023-04-27	14:00 - 16:00	2h
Lecture	Room 101	2023-04-28	08:00 - 10:00	2h
Workshop	Room 102	2023-04-28	10:00 - 12:00	2h
Seminar	Room 103	2023-04-28	12:00 - 14:00	2h
Guest Lecture	Room 104	2023-04-28	14:00 - 16:00	2h

					13.02070849.51914 ПЛ1				
						Літ.	Маса	Масштаб	
Зм.	Лист.	№ докум.	Підп.	Дата					
Розробив	Пестов О.Д.								
Перевірів	Киричек Г.Г.								
Т. контр.									
						Лист. 1	Листів 1		
Н. контр.	Щербак Н.В.					НУ «Запорізька політехніка» КНТ-519			
Затв.	Кудерметов Р.К.								