Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)
Кафедра комп'ютерних систем та мереж
(повна назва кафедри)

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня

бакалавр
(назва освітнього ступеня)

на тему: *Розподілена комп'ютерна система криптоаналізу на основі Raspberry Pi*

Виконала: студентка *IV* курсу, групи *СІс-43*
спеціальності *123 «Комп'ютерна інженерія»*
(шифр і назва спеціальності)

(підпис) *Халак Х.Р.* (прізвище та ініціали)

Керівник (підпис) *Луцків А.М.* (прізвище та ініціали)

Нормоконтроль (підпис) *Луцик Н.С.* (прізвище та ініціали)

Завідувач кафедри (підпис) *Осухівська Г.М.* (прізвище та ініціали)

Рецензент (підпис) *Марценко С.В.* (прізвище та ініціали)

Тернопіль
2022

Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет *комп'ютерно-інформаційних систем і програмної інженерії*
(повна назва факультету)

Кафедра *комп'ютерних систем та мереж*
(повна назва кафедри)

ЗАТВЕРДЖУЮ
Завідувач кафедри
Осухівська Г.М.
(підпис) (прізвище та ініціали)
« » 2022 р.

З А В Д А Н Н Я

НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня *бакалавр*
(назва освітнього ступеня)

за спеціальністю *123 «Комп'ютерна інженерія»*
(шифр і назва спеціальності)

студентці *Халак Христині Русланівній*
(прізвище, ім'я, по батькові)

1. Тема роботи *Розподілена комп'ютерна система криптоаналізу на основі Raspberry Pi*

Керівник роботи *Луцків Андрій Мирославович, к.т.н., доцент*
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Затверджені наказом ректора від «*23*» *березня* 2022 року № *4.7-180*

2. Термін подання студентом завершеної роботи *24.06.2022 р.*

3. Вихідні дані до роботи *Типова структура організації кластерів, апаратне забезпечення Raspberry PI, технологія організації безпровідних мереж*

4. Зміст роботи (перелік питань, які потрібно розробити)
Вступ. 1. Аналіз технічного завдання і способів організації розподілених систем.
2. Проектування та налаштування кластеру розподілених обчислень на основі Raspberry Pi
3. Програмна реалізація шифрування та дешифрування повідомлень у розподіленій комп'ютерній системі. 4. Безпека життєдіяльності, основи охорони праці. Висновки

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, слайдів)
1. *Архітектура розподіленої комп'ютерної системи криптоаналізу.*
2. *Принцип організації функціонування машини Enigma.*
3. *Алгоритм функціонування Enigma.*
4. *Алгоритм brute force.*
5. *Функція визначення положень роторів.*

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Безпека життєдіяльності,	*Лазарюк В.В., к.т.н., доц. каф. МТ*		
основи охорони праці			

7. Дата видачі завдання _____

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	*Розробка і затвердження технічного завдання*	*23.03-30.03.2022*	
2	*Аналіз технічного завдання*	*30.03-02.04.2022*	
3	*Визначення вимог до апаратного та програмного забезпечення розподіленої комп'ютерної системи*	*02.04-18.04.2022*	
4	*Проектування архітектури кластеру*	*19.04-04.05.2022*	
5	*Налаштування кластеру розподілених обчислень на основі Raspberry PI*	*04.05-16.05.2022*	
6	*Розробка програмного забезпечення управління кластером та шифрування/дешифрування текстових повідомлень*	*16.05-29.05.2022*	
7	*Безпека життєдіяльності, основи охорони праці*	*01.06-08.06.2022*	
8	*Оформлення кваліфікаційної роботи*	*09.06-18.06.2022*	
9	*Попередній захист кваліфікаційної роботи*	*18.06-22.06.2022*	
10	*Захист кваліфікаційної роботи*	*22.06-24.06.2022*	

Студент _____ *Халак Христина Русланівна*
(підпис) (прізвище та ініціали)

Керівник роботи _____ *Луцків Андрій Мирославович*
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Розподілена комп’ютерна система криптоаналізу на основі Raspberry Pi // Кваліфікаційна робота на здобуття освітнього ступеня бакалавр // Халак Христина Русланівна // ТНТУ, спеціальність 123 «Комп’ютерна інженерія»// Тернопіль, 2022 // с.– 84 , рис. – 58 , табл. – 5, аркушів А1 – 5, бібліогр. – 19.

Ключові слова: розподілена система, криптоаналіз, Raspberry PI, Enigma.

У результаті виконання кваліфікаційної роботи побудовано модель розподіленої комп’ютерної системи криптоаналізу на основі Raspberry Pi та реалізовано її вигляді кластеру на основі восьми мінікомп’ютерів для організації обчислень щодо шифрування/дешифрування повідомлень. При цьому забезпечено програмне управління серверними компонентами з клієнтської станції. В основі функціонування розподіленої системи криптоаналізу лежать алгоритми машини Enigma. Алгоритм, що використовувався для дешифрування текстових повідомлень, передбачає перебір можливих комбінацій щодо значень роторів.

Практичне значення одержаних результатів передбачає організацію кластера на основі мінікомп’ютерів Raspberry PI, налаштування параметрів паралельної і розподіленої обробки даних, програмну реалізацію алгоритмів шифрування/дешифрування текстових повідомлень, організацію програмного блок управління розподіленими обчисленнями, забезпечення можливості вибору типу криптоаналізу в залежності від структури і виду вхідних повідомлень.

ABSTRACT

Distributed Raspberry Pi-based cryptanalysis computer system // Кваліфікаційна робота на здобуття освітнього ступеня бакалавр // Khalak Khrystyna Ruslanivna // TNTU, speciality 123 «Computer engineering»// Ternopil, 2022 // p.– 84 , fig. – 58 , tab. – 2, posters A1 – 5, ref. – 19.

Keywords: distributed system, cryptanalysis, Raspberry PI, Enigma.

As a result of the qualification work, a model of a distributed computer cryptanalysis system based on the Raspberry Pi was built and implemented in the form of a cluster based on eight minicomputers for the organization of calculations for encrypting / decrypting messages. At the same time software management of server components from the client station is provided. The algorithms of the Enigma machine are the basis of the functioning of the distributed cryptanalysis system. The algorithm used to decrypt text messages involves searching for possible combinations of rotor values.

The practical significance of the obtained results includes the organization of a cluster based on Raspberry PI minicomputers, setting parameters of parallel and distributed data processing, software implementation of encryption / decryption algorithms for text messages, organization of software unit for distributed computing, incoming messages.

ЗМІСТ

ПЕРЕЛІК ОСНОВНИХ УМОВНИХ ПОЗНАЧЕНЬ,
СИМВОЛІВ І СКОРОЧЕНЬ

КС	Комп’ютерна система
ПЗ	Програмне забезпечення
ПО	Предметна область
РКС	Розподілена комп’ютерна система

ВСТУП

У загальному випадку, розподілена система представляє собою обчислювальне середовище, в якому різні компоненти функціонують на кількох комп'ютерах (або інших обчислювальних пристроях) у мережі. Ці пристрої виконують декомпозицію задач, координуючи свої зусилля, щоб виконати їх ефективніше, у порівнянні з тим, якби це ж завдання опрацьовував один пристрій.

Розподілені системи є важливим розвитком для IT та інформатики, оскільки зростаюча кількість пов'язаних задач є настільки масовими та складними, що один комп'ютер не зможе впоратися з ними самостійно. Але розподілені обчислення також пропонують додаткові переваги в порівнянні з традиційними обчислювальними середовищами.

Розподілені комп'ютерні системи зменшують ризики, пов'язані з наявністю єдиної точки відмови, підвищуючи надійність і відмовостійкість.

Сучасні розподілені системи, як правило, розраховані на масштабування майже в реальному часі. Крім того, можна на льоту залучати додаткові обчислювальні ресурси, підвищуючи продуктивність і ще більше скорочуючи час завершення задач.

Історично, розподілені обчислення були дорогими, складними в налаштуванні та в управлінні. Але завдяки платформам програмного забезпечення як послуги (SaaS), які пропонують розширені функціональні можливості, розподілені обчислення стали більш впорядкованими та доступними для великих і малих підприємств.

Як наслідок, усі види комп'ютерних завдань — від керування базами даних до відеоігор — використовують розподілені обчислення. Насправді, багато типів програмного забезпечення, такі як системи криптовалюти, наукове моделювання, технології блокчейн та платформи штучного інтелекту, були б взагалі неможливими без цих платформ.

У кваліфікаційній роботі пропонується розробити локальну розподілену комп’ютерну систему на основі Raspberry PI для розв’язання задач криптоаналізу, що дозволить підвищити ефективність використання апаратних ресурсів та забезпечити визначений рівень продуктивності при шифруванні і дешифруванні текстових повідомлень.

РОЗДІЛ 1 АНАЛІЗ ТЕХНІЧНОГО ЗАВДАННЯ І СПОСОБІВ ОРГАНІЗАЦІЇ РОЗПОДІЛЕНИХ СИСТЕМ

1.1 Аналіз технічного завдання на проектування комп'ютерної системи збору та аналізу даних з метеостанцій

Розподілена комп'ютерна система криптоаналізу на основі Raspberry Pi призначена для забезпечення ефективного шифрування та дешифрування повідомлень з використанням мінімальних ресурсів апаратного забезпечення. Дана система призначена для виконання задач, які покладаються на сучасні кластери, однак замість повноцінних серверів у даному випадку будуть використовуватись мінікомп'ютери Raspberry PI. На основі кластеру необхідно реалізувати функціональність щодо кодування і декодування повідомлень по типу машини Enigma, що використовувалась під час Другої світової війни.

Практичне застосування розподіленої комп'ютерної системи криптоаналізу важливе при організації процесів, де шифрування та дешифрування відіграють важливу роль. До таких сфер її застосування належать військова галузь, відділи кібербезпеки і кіберполіції, IT-компанії, які займаються кіберзахистом, а також у навчальному процесі при вивченні дисциплін, пов'язаних із захистом інформації.

Алгоритм, який пропонується для декодування повідомлень полягає у повному переборі можливі комбінацій – brute force.

Мета побудови розподіленої комп'ютерної системи криптоаналізу на основі Raspberry Pi полягає у створенні кластеру на основі восьми мінікомп'ютерів для організації обчислень щодо шифрування/дешифрування повідомлень.

До основних задач, які покликана розв'язати дана система належать:

– організація кластеру на основі мінікомп’ютерів Raspberry PI;

– налаштування параметрів паралельної і розподіленої обробки даних;

– програмна реалізація алгоритмів шифрування/дешифрування алгоритмів;

– організація блоку управління розподіленими обчисленням;

– надання засобів діагностики коректності функціонування кластера;

– забезпечення можливості криптоаналізу в залежності від типу вхідних повідомлень;

– підвищення криптостійкості комп’ютерних систем;

– забезпечення можливості фіксації результатів криптоаналізу;

– можливість інтеграції із суміжними системами;

– підвищення якості і надійності в процесі захисту інформації та обміну між вузлами комп’ютерних систем.

До основних задач, які має виконувати розподілена комп’ютерна система криптоаналізу на основі Raspberry Pi належать здатність шифрування текстових повідомлень і їх розкодування на основі принципів організації машини Enigma.

Складність побудови такої системи полягає у великій кількості можливих комбінацій щодо шифрування і відповідно великою кількістю невідомих параметрів при розкодуванні зашифрованого тексту.

Для того, щоб досягти мети роботи необхідно на практиці розв’язати дві основні задачі, зокрема, організація кластера для прискорення процесів шифрування/дешифрування із застосування паралельних і розподілених обчислень, та власне реалізації програмного забезпечення для емуляції роботи машини Enigma.

Організацію кластера необхідно реалізувати не менше, ніж на восьми станціях Raspberry PI, що, крім цього, передбачає наявність одного клієнта для управління ним. Це вимагає зміни налаштування конфігурацій серверних станцій і відповідно й клієнта. Окрім цього, комунікація між клієнтом і серверами повинна бути забезпечення на основі комунікаційної мережі – безпровідної комп’ютерної мережі.

Безпровідну комп’ютерну мережу потрібно налаштувати таким чином, щоб задіяти можливість формування динамічних IP-адрес маршрутизатором у межах станцій, які належать до кластеру.

Програмне забезпечення управління кластером повинно забезпечувати можливість запуску додатків з клієнта, а також віддаленого їхнього перезавантаження та вимкнення.

Розподілена комп’ютерна система криптоаналізу на основі Raspberry PI повинна виконувати функції з шифрування та дешифрування текстових повідомлень з можливістю генерації шифрованого ключа і без нього. Система повинна забезпечувати продуктивність виконання розподілених обчислень з використанням 32 процесорних ядер та 8 ГБ оперативної пам’яті на кластер. Керування розподіленою системою повинно виконуватися зі станції клієнта і передбачати можливість запуску програмного забезпечення, що підлягає паралельному і розподіленому опрацюванні текстових даних. Результатом успішної роботи кластера вважається повна відповідність одержаного зашифрованого повідомлення вхідному, яке підлягає дешифруванню.

Організація розподіленої комп’ютерної системи передбачає застосування таких структурних компонентів як:

– сервер на базі Raspberry PI – 8 шт.;

– клієнт на базі Raspberry PI – 1 шт.;

– маршрутизатор для функціонування розподіленої комп’ютерної системи;

– маршрутизатор для доступу до мережі Інтернет;

– бібліотеки Python для управління кластером;

– бібліотеки Python для реалізації алгоритмів машини Enigma.

Основними функціональними вимогами до розподіленої комп’ютерної системи криптоаналізу на основі Raspberry PI є:

– можливість налаштування діапазону IP-адрес для їх видачі серверам і клієнту системи;

– здатність віддаленого вимкнення серверів;

– можливість дистанційного перезавантаження серверів;

– можливість виконання керованого розподіленого обчислення на серверах;

– здатність запуску програмного забезпечення для виконання задач криптоаналізу;

– здатність виконувати алгоритми, передбачені алгоритмами функціонування машини Enigma;

– здатність розраховувати кількість комбінацій при шифруванні/дешифруванні повідомлень;

– забезпечення можливості реалізації алгоритму brute force;

– здатність до масштабованості кількості серверів у кластері.

Зв’язок між компонентами розподіленої комп’ютерної системи криптоаналізу на основі Raspberry PI організовано на основі технології безпровідного зв’язку WiFi. Обмін даними між клієнтом і серверами визначається налаштуванням маршрутизатора кластера. Доступ до мережі Інтернет забезпечується шляхом використання роутера, віддміного від маршрутизатора кластера для забезпечення коректності функціонування розподіленої системи.

До вимог щодо діагностики системи можна віднести наявність програмних засобів для налаштування конфігурації клієнта і сервера щодо можливості роботи у визначеній безпровідній комп’ютерній мережі. Діагностика серверів може проводитися віддалено або шляхом зміни даних на SD карті. Розклад за яким проводиться діагностика визначає стейкхолдер розподіленої комп’ютерної системи, або у випадку виникнення некоректної роботи компонентів системи.

Перспективами розвитку і модернізації розподіленої комп’ютерної системи є здатність до масштабування кількості серверних компонентів, що дозволить збільшити продуктивність виконання задач криптоаналізу. Окрім цього, важливим показником щодо перспектив розвитку системи є здатність переходу до новіших версій апаратного забезпечення Raspberry PI та оновлення

як системного так і прикладного програмного забезпечення управління кластером.

У разі внесення коректив, елементів додаткової функціональності або заміни існуючого програмного забезпечення, комп'ютерна система має надійно реагувати на ці фактори без втрати існуючих даних.

Розподілена комп'ютерна система криптоаналізу на основі Raspberry Pi повинна мати механізми авторизованого доступу до серверів та клієнта. Окрім цього, надійність функціонування системи повинна проявлятись у достовірності результатів криптоаналізу, їх стійкості та придатності до використання. Важливо підтримувати задану функціональність при зростанні навантаження на обчислювальні ресурси, визначені технічними характеристиками Raspberry PI та маршрутизатора.

При виникненні помилок або збоїв роботи розподіленої комп'ютерної системи, повинна бути забезпечена можливість надійного її функціонування до того часу, поки не буде виявлено причини їх виникнення та усунуто неполадки.

Основними вимогами відносно функцій і задач, які виконує розподілена комп'ютерна система криптоаналізу на основі Raspberry Pi є:

– забезпечення безпровідного доступу і комунікації між компонентами комп'ютерної системи;

– здатність віддаленого управління серверними станціями;

– здатність забезпечувати виконання функцій при емуляції машини Enigma;

– визначена продуктивність при застосуванні алгоритму повного перебору комбінацій brute force;

– можливість запуску задач криптоаналізу з клієнта;

– забезпечення стабільності результатів шифрування/дешифрування текстових повідомлень.

1.2 Аналіз особливостей шифрування та дешифрування Enigma

Enigma – це шифрувальна машина, створена на початку 20 століття для комерційних, дипломатичних та військових застосувань. Під час Другої світової війни машина була прийнята на озброєння німецьких військових для секретного зв'язку. Шифрований код Enigma був зламаний під час війни в Блетчлі-парку, попередниці GCHQ, що означає, що перехоплені повідомлення німецьких військових можна було декодувати та читати.

Вважається, що це вражаюче досягнення скоротило війну, врятувавши багато життів з обох сторін конфлікту. З електричної точки зору, машина Enigma - це просто акумулятор, 26 лампочок і ланцюг перемикача. У ньому немає електроніки, тому це електромеханічний пристрій. Шифрування досягається шляхом зміни шляху електричного струму через електричну схему машини (рисунок 1.1).

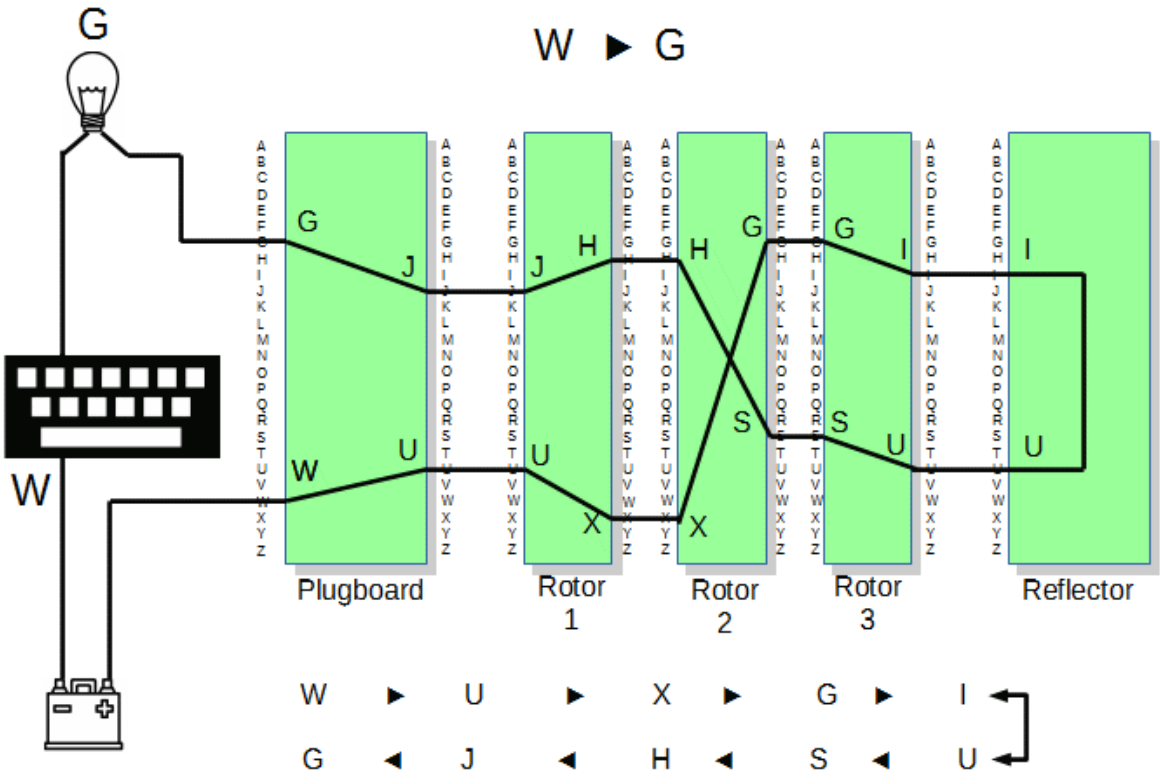


Рисунок 1.1 – Схема Enigma

На схемі вище (рисунок 1.1) можна побачити, як буква, введена на клавіатурі, проходить багато етапів транспонування, перш ніж буде доставлена до лампочки на панелі, що представляє собою зашифрований лист.

Користувач вводить на клавіатурі своє текстове повідомлення символ за символом і читає зашифрований текст, коли кожна лампочка світиться на панелі лампи у відповідь. Завдяки тому, яким чином досягається транспозиція, введена літера ніколи не шифрується як сама по собі (наприклад, введення A ніколи не засвітить лампочку для A). Діаграма може створити враження, що транспозиція букв не змінюється. Але це неправда — спосіб транспонування букв змінюється з кожною літерою, яка вводиться в машину Enigma. Ось чому код Enigma так важко зламати.

Транспозиція змінюється тому, що під час введення кожної літери шлях струму змінюється проходженні до лампочок.

Всередині машини міститься кілька роторів з 26 контактами (по одному на кожну літеру від A до Z) з'єднаними разом, щоб створити шлях струму через «серце» машини. Кожне роторне колесо має 26 електричних контактів з обох боків і змішану систему провідників всередині, так що введені літери переносяться з одного боку на інший. На практиці це означає, що певний ротор транспонує A в E, B в K, C в M і так далі.

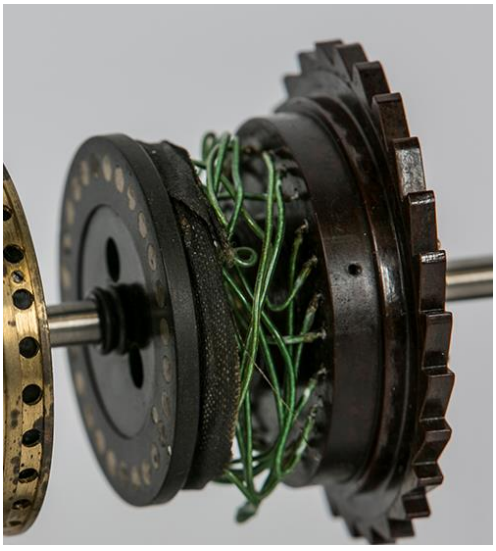


Рисунок 1.2 – Ротор машини Enigma

На рисунку 1.2 можна помітити безлад проводів всередині розширеного роторного колеса від захопленої у Другій світовій війні машини Enigma. Складаючи кілька роторів і використовуючи рефлектор на кінці, щоб повернути струм назад через ротори, кожна літера транспонується багато разів.

При використанні машини Enigma вибирається три ротори з п’яти доступних (були також машини з чотирма роторами). Налаштування транспозиції рефлектора є фіксованим, гарантуючи, що струм повертається назад через машину без зміни транспозиції. Отже, як змінюється шлях струму, що протікає через ці компоненти?

Ротори рухаються таким чином, що символ за символом використовується різна транспозиція, перший ротор просувається покроково в міру введення кожної літери повідомлення, щоразу створюючи новий шлях для поточного. В результаті користувач може ввести «LL», і обидві літери будуть зашифровані по-різному, тому результатом може бути «XV». Після того, як перший ротор переміститься на 26 позицій (один повний оберт), машина починає просування наступної позиції ротора, що знаходиться за поточною.

Частиною того, що ускладнює шифрування Enigma, є той факт, що кожен ротор можна використовувати в різній початковій позиції. Наприклад, якщо ротор установлено в позицію 10 на початку і буква A введена в машину, то він буде введений не там, де за замовчуванням входить A, а де J (літера 10 в алфавіті) за замовчуванням. Ротор рухається вперед на 26 кроків незалежно від його початкового положення. Щоб його було легше налаштувати, ротор позначається кільцем алфавіту. Таким чином, початкова позиція 10 може бути досягнута шляхом установки ротора так, щоб була видна літера J; початкова позиція 3 роторів «JFM» означатиме встановлення першого ротора на J, другого на F, а третього на M.

Крім того, літерні позначення можна зміщувати, надівши кругле кільце на ротор. Обертання контактного кільця приводить до обертання проводки всередині ротора. Наприклад, скажімо, що з контактним кільцем у положенні за замовчуванням, проводка ротора буде транспонувати A в E, B в K, C в M і так

далі. Переміщення контактного кільця на 1 означатиме, що буква A буде транспонована в K, B в M тощо.

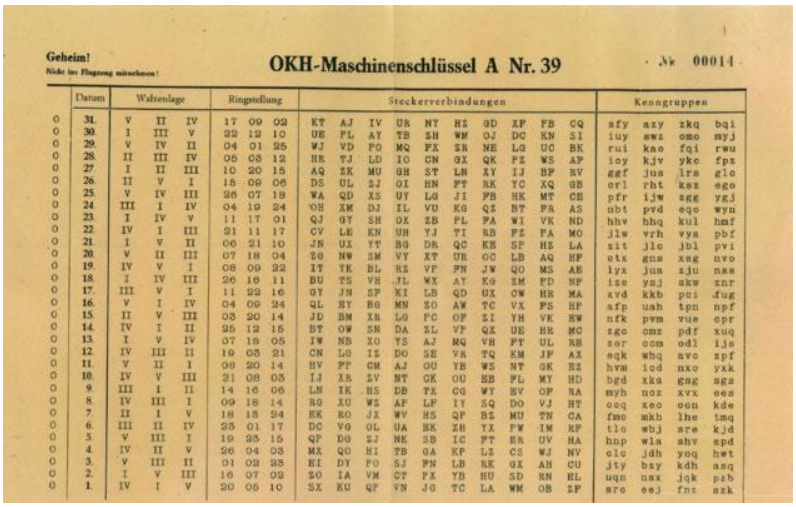
Німецька версія машини Enigma також має вбудовану панель (крайня ліва зелена коробка на діаграмі), яку можна вручну налаштувати так, щоб до десяти пар букв транспонувалися по ходу в ротори і знову, коли вони повертаються.

Поєднуючи три ротори з набору з п’яти, налаштування ротора з 26 позиціями та штепсельну плату з десятьма парами з’єднаних букв, машина Enigma, яку використовували військові Другої світової війни, мала 158962555217826360000 (майже 159 квінтильйонів) різних налаштувань.

Шифрування покладалося на те, щоб як машини Enigma, що є відправниками, так і одержувачі, були налаштовані однаково. Для цього використовувалися секретні ідентичні аркуші налаштувань як на станціях передачі, так і на станціях зв'язку. У цих аркушах зазначено:

– ротори, які слід вибрати, і в якому порядку їх вставляти в машину;
– прокручування кожного ротора;
– літери, які слід змінити за допомогою кільця;
– початкові положення ротора, які потрібно використовувати.

Кожного дня використовувалися різні налаштування машини, а положення запуску ротора навіть змінювалися кожні шість годин, тому налаштування машини були дуже чутливими до часу. Саме тому так ретельно охоронялися аркуші з налаштуваннями, які роздавали військові. Фото аркуша налаштувань Enigma, який зберігається в GCHQ, показано на рисунку 1.3.



Geheim!
Nicht ins Flugzeug mitnehmen!

OKH-Maschinenschlüssel A Nr. 39

№ 00011

	Datum	Walzenlage			Ringstellung			Steckerverbindungen										Kenngruppen			
0	31.	V	II	IV	17	09	02	KT	AJ	IV	UX	NT	HZ	GD	XF	FB	CQ	sfy	azy	zkq	bqi
0	30.	I	III	V	22	12	10	UE	PL	AY	TB	ZN	WM	OJ	DC	KN	SI	iuy	swz	omo	myj
0	29.	V	IV	II	04	01	25	WJ	VD	FO	MQ	FX	ZR	NE	LG	UC	BK	rui	kao	fqi	rwu
0	28.	II	III	IV	05	03	12	HR	TJ	LD	IO	CN	GI	QK	PZ	WS	AF	ioy	kjv	yko	fpx
0	27.	I	II	III	10	20	15	AQ	ZK	MU	GH	ST	LN	XY	IJ	BP	RV	ggf	jus	lra	glo
0	26.	II	V	I	15	09	06	DS	UL	ZJ	OI	HN	FT	RK	YC	XQ	GB	orl	rht	kmz	ego
0	25.	V	IV	III	26	07	18	WA	QD	XS	UY	LG	JI	FB	HK	MT	CE	pfr	ijw	zgg	ygj
0	24.	III	I	IV	04	19	24	OH	XM	DJ	IL	VU	KG	QZ	BT	FR	AS	nbt	pvd	eqo	wyu
0	23.	I	IV	V	11	17	01	QJ	GY	SH	OX	ZB	PL	FA	WI	VK	ND	hhv	hhq	kul	hmf
0	22.	IV	I	III	21	11	17	CV	LE	KN	UH	YJ	TI	RB	FZ	FA	MO	jlw	vrh	vya	pbf
0	21.	I	V	II	06	21	10	JN	UX	YT	BG	DR	QC	KE	SP	HZ	LA	zit	jlc	jbl	pvi
0	20.	V	II	III	07	18	04	ZG	NW	SM	VY	XT	UR	OC	LB	AQ	HF	otx	gns	xng	nvo
0	19.	IV	V	I	08	09	22	IT	YK	BL	RZ	VF	FN	JW	QO	MS	AE	lyx	jus	zju	nas
0	18.	I	IV	III	26	16	11	BU	TS	VH	JL	WX	AY	KG	ZM	PD	NF	ise	ysj	skw	znr
0	17.	III	V	I	11	22	16	GY	JN	SF	KI	LB	QD	UX	OW	HR	MA	xvd	kkb	pui	jug
0	16.	V	I	IV	04	09	24	QL	SY	BG	MN	ZO	AW	TC	VX	FS	HP	afp	uah	tpn	npf
0	15.	II	V	III	03	20	14	JD	BM	XR	LG	PC	OF	ZI	YH	VK	EW	nfk	pvm	vue	opr
0	14.	IV	I	II	25	12	15	BT	OW	SN	DA	ZL	VP	QX	UE	HR	MC	zgo	omz	pdf	xuq
0	13.	I	V	IV	07	18	05	IW	NB	XO	YS	AJ	MQ	VH	FT	UL	RK	zor	oom	odl	ijs
0	12.	IV	III	II	19	03	21	CN	LG	IZ	DO	SE	VR	TQ	KM	JP	AX	egk	whq	avc	zpf
0	11.	V	II	I	08	20	14	HV	PF	CM	AJ	OU	YB	WS	NT	GK	EZ	hvm	iod	nxo	yxk
0	10.	IV	V	III	21	08	03	IJ	XR	ZV	NT	GK	OU	BB	FL	MY	HD	bgd	xka	gsg	sga
0	9.	III	I	II	14	16	06	LN	IK	HS	DB	TX	CG	WY	ZV	OP	RA	myh	noz	xvx	eos
0	8.	IV	III	I	09	18	14	RG	XU	WZ	AF	LF	IY	SQ	DO	VJ	HT	ooq	xeo	oon	kde
0	7.	II	I	V	18	15	24	EK	KO	JZ	WV	HS	QP	BZ	MU	TN	CA	fmo	mkh	lhe	tmq
0	6.	III	II	IV	23	01	17	DC	VG	OL	UA	EK	ZH	YX	FW	IM	RP	tlo	wbj	sre	kjd
0	5.	V	III	I	19	25	15	QP	DG	ZJ	NK	SB	IC	FT	ER	UV	HA	hnp	wla	shv	spd
0	4.	IV	II	V	26	04	03	MX	QO	HI	TB	GA	KP	LZ	CS	WJ	NV	glo	jdh	yoq	hwt
0	3.	V	III	II	01	02	23	EI	DY	FO	SJ	FN	LB	RK	GX	AH	CU	jty	bzy	kdh	anq
0	2.	I	V	III	16	07	02	ZO	IA	VM	CT	PX	YB	HU	SD	RN	EL	uqn	nsx	jqk	pzb
0	1.	IV	I	V	20	05	10	SX	KU	QF	VN	JG	TC	LA	WM	OB	ZF	sro	eej	fnz	szk

Рисунок 1.3 – Фото аркуша налаштувань

Це аркуш налаштувань Enigma, знятий наприкінці Другої світової війни, який GCHQ випустив для цього проекту. У розгорнутому вигляді одного з рядків, показаних нижче (рисунок 1.4), можна побачити, як розташовані різні налаштування:

0	1.	IV	I	V	20	05	10	SX	KU	QF	VN	JG	TC	LA	WM	OB	ZF	sro	eej	fnz	szk

Рисунок 1.4 – Рядок параметрів з аркуша налаштувань Enigma

Налаштування, які були виявлені, стосуються першого дня місяця, отже, «1» у другому стовпці зліва. У наступному стовпці показано, що ротори IV, I і V слід вибирати та використовувати саме в такому порядку.

Четвертий стовпець містить налаштування контактного кільця: ротор IV слід перемістити до положення 20, ротор I до положення 5, а ротор V – до положення 10.

Далі йде підключення «розетки»: S до X, K до U, Q до F і так далі. Нарешті, початкова позиція ротора для чотирьох шестигодинних періодів дня – це «SRC», «EEJ, «FNZ» і «SZK». Крім того, були два рефлектори, B і C, один з яких був обраний для використання. Для програм шифрування та дешифрування припускають використання рефлектора B.

Для кожного повідомлення під час Другої світової війни відправник також вибирав для себе три символи як одноразовий ключ повідомлення — скажімо «RPF». Вони зашифрували цей ключ за допомогою налаштувань із аркуша налаштувань і записали результат — скажімо «QMD». Потім вони продовжували шифрувати своє повідомлення, використовуючи свій одноразовий ключ, тут «RPF», як початкові позиції роторів, записуючи зашифрований текст, який повертає машина. Зашифрована версія ключа «QMD», плюс зашифрований текст потім надсилалися одержувачу по радіо.

Під час Другої світової війни зашифровані Enigma повідомлення, зазвичай, надсилалися азбукою Морзе через короткохвильове радіо. Це означає, що їх можна було легко перехопити на певній відстані, тому німецькі військові в значній мірі покладалися на силу техніки шифрування, щоб зберегти свої повідомлення в секреті. Однак Британія перехопила та успішно розшифрувала повідомлення в Блетчлі-парку. Зашифрована передача Enigma виглядала б так:

Крок 1. Обрати ротори і трибуквенну клавішу повідомлення. Спочатку оператор знайшов на аркуші налаштувань рядок, який відповідає поточному дню місяця. Це дало змогу зрозуміти, як налаштувати машину Enigma, зокрема, які ротори вибрати і в якому порядку їх розмістити, а також визначити початкову позицію ротора для поточного шестигодинного періоду.

Крок 2. Обрати і зашифрувати трибуквенний ключ повідомлення. Оператор обирав одноразовий трибуквенний ключ повідомлення випадковим чином, однак він мав бути унікальним для кожного окремого повідомлення. Скажімо, запропоновано ключ «SCC». Очевидно, що цей ключ не можна було надіслати відкрито. Щоб зашифрувати його для передачі, оператор вводить «SCC» у машину Enigma, параметри якої встановлено відповідно до аркуша налаштувань, і отдержано, наприклад, «PWE» як зашифрований ключ. Тоді цей ключ можна було безпечно надіслати по радіоканалу. Принаймні протягом частини Другої світової війни німецька військова процедура полягала в тому, щоб двічі надіслати та зашифрувати ключ повідомлення. Використовуючи

приклад, оператор набрав би «SCCSCC» і отримав би «PWEHVF». При цьому існує недолік у повторенні клавіші повідомлення.

Раніше було сказано, що жоден простий текстовий лист не шифрується сам по собі. Це означає, що будь-хто, хто перехоплює повідомлення з кодуванням Enigma, знає, що жодна з букв у розшифрованому ключі повідомлення не може бути правильною. У даному прикладі перехоплення ключа «PWE» говорить про те, що «P» — це не перша літера, «W» — не друга, а «E» — не третя. Якщо ключ повідомлення надсилається двічі, як це робили німецькі військові, також відомо, що перша літера не може бути «H», друга не може бути «V», а третя не може бути «F». Це зменшує об'єм пошуку, який потрібно провести, щоб знайти букви простого тексту ключа повідомлення, оскільки вже можна виключити два варіанти для кожної літери в ключі.

Крок 3. Шифрування повідомлення за допомогою незашифрованого ключа. Після того, як ключ повідомлення був обраний і зашифрований, оператор налаштував ротори на незашифровану версію ключа і вводить повідомлення на клавіатурі. Цифри потрібно було прописати повністю, тому що машина Engima не має цифрових клавіш. Також не було пробілу, тому пробіл часто позначався «X». Наприклад, якщо потрібно зашифрувати «це повідомлення таємне», то ввели б «цеповідомленнятаємне».

Крок 4. Надсилання зашифрованого повідомлення по радіо.

Радіооператор надсилає зашифрований ключ і повідомлення азбукою Морзе, використовуючи позивні та скорочений текст, так само, як використовується скорочення в текстових повідомленнях, щоб зменшити кількість символів.

При проектуванні розподіленої комп'ютерної системи криптоаналізу на основі Raspberry PI запропоновано скористатися алгоритмом, що використовувався в Enigma, тобто фактично необхідно реалізувати функціонал машини шифрування/дешифрування на основі кластеру з мінікомп'ютерів.

РОЗДІЛ 2 ПРОЕКТУВАННЯ ТА НАЛАШТУВАННЯ КЛАСТЕРУ РОЗПОДІЛЕНИХ ОБЧИСЛЕНЬ НА ОСНОВІ RASPBERRY PI

2.1 Організація архітектури розподілених обчислень на базі Raspberry PI

Розподілена комп'ютерна система криптоаналізу представляє собою кластерний комп'ютер і є альтернативою хмарній інфраструктурі. Пропонується в якості обчислювальних ядер використати потужність восьми серверних ЦП (32 ядра), що дозволить запускати і виконувати обчислення з клієнтського вузла набагато швидше, ніж клієнт може виконувати їх самостійно.

В якості програмного забезпечення управління кластером пропонується використовувати Python 3, що проявляється у забезпеченні можливості запуску відповідних скриптів для криптоаналізу з клієнтської станції.

При організації розподіленої комп'ютерної системи необхідно виконати задачі, які умовно формують три стадії:

– організація мережі Wi-Fi для кластера за допомогою спеціального маршрутизатора;

– створення клієнтської машини;

– створення кластера на основі восьми серверів.

Насправді при організації розподіленої комп'ютерної системи не обов'язково використовувати вісім серверів, оскільки кластер працюватиме з будь-якою кількістю серверів з врахуванням обмежень, визначених продуктивністю WiFi маршрутизатора.

Для того, щоб візуалізувати навантаження на кожен з серверів у кластері, можна встановити світлодіодні матриці Pimoroni Unicorn HAT 8x8 на кожен сервер. Сценарій керування bash на клієнтській машині можна використовувати для зміни шаблонів на Unicorn HAT.

На рисунку 2.1 показано схему організації розподіленої комп'ютерної системи криптоаналізу на основі Raspberry PI.

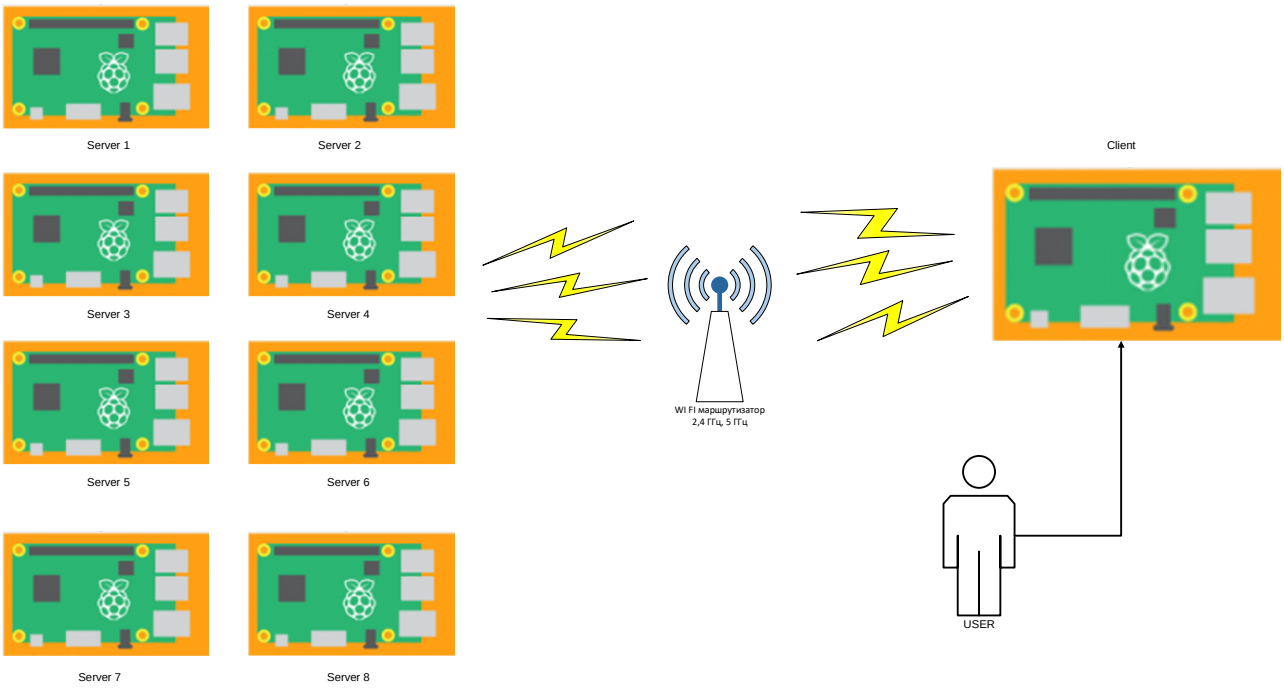


Рисунок 2.1 – Схема організації розподіленої комп'ютерної системи криптоаналізу на основі Raspberry PI

Аналізуючи схему на рисунку 2.1, можна побачити, що сам кластер для виконання задач криптоаналізу використовує 8 станцій. Користувач запускає Python скрипт з клієнтської станції, що комунікує з кластером через маршрутизатор, який працює на частоті 2,4 ГГц.

При організації кластеру розподіленої системи передбачається використання наступного апаратного забезпечення:

– Raspberry PI – 9 одиниць;
– матриця світлодіодів Unicorn HAT – 8 одиниць;
– кабель Micro USB – 8 одиниць;
– безпровідний маршрутизатор – 1 одиниця;
– концентратор живлення – 1 одиниця;
– кабель Ethernet – 1 одиниця.

При організації розподіленої комп’ютерної системи необхідно дев’ять Raspberry Pi 3, вісім з яких утворюють кластер. Лише один мінікомп’ютер потребує звичайної периферії, щоб виконувати роль клієнтської машини. У випадку наявності стійки, кластер можна закріпити, як показано на рисунку 2.2.

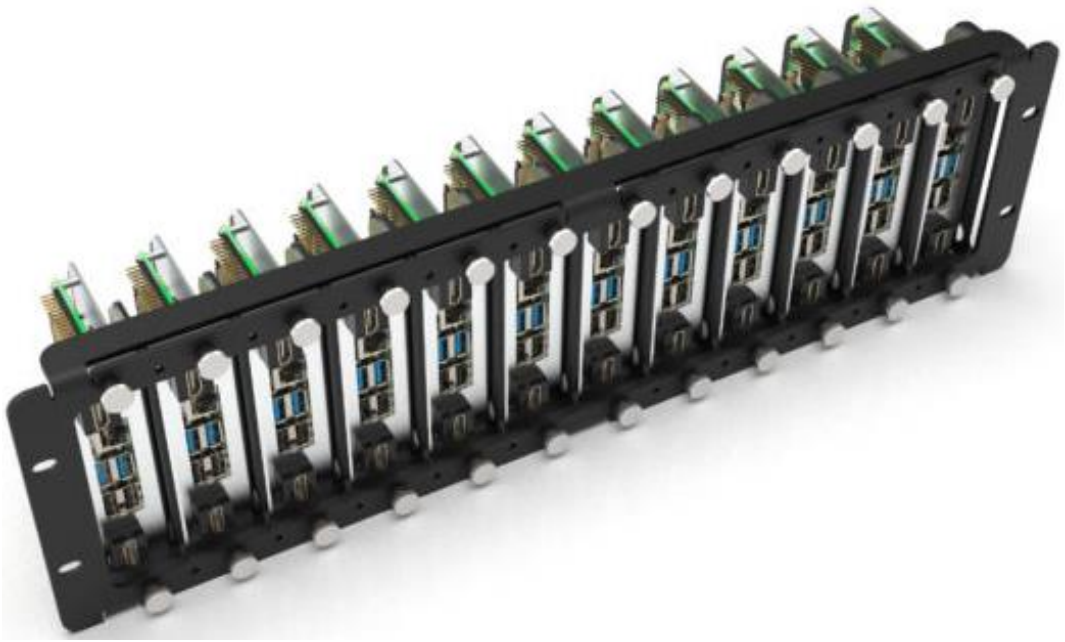


Рисунок 2.2 – 19” стійка з Raspberry PI

Не рекомендується застосування маршрутизатора, який використовувався у постачальника широкосмугових послуг. Хоча використання такого маршрутизатора може доволі не погано працювати, функції в маршрутизаторі можуть бути заблоковані або налаштовані на умови, відмінні від тих, які використовуються за замовчуванням.

При організації розподіленої комп’ютерної системи криптоаналізу необхідне використання таких бібліотек Python, як:

– dispy;
– nmap;
– psutil.

Raspberry PI у кластері будуть комунікувати через виділену локальну мережу WiFi, створену бездротовим маршрутизатором. Маршрутизатор не повинен бути підключений до мережі Інтернет при роботі кластера, а також він

не повинен бути доступний для онлайн налаштування. Рекомендується використовувати абсолютно новий маршрутизатор або такий, який містить налаштування за замовчуванням.

2.2 Налаштування безпровідної мережі для функціонування розподілної системи

Перш за все, для налаштування безпровідної мережі на основі якої буде розгорнуто кластер, потрібно увімкнути маршрутизатор та підключити його за допомогою Ethernet кабелю до комп'ютера або Raspberry PI. Важливим є те, щоб на ПК чи міні комп'ютері було встановлено веб-браузер.

Для подальшого налаштування безпровідного маршрутизатора потрібно дотримуватись інструкцій виробника пристрою. Перейшовши за допомогою веб-браузера до адміністративної частини програмного забезпечення налаштувань пристрою необхідно перш за все авторизуватися як адміністратор. Облікові дані для входу «адміністратора» надаються виробником WiFi маршрутизатора. Далі потрібно вказати назву безпровідної мережі (SSID), наприклад, «OctaPi». Приклад налаштування назви мережі показано на рисунку 2.3

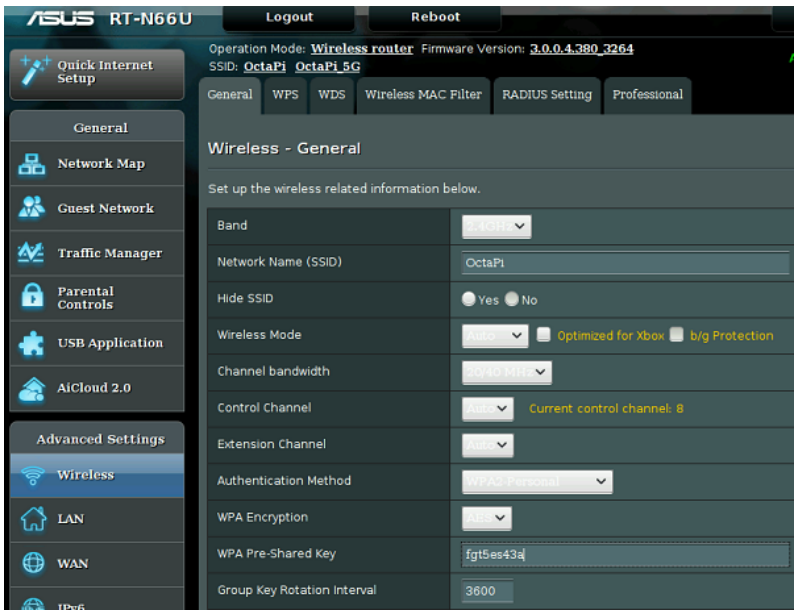


Рисунок 2.3 – Вікно налаштування назви безпровідної мережі

Оскільки Raspberry Pi 3 працюють лише з Wi-Fi в частотному діапазоні 2,4 ГГц, тому налаштування 5 ГГц можна або ігнорувати, або взагалі вимкнути.

Вказавши назву безпровідної мережі, наступний крок полягає у налаштуванні розділу LAN, зокрема LAN IP. Для цього потрібно змінити IP-адресу маршрутизатора на 192.168.1.1 – знову ж таки, інтерфейс адміністратора кожного маршрутизатора буде відрізнятися. На рисунку 2.4 показано приклад налаштування LAN IP.

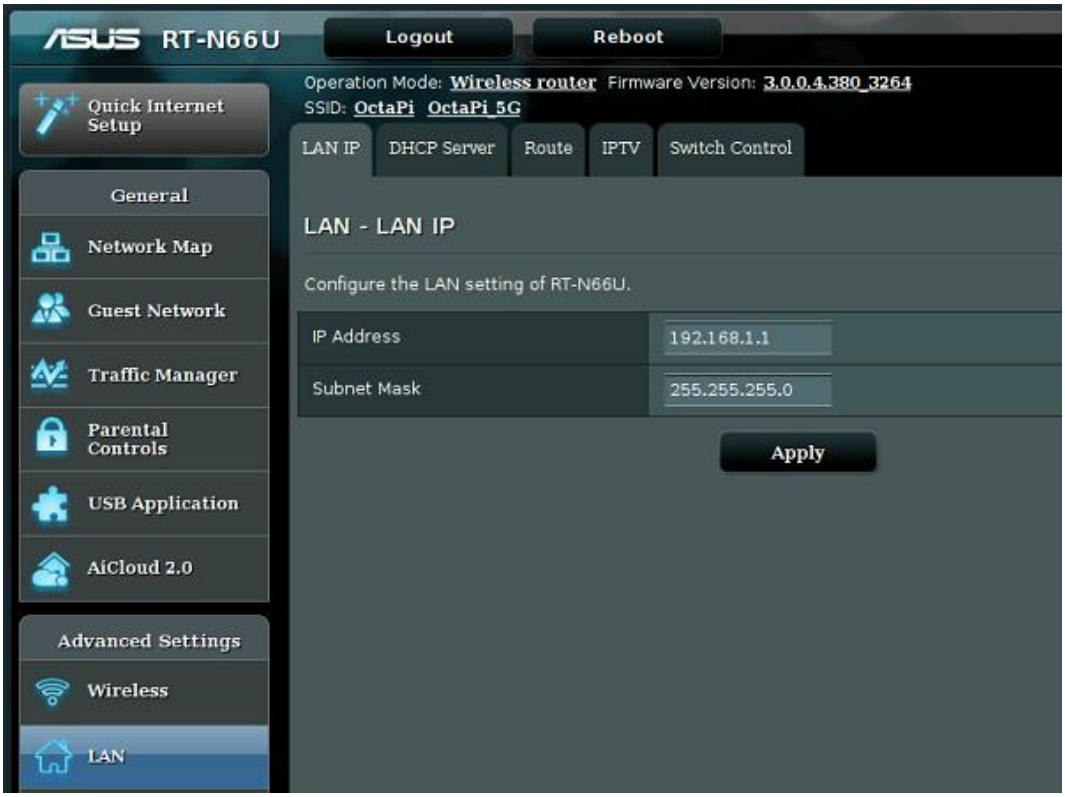


Рисунок 2.4 – Налаштування LAN IP маршрутизатора

Після цього кроку варто перезавантажити маршрутизатор і знову увійти як «адміністратор» для встановлення паролю мережі, що мітиться у розділі «Безпека бездротового зв’язку».

Далі потрібно налаштувати DHCP – протокол, який використовується для автоматичної видачі IP-адрес. Клієнт і сервери будуть використовувати це для визначення своїх IP-адрес. Налаштування DHCP можуть бути в розділі «LAN». Перш за все, варто переконатися, що DHCP увімкнено і після цього задати потрібний діапазону IP-адрес, які буде видавати DHCP. На рисунку 2.5 показано

налаштування DHCP, що видаватиме діапазон IP-адрес, починаючи з 192.168.1.2 і закінчуючи 192.168.1.254.

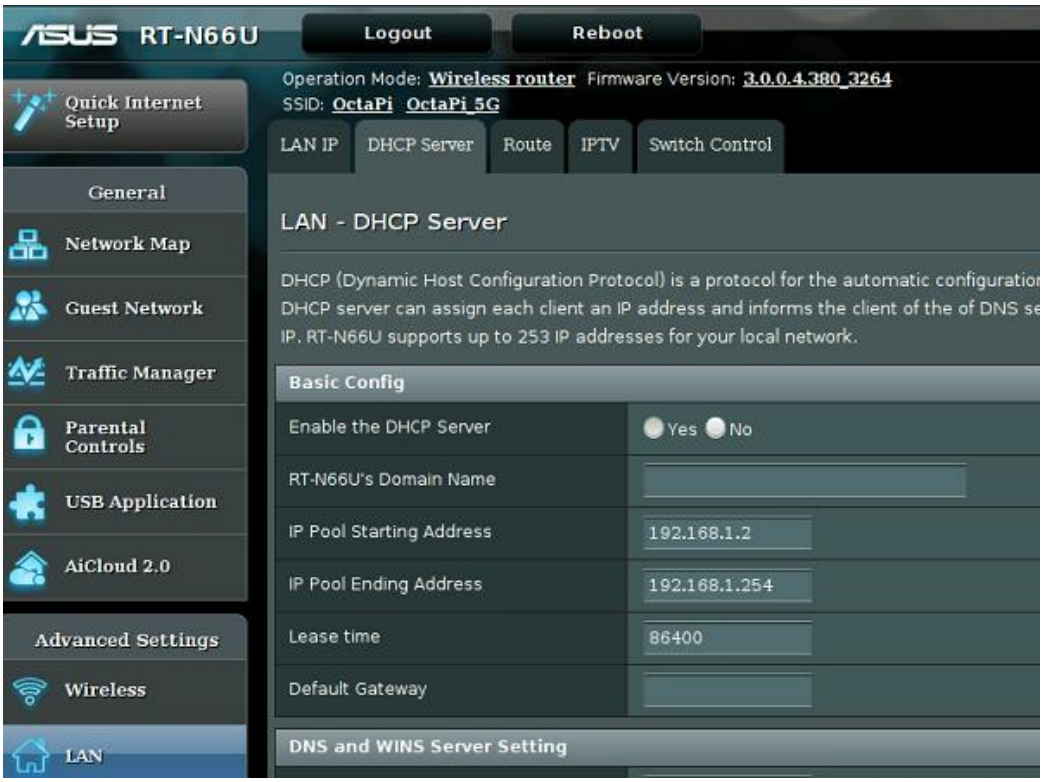


Рисунок 2.5 – Налаштування DHCP безпровідного маршрутизатора

Якщо маршрутизатор володіє налаштуванням часу «оренди» IP (Lease time), то значення часу варто встановити якомога більшим. Час «оренди» – це період часу до того, як DHCP перерозподілить IP-адреси, який повинен бути довгим, щоб уникнути переривання з'єднання між клієнтом і серверами.

Після проведення усіх маніпуляцій щодо налаштування маршрутизатора для набуття зміни і їх застосування потрібно перезавантажити WiFi-роутер.

2.3 Налаштування параметрів розподіленої комп'ютерної системи криптоаналізу

Як було зазначено раніше, один із комп'ютерів Raspberry Pi використовуватиметься як клієнтська машина, що надає доступ до серверів у кластері OctaPi. До клієнта потрібно підключити звичайні периферійні пристрої

(монітор, клавіатуру, мишу), щоб мати можливість використовувати його для керування кластером. На карті micro SD варто встановити останню версію Raspbian, дотримуючись інструкцій з інсталяції програмного забезпечення. Окрім цього, потрібно, щоб в даний момент часу, Raspberry Pi також був підключений до мережі Інтернет.

Наступні кроки полягають у встановлені необхідного програмного забезпечення, зокрема, бібліотек Python для роботи з кластером. Це можна роботи за допомогою терміналу, запуск якого показано на рисунку 2.6.

Рисунок 2.6 – Запуск терміналу в ОС Raspbian

У командному рядку необхідно ввести команду для інсталяції бібліотеки dispy, що представляє собою розподілену реалізацію Python, яка дозволить писати код на клієнті та запускати його на серверах. На рисунку 2.7 показано фрагмент командного рядка з цією командою.

```
sudo pip3 install dispy
```

Рисунок 2.7 – Інсталяції бібліотеки dispy

Додаткову інформацію щодо використання та призначення бібліотеки dispy можна знайти на відповідному сайті: розподілені та паралельні обчислення з/для Python.

Наступна бібліотека, яку необхідно встановити для коректного функціонування кластера – nmap. Для цього потрібно виконати команду, яка представлена на рисунку 2.8.

```
sudo apt-get install nmap
```

Рисунок 2.8 – Інсталяція бібліотеки nmap

Бібліотека nmap використовується для виявлення IP-адрес серверів Raspberry Pi, що утворюють кластер. За її допомогою можна вимикати або перезавантажувати вузли розподіленої комп'ютерної системи.

У випадку використання матриць світлодіодів Unicorn HAT, необхідно встановити відповідне програмне забезпечення для їхньої ініціалізації та налаштування. Для цього у командній стрічці потрібно виконати команду, як показано на рисунку 2.9.

```
curl https://get.pimoroni.com/unicornhat | bash
```

Рисунок 2.9 – Інсталяція ПЗ для управління Unicorn HAT

Після виконання команд і налаштувань, наведених вище команд, необхідно перезавантажити Raspberry PI. Наступний крок полягає у завантаженні з репозитарію github клієнтського програмного забезпечення для кластеру, однак перед цим потрібно переконатися, що поточне місцезнаходження відповідає директорії /home/pi. Для завантаження клієнтського ПЗ виконується команда, як наведено на рисунку 2.10.

```
git clone https://github.com/raspberrypilearning/octapi-setup.git
```

Рисунок 2.10 – Завантаження клієнтського ПЗ для кластера

Клієнтське програмне забезпечення містить приклади вихідного коду мовою програмування Python 3 і сценарій керування bash для перезавантаження та завершення роботи кластера. Сценарій керування також можна використовувати з Unicorn HAT. Задля забезпечення коректності функціонування кластера потрібно перемістити усі файли з папки клієнта, яку у директорію /home/pi, як показано на рисунку 2.11.

```
mv /home/pi/octapi-setup/client/* /home/pi
```

Рисунок 2.11 – Переміщення файлів у директорію /home/pi

2.3.1 Налаштування параметрів сервера у кластері

Кожен із комп'ютерів Raspberry Pi 3 у кластері має підготувати власну micro SD картку з відповідним програмним забезпеченням. Оскільки кожна карта ідентична, то можна налаштувати лише один сервер, перевірити його роботу, а потім реплікувати SD-карту для інших серверів.

Перший крок при налаштуванні сервера кластеру полягає у тому, щоб на новій картці micro SD встановити останню версію Raspbian. Далі у вікні терміналу, як і у випадку з клієнтом, потрібно встановити бібліотеку dispy, як показано раніше на рисунку 2.7.

Після встановлення dispy, подібним чином потрібно інсталювати psutil, ввівши цю команду в командний рядок (рисунок 2.12).

```
sudo pip3 install psutil
```

Рисунок 2.12 – Інсталяція psutil

Після виконання команди, показаної на рисунку 2.12, потрібно перезавантажити Raspberry Pi і знову відкрити термінал для того, щоб переконайтеся в тому, що поточне місце знаходження відповідає директорії /home/pi. Для цього можна скористатися командою cd /home/pi.

Наступний крок налаштування сервера розподіленої комп'ютерної системи полягає у завантаженні сценарію bash «start_unicorn.sh», який виконується командою, поданою на рисунку 2.13:

```
wget https://raw.githubusercontent.com/raspberrypilearning/octapi-setup/master/se
```

Рисунок 2.13 – Завантаження сценарію «start_unicorn.sh»

Виконання завантаженого скрипта виконується за допомогою команди, показаної на рисунку 2.14.

```
chmod u+x ./start_unicorn.sh
```

Рисунок 2.14 – Виконання завантаженого скрипта

Ще один скрипт, який необхідний для функціонування кластеру і який запускається під час завантаження на сервері – dispynode. Для запуску dispynode потрібно виконати команду, яка показана на рисунку 2.15.

```
nano start_dispynode.sh
```

Рисунок 2.15 – Запуск dispynode

Коли nano запуститься, то необхідно додати рядки коду, які наведено на рисунку 2.16.

```
#!/bin/sh -e
sleep 30
_IP=$(hostname -I | awk '{print $1}')
dispynode.py -i $_IP --daemon
```

Рисунок 2.16 – Скрипт налаштування параметрів dispynode

Скрипт, наведений на рисунку 2.16, виконує наступну функціональність: :

– перехід у режим сну на 30 секунд, щоб дозволити серверу підключитися до мережі;

– отримати IP-адресу сервера;

– запустити демон dispynode, який буде слухати інструкції від клієнта.

Після того, як виконано налаштування, описане вище, необхідно виконати збереження параметрів за допомогою комбінацій клавіш Ctrl+O, а потім Ctrl+X, щоб вийти з редактора nano. Для того, щоб скрипт був втконуватним необхідно у командному рядку ввести наступну команду (рисунок 2.17).

```
chmod +x start_dispynode.sh
```

Рисунок 2.17 – Забезпечення виконуваності скрипта dispynode з параметрами

Збережений скрипт потрібен для запуску під час завантаження сервера. Для цього можна використовувати crontab, як показано на рисунку 2.18.

```
crontab -e
```

Рисунок 2.18 – crontab

У кінці файлі, що відкривається після запуску crontab, необхідно додати рядок, який показано на рис. 2.19.

```
@reboot sudo /home/pi/start_dispynode.sh
```

Рисунок 2.19 – Модифікація файлу crontab

Далі потрібно перевірити чи увімкнутий SSH. Це робиться з метою, щоб забезпечити віддалений доступ з командного рядка до сервера. Для цього у меню

«Preferences» потрібно вибрати «Raspberry Pi Configuration» (рисунок 2.20), а потім на вкладці «Interfaces» переконатися у встановленні перемикача у положення, що відповідає SSH: enabled (рисунок 2.21).

Рисунок 2.20 – Вікно налаштувань Raspberry PI

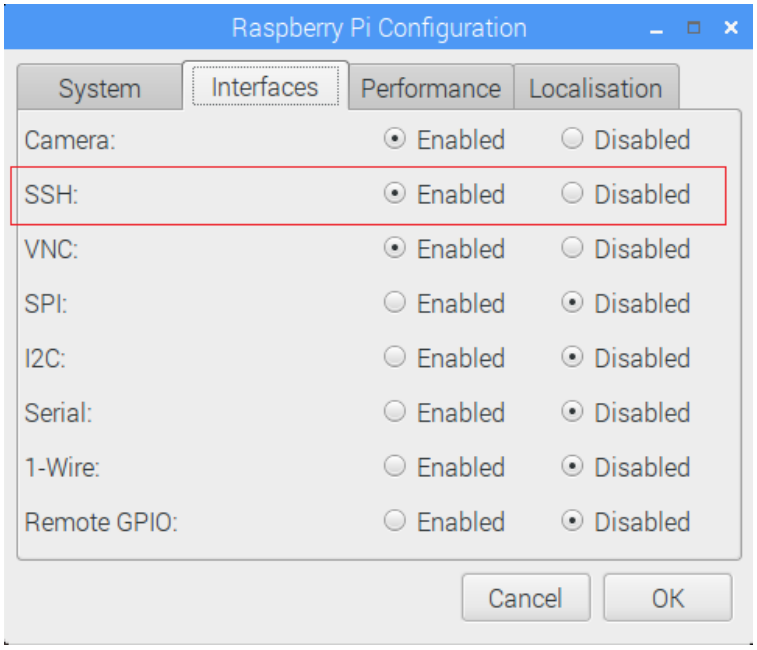


Рисунок 2.21 – Дозвіл на використання SSH

Наступний крок полягає у відключенні Raspberry PI від мережі Інтернет і підключенні до безпровідної мережі, організацію якої за допомогою маршрутизатора наведено у п. 2.2. Для цього потрібно натиснути на символ WiFi у верхній частині робочого столу та обрати мережу «OctaPi» (рисунок 2.23).

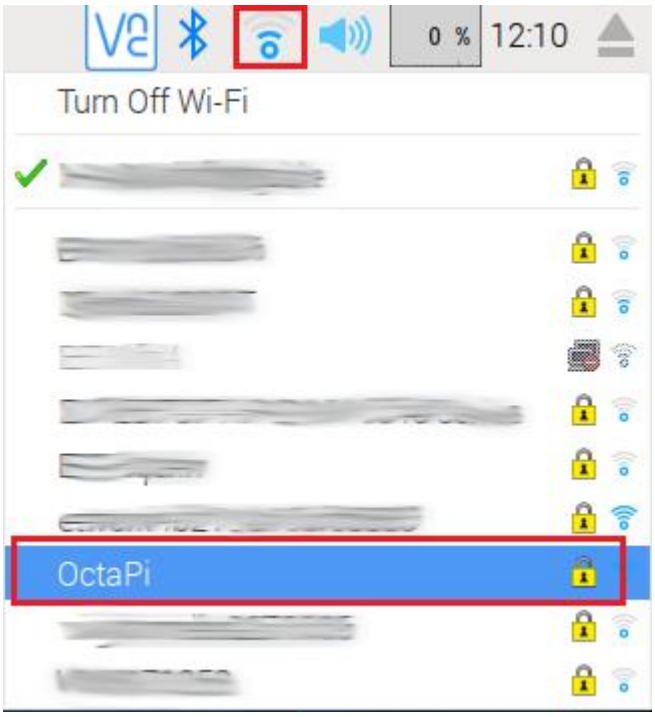


Рисунок 2.23 – Під’єднання до безпровідної мережі

При під’єднанні до маршрутизатора безпровідної мережі необхідно у відповідном полі вказати встановлений раніше пароль (рисунок 2.24).

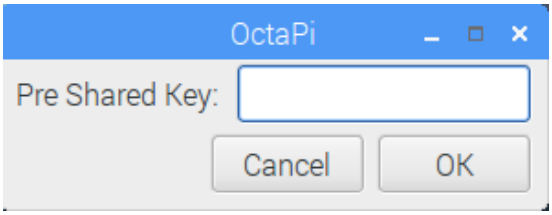


Рисунок 2.24 – Поле для вводу пароля маршрутизатора

Зробивши це, сервер запам’ятовуватиме облікові дані WiFi та входитиме у визначену мережу щоразу під час завантаження. Далі потрібно видалити будь-яку попередню інформацію про WiFi, щоб уникнути плутанини. У вікні

терміналу необхідно ввести таку команду, щоб відредагувати файл wpa_supplicant.conf (рисунок 2.25).

```
sudo nano /etc/wpa_supplicant/wpa_supplicant.conf
```

Рисунок 2.25 – Команда редагування файлу wpa_supplicant.conf

Вміст файлу, після виконання команди, виглядає таким чином, як показано на рисунку 2.26.

```
ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev
update_config=1
country=GB

network={
    ssid="OctaPi"
    psk="mynetworkpassword"
    key_mgmt=WPA-PSK
}
```

Рисунок 2.26 – Вміст файлу конфігурації

У відкритому файлі необхідно видалити усі розділи «network { }» для інших мереж, натиснути Ctrl + o, щоб зберегти, і Ctrl + x, щоб вийти. Якщо альтернативні мережі Wi-Fi не видалено, сервер може увійти в неправильну мережу і бути недоступним для клієнта. Після завершення налаштувань потрібно вимкнути сервер Raspberry Pi.

2.3.2 Налаштування параметрів клієнтської станції у безпровідній мережі

Для налаштування параметрів клієнта у створеній безпровідній мережі преш за все потрібно підключити монітор, клавіатуру та мишу та увімкнути Raspberry Pi, що містить клієнтську SD-карту.

Для клієнта потрібно повторити ті ж самі кроки, що й для сервера, увійшовши в мережу та видаливши будь-які альтернативні параметри WiFi з файлу wpa_supplicant. Далі, тримаючи клієнта увімкненим, необхідно завантажити один сервер Raspberry Pi шляхом підключення лише кабеля живлення. У вікні терміналу клієнта потрібно ввести команду, як представлено на рисунку 2.27, для того, щоб одержати IP-адресу клієнта Raspberry Pi.

```
hostname -I
```

Рисунок 2.27 – Команда для одержання адреси клієнта

Для того, щоб дізнатися IP-адресу сервера Raspberry Pi, потрібно виконати команду, як показано на рисунку 2.28.

```
nmap -sP 192.168.1.*
```

Рисунок 2.28 – Команда одержання IP-адреси сервера

Програмне забезпечення nmap сканує мережу, щоб знайти IP-адреси підключених до неї пристроїв. Це потрібно зробити у локальній мережі, щоб клієнтська машина могла спілкуватися з комп'ютерами Raspberry Pi, які утворюють кластер. Не варто запускати nmap у мережі, яка підключена до Інтернету, оскільки це є потужним програмним забезпеченням, і його використання для сканування мережі, може вважатися зломом, а в деяких країнах навіть незаконним.

Для створення ключа з метою аутентифікації клієнта з сервером, на клієнті з терміналу потрібно запустити ssh-keygen, як показано на рисунку 2.29.

```
ssh-keygen
```

Рисунок 2.29 – Запуск ssh-keygen

Після запуску ssh-keygen, у випадку, коли буде запит на місце збереження ключа, потрібно двічі натиснути Enter. Пароль варто залишити порожнім. Цей ключ використовується, щоб допомогти сценарію cluster_action.sh (постачається з клієнтським програмним забезпеченням) працювати на серверах. Далі необхідно скопіювати згенерований ключ і зберегти його на сервері. Для цього слід виконати команду, показану на рисунку 2.30, замінивши <remote ip> на IP-адресу сервера.

```
ssh-copy-id -i ~/.ssh/id_rsa.pub <remote ip>
```

Рисунок 2.30 – Код копіювання ssh-ключа

Далі буде запит на продовження підключення і введення паролю сервера, який є стандартним паролем для Raspberry. На цьому підготовка клієнта і сервера завершена. Тепер потрібно протестувати коректність функціонування розподіленої комп'ютерної системи.

2.4 Перевірка працездатності розподіленої комп'ютерної системи

Перед початком тестування працездатності побудованого кластера перш за все потрібно переконатися в тому, що:

– маршрутизатор безпровідної мережі увімкнено та повністю завантажено;

– клієнт завантажується з підключеними периферійними пристроями;

– сервер завантажується лише з підключеним кабелем живлення.

Далі потрібно відкрити термінал клієнта і переконатися, що поточною директорією є каталог /home/pi. Після цього необхідно ввести команду, як показано на рисунку 2.31 для запуску ПЗ compute.py, що завантажено разом із клієнтським програмним забезпеченням.

```
sudo python3 compute.py
```

Рисунок 2.31 – Запуск тестової програми

Сценарій compute.py Python виконує 15 завдань сервері. Усі вони лише випадкові затримки перед поверненням. Якщо сервер працює правильно, завдання будуть завершені приблизно за хвилину, а в терміналі відобразиться таблиця зі статистикою для програми, яка показана на рисунку 2.32.

```
Node |  CPUs |    Jobs |  Sec/Job | Node Time Sec |    Sent |    Rcvd
-------------------------------------------------------------------------
     raspberrypi |     4 |      16 |     13.2 |         211.5 |   2.8 K |   3.9 K

Total job time: 211.545 sec, wall time: 59.484 sec, speedup: 3.556
```

Рисунок 2.32 – Результат виконання тестової програми на кластері

Якщо сценарій compute.py не працює, необхідно переглянути попередні кроки і переконатися, що клієнт, сервер і маршрутизатор правильно налаштовані та працюють належним чином. Якщо тест спрацював, можна використовувати клієнт для вимкнення сервера вручну (рисунок 2.33).

```
ssh <remote_ip>
sudo shutdown -HP now
```

Рисунок 2.33 – Вимкнення сервера з клієнтської станції

Можливо, на даному етапі знадобиться знову використовувати nmap, щоб знайти IP-адресу сервера, якщо вона змінилася під час перезавантаження маршрутизатора WiFi. У майбутньому для цього буде використовуватися скрипт cluster_action.sh.

Після вимкнення сервера потрібно склонувати його micro SD карту ще на 7 карток.

Варто відмітити, що для забезпечення живлення 8-ми Raspberry PI можна просто підключити кожен із восьми комп’ютерів Raspberry Pi окремо за допомогою восьми стандартних блоків живлення, або використоати концентратор USB або зарядний пристрій, щоб централізовано живити їх.

Важливо, щоб на кожному Raspberry Pi 3 було достатньо живлення, щоб забезпечити 2,4 А, оскільки не всі USB-концентратори/зарядні пристрої можуть живити 2,4 А на кожному порті.

Якщо використовується Unicorn HAT, то необхідно встановити HAT на роз’єм GPIO кожного із серверів. За бажанням можна закріпити вісім Raspberry Pi 3 на дошці. На рисунку 2.34 показано приклад застосування Unicorn HAT.

Рисунок 2.34 – Застосування Unicorn HAT

Коли зроблено всі налаштування на 8-ми серверах, які утворюють розподілену комп’ютерну систему, потрібно знову запустити приклад програмного забезпечення compute.py, виконавши відповідну команду. Якщо кластер працює правильно, наприкінці запуску всі сервери, які використовуються для виконання завдання, будуть відображені у таблиці. Результат повинен виглядати так, як показано на рисунку 2.35.

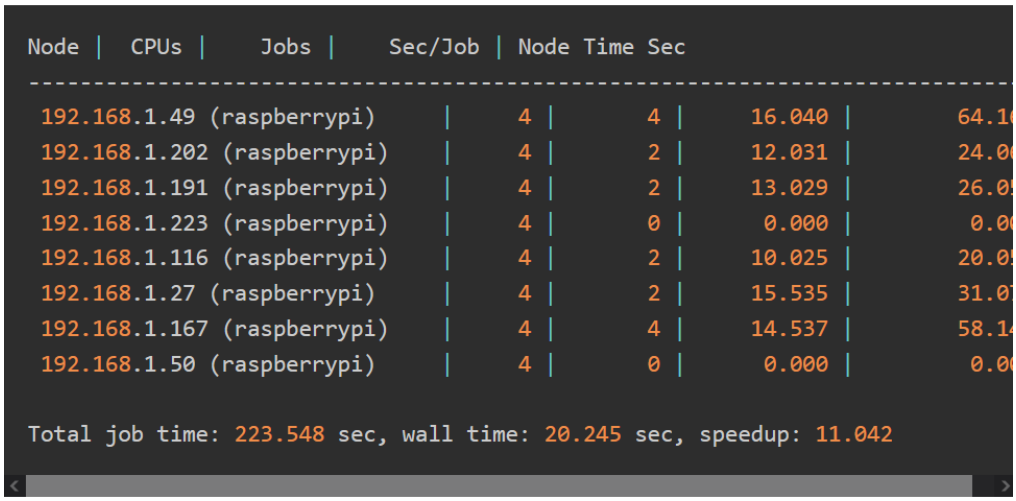


Рисунок 2.35 – Перевірка коректності роботи розподіленої комп'ютерної системи

Якщо запустити інший скрипт compute_pi_efficient.py, то у результаті його виконання одержують результат, як показано на рисунку 2.36.

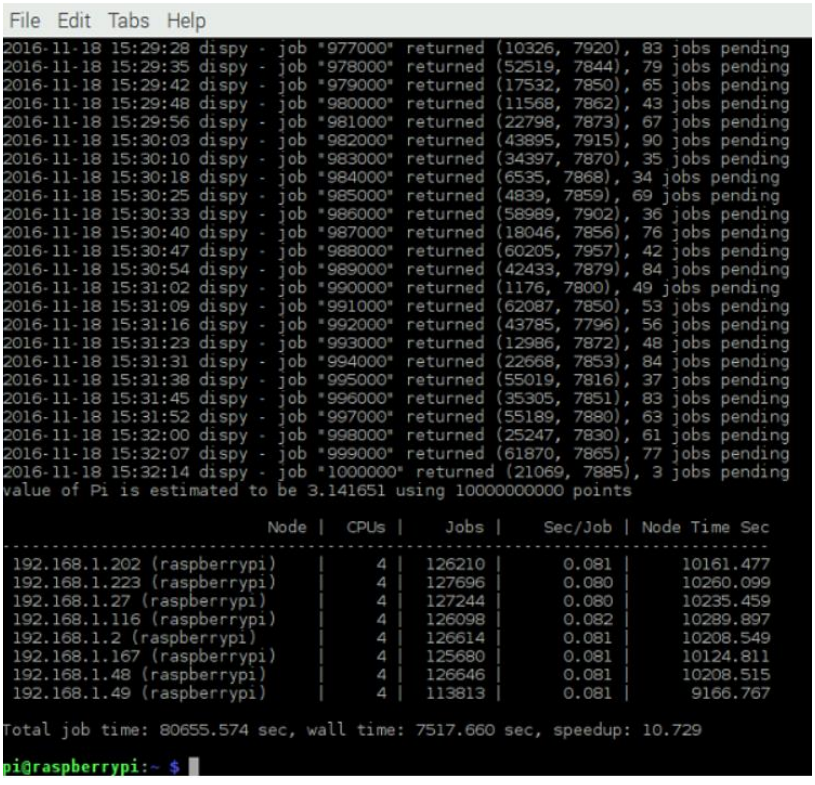


Рисунок 2.36 – Результат виконання «compute_pi_efficient.py»

Тепер, коли перевірено правильність налаштування і функціонування кластеру, можна використовувати сценарій cluster_action.sh для керування ним.

Сценарій cluster_action.sh виконується на клієнті і використовує SSH для адміністрування серверів (тому використовується ssh-keygen для аутентифікації клієнта з серверами). Він покладається на правильні IP-адреси серверів, зазначених у файлі ip_list. Рекомендується видалити файл ip_list під час першого завантаження кластера, щоб список відновився.

На клієнтській машині у терміналі потрібно встановити дозволи для визначеного сценарію функціонування кластера, щоб можна було його запустити, ввівши команду, показану на рисунку 2.37.

```
chmod u+x ./cluster_action.sh
```

Рисунок 2.37 – Запуск скрипта cluster_action.sh

Команду, навдену вище, потрібно виконати лише один раз. Параметрами для сценарію cluster_action є:

– reboot – перезавантажує всі сервери (клієнт і маршрутизатор ігноруються);

– shutdown – вимикає кожен сервер і переводить його в безпечний стан. Якщо сервер не вимикається належним чином, це може призвести до пошкодження micro SD карти та до того, що процесор не завантажиться під час наступного використання.

– date – розподіляє клієнтську дату та час (з точністю до хвилини) кожному серверу. Raspberry Pi 3 не має годинника реального часу, тому спочатку потрібно встановити правильний час на клієнті, наприклад так, як показано на рисунку 2.39.

```
sudo date -s "11 Apr 2022 12:42"
./cluster_action.sh date
```

Рисунок 2.39 – Встановлення дати і часу на клієнті

– unicorn – викликає сценарій start_unicorn.sh на кожному сервері та передає йому ім’я та розташування сценарію Pimoroni Python як параметр. Щоб це працювало, потрібно мати start_unicorn.sh в директорії /home/pi на кожному сервері, як описано раніше.

У результаті проведених маніпуляцій побудовано розподілену комп’ютерну систему у вигляді кластера, що складаєтсья з 8-ми серверів та одного клієнта. Наступний крок полягає у реалізації програмного забезпечення для виконання задач криптоаналізу.

РОЗДІЛ 3 ПРОГРАМНА РЕАЛІЗАЦІЯ ШИФРУВАННЯ ТА ДЕШИФРУВАННЯ ПОВІДОМЛЕНЬ У РОЗПОДІЛЕНІЙ КОМП'ЮТЕРНІЙ СИСТЕМІ

3.1 Шифрування повідомлень на прикладі машини Enigma

У роботі пропонується використати алгоритм і принцип роботи машини Enigma при реалізації функцій криптоаналізу у розподіленій комп'ютерній системі. Мова програмування, яка безпосередньо використана для цього – Python. Перш за все потрібно відкрити середовище IDLE і створити новий файл, зберігши його під назвою encrypt.py. IDLE – це інтегроване середовище розробки Python, яке можна використовувати для написання та виконання коду. Щоб відкрити IDLE потрібно перейти у меню та обрати «Python 3 IDLE». На рисунку 3.1 показано вибір середовища програмування.

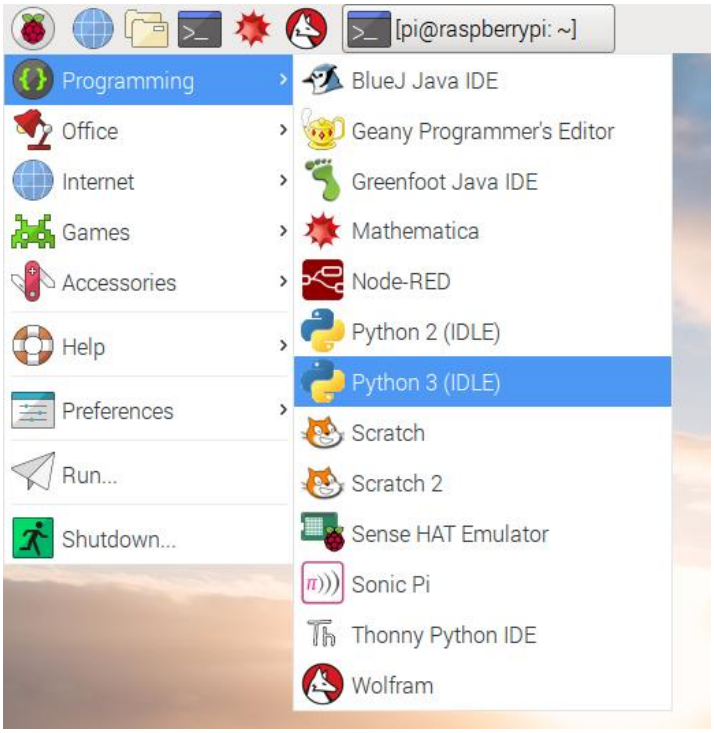


Рисунок 3.1 – Вибір середовища програмування

Щоб створити новий файл у IDLE можна натиснути «File», а потім «New File» у рядку меню IDLE. Це відкриє друге вікно, в якому буде написаний програмний код для шифрування і дешифрування повідомлень машиною Enigma. На рисунку 3.2 показано створення нового файлу у середовищі IDLE.

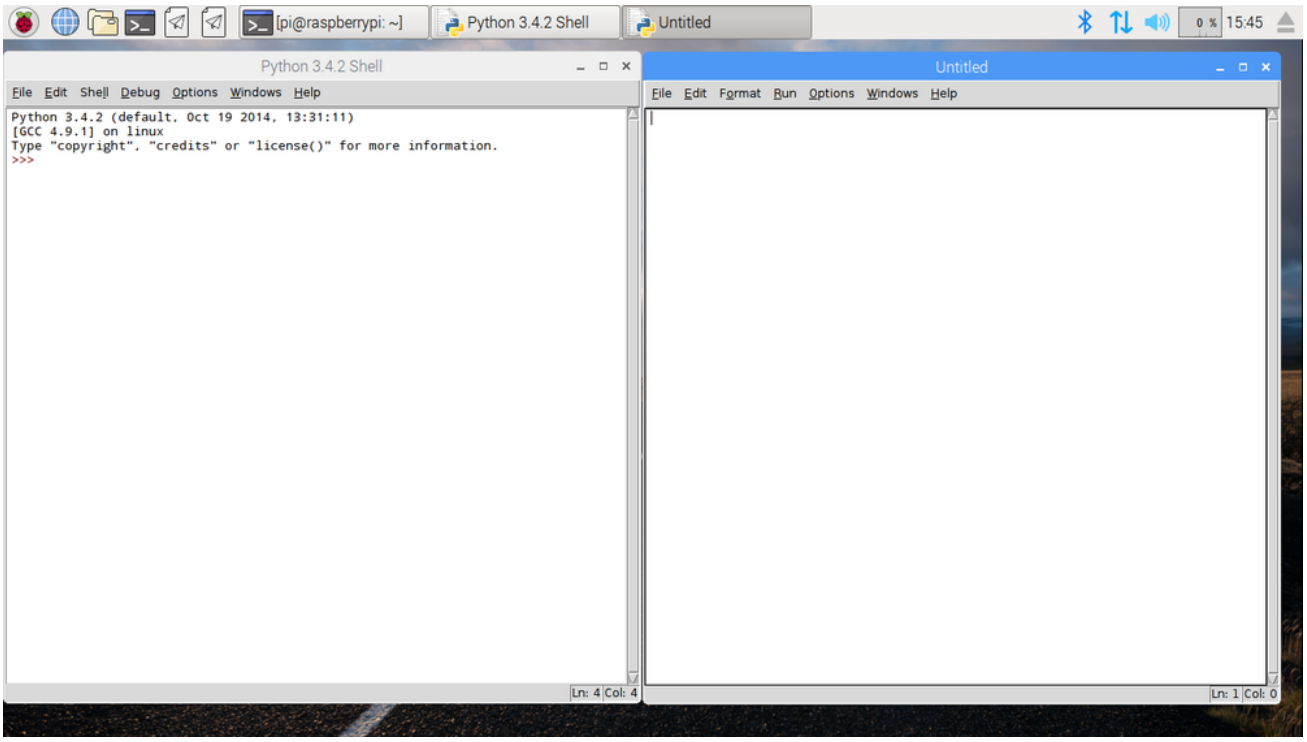


Рисунок 3.2 – Створення нового файлу

Створивши новий файл, необхідно імпортувати потрібні бібліотеки. Спочатку імпортується клас EnigmaMachine з Py-enigma, шляхом додавання стрічки коду, показаної на рисунку 3.3.

```
from enigma.machine import EnigmaMachine
```

Рисунок 3.3 – Імпорт класу EnigmaMachine

В якості вхідних налаштувань Enigma буде використовуватися повідомлення, яке показано на рисунку 3.4.

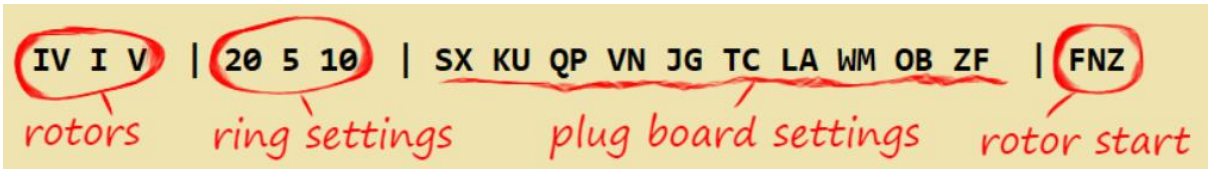


Рисунок 3.4 – Налаштування Enigma

Далі потрібно налаштувати об’єкт EnigmaMachine, використовуючи параметри, наведені на рисунку 3.4. Кожне налаштування має бути рядком і має бути введене точно так, як воно відображається на аркуші налаштувань. Наприклад, ротори будуть встановлені як "IV I V", як показано на рисунку 3.5 у вигляді Python-коду.

```
# Set up the Enigma machine
machine = EnigmaMachine.from_key_sheet(
    rotors='',
    reflector='B',
    ring_settings='',
    plugboard_settings='')
```

Рисунок 3.5 – Налаштування ротора

У даному випадку буде використовуватися рефлектор B для усіх подальших програм. Наступний крок полягає у встановленні початкової позиції ротора (рисунок 3.6).

```
# Set the initial position of the Enigma rotors
machine.set_display('FNZ')
```

Рисунок 3.6 – Встановлення початкової позиції ротора

Наступний крок полягає у виборі трьох випадкових літер для використання їх в якості ключа повідомлення. У даному випадку використано літери «BFR», однак можна обрати будь-які інші. Після цього виконується шифрування повідомлення з відповідним ключем. У результаті (рисунок 3.7) одержується зашифрований ключ, який надсилається разом з повідомленням.

```
# Encrypt the text 'BFR' and store it as msg_key
msg_key = machine.process_text('BFR')
print(msg_key)
```

Рисунок 3.7 – Шифрування ключа повідомлень

Далі потрібно написати рядок коду, щоб скинути початкові позиції ротора на незашифрований ключ повідомлення. В якості прикладу можна написати програмний код для опрацювання незашифрованого тексту «RASPBERRYPI» і одержати зашифрований текст з ключем «BFR». Для цього потрібно реалізувати стрічку коду, як показано на рисунку 3.8.

```
msg_key = machine.process_text('BFR')
```

Рисунок 3.8 – Шифрування повідомлення «RASPBERRYPI» ключем «BFR»

В загальному випадку, повний код щодо шифрування повідомлення, який необхідно реалізувати показано на рисунку 3.9.

```
plaintext = "RASPBERRYPI"
ciphertext = machine.process_text(plaintext)
print(ciphertext)
```

Рисунок 3.9 – Повний код шифрування повідомлення

У випадку використання ключа повідомлення «BFR», отриманий результат зашифрованого тексту має відповідати повідомленню «GON XXLXYFQNZIK».

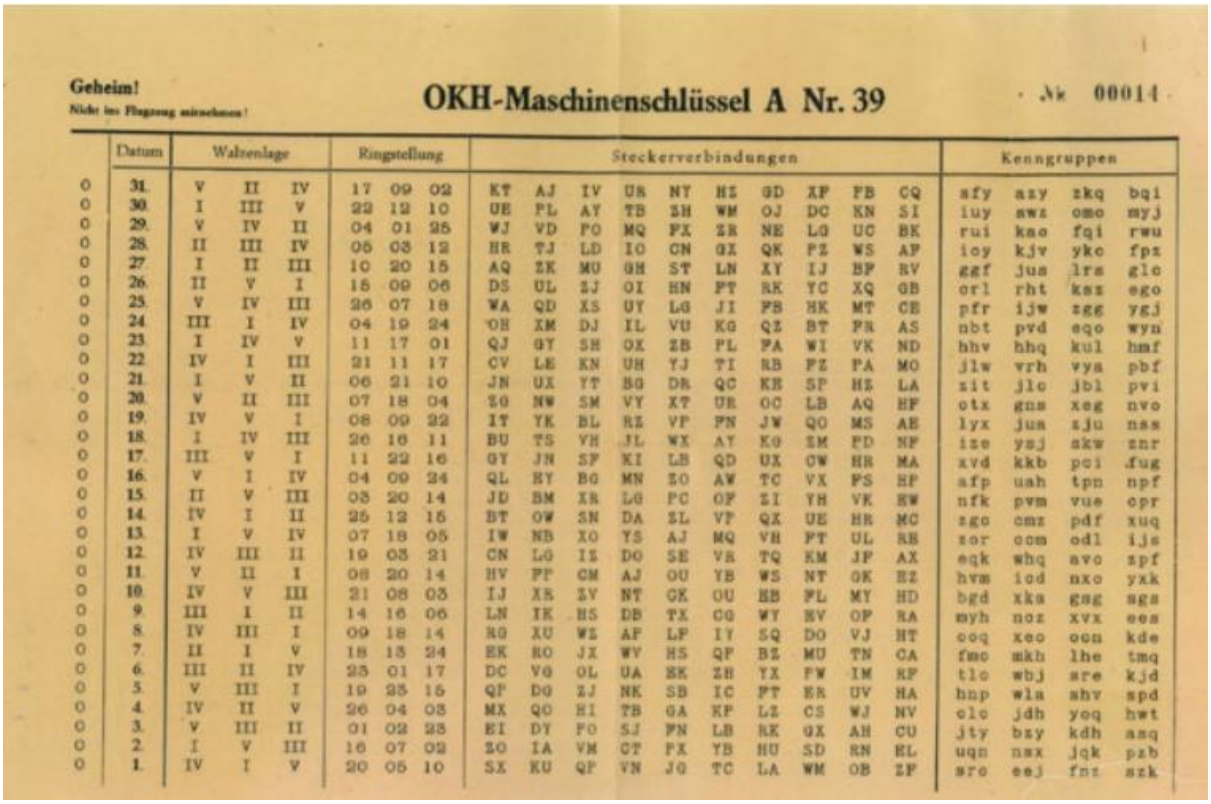
Окрім цього, якщо виникає необхідність, то можна скористатися командною стрічкою для введення цієї команди. У результаті одержується ідентичний результат що й і при написанні сценарію, наведеного на рисунку 3.9.

На рисунку 3.10 продемонстровано виконання шифрування з терміналу.

```
pyenigma.py -r IV I V -i 20 5 10 -p SX KU QP VN JG TC LA WM OB ZF -u B --start B
```

Рисунок 3.10 – Виконання шифрування з командного рядка

Слабким місцем системи Enigma є те, що існує ймовірність шифрування повідомлення самим повідомленням, тобто воно стає відкритим. Якщо зловмисник, який хоче зламати код, може усунути всі можливі рішення для криптоатаки де A розшифровується як A, і так далі. Таблиця ключів для шифрування, яка використовується в Enigma показана на рисунку 3.11.



Geheim!

Nicht ins Flugzeug mitnehmen!

OKH-Maschinenschlüssel A Nr. 39 · Nr 00014 ·

	Datum	Walzenlage			Ringstellung			Steckerverbindungen										Kenngruppen			
0	31.	V	II	IV	17	09	02	KT	AJ	IV	UR	NY	HI	GD	XF	FB	CQ	afy	azy	zkq	bqi
0	30.	I	III	V	22	12	10	UE	PL	AY	TB	ZH	WM	OJ	DC	KN	SI	iuy	swz	omo	myj
0	29.	V	IV	II	04	01	25	WJ	VD	FO	MQ	PX	ZR	NE	LG	UC	BK	rui	kae	fqi	rwu
0	28.	II	III	IV	06	03	12	HR	TJ	LD	IO	CN	GI	QK	PZ	WS	AP	ioy	kjv	yko	fpz
0	27.	I	II	III	10	20	15	AQ	ZK	MU	GH	ST	LN	XY	IJ	BP	RV	ggf	jus	lrs	glo
0	26.	II	V	I	15	09	06	DS	UL	ZJ	OI	HN	PT	RK	YC	XQ	GB	orl	rht	ksz	ego
0	25.	V	IV	III	26	07	18	WA	QD	XS	UY	LG	JI	FB	HK	MT	CE	pfr	ijw	zgg	ygj
0	24.	III	I	IV	04	19	24	OH	XM	DJ	IL	VU	KG	QZ	BT	PR	AS	nbt	pvd	eqo	wyn
0	23.	I	IV	V	11	17	01	QJ	GY	SH	OX	ZB	PL	FA	WI	VK	ND	hhv	hhq	kul	hmf
0	22.	IV	I	III	21	11	17	CV	LE	KN	UH	YJ	TI	RB	PZ	FA	MO	jlw	vrh	vya	pbf
0	21.	I	V	II	06	21	10	JN	UX	YT	BG	DR	QC	KE	SP	HS	LA	zit	jlo	jbl	pvi
0	20.	V	II	III	07	18	04	ZO	NW	SM	VY	XT	UR	OC	LB	AQ	HF	otx	gns	xeg	nvo
0	19.	IV	V	I	08	09	22	IT	YK	BL	RZ	VF	PN	JW	QO	MS	AE	lyx	jus	zju	nas
0	18.	I	IV	III	26	16	11	BU	TS	VH	JL	WX	AY	KG	ZM	FD	NP	ize	ysj	skw	znr
0	17.	III	V	I	11	22	16	GY	JN	SP	KI	LB	QD	UX	CW	HR	MA	xvd	kkb	pci	fug
0	16.	V	I	IV	04	09	24	QL	RY	BG	MN	ZO	AW	TC	VX	PS	HF	afp	uah	tpn	npf
0	15.	II	V	III	03	20	14	JD	BM	XR	LG	FC	OP	ZI	YH	VK	EW	nfk	pvm	vue	opr
0	14.	IV	I	II	25	12	15	BT	OW	SN	DA	ZL	VP	QX	UE	HR	MC	zgo	cmz	pdf	xuq
0	13.	I	V	IV	07	18	05	IW	NB	XO	YS	AJ	MQ	VH	FT	UL	RE	zor	ccm	odl	ijs
0	12.	IV	III	II	19	03	21	CN	LG	IZ	DO	SE	VR	TQ	KM	JP	AX	eqk	whq	avo	zpf
0	11.	V	II	I	08	20	14	HV	PF	CM	AJ	OU	YB	WS	NT	GK	EZ	hvm	iod	nxo	yxk
0	10.	IV	V	III	21	08	03	IJ	XR	ZV	NT	GK	OU	EB	FL	MY	HD	bgd	xka	gsg	sgs
0	9.	III	I	II	14	16	06	LN	IK	HS	DB	TX	CQ	WY	EV	OP	RA	myh	noz	xvx	ees
0	8.	IV	III	I	09	18	14	RG	XU	WZ	AF	LP	IY	SQ	DO	VJ	HT	coq	xeo	oon	kde
0	7.	II	I	V	18	15	24	EK	RO	JX	WV	HS	QP	BZ	MU	TN	CA	fmc	mkh	lhe	tmq
0	6.	III	II	IV	23	01	17	DC	VG	OL	UA	EK	ZR	YX	FW	IM	KP	tlo	wbj	sre	kjd
0	5.	V	III	I	19	23	15	QF	DG	ZJ	NK	SB	IC	PT	ER	UV	HA	hnp	wla	shv	spd
0	4.	IV	II	V	26	04	03	MX	QO	HI	TB	GA	KP	LZ	CS	WJ	NV	elc	jdh	yoq	hwt
0	3.	V	III	II	01	02	25	EI	DY	FO	SJ	PN	LB	RK	QX	AH	CU	jty	bzy	kdh	asq
0	2.	I	V	III	16	07	02	ZO	IA	VM	CT	FX	YB	HU	SD	RN	EL	uqn	nsx	jqk	pzb
0	1.	IV	I	V	20	05	10	SX	KU	QP	VN	JG	TC	LA	WM	OB	ZF	sro	eej	fns	szk

Рисунок 3.11 – Таблиця ключів для шифрування повідомлень Enigma

Таким чином, у даному підрозділі розглянути принципи шифрування та функціонування системи Enigma з використанням мови програмування Python, що в подальшому буде корисно для реалізації алгоритмів шифрування на організованому у розділі 2 кластері.

3.2 Дешифрування повідомлень з використанням Enigma

Дешифрування є процесом оберненим до шифрування. Основне завдання, яке необхідно при цьому розв’язати, полягає у визначенні ключа і перетворенні повідомлення в оригінальний текст. Для прикладу, маємо повідомлення, яке показано на рисунку 3.12.

PWE YJPYITREDSYUPIU

Рисунок 3.12 – Вхідне зашифроване повідомлення

Потрібно реалізувати програмний код за допомогою Py-enigma для моделювання роботи машини Enigma щодо розшифрування повідомлення.

Для цього у середовищі IDLE створюємо новий файл і зберігаємо його як decrypt.py. Аналізуючи рисунок 3.11 з налаштуваннями Enigma, можна встановити, що шифрувальна машина мала такі налаштування на момент надсилання повідомлення, як показано на рисунку 3.13.

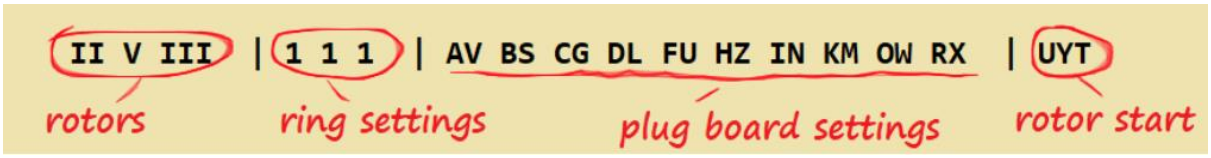


Рисунок 3.13 – Налаштування Enigma перед відправленням повідомлень

Далі потрібно імпортувати клас EnigmaMachine та налаштувати машину зі встановленими параметрами. Як і минулого разу також використовується рефлектор B. Також необхідно додати програмний код, щоб встановити початкові позиції роторів на U, Y і T, щоб забезпечити відповідність машині-відправнику. Ключем до повідомлення буде комбінація «PWE».

Перед відправкою ключ був зашифрований, щоб запобігти його несанкціонованому читанню. Спочатку потрібно використати машину Enigma,

щоб відновити фактичний ключ повідомлення, розшифрувавши «PWE», використовуючи початкові позиції ротора налаштувань: U, Y і T.

Для цього потрібно додати код (рисунок 3.14), щоб розшифрувати ключ, і запустити програму для відображення результату.

```
# Decrypt the text 'PWE' and store it as msg_key
msg_key = machine.process_text('PWE')
print(msg_key)
```

Рисунок 3.14 – Розшифрування ключа «PWE»

Наступний крок полягає у безпосередньому декодуванні зашифрованого тексту «YJPYITREDSYUPIU». Код дешифрування дуже подібний до того, який використовувався при дешифруванні ключа повідомлення. Однак в даному випадку потрібно ще створити змінну для зберігання результату декодування, а також використати Enigma для опрацювання зашифрованого повідомлення і виводу одержаного результату на екран. Скрипт декодування зашифрованого повідомлення показано на рисунку 3.15.

```
ciphertext = 'YJPYITREDSYUPIU'
plaintext = machine.process_text(ciphertext)

print(plaintext)
```

Рисунок 3.15 – Скрипт дешифрування повідомлення

Якщо процес дешифрування відбувся коректно і не виникало жодних помилок, то результат декодованого тексту матиме такий вигляд: «THISXISXWORKING».

Під час Другої світової війни криптографи Блетчлі-парку наполегливо працювали, щоб спробувати зламати шифр Enigma вручну для розшифрування перехоплених німецьких шифрів. Далі реалізуємо алгоритм brute-force для

декодування зашифрованого тексту машиною Enigma за допомогою Raspberry Pi.

Brute force не є оптимальним алгоритм декодування, а представляє собою пошук усіх можливих налаштувань машини, щоб спробувати знайти, який із них був використаний. Однак на даному етапі будемо вважати, що налаштування є відомими. Нехай перехоплене повідомлення матиме вигляд, як показано на рисунку 3.16.

Рисунок 3.16 – Зашифроване повідомлення

На додаток до зашифрованого тексту, будемо використовувати текст шифру, який є припущенням, яким може бути цей текст. Це може здатися шахрайством, але насправді використовується слабкість системи Enigma, яка використовувалася під час Другої світової війни: деякі тексти повідомлення були передбачуваними, особливо його початок. Наприклад, повідомлення про погоду були хорошим джерелом визначення початкових позицій ротора, оскільки вони часто містили слово «WETTER», німецьке слово «погода». пропонується використовувати цей зашифрований текст, щоб забезпечити запуск алгоритму brute force (рисунок 3.17).

```
ciphertext = "YJPYITREDSYUPIU"
cribtext = "THISXISXWORKING"
```

Рисунок 3.17 – Початкові налаштування алгоритму brute force

У випадку, коли алгоритм знайшов правильний вибір ротора та початкові позиції, то шифрований текст буде розшифровано з текстом ключа.

Для реалізації алгоритму brute force потрібно створити новий файл Python і зберегти його як bruteforce_standalone.py. Після цього додаються змінні, що

містять зашифрований текст і текст «шпаргалки», як рядки. Далі потрібно представити вибір трьох з п’яти роторів у коді Python. Окрім цього, можна було б написати код для створення комбінацій стану роторів, але оскільки їх не так багато, то їх можна визначити вручну. Список усіх можливих комбінацій роторів показано на рисунку 3.18 у вигляді колекції стірчок.

```
rotors = [ "I II III", "I II IV", "I II V", "I III II",
"I III IV", "I III V", "I IV II", "I IV III",
"I IV V", "I V II", "I V III", "I V IV",
"II I III", "II I IV", "II I V", "II III I",
"II III IV", "II III V", "II IV I", "II IV III",
"II IV V", "II V I", "II V III", "II V IV",
"III I II", "III I IV", "III I V", "III II I",
"III II IV", "III II V", "III IV I", "III IV II",
"III IV V", "IV I II", "IV I III", "IV I V",
"IV II I", "IV II III", "IV I V", "IV II I",
"IV II III", "IV II V", "IV III I", "IV III II",
"IV III V", "IV V I", "IV V II", "IV V III",
"V I II", "V I III", "V I IV", "V II I",
"V II III", "V II IV", "V III I", "V III II",
"V III IV", "V IV I", "V IV II", "V IV III" ]
```

Рисунок 3.18 – Можливі комбінації роторів

Стратегія полягатиме в тому, щоб вибирати кожен набір роторів у списку роторів по черзі та перевіряти, чи декодування зашифрованого тексту за допомогою цієї комбінації роторів отримує текст ключа. Однак це не так просто, як перевірити кожен можливий вибір роторів. Усередині тіла функції також потрібно буде виконувати пошук усіх можливих початкових позицій ротора для кожної комбінації роторів.

На даний момент припустимо, що налаштування контактного кільця «1 1 1» і параметри ключа «AV BS CG DL FU HZ IN KM OW RX». Далі потрібно створити функцію з назвою «find_rotor_start()», яка приймає три аргументи: вибір ротора, зашифрований текст і текст ключа, а всередині функції додаємо код для імпорту класу Enigma Machine, як показано на рисунку 3.19.

```
from enigma.machine import EnigmaMachine
```

Рисунок 3.19 – Імпорт класу EnigmaMachine

Імпорт модуля Py-Enigma всередину функції зроблено неспроста: це дозволяє повторно використовувати цей код пізніше на побудудованому кластером. Це дозволить масово і паралельно запускати сценарій таким чином, що за набагато менший час можна одержати результат, ніж на одному процесорі. Наступний крок полягає у написанні коду всередині функції для перевірки всіх можливих початкових позицій ротора для конкретно обраного ротора.

Послідовність кроків, за яким реалізується алгоритм у функції наступний:

– імпорт класу EngineMachine;

– створення стрічки, що буде містити алфавіт і за допомогою циклу дозволить перебір літер;

– оголошення об’єкту EnigmaMachine з використанням дефлектора B і параметрами, які визначені раніше;

– генерування усіх можливих комбінацій роторів, наприклад, якщо ротори починаються з A то перша позиція для тестування буде AAA. Наступна ітерація може бути AAB і так далі, поки ротор 3 не досягне кінця алфавіту. Після цього виконується зміщення другого ротора на одну позицію, а третій ротор скидають до початкового стану. Процедура повторюється знову, і як результат одержують комбінації ABA, після того ABB і т.д.

– для кожної позиції ротора у процесі дешифрування виконується кодування тексту і перевіряється чи результат збігається з вхідним зашифрованим текстом. Якщо тексти співпадають, то повертається значення ротора і починається розкодування самого повідомлення.

Реалізація описаного вище алгоритму засобами мови програмування Python представлена у вигляді функції find_rotor_start() та показана на рисунку 3.20.

```python
def find_rotor_start( rotor_choice, ciphertext, cribtext ):
    from enigma.machine import EnigmaMachine
    alphabet = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"

    machine = EnigmaMachine.from_key_sheet(
       rotors=rotor_choice,
       reflector='B',
       ring_settings='1 1 1',
       plugboard_settings='AV BS CG DL FU HZ IN KM OW RX')

    # Do a search over all possible rotor starting positions
    for rotor1 in alphabet:          # Search for rotor 1 start position
        for rotor2 in alphabet:      # Search for rotor 2 start position
            for rotor3 in alphabet:  # Search for rotor 3 start position

                # Generate a possible rotor start position
                start_pos = rotor1 + rotor2 + rotor3

                # Set the starting position
                machine.set_display(start_pos)

                # Attempt to decrypt the plaintext
                plaintext = machine.process_text(ciphertext)
                print( plaintext )

                # Check if decrypted version is the same as the crib text
                if plaintext == cribtext:
                    print("Valid settings found!")
                    return rotor_choice, start_pos

    # If we didn't manage to successfully decrypt the message
    return rotor_choice, "Cannot find settings"
```

Рисунок 3.20 – Реалізація функції find_rotor_start()

У більшості випадків функція не зможе відповідати текстам шифру, оскільки вибір ротора буде неправильним. В одному випадку тексти шифру та закодованого тексту будуть збігатися, тому що знайдено правильне налаштування машини.

В основній частині програми потрібно написати цикл, щоб викликати функцію один раз для кожної можливої комбінації вибору роторів у списку роторів.

Для кожного виклику функції варто виводи результати її виконання. Якщо повертається початкова позиція, яка не є повідомленням «Не вдається знайти налаштування», то необхідно перервати цикл — потрібні налаштування знайдено (рисунок 3.21).

```
for rotor_setting in rotors:
    rotor_choice, start_pos = find_rotor_start( rotor_setting, ciphertext, cribte
    print(rotor_choice + " " + start_pos )
    if start_pos != "Cannot find settings":
        break
```

Рисунок 3.21 – Фрагмент програми для знаходження комбінації ротора

Далі потрібно зберегти і запустити програму. Її виконання займе досить багато часу, але під час цього можна побачити результати для кожного вибору ротора. Після того, як алгоритм brute force знайшов ротори та початкову позицію необхідно підключити їх до програми decrypt.py, яка показана вище, і розшифрувати повне секретне повідомлення!

У даному випадку не було програмної реалізації щодо налаштування контактних кілець роторів. Контактне кільце зміщує проводку всередині ротора — зміщує її на один, і A з'єднується з місцем, де до цього було B, B з'єднується з місцем, де було C раніше, C з'єднується з місцем, де знаходилось D, і так далі.

Для того, щоб реалізувати налаштування кільця ротора, потрібно буде змінити функцію find_rotor_start() так, щоб вона запускалася багаторазово для кожного параметра ітерації ротора.

Кожен ротор машини Enigma може мати 26 положень контактного кільця: A до A (без зсуву), A до B (зсув на 1), ..., A до Z (зсув на 26). У машині Enigma з трьома роторами це означає, що доведеться виконати пошук положення

контактного кільця 26 разів для першого ротора, і все це 26 разів для другого ротора, і все це 26 разів для третього ротора.

Отже, програма шифрування методом brute force займе у 26 × 26 × 26 = 17576 разів більше часу. Це дуже довгий час, але можна розбити цю проблему на багато частин і запустити їх паралельно за допомогою побудованого кластеру.

3.3 Реалізація криптоаналізу на розподіленій комп'ютерній системі на основі Raspberry PI

Для проведення криптоаналізу (симуляції роботи машини Enigma) потрібні ресурси кластеру, до складу якого входять 8 Raspberry PI, що є серверами та один, який виконує функцію клієнта. Перш ніж почати реалізацію обчислень, необхідно встановити py-enigma на клієнтську машину та всі сервери, що знаходяться у кластері.

Перший крок полягає у встановленні py-enigma на клієнті. Для цього потрібно увімкнути клієнт-Raspberry PI, підключитися до мережі Інтернет і ввести команду, яка показана на рисунку 3.22.

```
sudo pip3 install py-enigma
```

Рисунок 3.22 – Встановлення py-enigma на клієнт Raspberry PI

Після встановлення ПЗ необхідно від'єднатися від маршрутизатора з доступом до Інтернет, приєднатися до роутера кластеру і виконати команду для відкриття файлу конфігурації wpa_supplicant.conf (рисунок 3.23).

```
sudo nano /etc/wpa_supplicant/wpa_supplicant.conf
```

Рисунок 3.23 – Відкриття файлу конфігурації

Щоб уникнути підключення клієнта до неправильної мережі потрібно видалити усі записи в wpa_supplicant.conf, які призначені для мереж WiFi, відмінних від тієї, що призначена для роботи розподіленої комп'ютерної системи. Далі необхідно натиснути Ctrl + O, щоб зберегти зміни, і Ctrl + X, щоб вийти з текстового редактора. Таку ж процедуру повторюють для кожного з серверів у кластері.

Щоб виконати вичерпний пошук усіх налаштувань контактних кілець ротора, потрібно виконати багато завдань на кластері за допомогою dispy, який встановлено під час його створення. Код для розподіленої комп'ютерної системи з використанням dispy дуже схожий на код, який було створено для автономного процесора. Використання пам'яті клієнтської машини кластера буде досить високим, тому потрібно буде запускати програму з одним налаштуванням кільця за раз.

Починати варто з коду, який написаний для реалізації алгоритму на одній машині. Для цього потрібно з вихідного коду видалити все, крім списку перестановок ротора та функції find_rotor_start() та імпортувати бібліотеки dispy і socket, як показано на рисунку 3.24.

```
import dispy, socket
```

Рисунок 3.24 – Імпорт необхідних бібліотек

Далі необхідно змінити функцію find_rotor_start() так, щоб вона тепер приймала додатковий параметр: ring_choice. Це буде рядок, що містить три числа, розділені пробілами, наприклад, '1 1 1'. Усередині функції потрібно встановити ring_choice в об'єкті EnigmaMachine як ring_choice, який було передано у функцію.

Наступний крок полягає у знаходженні двох місць, куди функція повертає значення (коли збіг знайдено, або коли всі можливості вичерпано, а збігу не знайдено). На додаток до повернення вибору ротора та початкової позиції,

потрібно додати код, щоб повернути ring_choice як друге повернуто значення, щоб функція повернула всього три значення.

В основній частині програми потрібно додати фрагмент коду, щоб дозволити користувачеві вводити шифрований текст, текст ключа та налаштування сповіщення. Це можна реалізувати за допомогою функції input() або зібравши аргументи з командного рядка за допомогою модуля argparse.

Створення об’єкту кластера показано нижче на рисунку 3.25.

```
s = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
s.connect(("8.8.8.8", 80)) # doesn't matter if 8.8.8.8 can't be reached
cluster = dispy.JobCluster(find_rotor_start, ip_addr=s.getsockname()[0], nodes='

jobs = []
id = 1
```

Рисунок 3.25 – Створення об’єкту кластера

Якщо мережа кластеру використовує інший діапазон IP-адрес за замовчуванням, то потрібно буде змінити частину коду з вузлами, щоб відобразити це.

Після цього виконується надсилання завдання find_rotor_start() у кластер з використанням методу (рисунок 3.26), подібного до циклу реалізації алгоритму brute force на одній машині.

```
# Submit the jobs for this ring choice
for rotor_choice in rotors:
    job = cluster.submit( rotor_choice, ciphertext, cribtext, ring_choice )
    job.id = id # Associate an ID to the job
    jobs.append(job)
    id += 1   # Next job
```

Рисунок 3.26 – Надсилання завдання у кластер

Далі потрібно дочекатися завершення завдань, перш ніж зберігати результати, які повернув кластер (рисунок 3.27).

```
print( "Waiting..." )
cluster.wait()
print( "Collecting job results" )
```

Рисунок 3.27 – Організація виводу результатів роботи кластера

Останній крок полягає у відборі результатів, щоб побачити, чи не повернуло жодне із завдань find_rotor_start() рядок «Не вдається знайти налаштування», і в цьому випадку повернений рядок повинен бути дійсною початковою позицією ротора (рисунок 3.28, рисунок 3.29).

```
# Collect and check through the jobs for this ring setting
found = False
for job in jobs:
    # Wait for job to finish and return results
    rotor_setting, ring_setting, start_pos = job()

    # If a start position was found
    if start_pos != "Cannot find settings":
        found = True
        print( "Rotors %s, ring %s, message key was %s, using crib %s" % (rotor_
```

Рисунок 3.28 – Перевірка результатів роботи кластера

```
if found == False:
    print( 'Attack unsuccessful' )

cluster.print_status()
cluster.close()
```

Рисунок 3.29 – Закриття кластеру

Виконання програми для розшифрування тексту «FKFPQZYVON» з ключем «CHELTENHAM» і налаштуваннями позицій '1 1 1' займе близько 30 секунд. Приклад роботи програми з аргументами, переданими з командного рядка показана на рисунку 3.30.

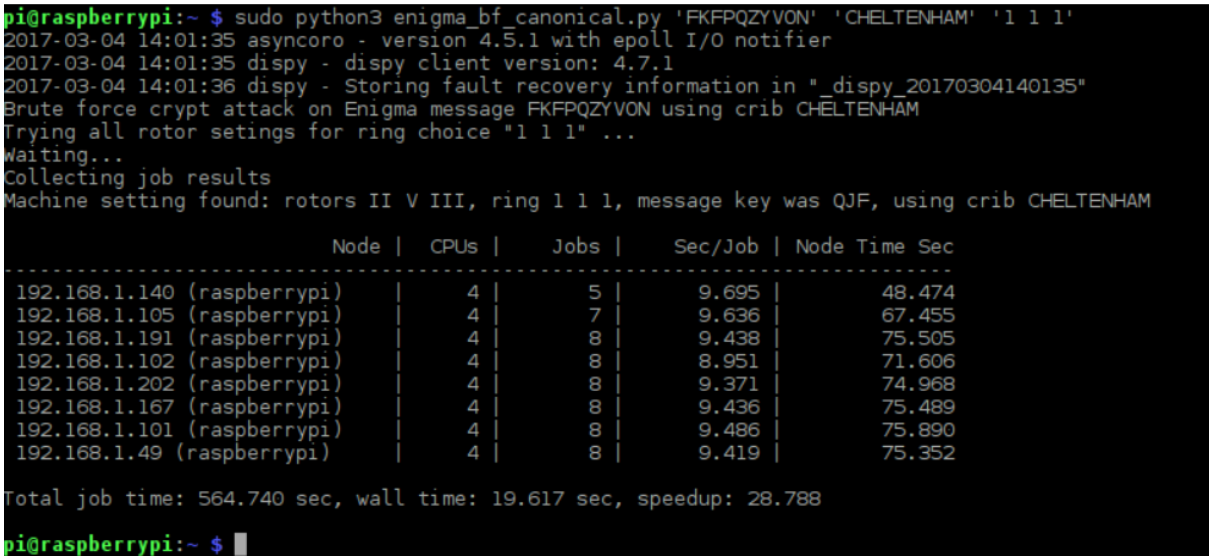


Рисунок 3.30 – Результат виконання програми з дешифрування повідомлення на кластері

Для однакового вибору ротора іноді можна знайти кілька дійсних налаштувань машини з різними параметрами кільця та початковими положеннями ротора. Наприклад, можна знайти початкові позиції «ABC» з налаштуваннями кільця «1 1 1», а також «ABD» з «1 1 2», як два дійсні результати. Це не помилка: обидва параметри машини дійсні.

Насправді існує кілька дійсних налаштувань машини, оскільки переміщення контактного кільця ротора створює кілька еквівалентних рішень алгоритму. Це не ще один приклад помилки в техніці шифрування Enigma, але показує, як змінилася природа кіберзагрози за більш ніж 75 років після Другої світової війни. Спочатку ризик становили люди, які розшифровували букву за буквою. Зміна контактного кільця ротора означала, що ротори просувалися в несподіваних положеннях, створюючи розрив кожні 26, 26×26 та 26×26×26 символів, що означає, що зловмиснику доведеться продовжувати починати знову.

З даною криптоатакою на базі Raspberry Pi з використанням простого пошуку із застосуванням brute force у всьому діапазоні можливих налаштувань машини виявлено, що налаштування контактного кільця ротора створює кілька можливих рішень. Тому ця функція є слабкою стороною, оскільки для пошуку правильного рішення потрібно менше ресурсів.

Якщо запускати програмний код кілька разів із все меншою і меншою кількістю символів шифрованого тексту, то виявиться, що для отримання правильних налаштувань машини достатньо лише чотирьох символів тексту шифру (а також кількох неправильних рішень). З менш ніж чотирма символами існує багато неоднозначності, що проявляється у складності знайти правильне рішення серед усіх неправильних.

Таким чином, у кваліфікаційній роботі реалізовано програмне забезпечення, що емулює роботу машини Enigma для шифрування і декодування повідомлень за визначеним алгоритмом і дозволяє проводити криптоаналіз із застосуванням brute force.

РОЗДІЛ 4 БЕЗПЕКА ЖИТТЄДІЯЛЬНОСТІ, ОСНОВИ ОХОРОНИ ПРАЦІ

4.1 Вплив шуму на організм людини та розробка заходів щодо його зниженню до допустимих величин

Шум – це сукупність звуків різноманітної частоти та інтенсивності, що виникають у результаті коливального руху частинок у пружних середовищах (твердих, рідких, газоподібних). Шумом також вважають будь-який небажаний для людини звук [17].

Важливою характеристикою шуму є його частотний склад. Якщо в складі шуму переважають звуки з частотою коливань до 400 Гц, такий шум називається низькочастотним, якщо переважають звуки з частотою 400 – 1000 Гц – середньочастотним, якщо понад 1000 Гц – високочастотним [18].

Низькочастотний шум інтенсивністю до 100 дБ не викликає відчутної несприятливої дії на орган слуху; для середньочастотного шуму ця норма складає 85 – 90 дБ; для високочастотного – 75 – 85 дБ. Несприятливі суб'єктивні відчуття і вплив на організм людини зумовлює високочастотний шум [18].

Шум підступний, його шкідливий вплив на організм відбувається незримо, непомітно. Організм людини проти шуму практично беззахисний.

Вплив шуму на організм умовно поділяють на:

– специфічний, що спричиняє зміни в органі слуху;

– неспецифічний – з боку інших органів і систем.

Основну увагу приділяють стану органа слуху, тому що слуховий аналізатор першим сприймає звукові коливання і потерпає від впливу шуму на організм.

Дія шуму на організм людини пов'язана головним чином із застосуванням нового, високопродуктивного устаткування, з механізацією або автоматизацією

трудових процесів: переходом на великі швидкості при експлуатації різних верстатів і агрегатів [18].

Джерелами шуму можуть бути двигуни, насоси, компресори, пневматичні та електричні інструменти, молоти, дробарки, верстати, центрифуги та інше обладнання, що має рухомі деталі.

Крім того, за останні роки, у зв'язку із значним розвитком міського транспорту, зросла інтенсивність шуму і в побуті.

Короткочасний, навіть одноразовий вплив шуму високої інтенсивності може спричинити повну загибель спірального органа або розрив барабанної перетинки, що супроводжується почуттям закладеності та різким болем у вухах. Наслідком баротравми нерідко буває повна втрата слуху [18].

У виробничих умовах такі травми спостерігаються надзвичайно рідко, здебільшого під час аварій чи вибухів.

Основною ознакою впливу шуму є зниження слуху по типу кохлеарного невриту. Професійне зниження слуху зазвичай буває двостороннім. Стійкі зміни слуху, як правило, розвиваються повільно, нерідко їм передує адаптація до шуму, яка характеризується нестійким зниженням слуху, що виникає безпосередньо після його впливу і зникає після припинення його дії [19].

Початкові прояви професійної приглухуватості найчастіше зустрічаються у осіб зі стажем роботи в умовах шуму близько 5 років. При високих рівнях шуму слухова чутливість падає вже через 1 – 2 роки, при середніх – виявляється набагато пізніше, через 5 – 10 років, тобто зниження слуху відбувається повільно, хвороба розвивається поступово.

У працюючих в умовах шуму основними скаргами є: зниження слуху, головний біль тупого характеру, відчуття важкості і шуму в голові, що виникають до кінця робочої зміни або після роботи, запаморочення при зміні положення тіла, підвищена дратівливість, швидка стомлюваність, зниження працездатності, уваги, підвищена пітливість, порушення ритму сну (сонливість вдень, тривожний сон у нічний час). Можуть спостерігатися неприємні відчуття

в області серця у вигляді поколювань, серцебиття. Відзначається виражена нестійкість пульсу і артеріального тиску, особливо в період перебування в умовах шуму.

Ефективний захист працюючих від несприятливого впливу шуму вимагає здійснення комплексу організаційних, технічних і медичних заходів. Особливо важливо заздалегідь приймати відповідні заходи захисту від шуму.

З метою підвищення ефективності боротьби з шумом введено обов'язковий гігієнічний контроль об'єктів, що генерують шум, реєстрація фізичних факторів, що роблять шкідливий вплив на навколишнє середовище і негативно впливають на здоров'я людей. Ефективним шляхом вирішення проблеми боротьби з шумом є зниження його рівня в самому джерелі за рахунок зміни технології і конструкції машин. До заходів цього типу відносяться заміна гучних процесів безшумними, ударних – безударними, наприклад заміна клепки пайкою, кування і штампування – обробкою тиском, застосування віброізоляції, глушників, звукоізолюючих кожухів та інші. У деяких випадках зниження рівня шуму досягається застосуванням звукопоглинальних пористих матеріалів, покритих перфорованими листами алюмінію, пластмас.

При необхідності підвищення коефіцієнта звукопоглинання в області високих частот звукоізолюючі шари покривають захисною оболонкою з дрібною і частою перфорацією, застосовують також штучні звукопоглиначі у вигляді конусів, кубів, закріплених над обладнанням, що є джерелом підвищеного шуму. У тих випадках, коли технічні засоби не забезпечують досягнення вимог чинних нормативів, необхідно обмеження тривалості впливу шуму та застосування засобів індивідуального захисту органу слуху. Їх використовують тоді, коли технічні засоби боротьби з шумом не забезпечують зниження його до безпечних меж. Засоби індивідуального захисту органів слуху поділяють на три типи: вкладиші, навушники і шоломи.

Важливе значення у попередженні розвитку шумової патології мають попередні (під час прийняття на роботу) і періодичні (протягом трудової діяльності) медичні огляди. Згідно з наказом Міністерства охорони здоров'я України від 21.05.2007 № 246 «Про затвердження Порядку проведення медичних

оглядів працівників певних категорій» таким оглядам підлягають особи, які працюють на виробництвах, де шум перевищує гранично допустимий рівень.

Медичними протипоказаннями до допуску на роботу, пов’язану з впливом інтенсивного шуму, крім загальних медичних протипоказань є наступні захворювання:

– стійке зниження слуху, хоча б на одне вухо, будь-якого походження;

– отосклероз і інші хронічні захворювання вуха з несприятливим прогнозом;

– порушення функції вестибулярного апарата, у тому числі хвороба Меньєра;

– виражена вегетативно-судинна дистонія;

– гіпертонічна хвороба (всі стадії).

Кратність проведення періодичних медичних оглядів встановлюється в залежності від інтенсивності шуму. Огляди проводяться за участю отоларинголога, невропатолога і терапевта.

4.2 Вплив діяльності людини на довкілля

В умовах науково-технічного прогресу значно ускладнились взаємовідносини суспільства з природою. Людина отримала можливість впливати на хід природних процесів, підкорила сили природи, почала опановувати майже всі доступні відновні і невідновні природні ресурси, але разом з тим забруднювати і руйнувати довкілля.

За оцінкою Всесвітньої організації охорони здоров'я (ВООЗ), із більш ніж 6 млн. відомих хімічних сполук практично використовується до 500 тис. сполук; із них біля 40 тис. мають шкідливі для людини властивості, а 12 тис. є токсичними.

До кінця XX століття початку XXI століття забруднення навколишнього середовища відходами, викидами, стічними водами всіх видів промислового виробництва, сільського господарства, комунального господарства міст набуло глобального характеру і поставило людство на грань екологічної катастрофи.

Втручання людини у природні процеси різко зростає і може спричиняти зміну режиму ґрунтових і підземних вод у цілих регіонах, поверхневого стоку, структури ґрунтів, інтенсифікацію ерозійних процесів, активізацію геохімічних та хімічних процесів у атмосфері, гідросфері та літосфері, зміни мікроклімату тощо.

Сучасна діяльність, наприклад, будівництво гідротехнічних споруд, шахт, рудників, доріг, свердловин, водойм, дамб, деформація суші ядерними вибухами, будівництво гігантських міст, обводнення і озеленення пустель, та інші повсякденні аспекти діяльності людини, вже викликали значні видимі і приховані зміни довкілля.

В історичному плані виділяють декілька етапів зміни біосфери людством, які увінчались екологічними кризами та революціями, а саме:

– вплив людства на біосферу як звичайного біологічного виду;

– надінтенсивне полювання без змін екосистем у період становлення людства;

– зміни екосистем внаслідок процесів, що відбуваються природнім шляхом: випасання, посилення росту трав шляхом випалювання тощо;

– інтенсифікація впливу на природу шляхом розорювання ґрунтів та вирубування лісів;

– глобальні зміни всіх екологічних компонентів біосфери в цілому.

Вплив людини на біосферу зводиться до чотирьох головних форм [18]:

1) зміна структури земної поверхні (розорювання степів, вирубування лісів, меліорація, створення штучних водойм та інші зміни режиму поверхневих вод тощо);

2) зміна складу біосфери, кругообігу і балансу тих речовин, які її складають (добування корисних копалин, створення відвалів, викиди різних речовин у атмосферу та водойми);

3) зміна енергетичного, зокрема теплового, балансу окремих регіонів земної кулі і всієї планети;

4) зміни, які вносяться у біоту (сукупність живих організмів) внаслідок знищення деяких видів, руйнування їх природних місць існування, створення

нових порід тварин та сортів рослин, переміщення їх на нові місця існування тощо.

Під забрудненням навколишнього середовища розуміють надходження в біосферу будь-яких твердих, рідких і газоподібних речовин або видів енергії (теплоти, звуку, радіоактивності і т.п.) у кількостях, що шкідливо впливають на людину, тварин і рослини як безпосередньо, так і непрямим шляхом.

Безпосередньо об'єктами забруднення (акцепторами забруднених речовин) є основні компоненти екотопу (місце існування біотичного угруповання) [19]:

– атмосфера,

– вода,

– ґрунт.

Опосередкованими об'єктами забруднення (жертвами забруднення) є складові біогеоценозу:

– рослини,

– тварини,

– гриби,

– мікроорганізми.

Втручання людини в природні процеси в біосфері, котре викликає небажані для екосистем антропогенні зміни, можна згрупувати за наступними видами забруднень:

– інгредієнтне забруднення - забруднення сукупністю речовин, кількісно або якісно ворожих природним біогеоценозам (інгредієнт - складова частина складної сполуки або суміші);

– параметричне забруднення пов'язане зі зміною якісних параметрів навколишнього середовища (параметр навколишнього середовища - одна з його властивостей, наприклад, рівень шуму, радіації, освітленості);

– біоценотичне забруднення полягає у впливі на склад та структуру популяції живих організмів;

– стаціально-деструкційне забруднення (стація- місце існування популяції, деструкція - руйнування) викликає зміну ландшафтів та екологічних систем в процесі природокористування.

Фахівці по різному класифікують забруднення природного середовища, в залежності від того, який принцип беруть за основу класифікації, зокрема - за типом походження, за часом взаємодії з довкіллям, за способом впливу.

За просторовим поширенням (розміру охоплюючих територій) забруднення, згідно [19], поділяють на:

– локальні забруднення характерні для міст, значних промислових підприємств, районів видобутку тих або інших корисних копалин, значних тваринницьких комплексів.

– регіональні забруднення охоплюють значні території й акваторії, що підлягають впливу значних промислових районів.

– глобальні забруднення частіше всього викликаються атмосферними викидами, поширюються на великі відстані від місця свого виникнення і створюють несприятливий вплив на крупні регіони, а іноді і на всю планету.

За силою та характером дії на навколишнє середовище забруднення бувають:

– фонові;

– імпактні (від англ. імпект - удар; синонім - залпові);

– постійні (перманентні);

– катастрофічні.

За джерелами виникнення забруднення поділяють на:

– промислові (наприклад, SO2);

– транспортні (наприклад, альдегіди вихлопів автотранспорту);

– сільськогосподарські (наприклад, пестициди);

– побутові (наприклад, синтетичні мийних засобів).

За типом походження розрізняють:

– фізичні забруднення – це зміни теплових, електричних, радіаційних, світлових полів у природному середовищі, шуми, вібрації, гравітаційні сили, спричинені людиною.

– механічні забруднення – це різні тверді частки та предмети (викинуті як непридатні, спрацьовані, вилучені з вжитку).

– зімічні забруднення –тверді, газоподібні й рідкі речовини, хімічні елементи й сполуки штучного походження, які надходять - у біосферу, порушуючи встановлені природою процеси кругообігу речовин і енергії.

– Біологічні забруднення – різні організми, що з'явилися завдяки життєдіяльності людства - бактеріологічна зброя, нові віруси (збудники СНІДу, хвороби легіонерів, епідемій, інших хвороб, а також катастрофічне розмноження рослин чи тварин, переселених з одного середовища в інше людиною чи випадково.

Джерела забруднення дуже різноманітні: серед них не тільки промислові підприємства і паливно-енергетичний комплекс, але і побутові відходи, відходи тваринництва, транспорту, а також хімічні речовини, які людина цілеспрямовано вводить до екосистеми для захисту корисних продуцентів і консументів від шкідників, хвороб і бур'янів.

Серед інгредієнтів забруднення – тисячі хімічних сполук, особливо важкі метали та оксиди, токсичні речовини та аерозолі. Різні джерела викидів можуть бути однаковими за складом і характером забруднюючих речовин. Так вуглеводні надходять у атмосферу і при спалюванні палива, і від нафтопереробної промисловості, і від газовидобувної промисловості.

Джерела забруднюючих речовин різноманітні, також багаточисельні види відходів і характер їхнього впливу на компоненти біосфери. Біосфера забруднюється твердими відходами, газовими викидами і стічними водами металургійних, металообробних і машинобудівних заводів. Величезної шкоди завдають водяним ресурсам стічні води целюлозно-паперової, харчової, деревообробної, нафтохімічної промисловості [19].

Розвиток автомобільного транспорту призвів до забруднення атмосфери міст і транспортних комунікацій важкими металами і токсичними вуглеводнями, а постійне зростання масштабів морських перевезень викликало майже повсюдне забруднення морів і океанів нафтою і нафтопродуктами. Масове застосування мінеральних добрив і хімічних засобів захисту рослин призвело до появи отрутохімікатів в атмосфері, ґрунтах і природних водах, забрудненню

біогенними елементами водойм, водотоків і сільськогосподарської продукції (нітрати, пестициди і т.п.).

При гірських розробках на поверхню землі витягаються мільйони тон різноманітних, найчастіше фітотоксичних гірських порід, що утворюють терикони і відвали, що пилять і горять . В процесі експлуатації хімічних заводів і теплових електростанцій також утворюються величезні кількості твердих відходів (недогарок, шлаки, золи і т.п.), що складуються на великих площах, вчиняючи негативний вплив на атмосферу, поверхневі і підземні води, ґрунтовий покров (пилування, виділення газів і т.п.).

ВИСНОВКИ

У результаті виконання кваліфікаційної роботи побудовано модель розподіленої комп'ютерної системи криптоаналізу на основі Raspberry Pi та реалізовано її вигляді кластеру на основі восьми мінікомп'ютерів для організації обчислень щодо шифрування/дешифрування повідомлень. При цьому забезпечено програмне управління серверними компонентами з клієнтської станції. В основі функціонування розподіленої системи криптоаналізу лежать алгоритми машини Enigma. Алгоритм, що використовувався для дешифрування текстових повідомлень, передбачає перебір можливих комбінацій щодо значень роторів.

При виконанні кваліфікаційної роботи забезпечено розв'язання таких задач:

– організовано кластер на основі мінікомп'ютерів Raspberry PI;

– налаштовано параметри паралельної і розподіленої обробки даних;

– програмно реалізовано алгоритми шифрування/дешифрування текстових повідомлень;

– організовано програмний блок управління розподіленими обчисленнями;

– обгрунтовано застосування засобів діагностики коректності функціонування кластера;

– забезпечено можливість вибору типу криптоаналізу в залежності від структури і виду вхідних повідомлень;

– забезпечено можливість фіксації результатів криптоаналізу;

– передбачено можливість гнучкого налаштування параметрів серверів.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Петин В. Микрокомпьютеры Raspberry Pi: Практическое руководство. БХВ-Петербург. 2015. 240 с.

2. Магда Ю. Raspberry Pi. Руководство по настройке и применению. Litres. 2017 р. 161 с.

3. Макаров С. Arduino Uno и Raspberry Pi 3: от схемотехники к интернету вещей. Litres. 2019 р. 202 с.

4. Тиммонс-Браун М. Робототехника на Raspberry Pi для юных конструкторов и программистов Робототехника на Raspberry Pi для юных конструкторов и программистов. БХВ-Петербург. 2020. 208 с.

5. Петин В. Датчики для Arduino и Raspberry Pi в проектах Internet of Things. БХВ-Петербург. 2016. 320 с.

6. Python 3.9.2 documentation. URL: https://docs.python.org/3/ (дата звернення 08.04.2022 р.)

7. Mathematical statistics functions. URL: https://docs.python.org/3/library/statistics.html (дата звернення 08.04.2022 р.)

8. Краткое руководство по библиотеке Python Requests. URL: https://pythonru.com/biblioteki/kratkoe-rukovodstvo-po-biblioteke-python-requests (дата звернення 16.04.2022 р.)

9. JSON encoder and decoder. URL: https://docs.python.org/3/library/json.html (дата звернення 29.04.2022 р.)

10. SQL Syntax. URL: https://www.w3schools.com/sql/sql_syntax.asp (дата звернення 06.05.2022 р.)

11. Пасічник В., Резніченко В. Організація баз даних та знань. К:. Видавнича група BHV, 2006. 384 с.

12. Ворона В. А., Тихонов В. А. Системы контроля и управления доступом. М.: Горячая линияТелеком. 2010. 272 с.

13. Бесекерский В.А. Руководство по проектированию систем автоматического управления. Москва.: Высшая школа, 2007. 295с.

14. Кузин Л.Т. Расчет и проектирование дискретных систем управления.- М.: ГН ТИМЛ, 2012.- 648 с.

15. Лучшие одноплатники на базе чипа RP2040 в 2022 году. Часть 1. URL: https://habr.com/ru/hub/raspberrypi/ (дата звернення 10.05.2022 р.)

16. Raspberry Pi Computer Boards. URL: https://www.okdo.com/c/pi-shop/the-raspberry-pi/ (дата звернення 15.05.2022 р.).

17. Жидецький В.Ц. Охорона праці користувачів комп'ютерів. Львів: Афіша, 2000. 176 с.

18. НПАОП 0.00-7.15-18 «Вимоги щодо безпеки та захисту здоров'я працівників під час роботи з екранними пристроями»/Міністерство соціальної політики України. Офіц. вид. К. : Парлам. вид-во, 2018. 24 с.

19. Желібо Є., Заверуха Н., Зацарний В. Безпека життєдіяльності. К.: 2001. 483 с.

Додаток А.

Технічне завдання

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії

Кафедра комп'ютерних систем та мереж

"Затверджую"

Завідувач кафедри КС

_____ Осухівська Г.М.

"___" _____2022 р

РОЗПОДІЛЕНА КОМП'ЮТЕРНА СИСТЕМА КРИПТОАНАЛІЗУ НА ОСНОВІ RASPBERRY PI

ТЕХНІЧНЕ ЗАВДАННЯ

на _*10*_ листках

Вид робіт: Кваліфікаційна робота

На здобуття освітнього ступеня «Бакалавр»

Спеціальність 123 «Комп'ютерна інженерія»

«УЗГОДЖЕНО»	«ВИКОНАВЕЦЬ»
Керівник кваліфікаційної роботи	Студентка групи СІс-43
_____ к.т.н., доц. Луцків А.М.	_____ Халак Х.Р.
«____» _____ 2022 р.	«____» _____ 2022 р.

Тернопіль 2022

1 Загальні відомості

1.1 Повна назва та її умовне позначення

Повна назва теми кваліфікаційної роботи: «Розподілена комп’ютерна система криптоаналізу на основі Raspberry Pi».

Умовне позначення кваліфікаційної роботи: КС КРБ 123.235.00.00

1.2 Виконавець

Студентка групи СІс-43, факультету комп’ютерно-інформаційних систем і програмної інженерії, кафедри комп’ютерних систем та мереж, Тернопільського національного технічного університету імені Івана Пулюя, Халак Христина Русланівна.

1.3 Підстава для виконання роботи

Підставою для виконання кваліфікаційної роботи є наказ по університету (№ 4.7-180 від 23.03.2022 р.)

1.4 Планові терміни початку та завершення роботи

Плановий термін початку виконання кваліфікаційної роботи – 23.03.2022 р.

Плановий термін завершення виконання кваліфікаційної роботи – 24.06.2022 р.

1.5 Порядок оформлення та пред’явлення результатів роботи

Порядок оформлення пояснювальної записки та графічного матеріалу здійснюється у відповідності до чинних норм та правил ІСО, ГОСТ, ЕСКД, ЕСПД та ДСТУ.

Пред’явлення проміжних результатів роботи з виконання кваліфікаційної роботи здійснюється у відповідності до графіку, затвердженого керівником роботи.

Попередній захист кваліфікаційної роботи відбувається при готовності роботи на 90% , наявності пояснювальної записки та графічного матеріалу.

Пред’явлення результатів кваліфікаційної роботи відбувається шляхом захисту на відповідному засіданні ЕК, ілюстрацією основних досягнень за допомогою графічного матеріалу.

2 Призначення і цілі створення системи

2.1 Призначення системи

Розподілена комп’ютерна система криптоаналізу на основі Raspberry Pi призначена для забезпечення ефективного шифрування та дешифрування повідомлень з використанням мінімальних ресурсів апаратного забезпечення. Дана система призначена для виконання задач, які покладаються на сучасні кластери, однак замість повноцінних серверів у даному випадку будуть використовуватись мінікомп’ютери Raspberry PI. На основі кластеру необхідно реалізувати функціональність щодо кодування і декодування повідомлень по типу машини Enigma, що використовувалась під час Другої світової війни.

Практичне застосування розподіленої комп’ютерної системи криптоаналізу важливе при організації процесів, де шифрування та дешифрування відіграють важливу роль. До таких сфер її застосування належать військова галузь, відділи

кібербезпеки і кіберполіції, ІТ-компанії, які займаються кіберзахистом, а також у навчальному процесі при вивченні дисциплін, пов’язаних із захистом інформації.

Алгоритм, який пропонується для декодування повідомлень полягає у повному переборі можливі комбінацій – brute force.

2.2 Мета створення системи

Мета побудови розподіленої комп’ютерної системи криптоаналізу на основі Raspberry Pi полягає у створенні кластеру на основі восьми мінікомп’ютерів для організації обчислень щодо шифрування/дешифрування повідомлень.

До основних задач, які покликана розв’язати дана система належать:

– організація кластеру на основі мінікомп’ютерів Raspberry PI;
– налаштування параметрів паралельної і розподіленої обробки даних;
– програмна реалізація алгоритмів шифрування/дешифрування алгоритмів;
– організація блоку управління розподіленими обчисленням;
– надання засобів діагностики коректності функціонування кластера;
– забезпечення можливості криптоаналізу в залежності від типу вхідних повідомлень;
– підвищення криптостійкості комп’ютерних систем;
– забезпечення можливості фіксації результатів криптоаналізу;
– можливість інтеграції із суміжними системами;
– підвищення якості і надійності в процесі захисту інформації та обміну між вузлами комп’ютерних систем.

2.3 Характеристика об'єкту

2.3.1 Основні задачі та функції об’єкту

До основних задач, які має виконувати розподілена комп’ютерна система криптоаналізу на основі Raspberry Pi належать здатність шифрування текстових повідомлень і їх розкодування на основі принципів організації машини Enigma.

Складність побудови такої системи полягає у великій кількості можливих комбінацій щодо шифрування і відповідно великою кількістю невідомих параметрів при розкодуванні зашифрованого тексту.

Для того, щоб досягти мети роботи необхідно на практиці розв’язати дві основні задачі, зокрема, організація кластера для прискорення процесів шифрування/дешифрування із застосування паралельних і розподілених обчислень, та власне реалізації програмного забезпечення для емуляції роботи машини Enigma.

Організацію кластера необхідно реалізувати не менше, ніж на восьми станціях Raspberry PI, що, крім цього, передбачає наявність одного клієнта для управління ним. Це вимагає зміни налаштування конфігурацій серверних станцій і відповідно й клієнта. Окрім цього, комунікація між клієнтом і серверами повинна бути забезпечення на основі комунікаційної мережі – безпровідної комп’ютерної мережі.

Безпровідну комп’ютерну мережу потрібно налаштувати таким чином, щоб задіяти можливість формування динамічних IP-адрес маршрутизатором у межах станцій, які належать до кластеру.

Програмне забезпечення управління кластером повинно забезпечувати можливість запуску додатків з клієнта, а також віддаленого їхнього перезавантаження та вимкнення.

3 Вимоги до системи

3.1 Вимоги до системи в цілому

Розподілена комп’ютерна система криптоаналізу на основі Raspberry PI повинна виконувати функції з шифрування та дешифрування текстових повідомлень з можливістю генерації шифрованого ключа і без нього. Система повинна забезпечувати продуктивність виконання розподілених обчислень з використанням 32 процесорних ядер та 8 ГБ оперативної пам’яті на кластер. Керування розподіленою системою повинно виконуватися зі станції клієнта і передбачати можливість запуску

програмного забезпечення, що підлягає паралельному і розподіленому опрацюванні текстових даних. Результатом успішної роботи кластера вважається повна відповідність одержаного зашифрованого повідомлення вхідному, яке підлягає дешифруванню.

3.1.1 Вимоги до структури та функціонування системи

Організація розподіленої комп'ютерної системи передбачає застосування таких структурних компонентів як:

– сервер на базі Raspberry PI – 8 шт.;
– клієнт на базі Raspberry PI – 1 шт.;
– маршрутизатор для функціонування розподіленої комп'ютерної системи;
– маршрутизатор для доступу до мережі Інтернет;
– бібліотеки Python для управління кластером;
– бібліотеки Python для реалізації алгоритмів машини Enigma.

Основними функціональними вимогами до розподіленої комп'ютерної системи криптоаналізу на основі Raspberry PI є:

– можливість налаштування діапазону IP-адрес для їх видачі серверам і клієнту системи;
– здатність віддаленого вимкнення серверів;
– можливість дистанційного перезавантаження серверів;
– можливість виконання керованого розподіленого обчислення на серверах;
– здатність запуску програмного забезпечення для виконання задач криптоаналізу;
– здатність виконувати алгоритми, передбачені алгоритмами функціонування машини Enigma;
– здатність розраховувати кількість комбінацій при шифруванні/дешифруванні повідомлень;
– забезпечення можливості реалізації алгоритму brute force;
– здатність до масштабованості кількості серверів у кластері.

3.1.2 Вимоги до способів та засобів зв’язку між компонентами системи

Зв’язок між компонентами розподіленої комп’ютерної системи криптоаналізу на основі Raspberry PI організовано на основі технології безпровідного зв’язку WiFi. Обмін даними між клієнтом і серверами визначається налаштуванням маршрутизатора кластера. Доступ до мережі Інтернет забезпечується шляхом використання роутера, віддміного від маршрутизатора кластера для забезпечення коректності функціонування розподіленої системи.

3.1.3 Вимоги по діагностуванню системи

До вимог щодо діагностики системи можна віднести наявність програмних засобів для налаштування конфігурації клієнта і сервера щодо можливості роботи у визначеній безпровідній комп’ютерній мережі. Діагностика серверів може проводитися віддалено або шляхом зміни даних на SD карті. Розклад за яким проводиться діагностика визначає стейкхолдер розподіленої комп’ютерної системи, або у випадку виникнення некоректної роботи компонентів системи.

3.1.4 Перспективи розвитку, модернізація системи

Перспективами розвитку і модернізації розподіленої комп’ютерної системи є здатність до масштабування кількості серверних компонентів, що дозволить збільшити продуктивність виконання задач криптоаналізу. Окрім цього, важливим показником щодо перспектив розвитку системи є здатність переходу до новіших версій апаратного забезпечення Raspberry PI та оновлення як системного так і прикладного програмного забезпечення управління кластером.

У разі внесення коректив, елементів додаткової функціональності або заміни існуючого програмного забезпечення, комп’ютерна система має надійно реагувати на ці фактори без втрати існуючих даних.

3.1.5 Вимоги до надійності системи

Розподілена комп’ютерна система криптоаналізу на основі Raspberry Pi повинна мати механізми авторизованого доступу до серверів та клієнта. Окрім цього, надійність функціонування системи повинна проявлятись у достовірності результатів криптоаналізу, їх стійкості та придатності до використання. Важливо підтримувати задану функціональність при зростанні навантаження на обчислювальні ресурси, визначені технічними характеристиками Raspberry PI та маршрутизатора.

При виникненні помилок або збоїв роботи розподіленої комп’ютерної системи, повинна бути забезпечена можливість надійного її функціонування до того часу, поки не буде виявлено причини їх виникнення та усунуто неполадки.

3.1.6 Вимоги до функцій та задач, які виконує система

Основними вимогами відносно функцій і задач, які виконує розподілена комп’ютерна система криптоаналізу на основі Raspberry Pi є:

– забезпечення безпровідного доступу і комунікації між компонентами комп’ютерної системи;

– здатність віддаленого управління серверними станціями;

– здатність забезпечувати виконання функцій при емуляції машини Enigma;

– визначена продуктивність при застосуванні алгоритму повного перебору комбінацій brute force;

– можливість запуску задач криптоаналізу з клієнта;

– забезпечення стабільності результатів шифрування/дешифрування текстових повідомлень.

3.1.7 Вимоги до апаратного забезпечення

Вимоги до апаратного забезпечення метеостанції на базі Raspberry PI 3:

– 4-ядерний 64-бітний процесор з тактовою частотою 1,2 ГГц;

– інтерфейс передачі даних з підтримкою протоколу Wi-Fi 802.11n;

– підтримка Bluetooth 4.1;

– об’єм оперативної пам’яті – 1GB;
– наявність USB-портів;
– Ethernet порт;
– слот під microSD;
– графічне ядро VideoCore IV 3D;
– послідовний інтерфейс камери (CSI);
– послідовний інтерфейс монітора (DSI);
– 40 пінів вводу/виводу (GPIO);
– HDMI – порт.

Вимоги до маршрутизатора:

– підтримка частотного діапазону 2,4 ГГц;
– швидкість передчі даних на рівні 100 Мб/с;
– радіус покриття без втрат даних – 10 м.

3.1.8 Вимоги до програмного забезпечення

Програмне забезпечення мікроконтролера Raspberry PI – операційна система Raspbian, Python 3.

4 Вимоги до документації

Документація повинна відповідати вимогам ЄСКД та ДСТУ

Комплект документації повинен складатись з:

– пояснювальної записки;
– графічного матеріалу:

1 Архітектура розподіленої комп’ютерної системи криптоаналізу.
2 Принцип організації функціонування машини Enigma.
3 Алгоритм функціонування Enigma.
4 Алгоритм brute force;
5 Функція визначення положень роторів.

*Примітка: У комплект документації можуть вноситися міни та доповнення в процесі розробки.

5 Стадії та етапи проектування

Таблиця 1 – Стадії та етапи виконання кваліфікаційної роботи бакалавра

№ етапу	Назва етапу виконання кваліфікаційної роботи	Термін виконання
1	Розробка і затвердження технічного завдання	23.03-30.03.2022
2	Аналіз технічного завдання	30.03-02.04.2022
3	Визначення вимог до апаратного та програмного забезпечення комп'ютерної системи	02.04-18.04.2022
4	Проектування схеми метеостанції	19.04-04.05.2022
5	Проектування та реалізація програмного забезпечення метеостанцій	04.05-16.05.2022
6	Розробка інструкцій із встановлення та налаштування параметрів комп'ютерної системи збору та аналізу даних з метеостанцій	16.05-29.05.2022
7	Безпека життєдіяльності, основи охорони праці	01.06-08.06.2022
8	Оформлення кваліфікаційної роботи	09.06-18.06.2022
9	Попередній захист кваліфікаційної роботи	18.06-22.06.2022
10	Захист кваліфікаційної роботи	22.06-25.06.2022

6 Додаткові умови виконання кваліфікаційної роботи

Під час виконання кваліфікаційної роботи у дане технічне завдання можуть вноситися зміни та доповнення.