МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Західноукраїнський національний університет
Факультет комп'ютерних інформаційних технологій
Кафедра кібербезпеки

ГНАТЮК Анастасія Сергіївна

Метод квантового розподілу ключів на основі модифікації протоколу BB84 з використанням системи залишкових класів / Method of quantum key distribution based on a modified BB84 protocol using the Residue Number System

спеціальність: 125 – Кібербезпека та захист інформації

освітньо-професійна програма –Кібербезпека

Кваліфікаційна робота

Виконала студенка групи КБм -21
А.С. Гнатюк

Науковий керівник
д.т.н., професор М.М.Касянчук

Кваліфікаційну роботу допущено
до захисту:

«____» _____ ___2025 р.

Завідувач кафедри
_____ В.В.Яцків

ТЕРНОПІЛЬ – 2025

Факультет комп'ютерних інформаційних технологій
Кафедра кібербезпеки
Освітній ступінь «магістр»
спеціальність: 125 – Кібербезпека та захист інформації
освітньо-професійна програма – Кібербезпека

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ В.В.Яцків
«____» _____ 2024 року

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТЦІ

Гнатюк Анастасії Сергіївній
(прізвище, ім'я, по батькові)

1. Тема кваліфікаційної роботи:
Метод квантового розподілу ключів на основі модифікації протоколу BB84 з використанням системи залишкових класів / Method of quantum key distribution based on a modified BB84 protocol using the Residue Number System
керівник роботи д.т.н., професор М.М. Касянчук

затверджені наказом по університету від 29 листопада 2024 року № 938.

2. Строк подання студентом закінченої кваліфікаційної роботи 5 грудня 2025року.
3. Вихідні дані до кваліфікаційної роботи: завдання на кваліфікаційну роботу студента, наукові статті, технічна література.

4. Основні питання, які потрібно розробити:
- проаналізовано теоретичні основи СЗК, а також принципи квантової криптографії та механізм роботи протоколу BB84;
- визначено способи інтеграції СЗК у протокол BB84 та досліджено їх переваги та недоліки;
- розроблено програмну реалізацію модифікованого протоколу BB84 із використанням СЗК при кодуванні квантових станів та постобробці ключа шифрування.

5. Перелік графічного матеріалу у роботі:
- модель квантової криптографії;
- схема роботи протоколу BB84 при використанні СЗК в постобробці ключа;
- схема роботи протоколу BB84 при введені СЗК для кодування квантових станів;

- кодування класу BB84_SZK;
- кодування функції bb84_classic;
- кодування функції simulate_bob;
- кодування функції szk_postprocess для кодування станів;
- кодування функції szk_postprocess для постобробки;
- кодування функції run_simulation для СЗК у кодуванні станів;
- кодування функції run_simulation для СЗК у постобробці;
- головне вікно програми;
- вивід результатів роботи програми.

6. Консультанти розділів кваліфікаційної роботи

	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання 29 листопада 2024 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строки виконання етапів кваліфікаційної роботи	Примітка
1	Теоретичні основи квантової криптографії та системи залишкових класів	12.2024 р. – 03.2025 р.	
2	Метод модифікації протоколу BB84 з використанням СЗК	03.2025 р. – 06.2025 р.	
3	Реалізація та аналіз модифікованого протоколу BB84	06.2025 р. – 11.2025 р.	

Студентка _____ А. С. Гнатюк
(підпис)

Керівник роботи _____ М. М. Касянчук
(підпис)

АНОТАЦІЯ

Гнатюк А. С. Метод квантового розподілу ключів на основі модифікації протоколу BB84 з використанням системи залишкових класів. – Рукопис.

Дослідження на здобуття освітнього ступеня «магістр» за спеціальністю 125 «Кібербезпека та захист інформації», освітньо-професійна програма «Кібербезпека». – Західноукраїнський національний університет, Тернопіль, 2025.

У роботі виконано аналіз принципів роботи протоколів квантового розподілу ключів та математичних основ СЗК. Розроблено модифіковані методи формування квантового ключа шляхом інтеграції СЗК у протокол BB84 на етапах постобробки та кодування квантових станів. Реалізовано алгоритмічні моделі передачі ключа із використанням залишків та виконано програмну реалізацію запропонованого методу. Показано, що застосування СЗК дозволяє зменшити обсяг збережених даних, підвищити пропускну здатність каналу та забезпечити додаткову стійкість ключа до атак, включно з атаками на класичний канал та атаками на зменшення ентропії ключа.

КЛЮЧОВІ СЛОВА: КВАНТОВИЙ РОЗПОДІЛ КЛЮЧІВ, ПРОТОКОЛ BB84, СИСТЕМА ЗАЛИШКОВИХ КЛАСІВ, КРИПТОГРАФІЯ, МОДУЛЬНА АРИФМЕТИКА.

ABSTRACT

Hnatiuk A. S. Method of Quantum Key Distribution Based on a Modified BB84 Protocol Using the Residue Number System. – Manuscript.

A research study submitted for the Master's degree in specialty 125 "Cybersecurity and Information Protection", educational and professional program "Cybersecurity". – Western Ukrainian National University, Ternopil, 2025.

In this research, an analysis of the operating principles of quantum key distribution protocols and the mathematical foundations of the RNS has been conducted. Modified methods for quantum key generation have been developed by integrating RNS into the BB84 protocol at the stages of post-processing and quantum state encoding. Algorithmic models of key transmission using residues were created, and a software implementation of the proposed method was carried out. It has been demonstrated that the use of RNS reduces the amount of stored data, increases channel throughput, and provides additional resistance against attacks, including attacks on the classical channel and attacks aimed at reducing key entropy.

Keywords: QUANTUM KEY DISTRIBUTION, BB84 PROTOCOL, RESIDUE NUMBER SYSTEM, CRYPTOGRAPHY, MODULAR ARITHMETIC.

ЗМІСТ

ВСТУП

Стрімкий розвиток інформаційно-комунікаційних технологій (ІКТ) та глобальна цифровізація суспільства привели до масштабного зростання обсягів даних, що передаються і обробляються в сучасних телекомунікаційних і комп'ютерних системах. Зміцнення цифрового середовища супроводжується підвищеною залежністю суспільства від національних та корпоративних інформаційних ресурсів, що підкреслено у міжнародних стратегіях інформаційної безпеки та документах, присвячених формуванню глобального цифрового простору [1].

В умовах постійного розширення інформаційного простору телекомунікаційні системи, інформаційні мережі, автоматизовані системи управління та цифрові сервіси утворюють критичну інформаційну інфраструктуру (КІІ), від стійкості якої залежить безпека державних, комерційних і соціальних процесів. Її суб'єкти несуть відповідальність за збереження конфіденційності, цілісності, доступності та достовірності даних, що обумовлює необхідність захисту каналів інформаційного обміну і застосування криптографічних механізмів, здатних протистояти сучасним та перспективним загрозам [2].

У той же час стрімкий розвиток обчислювальних технологій, зокрема квантових обчислень, ставить під сумнів надійність класичних криптографічних протоколів. Алгоритми, які сьогодні вважаються стійкими, потенційно можуть бути зламані з появою квантових обчислювальних систем, що робить проблему захисту інформації системною загрозою на міждержавному рівні [3,4].

Одним із перспективних рішень є використання методів квантової криптографії, зокрема квантового розподілу ключів (QKD), що дозволяє реалізувати криптографічний захист на фізичному рівні шляхом використання квантово-механічних властивостей фотонів. Протокол BB84 є найбільш відомим представником даного класу, однак практичне застосування супроводжується

низкою викликів, пов’язаних із помилками вимірювань, високою чутливістю до атак і складністю постобробки ключових даних [5,6].

Інтеграція математичних методів, зокрема системи залишкових класів (СЗК), відкриває можливість оптимізації обробки квантових ключів, підвищення пропускної здатності каналу та підсилення стійкості протоколів QKD до атак, що впливають на класичний та квантовий канали обміну [7,8]. Це дозволяє розглядати взаємодію квантових і класичних криптосистем як комплексне рішення, що формує перспективи побудови безпечного інформаційного середовища та постквантової інфраструктури [9,10].

Мета роботи. Метою даної роботи є розробка та дослідження методів інтеграції СЗК у протокол квантового розподілу ключа BB84 для оптимізації генерації, передачі та обробки криптографічного ключа, а також створення програмної реалізації такого поєднання.

Для досягнення поставленої мети вирішуються наступні завдання:

- проаналізовано теоретичні основи СЗК, а також принципи квантової криптографії та механізм роботи протоколу BB84;
- визначено способи інтеграції СЗК у протокол BB84 та досліджено їх переваги та недоліки;
- розроблено програмну реалізацію модифікованого протоколу BB84 із використанням СЗК при кодуванні квантових станів та постобробці ключа шифрування.

Об’єкт дослідження. Процес квантового розподілу ключів в протоколі BB84 на основі використання СЗК.

Предмет дослідження. Методи інтеграції СЗК у протокол BB84, їх вплив на пропускну здатність, гнучкість та безпеку ключа, а також програмна реалізація такого поєднання.

Методи дослідження. Математичні методи модульної арифметики, квантово-криптографічні моделі, алгоритми оптимізації постобробки, методи статистичного аналізу помилок.

Наукова новизна одержаних результатів.

1. Вперше запропоновано метод інтеграції СЗК у процес формування та постобробки ключів протоколу BB84**,** що дозволяє підвищити стійкість до атак, спрямованих на класичний канал обробки даних та маніпуляцію помилками;

2. Розроблено модель представлення квантового ключа у вигляді набору залишків**,** що дозволяє зменшити обчислювальні витрати на класичну постобробку, зокрема на корекцію помилок і підсилення конфіденційності;

3. Обґрунтовано вплив параметрів СЗК на стійкість модифікованого протоколу BB84**,** що дало змогу визначити оптимальні умови для підвищення ефективності захисту та мінімізації об’єму переданої додаткової інформації;

4. Встановлено закономірності між кількістю модулів у СЗК та рівнем стійкості до різноманітних атак, що дозволило підвищити стійкість протоколу без зміни квантової частини обміну.

Практичне значення отриманих результатів. Здійснена програмна реалізація методу інтеграції СЗК у протокол BB84 при кодуванні квантових станів та постобробці ключа шифрування.

Публікація та апробація КР.

1. Гнатюк А., Касянчук М. Метод формування квантового ключа за протоколом BB84 на основі системи залишкових класів. *Кібербезпека державних інституцій та подолання кризових станів*: матеріали III Міжнар. наук.-практ. конф. в 2 т. (Київ – Прага – Таллінн – Тернопіль), 14 листоп. 2024 р. / ІСЗЗІ КПІ ім. Ігоря Сікорського. Київ, 2024. Т. 1. С. 398-399 [11].

2. Гнатюк А., Касянчук М., Басістий П. Модифікація квантового протоколу BB84 за допомогою системи залишкових класів. Матеріали XIV міжнародної науково-практичної конференції «Безпека інформаційних технологій: ITSEC-2025». С. 47-49 [12].

3. A. Hnatiuk; M. Kasianchuk; R. Shevchuk; L. Tymoshenko. Method of Quantum Key Distribution Based on a Modified BB84 Protocol Using the Residue Number System. 2025 15th International Conference on Advanced Computer Information Technologies (ACIT), Sibenik, Croatia, 2025, pp. 466-470 [13].

1 ТЕОРЕТИЧНІ ОСНОВИ КВАНТОВОЇ КРИПТОГРАФІЇ ТА СИСТЕМИ ЗАЛИШКОВИХ КЛАСІВ

1.1 Основні принципи квантової криптографії

Квантова криптографія відноситься до різних методів кібербезпеки для шифрування та безпечної передачі даних, базуючись на натуральних та незмінних законах квантової механіки [14,15]. Хоча вона досі на ранніх стадіях розвитку, квантова криптографія має потенціал стати набагато безпечнішою, ніж попередні види криптографічних алгоритмів і є, в теорії, незламною.

На відміну від традиційної криптографії, яка побудована на математиці, квантова криптографія заснована на законах фізики:

1) принцип невизначеності Гейзенберга стверджує, що в квантовій системі можна з точністю знати лише одну з пари спряжених величин, наприклад, положення та імпульс (вимірювання положення частинки призведе до порушення її швидкості). Квантова криптографія використовує це, застосовуючи поляризацію фотонів (оскільки фотони можуть передаватися через волоконно-оптичні лінії) у різних базисах як спряжені властивості [14];

2) теорема про заборону клонування є наслідком попереднього принципу і стверджує, що неможливо створити ідентичні копії невідомого квантового стану [16]. Завдяки цьому можна виявити, чи хтось перехопив квантовий канал під час критичної передачі інформації;

3) квантова заплутаність це явище, при якому стани двох або більше об'єктів можуть бути описані у відношенні один до одного, навіть якщо окремі об'єкти просторово розділені, що призводить до кореляцій між фізичними властивостями систем [17].

Ці властивості унеможливлюють вимірювання квантового стану будь-якої системи без її порушення.

Фотони використовуються у квантовій криптографії через те, що вони мають всі потрібні атрибути: їхня поведінка добре зрозуміла, і вони є носіями інформації в оптичному волокні. Одним з найкращих прикладів використання

квантової криптографії є метод квантового розподілу ключів, який забезпечує безпечний метод передачі ключа.

В теорії, квантова криптографія працює за моделлю, розробленою ще в 1984 році [18]. Вона припускає, що дві людини, яких звати Аліса та Боб, хочуть провести безпечний обмін повідомленнями. Аліса ініціює повідомлення через надсилання ключа Бобу. Ключ – це потік фотонів, які подорожують в одному напрямку. Кожен фотон несе в собі один біт даних (0 або 1). Проте, на додачу до їх лінійної подорожі, ці фотони коливаються або вібрують певним чином.

Так, перед тим як Аліса (відправник) ініціює повідомлення, фотони проходять через поляризатор. Поляризатор – це пристрій, який перетворює неполяризований промінь електромагнітних хвиль (тобто світло), в промінь з одним станом поляризації.

Стани поляризації можуть бути такими: вертикальна (1 біт) та горизонтальна (0 біт); 45 градусів (1 біт) та 135 градусів (0 біт), як показано в таблиці 1.1. Передача має одну з двох поляризацій, що представляють один біт (0 або 1), у будь-якій схемі, яку вона використовує [18,19].

Таблиця 1.1 – Стани поляризації

Базис	Бітове значення	Стан поляризації
Прямолінійний (+)	1	↑
	0	→
Діагональний (x)	1	↗
	0	↖

Далі ці фотони проходять через оптичне волокно від поляризатора до отримувача - Боба. Цей процес використовує розділювач променя, який зчитує поляризацію кожного фотона. Коли Боб отримує фотонний ключ, він не знає правильного стану поляризації фотонів, тому одна поляризація вибирається

випадковим чином. Після цього Аліса порівнює використані Бобом базиси для поляризації ключа і доводить до відома Боба, який базис вона використовувала для відправки кожного фотона. Боб повідомляє Алісу, які базиси збіглися. Фотони, зчитані неправильним чином, потім відкидаються, а решта послідовності стає ключем.

Припустимо, що є Єва (зловмисник), яка прослуховує канал обміну ключем і має ті ж інструменти, що і Боб. Але Боб має перевагу, якої не має Єва, а саме: він може говорити з Алісою для підтвердження типу поляризатора для кожного фотона. В результаті Єва неправильно відображає фінальний ключ. Аліса і Боб також будуть знати, якщо їх підслуховували, адже перехоплення Євою потоку фотонів змінює їх позиції. В результаті вони надходять не в тому стані, який очікували відправник та отримувач, що показано на рисунку 1.1 [20].

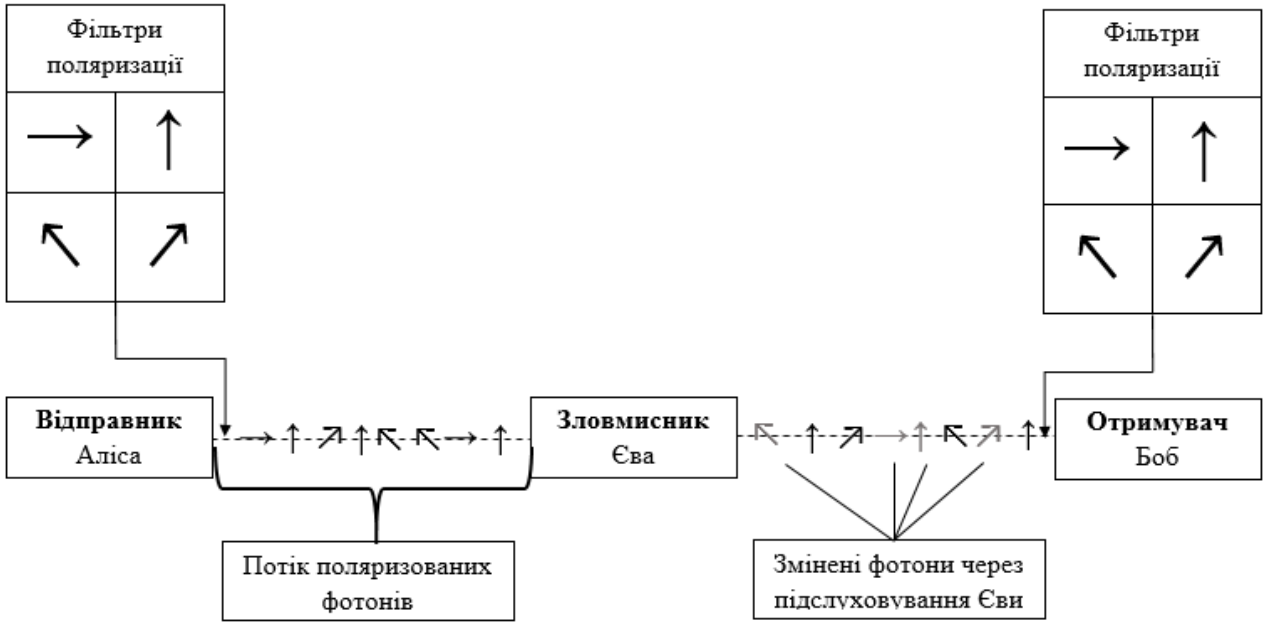

Рисунок 1.1 – Модель квантової криптографії

Квантова криптографія дозволяє користувачам спілкуватися більш безпечно порівняно з традиційною криптографією. Після обміну ключами між залученими сторонами є невелике занепокоєння, що зловмисник зможе декодувати дані без ключа. Якщо ключ спостерігається при його створенні, очікуваний результат змінюється, що помічає як відправник, так і отримувач.

Перевагами квантової криптографії є:

1) забезпечення безпечного зв'язку. Квантова криптографія базується на законах фізики, що є більш складним і безпечним методом шифрування ніж складні для зламу числа;

2) виявлення підслуховування. Якщо третя особа намагатиметься прочитати закодовані дані, тоді стан квантів змінюється, змінюючи очікуваний користувачами результат;

3) різноманіття методів безпеки. Оскільки використовується багато протоколів квантової криптографії деякі з них, наприклад QKD, для підвищення безпеки поєднуються з класичними методами шифрування [21,22].

Потенційні обмеженнями квантової криптографії можуть бути такі:

1) зміни поляризації та можливості виникнення помилки. Фотони можуть змінити поляризацію під час передачі, що потенційно збільшує можливості помилок;

2) дальність. Максимальна відстань досяжності зазвичай лежала в межах 400-500 кілометрів;

3) витрати. Квантова криптографія зазвичай потребує власну інфраструктуру, що використовує оптичні канали і повторювачі;

4) кількість точок призначення. При квантовій передачі неможливо надіслати ключі до двох і більше місць по одному квантовому каналу [23,24].

На даний момент, традиційного шифрування даних було достатньо для підтримки безпечного зв'язку в більшості середовищ кібербезпеки. Проте, розвиток квантових обчислень створює загрозу існування навіть для найстійкіших традиційних алгоритмів шифрування.

Як квантова криптографія, квантові обчислення – це стрімко зростаюча технологія, яка також використовує закони квантової механіки. Порівняно до найшвидших та найпросунутіших класичних комп'ютерів, квантові комп'ютери мають потенціал для вирішення складних завдань набагато швидше.

Математик Пітер Шор вперше описав загрозу, яку становлять квантові комп'ютери для традиційних систем захисту ще у 1994 році [23]. Сьогодні,

криптосистеми можна розділити на дві основні категорії: симетричні системи, які використовують один ключ для шифрування та розшифрування даних; та асиметричні, що використовують публічний ключ, який може бути переглянути будь-ким, та секретний ключ, який доступний тільки для авторизованих осіб. Обидва типи криптосистем створюють дані ключі через перемноження великих простих чисел і покладаються на масивні обчислювальні потужності, що потрібні для ділення великих чисел для забезпечення захищеності цих ключів від хакерів та підслуховувачів.

Навіть найпотужнішим суперкомп'ютерам на світі будуть потрібні тисячі років щоб математично зламати сучасні алгоритми шифрування, як AES (Advanced Encryption Standard) або RSA. Відповідно до алгоритма Шора, ділення великих чисел на класичних комп'ютерах потребуватиме стільки обчислювальної потужності, що хакеру знадобилося б багато життів, щоб навіть наблизитися до результату. Проте, повністю функціональний квантовий комп'ютер, якщо його створять, може потенційно знайти рішення за лічені хвилини.

З цієї причини, можливості використання квантової криптографії незліченні, як і для будь-якої іншої форми криптографії. У випадку, якщо все від корпоративної інформації до державних таємниць має бути захищеним, коли квантові обчислення звели існуючі криптографічні алгоритми в ніщо, квантова криптографія може стати єдиним способом для захисту приватних даних.

В той час, як комп'ютерні науковці по всьому світу працюють над розробкою практичних квантових технологій, є важливою розробка нових форм криптографії, щоб підготуватися до ери квантових обчислень. Незважаючи на те, що квантові комп'ютери колись вважалися лише теорією, експерти відзначили, що ми можемо вступити в еру квантових технологій впродовж наступних 20-50 років.

1.2 Опис та механізм роботи протоколу BB84. Порівняння основних протоколів квантового розподілу ключів

Протокол BB84, який було створено Чарльзом Беннетом та Жилем Брассаром в 1984 році, це протокол квантового розподілу ключа, який використовує комбінацію квантового каналу та незахищеного класичного каналу, який потребує аутентифікації з метою розподілу секретного ключа між двома сторонами. Протокол забезпечує безпеку ключа через визначення будь-яких спроб підслуховування квантового каналу, надаючи засоби визначення потенційних порушень безпеки. Протокол BB84 використовує чотири квантові стани, які випадковим чином підготовлені відправником (Алісою), в одному з двох базисів (прямолінійному або діагональному), як показано в таблиці 2.1.

В таблиці 1.2 можна спостерігати передачу бітів від Аліси до Боба, яка призведе до формування спільного секретного ключа, який також відомий, як ключ зсуву в квантовому розподілі ключа [18,25].

Таблиця 1.2 – Передача бітів від Аліси до Боба в квантовому каналу з використанням протоколу BB84

Квантове представлення та виявлення	Бітова послідовність Аліси	1	0	1	0	0	1	0	1
	Поляризація послідовності Аліси	↑	↖	↗	↖	→	↑	↖	↗
	Фільтри поляризації Боба	↑	→	↗	→	↖	↑	↖	↗
	Значення послідовності Боба	1	0	1	0	0	1	0	1
Обговорення по класичному каналу									
	Боб узгоджує свої базиси з Алісою	+	+	x	+	x	+	x	x
	Відсіювання розбіжностей	*	-	*	-	-	*	*	*
	Секретний ключ Аліси і Боба	1		1			1	0	1

Протокол BB84 складається з наступних кроків:

1) Аліса генерує випадкову послідовність бітів, а потім кодує кожен біт в випадково вибрану поляризацію, використовуючи один з двох базисів;

2) певна інформація кодується в неортодоксальних квантових станах як єдині фотони з напрямками поляризації 0, 45, 90 та 135 градусів (відповідно до таблиці 1.1);

3) Аліса надсилає закодовані фотони Бобу через квантовий канал;

4) Боб випадковим чином вибирає один з чотирьох фільтрів поляризації для виміру кожного фотону, відправленого Алісою;

5) Боб виміряє кожен фотон і записує результати в формі нулів та одиниць;

6) після передачі Аліса та Боб обговорюють базиси, які вони використовували для кожного біта по класичному каналу;

7) Аліса та Боб відкидають біти, базиси яких не співпали;

8) біти, що залишилися, формують ключ зсуву;

9) щоб забезпечити безпеку ключа, Аліса і Боб проводять тестування для визначення потенційного підслуховування за допомогою перевірки певної частини їх ключа зсуву через класичний канал;

10) нарешті, вони використовують методи виправлення помилок і посилення конфіденційності, щоб отримати коротший, але безпечний спільний секретний ключ [26,27].

Щоб забезпечити максимальну безпеку ключа, згенерованого за допомогою протоколу BB84, Аліса та Боб ретельно проводять тестування для визначення факту підслуховування, що включає в себе порівняння випадково вибраної частини ключа за допомогою класичного каналу. Аліса передає свою випадково вибрану частину Бобу, який потім порівнює її зі своєю частиною, після чого ділиться результатами з Алісою. Таке порівняння дає змогу підтвердити чи спростувати присутність підслуховування в квантовому каналі, підтверджуючи безпеку ключа. Відповідно, відсутність збігу визначає потенційне підслуховування, що призводить до негайного завершення

протоколу. Цей тест, названий «посилення конфіденційності», є невід'ємною частиною захисту розповсюдження ключа.

Підсумовуючи протокол, техніки виправлення помилок та посилення конфіденційності вводяться для формування коротшого, але більш безпечного спільного секретного ключа. Точність автентичності ключа залежить від вимірювання Алісою і Бобом фотонів в однакових базисах, гарантуючи 100% точність, або в різних базисах, знижуючи точність до 50%. По суті, протокол BB84 дозволяє двом сторонам створити спільний секретний ключ за допомогою незахищеного каналу, гармонізуючи принципи квантової механіки з класичними способами комунікації.

У сучасній квантовій криптографії існує низка протоколів квантового розподілу ключів, які різняться між собою різними принципами роботи, рівнем стійкості до атак та складністю реалізації. Для їх подальших досліджень та модифікацій важливо розуміти принципи роботи вже існуючих протоколів:

1) протокол BB84, який було описано вище, вважається першою функціональною моделлю квантового розподілу ключів, та його досить легко модифікувати;

2) протокол E91 було вперше запропоновано Артуром Екертом в 1991 році. На відміну від BB84, E91 базується на квантовій заплутаності. Пари заплутаних фотонів розподіляються між Алісою і Бобом, які потім виміряють їх у різних базисах. Безпека протоколу перевіряється через порушення нерівностей Белла, що вказують на присутність квантової заплутаності та відсутність локальних прихованих змінних. Такий підхід підсилює безпеку, що дозволяє визначити певні типи атак, до яких вразливий протокол BB84. Проте, протокол E91 більш складний до використання, оскільки він потребує генерації заплутаних пар фотонів, а це процес, який є експериментально вимогливим;

3) протокол B92 було запопоновано Чарльзом Беннетом у 1992 році, та є спрощеним варіантом протоколу BB84. Він використовує лише два неортогональні квантові стани, зменшуючи складність фізичного обладнання. Хоча він і простіший в принципі, даний протокол жертвує ефективністю і

безпекою в порівнянні до BB84. Він потребує більш комплексної пост-обробки для забезпечення цілісності і є більш піддатливим до певних атак перехоплення та повторного надсилання через його зменшений простір станів;

4) протокол DPS (Differential Phase Shift) було запропоновано у 2002 році. У ньому квантова інформація кодується у відносній фазі між послідовними оптичними імпульсами. Приймач вимірює різницю фаз, що дозволяє йому відновити бітову послідовність. Основною перевагою протоколу DPS є те, що він не потребує ідеальних однофотонних джерел і може використовувати слабкі когерентні імпульси, що робить цього практичнішим до використання в реальних оптичних мережах. Проте реалізація протоколу технічно складна, а його строгий аналіз безпеки є нетривіальним завданням;

5) протокол SARG04 було запропоновано в 2004 році науковцями Сарані, Асіном, Ріборд та Гісіном, є іншим варіантом BB84 та розроблювався спеціально для захисту від атак розчеплення числа фотонів. Поки він використовує такі ж стани поляризації, як протокол BB84, самі біти ключа кодуються не в сам стан поляризації, а в її базис. Така незначна зміна збільшує стійкість протоколу до певних типів атак, особливо коли використовуються слабкі когерентні джерела світла. Хоча протокол SARG04 вводить додаткову складність реалізації, він є безпечнішим у випадку, коли справжні однофотонні джерела недоступні;

6) в 2012 році було зроблено значне покращення у розробці незалежного від вимірювального пристрою квантового розподілу ключів (Measurement-Device Independent Quantum Key Distribution – MDI-QKD). Даний протокол привертає увагу до значної вразливості у системах квантового розподілу ключів а саме: атаки побічних каналів на вимірювальні пристрої. В MDI-QKD, Аліса і Боб готують квантові стани та надсилають їх третій ненадійній стороні, яка виконує вимірювання стану Белла. Оскільки процес вимірювання більше не є надійним, протокол стає імунним до всіх подібних атак. MDI-QKD надає сильний баланс між теоретичною безпекою і практичним

застосуванням, хоча він потребує точної синхронізації і видимого втручання між вхідними квантовими станами.

У таблиці 1.3 наведено порівняння квантових протоколів розподілу ключів, які було наведено вище [17,19,20,28,29].

Таблиця 1.3 – Порівняння протоколів квантового розподілу ключа

Протокол	Рік	Використані стани	Переваги	Недоліки
1	2	3	4	5
BB84	1984	4 стани поляризації у 2х базисах (прямолінійний і діагональний)	Простота реалізації. Висока безпека в теорії і на практиці.	Чутливий до атак на реальні канали (наприклад photon number splitting)
E91	1991	Пари заплутаних фотонів	Використання квантової заплутаності. Перевірка безпеки через нерівності Белла.	Складна технічна реалізація. Вимогливість до апаратури.
B92	1992	Лише 2 незалежні стани	Простіший за BB84. Мінімум обладнання.	Менш ефективний. Слабший захист від атак.
DPS	2002	Фазові когерентні стани	Не потребує ідеальних однофотонних джерел. Практична стійкість до атак.	Складна апаратна реалізація. Важкий аналіз безпеки.

Продовження таблиці 1.3

1	2	3	4	5
SARG04	2004	Модифікація BB84 (4 стани, інший спосіб обробки)	Краще протистоїть PNS-атакам. Підходить для недосконалих джерел фотонів.	Складніша процедура пост-обробки. Нижча ефективність у деяких випадках.
MDI-QKD	2012	Вимірювання стану Бела	Усунення вразливостей через атаки на детектори. висока практична безпека.	Складна технічна реалізація. Потреба в синхронізації та надійному проміжному вузлі

Серед наведених протоколів для подальшого дослідження було обрано BB84, оскільки він є першим і найвідомішим квантовим протоколом розподілу ключів, що став основою для більшості сучасних модифікацій. Його головними перевагами є простота реалізації**,** добра вивченість у теорії та практиці та доведена стійкість до широкого спектра атак. Крім того, BB84 легко піддається удосконаленню та оптимізації, зокрема в частині пост-обробки ключа, що і робить його найбільш придатним для реалізації в рамках цієї роботи.

1.3 Математичні основи системи залишкових класів

Стародавнє дослідження системи залишкових класів (СЗК) починається з математичної загадки з книги III століття н.е., «Суань-цзін», що була написана математиком Сунь Дзи. Вона виглядає наступним чином:

У нас є деяка кількість предметів, число яких невідоме.

Якщо рахувати їх трійками, то залишиться 2,

якщо рахувати їх п'ятірками, то залишиться 3,

якщо рахувати їх сімками, то залишиться 2.

Скільки всього предметів?

Простими словами, питання звучатиме так: яке число при діленні на 3, 5, 7 отримає залишки 2, 3 і 2 відповідно? Сунь Дзи дав вирішення цієї загадки за правилом Тай Єнь (Tai Yen), яке, у 1247 році, було узагальнено іншим китайським математиком Цінь Цзюшао (Qin Jiushao), і стало відомим як китайська теорема про залишки (КТЗ) [30,31].

У 1950-х роках СЗК почали використовувати в комп'ютерних науках для швидких і відмовостійких обчислень завдяки трьом властивостям:

1) паралельне додавання та множення для економії часу;

2) непозиційність - помилка в одній цифрі не впливає на інші;

3) відсутність значущого порядку цифр - помилки зменшують лише діапазон, а не точність [14,15,32].

Проте СЗК має наступні обмеження: складність реалізації ділення, добування кореня та порівняння, а також труднощі з переведенням у звичний для людини вигляд. Останнім часом інтерес до СЗК зріс через її переваги в швидких, енергоефективних обчисленнях (наприклад, у мобільних пристроях) та стійкості до збоїв у спеціалізованих комп'ютерних чіпах. Вона застосовується в цифровій обробці сигналів, кібербезпеці, обробці зображень тощо.

Код залишків найлегше розглядати в термінах лінійних конгруенцій. Відношення конгруенції описується так: $\mathrm{A} \equiv \mathrm{a} \bmod \mathrm{b}$, що читається, як число *A* конгруентне до *a* за модулем *b*. Конгруенція стверджує, що $\mathrm{A} = \mathrm{a} + \mathrm{bt}$ є дійсним для деякого значення *t*, де *A, a, b* і *t* є цілими числами, де *a* – це залишок, а *b* – основа або модуль числа *A*.

Прикладом конгруенції можна вважати наступні співвідношення [33,34]:

$$17 \equiv 12 \bmod 5$$

$$17 \equiv 7 \bmod 5$$

$$17 \equiv 2 \bmod 5$$

В цих прикладах, цілі числа 12, 7 і 2 формують клас залишків 17 за модулем 5. Певної важливості набуває останнє позитивне значення залишків класу, яким в цьому випадку є двійка. Для останнього позитивного значення виконується умова $0 \leq a \leq b$.

Нехай є наступний набір конгруенцій: $A_1 \equiv a_1 \bmod b \ \ldots \ A_n \equiv a_n \bmod b$. Тоді виконуватимуться наступні умови:

1) конгруенції з одним і тим же модулем можна додати і результатом стане дійсна конгруентність: $\sum_{i=1}^{n} A_i \equiv (\sum_{i=1}^{n} a_i) \bmod b$;

2) конгруенції з одним і тим же модулем можна перемножити, і в результаті вийде дійсна конгруенція: $\prod_{i=1}^{n} A_i \equiv (\prod_{i=1}^{n} a_i) \bmod b$;

3) конгруенції є транзитивними. Тобто, якщо $A \equiv B$ і $B \equiv C$, то $A \equiv C$;

4) коректне конгруентне співвідношення зберігається, якщо число, залишок і модуль поділити на їхній спільний множник, або якщо число і залишок поділити на спільний множник, який є взємнопростим з модулем. Тобто якщо $A \equiv a \bmod b$, і d є їх спільним дільником, то: $\frac{A}{d} \equiv \frac{a}{d} \bmod \frac{b}{d}$ і $\frac{A}{d} \equiv \frac{a}{d} \bmod b$ відповідно.

Код залишків - це набір найменших додатних залишків числа за різними модулями (базами). Модулі мають бути взаємно простими, інакше виникає надлишковість. Наприклад, бази 2 і 6, які не є взаємнопростими, дають лише 6 унікальних станів, коли взаємнопрості базиси 3 і 4 – дванадцять, що далі підтверджується в таблиці 1.4.

Таблиця 1.4 – Надлишковість представлення на основі невзаємнопростих модулів

Останній додатній залишок				
1	2	3	4	5
Число	mod 2	mod 6	mod 3	mod 4
0	0	0	0	0
1	1	1	1	1

Продовження таблиці 1.4

1	2	3	4	5
2	0	2	2	2
3	1	3	0	3
4	0	4	1	0
5	1	5	2	1
6	0	0	0	2
7	1	1	1	3
8	0	2	2	0
9	1	3	0	1
10	0	4	1	2
11	1	5	2	3
12	0	0	0	0
13	1	1	1	1
14	0	2	2	2

Числова система, показана у таблиці 1.5, використовує прості базиси 2, 3, 5 і 7. Отже, числова система має 210 станів, які можуть відповідати цілим числам від 0 до 209 [35,36].

Таблиця 1.5 – Натуральні числа та відповідні залишки

Натуральні числа / Базиси	2	3	5	7	Натуральні числа / Базиси	2	3	5	7
1	2	3	4	5	6	7	8	9	10
0	0	0	0	0	15	1	0	0	1
1	1	1	1	1	16	0	1	1	2
2	0	2	2	2	17	1	2	2	3
3	1	0	3	3	18	0	0	3	4
4	0	1	4	4	19	1	1	4	5

Продовження таблиці 1.5

1	2	3	4	5	6	7	8	9	10
5	1	2	0	5	20	0	2	0	6
6	0	0	1	6	21	1	0	1	0
7	1	1	2	0	22	0	1	2	1
8	0	2	3	1	23	1	2	3	2
9	1	0	4	2	24	0	0	4	3
10	0	1	0	3	25	1	1	0	4
11	1	2	1	4	26	0	2	1	5
12	0	0	2	5	27	1	0	2	6
13	1	1	3	6	28	0	1	3	0
14	0	2	4	0	29	1	2	4	1

В даній таблиці показано репрезентацію залишкових чисел, відповідних до додатних цілих чисел в заданому діапазоні, решту з яких можна знайти за допомогою наступних конгруенцій:

$$\mathrm{A} \equiv \mathrm{a} \bmod 2;$$

$$\mathrm{A} \equiv \mathrm{b} \bmod 3;$$

$$\mathrm{A} \equiv \mathrm{c} \bmod 5;$$

$$\mathrm{A} \equiv \mathrm{d} \bmod 7;$$

де *a, b, c* і *d* – числа, які стосуються базисів 2, 3, 5 і 7 відповідно. Дані конгруенції визначають *a, b, c* і *d* для залишкової репрезентації числа *A*.

Представлення чисел у СЗК складається з кількох цифр і передбачається, що воно знаходиться у взаємно однозначній відповідності з деякими додатними цілими числами з дійсної числової системи. Цифри представлення у СЗК є найменшими додатними залишками цих додатних цілих чисел за різними модулями, які формують базиси залишкового представлення. Це є прямим наслідком структури СЗК і властивостей лінійних конгруенцій, тому дії додавання та множення є можливими з однією умовою: система залишків повинна мати певну кількість станів, достатньої для представлення згенерованої

суми або добутку. Якщо ця умова не виконується, то СЗК переповниться, і більше, ніж одна сума або добуток у реальній числовій системі можуть відповідати одному залишковому представленню. Для числа в СЗК із достатньою кількістю станів існує ізоморфне відношення між операціями додавання та множення в цій системі та кінцевою системою цілих додатних чисел.

Кожна цифра СЗК отримується, зважаючи на різні базиси або модулі. З цього слідує, що правила арифметики будуть відрізнятися для кожної цифри. Наприклад, додавання та множення цифр за модулями 2 і 3 слідують правилам, описаним у таблиці 1.6.

Таблиця 1.6 – Сума та добуток за модулями 2 і 3

$\oplus$	0	1
0	0	1
1	1	0

сума mod 2

$\odot$	0	1
0	0	0
1	0	1

добуток mod 2

$\oplus$	0	1	2
0	0	1	2
1	1	2	0
2	2	0	1

сума mod 3

$\odot$	0	1	2
0	0	0	0
1	0	1	2
2	0	2	1

добуток mod 3

Додавання двох систем залишків залежить від помодульного додавання відповідних цифр, які мають відповідати одне одному за базисами або модулями. Модулярне додавання цифр, які мають різні базиси, не визначено.

Множення систем залишків залежить від отриманого добутку відповідних залишків за модулем. Дії додавання та множення двох СЗК позначаються так:

$$S = A \oplus B;$$

$$p = A \odot B.$$

Нехай СЗК має базиси 2, 3, 5 та 7. Припускається, що присутні ізоморфні відношення між нею та реальними додатними числами від 0 до 209. Ізоморфні відношення існують для дій додавання та множення тоді, коли добуток або сума менші за 210. Наступні приклади, що використовують залишки, демонструють

дії додавання та множення і наявність ізоморфізму або його відсутність у випадку переповнення. Залишки буде виділено з використанням дужок.

29+27=S=56;

29 ⇔ (1 2 4 1), 27 ⇔ (1 0 2 6), 56 ⇔ (0 2 1 0);

(1 2 4 1) ⊕ (1 0 2 6) = (0 2 1 0).

Наступні дії вважаються додаванням двох репрезентацій залишків, які було виконано у попередньому рядку:

$1+1 \equiv 0 \bmod 2$;

$2+0 \equiv 2 \bmod 3$;

$4+2 \equiv 1 \bmod 5$;

$1+6 \equiv 0 \bmod 7$.

Нехай є сума двох чисел, яка перевищить 209:

S = 10+200; (0 1 0 2) ⊕ (0 2 0 4) = (0 0 0 6).

Отримана репрезентація залишків (0 0 0 6) відповідає дійсному додатному числу 90. В цьому прикладі сума переповнила дані залишки. Сума, отримана в результаті даних дій, є правильною сумою за модулем 210: $300 \equiv 90 \bmod 210$.

Кінцева система дійсних чисел та СЗК мають однакові характеристики переповнення. Сума, що залишається після переповнення, є правильною сумою відносно модуля, числово рівного кількості станів у скінченній числовій системі.

Далі наведено приклад множення в СЗК:

p = 10*17 = 170;

10 ⇔ (0 1 0 3), 17 ⇔ (1 2 2 3), 170 ⇔ (0 2 0 2);

(0 1 0 3) ⊙ (1 2 2 3) = (0 2 0 2).

Наступні дії вважаються множенням двох репрезентацій залишків, які було виконано у попередньому рядку:

$1*0 \equiv 0 \bmod 2$;

$1*2 \equiv 2 \bmod 3$;

$0*2 \equiv 0 \bmod 5$;

$3*3 \equiv 2 \bmod 7$.

Переповнення в результаті множення не відрізняється від свого аналога при виконанні додавання. Нехай добуток отримано від множення залишків чисел 10 і 100. В результаті вийде число 160 за модулем 210, оскільки $1000 \equiv 160 \bmod 210$ [37,38].

Дію віднімання в СЗК можна здійснити використовуючи доповняльне представлення, що складається з адитивних обернених значень додатного залишкового представлення. Варто зазначити, що є проблема з репрезентацією від'ємних чисел, а саме – деякі механізми, які дозволять використання додатних або від'ємних чисел, повинні бути включені в числову систему. Адитивне обернення завжди існує, оскільки кожен елемент залишків є частиною представлення і визначається наступним чином: $\mathrm{a} \oplus \mathrm{a}' = 0$. Цю формулу можна застосувати як і для певної цифри системи залишків, так і для повної СЗК. Наступний приклад прораховано відносно СЗК з модулем 210:

a = (1 2 4 1), a' = (1 1 1 6) → (1 2 4 1) $\bigoplus$ (1 1 1 6) = (0 0 0 0).

Наступні приклади було обрано для ілюстрації процесу віднімання і, певною мірою, складнощі, що пов'язані з ним:

$$0 = \mathrm{A} \ominus \mathrm{B} = \mathrm{A} \oplus \mathrm{B}'$$

Для початку розглянемо приклад, в якому A більше за B. Нехай A = 200, а B = 100. В СЗК $B' = (0\ 2\ 0\ 5)$, тоді $C = A \oplus B' = (0\ 2\ 0\ 4) \oplus (0\ 2\ 0\ 5) = (0\ 1\ 0\ 2)$. Залишкове представлення різниці відповідає числу 100 у домені дійсних чисел. Далі, розглянемо випадок, коли B більше за A: $A' = (0\ 1\ 0\ 3)$, тоді $D = A' \oplus B$ і $(0\ 1\ 0\ 3) \oplus (0\ 1\ 0\ 2) = (0\ 2\ 0\ 5)$.

Різниця $(0\ 2\ 0\ 5)$ – це адитивна інверсія з $(0\ 1\ 0\ 2)$. Правильна інтерпретація репрезентації $(0\ 2\ 0\ 5)$ може відповідати або +110, або -100, окрім випадків, коли надано додаткову інформацію.

Складнощі, пов'язані з розумінням належності цілого числа до додатних чи від'ємних чисел, можна частково усунути через поділ діапазону залишкових чисел на дві частини. Це саме той метод, який використовується в машинній репрезентації додатних та від'ємних натуральних чисел. Для системи натуральних чисел, можна отримати дві різні машинні представлення від'ємних

чисел, які зазвичай позначаються як доповнення основи і зменшене доповнення основи.

Нехай залишкове представлення з модулями 2, 3, 5 і 7, розділено на дві частини. Залишкові представлення, які відповідають натуральним числам від 0 до 104, вважаються додатними, а ті, які відповідають натуральним числам від 105 до 209 є зворотними представленнями, які асоціюються з від'ємними цілими числами від -1 до -105. Діапазон саме цієї системи числення лежить від -105 до +104. Арифметичні правила, що стосуються знаку та переповнення в цій числовій системі, збігаються з правилами, які зазвичай використовуються в арифметиці доповнення основи.

Процес ділення залишкових кодів ускладняється двома факторами. Першим є відсутність мультиплікативного зворотного до нульового елемента. Другою складністю є те, що процеси ділення залишків та звичайного ділення співпадають тільки тоді, коли часткою є ціле число. Спершу, потрібно розглянути проблему ділення залишків елементів одного поля, а потім – кількох полів, які вважаються кодом залишків. Процес ділення в вигляді рівняння визначається наступним чином:

$\frac{a}{b} = q$, або $a = b * q$.

Різниця між нормальною арифметикою та арифметикою залишків полягає в тому, що в останній добуток b*q не обов'язково має дорівнювати a; потрібна лише конгруентність між a та b*q, де $b * q \equiv a \bmod m_n$.

Множення на мультиплікативне зворотне число b, що позначається, як /b дає наступну конгруенцію: $q \equiv \frac{a}{b} \bmod m_n$.

Правильна інтерпретація числа q у даній конгруенції є такою, що число a отримується через формування модульної суми, що складається з представлень b та q. Сума виконується в замкненій і скінченній модульній числовій системі з основою m_n. Таким чином, q відповідає частці лише тоді, коли ця частка має ціле значення.

Код представлень залишків числа складається з багатьох цифр, де $A = (a_1, a_2, \dots, a_n)$. Кожна така цифра відповідає різним простим базисам. Числова система – це система за модулем m, де $\mathrm{m} = \prod_{\mathrm{i}=1}^{\mathrm{n}} \mathrm{m_i}$.

Ділення двох чисел в залишковому коді може бути представлено через систему конгруенцій. Рішення $Q = (q_1, q_2, \dots, q_n)$ має задовільнити всі конгруентні відносини системи. Нульова цифра в дільнику $B = (b_1, b_2, \dots, b_n)$ означає, що B та m не є взаємнопростими, тому мультиплікативної інверсії B не існує.

$$Q * B \not\equiv A \bmod m$$

У випадку, коли $b_i = 0$ та $a_1 = 0$, правильно буде записати конгруенцію так [39,40,41]:

$$\frac{QB}{m_i} \equiv \frac{A}{m_i} \bmod \frac{m}{m_i}$$

У першому розділі було проведено аналіз сучасних підходів до забезпечення криптографічної безпеки та детально розглянуто фундаментальні принципи функціонування протоколів квантового розподілу ключів. Показано, що квантові методи захисту інформації, зокрема протокол BB84, забезпечують високий рівень безпеки завдяки використанню фізичних законів квантової механіки, однак залишаються чутливими до низки практичних обмежень та атак, спрямованих як на квантовий, так і на класичний канали. Додатково було досліджено математичний апарат системи залишкових класів, який демонструє потенціал для оптимізації процесів обробки квантових ключів та підвищення стійкості протоколів QKD до окремих типів загроз. Проведений аналіз створює необхідне теоретичне підґрунтя для подальшого розроблення модифікованих методів формування квантового ключа на основі поєднання BB84 та СЗК, що розглядаються у наступних розділах [14,15].

2 МЕТОД МОДИФІКАЦІЇ ПРОТОКОЛУ BB84 З ВИКОРИСТАННЯМ СЗК

2.1 Інтеграція СЗК у протокол BB84

Поєднання СЗК з протоколом BB84 ґрунтується на використанні математичних властивостей модульної арифметики для розширення або оптимізації процесу генерації, передачі чи обробки криптографічного ключа в квантовому протоколі. У класичному протоколі BB84 ключ генерується у двійковій системі через кодування квантових станів фотонів, а СЗК дозволяє перейти від двійкової системи до системи з більшим базисом (наприклад, mod4: 0, 1, 2, 3 або mod8: 0, 1, ..., 7). Існує кілька способів реалізації, основними ідеями яких є:

1) розширення кодування інформації, де СЗК використовується для кодування більшої кількості станів у квантових частинках (фотонах), що дозволяє передавати більше бітів за один квантовий стан;

2) оптимізація постобробкиде СЗК застосовуватиметься для трансформації секретного двійкового ключа в компактніший або адаптивний формат, сумісний із системами, що базуються на модульній арифметиці;

3) підвищення гнучкості протоколу, оскільки СЗК надає додаткової можливості адаптації BB84 до різних криптографічних задач, використовуючи багатозначну логіку залишків.

Існує кілька способів реалізації, які залежать від етапу введення СЗК – у квантовій частині (генерація, передача) чи в класичній (постобробка). Наприклад, у квантовій частині СЗК замінює біти на залишки певного модуля, що кодуються через різні стани фотонів, або у постобробці СЗК може використовуватися для стиснення та перетворення двійкового ключа в залишок за модулем, адаптуючи його до подальшого використання. Порівняння підходів наведено в таблиці 2.1 [18,42].

Таблиця 2.1 – Порівняння підходів

Аспект	СЗК у кодуванні станів	СЗК у постобробці
Етап введення	Генерація, передача	Постобробка
Пропускна здатність	Збільшується (2+ бітів)	Залишається 1 біт
Складність апаратури	Висока (більше станів)	Низька (без змін у квантовій частині)
Сумісність	Потребує адаптації	Висока з модульними системами
Гнучкість	Обмежена апаратурою	Висока (легко змінювати модуль)

Введення СЗК в протокол BB84 принесе низку переваг:

1) збільшення пропускної здатності ключа. У класичному протоколі BB84 один фотон передає один біт інформації. Якщо ввести СЗК, кожен фотон може кодувати декілька бітів. Наприклад, послідовність із 4 фотонів у протоколі BB84 передає 4 біти (наприклад 0110), а при введені СЗК, якщо взяти модуль 4 – 8 бітів (2, 1, 3, 0 – 10, 01, 11, 00 у двійковій системі). Таким чином, підвищується ефективність використання квантового каналу, що особливо корисно при обмеженій кількості фотонів чи високій вартості передачі;

2) простота інтеграції з модульними криптосистемами. Оскільки багато сучасних криптографічних алгоритмів (як RSA, алгоритми на основі КТЗ) використовують модульну арифметику, то ключ у форматі СЗК може безпосередньо застосовуватись у таких системах без додаткових перетворень;

3) можливість стиснення ключа (у постобробці). У постобробці довгий двійковий ключ можна стиснути в одне або кілька чисел у системі залишків, що зменшує кількість даних для зберігання чи передачі. Наприклад, секретний ключ довжиною в чотири біти 0110 у mod 4 стає 6 mod 4 = 2, де 6 – десятковий формат даного ключа. Тобто тепер, замість зберігання 4 чисел ключа можна зберегти єдиний залишок. Таким чином проводиться зменшується об’єм ключа із

збереженням його унікальності, що корисно для систем з обмеженими ресурсами;

4) підвищення гнучкості протоколу. СЗК дозволяє легко масштабувати протокол, змінюючи модуль залежно від потреб безпеки чи пропускної здатності, що забезпечує адаптивність до різних сценаріїв використання без зміни квантової основи протоколу;

5) збереження квантової безпеки. Незалежно від того, на якому етапі вводиться СЗК, ключові принципи безпеки протоколу BB84, такі як виявлення підслуховування через квантові помилки, зберігаються;

6) потенційна оптимізація корекції помилок. У постобробці СЗК може бути використано для кодування контрольних даних або створення залишків для перевірки, які допоможуть виявити та виправити помилки. Наприклад, двійковий ключ розбиватиметься на блоки, для кожного з яких обчислюватиметься залишок (контрольна сума) за модулем, що призведе до покращення стійкості до шумів у каналі передачі [43,44].

Проте також варто зазначити можливі недоліки застосування такого введення:

1) складність реалізації. У квантовій частині потрібне високоточне обладнання для розрізнення більшої кількості станів;

2) втрата інформації. У постобробці стиснення через СЗК може призвести до втрати частини даних, якщо не використовувати додаткові механізми, як, наприклад, розбиття на блоки;

3) обчислювальні витрати. Перетворення чи аналіз залишків може вимагати додаткових ресурсів, особливо для великих модулів.

2.2 Алгоритм передачі квантового ключа з використанням СЗК

Розглянемо кожен запропонований етап використання СЗК окремо. Процес введення СЗК для постобробки ключа показано на рисунку 2.1, а його

приклад – в таблиці 2.2. Розглянемо кожен запропонований етап використання СЗК окремо. Для виконання даного процесу виконуються наступні кроки:

1) Аліса та Боб домовляються про певний модуль по класичному каналу;

2) проводяться всі дії відповідно до виконання протоколу BB84;

3) після отримання спільного секретного ключа вони перетворюють його в десяткову систему числення та обчислюють за обговореним модулем.

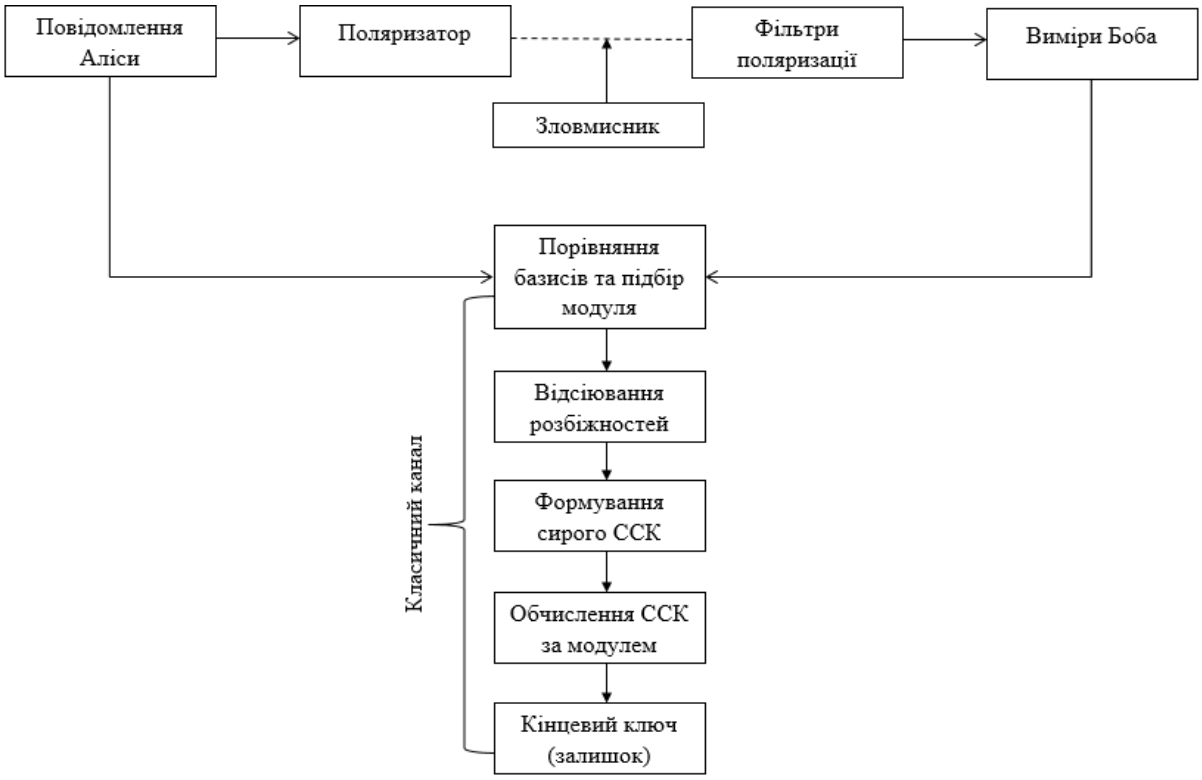

Рисунок 2.1 – Схема роботи протоколу BB84 при використанні СЗК в постобробці ключа

Таблиця 2.2 – Використання СЗК при постобробці ключа

1	2	3	4	5	6	7	8	9	10
Квантове представлення та виявлення	Бітова послідовність Аліси	1	0	1	0	0	1	0	1
	Поляризація послідовності Аліси	↑	↖	↗	↖	→	↑	↖	↗
	Фільтри поляризації Боба	↑	→	↗	→	↖	↑	↖	↗
	Значення послідовності Боба	1	0	1	0	0	1	0	1

Продовження таблиці 2.2

1	2	3	4	5	6	7	8	9	10
Обговорення по класичному каналу									
	Боб узгоджує свої базиси з Алісою	+	+	x	+	x	+	x	x
	Відсіювання розбіжностей	*	-	*	-	-	*	*	*
	Секретний ключ Аліси і Боба	1		1			1	0	1
Фінальний ключ:		11101 = 29 mod 17 = 12							

Отримане число і стане їх фінальним ключем для подальшого шифрування та розшифрування повідомлень.

При введенні СЗК для кодування квантових станів, на відміну від поляризації фотонів як бітів в двійковій системі числення, вони поляризуються як залишки певного числа. Нехай модулем було обрано число 28. Всі можливі залишки даного числа є від 0 до 27. Аліса та Боб обговорюють по класичному каналу, які залишки вони використовуватимуть і якій поляризації ті належатимуть, як показано в таблиці 2.3.

Таблиця 2.3 – Визначення даних для поляризації

mod 28		
Базис	Обране значення	Стан поляризації
Прямолінійний (+)	7	↑
	3	→
Діагональний (x)	10	↗
	25	↖

Після виконання даних дій Аліса та Боб виконують ті ж кроки, як у протоколі BB84. Після перевірки, за відсутності зловмисника, вони обчислюють отриманий спільний секретний ключ за модулем, і отримують фінальний ключ

для шифрування. Схему виконання даного процесу показано на рисунку 2.2., а приклад його виконання - в таблиці 2.4 [48].

В результаті виконання даної послідовності дій, Аліса та Боб отримали фінальний ключ через обчислення отриманого спільного секретного ключа 71072510 mod 28 = 26, який, за потреби, можна перевести даний ключ в двійкову систему числення, що дорівнюватиме 11010.

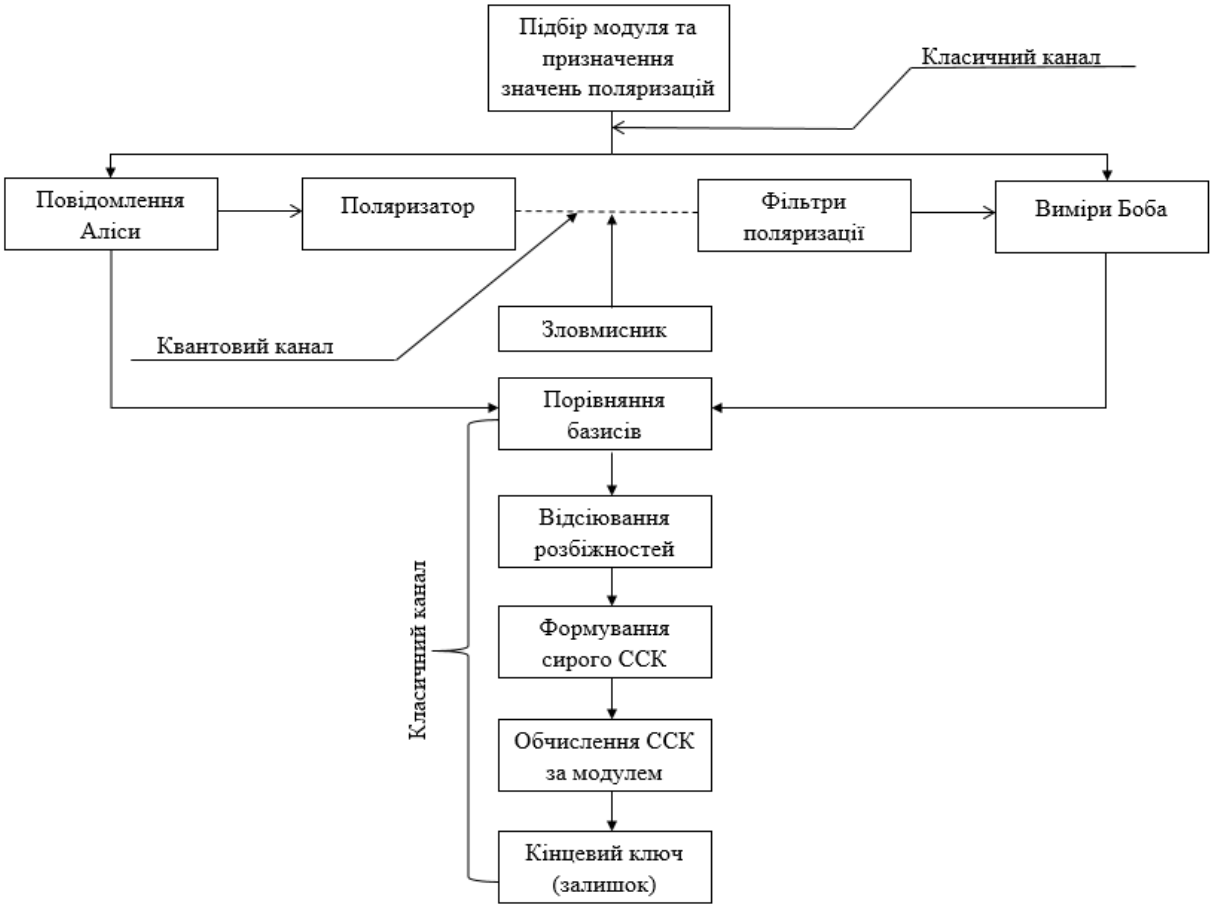

Рисунок 2.2 – Схема роботи протоколу BB84 при введені СЗК для кодування квантових станів

Таблиця 2.4 – Використання СЗК для кодування квантових станів

Квантове представлення та виявлення	Бітова послідовність Аліси	1	0	1	0	0	1	0	1
	Поляризація послідовності Аліси	↑	↖	↗	↖	→	↑	↖	↗
	Фільтри поляризації Боба	↑	→	↗	→	↖	↑	↖	↗
	Значення послідовності Боба	1	0	1	0	0	1	0	1
Обговорення по класичному каналу									
	Боб узгоджує свої базиси з Алісою	+	+	x	+	x	+	x	x
	Відсіювання розбіжностей	*	-	*	-	-	*	*	*
	Секретний ключ Аліси і Боба	1		1			1	0	1
Фінальний ключ:		11101 = 29 mod 17 = 12							

Після того, як було розглянуто принципи роботи модифікованого протоколу BB84 при введенні СЗК в різні етапи формування ключа, варто детальніше розглянути їх захищеність та ефективність.

2.3 Захищеність і ефективність модифікованого протоколу

Для повного розуміння впливу запропонованих модифікацій спочатку варто розглянути класичний протокол BB84, його захищеність і ефективність, а потім провести порівняння з модифікованими версіями.

Класичний протокол BB84 покладається на такі принципи квантової механіки, як принцип невизначеності Гейзенберга, теорему про заборону клонування та квантову заплутаність.

Потенційні ризики можуть бути такими:

1) шуми в квантовому каналі можуть підвищувати кількість помилок, що ускладнює розрізнення природних помилок від підслуховування;

2) атаки на апаратне забезпечення, як маніпуляція детекторами, можуть обійти квантову безпеку, але це залежить від реалізації а не протоколу;

3) класичний канал, що використовується для узгодження базисів, вразливий до атак «людина посередині» (англ. Man-In-The-Middle, MITM), якщо він не захищений додатковими криптографічними методами.

Для запобігання даних ризиків можна використати такі заходи безпеки:

1) використання автентифікованого класичного каналу для захисту від маніпуляцій;

2) регулярне тестування апаратного забезпечення для виявлення вразливостей;

3) застосування протоколів корекції помилок і згущення ключа для мінімізації впливу шумів.

Ефективність класичного протоколу BB84 полягає у наступному:

1) пропускна здатність. В даному протоколі кожен фотон передає один біт інформації. Однак, через відсіювання бітів, базиси яких не збіглися, ефективність пропускної здатності лежить в межах 50%. Наприклад із 100 надісланих фотонів секретний ключ може складатися приблизно з 50 бітів, а після корекції помилок та згущення – ще менше;

2) обчислювальна складність. Генерація бітів і базисів має складність O(n), де n – кількість фотонів. Корекція помилок і згущення ключа також ускладнюють обчислення, але це практично не впливає на сучасні системи. Таким чином, обчислювальна складність залишається низькою;

3) практична ефективність. Протокол BB84 є відносно простим у реалізації, ефективним у каналам з низьким рівнем шуму та добре інтегрується з класичними криптосистемами.

В свою чергу обмеженнями є низька пропускна здатність, висока вартість квантового обладнання та необхідність класичної постобробки, яка дає незначну затримку.

При введенні СЗК в постобробку ключа квантова безпека класичного протоколу BB84 зберігається. Також варто зазначити, що компрометація ключа

за допомогою СЗК є неможливою, оскільки перетворення відбувається після формування секретного ключа.

Потенційними ризиками можуть бути:

1) узгодження модуля по класичному каналу, який можуть перехопити або модифікувати, що вплине на фінальний ключ. Однак секретний ключ скомпрометовано не буде, потрібно буде лише провести повторне узгодження;

2) стиснення ключа до залишку може знизити унікальність, якщо, наприклад, вибрати малий модуль для великого ключа. Теоретично це полегшить атаку повного перебору, хоча при використанні правильного модуля такий ризик є мінімальним.

Додатковими заходами безпеки можуть бути:

1) використання кількох попарно простих модулів і КТЗ для збереження унікальності ключа;

2) застосування контрольних сум або хеш-функцій для перевірки цілісності ключа;

3) захист класичного каналу автентифікацією.

Ефективність введення СЗК для постобробки ключа полягає в наступному:

1) пропускна здатність залишається такою ж, як у класичному протоколі BB84, адже СЗК впливає лише на постобробку через стиснення ключа, що зменшує обсяг даних для зберігання та передачі;

2) обчислювальна складність. Перетворення двійкового ключа в десятковий та обчислення залишку має складність O(n). Використання кількох модулів і КТЗ приведе до підвищення складності до O(k log P), де k –це кількість модулів, а P – їх добуток;

3) практична ефективність. Стиснення ключа оптимізує використання ресурсів у системах з обмеженою пам’яттю чи пропускною здатністю. Також такий формат ключа полегшує його інтеграцію з модульними криптосистемами без потреби подальших перетворень, а його залишки можуть слугувати контрольними сумами для корекції помилок, підвищуючи стійкість до шумів.

Обмеженнями ж є невелика затримка під час підбору модуля та можлива втрата унікальності якщо він занадто малий.

При введенні СЗК у кодування квантових станів квантова безпека зберігається, а більша кількість станів не послаблює безпеку, оскільки кожен стан є унікальним та захищений законами квантової механіки.

Потенційними ризиками можуть бути:

1) збільшення кількості станів можу підвищити можливість виникнення помилки через складність розрізнення станів у шумному каналі чи при неточному обладнанні, що може привести до помилкового відкидання ключа;

2) узгодження залишків і їхньої відповідності через класичний канал може створити додаткову вразливість, хоча вона не впливає на квантову фазу.

Додатковими заходами безпеки можуть стати:

1) використання кількох базисів для залишків, що ускладнить їх перехоплення;

2) динамічна заміна модуля та відповідностей залишків для підвищення стійкості.

Ефективність введення СЗК в кодування станів полягає у наступному:

1) пропускна здатність значно збільшується, адже кожен фотон передає $\log_2 m$ бітів. Наприклад, для mod 4 передається 2 біти/фотон, а якщо використовувати mod 8 вона може збільшитися до 3 бітів/фотон;

2) обчислювальна складність, яка в загальному залишається незмінною від класичного протоколу BB84, хоча і потребує точнішого обладнання;

3) практична ефективність полягає у збільшенні пропускної здатності, що знижує кількість фотонів і час передачі, що можна використати для каналів із високими втратами чи вартістю. Гнучкість модуля дозволить адаптувати протокол до різних сценаріїв без зміни апаратної бази.

Обмеженнями такого підходу можуть бути:

1) необхідність складнішого обладнання для розрізнення більшої кількості станів, що може підвищити вартість і знизити практичність;

2) вища кількість помилок через складність розрізнення станів знижує частку успішно переданих фотонів;

3) висока апаратна складність для великих модулів.

Для кращого розуміння захищеності та ефективності класичного протоколу BB84 та запропонованих методів його модифікації було створено порівняльну таблицю 2.5 [17,18,19,20].

Таблиця 2.5 – Точне порівняння аспектів класичного протоколу BB84, та його модифікацій

Аспект	Класичний BB84	СЗК у постобробці	СЗК у кодуванні станів
1	2	3	4
Захищеність	Висока	Висока, ризик через узгодження модуля	Висока, ризик через шум і узгодження
Виявлення підслуховування	Через помилки	Незмінне	Незмінне, але вища можливість виявлення помилок
Вразливості	Шуми, атаки на обладнання, класичний канал	Втрата унікальності при малому модулі	Збільшення помилок через складність станів
Пропускна здатність	~0,5 біта/фотон	~0,5 біта/фотон	$\log_2 m$ бітів/фотон
Стиснення ключа	Відсутнє	Високе (зменшення обсягу даних)	Відсутнє (ключ у залишках)
Обчислювальна складність	Низька	Низька	Низька

Продовження таблиці 2.5

1	2	3	4
Апаратна складність	Середня (2 базиси, 4 стани)	Середня (як в класичному BB84)	Висока (m станів, точніше обладнання)
Гнучкість	Низька (фіксоване кодування)	Висока (зміна модуля)	Висока (зміна модуля)
Практичність	Висока	Висока (проста інтеграція)	Середня (залежить від апаратних можливостей)

Також не варто забувати про вже існуючі модифікації протоколу BB84. Для повного розуміння запропонованих у даній роботі модифікацій, варто провести їх порівняльний аналіз з протоколами B92, E91 та SARG04. У таблиці 2.6 наведено аспекти даних протоколів [17,19,20,28,29].

Таблиця 2.6 – Аспекти існуючих модифікацій протоколу BB84

Аспект	B92	E91	SARG04
1	2	3	4
Захищеність	Середня, менше ніж у BB84 (менше станів – легше атакувати)	Висока (ґрунтується на квантових заплутаних станах та тестах Белла)	Вища за BB84 у випадку фотонних атак
Виявлення підслуховування	Менш ефективне, ніж у BB84	Дуже сильне (кореляції заплутаних пар легко виявляють втручання)	Вища стійкість до атак на багаторазове використання

Продовження таблиці 2.6

1	2	3	4
Вразливості	Вразливий до оптимальних атак при шумі	Чутливий до недосконалих детекторів	Менш стійкий до шуму, ніж E91,але краще, ніж B92
Пропускна здатність	Вища, ніж у BB84 (менше станів)	Нижча (через складність генерації заплутаних станів)	Середня
Стиснення ключа	Не передбачено	Обмежене	Неявна (ефективність трохи знижена)
Обчислювальна складність	Низька	Висока (потрібна обробка заплутаних станів та тестів Белла)	Середня
Апаратна складність	Низька (2 стани замість 4)	Висока (потрібні джерела заплутаних пар)	Середня (схожа на BB84, але з модифікаціями)
Гнучкість	Низька (обмежений набір станів)	Висока (може застосовуватись для багато-користувацьких сценаріїв)	Середня
Практичність	Використовується рідко через слабшу безпеку	Обмежено практичний (дорога і складна апаратура)	Практичний компроміс між BB84 і B92

У порівнянні з класичними протоколами B92, E91 та SARG04, модифікований BB84 із використанням СЗК демонструє підвищену стійкість до підслуховування та помилок. Використання СЗК у кодуванні станів посилює захищеність на фізичному рівні, тоді як у постобробці забезпечує ефективну корекцію та стиснення ключа без значного ускладнення апаратної частини. Хоча E91 залишається найзахищенішим у теорії, його практична реалізація складна; B92 — простий, але менш безпечний; SARG04 забезпечує баланс між безпекою та практичністю. Модифікований BB84 з СЗК поєднує високу безпеку та відносну практичність, що робить його перспективним для сучасних систем квантового розподілу ключів.

Отже, можна зробити висновок, що кожен протокол має місце бути, залежно від потреби. Класичний прокол BB84 може забезпечити простоту та стандартизацію, при введенні СЗК у постобробку робиться акцент на компактності ключа, а при СЗК у кодуванні станів отримується максимальна пропускна здатність.

3 РЕАЛІЗАЦІЯ ТА АНАЛІЗ МОДИФІКОВАНОГО ПРОТОКОЛУ BB84

На основі запропонованого в другому розділі методу було розроблено програмну реалізацію, що дозволяє практично оцінити ефективність інтеграції системи залишкових класів у процес формування квантового ключа за протоколом BB84.

3.1 Програмна реалізація методів

Програмну реалізацію написано на мові Python. Обрана мова програмування має такі переваги, як простий і зрозумілий синтаксис, підтримка об’єктно-орієнтованого програмування; наявність вбудованих бібліотек, висока портативність завдяки кросплатформності.

Було програмно реалізовано модифікований протокол BB84 із використанням СЗК у двох підходах: кодуванні квантових станів та постобробці ключа. Основна мета - демонстрація розширення можливостей протоколу BB84 через використання СЗК для підвищення пропускної здатності або оптимізації ключа.

Завдання реалізації полягає в наступному:

1) розробка алгоритмів генерації залишків і двійкових бітів для двох підходів із підтримкою довільних модулів;

2) забезпечення введення користувацьких параметрів через графічний інтерфейс із перевіркою коректності даних;

3) реалізація кодування залишків у поляризацію фотонів (для СЗК у кодуванні) та обробку двійкових бітів (для СЗК у постобробці);

4) створення гнучкої постобробки ключа з можливістю використання одного або кількох модулів;

5) розробка адаптивного відображення результатів, де розмір вікна залежить від найдовшого рядка.

Основними етапами роботи програми є:

1) введення користувацьких параметрів, а саме: довжина ключа, модуль, стани поляризації (для кодування) або діапазон модулів (для постобробки);

2) генерація початкових даних Аліси, а саме: послідовність залишків (для кодування) або двійкових бітів (для постобробки) та базисів;

3) кодування у квантові стани через поляризацію фотонів;

4) симуляція вимірювання Бобом із випадковими базисами та формування секретного ключа;

5) постобробка ключа: конкатенація залишків у десяткове число та взяття залишку за модулем (або модулями);

6) відображення результатів у графічному інтерфейсі з адаптивним розміром вікна.

При розробці програмної реалізації було використано наступні бібліотеки:

1) import random, яка використовується для генерації псевдо-випадкових чисел;

2) import tkinter as tk – використовується для введення всіх класів, фукцій та змінних, необхідних для створення графічного користувацького інтерфейсу (Graphical User Interface - GUI), а приставка «as tk» - для спрощеного використання. Наприклад, замість того, щоб прописувати tkinter.Toplevel(), тепер можна використовувати коротше tk.Toplevel();

3) from tkinter import scrolledtext, messagebox – використовується для імпорту лише певних компонентів з бібліотеки tkinter. Тут scrolledtext відповідає за автоматичне додавання прокручування сторінки для полегшення роботи з великою кількістю тексту; а messagebox – за виведення різних повідомлень у стандартних вікнах (наприклад, інформація, повідомлення, помилки і т.д.).

Далі буде розглянуто компоненти коду, що забезпечують модульність та гнучкість.

Клас BB84_SZK (для СЗК у кодуванні станів), реалізує генерацію послідовності залишків, їх перетворення у поляризації та визначення базисів вимірювання. Детальніше кодування даного класу можна розглянути на рисунку 3.1.

```python
class BB84_SZK:
    def __init__(self, length, modulus, remainder_to_polarization):
        self.length = length
        self.modulus = modulus
        self.remainder_to_polarization = remainder_to_polarization
        self.allowed_remainders = list(remainder_to_polarization.keys())
        self.sequence = [random.choice(self.allowed_remainders) for _ in range(length)]
        self.bases = self.assign_bases()
        self.states = self.encode()

    def encode(self):
        return [self.remainder_to_polarization[val] for val in self.sequence]

    def assign_bases(self):
        bases = []
        for val in self.sequence:
            polarization = self.remainder_to_polarization[val]
            if polarization in ["0°", "90°"]:
                bases.append("+")
            else:
                bases.append("×")
        return bases
```

Рисунок 3.1 – Кодування класу BB84_SZK

Тут, у конструкторі задаються довжина послідовності, модуль, словник відповідностей «залишок → поляризація», а також генерується послідовність допустимих залишків. Базиси та стани формуються автоматично під час ініціалізації.

Метод **encode()** перетворює кожен залишок у відповідну поляризацію згідно зі словником, а метод **assign_bases()** визначає базис («+» або «×») на підставі типу поляризації (прямолінійна або діагональна).

Функція bb84_classic (для СЗК у постобробці), яка проводить генерацію двійкових даних Аліси, та модуляцію вимірювання Боба та повертає усі проміжні результати (біти, базиси, поляризацію, секретний ключ). Кодування даної функції показано на рисунку 3.2:

```python
def bb84_classic(length):
    alice_bits = [random.randint(0, 1) for _ in range(length)]
    alice_bases = [random.choice(["+", "×"]) for _ in range(length)]
    alice_polarization = []
    for bit, base in zip(alice_bits, alice_bases):
        if base == "+":
            alice_polarization.append("0°" if bit == 0 else "90°")
        else:
            alice_polarization.append("45°" if bit == 0 else "135°")

    bob_bases = [random.choice(["+", "×"]) for _ in range(length)]
    bob_polarization = []
    bob_bits = []
    for bit, a_base, b_base in zip(alice_bits, alice_bases, bob_bases):
        if a_base == b_base:
            bob_bits.append(bit)
            bob_polarization.append("0°" if bit == 0 and b_base == "+" else "90°" if bit == 1 and b_base == "+" else "45°" if bit == 0 else "135°")
        else:
            rand_bit = random.randint(0, 1)
            bob_bits.append(rand_bit)
            bob_polarization.append("0°" if rand_bit == 0 and b_base == "+" else "90°" if rand_bit == 1 and b_base == "+" else "45°" if rand_bit == 0 else "135°")

    raw_key = [a for a, ab, bb in zip(alice_bits, alice_bases, bob_bases) if ab == bb]
    return alice_bits, alice_bases, alice_polarization, bob_bases, bob_polarization, bob_bits, raw_key
```

Рисунок 3.2 – Кодування функції bb84_classic

За генерацію бітів та базисів Аліси відповідають alice_bits = [random.randint(0, 1) for _ in range(length)], що генерує випадкові біти: та alice_bases = [random.choice(["+", "×"]) for _ in range(length)], що випадково вибирає базис для кожного біта.

За поляризацію бітів Аліси відповідає alice_polarization = [] та функція for, яка присвоює відповідну поляризації відповідно до базиса.

За вибір базисів Боба відповідає bob_bases = [random.choice(["+", "×"]) for _ in range(length)], де для кожного біта ін вибирає випадковий базис, не знаючи вибору Аліси заздалегідь.

За імітацію вимірювання Боба відповідають bob_polarization = [], bob_bits = [] та функцію for, де, якщо базиси Аліси та Боба співпали, то Боб точно відновлює правильний біт та відповідну поляризацію, а якщо вони різні, то Боб отримує випадковий результат (0 або 1).

За формування сирого ключа відповідає raw_key = [a for a, ab, bb in zip(alice_bits, alice_bases, bob_bases) if ab == bb], де raw_key – цу частина бітів, які співпали, а return alice_bits, alice_bases, alice_polarization, bob_bases, bob_polarization, bob_bits, raw_key повертає всі вищевказані значення.

Функція simulate_bob (для СЗК у кодуванні), яка проводить моделювання вимірювання Боба для багатозначних залишків, показано на рисунку 3.3.

```
def simulate_bob(alice):
    bob_bases = [random.choice(["+", "×"]) for _ in range(alice.length)]
    bob_polarization = []
    bob_bits = []
    for val, a_base, b_base, state in zip(alice.sequence, alice.bases, bob_bases, alice.states):
        if a_base == b_base:
            bob_bits.append(val)
            bob_polarization.append(state)
        else:
            rand_val = random.randint(0, alice.modulus-1)
            bob_bits.append(rand_val)
            bob_polarization.append(random.choice(["0°", "90°", "45°", "135°"]))
    raw_key = [a for a, ab, bb in zip(alice.sequence, alice.bases, bob_bases) if ab == bb]
    return bob_bases, bob_polarization, bob_bits, raw_key
```

Рисунок 3.3 – Кодування функції simulate_bob

Тут, для кожного елемента обирається випадковий базис. Якщо базиси співпадають, Боб правильно відновлює залишок; у протилежному випадку

отримує випадкове значення. «Сирий ключ» формується аналогічно класичному BB84 - лише зі збігів базисів.

Функція szk_postprocess у кодуванні станів перетворює ключ у десяткове число та обчислює залишок(и) за модулем(ями) як показано на рисунку 3.4.

```
def szk_postprocess(key, modulus):
    decimal_str = ''.join(map(str, key))
    decimal = int(decimal_str) if decimal_str else 0
    final_key = decimal % modulus
    return decimal, final_key
```

Рисунок 3.4 – Кодування функції szk_postprocess для кодування станів

Тут функціонал коду наступний:

1) binary = ''.join(map(str, key)), що перетворює список бітів (key) у рядок;

2) decimal = int(binary, 2) if binary else 0 перетворює бінарний рядок у десяткове число, а якщо ключ порожній, то повертає 0;

3) final_key = [decimal % mod for mod in modulus_list] if modulus_list else [decimal] обчислює залишки від ділення числа decimal на всі модулі з modulus_list, а якщо він порожній, то повертає саме число;

4) return decimal, final_key повертає десяткове представлення ключа та список залишків за модулями.

У постобробці функція szk_postprocess підтримує список модулів, повертаючи кортеж залишків, як показано на рисунку 3.5.

```
def szk_postprocess(key, modulus_list):
    binary = ''.join(map(str, key))
    decimal = int(binary, 2) if binary else 0
    final_key = [decimal % mod for mod in modulus_list] if modulus_list else [decimal]
    return decimal, final_key
```

Рисунок 3.5 – Кодування функції szk_postprocess для постобробки

Тобто, сирий ключ інтерпретується як десяткове число, після чого обчислюється залишок за одним модулем. Функція повертає десяткове представлення ключа та його залишок.

Після опису окремих компонентів протоколу — генерації бітів та базисів, кодування станів, моделювання вимірювань Боба та постобробки ключа — логічним кроком є об’єднання їх у єдиний робочий цикл. Саме для цього було створено run_simulation та GUI, який здійснює реалізацію графічного інтерфейсу для введення параметрів і запуску симуляції з перевіркою коректності даних.

Розглянемо функцію run_simulation для СЗК у кодуванні станів за допомогою рисунку 3.6.

```
def run_simulation():
    try:
        length = int(length_entry.get())
        modulus = int(modulus_entry.get())
        if modulus < 2:
            raise ValueError("Модуль має бути >= 2")

        remainder_to_polarization = {
            int(horiz_remainder.get()): "0°",
            int(vert_remainder.get()): "90°",
            int(diag_remainder.get()): "45°",
            int(antidiag_remainder.get()): "135°"
        }
        if any(r < 0 or r >= modulus for r in remainder_to_polarization.keys()):
            raise ValueError("Введіть коректні залишки (від 0 до modulus-1)")
        if len(set(remainder_to_polarization.keys())) != 4:
            raise ValueError("Залишки повинні бути унікальними")

        alice = BB84_SZK(length, modulus, remainder_to_polarization)
        bob_bases, bob_pol, bob_bits, raw_key = simulate_bob(alice)
        decimal_key, final_key = szk_postprocess(raw_key, modulus)

        result_text = (f"Початковий рядок значень Аліси: {alice.sequence}\n"
                       f"Базис Аліси: {alice.bases}\n"
                       f"Поляризація Аліси: {alice.states}\n"
                       f"Базис Боба: {bob_bases}\n"
                       f"Поляризація Боба: {bob_pol}\n"
                       f"Отримані значення Боба: {bob_bits}\n"
                       f"Секретний ключ: {raw_key}\n"
                       f"Переведення в десяткове: {decimal_key}\n"
                       f"Ключ у СЗК (mod {modulus}): {final_key}")
        show_results("Результати симуляції (СЗК у кодуванні)", result_text)
    except ValueError as e:
        messagebox.showerror("Помилка", str(e))
```

Рисунок 3.6 – Кодування функції run_simulation для СЗК у кодуванні станів

Дана функція відповідає за:

1) отримання параметрів від користувача з графічного інтерфейсу;

2) перевірку коректності введених даних: модуль ≥ 2, залишки в межах допустимого діапазону, унікальність залишків для кожної поляризації;

3) формування відображення залишок → поляризація;

4) створення об’єкта Аліси за допомогою класу BB84_SZK, що генерує початкову послідовність бітів, базиси та поляризації;

5) виклик функції simulate_bob(), яка моделює дії Боба (вибір базисів, отримані поляризації та бітові значення);

6) використання функції szk_postprocess() для перетворення згенерованого «сирого» ключа у десятковий формат і подальшого отримання ключа в системі залишкових класів;

7) формування підсумкового тексту з усіма етапами симуляції;

8) виклик вікна show_results() для відображення результатів симуляції у зручному вигляді;

9) обробку можливих помилок введення та повідомлення користувача через messagebox.showerror().

Варто також переглянути відмінності функції run_simulation для СЗК у постобробці, яку показано на рисунку 3.7.

```
def run_simulation():
    try:
        length = int(length_entry.get())
        mod_min = int(mod_min_entry.get())
        mod_max = int(mod_max_entry.get())
        num_mods = int(num_mods_entry.get())
        if mod_min < 2 or mod_max < mod_min:
            raise ValueError("Діапазон модулів має бути >= 2 і max >= min")
        if num_mods < 1:
            raise ValueError("Кількість модулів має бути >= 1")

        modulus_list = [random.randint(mod_min, mod_max) for _ in range(num_mods)]
        alice_bits, alice_bases, alice_pol, bob_bases, bob_pol, bob_bits, raw_key = bb84_classic(length)
        decimal_key, final_key = szk_postprocess(raw_key, modulus_list)

        result_text = (f"Початковий рядок бітів Аліси: {alice_bits}\n"
                       f"Базис Аліси: {alice_bases}\n"
                       f"Поляризація Аліси: {alice_pol}\n"
                       f"Базис Боба: {bob_bases}\n"
                       f"Поляризація Боба: {bob_pol}\n"
                       f"Отримані біти Боба: {bob_bits}\n"
                       f"Секретний ключ: {raw_key}\n"
                       f"Переведення в десяткове: {decimal_key}\n"
                       f"Кількість модулів: {num_mods}, модулі: {modulus_list}\n"
                       f"Ключ у СЗК: {tuple(final_key) if num_mods > 1 else final_key[0]}")
        show_results("Результати симуляції (СЗК у постобробці)", result_text)
    except ValueError as e:
        messagebox.showerror("Помилка", str(e))
```

Рисунок 3.7 – Кодування функції run_simulation для СЗК у постобробці

У даному випадку відбувається наступне:

1) зчитування та перевірка вхідних даних**.** Отримуються значення length (довжина бітової послідовності), mod_min (мінімальне значення модуля), mod_max (максимальне значення модуля) та num_mods (кількість модулів). Також виконуються перевірки: мінімальний модуль ≥ 2, верхня межа не менша за нижню, кількість модулів ≥ 1;

2) генерація модулів. Формується список випадкових модулів (modulus_list) у заданому діапазоні розміром num_mods;

3) симуляція протоколу bb84. Викликається функція bb84_classic(length), яка моделює класичний BB84 (без СЗК на етапі кодування), а на виході отримуються біти та базиси Аліси, поляризації, вибір базисів Боба, його результати та «сирий» ключ;

4) постобробка ключа. «Сирий» ключ переводиться в десяткове число, а ділі обчислюється ключ у системі залишкових класів за всіма модулями зі списку modulus_list;

5) формування результату. Створюється текстовий звіт, який містить усі етапи симуляції: біти та базиси Аліси й Боба, поляризації, отримані біти, «сирий» ключ, його десяткове подання, вибрані модулі та фінальний ключ у СЗК;

6) відображення результату. Викликається функція show_results(...) для показу результатів у окремому вікні, а у випадку помилкових даних (наприклад, некоректні модулі) виводиться повідомлення про помилку через messagebox.showerror.

Детальний розгляд функції show_results та блоку кодування графічного інтерфейсу не вважається доцільним, оскільки вони не впливають на саму програмну реалізацію запропонованих модифікацій.

Для реалізації СЗК у кодуванні станів програма симулює модифікований протокол BB84, де кожен фотон кодує залишок у системі mod N (N задається користувачем) через чотири стани поляризації.

На рисунку 3.8 показано основне вікно програми, що призначене для налаштування змінних. В даному випадку це довжина ключа Аліси, бажаний модуль та залишки в межах (0, N-1) для кожного стану поляризації.

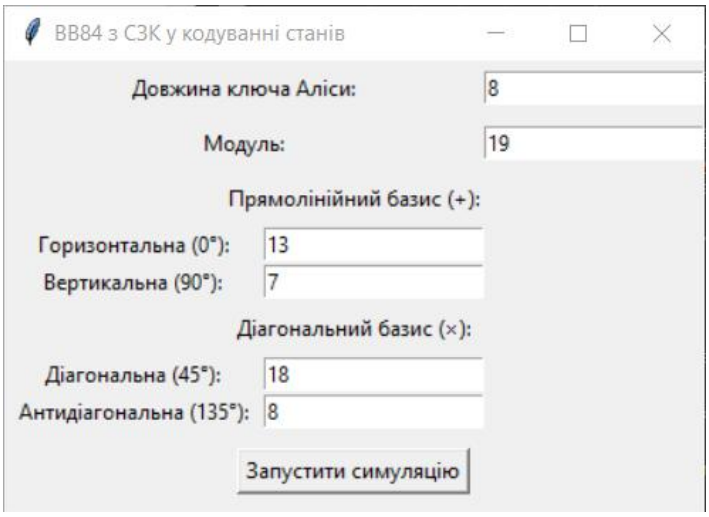

Рисунок 3.8 – Головне вікно програми

Після натискання на кнопку «Запустити симуляцію» програма виводить діалогове вікно з початковим рядком значень Аліси, базиси та поляризації Аліси та Боба, отримані значення Боба, обчислений секретний ключ та його підготовка до обчислення за заданим модулем, що можна побачити на рисунку 3.9.

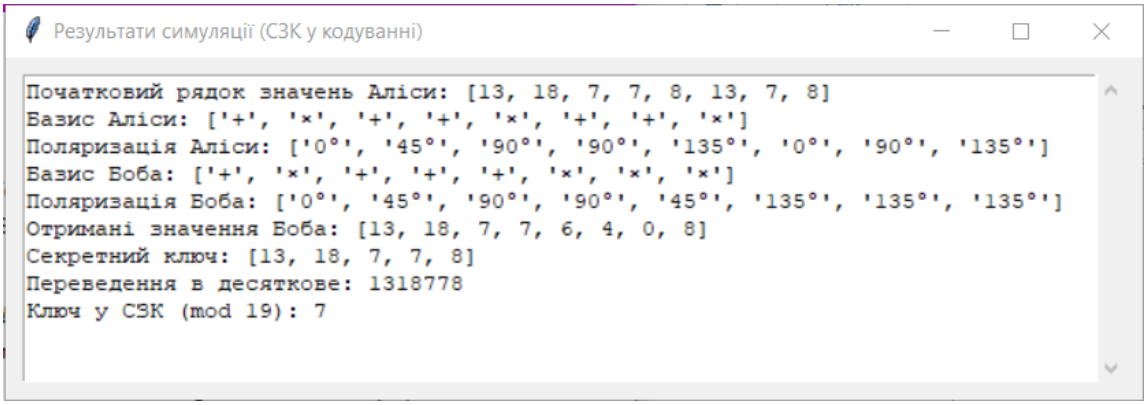

Рисунок 3.9 - Вивід результатів роботи програми

Код програмної реалізації даного методу введення СЗК в протокол BB84 можна переглянути в додатку А.

Для реалізації СЗК у постобробці програма моделює класичний протокол BB84 із двійковими бітами, а потім застосовує СЗК із кількома модулями для отримання фінального ключа у вигляді залишків.

На рисунку 3.10 показано основне вікно програми, що призначене для задання змінних. В даному випадку, це довжина ключа Аліси, діапазон модулів та їх кількість.

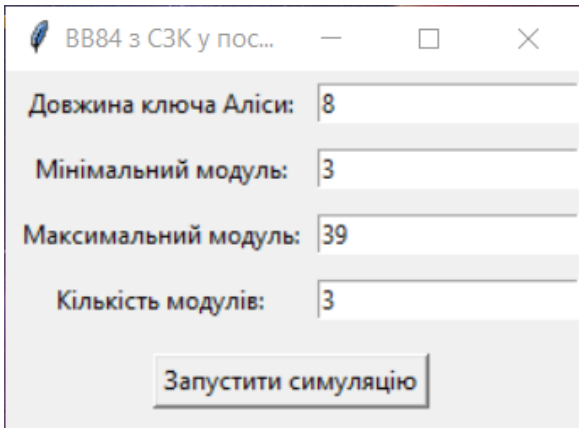

Рисунок 3.10 – Головне вікно програми

Після натискання кнопки «Запустити симуляцію» програма виводить діалогове вікно з початковим рядком бітів Аліси, базиси та поляризації Аліси та Боба, отримані біти Боба, секретний ключ, переведення його у десяткове число, обрані модулі та фінальний ключ у вигляді залишків, що можна побачити на рисунку 3.11.

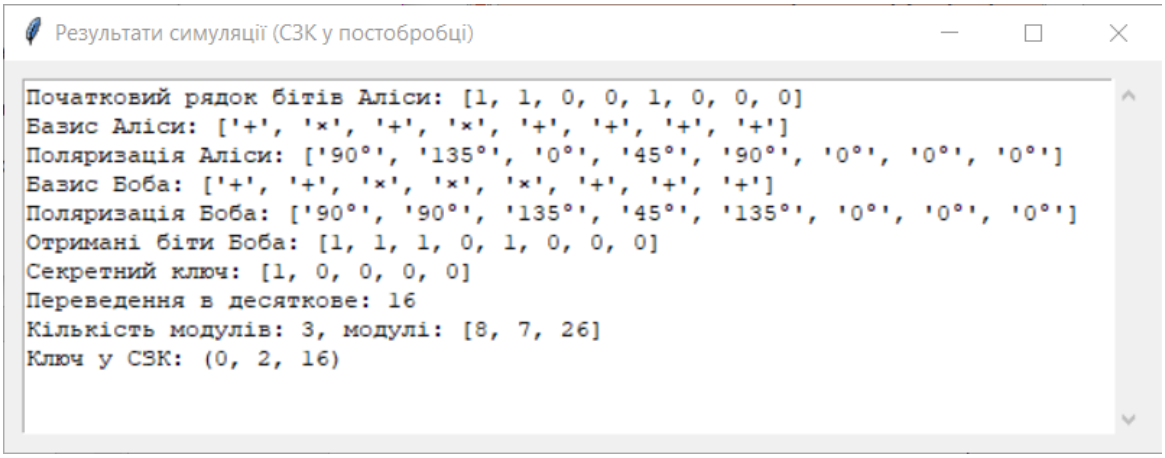

Рисунок 3.11 - Вивід результатів роботи програми

Код програмної реалізації даного методу введення СЗК в протокол BB84 наведено в додатку Б.

3.2 Дослідження стійкості модифікованих протоколів до атак

Квантовий розподіл ключа вважається процесом генерації та обміну ключами, які є безпечними, оскільки вони базуються на законах квантової фізики. Творцями першого протоколу квантового розподілу ключів, а саме BB84, було доведено його захист проти певних типів атак. З тих пір було запропоновано декілька покращень безпеки. Проблеми безпеки квантового розподілу ключів є дуже важливою і має бути дослідженою. Хоча протокол створений незламним, є деякі можливі загрози в теорії і на практиці.

Розглянемо атаки, які найчастіше зустрічаються при роботі з протоколом квантового розподілу ключів BB84:

1) intercept-resend, або перехоплення і повторне відправлення, також можна назвати атакою «підслуховування», яку вже було розглянуто в пункті 1.2. Тут, зловмисник Єва перехоплює кожен поляризований фотон, надісланий Алісою Бобу, вимірює їх у вибраному базисі, і тоді відправляє власні поляризовані фотони, що базуються на її вимірюваннях. Оскільки вона не може ідеально відтворити невідомий квантовий стан, цей процес спотворює потік поляризованих фотонів, знижуючи можливу кількість співпадінь між вимірами Аліси і Боба, тим самим збільшуючи кількість помилок;

2) photon-Number Splitting (PNS) – це атака на багатофотонні імпульси. Вона використовує слабку когерентність в однофотонних джерелах та недосконалості фотонних детекторів. Використовуючи ймовірність використання багатофотонних імпульсів, зловмисник може перехопити та виміряти стан фотонів без значного впливу на квантові кореляції. В результаті, атака може бути непомітною, оскільки вона не викликає значних помилок під час передачі квантів;

3) trojan-horse, або Троянський кінь, зловмисник вводить додаткові фотони або сигнали в пристрої квантової комунікації, як поляризатор Аліси або зчитувач Боба. Атака використовує будь-який ненавмисний «витік» або взаємодію квантового каналу з його оточенням, що дає зловмиснику можливість вилучати чутливу інформацію про квантову комунікацію без прямого впливу на квантові стани фотонів, що передаються;

4) man-in-the-Middle, або Людина-по-Середині, є менш поширеним типом атак, коли є автентифікований класичний канал. Проте, якщо даний канал, який використовується для порівняння базисів, погано захищений, то зловмисник Єва може «розмістити» себе між Алісою і Бобом. Тоді вона встановить окремі ключі для кожного з них, ефективно беручи контроль над комунікацією. Цей тип атак може бути застосованим як до квантового каналу, так і до класичного.

5) faked state attack, або атака «підробного стану». Тут Єва підготовлює та відправляє підроблені закодовані фотони, які повторюють очікувану поведінку справжніх сигналів. Вона може змусити детектори Боба отримати передбачувані результати вимірювання через акуратне формування цих станів. Цей контроль над результатами вимірювань дозволяє Єві вилучити ключову інформацію, утримуючи допустимий рівень помилок для Аліси і Боба;

6) time-shift attack, або атака «зсуву часу», використовує малі затримки і варіації ефективності в детекторах Боба, що дає Єві змогу затримати або прискорити час «прибуття» поляризованих фотонів так, що один з детекторів «зреагує» на фотони з більшою ймовірністю, ніж інші. Такий вибірковий тригер детектора може призвести до часткового витоку інформації про ключ.

Розглянемо рівень захисту протоколу BB84 та його існуючих модифікацій і таблиці 3.1 [42,43,44,45,46,47]:

Таблиця 3.1 – Рівень захисту протоколів BB84, B92, E91, SARG04

Атака	BB84	B92	E91	SARG04
1	2	3	4	5
Intercept-resend	Середня-висока. Атака створює ≈25% QBER, легко виявляється	Низька. Менше базисів>менше помилок>легше приховати	Висока. Порушення нерівностей негайно видає перехоплення	Середня-висока. Складніше провести атаку через специфіку процесу зсуву
Photon-number splitting (PNS)	Низька. Без станів-приманок класично вразливий	Низька. Дуже чутливий до даної атаки	Середня. Атака можлива, але кореляції ускладняються	Висока. Розроблений, як альтернатива BB84 саме проти PNS
Trojan-horse attack	Середня. Вимагає апаратних фільтрів	Низька. Немає вбудованих механізмів	Середня. Складніше через двосторонню структуру	Середня. Вимагає апаратного захисту
Man-in-The-Middle	Середня. Потрібна автентифікація класичного каналу	Низька. Без захисту канал легко підмінити	Висока. Тести роблять підміну помітною	Середня. Потребує автентифікації, як BB84

Продовження таблиці 3.1

1	2	3	4	5
Faked state attack	Середня. Можливі атаки на детектор	Низька.-середня. Малий набір станів спрощує підробку	Середня. Підробка може створити кореляції	Висока. Краще захищений від детекторних атак
Time-shift attack	Низька.-середня. Залежить від ефективності детекторів	Низька. Більш вразливий, ніж BB84	Середня. Статистика кореляцій швидко виявляє аномалії	Середня-висока. Більш стійкий завдяки модифікаціям

Також варто вказати і інші можливі атаки на протоколи квантового розподілу ключів:

1) detector blinding attack, або атака «засліплення детектора», полягає в тому, що зловмисник надсилає сильні оптичні сигнали, щоб перевантажити детектори системи квантового розподілу ключів, що призводить до їхньої несправності та до вимірювання некоректних квантових станів. Така маніпуляція не передбачає втручання в квантові стани, а радше використовує недосконалості компонентів детектора, що призводить до помилкових результатів у процесі обміну ключами;

2) channel tampering attack, або атака впливу на канал. Тут зловмисник може фізично порушити або змінити квантовий канал, наприклад через розрізання або заломлювання волокна. Хоча дана атака не спричиняє витоку інформації, вона компрометує доступність безпечного каналу;

3) electromagnetic leakage, або «електромагнітний витік», стає можливою, коли чутливі компоненти системи квантового розподілу ключів, як детектори,

генератори випадкових чисел, або високошвидкісна електроніка, ненавмисно випромінюють радіочастоти або електромагнітні сигнали, корельовані з їхніми внутрішніми сигналами. Зловмисник, який зчитує ці сигнали, може зробити висновок про важливі деталі процесу квантового формування ключа, як базові налаштування або події виявлення, тим самим підриваючи безпеку протоколу;

4) phase-remmaping attack. В деяких системах квантового розподілу ключів, модулятори фаз, які використовуються для кодування квантових станів, можуть бути недосконалими. В даній атаці, Єва використовує ці недосконалості через надсилання сигналів, які змушують модулятор видавати фази відмінні від задуманого значення. Через зміну даних фаз, Єва може корелювати її вимірювання з справжнім ключем без створення помітних помилок;

5) nonrandom-phase attack. Протоколи квантового розподілу ключів припускають що модуляція фаз є насправді випадковою. Проте, можуть з'явитися певні шаблони, якщо модулятор фаз має зміщення або використовує генератор псевдо-випадкових чисел. В даній атаці, Єва використовує дану передбачувану поведінку, щоб «вгадати» закодовану інформацію, тим самим компрометуючи безпеку протоколу;

6) detector-after-gate attack. В даній атаці, Єва використовує короткий період відновлення одразу після активного вікна затворного лавинного фотодіода. Хоча затвор закрито, діод ще не повністю погас і є чутливим до і залишається часково чутливим на декілька наносекунд, реєструючи фотони через залишкове зміщення або після-пульсації. В цей час, Єва відправляє ретельно підібрані світлові імпульси для тригеру вимірювання. Це дозволяє їй зміщувати результати вимірювань без створення великої кількості помилок, потенційно призводячи до витоку чутливої інформації;

7) double-click attack. В деяких реалізаціях систем квантового розподілу ключів, коли два детектора реєструють клік, тобто відбувається подвійний клік, відбувається випадкове присвоювання значень. Єва може використати таку поведінку через ретельно підібрані пульсації. Таким сином, вона може внести

упередження або отримати часткову інформацію про ключ, без суттєвого збільшення рівня помилок.

Всі вищевказані атаки можна поділити на дві групи, а саме: Main-channel attacks, де під атаку підпадає квантовий канал, і Side-channel attacks – атака на обладнання та фізичну реалізацію. Всі вище перераховані атаки було розділено на дані категорії в таблиці 3.2.

Таблиця 3.2 – Розподіл атак на категорії

Main-channel attacks	Side-channel attacks
Intercept-resend	Electromagnetic leakage
Photon Number Splitting (PNS)	Phase-remapping attack
Trojan-horse	Nonrandom-phase attack
Man-in-The-Middle	Detector-after-gate attack
Faked-state attack	Double-click attack
Time-shift attack	
Detector-blinding attack	
Channel tampering	

Традиційні протоколи квантового розподілу ключів базуються на довірених моделях пристроїв і мають певну кількість контрзаходів для запобігання різноманітним атакам. Наприклад, вразливості безпеки, які використовують у PNS, можна перекрити, використовуючи «протоколи-приманки». Атака перехоплення та повторного відправлення зазвичай визначається через аналіз кількості помилок і виправляється через надійні процедури покращення безпеки. На додачу, можна запобігти атаці «Людина-по-Середині» через використання автентифікації класичних каналів.

Щодо атак на бокові канали, можна використати декілька апаратних стратегій. Атака «Троянський кінь» блокуються оптичними ізоляторами і фільтрами довжини хвиль. Тимчасові затвори вводяться для забезпечення реакції детекторів на сигнали лише в певний період часу. Можна запобігти атакам

«засліплення детектора» через введення незалежного від приладу вимірювання квантового розподілу ключа, що ефективно зміщує перевірку безпеки з детекторів на підтвердження заплутаності між сторонами, шо спілкуються. На додачу, можна запобігти електромагнітним атакам на бокові канали через введення технік захисту та фільтрування на фізичному рівні, щоб «подавити» ненавмисний витік даних та електромагнітні випромінювання.

Проте, незважаючи на багатообіцяючі контрзаходи, які було розроблено для запобігання відомим атакам на протоколи квантового розподілу ключів, їх практичне введення може стикнутися з декількома викликами, що можуть знецінити дані рішення. Реальні системи практично ніколи не досягають ідеальної поведінки, яку припускають у перевірці на безпеку, що призводить до вразливостей навіть за присутності контрзаходів. Серед основних практичних викликів для систем квантового розподілу ключів є:

1) недосконалі однофотонні джерела. Більшість систем квантового розподілу ключів покладаються на слабкі когерентні імпульси, а не на «справжні» однофотонні джерела. Поки «протоколи-приманки» допомагають запобігти атакам PNS, вони потребують точного контролю інтенсивності імпульсів і фаз. Будь-які коливання і неточності можуть ненавмисно пропустити багатофотонне випромінювання, надаючи підслуховувачу доступ до каналу;

2) поведінка неідеального детектора. У теоретичних детекторах припускається рівномірна ефективність і точні часові характеристики, але на практиці вони можуть мати невідповідності ефективності, тимчасові неполадки та інші недосконалості. Навіть з тимчасовим затвором, де індикатори активні лише певний ліміт часу, незначні відмінності у реакціях детектора можуть бути використані для атаки «зсуву часу» або «підробного стану». Наприклад, у протоколі E91 Аліса кодує фотони в обраному базисі (діагонільному або прямолінійному), поки Боб вимірює дані кубіти, використовуючи базис, що обертається на $\pi/8$ оберту. За реалістичного сценарію, якщо вимірювальний пристрій Боба має недоліки або неправильне калібрування, закодовані фотони можуть зчитатися як шум. Проте підслуховувач Єва може використати ці

недосконалості через атаку «засліплення детектора», де використання лазеру високої інтенсивності дозволить Єві примусово перевести детектор Боба в інший робочий режим, тим самим порушуючи безпеку протоколу без виникнення помітних аномалій;

3) нехарактеризовані бокові канали. Багато контрзаходів припускають повну і точну характеризацію пристроїв QKD. Проте, практичні системи можуть мати нехарактеризовані або приховані бокові канали через специфіку виробництва, вплив навколишнього середовища або старіння компонентів. Наприклад, коли оптичні ізолятори і фільтри довжини хвилі допомагають заблокувати атаку «Троянський кінь», деякі недоліки або неочікувані шляхи витоку інформації можуть існувати.

4) проблеми з системною інтеграцією і синхронізацією. Протоколи квантового розподілу ключа припускають ідеальну синхронізацію між відправником і отримувачем, разом з бездоганною роботою класичних каналів автентифікації. Насправді, підтримка синхронізації на великих відстанях, калібрація «постійності» обладнання і забезпечення безпеки класичного каналу в реалістичних умовах представляють значні технічні складнощі, які можуть зменшити ефективність контрзаходів.

Ці практичні виклики показують, чому навіть акуратно підібрані контрзаходи можуть не спрацювати в реальних умовах. Вони підкреслюють потребу в постійних покращеннях у виробництві пристроїв, технік калібрації та, можливо і найбільш важливо, створення протоколів нового покоління, які мінімізують залежність від загальних припускань про роботу індивідуальних компонентів.

У свою чергу, модифіковані версії протоколів BB84, в яких використовується СЗК у постобробці ключа та кодуванні станів, змінюють структуру формування фінального ключа, що впливає на класичні та квантові атаки різних типів. Важливо зазначити, що СЗК не змінює квантової фізики передачі фотонів, але впливає на інформацію, яку потенційно може отримати зловмисник, та на складність відновлення ключа.

Розглянемо вплив використання СЗК у постобробці ключа на стійкість модифікованого протоколу BB84:

1) атаки intercept-resend, faked-state, man-in-the-middle. СЗК не впливає на квантову частину передачі, але підвищує стійкість фінального ключа, оскільки Єва отримує лише набір залишків, а не бітів і без повного набору модулів відновити ключ не можливо. Отже, покращення полягає в тому, що якщо Єва отримує лише частину інформації, вона не може повністю відтворити значення ключа. Також, модулі в СЗК можуть бути взаємно простими й великими, що підвищує криптографічну неповноту перехоплених даних. Отже, стійкість помірно підвищується через криптографічні властивості СЗК, хоча сам протокол фізично не змінюється;

2) атака PNS. СЗК не змінює фізичну чутливість BB84 до PNS, але при перехопленні частини фотонів, Єва бачить лише частину залишків. також, якщо розподіл модулів ускладнений, часткова інформація мало корисна. Покращенням є інформаційна стійкість проти часткового витоку. Недоліком в свою чергу є незмінна вразливість фізичного каналу, потрібно використовувати «стани-приманки»;

3) атаки Trojan-horse, time-shift, detector-blinding. Оскільки ці атаки працюють на рівні апаратури, СЗК не може запобігти проникненню сигналу чи зміщенню часу, але дозволяє додаткові перевірки зв’язності та узгодженості залишків та ускладнює точне відновлення корисної інформації зі зламаних детекторів. Це надає обмежений захист і доступна інформація зменшується;

4) атака channel-tampering. СЗК ніяк не вплине на захист від даної атаки;

5) атаки phase-remapping, nonrandom-phase. Фазові атаки змінюють кодування квантових станів. СЗК у постобробці не захищає квантову частину, але змінює форму корисних даних, та зменшує інформаційну цінність часткових вимірювань;

6) атака electromagnetic side-channel. СЗК надає додаткові перевірки постійності даних та реконструкцію повних значень з багатьох незалежних

залишків. Варто зазначити, що часткові витоки ЕМ-випромінювання не дають повного ключа;

7) атаки double-click, detector-after-gate. Детекторні атаки працюють на рівні системи виявлення. СЗК у постобробці ускладнює прямий зв'язок між детекторною подією та бітовим значенням та дозволяє кодувати усі події у високорозрядні залишкові значення. В результаті, інформативність детекторного витоку знижується, але не блокується.

Отже, при використанні СЗК у постобробці підвищується інформаційна стійкість та зменшується цінність часткового витоку, при чому не змінює фізичні властивості та вразливості квантового каналу.

У свою чергу, СЗК у кодуванні квантових станів є сильнішим варіантом, адже етап квантового кодування змінюється. Тут, квантовий стан кодує залишок а певним модулем, фотон може представляти більше, ніж один біт інформації та відповідність стану до залишку відома лише Алісі та Бобу. Це створює неоднозначність станів для Єви, навіть при успішному перехопленні фотонів. Розглянемо стійкість даної модифікації до різних типів атак:

1) атака intercept-resend. Єва не знає відповідності «поляризація – модуль - залишок». Тобто навіть отримавши стан, вона не знає модуль, чи було використано той самий набір модулів під час цієї передачі, та яке відображення станів на залишки. Це значно ускладнює проведення конкретного повторного надсилання;

2) атака PNS. Тут, на відміну BB84, один фотон не містить однозначної інформації – лише залишок, модуль може змінюватися і для відновлення значення необхідно мати повний набір залишків;

3) атаки Trojan-horse, time-shift, detector-blinding. Навіть якщо Єва отримає частину внутрішньої інформації (наприклад, залишковий модуль), то вона не отримає всього набору модулів, залишок не відновлює значення, а неправильний модуль знецінює отриману інформацію;

4) атака man-in-the-middle. Зміна кодування станів ускладнює підміну, оскільки Єва повинна знати точне значення всіх модулів, знати правила

кодування і замінювати фотони з узгодженими фазами. Навіть невелике відхилення приведе до виявлення помилки;

5) атаки phase-remapping, nonrandom-phase. Ці атаки намагаються дізнатися, які значення фаз відповідають яким символам. Тут фаза - це залишок за модулем, модулі змінюються і відповідність не є статичною. Отже, зловмиснику складно вибудувати кореляційну карту;

6) атаки detector-after-gate, double-click. СЗК не блокує детекторні атаки, але навіть при відомому результаті детектора значення ключа не відновлюється повністю, а статистичні зміни залишків виявляються при перевірці постійності;

7) атаки Атака electromagnetic leakage. Витік внутрішніх даних дає лише частину залишкових значень, а без повного набору модулів атака малоефективна.

Отже, СЗК у кодуванні станів змінює спосіб передачі квантової інформації, радикально ускладнює моделювання станів та робить атаки на квантовий канал і фазові атаки набагато складнішими.

Розглянемо підсумкову оцінку стійкості модифікованих протоколів BB84 у таблиці 3.3.

Таблиця 3.3 – Підсумкова оцінка стійкості модифікованих протоколів BB84

Категорія атак	Класичний BB84	BB84+СЗК у постобробці	BB84+СЗК у кодуванні станів
1	2	3	4
Main-channel	Середня	Середня-висока	Висока
PNS	Низька	Середня	Висока
Trojan-horse, time-shift, detector-blinding	Низька	Середня	Середня-висока
MiTM	Середня (при автентифікації)	Середня	Висока

Продовження таблиці 3.3

1	2	3	4
Faked-state, intercept-resend	Середня-висока	Висока	Дуже висока
Side-channel	Низька-середня	Середня	Середня-висока

Отже, використання системи залишкових класів у протоколі BB84 дає змогу підвищити стійкість до окремих класів атак на етапі постобробки та завадостійкості кодування станів. У постобробці СЗК дозволяє поділити ключ на незалежні модульні компоненти, зменшуючи ефективність атак, які залежать від кореляцій між бітами (наприклад, узгоджених статистичних атак). Навіть якщо підслуховувач отримує часткову інформацію про один модуль, повне відновлення ключа залишається математично складною задачею без інформації про всі модулі.

У кодуванні квантових станів застосування СЗК може бути використане для побудови надлишкових або мультиплексованих схем, де різні модулі передаються в різних базисах або у вигляді різних підпослідовностей. Це дозволяє ефективно виявляти й локалізувати спроби атаки шляхом контролю узгодженості модульних компонентів. Окрім того, СЗК може використовуватись як додатковий «структурний шар» захисту: навіть при перехопленні фотонів підслуховувач не може відтворити повну структуру ключа, якщо не знає модулярних параметрів. Такий підхід також підсилює процедури виявлення помилок та скорочення ключа, оскільки модульні невідповідності дозволяють швидше виявити спроби підміни або маніпуляції станами.

Проте, незважаючи на потенційні переваги, інтеграція залишково-класової арифметики в протокол BB84 висуває низку практичних обмежень. Однією з ключових проблем є чутливість СЗК до помилок у певних конфігураціях модулів**.** Якщо модулі підібрано неправильно або між ними існує кореляція, навіть одинична помилка в одному каналі може призвести до неможливості

коректно відновити вектор ключа. Це створює підвищені вимоги до точності вимірювань, стабільності базисів та мінімізації шумів у квантовому каналі.

Кодування станів із використанням СЗК потребує також значно більш жорсткої синхронізації між відправником і приймачем, оскільки різні модульні компоненти можуть передаватися з різною інтенсивністю або з різними характеристиками фазового шуму. Будь-яка асиметрія між квантовими та класичними каналами створює ризик неправильного відновлення ключа навіть за відсутності атак. Додаткову складність становить той факт, що СЗК вимагає точної початкової домовленості про вибір модулів та їх параметрів, а отже збільшує навантаження на класичний канал автентифікації та калібрацію системи. У випадку атаки на синхронізацію або спотворення класичних повідомлень підслуховувач може навмисно ввести помилки у модульні компоненти, спричиняючи некоректне збирання ключа без явного порушення статистики квантових вимірювань.

ВИСНОВКИ

У проведеній роботі було досліджено стійкість класичного та модифікованих варіантів протоколу BB84 до широкого спектра атак на основний канал та бокові канали, включно з перехопленням і повторним відправленням, PNS-атаками, фазовими атаками, атаками засліплення детектора, атаками типу «Троянський кінь» та низкою інших фізичних та програмно-апаратних загроз. Особливу увагу було приділено використанню системи залишкових класів (СЗК) як інструменту підвищення криптостійкості протоколу на етапі постобробки ключа та під час кодування квантових станів.

Аналіз показав, що застосування СЗК у постобробці ключа підвищує інформаційну безпеку системи, оскільки частковий витік одного або кількох залишків не дає змоги відновити повне значення ключа без усіх модульних компонентів. Ця властивість ускладнює реалізацію низки атак, зокрема статистичних та кореляційних, а також зменшує ефективність атак, пов’язаних із детекторними та електромагнітними витоками. Проте СЗК у постобробці не усуває основні фізичні вразливості квантового каналу.

Застосування СЗК на етапі кодування станів забезпечує значно вищий рівень захищеності. Така модифікація підвищує неоднозначність для підслуховувача, робить складнішими атаки на кодування та відтворення фотонів, а також зменшує ефективність PNS-атак, фазових атак та атак, побудованих на спотворенні базисів. Кодування на основі СЗК створює додатковий рівень структури даних, що ускладнює перехоплення та відтворення стійких до помилок модульних компонентів.

Разом з тим інтеграція СЗК у протокол BB84 супроводжується практичними викликами: підвищеними вимогами до синхронізації, чутливістю до помилок, необхідністю точного калібрування обладнання та ризиком виникнення нехарактеризованих бокових каналів. Усе це підкреслює важливість подальших досліджень, спрямованих на взаємну оптимізацію квантових та класичних механізмів у нових гібридних протоколах.

Загалом, результати роботи підтверджують, що модифікований протокол BB84 із застосуванням системи залишкових класів є перспективним напрямом розвитку квантового розподілу ключів, здатним забезпечити підвищену стійкість до реальних атак і водночас зберегти основні переваги квантової криптографії. Подальші дослідження в цьому напрямі можуть стати основою для створення практично реалізовних, масштабованих і високобезпечних систем QKD нового покоління.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Scarani, V., Bechmann-Pasquinucci, H., Cerf, N. J., Dušek, M., Lütkenhaus, N., & Peev, M. (2009). The security of practical quantum key distribution. Reviews of Modern Physics, 81(3), 1301-1350.

2. Diamanti, E., Lo, H. K., Qi, B., & Yuan, Z. (2016). Practical challenges in quantum key distribution. npj Quantum Information, 2(1), 1-12.

3. Xu, F., Ma, X., Zhang, Q., Lo, H. K., & Pan, J. W. (2020). Secure quantum key distribution with realistic devices. Reviews of Modern Physics, 92(2), 025002.

4. Shor, P. W. (1999). Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. SIAM Review, 41(2), 303-332.

5. Pirandola, S., Andersen, U. L., Banchi, L., Berta, M., Bunandar, D., Colbeck, R., ... & Wallden, P. (2020). Advances in quantum cryptography. Advances in Optics and Photonics, 12(4), 1012-1236.

6. Gisin, N., Ribordy, G., Tittel, W., & Zbinden, H. (2002). Quantum cryptography. Reviews of Modern Physics, 74(1), 145.

7. Lo, H. K., Curty, M., & Tamaki, K. (2014). Secure quantum key distribution. Nature Photonics, 8(8), 595-604.

8. Dušek, M., Lütkenhaus, N., & Hendrych, M. (2006). Quantum cryptography. Progress in Optics, 49, 381-454.

9. Brassard, G., & Salvail, L. (1994). Secret-key reconciliation by public discussion. In Workshop on the Theory and Application of Cryptographic Techniques (pp. 410-423). Springer.

10. Gottesman, D., Lo, H. K., Lütkenhaus, N., & Preskill, J. (2001). Security of quantum key distribution with imperfect devices. arXiv preprint quant-ph/0212066.

11. Гнатюк А., Касянчук М. Метод формування квантового ключа за протоколом BB84 на основі системи залишкових класів. *Кібербезпека державних інституцій та подолання кризових станів*: матеріали III Міжнар. наук.-практ. конф. в 2 т. (Київ – Прага – Таллінн – Тернопіль), 14 листоп. 2024 р. / ІСЗЗІ КПІ ім. Ігоря Сікорського. Київ, 2024. Т. 1. С. 398-399

12. Гнатюк А., Касянчук М., Басістий П. Модифікація квантового протоколу BB84 за допомогою системи залишкових класів. Матеріали XIV міжнародної науково-практичної конференції «Безпека інформаційних технологій: ITSEC-2025». С. 47-49.

13. A. Hnatiuk; M. Kasianchuk; R. Shevchuk; L. Tymoshenko. Method of Quantum Key Distribution Based on a Modified BB84 Protocol Using the Residue Number System. 2025 15th International Conference on Advanced Computer Information Technologies (ACIT), Sibenik, Croatia, 2025, pp. 466-470.

14. Omondi, A., & Premkumar, B. (2007). Residue number systems: Theory and implementation. Imperial College Press.

15. Mohan, P. V. A. (2002). Residue number systems: Algorithms and architectures. Springer Science & Business Media.

16. Garnerone, S., Zanardi, P., & Lidar, D. A. (2009). Adiabatic quantum computation in open systems. Physical Review Letters, 102(11), 110503.

17. Ekert, A. K. (1991). Quantum cryptography based on Bell's theorem. Physical Review Letters, 67(6), 661.

18. Bennett, C. H., & Brassard, G. (1984). Quantum cryptography: Public key distribution and coin tossing. Theoretical Computer Science, 560(12), 7-11.

19. Bennett, C. H. (1992). Quantum cryptography using any two nonorthogonal states. Physical Review Letters, 68(21), 3121.

20. Scarani, V., Acin, A., Ribordy, G., & Gisin, N. (2004). Quantum cryptography protocols robust against photon number splitting attacks for weak laser pulse implementations. Physical Review Letters, 92(5), 057901.

21. Lo, H. K., Ma, X., & Chen, K. (2005). Decoy state quantum key distribution. Physical Review Letters, 94(23), 230504.

22. Ananthaswamy, A. (2013). Quantum cryptography: The ultimate code. New Scientist, 217(2906), 36-39.

23. Zhang, Q., Xu, F., Chen, Y. A., Peng, C. Z., & Pan, J. W. (2018). Large scale quantum key distribution: Challenges and solutions. Optics Express, 26(18), 24260-24273.

24. Lucamarini, M., Yuan, Z., Dynes, J. F., & Shields, A. J. (2018). Overcoming the rate–distance limit of quantum key distribution without quantum repeaters. Nature, 557(7705), 400-403.

25. Korzh, B., Lim, C. W., Houlmann, R., Gisin, N., Li, M. J., Nolan, D., ... & Zbinden, H. (2015). Provably secure and practical quantum key distribution over 307 km of optical fibre. Nature Photonics, 9(3), 163-168.

26. Gleim, A. V., Egorov, V. I., Nazarov, Y. V., Chaldaev, N. N., Anisimov, A. A., Chistyakov, V. V., ... & Gaidash, A. A. (2016). Secure polarization-independent subcarrier quantum key distribution in optical fiber channel using BB84 protocol with a strong reference. Optics Express, 24(3), 2619-2633.

27. Wang, S., Chen, W., Yin, Z. Q., Li, H. W., He, D. Y., Li, Y. H., ... & Han, Z. F. (2016). Field and long-distance demonstration of a fully continuous-variable quantum key distribution system. Nature Communications, 7(1), 1-7.

28. Lo, H. K., Curty, M., & Qi, B. (2012). Measurement-device-independent quantum key distribution. Physical Review Letters, 108(13), 130503.

29. Inoue, K., Waks, E., & Yamamoto, Y. (2002). Differential phase shift quantum key distribution. Physical Review Letters, 89(3), 037902.

30. Sziklai, P. (1958). A theory of residue number systems. IRE Transactions on Electronic Computers, EC-7(3), 179-187.

31. Akushskii, I. Y., & Juditskii, D. I. (1968). Machine arithmetic in residue classes. Moscow: Sovetskoe Radio.

32. Omondi, A. R. (1994). Computer arithmetic systems: Algorithms, architecture and implementation. Prentice Hall PTR.

33. Piestrak, S. J. (1995). A high-speed realization of a residue to binary converter. IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing, 42(10), 661-663.

34. Bajard, J. C., Didier, L. S., & Kornerup, P. (1998). An RNS Montgomery modular multiplication algorithm. IEEE Transactions on Computers, 47(7), 766-776.

35. Nozaki, H., Motoyama, M., Shimada, R., & Kawaguchi, H. (2000). Implementation of RSA algorithm based on RNS Montgomery multiplication. In

Cryptographic Hardware and Embedded Systems—CHES 2001 (pp. 364-376). Springer.

36. Schinianakis, D. M., Fournaris, A. P., Michail, H. E., Kakarountas, A. P., & Stouraitis, T. (2006). An RNS implementation of an F_p elliptic curve point multiplier. IEEE Transactions on Circuits and Systems I: Regular Papers, 56(6), 1202-1213.

37. Bajard, J. C., & Imbert, L. (2004). A full RNS implementation of RSA. IEEE Transactions on Computers, 53(6), 769-774.

38. Kawamura, S. I., Koike, M., Saki, M., & Izumi, T. (2000). Implementation of a fast modular reduction algorithm for elliptic curve cryptosystems. IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, 83(5), 755-765.

39. Esmaeildoust, M., Schinianakis, D., Javashi, H., Stouraitis, T., & Navi, K. (2013). Efficient RNS implementation of elliptic curve point multiplication over GF(p). IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 21(5), 987-990.

40. Guillermin, D. (2010). RNS-enabled digital signal processor design. In Design, Automation & Test in Europe Conference & Exhibition (DATE), 2010 (pp. 319-322). IEEE.

41. Chaves, R., & Sousa, L. (2007). RDSP: A RNS DSP implementation for reconfigurable radio. Journal of Signal Processing Systems, 51(3), 259-271.

42. Brassard, G., Lütkenhaus, N., Mor, T., & Sanders, B. C. (2000). Limitations on practical quantum cryptography. Physical Review Letters, 85(6), 1330.

43. Lütkenhaus, N. (2000). Security against individual attacks for realistic quantum key distribution. Physical Review A, 61(5), 052304.

44. Makarov, V., Anisimov, A., & Skaar, J. (2006). Effects of detector efficiency mismatch on security of quantum cryptosystems. Physical Review A, 74(2), 022313.

45. Qi, B., Fung, C. H. F., Lo, H. K., & Ma, X. (2007). Time-shift attack in practical quantum cryptosystems. Quantum Information & Computation, 7(1), 73-82.

46. Lydersen, L., Wiechers, C., Wittmann, C., Elser, D., Skaar, J., & Makarov, V. (2010). Hacking commercial quantum cryptography systems by tailored bright illumination. Nature Photonics, 4(10), 686-689.

47. Jain, N., Wittmann, C., Lydersen, L., Wiechers, C., Elser, D., Marquardt, C., ... & Leuchs, G. (2011). Device calibration of a fiber-optic quantum key distribution system using a wavelength-tunable laser. Journal of Applied Physics, 109(10), 106105.

```
import random
import tkinter as tk
from tkinter import scrolledtext, messagebox

class BB84_SZK:
    def __init__(self, length, modulus, remainder_to_polarization):
        self.length = length
        self.modulus = modulus
        self.remainder_to_polarization = remainder_to_polarization  # Словник:
залишок -> поляризація
        self.allowed_remainders = list(remainder_to_polarization.keys())  #
Дозволені залишки
        self.sequence = [random.choice(self.allowed_remainders) for _ in
range(length)]
        self.bases = self.assign_bases()  # Призначаємо базиси на основі
поляризації
        self.states = self.encode()

    def encode(self):
        # Кодування залишків у поляризацію за словником
        return [self.remainder_to_polarization[val] for val in self.sequence]

    def assign_bases(self):
        # Призначаємо базиси на основі поляризації
        bases = []
        for val in self.sequence:
            polarization = self.remainder_to_polarization[val]
            if polarization in ["0°", "90°"]:
                bases.append("+")
            else:  # 45° або 135°
                bases.append("×")
        return bases

def simulate_bob(alice):
    # Базиси Боба генеруються випадково, але враховують коректність вимірювання
    bob_bases = [random.choice(["+", "×"]) for _ in range(alice.length)]
    bob_polarization = []
    bob_bits = []
    for val, a_base, b_base, state in zip(alice.sequence, alice.bases, bob_bases,
alice.states):
        if a_base == b_base:
            bob_bits.append(val)
            bob_polarization.append(state)
        else:
            rand_val = random.randint(0, alice.modulus-1)
            bob_bits.append(rand_val)
            # Якщо базис не збігається, Боб отримує випадкову поляризацію із
можливих
            if b_base == "+":
                bob_polarization.append(random.choice(["0°", "90°"]))
```

```
            else:
                bob_polarization.append(random.choice(["45°", "135°"]))
    raw_key = [a for a, ab, bb in zip(alice.sequence, alice.bases, bob_bases) if ab
== bb]
    return bob_bases, bob_polarization, bob_bits, raw_key

def szk_postprocess(key, modulus):
    decimal_str = ''.join(map(str, key))
    decimal = int(decimal_str) if decimal_str else 0
    final_key = decimal % modulus
    return decimal, final_key

def show_results(title, text):
    result_window = tk.Toplevel()
    result_window.title(title)
    lines = text.split('\n')
    max_length = max(len(line) for line in lines)
    text_area = scrolledtext.ScrolledText(result_window, width=max_length + 2,
height=len(lines) + 2, wrap=tk.NONE)
    text_area.insert(tk.END, text)
    text_area.pack(padx=10, pady=10)
    text_area.config(state='disabled')

def run_simulation():
    try:
        length = int(length_entry.get())
        modulus = int(modulus_entry.get())
        if modulus < 2:
            raise ValueError("Modulus shall be >= 2")

        # Залишки, введені користувачем, та їх прив’язка до поляризації
        remainder_to_polarization = {
            int(horiz_remainder.get()): "0°",
            int(vert_remainder.get()): "90°",
            int(diag_remainder.get()): "45°",
            int(antidiag_remainder.get()): "135°"
        }
        if any(r < 0 or r >= modulus for r in remainder_to_polarization.keys()):
            raise ValueError("Enter correct residues (from 0 to modulus-1)")
        if len(set(remainder_to_polarization.keys())) != 4:  # Перевірка на
унікальність залишків
            raise ValueError("Residues shall be unique")

        alice = BB84_SZK(length, modulus, remainder_to_polarization)
        bob_bases, bob_pol, bob_bits, raw_key = simulate_bob(alice)
        decimal_key, final_key = szk_postprocess(raw_key, modulus)

        result_text = (f"Alice's initial data sequence: {alice.sequence}\n"
                       f"Alice's basis: {alice.bases}\n"
                       f"Alice's polarization: {alice.states}\n"
                       f"Bob's basis: {bob_bases}\n"
```

```
                      f"Bob's polarization: {bob_pol}\n"
                      f"Bob's received value sequence: {bob_bits}\n"
                      f"Secret key: {raw_key}\n"
                      f"Converting to decimal number: {decimal_key}\n"
                      f"Key in RNS (mod {modulus}): {final_key}")
        show_results("Simulation results (RNS in Quantum States Encoding)",
result_text)
    except ValueError as e:
        messagebox.showerror("Error", str(e))

# Графічний інтерфейс
root = tk.Tk()
root.title("BB84 & RNS in Quantum States Encoding")

tk.Label(root, text="Alice's key lenght:").grid(row=0, column=0, columnspan=2,
padx=5, pady=5)
length_entry = tk.Entry(root)
length_entry.grid(row=0, column=2)

tk.Label(root, text="Moduli:").grid(row=1, column=0, columnspan=2, padx=5, pady=5)
modulus_entry = tk.Entry(root)
modulus_entry.grid(row=1, column=2)

tk.Label(root, text="Rectilinear bases (+):").grid(row=2, column=0, columnspan=3,
pady=5)
tk.Label(root, text="Horizontal (0°):").grid(row=3, column=0, padx=5)
horiz_remainder = tk.Entry(root)
horiz_remainder.grid(row=3, column=1)
tk.Label(root, text="Vertical (90°):").grid(row=4, column=0, padx=5)
vert_remainder = tk.Entry(root)
vert_remainder.grid(row=4, column=1)

tk.Label(root, text="Diagonal bases (×):").grid(row=5, column=0, columnspan=3,
pady=5)
tk.Label(root, text="Diagonal (45°):").grid(row=6, column=0, padx=5)
diag_remainder = tk.Entry(root)
diag_remainder.grid(row=6, column=1)
tk.Label(root, text="Antidiagonal (135°):").grid(row=7, column=0, padx=5)
antidiag_remainder = tk.Entry(root)
antidiag_remainder.grid(row=7, column=1)

tk.Button(root, text="Run Simulation", command=run_simulation).grid(row=8,
column=0, columnspan=3, pady=10)

root.mainloop()
```

ДОДАТОК Б. Код для використання СЗК в постоброці ключа

```
import random
import tkinter as tk
from tkinter import scrolledtext, messagebox
def bb84_classic(length):
    alice_bits = [random.randint(0, 1) for _ in range(length)]
    alice_bases = [random.choice(["+", "×"]) for _ in range(length)]
    alice_polarization = []
    for bit, base in zip(alice_bits, alice_bases):
        if base == "+":
            alice_polarization.append("0°" if bit == 0 else "90°")
        else:
            alice_polarization.append("45°" if bit == 0 else "135°")

    bob_bases = [random.choice(["+", "×"]) for _ in range(length)]
    bob_polarization = []
    bob_bits = []
    for bit, a_base, b_base in zip(alice_bits, alice_bases, bob_bases):
        if a_base == b_base:
            bob_bits.append(bit)
            bob_polarization.append("0°" if bit == 0 and b_base == "+" else "90°"
if bit == 1 and b_base == "+" else "45°" if bit == 0 else "135°")
        else:
            rand_bit = random.randint(0, 1)
            bob_bits.append(rand_bit)
            bob_polarization.append("0°" if rand_bit == 0 and b_base == "+" else
"90°" if rand_bit == 1 and b_base == "+" else "45°" if rand_bit == 0 else "135°")
    raw_key = [a for a, ab, bb in zip(alice_bits, alice_bases, bob_bases) if ab ==
bb]
    return alice_bits, alice_bases, alice_polarization, bob_bases,
bob_polarization, bob_bits, raw_key
def szk_postprocess(key, modulus_list):
    binary = ''.join(map(str, key))
    decimal = int(binary, 2) if binary else 0
    final_key = [decimal % mod for mod in modulus_list] if modulus_list else
[decimal]
    return decimal, final_key
def show_results(title, text):
    result_window = tk.Toplevel()
    result_window.title(title)
    lines = text.split('\n')
    max_length = max(len(line) for line in lines)
    text_area = scrolledtext.ScrolledText(result_window, width=max_length + 2,
height=len(lines) + 2, wrap=tk.NONE)
    text_area.insert(tk.END, text)
    text_area.pack(padx=10, pady=10)
    text_area.config(state='disabled')
def run_simulation():
    try:
        length = int(length_entry.get())
        mod_min = int(mod_min_entry.get())
```

```
        mod_max = int(mod_max_entry.get())
        num_mods = int(num_mods_entry.get())
        if mod_min < 2 or mod_max < mod_min:
            raise ValueError("Modulus range shall be >= 2 and max >= min")
        if num_mods < 1:
            raise ValueError("Number of modules shall be >= 1")

        modulus_list = [random.randint(mod_min, mod_max) for _ in range(num_mods)]
        alice_bits, alice_bases, alice_pol, bob_bases, bob_pol, bob_bits, raw_key =
bb84_classic(length)
        decimal_key, final_key = szk_postprocess(raw_key, modulus_list)

        result_text = (f"Alice's initial bit sequence: {alice_bits}\n"
                       f"Alice's basis: {alice_bases}\n"
                       f"Alice's polarization: {alice_pol}\n"
                       f"Bob's basis: {bob_bases}\n"
                       f"Bob's polarization: {bob_pol}\n"
                       f"Bob's received bit sequence: {bob_bits}\n"
                       f"Secret key: {raw_key}\n"
                       f"Converting to decimal number: {decimal_key}\n"
                       f"Number of modules: {num_mods}, modules: {modulus_list}\n"
                       f"Key in RNS: {tuple(final_key) if num_mods > 1 else
final_key[0]}")
        show_results("Simulation results (RNS in Post-Processing)", result_text)
    except ValueError as e:
        messagebox.showerror("Error", str(e))

# Графічний інтерфейс
root = tk.Tk()
root.title("BB84 & RNS in Key Post-Processing")

tk.Label(root, text="Alice's key lenght:").grid(row=0, column=0, padx=5, pady=5)
length_entry = tk.Entry(root)
length_entry.grid(row=0, column=1)

tk.Label(root, text="Minimal moduli:").grid(row=1, column=0, padx=5, pady=5)
mod_min_entry = tk.Entry(root)
mod_min_entry.grid(row=1, column=1)

tk.Label(root, text="Maximal moduli:").grid(row=2, column=0, padx=5, pady=5)
mod_max_entry = tk.Entry(root)
mod_max_entry.grid(row=2, column=1)

tk.Label(root, text="Number of modules:").grid(row=3, column=0, padx=5, pady=5)
num_mods_entry = tk.Entry(root)
num_mods_entry.grid(row=3, column=1)

tk.Button(root, text="Run Simulation", command=run_simulation).grid(row=4,
column=0, columnspan=2, pady=10)

root.mainloop()
```

ДОДАТОК В. Сертифікати про участь в конференціях

CERTIFICATE

OF CONFERENCE PARTICIPANT

Anastasiia HNATIUK

took part in the III International Scientific and Practical Conference
"CYBERSECURITY OF STATE INSTITUTIONS AND CRISIS MANAGEMENT"
(0.2 ECTS credits)

14 November 2024

Head of Institute of Special Communications and Information Protection of the National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute"
Brigadier General, PhD, Professor Oleksandr PUCHKOV

СЕРТИФІКАТ

ЗА УЧАСТЬ У КОНФЕРЕНЦІЇ

АНАСТАСІЯ ГНАТЮК

Взяв (ла) участь у роботі:
XIV міжнародної науково-технічної конференції
"Безпека інформаційних технологій: ITSEC-2025"
22-24 травня 2025 року (30 годин)

ПРЕЗИДЕНТ ГО "АСКБ" ОЛЕКСАНДР КОРЧЕНКО

ДУІКТ ЗНУ ЗАХІДНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ Cyber Security Department

м. Тернопіль, Україна

WEST UKRAINIAN NATIONAL UNIVERSITY
VELEUČILIŠTE U ŠIBENIKU
Wrocław University of Economics and Business
TECHNISCHE HOCHSCHULE DEGGENDORF THD
Catholic University in Ružomberok

CERTIFICATE

IS HEREBY GRANTED TO

Anastasiya Hnatiuk

FOR PARTICIPATING IN THE 2025 15TH INTERNATIONAL CONFERENCE ON ADVANCED COMPUTER INFORMATION TECHNOLOGIES 17-19 SEPTEMBER, 2025 IN ŠIBENIK, CROATIA

CHAIRMAN
OF THE LOCAL ORGANIZING
COMMITTEE

FRANE UREM

ДОДАТОК Г. Копії публікацій

Анастасія ГНАТЮК;
Михайло КАСЯНЧУК, д.т.н., професор

МЕТОД ФОРМУВАННЯ КВАНТОВОГО КЛЮЧА ЗА ПРОТОКОЛОМ BB84 НА ОСНОВІ СИСТЕМИ ЗАЛИШКОВИХ КЛАСІВ

Анотація. Протокол BB84 захищеного розподілу ключів є базовим у квантовій криптографії. У роботі пропонується його покращення за допомогою системи залишкових класів. Запропонований підхід дає змогу зменшити кількість використаних кубітів, підвищити ефективність та безпеку процесу передачі ключа.

Summary. The BB84 protocol for secure key distribution is fundamental in quantum cryptography. This paper proposes an improvement by incorporating residue number system. The proposed approach reduces the number of qubits and enchances efficiency and security of key transfer process.

Ключові слова: протокол BB84, квантовий ключ, система залишкових класів, розподіл ключів, кубіт.

Протокол BB84 є першим і найбільш відомим квантовим протоколом для розподілу криптографічного ключа, що забезпечує інформаційно-теоретичну безпеку за рахунок законів квантової механіки. Основною перевагою протоколу є його здатність виявляти спроби перехоплення інформації через квантові вимірювання. Проте вдосконалення цього протоколу можливе за рахунок інтеграції системи залишкових класів (СЗК), що дозволяє підвищити ефективність розподілу ключів.

Протокол BB84 використовує два базиси для передавання квантових бітів (кубітів): базис з прямими та діагональними поляризаціями. Відправник (Аліса) передає кубіти, обираючи випадковим чином один із базисів, а одержувач (Боб) також випадково обирає базис для вимірювання. Після цього, обравши однакові базиси, обидві сторони можуть отримати однакові біти, що потім використовуються для формування криптографічного ключа.

Система залишкових класів (СЗК) — це математичний метод, який дозволяє представити велике число через залишки за набором попарно взаємно простих модулів. Використання СЗК для формування ключа у протоколі BB84 може значно підвищити безпеку та ефективність ключового обміну за рахунок таких його властивостей:

1) СЗК дозволяє використовувати більший обсяг інформації з кожного переданого кубіта. Оскільки СЗК представляє числа через залишки, то можна зменшити кількість переданих кубітів для отримання того самого обсягу інформації, що знижує можливість помилок та підвищує пропускну здатність системи;

2) під час використання СЗК кожна частина ключа передається за різними модулями, що робить спроби перехоплення ключа менш ефективними, оскільки зловмиснику необхідно відновити всі залишки для правильного відновлення ключа.

Нехай криптографічний ключ складається з N-бітової послідовності. Для визначеності візьмемо $N = 8$ біт. У традиційному BB84 кожен біт передається окремо і потрібно передати 8 кубітів.

Для використання СЗК потрібно обрати попарно взаємно прості числа: p_1=3, p_2=5, p_3=7. Тоді ключ можна представити як набір залишків за цими модулями. Наприклад, представлене у десятковій системі числення число 13 має такі залишки:

$$13 \bmod 3 = 1$$
$$13 \bmod 5 = 3$$
$$13 \bmod 7 = 6$$

Замість того, щоб передавати весь ключ як 8-бітове число, можна передати лише залишки (1, 3, 6) за трьома різними модулями. Це дозволяє зменшити кількість переданих кубітів, водночас забезпечуючи високу безпеку.

Цей підхід дозволяє оптимізувати обмін ключами та забезпечити більшу стійкість до помилок та атак, спрямованих на злам протоколу. Завдяки використанню попарно взаємно простих модулів, відновлення ключа стає складним завданням для зловмисника, оскільки необхідно перехопити всі залишки та правильно обчислити вихідне число. Також зниження кількості переданих кубітів зменшує ресурсні витрати на квантовий обмін та підвищує загальну пропускну здатність системи. Таким чином, інтеграція СЗК створює нові можливості для вдосконалення квантових криптографічних протоколів і підвищує їхню ефективність у сучасних умовах кіберзагроз.

Висновки. Використання СЗК у протоколі BB84 дозволяє покращити ефективність і безпеку квантового обміну ключами. Зменшення кількості кубітів, якими потрібно обмінюватися, знижує ймовірність помилок і полегшує виявлення спроб несанкціонованого доступу. Кім того, представлення ключа через залишки ускладнює перехоплення інформації. Таким чином, інтеграція СЗК у протокол BB84 може стати важливим кроком у розвитку квантової криптографії.

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ
ЗАХІДНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
UNIVERSITY OF THE NATIONAL EDUCATION COMMISSION, POLAND
TECHNICAL UNIVERSITY IN PRAGUE, CZECH REPUBLIC
Наукова школа "Кібербезпека"
Навчально-науковий інститут Кібербезпеки та захисту інформації ДУІКТ
Кафедра кібербезпеки ЗУНУ
ГО «АСОЦІАЦІЯ СПЕЦІАЛІСТІВ КІБЕРБЕЗПЕКИ»
ГО «АВТОМАТИЗАЦІЯ І КІБЕРБЕЗПЕКА»

ITSec-2025

Безпека інформаційних технологій

МАТЕРІАЛИ

XIV Міжнародної науково-технічної конференції

22-24 травня 2025
м. Тернопіль (Україна)

Модифікація квантового протоколу BB84 за допомогою системи залишкових класів

УДК 004.056.55

Анастасія Гнатюк[1], Михайло Касянчук[2], Павло Басістий[3]

[1,2]*Західноукраїнський національний університет,* [3]*Тернопільський національний педагогічний університет імені Володимира Гнатюка,*
[1]*anastasia.hn88@gmail.com,* [2]*kasyanchuk@ukr.net,* [3]*basi@ukr.net*

Квантова криптографія використовує принципи квантової механіки для встановлення каналів зв'язку між двома об'єктами і визначає методи безпечного спілкування через введення принципів квантової механіки для генерації ключів шифрування [1]. На відміну від класичної, квантова криптографія забезпечує свою безпеку через незмінні закони фізики, що призводить до її виняткової захищеності навіть відносно найскладніших обчислювальних систем [2].

Квантовий протокол розподілу ключів BB84 покладається на такі принципи квантової фізики:

1) принцип невизначеності Гейзенберга, який стверджує, що в квантовій системі можна точно визначити лише одну з пари спряжених величин, наприклад, координата та імпульс (вимірювання положення частинки призведе до порушення її швидкості). Квантова криптографія використовує це, застосовуючи поляризацію фотонів;

2) теорема про заборону клонування як наслідок попереднього принципу стверджує, що неможливо створити ідентичні копії невідомого квантового

стану. Завдяки цьому можна виявити, чи хтось перехопив квантовий канал під час критичної передачі інформації;

3) квантова заплутаність: незалежно від відстані дві квантові частинки можуть бути заплутані. Коли певна властивість вимірюється в одній частинці, то корельований стан цієї властивості з'явиться в іншій частинці.

Протокол BB84 використовує квантові стани для обміну секретним ключем між двома сторонами — Алісою (відправником) та Бобом (отримувачем). Він складається з двох фаз: квантової (передача фотонів) і класичної (узгодження ключа та виправлення помилок).

Передбачається два способи використання СЗК в протоколі BB84: для кодування квантових станів та постобробки ключа. Розглянемо кожен спосіб окремо.

Для використання СЗК в кодуванні станів, Аліса та Боб перед початком формування секретного ключа повинні обрати певний модуль та довільні залишки при діленні на нього, а також присвоїти деякий залишок кожному фільтру поляризації. Після цього відбувається процес формування ключа згідно протоколу BB84. Далі отриманий секретний ключ перевіряється на підслуховування та обчислюється за заздалегідь обраним модулем. Таким чином отримується фінальний ключ.

Даному методу інтеграції протоколу BB84 та СЗК властиві такі переваги:

1) збільшення пропускної здатності ключа, оскільки у класичному виконанні протоколу BB84 одним фотоном кодується один біт. При використанні СЗК із модулем m кожен фотон передаватиме $\log_2 m$ бітів. Наприклад, для модуля 4 із залишками (0, 1, 2, 3) буде передаватися 8 бітів замість чотирьох, як в класичному протоколі;

2) гнучкість кодування досягається за допомогою можливості адаптації протоколу BB84 до різних модулів, що дає змогу масштабувати обсяг інформації залежно від потреб системи без зміни апаратної основи;

3) сумісність із криптосистемами, які працюють із багатозначними станами, що може бути корисним для майбутніх квантових алгоритмів;

4) збереження квантової безпеки завдяки основному принципу протоколу BB84 – виявлення підслуховування через квантові помилки.

Для використання СЗК в постобробці ключа Аліса та Боб спершу повинні виконати ті ж кроки, що і в класичному протоколі BB84. Після цього по класичному каналу вони обговорюють кількість модулів та їх значення і отриманий секретний ключ переводять в десяткову систему числення. Фінальний ключ отримується після обчислення залишків секретного ключа за обраними модулями.

Перевагами даного методу інтеграції є:

1) стиснення ключа внаслідок перетворення двійкового числа в десяткове і пошук його залишку за модулем;

2) простота інтеграції з модульними криптосистемами досягається через використання ключа у форматі залишку, який може бути застосованим в системах на основі модульної арифметики без додаткових перетворень;

3) оптимізація корекції помилок через використання залишків в якості контрольних сум для блоків ключа, що полегшить виявлення помилок при передачі каналами зв'язку;

4) гнучкість масштабування через можливість зміни модуля, що дозволить адаптувати розмір ключа до потреб системи без змін у квантовій частині протоколу;

5) збереження безпеки в зв'язку з незмінністю квантової частини протоколу BB84.

Отже, введення СЗК у кодування квантових станів більше підходить для систем, де пріоритетом є максимізація пропускної здатності квантового каналу, а введення СЗК у постобробку ключа є простішим у реалізації та ідеальним для систем із обмеженими ресурсами чи потребою в інтеграції з модульними алгоритмами. Крім того, перший метод передбачає кодування квантових станів в СЗК за допомогою обраних залишків певного модуля та обрахунок отриманого секретного ключа за модулем. Другий метод дозволяє стиснути отриманий з протоколу BB84 двійковий секретний ключ у більш компактний вигляд через переведення його в десяткову систему числення та обчислення за обраним модулем.

1. S.Pirandola at al. "Advances in quantum cryptography." *Advances in Optics and Photonics*. Vol.12, no.4, pp.1012-1236, 2020.

2. M.Swan, R. dos Santos, F.Witte. "Quantum Information Science." *IEEE Internet Comput.* Vol.26, pp.7–14, 2021.

2025 15th International Conference on Advanced Computer Information Technologies (ACIT) | 979-8-3315-9544-2 | 979-8-3315-9543-2 (USB) | 2025 IEEE | DOI: 10.1109/ACIT65614.2025.11185943

IEEE
CROATIA SECTION

2025 15TH INTERNATIONAL CONFERENCE ON

ADVANCED COMPUTER INFORMATION TECHNOLOGIES

ŠIBENIK, CROATIA
17-19 SEPTEMBER 2025

ORGANIZERS:

VELEUČILIŠTE U ŠIBENIKU

THD

PART NUMBER: CFP25S92-PRT

ISBN: 979-8-3315-9543-2

ISSN: 2770-5218

ACIT.TECH

WEST UKRAINIAN NATIONAL UNIVERSITY, UKRAINE
ŠIBENIK UNIVERSITY OF APPLIED SCIENCES, CROATIA
WROCLAW UNIVERSITY OF ECONOMICS AND BUSINESS, POLAND
DEGGENDORF INSTITUTE OF TECHNOLOGY, GERMANY
CATHOLIC UNIVERSITY IN RUŽOMBEROK, SLOVAKIA
IEEE CROATIA SECTION

2025 15th International Conference on

ADVANCED COMPUTER INFORMATION TECHNOLOGIES ACIT'2025

Conference Proceedings

Šibenik, Croatia
17-19 September 2025