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

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

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня

магістр
(назва освітнього ступеня)

на тему: "*Методи формування псевдовипадкових чисел в криптографічних засобах захисту банківських інформаційних систем*"

Виконав(ла): студент(ка) *VI* курсу, групи *СБм-61*
спеціальності *125 «Кібербезпека»*
(шифр і назва спеціальності)

	(підпис)	Баняс Б.М. (прізвище та ініціали)
Керівник	(підпис)	Александер М.А. (прізвище та ініціали)
Нормоконтроль	(підпис)	Кареліна О.В. (прізвище та ініціали)
Завідувач кафедри	(підпис)	Загородна Н.В. (прізвище та ініціали)
Рецензент	(підпис)	Дуда О.М. (прізвище та ініціали)

Тернопіль
20 21

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

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

Кафедра *кібербезпеки*
(повна назва кафедри)

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Загородна Н.В.
(підпис) (прізвище та ініціали)
« » 20 21 р.

З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня Магістр
(назва освітнього ступеня)

за спеціальністю 125 "Кібербезпека"
(шифр і назва спеціальності)

студенту Банясу Богдану Мироновичу
(прізвище, ім'я, по батькові)

1. Тема роботи Методи формування псевдовипадкових чисел в криптографічних засобах захисту банківських інформаційних систем

Керівник роботи Александр Марек Богуслав Антонович
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Затверджені наказом ректора від «___» _____ 20___ року № _____

2. Термін подання студентом завершеної роботи _____

3. Вихідні дані до роботи _____

4. Зміст роботи (перелік питань, які потрібно розробити)
Вступ. 1.Аналіз відомих методів формування псевдовипадкових чисел у криптографічних засобах захисту банківських інформаційних систем 2. Розробка методу формування псевдовипадкових чисел на основі надлишкових блокових кодів. 3. Розробка програмної реалізації генератора псевдовипадкових чисел та дослідження його ефективності. 4. Охорона праці та безпека в надзвичайних ситуаціях. Висновки.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, слайдів)
1. Структурна схема національної платіжної системи
2. Класифікація загроз інформаційних ресурсів
3. Класифікація засобів криптографічного захисту інформації
4. Генератори псевдовипадкових чисел
5. Методи формування псевдовипадкових чисел
6. Результати досліджень статистичної безпеки відомих генераторів

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
4.1 Охорона праці	Осухівська Г.М.		
4.2 Безпека в надзвичайних ситуаціях	Клепчик В.М.		

7. Дата видачі завдання _____

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Аналіз теми та завдань	14.09-29.09.2021	
2	Ознайомлення з принципами роботи захисту інформаційних систем	30.09-12.10.2021	
3	Аналіз засобів захисту банківських інформаційних систем	14.10-19.10.2021	
4	Дослідження методів формування чисел на основі блокових кодів	20.10-30.10.2021	
5	Розробка програмного генератора псевдовипадкових чисел	02.11-24.11.2021	
6	Дослідження ефективності програмного генератора	25.11-30.11.2021	
7	Охорона праці та безпека в надзвичайних ситуаціях	01.12-05.12.2021	
8	Оформлення кваліфікаційної роботи	06.12-13.12.2021	
9	Перевірка на антиплагіат	14.12.2021	
10	Захист кваліфікаційної роботи	22.12.2021	

Студент _____ Баняс Б.М.
(підпис) (прізвище та ініціали)

Керівник роботи _____ _____
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Пояснювальна записка до магістерської дипломної роботи: 109с., 34 рис., 6 табл., 70 джерел.

Об’єктом дослідження є процес формування псевдовипадкових чисел в криптографічних засобах захисту банківських інформаційних систем.

Предметом дослідження є метод формування псевдовипадкових чисел на основі надлишкових блокових кодів в криптографічних засобах захисту банківських інформаційних систем.

Метою роботи є підвищення стійкості криптографічних засобів захисту банківських інформаційних систем на основі використання методів формування псевдовипадкових чисел.

Методами розробки обрано: при аналізі методів і алгоритмів генерації псевдовипадкових чисел використані методи теорії захисту інформації. При дослідженні статистичної безпеки та оцінці швидкодії генерації псевдовипадкових чисел на основі надлишкових блокових кодів використані методи математичної статистики.

В результаті роботи проведено аналіз сучасних методів формування псевдовипадкових чисел, розглянуто вдосконалений метод формування псевдовипадкових чисел на основі на основі надлишкових блокових кодів, проведено дослідження статистичної безпеки розглянутого генератора та приведені можливі варіанти його практичного використання.

ГЕНЕРАТОР ПСЕВДОВИПАДКОВИХ ЧИСЕЛ, НАДЛИШКОВІ КОДИ, ДЕКОДУВАННЯ ВИПАДКОВОГО КОДУ, СТАТИСТИЧНА БЕЗПЕКА.

ABSTRACT

An explanatory message of attestation work: 109 p., 34 pic., 6 tabl., 70 sources.

The object of the research is the process of generating random numbers of cryptographic protection of bank information systems.

The research object is the method of generating random numbers based on block codes in excess of cryptographic protection of bank information systems.

The purpose of work is to improve the stability of cryptographic protection of bank information systems through the use of methods of generating random numbers.

As the development method were chosen: the analysis methods and algorithms to generate random numbers used methods of the theory of information security. A study of statistical assessment of safety and performance generation of random numbers based on redundant block codes used methods of mathematical statistics.

The result of work are the analysis of modern methods of generating random numbers, is considered an improved method of forming the basis of random numbers based on redundant block codes, a study of statistical safety reporting generator and given options for its practical use.

PSEUDORANDOM NUMBER GENERATOR, SURPLUSES CODES, DECODING INCIDENTAL CODE, STATISTICAL SECURITY.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ І СКОРОЧЕНЬ

ПВЧ – псевдовипадкове число

ГПВЧ – генератор псевдопивадкових чисел

ПВП – псевдовипадкова послідовність

МКІ – методи крипто перетворень інформації

RC4 – потоковий шифр

RC5 – швидкий блоковий шифр

RC6 – симетричний блоковий криптографічний алгоритм

RGAIN – генератор “зерно”

RSA – криптографічний алгоритм з відкритим ключем

GPSSD – доказово стійкий ГПВЧ на надмірних кодах

ВСТУП

В сучасних умовах широкого застосування обчислювальної техніки і засобів обміну інформацією поширюються можливості її просочення та несанкціонованого доступу до неї зі злочинною метою. Особливе місце в цих проблемах займає задача захисту економічної інформації, яка набуває всебічного, масштабного характеру та державної ваги. Одним з факторів забезпечення безпеки економічної інформації є захист фінансово-банківської системи. Розвиток засобів, методів і форм автоматизації процесів обробки інформації і ріст обсягів оброблюваних даних у сучасних внутріплатіжних банківських системах роблять інформацію набагато більш уразливою. Тому виникає потреба забезпечення певного рівня безпеки даних. Сучасні механізми забезпечення цілісності й автентичності даних у автоматизованих банківських системах не забезпечують потреби безпеки у повній мірі. Таким чином, дослідження та аналіз існуючих методів формування псевдовипадкових чисел у криптографічних засобах захисту банківських інформаційних систем є актуальним.

Дослідження в магістерській роботі проводилися у відповідності з наступними нормативними актами.

1. Концепція розвитку зв'язку України до 2010 року, затверджена постановою Кабінету Міністрів України “Про Концепцію розвитку зв'язку України до 2010 долі” від 9 грудня 1999 р. №2238.

2. Концепція Національної програми інформатизації схваленої Законом України "Про Концепцію Національної програми інформатизації" від 4 лютого 1998 р. № 75/98-ВР.

3. Державна науково-технічна програма “Створення перспективних телекомунікаційних систем і технологій”.

Мета і завдання дослідження: підвищення стійкості криптографічних засобів захисту банківських інформаційних систем на основі використання методів формування псевдовипадкових чисел.

Для досягнення поставленої мети необхідно вирішити наступні задачі:

- проаналізувати відомі методи генерації псевдовипадкових чисел в криптографічних засобах захисту банківських інформаційних систем;
- дослідити метод генерації псевдовипадкових чисел на основі надлишкових блокових кодів;
- дослідити статистичну безпеку та оцінити швидкодію розробленого генератора псевдовипадкових чисел та порівняти з програмними реалізаціями відомих генераторів.

Об’єктом дослідження є процес формування псевдовипадкових чисел в криптографічних засобах захисту банківських інформаційних систем.

Предметом дослідження є метод генерації псевдовипадкових чисел на основі надлишкових блокових кодів в криптографічних засобах захисту банківських інформаційних систем.

При аналізі методів і алгоритмів генерації псевдовипадкових чисел використані методи теорії захисту інформації. При дослідженні статистичної безпеки та оцінці швидкодії генерації псевдовипадкових чисел на основі надлишкових блокових кодів використані методи математичної статистики.

Теоретична значущість полягає в удосконаленій методиці формування псевдовипадкових послідовностей методом GPSSD шляхом збільшення довжини періоду формованих послідовностей псевдовипадкових чисел.

Практична цінність результатів магістерських досліджень становить дослідження статистичної безпеки та швидкодії генератора псевдовипадкових чисел на основі надлишкових блокових кодів.

Наукове значення полягає у проведенні аналізу існуючих методів генерації псевдовипадкових чисел в криптографічних засобах захисту банківських інформаційних систем.

Достовірність отриманих результатів обґрунтовується їх несуперечністю основним положенням алгебраїчної теорії кодів, статистичної теорії зв'язку і теорії інформації.

РОЗДІЛ 1.
АНАЛІЗ ВІДОМИХ МЕТОДІВ ФОРМУВАННЯ ПСЕВДОВИПАДКОВИХ ЧИСЕЛ У КРИПТОГРАФІЧНИХ ЗАСОБАХ ЗАХИСТУ БАНКІВСЬКИХ ІНФОРМАЦІЙНИХ СИСТЕМ

У розділі проводиться дослідження прикладних питань теорії захисту інформації, пов'язаних з формуванням псевдовипадкових чисел. На основі досліджень критеріїв та показників ефективності генераторів псевдовипадкових чисел, а також аналізу відомих генераторів, обґрунтовується вибір напрямку досліджень і математично формалізується постановка наукової задачі.

1.1. Дослідження прикладних питань теорії захисту інформації, пов'язаних з формуванням псевдовипадкових чисел

В умовах зростання інформатизації суспільства, застосування засобів обчислювальної техніки та комп'ютерних систем набувають актуальності питання інформаційної безпеки, серед яких необхідність захисту цінної конфіденційної та секретної інформації на підприємствах державного і приватного рівня, в органах державного управління, банківських системах та інших. Збільшення обсягів даних, що оброблюються і передаються в комп'ютерних системах і мережах, в банківських системах в першу чергу, вимагають нові підходи застосування протоколів і механізмам, що забезпечують безпеку передавання даних [3, 5, 8, 9, 19, 25, 36].

Національна платіжна система – складна система багаторівневого централізованого управління для забезпечення якісного стратегічного каналу проведення фінансових транзакцій [9, 32]. Структурна схема національної платіжної системи наведена на рис. 1.1.

Так як дана система належить до складних багаторівневих систем управління критичного застосування, то існує необхідність надання

застосовуваним у них програмним засобам заданих властивостей безпеки і здатності протистояти руйнуванню, порушень функціонування системи, збоїв, навмисним діям зловмисників і помилок різних видів при виконанні критичної системою основної цільової функції [3, 5 19, 32].

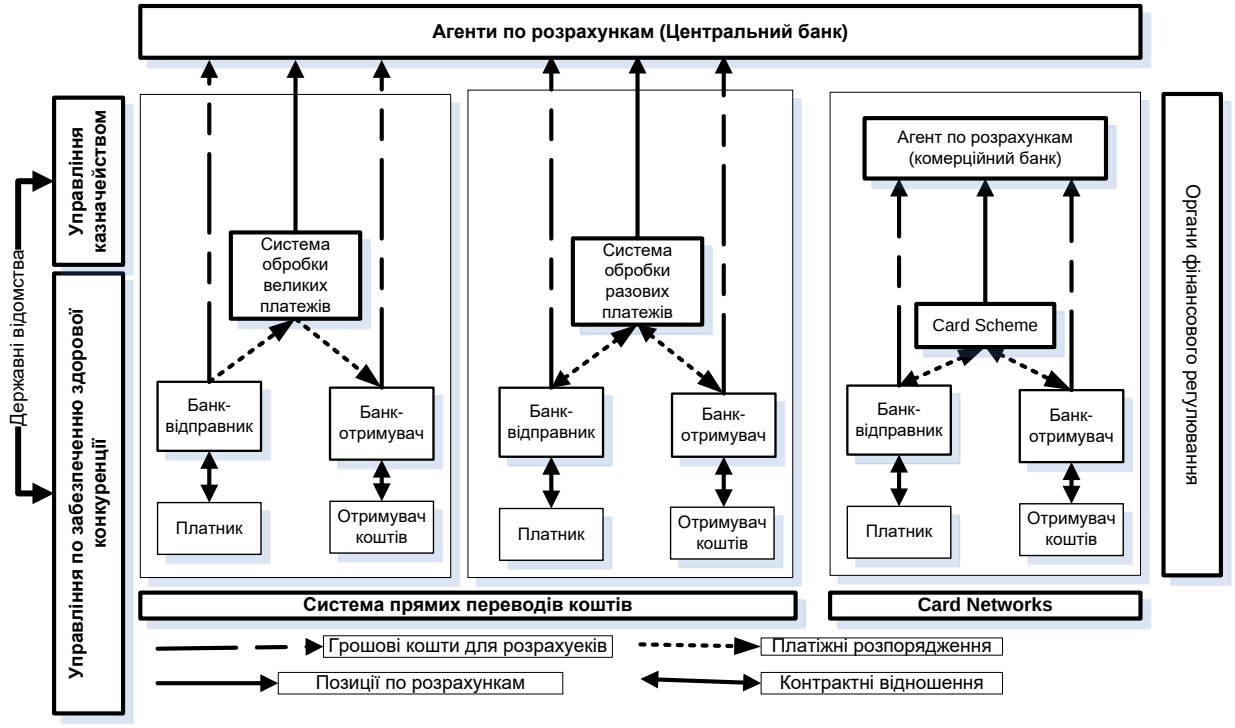

Рис. 1. 1. Структурна схема національної платіжної системи

Не дивлячись на застосування будь-яких криптографічних алгоритмів на різних рівнях захисту ці системи піддаються різним атакам і загрозам, що поділяються на загрози фінансових ресурсів і загрози інформаційних ресурсів, які поділяються на зовнішні (технічні) і внутрішні (неправомірні дії співробітників) [12, 13, 15].

Загроза безпеці інформаційної системи є можливий вплив на цю систему, що може завдати шкоди її безпеці. Збиток безпеки порушує стан захищеності інформації, яка знаходиться і обробляється в інформаційній системі.

На рис. 1.2 представлені основні типи загроз інформаційних ресурсів.

Найпоширенішими і привабливими цілями для зловмисників є клієнтські програми. У 2019 році в клієнтських додатках було виявлено 9 (1,82%) вразливостей критичного степеня небезпеки, 228 (46,06%) вразливостей високого степеня небезпеки, 99 (20%) - середнього, і 159 (32,12%) - низького степеня небезпеки. При цьому в серверних застосуваннях більшість вразливостей низькою і середнього степеня небезпеки (45,42% і 44,32% відповідно), високою - 10,26% і жодної критичної уразливості степеня небезпеки.

Рис 1.2. Типи загроз інформаційних ресурсів

Найпоширенішими типами впливу на сьогоднішній день є: компрометація системи, неавторизоване зміна даних, міжсайтовий скріптінг і відмова в обслуговуванні. Всі типи впливів відображені на рис.1.3.

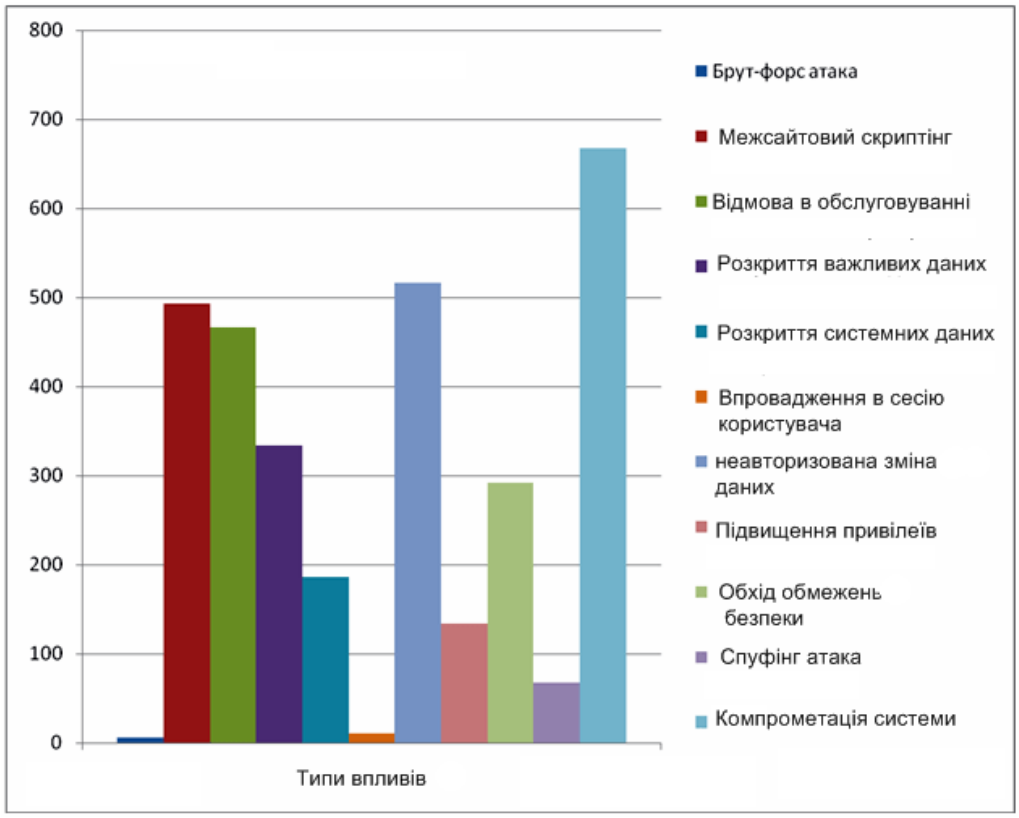

Рис. 1.3. Типи впливів вразливостей

Степінь небезпеки вразливостей, яка виставляється для кожного повідомлення безпеки, залежить від типу впливу на програму або систему, наявності виправлення або тимчасового рішення, представленого виробником, наявності експлоїта і можливості масової експлуатації уразливості. Зазвичай використовується 4 степеня оцінки небезпеки вразливостей: критична, висока, середня і низька.

У 2019 р. більшість вразливостей представляли середню та низьку (43,81% і 37,37% відповідно) степінь небезпеки, 18,41% вразливостей представляв високу і 0,41% критичну степінь небезпеки.

Щоб забезпечити захисту від розглянутих вище загроз застосовують різні криптографічні механізми. Для побудови механізмів інформаційної безпеки використовують традиційно методи криптографічного оброблення інформації. В першу чергу методи симетричної та несиметричної криптографії. Симетричні методи створюються на простих і легких в реалізації блоках підстановок і перестановок, методи несиметричної криптографії на використанні теоретико-складної задачі (факторизації, дискретного логарифмування та ін.) [5, 7, 15, 16, 19].

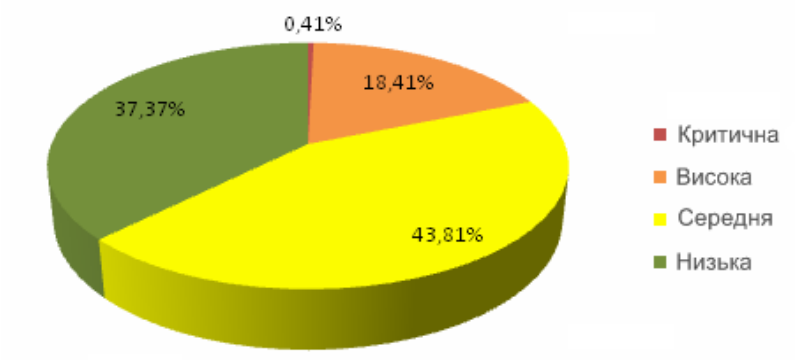

Рис. 1.4. Ступінь небезпеки вразливостей в 2019 р.

З розвитком сучасних механізмів забезпечення інформаційної безпеки систем і технологій важливе місце займає використання випадкових чисел та генераторів псевдовипадкових чисел. Вони вирішують наступні завдання: хешують інформації; будують синхроні і самосинхронізуючі потокові шифри; формують ключові дані і т.д. [22, 58, 59, 61].

Випадкові числа знаходять широке застосування у прикладних науках. Важливу роль вони відіграють при використанні криптографії в різних мережевих додатках, що відносяться до безпеки. Певні алгоритми, засновані на засобах криптографії, використовують випадкові числа. Прикладами таких алгоритмів можуть служити наступні [20, 21, 23, 58]:

- генерування сеансових ключів;

- схема взаємної ідентифікації. Сценарії розподілу ключів використовують оказії, щоб виключити можливість атаки на основі відтворення повідомлень. Використання випадкових чисел не дозволяє противнику визначити значення оказії;
- генерування ключів для алгоритму шифрування RSA з відкритим ключем [20, 28, 56, 58, 59].

Характеристики систем залежать від функцій криптографічних підсистем, котрі визначені не тільки алгоритмікою, але і показниками випадкових чисел які використовуються [20, 28].

Так як безпека криптосистеми основана на ключі, то використовуючи мало надійний процес генерування ключів, вся криптосистема так само вразлива.

Основні вимогами до послідовності випадкових чисел являється непередбачуваність і випадковість.

Для утворення послідовностей псевдовипадкового числа передбачають, вона має бути випадкова в деякому певному статистичному сенсі. Наступні два критерії використовуються для доказування, що послідовність чисел є випадковою:

- однорідний розподіл: розподіл чисел у послідовності має бути однорідним; це означає, що частота появи кожного числа має бути приблизно однаковою.
- незалежність: ні одне значення в послідовності не повинно залежати від інших.

В додатках, таких як взаємна аутентифікація та генерація ключа сесії, немає жорсткої вимоги, щоб послідовність чисел була статистично випадковою, але члени послідовності повинні бути непередбачувані. За реальної випадкової послідовності кожне число статистично не залежить від інших чисел і, отже, непередбачуваним. Однак правильні випадкові числа на практиці використовуються досить рідко, частіше послідовність чисел, яка повинна бути

випадковою, створюється деяким алгоритмом. В даному випадку необхідно, щоб порушник не міг передбачити наступні елементи послідовності, ґрунтуючись на знанні попередніх елементів і використовуваного алгоритму [58 – 59].

Генератори випадкових чисел (ГВЧ) за способом отримання чисел поділяються на (рис 1.5):

- фізичні;
- табличні;
- алгоритмічні.

Фізичний генератор випадкових чисел – прилад генеруючий послідовності випадкових чисел беручи за основу параметри протікання фізичного процесу які вимірюються. Прикладом фізичних ГВЧ можуть служити: гральні кості, поділений на сектори з цифрами барабан зі стрілкою, генератори шумів фізичні, подібні на детектори подій іонізуючої радіації, розрядні газові трубки і конденсатор який має течію. Дані процеси теоретично абсолютно непередбачувані. Генератори випадкових чисел створені на квантових процесах включають в себе спеціальний підсилювач та перетворювач. Перший посилює слабкі сигнали, які отримуються результатом проходження фізичних явищ, до необхідних розмірів, перетворених перетворювачем.

Рис. 1.5. Класифікація генераторів псевдовипадкових чисел за способом отримання чисел

Апаратні генератори випадкових чисел не виділяються високою швидкістю та виробляють при нагоді зміщені послідовності. Використання таких генераторів напряму пов’язане з потребами конкретної предметної області та від пристрою самого генератора [56].

Недолік спостережень фізичних явищ – їх велика вартість в порівнянні з математичними генераторами, також можуть знадобитися зусилля з боку користувача. До того ж немає ніякої доказовою випадковості. Можливі також проблеми з балансом між частотою з'являються значень, їх кореляцією. Так само ці пристрої в додатках мережевої безпеки застосовуються обмежено. Проблеми викликають грубі атаки на такі пристрої [20, 22, 53, 56].

Альтернативним рішенням є табличні ГВЧ, які в якості джерела випадкових чисел використовують спеціальним чином складені таблиці, що містять перевірені некорельовані, тобто ніяк не залежні один від одного, цифри. Дані набори забезпечують обмежене джерело чисел порівнюючи з кількістю необхідною додаткам мережевої безпеки [53, 56].

Тому криптографічні програми зазвичай використовують алгоритмічні методи генерування випадкових чисел.

Ці алгоритми є детермінованими і, отже, передбачувані. Комп'ютери не можуть самі по собі генерувати дійсно випадкові числа. Тим не менше, коли алгоритм надійний, послідовність яка отримана проходитиме немало тестів на випадковість. Будь-які програмні ГВЧ, що не користуються зовнішніми «джерелами ентропії» і використовують тільки алгебраїчні перетворення для формування чергового числа, не дають чисто випадкових чисел. На виході такого ГВЧ послідовність показує випадковою, але дійсно підпорядковується

закону і, рано чи пізно зациклиться. Числа називаються псевдовипадковими [56 – 58].

Генератор псевдовипадкових чисел (ГПВЧ) – алгоритм, котрий генерує певну послідовність чисел, чиї елементи не залежать один від другого і підлягають заданому розподілу.

Для визначення вимог до ГПВЧ, а також для математичної формалізації постановки наукової задачі, проведемо дослідження критеріїв та показників ефективності генераторів псевдовипадкових чисел.

1.2. Обґрунтування критеріїв та показників ефективності генераторів псевдовипадкових чисел

Проведення тестувань генераторів випадкових і псевдовипадкових чисел (ГВЧ і ГПВЧ), які використовуються в криптографічних додатках, є актуальними задачами як у практичному, так і в теоретичному плані. Незважаючи на велику кількість напрацювань в даній галузі, у розробників, є потреба в придатному інструментарії, спроможному приділити прийнятну метрику, котра позволить досліджувати ступінь випадковості послідовностей, які породжується ГВЧ (ГПВЧ). Також, надати розробникам достатню кількість даних з метою ухвалити рішення щодо «якості» генератора [21 – 23, 29].

На теперішній час розроблено багато різних типів ГВЧ (ГПВЧ). Для демонстрації їхніх статистичних властивостей застосовувались різні підходи до статистичного тестування. Дуже часто методику тестування та набір пропонувався самим розробником генератора. Отже, ситуація яка склалася характеризується неможливістю об'єктивно порівняти генератори. Розглянемо критерії, яким повинен задовольняти сучасний ГПСЧ [26, 42]:

- великий період послідовності ПВЧ, що формується, $L > 2^{128} - 1$;

- максимальний період $L = L_{max}$, послідовності ПВЧ, що формується $L_{max} = 2^l - 1,$

  де l – довжина секретного ключа, тобто $l > 128$ біт;
- висока скритність $S = 1,$ визначається можливістю протистояти виявленню зловмисником елементів послідовності ПВЧ з відомих (перехоплених) елементів (тобто неможливість рішення криптоаналітиком задачі визначення псевдовипадкової послідовності за відомим фрагментом (визначення $(i-1)$-го елемента γ_{i-1} послідовності на основі відомого фрагменту псевдовипадкової послідовності $\gamma_i \gamma_{i+1} \gamma_{i+2} ... \gamma_{i+b-1}$ кінцевої довжини b, визначення ключової інформації за відомим фрагментом послідовності кінцевої довжини):

$$S = \frac{B}{L},$$

де B – дуже мала величина членів послідовності ПВЧ, потрібних для однозначного відновлення алгоритму формування ПВЧ;
- мала ймовірність розкриття алгоритму формування ПВЧ, що задається секретним ключем

$$P_k = \max\{P_{к1}, P_{к2}, ..., P_{кM}\} \le 2^{-l},$$

де $P_{кi}$ – ймовірність правильного розкриття алгоритму формування ПВЧ за умови використання порушником і-ого методу криптоаналізу;
- не малий безпечний час

$$T_в = \min\{T_{в1}, T_{в2}, ..., T_{вL}\} \ge \frac{2^l}{\gamma \cdot \psi},$$

$$T_{\text{Bi}} = \frac{S_{Bi}}{\gamma \cdot \Psi},$$

де S_{B_i} –тимчасова складність алгоритму, який реалізує і-ий метод крипто аналізу;

γ – const;

Ψ – працездатність обчислювальної системи зловмисника;

- забезпечення статистичної безпеки (за методикою NIST STS);
- висока швидкість формування ПСЧ $V_{\text{їð.}} \approx 10^7 \div 10^9$ біт/с (при програмній реалізації) и $V_{\text{àïï.}} \approx 10^9 \div 10^{10}$ біт/с (при апаратній реалізації).

Забезпечення останньої вимоги можливо при невеликій кількості обчислювальних операцій виконуваних генератором ПВЧ для формування одного біта послідовності ($V \approx 1 \div 10$ операцій/біт).

Однією з основних вимог, що вимагають від криптостійкого генератора є статистична безпека. Для проведення тестування генераторів випадкових і псевдовипадкових чисел можна скористатися такими програмними пакетами [23, 41, 42, 64]:

- NIST Statistical Test Suite;
- TEST-U01;
- CRYPT-X;
- The pLab Project;
- DIEHARD;
- ENT.

На сьогоднішній день методика проведення статистичного тестування ГВЧ (ГПВЧ) з використанням пакета NIST STS найкращим чином відповідає потребам всіх зацікавлених сторін. Рекомендований під час конкурсу національний стандарт США блочного шифрування пакет тестів NIST STS

надає можливість якісно досліджувати псевдовипадкової послідовності. Розглянутий пакет статистичних тестів призначений вирішувати наступні завдання [41, 64]:

- розробка нових ГПВЧ;
- перевірка коректності реалізації ГПВЧ;
- дослідження степеня випадковості використовуваних ГПВЧ.

Пакет NIST STS має 16 тестів розроблених з метою перевірки гіпотези випадковості двійкових послідовностей будь-якої довжини, що генерує ГВЧ або ГПВЧ. Тести направлені виявляти дефекти випадковості. Принципом тестування являється перевірка гіпотези H_0, вона заключається в тому, що тестована послідовність є випадковою. Альтернативна гіпотезою H_a є послідовність, що тестується не випадкова. Результатом виконання тесту нульова гіпотеза приймається, або не приймається (відкидається).

Для остаточного рішення проходження послідовністю псевдовипадкових чисел тесту використовуються три варіанти.

Двійкова послідовність $S = \{s_1, s_2, \ldots, s_n\}$, $s_i \in \{0,1\}$ довжиною n біт. Потрібно прийняти рішення, чи проходить вона статистичний тест. Є наступні варіанти для вирішення цього завдання.

1. Використання критерію на основі завдання порогового рівня для прийняття рішення. Підхід ґрунтується на обчисленні послідовності S статистики тесту c(S), для подальшого порівнянням з граничним рівнем $C_{гран}(S)$. Критерій ухвалення рішення формулюється таким чином: двійкова послідовність S не проходить статистичний тест кожного разу, коли статистика тесту c(S) приймає значення не більше, чим граничний рівень $C_{гран}(S)$.

2. Критерій приймання рішення завданням фіксованого довірчого інтервалу. В цьому випадку критерій ухвалення рішення формулюється так: допускаємо, двійкова послідовність S не проходить тест, значення статистики тесту c(S) лежить не в межах довірчого інтервалу величин статистики, обчисленого для які задано.

Цей критерій надійніший в порівнянні з першим. Необхідно враховувати, що для різним рівнів значущості відповідають різні довірчі інтервали.

3. Наступний підхід побудови критерію обчислення для статистики тесту c(S) відповідної величини ймовірності P. Статистика тесту в даному випадку розглянута як реалізація випадкової величини, що підкорена відомому закону розподілу. Статистка тесту формується так, щоб її не малі значення вказували на будь-який дефект випадковості послідовності. Величина ймовірності P є ймовірність, що статистика тесту набуде значення не менше, ніж видно в припущенні випадковості послідовності. Таким чином, не великі значення P (P <0,05 або P <0,01) визначаються, що послідовність не випадкова. Вирішальне визначення формулюють наступним чином: фіксованому рівню значущості α, послідовність двійкова S не зможе пройти статистичний тест коли величина ймовірності P <α. Величину α вибирати з інтервалу [0,001, 0,01].

Необхідно вказати, що тестування, проведене з використанням пакета NIST STS, проводиться з врахуванням припущень:

- тест, використаний до послідовності застосовується до довільної підпослідовності;
- не правильно про якість генератора робити висновок, використовуючи результати аналізу послідовності одного початкового заповнення.

Фактично, обчислюється 189 значень ймовірності P в залежності від вхідних параметрів. У табл. 1.1 (див. додаток В). наведено дані про всі тести із вказанням величини значень ймовірності P, які обчислюються, також фізичної величини статистики тесту і дефекту, для пошуку якої використовують тест тест [23, 41, 42, 64].

Отже, результат тестування послідовності двійковій формує вектор величин ймовірності P = (P_1, P_2, ..., P_{189}). Аналізуючи складові Pi вектора вказують на конкретні дефекти випадковості тестованих послідовностей.

На основі пакета можуть бути побудовані методики більш глибоко статистичного і структурного аналізу послідовностей. Так, для більш надійної

оцінки генераторів доцільно проводити не одне випробування, а як мінімум три (одне випробування - побудова одного повного статистичного портрета). При повторенні висновків з генератора на основі аналізу кожного з трьох статистичних портретів ступінь невизначеності щодо властивостей генератора істотно зменшиться і надійність рішення збільшиться. Розглянемо найбільш відомі методи формування ПВЧ, визначимо їх переваги і недоліки, обґрунтуємо вибір напрямку подальших досліджень.

1.3. Дослідження відомих методів формування псевдовипадкових чисел у криптографічних засобах захисту банківських інформаційних систем

Формування ПВЧ здійснюється за допомогою відповідних ГПВЧ реалізованих на використанні відомих методів: криптостійкі і некриптостійкі, які можна поділити на класи. Класифікація некриптостійких методів наведена на рис. 1.6.

Рис. 1.6. Некриптостійкі методи формування псевдовипадкових чисел

Широко використовуваним класом некриптостійких генераторів, є конгруентні генератори [7, 22, 28, 5, 57 – 59]. Для генераторів даного класу можна зробити висновок про те, які властивості мають вихідні послідовності генераторів якщо розглядати з точки зору випадковості та періодичності.

Найчастіше на практиці використовуються лінійні конгруентний генератори. Сформовані в таких генераторах послідовності можна представити у вигляді аналітичного виразу:

$$x_i = (ax_{i-1} + b) \bmod m, \tag{1.1}$$

де x_i – і-й елемент псевдовипадковою послідовності;

$a \neq 0$ – множник;

b – приріст;

m – потужність послідовності (модуль).

Період цього генератора менше, ніж m. Якщо a, b і m задані правильно, то генератор буде з максимальним періодом, і період генератора буде рівнятися m (b взаємно просте з m.)

Для лінійної конгруентної послідовності максимальне значення $m_{\max}$ періоду досягається тоді і тільки тоді, коли виконані наступні вимоги:

НОД(b, m)=1;

$m = p_1^{r1} * p_2^{r2} \ldots p_k^{rk} \Rightarrow (\forall i \in \overline{1,k} \Rightarrow a - 1 = l_i * p_i)$, p_i прості числа $\forall i \in \overline{1,k}$;

$m = 4 * l_1 \Rightarrow a - 1 = 4 * l_2$, $l_1, l_2 \in N$.

Лінійні конгруентні генератори не дозволено використати в криптографії, за їх передбачувані. Так само ненадійними є квадратичний генератор:

$$Xn = (aX_{n-1}^2 + bX_{n-1} + c) \bmod m$$

та кубічний генератор:

$$Xn = (aX_{n-1}^3 + bX_{n-1}^2 + c\ X_{n-1} + d) \bmod m.$$

Основними перевагами конгруентних генераторів являються:

- максимальний період послідовності, що формується;

- програмна й апаратна реалізації проста;

- на основі генераторів, що мають властивості, потрібні для рішення прикладних питань інформаційного захисту інформації можливість побудови.

Одним з недоліків генератора є формування псевдовипадкових чисел які мають низьку криптостійкість до різних видів криптоаналізу (кореляційний, інверсний та ін.). Отже, використання конгруентних генераторів можливо для рішення завдань інформаційного захисту як складені елементи криптосхем [57 – 59].

Однак, лінійні конгруентні генератори ефективні у більшості використовуваних емпіричних тестів показують гарні статистичні характеристики.

Іншим класом ГПВЧ є генератори засновані на використанні регістрів зсуву за лінійними зворотними зв'язками (РЗЛЗЗ). Вони складаються з двох частин: функції зворотного зв'язку і регістра зсуву [28, 49, 53]. Регістр зсуву є послідовність бітів. Довжиною зсуву регістра визначається кількість бітів. Коли довжина відповідає n бітам, то регістр є n-бітовим регістром зсуву. При витягуванні біта, всі біти регістру зсуву переміщуються на 1 позицію вправо. Функцією всіх інших бітів регістра є новий крайній лівий біт. Схема регістру зсуву представлена на рис. 1.7.

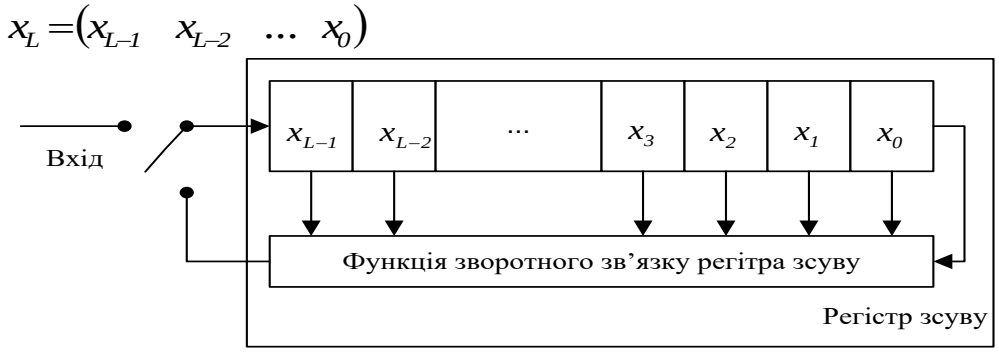

Рис. 1.7. Регістр зсуву зі зворотнім зв'язком

Простим видом регістра зсуву який має зворотній зв’язок являється лінійний регістр зсуву зі зворотним зв'язком (linear feedback shift register, або LFSR). Іноді такий регістр називається конфігурацією Фіббоначі. LFSR частіше інших регістрів зсуву використовується в криптографії. LFSR часто використовуються для створення алгоритмів для шифрування [53, 57]. Схема лінійного регістру зсуву який має зворотній зв’язок подано на рис. 1.8.

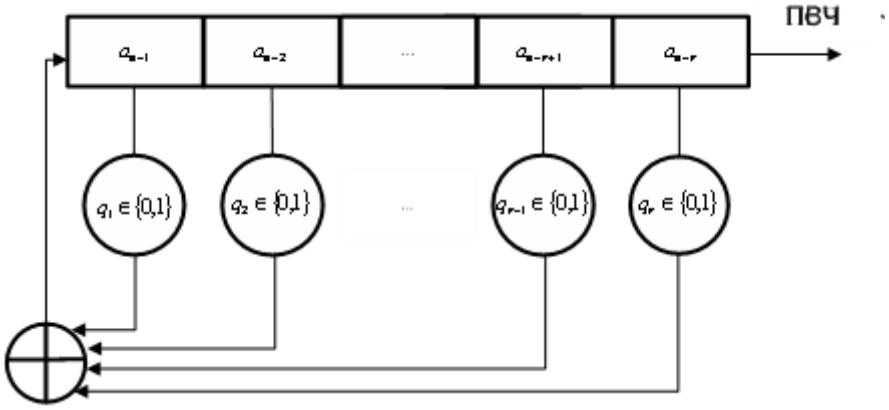

Рис. 1.8. Лінійний регістр зсуву

Також до не криптостійких методів формування псевдовипадкових чисел відносяться адитивні генератори (запізнюючі генератори Фіббоначі). Вони є ефективні, їхнім результатом являються випадкові слова, а не біти випадкові біти. Вони не є безпечними, але використовуються як блоки для безпечних генераторів .

Початковий стан генератора являє собою масив n-бітових слів: 8-бітових слів, 16-бітових слів, 32-бітових слів, і т.д.: X_1, X_2, X_3,..., X_m. Цей початковий стан і є ключем. і-е слово генератора виходить як:

$$X_i = (X_{i-a} + X_{i-b} + X_{i-c} + + X_{i-m}) \bmod 2^n.$$

При правильному виборі коефіцієнтів a, b, c, ..., m період цього генератора не менше 2^n-1.

Класифікація криптостійких методів наведена на рис. 1.9.

До криптостійких ГПВЧ можна віднести генератори, побудовані на основі потокових шифрів.

Потокові шифри діляться на синхронні та ті що само синхронізуються [52, 57 – 59, 61]. У синхронних потокових шифрах послідовність, що шифрується, генерується незалежно від вхідної послідовності, кожний елемент (біт) якої в такий спосіб шифрується незалежно від інших елементів. В синхронних потокових шифрах відсутній ефект розмноження помилок.

В потокових шифрах, що само синхронізуються, елементи вхідної послідовності зашифровуються з урахуванням попередніх елементів, які беруть участь у формуванні ключової послідовності. У шифрах, що само синхронізуються, має місце ефект розмноження помилок.

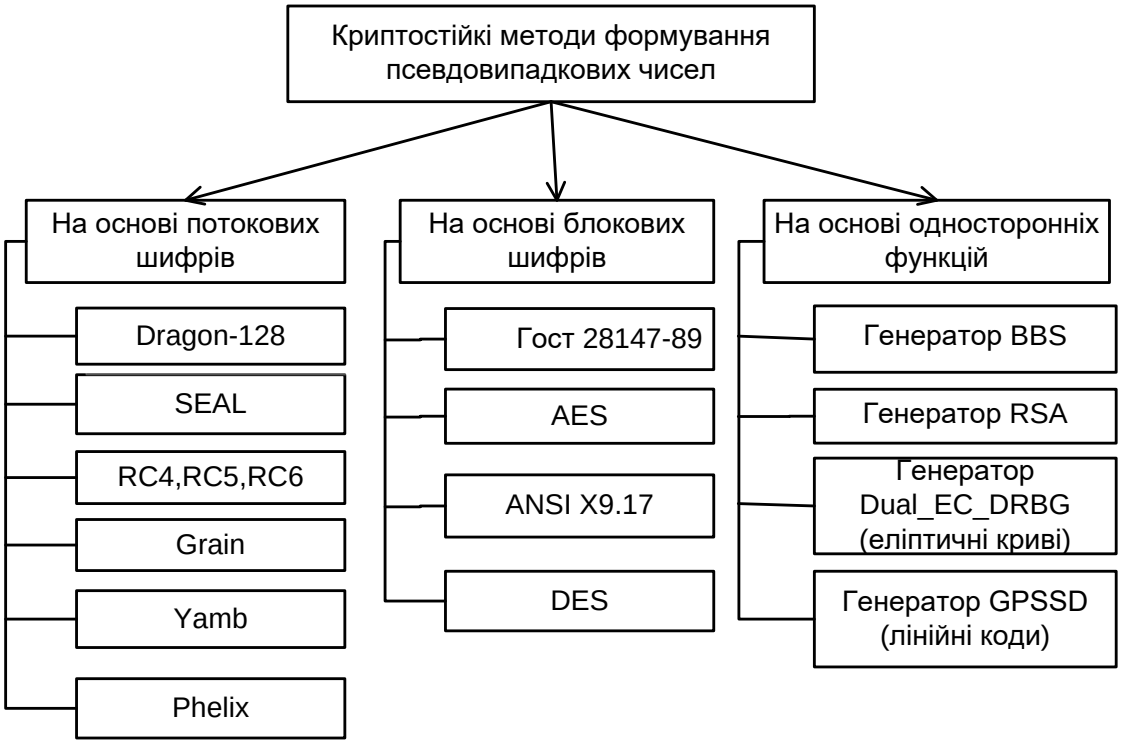

Рис. 1.9. Криптостійкі методи формування псевдовипадкових чисел

Одним із прикладів генератора на основі потокових шифрів є генератор SEAL. Особливість SEAL в тому, що є дійсності є не традиційним потоковим шифром, а являє собою сімейство псевдовипадкових функцій.

Практично, що SEAL сімейство псевдовипадкових функцій, він зручний у деяких додатках, де не використовуються потокові шифри. При використанні потокових шифрів, створюється односпрямована послідовність бітів: одним їз способів визначення і-ого біта, зі знанням ключа і позиції, є генерування бітів аж до і-ого. Відмінність в тому, що можна без проблем отримати доступ до будь-якої позиції потоку ключів.

Основним перевага ГПВЧ побудованих на потокових шифрах висока швидкість перетворення, порівнюючи її з надходженням вхідної інформації. Таким чином, забезпечується формування ПВЧ у масштабі реального часу [57 – 59].

Недоліків є необхідність синхронізації на приймальній передавальній стороні.

Наступним класом криптостійких генераторів є ГПВЧ побудовані на блокових шифрах [7, 20, 28, 49, 53]. Робота таких генераторів полягає в застосуванні до блоку відкритого тексту багаторазового математичного перетворення. Багаторазовість застосування спричиняє те, що результуюче перетворення виявляється криптографічно більше складним, чим саме перетворення. Основна мета здійснюваних перетворень – це створити залежність кожного біта блоку зашифрованого повідомлення від кожного біта ключа й кожного біта блоку відкритого повідомлення. Перетворення, що лежать в основі даних алгоритмів можна розділити на "складні" перетворення, у сучасних алгоритмах це звичайно нелінійні операції, і "прості" перетворення, в основі яких лежать операції, що перемішують. Аналітична складність

розкриття алгоритмів блокового шифрування лежить в основному на конструкції першого типу перетворень.

Перевага ГПВЧ побудованих на блокових шифрах є: гарні статистичні властивості псевдовипадкової послідовності яка формується та стійкість до різних видів криптоанализу (кореляційний, інверсний і ін.) [28, 49, 52, 61].

Недоліком блокового шифрування є:

- слабка чутливість криптосхем до випадання чи вставки цілої кількості блоків;
- існування проблеми крайнього блоку неповної довжини.

Особливим напрямком у розвитку криптостійких генераторів є методи, що допускають можливість застосування моделі доказовою стійкості. До них належать методи, засновані на вирішенні односторонніх функцій [11, 28].

Функція $F:\{0, 1\}^n \rightarrow \{0, 1\}^m$ називається односторонньою функцією, якщо:

- функцію F можна обчислити за поліноміальний час;
- не існує поліноміального алгоритму, що вірно обчислює $F-1$ с достатньою ймовірністю;
- існує предикат $h:\{0, 1\}^n \rightarrow \{0, 1\}$, т.ч. по F (x) трудно обчислити h(x).

На сьогоднішній день теорія алгоритмів не дозволяє довести не існування ефективних алгоритмів рішення того або іншого завдання. Тому, строго говорячи, не відома одностороння функція. Однак запропоновано кілька функцій, які можуть виявитися односторонніми - для цих функцій , незважаючи на інтенсивні дослідження, не відомі ефективні алгоритми інвертування [11, 21 – 23].

Найбільш часто використаються "односторонні" функції, запозичені з теорії чисел:

- функція $F(a,b) = a \times b$, тобто добуток двох чисел. Якщо a і b — прості числа, то по відомому $c = a \times b$ можна однозначно визначити a і b. Ця задача називається задачею факторизації числа. Дотепер не відомий жоден

поліноміальний алгоритм для рішення завдання факторизації, хоча для обчислення добутку чисел (тобто самої функції F) такі алгоритми відомі.

- функція $F(a, n) = a^n \pmod M$, де a, n и M — цілі числа. При відомих a, n і M значення $b = a^n \pmod M$ може бути обчислене за поліноміальну кількість кроків. Однак для зворотного завдання – визначити n по відомим a, b та M (завдання дискретного логарифмування) – поліноміальні алгоритми, у загальному випадку, поки не відомі [11].

Не будь-яка одностороння функція не може бути використана для шифрування. Для використання в криптографії необхідно, щоб завдання інвертування перетворення, що шифрує (тобто обчислення t по F(t)) було розв'язне за прийнятний час, але зробити це може тільки той, хто знає секретний ключ. Такі функції називаються односторонніми функціями із секретом (або з потайним ходом).

Одностороння функція із секретом - це функція $F_k: X \rightarrow Y$, що залежить від параметра k (цей параметр називається секретом), для якої виконуються наступні умови:

- при будь-якому k існує ефективний алгоритм, що обчислює $F_k(x)$ для будь-якого x;

- при невідомому k не існує ефективного алгоритму інвертування функції F_k;

- при відомому k існує ефективний алгоритм інвертування функції F_k.

Для криптографії практичних цілей побудовано функції, які можуть виявитися із секретом. Для них друга властивість не доведена, але вважається, що завдання розщеплювати еквівалентне досліджуваному важкому математичному завданню. Самою популярною є теоретико-числова функція, за допомогою якої побудовано шифр RSA [11, 23].

Застосування функцій із секретом у криптографії дозволяє:

- обмін закритими повідомленнями з використанням відкритих каналів зв'язку, тобто для попереднього обміну ключами відмовитися від секретних каналів зв'язку;

- в завдання розкриття шифру включити важку математичну задачу, яка підвищує обґрунтованість стійкості шифру;

- вирішувати нові криптографічні задачі, відмінні від шифрування.

Генератори, засновані на рішенні односторонніх функцій, називаються доказово стійкими генераторами. До доказовостійких генераторів відносяться ГПВЧ BBS і RSA.

Генератор BBS (Blum-Blum-Shub), стійкість якого ґрунтується на теоретико - складному завданні обчислення примітивних квадратних коренів по модулі числа Блюма, еквівалентної по обчислювальній складності завданню факторизації (розкладання числа на співмножники).

Істотним недоліком таких генераторів є висока обчислювальна складність, що визначається, насамперед, великою розрядністю чисел, над якими необхідно виконувати математичні операції, що істотно знижує швидкість формування ПВЧ у порівнянні з генераторами, заснованими на блокових або потокових шифрах [20, 28, 57 – 59].

Перспективний напрямком розвитку формування ПВЧ є дослідження розробка ГПВЧ побудованих на декодуванні випадкового коду GPSSD (Pseudo-Random Generator Provably as Secure as Syndrome Decoding), де завдання криптоанализу фактично зводиться до рішення теоретико-складного завдання синдромного декодування [26, 30, 31, 62, 67].

Крім того, застосування ГПВЧ заснованих на проблемі синдромного декодування дозволяє скоротити час формування ПВЧ у порівнянні з доказово стійкими генераторами.

Проведемо дослідження генератора псевдовипадкових чисел на основі надлишкових кодів, досліджуємо перспективні напрямки його розвитку.

1.4. Дослідження ефективності генераторів псевдовипадкових чисел на основі надлишкових блокових кодів, обґрунтування вибору напрямку досліджень та постановка наукової задачі

Основна ідея генератора на основі надлишкових блокових кодів полягає у використанні алгебраїчного блокового коду з легко реалізованими алгоритмами кодування та декодування [26, 30, 31]. Використовуючи маскування алгебраїчного коду під випадковий, для зловмисника задача декодування є обчислювально складна. Не знаючи правило маскування, супротивник буде змушений застосовувати складний декодер випадкового коду, процес декодування-кодування, еквівалентний односторонній криптографічній функції, в якій складність декодування, росте по експоненті з залежністю від величини коду або виправляючої його здатності [30, 31, 65].

Формалізовано постановку задачі дослідження сформулюємо як завдання підвищення швидкодії доказово стійких генераторів ПВЧ, що задовольняють системі обмежень на окремі показники стійкості:

$$\min(V):\left\{\begin{array}{l} L>2^{128}-1,\ L=L_{\max}=2^{l}-1,\ S=\dfrac{B}{L}=1, \\ P_{k}=\max\left\{P_{к1},P_{к2},...,P_{кM}\right\}\le 2^{-l},T_{в}=\min\left\{T_{в1},T_{в2},...,T_{вL}\right\}\ge\dfrac{2^{l}}{\gamma\cdot\Psi}\end{array}\right\}$$

У ході досліджень був проведений аналіз статистичних властивостей добре відомих ГПВЧ і генератора GPPSD заснованого на проблемі синдромного декодування [64]. Результати досліджень представлені в табл. 1.2.(див. додаток В). Отримані результати показали, що найбільші показники статистичної безпеки показали такі ГПВЧ як: лінійний конгруентний генератор, BBS, GPSSD.

Проведені дослідження ефективності генератора на надлишкових кодах (GPSSD), показали, що поряд з високими показниками статистичної безпеки і високою швидкістю формування, даний генератор має один істотний недолік: період сформованої послідовності не є максимальним, що не задовольняє одній з основних вимог до кріптогрфічно стійкого генератора. Очікуваний період формованої послідовності повинен був скласти $2^{24}-1$ біт, але насправді він виявився на 5 порядків нижче максимального, що потенційно може привести до появи криптографічних атак.

Таким чином перспективним в розвитку доказово стійких ГПВЧ є застосування методів надлишкового кодування, з метою відновлення секретного правила формування ПВЧ розв'язанням теоретико-складної задачі.

Висновки до розділу 1

1. Проведений аналіз показав, що основними вимогами, що висуваються до криптостійких генераторів ПВЧ є: криптографічна стійкість, великий період формованої ПВП, статистична безпека за методикою NIST STS [48], ефективна апаратна й програмна реалізації.

2. В ході дослідження встановлено, що особливим напрямком у розвитку криптостійких генераторів є методи, що допускають застосування моделі доказової безпеки. До них належать генератори, засновані на використанні односторонніх функцій. Але основним недоліком таких генераторів є їх висока складність. Проведені дослідження показали, що швидкодія доказова стійких генераторів на 3-4 порядку нижче в порівнянні з генераторами заснованих на потоковими або блоковими шифрами.

3. Одним з перспективних напрямків, у розвитку доказово стійких генераторів, є методи, засновані на використанні надлишкових кодів. Розробка й застосування доказове стійких генераторів на основі надлишкових кодів

дозволяє скоротити час формування ПВЧ у порівнянні з відомими доказове стійкими генераторами.

4. Проведені дослідження ефективності відомого генератора на надлишкових кодах (GPSSD), показали, що даний генератор володіє істотним недоліком: період формованої послідовності не є максимальним, але має високі показники статистичної безпеки й високу швидкість формування ПВП. Таким чином, важливим напрямком подальших досліджень є розробка методів і обчислювальних алгоритмів для підвищення швидкодії формування ПСЧ і забезпечення максимального періоду формованої послідовності ГПВЧ заснованих на надлишкових блокових кодах.

РОЗДІЛ 2.
РОЗРОБКА МЕТОДУ ФОРМУВАННЯ ПСЕВДОВИПАДКОВИХ ЧИСЕЛ НА ОСНОВІ НАДЛИШКОВИХ БЛОКОВИХ КОДІВ

Проведений огляд та порівняльні дослідження відомих методів формування ПВЧ, дослідження ефективності існуючих генераторів показали, що перспективним напрямком подальшого розвитку є методи формування ПВЧ на основі надлишкових блокових кодів. Завдання криптоаналізу подібних генераторів зводиться до вирішення теоретико-складного завдання декодування випадкового коду, а відповідні методи формування ПВЧ відносять до групи доказовою стійких генераторів.

В даному розділі розглядаються основні положення теорії алгебраїчного надлишкового кодування, і формулюється теоретико-складне завдання декодування випадкового коду, що лежить в основі теоретичного обґрунтування криптографічної стійкості синтезованого генератора.

2.1. Основні положення алгебраїчної теорії надлишкового кодування і формулювання теоретико-складного завдання декодування випадкового коду

Введемо основні терміни і визначення, які використовуються в теорії надмірного кодування [6, 14, 18, 24, 34, 39, 50, 60], розглянемо основні аналітичні співвідношення і сформулюємо теоретико-складне завдання декодування випадкового коду.

Скінченим полем або полем Галуа називають поле, яке складається з скінченого числа елементів. Позначимо таке поле $GF(q)$, де q – кількість елементів поля.

Властивості поля:

- існує дві операції, що використовуються для комбінування елементів: множення і складання;

- результатом множення або складання двох елементів є третій елемент, що лежить в цьому полі;

- поле завжди містить мультиплікативну групу (1, 2, 3, ..., q) і мультиплікативну одиницю 1, адитивну групу (0, 1, ..., q-1) і адитивну одиницю 0 для будь-якого елемента a;

- кінцеві поля існують коли кількість елементів є простим числом. Для кожного допустимого значення q існує рівно одне поле.

Розглянемо векторний простір $GF^n(q)$ – як множину n-послідовностей елементів з $GF(q)$ з компонентним складанням та множенням на скаляр. Під векторним простором M(P) над полем P розуміємо непусту множину M, на якій введено операції:

- складання, тобто кожній парі елементів множини x,y що належать M ставиться у відповідність елемент тієї ж множини, що позначається x+y;

- множення на скаляр (тобто елемент поля P), тобто будь-якому елементу λ, що належить P та будь-якому елементу x, що належить M ставиться у відповідність елемент з M(P).

Елементи множини M називають векторами, а елементи поля P – скалярами.

Введемо деякі поняття з теорії кодування:

Мета кодування інформації це контроль (виявлення і виправлення) помилок, під час передавання повідомлення по каналу з шумами. Кодуючий пристрій вносить надмірність (перевірочну частину довжини r, $r = n - k$) для контролю помилок у дані, що передаються. Приймач, після аналізу властивостей частини, що перевіряється на відповідність переданим даними, декодер зменшує помили, котрі виникають при передаванні.

Лінійний (n, k, d) код V довжини n і рангу k є лінійним підпростором $GF^k(q)$ у просторі $GF^n(q)$ розмірності k векторного простору, тобто не порожня безліч n-послідовностей (кодових слів) над $GF(q)$.

Відстанню по Хеммінга між 2-ма q-іншими послідовностями x та y довжини n називається число позицій, в яких вони різні. Ця відстань позначається через d - мінімальна кодова відстань (мінімальна вага ненульового кодового слова).

Величину $R = \frac{k}{n}$ називають відносною швидкістю коду, а величину $\delta = \frac{d}{n}$ – відносною мінімальною кодовою відстанню.

Нехай вектори $x_1 = (x_{11},\ldots,x_{1n})$, $x_2 = (x_{21},\ldots,x_{2n}),\ldots,x_k = (x_{k1},\ldots,x_{kn})$ є базисом лінійного простору $GF^n(q)$, тобто лінійний код як лінійний підпростір $GF^n(q)$ однозначно задається набором базисних векторів - породжувальною матрицею G коду V розмірності $k \times n$. Будь-яке кодове слово є лінійною комбінацією рядків з G. Для кодування може використовуватися будь-яка взаємно однозначна відповідність інформаційних k – послідовностей і n – послідовностей кодових слів, що задають відображення

$$\varphi : GF^k(q) \to GF^n(q).$$

Якщо k – розрядне інформаційне слово

$$I = (I_0 \quad I_1 \quad \ldots \quad I_{k-1});$$

$$I_i \in GF(q);\ i = 0,1,\ldots,k-1,$$

то n - розрядне кодове слово

$$C = (C_0 \quad C_1 \quad \ldots \quad C_{n-1});$$

$$C_i \in GF(q);\ i = 0,1,...,n-1,$$

можемо представити виразом:

$$C = I \cdot G. \qquad (2.1)$$

Для деяких (циклічних) лінійних кодів, що допускають формальний математичний опис многочленами від однієї формальної змінної (поліноміальний опис) відповідність інформаційного й кодового слів зручно задавати через добуток многочленів у кільці многочленів, ідентичному по своїй структурі простору $GF^n(q)$[14, 38, 48]. Многочленом над полем $GF^n(q)$ називається математичний вираз:

$$f(x) = f_{n-1}x^{n-1} + f_{n-2}x^{n-2} + ... + f_1x + f_0,$$

де x називається невизначеною змінної, коефіцієнти f_{n-1} ... f_0 належать полю $GF^n(q)$ і є цілими числами.

У загальному вигляді породжуючий многочлен дорівнює:

$$f(x) = a_0 + a_1x + a_2x^2 + ... + a_{n-1}x^{n-1}.$$

Елементами поля є всі многочлени, отримані шляхом підстановки характеристики поля в породжуючий многочлен.

Дійсно, циклічний код є окремим випадком підпростору, що володіє додатковою властивістю циклічності. Кожний вектор з $GF^n(q)$ можна представити многочленом від формальної змінної x степеня не вище $n-1$. Компоненти вектора ототожнюються з коефіцієнтами многочлена.

Множина многочленів має структуру векторного простору, ідентичній структурі простору $GF^n(q)$, а так само структуру кільця многочленів

$$GF(q)[x]/(x^n - 1).$$

В кільці многочленів визначене множення над многочленами:

$$p_1(x) \cdot p_2(x) = R_{x^n-1}[p_1(x) \cdot p_2(x)],$$

де $R_b[a]$ – залишок від ділення многочлена a на многочлен b.

Циклічне зрушення в термінах алгебри многочленів запишеться у вигляді

$$x \cdot p(x) = R_{x^n-1}[x \cdot p(x)].$$

Якщо кодові слова (n, k, d) коду над $GF(q)$ задаються у вигляді многочленів, то код V є підмножиною кільця

$$GF(q)[x]/(x^n - 1).$$

Код V є циклічним, якщо разом з кодовим словом $C(x)$ він містить також многочлен $x \cdot C(x)$.

Єдиний приведений ненульовий многочлен $g(x)$ найменшого степеня

$$r = n - k$$

однозначно задає (n, k, d) циклічний код над $GF(q)$ і позначається многочленом, що породжує, причому

$$g(x) = \prod_i (x - \beta^i),$$

де $\beta^i \in GF(q^m)$.

Аналогічно виразу (2.1), визначимо відповідність інформаційного многочлена

$$I(x) = I_0 + I_1 \cdot x + ... + I_{k-1} \cdot x^{k-1},\ \deg(I(x)) = k-1,$$

$$I_i \in GF(q),\ i = 0,1,...,k-1,$$

і кодового многочлена

$$C(x) = C_0 + C_1 \cdot x + ... + C_{n-1} \cdot x^{n-1},\ \deg(C(x)) = n-1,$$

$$C_i \in GF(q),\ i = 0,1,...,n-1,$$

наступним виразом:

$$C(x) = I(x) \cdot g(x). \qquad (2.2)$$

Якщо

$$g(x) = g_0 + g_1 \cdot x + ... + g_{n-k} \cdot x^{n-k},\ \deg(g(x)) = n-k,$$

$$g_i \in GF(q),\ i = 0,1,...,n-k,$$

то в матричній формі циклічний код задається своєю породжувальною матрицею:

$$G = \begin{pmatrix} 0 & \dots & g_{n-k} & g_{n-k-1} & \dots & g_1 & g_0 \\ 0 & \dots & g_{n-k-1} & \dots & g_1 & g_0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ g_{n-k} & g_{n-k-1} & \dots & \dots & \dots & 0 & 0 \end{pmatrix}.$$

Лінійний підпростір, котрий прирівнює код V, має ортогональне доповнення, базис його є перевірочна матриця H коду V, або матриця рангу

$$rank(H) = r, r = n - k.$$

Розмірність перевірочної матриці $r \times n$, причому

$$G \cdot H^T = 0, \tag{2.3}$$

де під “0” розуміється $k \times r$ матриця нульових елементів з $GF(q)$.

Для поліноміальних кодів умова (2.3) еквівалентна рівнянню:

$$g(x) \cdot h(x) = x^n - 1,$$

або, що еквівалентно,

$$R_{x^n-1}[g(x) \cdot h(x)] = 0, \tag{2.4}$$

де многочлен $h(x)$ – перевірочний многочлен коду V,

$$h(x) = h_0 + h_1 \cdot x + \dots + h_k \cdot x^k,\ \deg(h(x)) = k,$$

$$h_i \in GF(q),\ i = 0, 1, \dots, k.$$

Вираз (2.4) – суть умова взаємооберненості многочленів $g(x)$ та $h(x)$ в кільці

$$GF(q)[x]/(x^n - 1).$$

Відповідна перевірочна матриця прийме вид:

$$H = \begin{pmatrix} 0 & 0 & \dots & \dots & \dots & h_{k-1} & h_k \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & h_0 & h_1 & \dots & h_k & 0 & 0 \\ h_0 & h_1 & \dots & h_k & \dots & 0 & 0 \end{pmatrix}.$$

При розгляді H як векторів лінійного підпростору, отримаємо лінійний код $V^{\perp}$, що є подвійним до V. Будь-яка n – послідовність

$$C = (C_0 \quad C_1 \quad \dots \quad C_{n-1})$$

Кодове слово V у тому випадку, коли вона ортогональна строчці матриці H, тобто

$$C \cdot H^T = 0. \tag{2.5}$$

Для поліноміальних кодів вираз (2.5) перепишеться у вигляді:

$$R_{x^n-1}[C(x) \cdot h(x)] = 0. \tag{2.6}$$

Вектор

$$S = (S_0 \quad S_1 \quad \ldots \quad S_{r-1});$$

$$S_i \in GF(q);\ i = 0, 1, \ldots, r-1$$

який в теорії кодування називається синдромом, обчислюється множенням кодового слова з помилками

$$C^* = C + E$$

на транспоновану перевірочну матрицю H коду V:

$$S = C^* \cdot H^T = C \cdot H^T + E \cdot H^T = E \cdot H^T, \qquad (2.7)$$

де E – вектор помилок:

$$E = (E_0 \quad E_1 \quad \ldots \quad E_{n-1});$$

$$E_i \in GF(q),\ i = 0, 1, \ldots, n-1;$$

C^* – кодове слово спотворене помилками:

$$C^* = (C^*_0 \quad C^*_1 \quad \ldots \quad C^*_{n-1});$$

$$C^*_i \in GF(q);\ i = 0, 1, \ldots, n-1.$$

Отже, значення синдрому S залежить від вектора помилок E і но не кодового слова C.

Для лінійних кодів, що допускають поліноміальний опис, аналогом вектора S є синдромний многочлен

$$s(x) = s_0 + s_1 \cdot x + ... + s_{r-1} \cdot x^{r-1},\ \deg(s(x)) = r - 1,$$

$$s_i \in GF(q),\ i = 0, 1, ..., r - 1.$$

Вираз (2.7) перепишеться у вигляді:

$$s(x) = R_{x^n-1}[C^*(x) \cdot h(x)] = R_{x^n-1}[C(x) \cdot h(x)] + R_{x^n-1}[E(x) \cdot h(x)] = R_{x^n-1}[E(x) \cdot h(x)],$$

де $C^*(x)$ – кодовий многочлен з помилками:

$$C^*(x) = C^*_0 + C^*_1 \cdot x + ... + C^*_{n-1} \cdot x^{n-1},\ \deg(C^*(x)) = n - 1,$$

$$C^*_i \in GF(q),\ i = 0, 1, ..., n - 1,$$

$E(x)$ – многочлен помилок:

$$E(x) = E_0 + E_1 \cdot x + ... + E_{n-1} \cdot x^{n-1},\ \deg(E(x)) = n - 1,$$

$$E_i \in GF(q),\ i = 0, 1, ..., n - 1.$$

Таким чином, значення синдромного многочлена $s(x)$ залежить від значення многочлена помилок $E(x)$ но не кодового многочлена $C(x)$.

Отримаємо:

$$s(x) = R_{x^n-1}\left[E(x)\cdot h(x)\right] = R_{g(x)}\left[E(x)\right], \tag{2.8}$$

тобто, значення синдромного многочлена дорівнює залишку від розподілу ділення многочлена помилок $E(x)$ на многочлен, що породжує $g(x)$ коду V.

Завдання декодування (n, k, d) коду V над $GF(q)$ полягає в знаходженні кодового слова C за відомими матрицями G і H і кодовим словом з помилками $C*$. З урахуванням вищевикладеного, завдання декодування можна переформулювати таким чином: знайти вектор помилок E якщо відома синдромна послідовність S.

У загальному випадку, для випадкового лінійного коду, тобто для коду з випадково незалежними і рівноймовірно вибраними k лінійно незалежними

векторами з $GF^n(q)$, що створюють базис лінійного підпростору $GF^k(q) \subseteq GF^n(q)$, сформульоване вище завдання суть теоретико-складного завдання декодування випадкового коду. Складність його рішення, наприклад, кореляційним способом, за допомогою порівняння кодового слова з помилками $C*$ з усіма кодовими словами C (n, k, d) коду V над $GF(q)$ збільшується експоненціально від параметрів коду (для повного перебирання кодових слів буде потрібно q^k спроб).

З використанням введених визначень і позначень теорії надмірного кодування нижче пропонується метод швидкого формування ППВЧ доказової стійкості. Він заснований на застосуванні обчислювально ефективних алгоритмів надмірного кодування і дозволяє за рахунок зведення відновлення секретних ключових даних до рішення теоретико-складного завдання декодування випадкового коду, забезпечити високу криптографічну стійкість синтезованих генераторів ППВЧ. Відповідно до загальної постановки відновлення секретного ключа, який параметризує роботу відповідного генератора у розроблювальному методі формування ПВЧ, повинне бути

сполучене зі знаходженням кодового слова по відомим перевірочним матрицям коду і/або матрицям, що породжують й кодовому слову з помилками.

2.2. Розробка методу формування псевдовипадкових чисел на основі надлишкових блокових кодів

В попередньому розділі в результаті проведених досліджень показано, що метод формування ПВЧ на основі надлишкових кодів має високі показники статистичної безпеки та високу швидкість формування ПВП. Але розглянутий метод формування ПВЧ не дозволяє формувати послідовності максимального періоду. Для усунення виявленого науково-технічного протиріччя й рішення проблемної ситуації необхідно розробити вдосконалений метод формування ПВЧ з зведенням завдання криптоанализу до рішення теоретико-складного завдання декодування випадкового коду.

Відомий метод формування ПВЧ (прототип) складається з наступних етапів [26, 30, 31].

Етап 1. Псевдовипадкове формування рівноважної двійкової послідовності.

Етап 2. Обчислення синдромної послідовності, що відповідає сеансовому ключу.

Етап 3. Формування фрагмента ПВЧ і сеансового ключа.

Структура відомого методу-прототипу представлена на рис. 2.1.

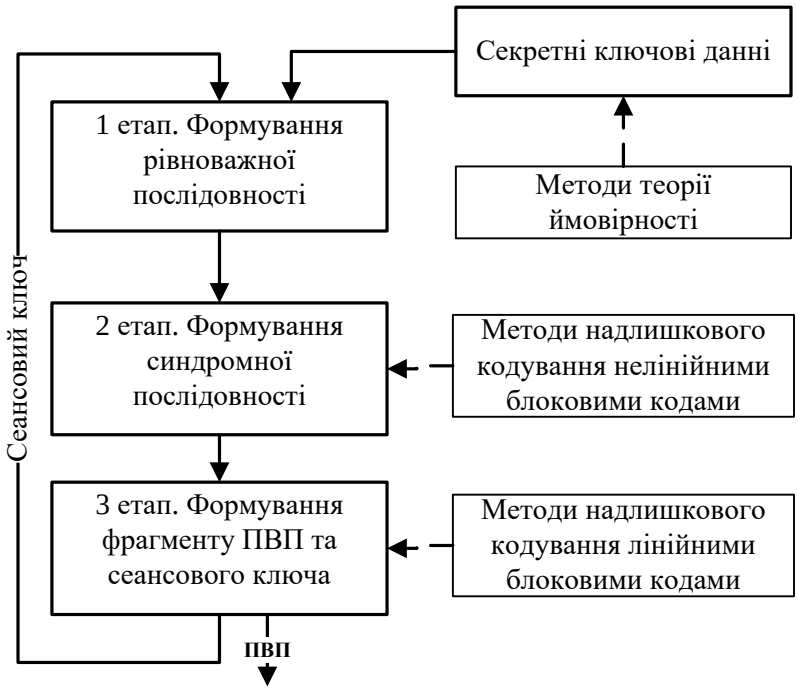

Рис. 2.1. Структурна схема методу-прототипу формування ПВЧ

На першому етапі методу-прототипу з використанням методів надлишкового кодування нелінійними блоковими кодами формуються рівноважні посилки, що відповідають уведеним секретним ключовим даним. На цьому етапі використовуються так звані. рівноважні коди, що належать до великого класу нелінійних блокових кодів [30 – 31].

На другому етапі методу-прототипу з використанням методів надлишкового кодування лінійними блоковими кодами по сформованих рівноважних посилках формується синдромна послідовність. На цьому етапі використовується довільний клас лінійних блокових кодів.

На третьому, заключному етапі по сформованій синдромній послідовності формується фрагмент ПВЧ і сеансовий ключ, що буде використовуватися на наступній ітерації роботи. На цьому етапі використовується найпростіший поділ сформованої синдромної послідовності на дві частини:

- перша - використовується у наступній ітерації як сеансовий ключ;

- друга - приймається за результат формування фрагмента ПВЧ.

Аналіз структурної схеми методу, наведеної на рис. 2.1., показує, що жоден із застосовуваних у структурі методу типів перетворень не забезпечує формування послідовностей максимального періоду. З метою усунення негативного ефекту, спостережуваного у відомому методі-прототипі формування ПВЧ, у магістерській роботі пропонується вдосконалений метод формування ПВЧ із зведенням завдання криптоанализу до рішення теоретико-складного завдання декодування випадкового коду.

Пропонований удосконалений метод формування ПВЧ складається з наступних етапів:

Етап 1. Формування сеансового ключа.

Етап 2. Псевдовипадкове формування рівноважної двійкової послідовності.

Етап 3. Обчислення синдромної послідовності, що відповідає сеансовому ключу.

Етап 4. Формування фрагмента ПВЧ і послідовності для зворотного зв'язку.

Структура вдосконаленого методу представлена на рис. 2.2. На рисунку виділені елементи відмінні від методу-прототипу.

На першому етапі пропонованого методу формуються сеансові ключові дані – послідовності максимального періоду [26, 30, 31]. Цей етап є основним відмітним елементом пропонованого методу від відомого методу-прототипу. Реалізація процедур формування послідовностей максимального періоду може бути виконана різними способами, наприклад, з використанням регістру зсуву з лінійними зворотними зв’язками РЗЛЗЗ [30, 58, 59]. Початкове заповнення регістра відповідає значенню уведених секретних ключових даних.

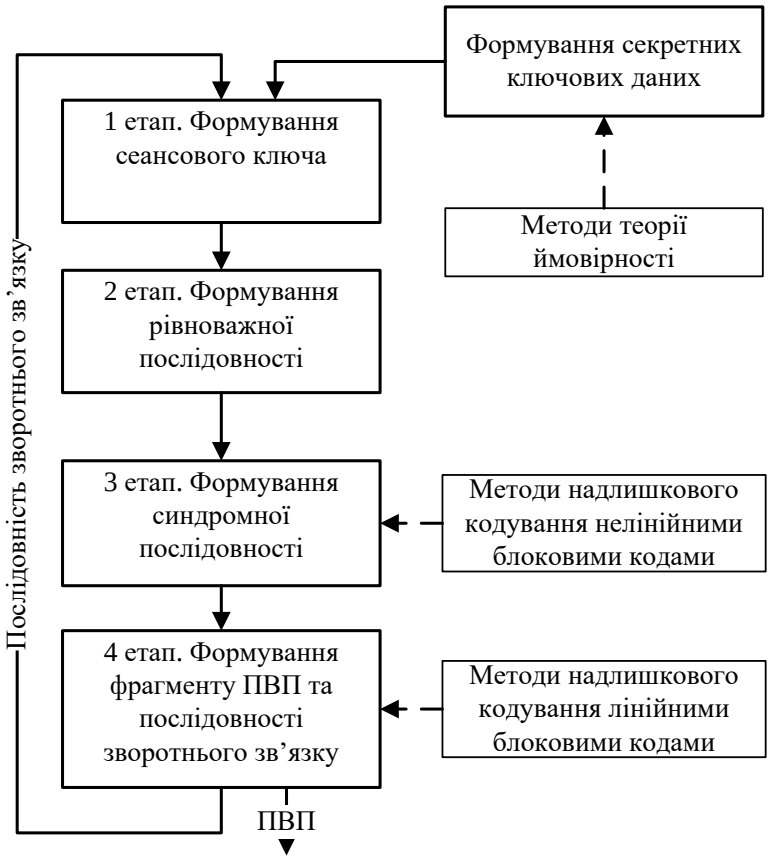

Рис. 2.2. Структурна схема вдосконаленого методу формування ПВЧ

Введемо наступні позначення:

нехай $K = \{K_1, K_2, ..., K_i\}$ – безліч секретних ключів, $|K| = M$, де:

$$M = q^m,\ m = \lfloor \log_q (C_n^w) \rfloor,\ C_n^w = \frac{n!}{w! \cdot (n-w)!},$$

де n – довжина рівноважної послідовності,

w – заздалегідь задана константа (вага послідовності - число ненульових елементів рівноважної послідовності),

$$K_i = (K_{i0} \quad K_{i1} \quad ... \quad K_{im-1}),$$

$$K_i \in K \subseteq GF^M(q),\ \ K_{ij} \in GF(q);$$

– $C_K = \{C_{K1}, C_{K2}, ..., C_{K_M}\}$ – множина послідовностей сеансових ключів:

$$C_{K_i} = (C_{Ki_0} \quad C_{Ki_1} \quad ... \quad C_{Ki_{m-1}}),$$

$$C_{Ki} \in C_K \subseteq GF^M(q),\ C_{Kij} \in GF(q);$$

– $C^*_K = \{C^*_{K1}, C^*_{K2}, ..., C^*_{Kq^m}\}$ – множина рівноважних послідовностей, тобто множина кодових слів рівноважного коду:

$$C^*_{Ki} = (C^*_{Ki0} \quad C^*_{Ki1} \quad ... \quad C^*_{Kin-1}),$$

$$C^*_{Ki} \in C^*_K \subseteq GF^n(q),\ C^*_{Kij} \in GF(q),\ w(C^*_{Ki}) = w;$$

– $S_K = \{S_{K1}, S_{K2}, ..., S_{Kq^m}\}$ – множина синдромних послідовностей лінійного блокового (n, k, d) коду, $r = n - k$:

$$S_{Ki} = (S_{Ki0} \quad S_{Ki1} \quad ... \quad S_{Kir-1}),$$

$$S_{Ki} \in S_K \subseteq GF^r(q),\ S_{Kij} \in GF(q);$$

– $S^*_K = \{S^*_{K1}, S^*_{K2}, ..., S^*_{Kq^r}\}$ – множина послідовності для зворотного зв’язку:

$$S*_{Ki} = \left(S*_{Ki0} \quad S*_{Ki1} \quad \ldots \quad S*_{Kim-1}\right),$$

$$S*_{Ki} \in S*_{K} \subseteq GF^{M}(q),\ S*_{Kij} \in GF(q).$$

На першому етапі пропонованого методу за введеною секретною ключовою послідовністю $K_i = \left(K_{i0} \quad K_{i1} \quad \ldots \quad K_{im-1}\right)$ і послідовністю для зворотнього зв'язку $S*_{Ki} = \left(S*_{Ki0} \quad S*_{Ki1} \quad \ldots \quad S*_{Kim-1}\right)$ формується послідовність сеансового ключа $C_{K_i} = \left(C_{Ki_0} \quad C_{Ki_1} \quad \ldots \quad C_{Ki_{m-1}}\right)$ як результат відображення:

$$\varphi : (K) \times (S*_{K}) \to C_K . \tag{2.9}$$

Відображення (2.9) реалізується сукупністю операцій формування послідовностей максимального періоду за введеної секретної ключовою послідовності $K_i = \left(K_{i0} \quad K_{i1} \quad \ldots \quad K_{im-1}\right)$ в комбінації з ітеративною процедурою над послідовністю зворотного зв'язку $S*_{Ki} = \left(S*_{Ki0} \quad S*_{Ki1} \quad \ldots \quad S*_{Kim-1}\right)$. В найпростішому варіанті відображення (2.9) реалізується за допомогою використання РЗЛЗЗ з початковим станом рівним введеній секретній ключовій послідовності $K_i = \left(K_{i0} \quad K_{i1} \quad \ldots \quad K_{im-1}\right)$ з подальшим елементним підсумовуванням в арифметиці кінцевого поля з послідовністю зворотного зв'язку на кожній ітерації процесу формування ППВЧ.

Структурна схема побудови пристроїв формування послідовностей сеансового ключа наведена на рис. 2.3.

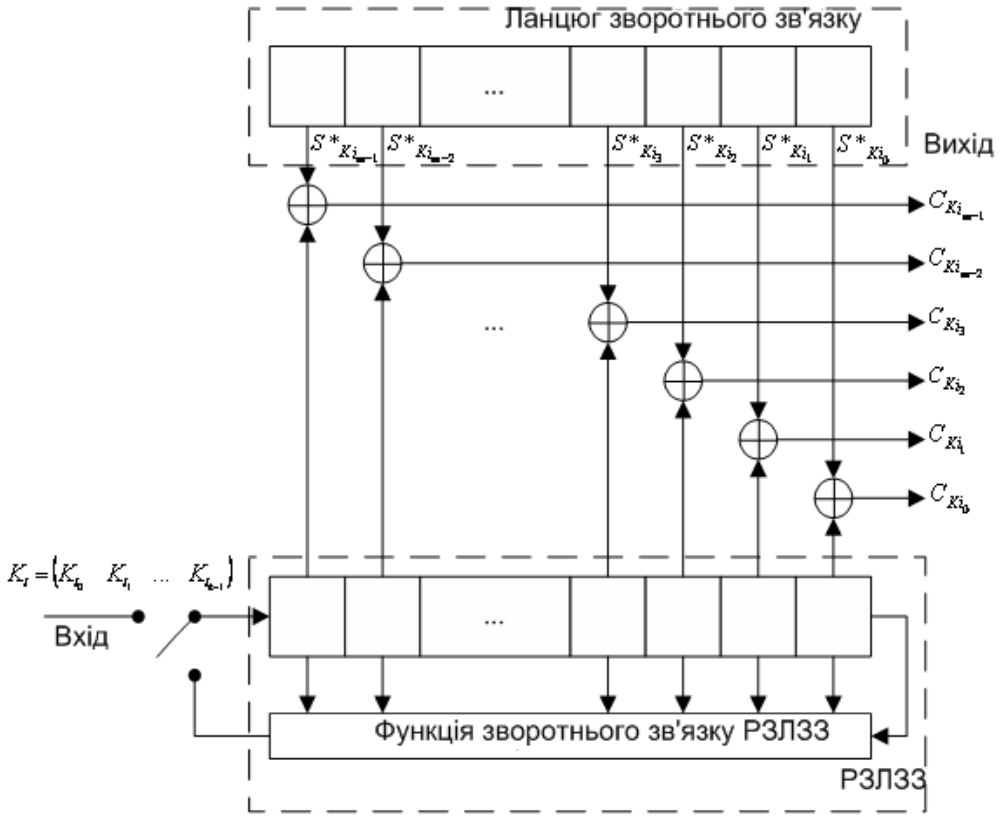

Рис. 2.3. Структурна схема пристрою формування сеансових ключів з використанням РЗЛЗЗ

Структурна схема наведена на рис. 2.3 функціонує таким чином. В плині перших k тимчасових відліків ключ (перемикач) перебуває у верхньому положенні, а регістр зсуву заповнюється ключовою послідовністю $K_i = (K_{i0} \quad K_{i1} \quad ... \quad K_{ik-1})$.

В плині наступних $q^k - 1$ часових відліків ключ (перемикач) перебуває в нижньому положенні і на вихід пристрою подаються значення, збережені в осередках регістра зсуву. На кожному часовому інтервалі на вихід пристрою подаються значення, збережені в осередках регістра зсуву. На кожному часовому інтервалі інформація, збережена в регістрі, переміщається на один осередок вправо, а зворотному зв'язку РЗЛЗЗ надходить значення, збережене в крайньому правому осередку. Функція зворотного зв'язку задає конкретний вид

комутацій ланцюга зворотного зв'язку і забезпечує формування псевдовипадкової послідовності максимального періоду.

На другому етапі пропонованого методу з використанням алгоритмів рівноважного кодування здійснюється перетворення послідовності сеансового ключа в рівноважну послідовність, тобто здійснюється відображення

$$\gamma : C_K \to C^*_K, \tag{2.10}$$

де

$$C^*_{Ki} = \left(C^*_{Ki0} \quad C^*_{Ki1} \quad \ldots \quad C^*_{Kin-1}\right),$$

$$C^*_{Ki} \in C^*_K \subseteq GF^n(q),\ C^*_{Kij} \in GF(q),\ w\left(C^*_{Ki}\right) = w,$$

n – довжина рівноважної послідовності,

w – число ненульових елементів послідовності (вага послідовності).

На третьому етапі за сформованою рівноважною послідовністю $C^*_{Ki} = \left(C^*_{Ki0} \quad C^*_{Ki1} \quad \ldots \quad C^*_{Kin-1}\right)$ з використанням методів надлишкового кодування лінійними блоковим кодами формується синдромна послідовність, тобто здійснюється відображення:

$$\phi : C^*_K \to S_K, \tag{2.11}$$

деі

$$S^*_{Ki} = \left(S^*_{Ki0} \quad S^*_{Ki1} \quad \ldots \quad S^*_{Kim-1}\right),$$

$$S*_{\kappa i} \in S*_{\kappa} \subseteq GF^{M}(q),\ S*_{\kappa ij} \in GF(q).$$

Значення синдромної послідовності лінійного блокового коду може бути отримано за допомогою матричного множення вектора-рядку $C*_{\kappa i} = \left(C*_{\kappa i0} \quad C*_{\kappa i1} \quad \ldots \quad C*_{\kappa in-1}\right)$ на транспоновану перевірочну матрицю коду. Перевірочна матриця коду (позначимо її символом H_x) використовується як секретний довготривалий ключ.

Якщо використовуваний код допускає поліноміальний опис, синдромна послідовність може бути отримана за допомогою взяття многочлена з коефіцієнтами з вектора $C*_{\kappa i} = \left(C*_{\kappa i0} \quad C*_{\kappa i1} \quad \ldots \quad C*_{\kappa in-1}\right)$ за модулем породжуючого многочлена коду. Породжуючий многочлен такого коду (позначимо його символом $g(x)_x$) використовується як секретний довготривалий ключ.

На четвертому етапі отримана синдромна послідовність піддається найпростішому перетворенню з метою формування послідовності для зворотного зв'язку й фрагмента шуканої ППВЧ. Процес формування послідовності для зворотного зв'язку формалізуємо у вигляді відображення:

$$\varphi : S_{\kappa} \to S*_{\kappa}, \tag{2.12}$$

де

$$S_{\kappa i} = \left(S_{\kappa i0} \quad S_{\kappa i1} \quad \ldots \quad S_{\kappa ir-1}\right),$$

$$S_{\kappa i} \in S_{\kappa} \subseteq GF^{r}(q),\ S_{\kappa ij} \in GF(q).$$

Сформована в результаті виконання операцій методу ППВЧ є результатом декількох функціональних відображень, в загальному вигляді:

$$\text{ПВП} = \phi\left(H_x, \gamma\left(\varphi\left(\left(K_i\right) \times \left(\varphi\left(S_{\kappa j}\right)\right)\right)\right)\right). \qquad (2.13)$$

Слід відзначити, що функціональна відповідність $\varphi\left(S_{\kappa_j}\right)$ реалізує відображення множини синдромних послідовностей в безліч послідовностей зворотного зв'язку, тобто вираз (2.13) справедливо тільки для того фрагмента ППВЧ, який формується на ітерації, що відповідає аргументу функції $\varphi\left(S_{\kappa_j}\right)$. На наступній ітерації при фіксованому значенні ключа K_i фрагмент ППВЧ залежить від іншого значення аргументу функції, тобто від $S_{\kappa l}$, $l \neq j$.

Знаходження ключових даних K_i за відомою (перехопленою) ППВЧ пов'язане з пошуком обчислювально ефективних алгоритмів виконання зворотного відображення ϕ^{-1}(ПВП). Це завдання еквівалентне теоретико-складному завданню задачі декодування випадкового коду. Крім того, наявність у структурі методу операцій формування послідовностей максимального періоду (реалізованого, наприклад, за допомогою РЗЛЗЗ), дозволяє формувати ППВЧ з гарантованим максимальним періодом:

$$L = q^m - 1.$$

Розроблений удосконалений метод формування ПВЧ, дозволяє по заданим ключовим даним K_i та з уведеного довгострокового ключа - перевірочній матриці H_x за кінцеву кількість кроків формувати ПВЧ з відомого вектору-синдрому як функції від секретного вектора-ключа. Обчислювальні алгоритми і пропозиції програмній і апаратній реалізації відображення (2.13) розглянуті в наступному підрозділі.

2.3. Розробка обчислювальних алгоритмів формування псевдовипадкових чисел на основі надлишкових блокових кодів

Запропоновані у підрозділі 2.2 методи формування ПВП складаються із процедур і операцій формування рекурентних послідовностей максимального періоду, нелінійного кодування рівноважними кодами, лінійного кодування надлишковими блоковими кодами й усікання сформованих послідовностей. Розглянемо обчислювальні аспекти реалізації запропонованих методів формування ПВЧ і обґрунтуємо основні елементи, що реалізують їх обчислювальні алгоритми.

Процес формування ПВЧ запропонованим методом можна описати сукупністю наступних співвідношень:

на першому етапі по уведеним секретним ключовим даним

$$K_i = \left(K_{i0} \quad K_{i1} \quad \ldots \quad K_{im-1}\right),$$

$$K_i \in K \subseteq GF^M(q),\ K_{ij} \in GF(q)$$

і послідовності зворотного зв'язку

$$S*_{Ki} = \left(S*_{Ki0} \quad S*_{Ki1} \quad \ldots \quad S*_{Kim-1}\right),$$

$$S*_{Ki} \in S*_K \subseteq GF^M(q),\ S*_{Kij} \in GF(q)$$

з використанням рекурентного правила

$$C_{K i_j} = -\sum_{l=1}^{m} h_l K_{i\,j-l} + S^*_{i\,j} \qquad (2.14)$$

формується послідовність сеансового ключа

$$C_{K_i} = \left(C_{K i_0} \quad C_{K i_1} \quad \ldots \quad C_{K i_{m-1}}\right),$$

$$C_{K i} \in C_K \subseteq GF^M(q),\ C_{K i j} \in GF(q);$$

на другому етапі по знайденій послідовності сеансового ключа з використанням рівняння:

$$C_{Ki} = \sum_{j=0}^{n-1} \begin{pmatrix} j \\ C^*_{K i_j} \end{pmatrix} \qquad (2.15)$$

що встановлює правило рівноважного кодування, тобто перетворення послідовності сеансового ключа в послідовність біноміального коду, формується рівноважна послідовність

$$C^*_{Ki} = \left(C^*_{K i0} \quad C^*_{K i1} \quad \ldots \quad C^*_{K i n-1}\right),$$

$$C^*_{K i} \in C^*_K \subseteq GF^n(q),\ C^*_{K i j} \in GF(q),\ w\left(C^*_{K i}\right) = w;$$

на третьому етапі з сформованої рівноважної послідовності й перевірочній матриці надлишкового лінійного блокового коду з використанням рівності

$$S_{Ki} = C^*_{Ki} \cdot H^T = \left(C^*_{Ki0} \quad C^*_{Ki1} \quad \ldots \quad C^*_{Kin-1}\right) \cdot \begin{pmatrix} h_{0,0} & h_{0,1} & \ldots & h_{0,n-1} \\ h_{1,0} & h_{1,1} & \ldots & h_{1,n-1} \\ \ldots & \ldots & \ldots & \ldots \\ h_{r-1,0} & h_{r-1,1} & \ldots & h_{r-1,n-1} \end{pmatrix}^T \qquad (2.16)$$

формується синдромна послідовність лінійного блокового (n,k,d) коду, $r = n-k$:

$$S_{Ki} = \left(S_{Ki0} \quad S_{Ki1} \quad \ldots \quad S_{Kir-1}\right),$$

$$S_{Ki} \in S_K \subseteq GF^r(q),\ S_{Kij} \in GF(q);$$

на четвертому етапі по сформованій синдромній послідовності за допомогою усікання елементів формується фрагмент ПВЧ

$$\text{ПВП} = \phi\left(H_X, \gamma\left(\varphi\left(\left(K_i\right) \times \left(\varphi\left(S_{Kj}\right)\right)\right)\right)\right) \qquad (2.17)$$

і послідовність зворотного зв'язку

$$S^*_{Ki} = \left(S^*_{Ki0} \quad S^*_{Ki1} \quad \ldots \quad S^*_{Kim-1}\right),$$

$$S^*_{Ki} \in S^*_K \subseteq GF^M(q),\ S^*_{Kij} \in GF(q),$$

що використовується на наступному циклі першого етапу запропонованого методу.

На першому етапі вміст регістра зсуву (m елементів з $GF(q)$) складається в арифметиці кінцевого поля Галуа з відповідним елементом послідовності зворотного зв'язку (рис. 2.3). Іншими словами, формування послідовності

сеансового ключа з m елементів реалізується з використанням m комірок пам'яті за один часовий інтервал (такт).

Розглянемо другий етап формування ПВЧ. З виразу (2.15) слідує, що правило формування рівноважної послідовності відповідає розкладанню C_{K_i} у біноміальний код, тобто поданню $C_{\kappa i}$ через відповідну суму біноміальних коефіцієнтів $\begin{pmatrix} C^*_{\kappa i_j} \\ j \end{pmatrix}$. Один з найбільш простих і ефективних алгоритмів двійкового біноміального кодування запропонований у роботі [50, 60], суть якого зводитися до виконання наступних елементарних кроків (рис. 2.4):

1.Ввести параметри: довжина n формованої рівноважної послідовності $C^*_{\kappa i}$, її вага $w = w(C^*_{\kappa i})$ та число $C_{\kappa i} < \begin{pmatrix} n \\ w \end{pmatrix}$, що підлягає рівноважному двійковому кодуванню.

2. Прийняти рівним: $l = 0,\ j = 0$.

3. Обчислити число $v = \begin{pmatrix} n-l \\ w-j \end{pmatrix}$.

4. Якщо $v \le C_{\kappa i}$ прийняти рівним:

$$C^*_{\kappa i_{n-1-l}} = 1,$$
$$C_{\kappa i} = C_{\kappa i} - v,$$
$$l = l + 1,$$
$$j = j + 1$$

та перейти до кроку 3.

5. Якщо $v > C_{\kappa i}$ прийняти рівним :

$$C^*_{\kappa i_{n-1-l}} = 0,$$

$$l = l + 1$$

і перейти до кроку 3.

6. Вивід вектора $C*_{Ki} = \left(C*_{Ki0} \quad C*_{Ki1} \quad \ldots \quad C*_{Kin-1}\right)$.

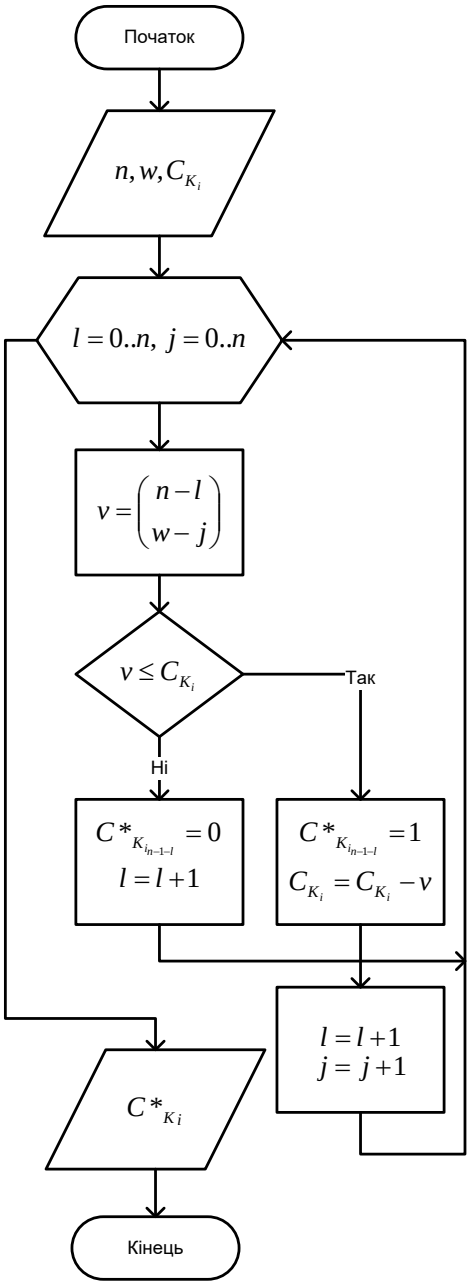

Рис. 2.4. Схема алгоритму рівноважного кодування

Таким чином, як показує аналіз розглянутого алгоритму, формування рівноважної послідовності реалізується за кінцеве число кроків з використанням елементарних арифметичних операцій. Всього, для формування рівноважної послідовності $C*_{\kappa i}$ довжини n буде потрібно виконати не більше n операцій порівнянь і n операцій вирахувань, а також буде потрібно не більше n раз обчислити число $v=\binom{n-l}{w-j}$. На цьому, найбільш витратному кроці алгоритму, для обчислення v буде потрібно не більше $n-1$ множень і n розподілів, тобто робота всього алгоритму займе $(n-1)n<n^2$ множень і розподілів і не більш ніж по n операцій порівняння й вирахування. Слід зазначити, що для великих m та n виконання розглянутих перетворень може істотно ускладнити практичну реалізацію. Дійсно, практична реалізація розглянутого алгоритму, наприклад для $m=500$ та $n=1000$, потребує $<n^2=10^6$ операцій множень і розподілів над 500 бітними числами у двійковому поданні. Для практичної реалізації алгоритмів формування ПВЧ у магістерській роботі використана спрощена схема формування рівноважних послідовностей.

Спрощена схема заснована на використанні простих і обчислювально ефективних криптографічних примітивів: нелінійних вузлів замін (S-box, таблиць підстановок) і мультиплексорів.

Основною вимогою до рівноважної послідовності, що формується, є умова:

$$w(C*_{\kappa i})=w, \tag{2.18}$$

де $C*_{\kappa i}=(C*_{\kappa i0}\quad C*_{\kappa i1}\quad \ldots\quad C*_{\kappa in-1})$ - рівноважної послідовності, що формується.

Виконання умови (2.18) обумовлено одним з основних вимог до ГПВЧ - максимальний період формованої послідовності. Дійсно, виконання умови (2.18) у термінах теорії надлишкових кодів інтерпретується як

$$w(C^*_{Ki}) = w = t,$$

де t – здатність, що виправляє (n,k,d) надлишкового лінійного блокового коду, $t = \lfloor (d-1)/2 \rfloor$, що забезпечує взаємо-однозначну функціональну відповідність між безліччю всіх можливих рівноважних векторів (векторів помилок – у термінах теорії кодів) і безліччю синдромних послідовностей (безліччю фрагментів ПВЧ). Тобто при максимальному періоді рівноважних послідовностей (забезпечуваному відповідним правилом формування) період формованої ПВЧ буде так само максимальним.

За своєю суттю алгоритм формування рівноважного вектора складається в послідовному виконанні наступних кроків:

1. Введення вхідної послідовності ($C_{K_i} = \left(C_{Ki_0} \quad C_{Ki_1} \quad \ldots \quad C_{Ki_{m-1}}\right)$
2. Розбивка вхідної послідовності на $w(C^*_{Ki}) = w = t$ підблоків по $\log_q\left(\frac{n}{t}\right)$ символів $GF(q)$ у кожному;
3. Нелінійна заміна кожного підблоку за допомогою таблиці підстановок;
4. Мультиплексування отриманих блоків даних;
5. Формування вихідної послідовності за допомогою конкатенації лічених значень.

Спрощена схема формування рівноважної послідовності представлена на рис. 2.5.

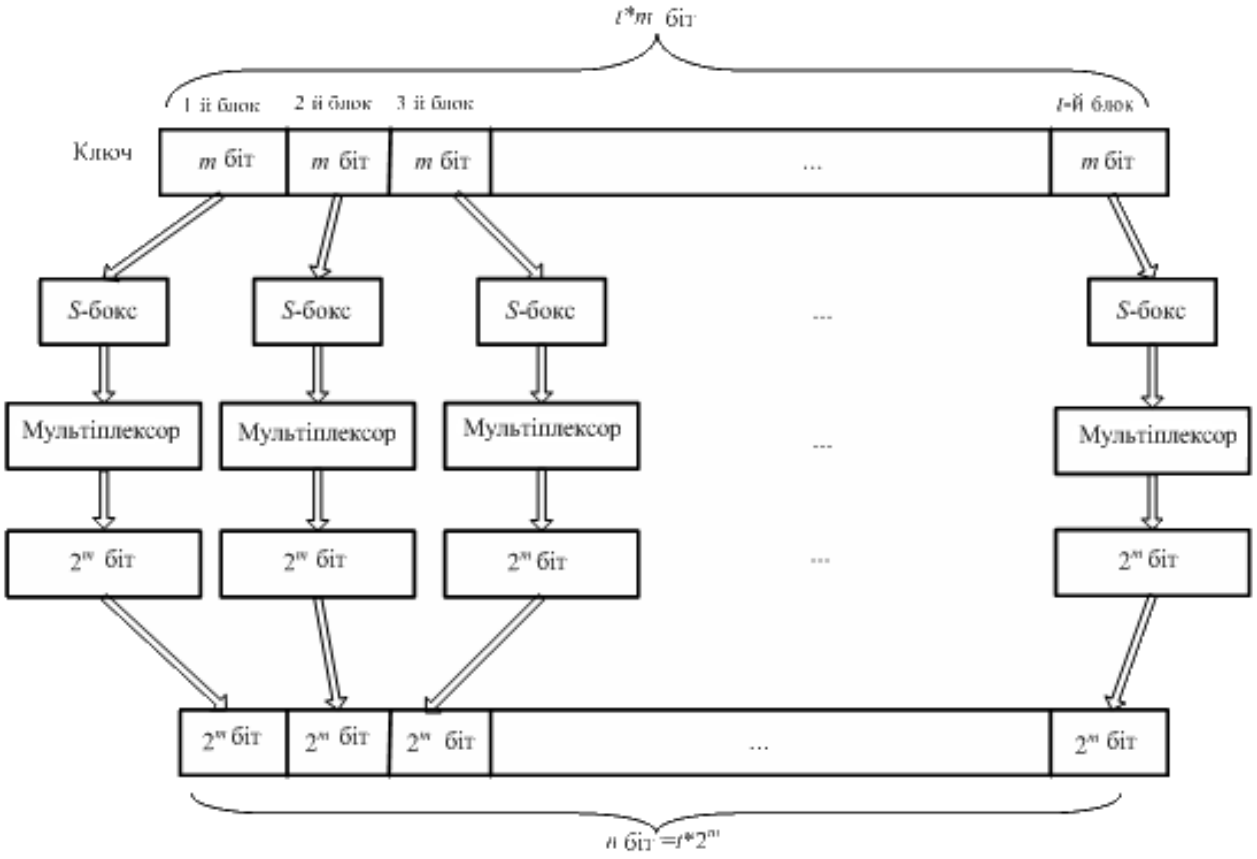

Рис. 2.5. Спрощена схема формування рівноважної послідовності

В результаті виконання всіх кроків розглянутого алгоритму формується рівноважна послідовність із числом ненульових елементів строго $w(C^*_{Ki}) = w = t$, тобто. задовольняється умова (2.18). Крім того, сформована послідовність містить у кожному з підблоків один з ненульових елементів, конкретне розміщення якого задається псевдовипадково з використанням нелінійного вузла замін.

Для приклада на рис. 2.6 представлена схема формування рівноважної послідовності в ГПВЧ на основі надлишкового блокового (64, 24, 16) коду.

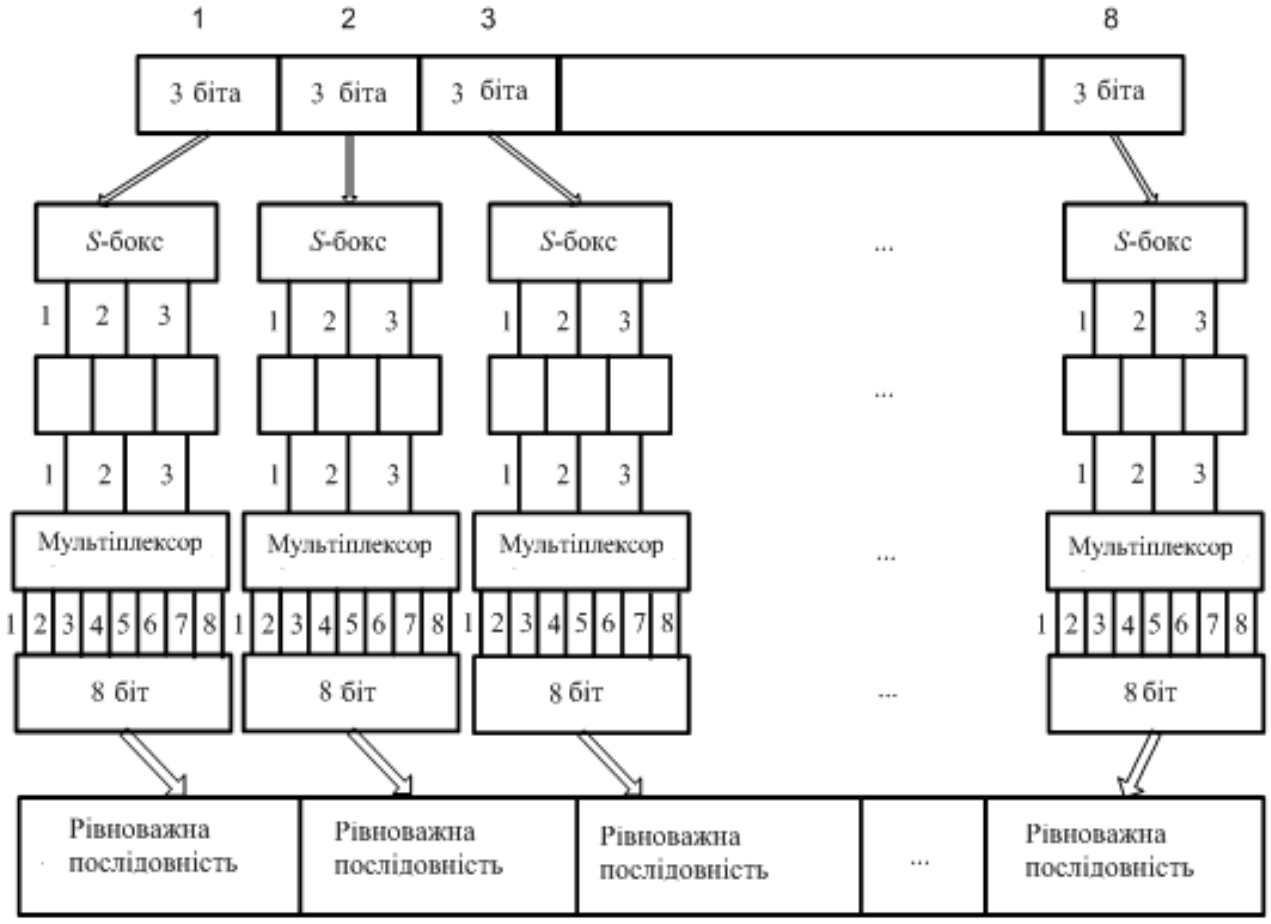

Рис. 2.6. Схема формування рівноважної послідовності на основі надлишкового блокового (64, 24, 16) коду

Таким чином для формування $((r-m)$ елементів ПВП знадобиться m комірок пам'яті та один часовий інтервал (перший етап), n^2+2n часових інтервалів та n комірок пам'яті (другий етап), w часових інтервалів та $n \cdot r$ комірок пам'яті (третій етап).

Обчислювальна складність алгоритму формування ПВЧ з використанням запропонованих процедур рівноважного кодування буде складати:

ємнісні витрати при формуванні одного символу ПВЧ

$$\text{ЄВ} = \frac{m+nr+m^2}{r-m};$$

часові витрати при формуванні одного символу ПСЧ

$$\text{ЧВ} = \frac{w+2}{r-m}.$$

Обчислювальна складність алгоритму формування ПВЧ з використанням запропонованих процедур рівноважного кодування при програмній реалізації на l - розрядній обчислювальній машині буде складати:

ємнісні витрати при формуванні одного символу ПВЧ

$$\text{ЄВ} = \frac{\left\lceil \frac{m}{l} \right\rceil + n \cdot \left\lceil \frac{r}{l} \right\rceil + \left\lceil \frac{m}{l} \right\rceil^2}{r-m};$$

часові витрати при формуванні одного символу ПСЧ

$$\text{ЧВ} = \frac{\left\lceil \frac{r}{l} \right\rceil w + \left\lceil \frac{m}{l} \right\rceil + \left\lceil \frac{\log_q\left(\frac{n}{t}\right)}{l} \right\rceil t}{r-m}.$$

Розглянемо третій етап формування ПВЧ, на якому з сформованій рівноважній послідовності

$$C^*_{Ki} = \left(C^*_{Ki0} \quad C^*_{Ki1} \quad \ldots \quad C^*_{Kin-1}\right)$$

і перевірочній матриці надлишкового лінійного блокового коду

$$H = \begin{pmatrix} h_{0,0} & h_{0,1} & \dots & h_{0,n-1} \\ h_{1,0} & h_{1,1} & \dots & h_{1,n-1} \\ \dots & \dots & \dots & \dots \\ h_{r-1,0} & h_{r-1,1} & \dots & h_{r-1,n-1} \end{pmatrix}$$

з використанням рівності (2.16) формується синдромна послідовність

$$S_{Ki} = \left(S_{Ki0} \quad S_{Ki1} \quad \dots \quad S_{Kir-1}\right).$$

З виразу (2.16) слідує, що для формування одного символу S_{Kij} синдромної послідовності S_{K_i} необхідно виконати n операцій множення й $n-1$ операцію додавання в арифметиці кінцевого поля Галуа. Отже, для формування синдромної послідовності всього буде потрібно $n \cdot r$ операцій множення й $(n-1) \cdot r$ операцій додавання над $GF(q)$. У той же час, слід зазначити, що тільки w елементів послідовності

$$C*_{Ki} = \left(C*_{Ki0} \quad C*_{Ki1} \quad \dots \quad C*_{Kin-1}\right)$$

не дорівнюють нулю. Практично це означає, що при використанні двійкових кодів для виконання операції формування синдромної послідовності буде потрібно виконати не більше w операцій додавання (XOR) над стовпцями матриці H, де номера стовпців, що складаються, задають номера ненульових елементів послідовності $C*_{Ki}$.

Четвертий етап формування ПВЧ запропонованим методом відповідає усіканню синдромної послідовності й формуванню фрагмента ПВЧ (вираз (2.17)). На практиці подібна операція може складатися в простій вибірці m елементів для ланцюга зворотного зв'язку й виводу $(r-m)$ елементів, що

залишилися як фрагмент ПСЧ, тобто не потребує обчислювальних витрат, а організується логікою роботи пристрою формування ПВЧ.

Очевидно, що складність формування ПВЧ залежить поліноміально від кодових параметрів використовуваних надлишкових блокових кодів. Схема алгоритму формування ПВЧ запропонованим методом представлена на рис. 2.7.

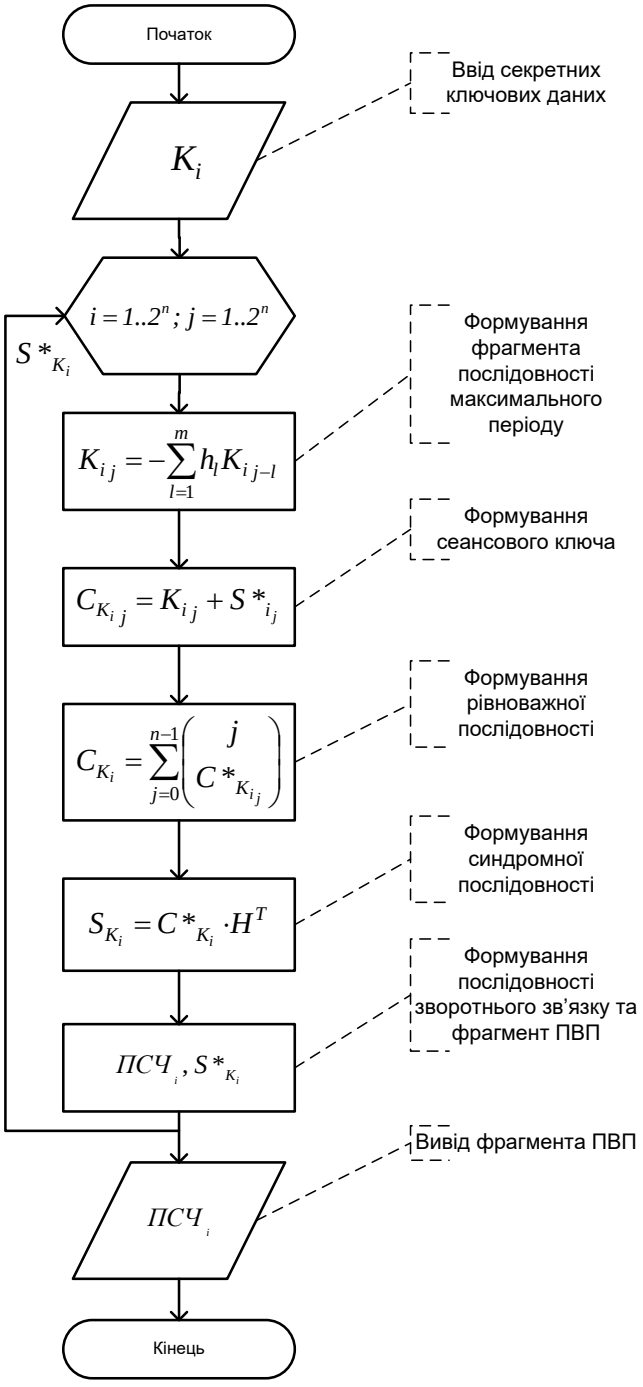

Рис. 2.7. Схема алгоритму формування ПВЧ вдосконаленим методом

Висновки до розділу 2

1. Алгебраїчна теорія надлишкового кодування й, зокрема , методи кодування і декодування лінійних блокових кодів є ефективним механізмом побудови ГПВЧ. Представивши для потенційного зловмисника застосовуваний надлишковий код як випадковий код, вдається звести завдання криптоаналізу до рішення теоретико-складного завдання декодування випадкового коду.

2. Проведені дослідження показали, що розроблений метод формування ПВЧ дозволяє будувати прості та обчислювально ефективні генератори, швидкість формування ПВЧ визначається швидкістю формування синдромної послідовності. Основним обчислювально витратним етапом розроблених методів є процедури рівноважного кодування.

3. В ході проведених досліджень установлено, що у відомих алгоритмах рівноважного кодування процедура формування рівноважної послідовності припускає обчислення арифметичних операцій (множення та розподіли) над $GF(q)$, де потужність залежить від довжини й ваги послідовності. Але на практиці необхідно реалізовувати ці операції над дуже великими числами, що веде до обчислювальної складності цих алгоритмів. Для зниження обчислювальної складності формування ПВЧ і ефективної програмної й апаратної реалізації в магістерській роботі використовується спрощена схема формування рівноважної послідовності. Спрощена схема заснована на використанні простих і обчислювально ефективних криптографічних примітивів: нелінійних вузлів замін (S-box, таблиць підстановок) і мультіплексорів.

РОЗДІЛ 3.
РОЗРОБКА ПРОГРАМНОЇ РЕАЛІЗАЦІЇ ГЕНЕРАТОРА ПСЕВДОВИПАДКОВИХ ЧИСЕЛ ТА ДОСЛІДЖЕННЯ ЙОГО ЕФЕКТИВНОСТІ

В даному розділі приводиться опис програмної реалізації генератора псевдовипадкових чисел на основі надлишкових блокових кодів, а також проводиться експериментальне дослідження статистичної безпеки та оцінка швидкодії розробленої програмної реалізації розглянутого генератора.

3.1. Розробка програмної реалізації генератора псевдовипадкових чисел на основі надлишкових блокових кодів

Розробка програмної реалізації ГПВЧ із використанням запропонованого методу виконана мовою програмування високого рівня C# з використанням середовища розробки MS Visual Studio 2008. На рис. 3.1 наведена основна форма додатка. У верхній частині форми представлені поля, необхідні для введення ключових даних і вибору розміру формованої гами. Ключові дані можливо вводити вручну в першому полі верхньої частини форми. Ключ – це 32-бітне число. При невірному введенні або при відсутності ключових даних видасться відповідне повідомлення (рис. 3.2, 3.3). Так само існує можливість згенерувати ключові дані випадковим чином: для цього необхідно нажати кнопку "Случайный ключ". Наступним кроком роботи з додатком є вибір розміру формованої гами. Даний параметр вибирається в другому полі. Для генерації псевдовипадкової послідовності необхідно нажати на кнопку "Расчитать". Послідовність генерується й записується у два файли: rezultBin.txt і rezult.txt, які знаходяться у папці з виконуваним файлом додатка. Файл rezultBin.txt містить послідовність представлену у двійковому виді (рис. 3.4), а файл rezult.txt (рис. 3.5) – у десятковому. При роботі додатка можливі критичні

ситуації (помилки при зчитуванні або записі інформації в файл) при настанні яких додаток виведе відповідні повідомлення (рис. 3.6).

В середній частині форми (рис. 3.1) додатка розміщені три текстових поля. Перше поле призначене для введення відкритого тексту вручну. При необхідності є можливість завантаження відкритого тексту з текстового файлу (кнопка "Открыть") (рис. 3.7).

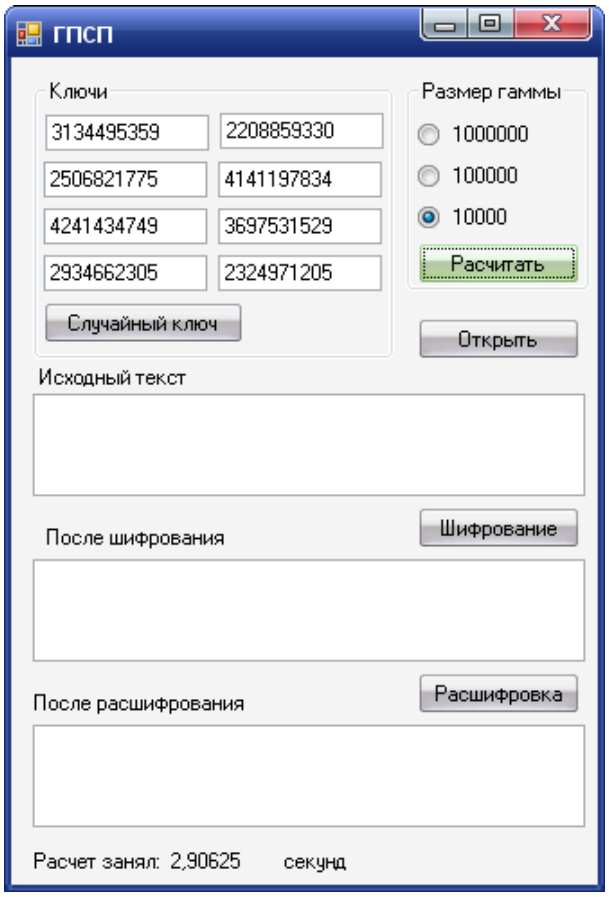

Рис. 3.1. Основна форма додатка

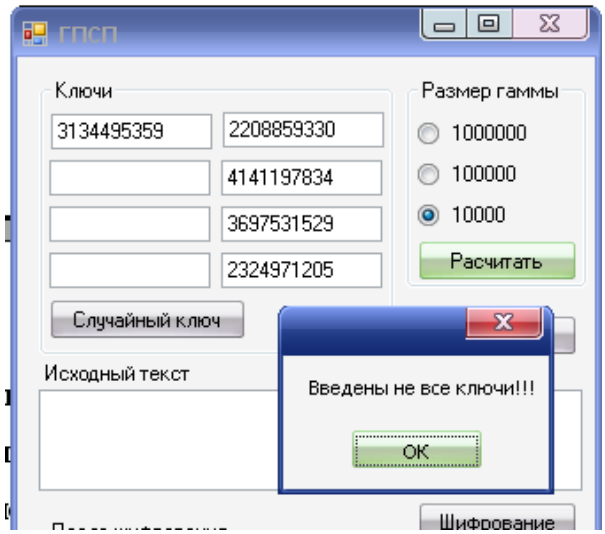

Рис. 3.2. Повідомлення про неповне введення ключових даних

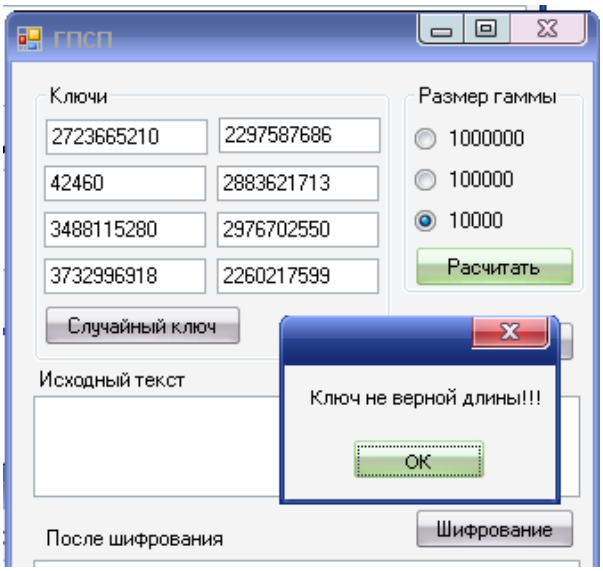

Рис. 3.3. Повідомлення про невірне введення ключової інформації

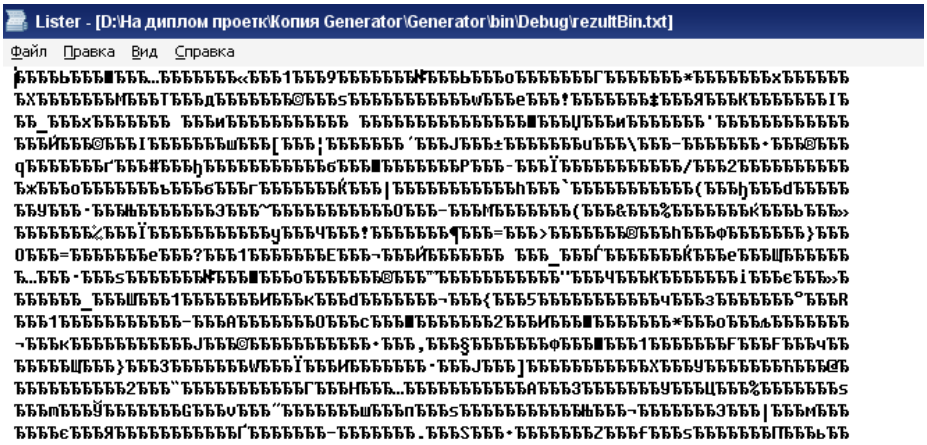

Рис. 3.4. Вміст файлу rezultBin.txt

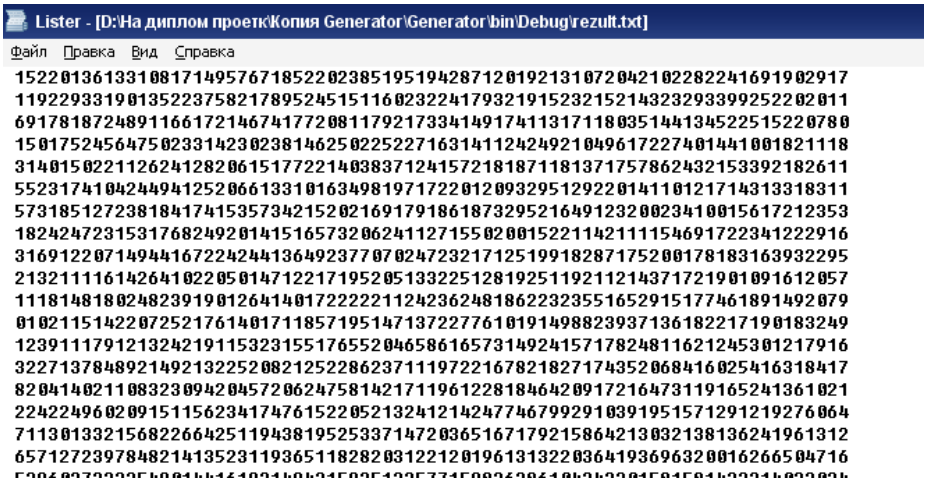

Рис. 3.5. Вміст файлу rezult.txt

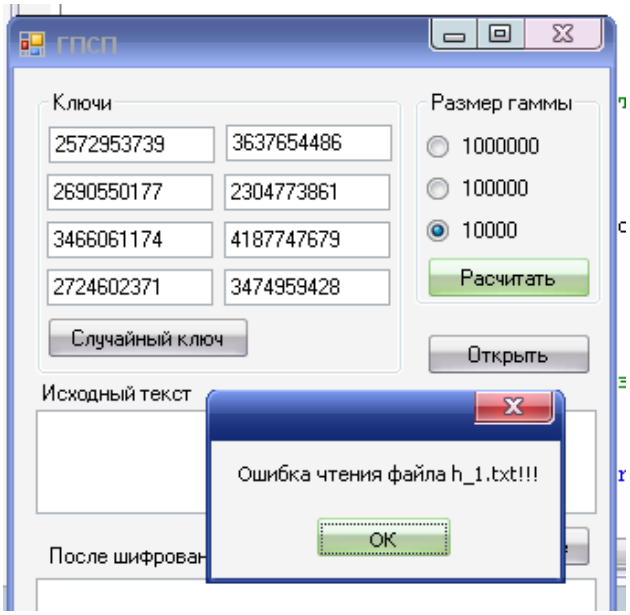

Рис .3.6. Повідомлення про помилку читання файлу

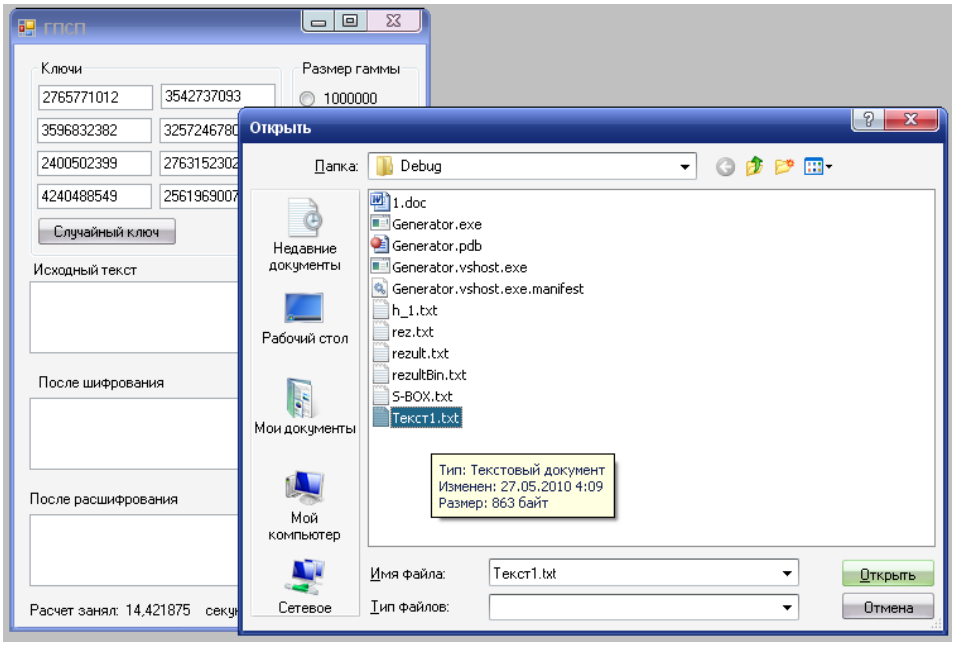

Рис. 3.7. Вікно вибору текстового файлу з відкритим текстом

Після введення відкритого тексту необхідно нажати кнопку "Шифрование" і на відкритий текст буде накладена згенерована гама. Результат операції виводиться в текстовому полі "После шифрования". Для зворотного

перетворення необхідно нажати кнопку "Расшифровка" і до шифротексту буде застосована така сама операція, результатом якої буде вхідний текст (рис. 3.8).

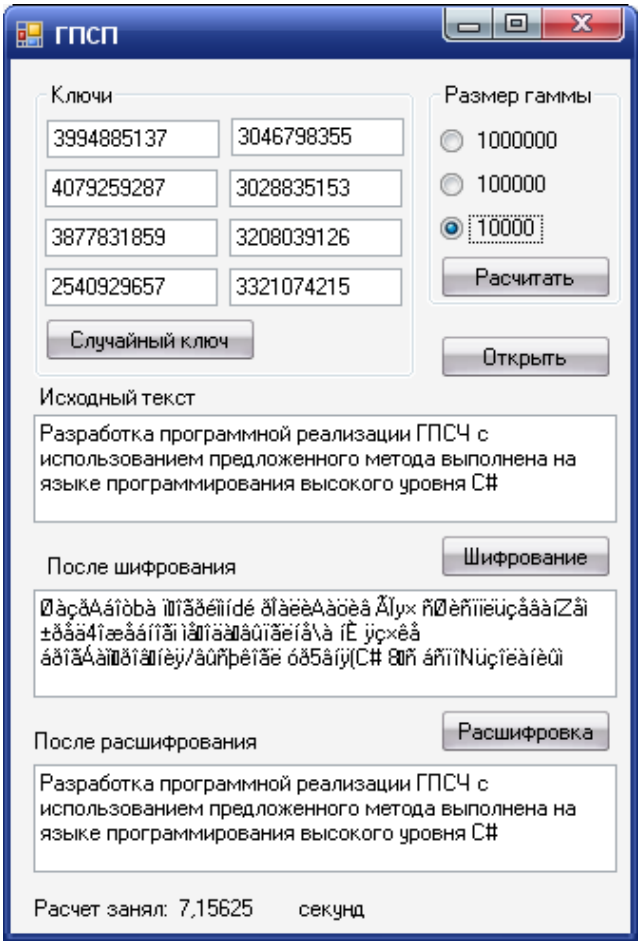

Рис. 3.8. Результати гамування і зворотної йому операції

У нижній частині головного вікна додатка виводиться інформація про тривалість кожної операції (генерації псевдовипадкової послідовності, гамування й зворотної їй).

Лістинг вихідного коду програмної реалізації наведений у додатку А.

З використанням розробленої програмної реалізації запропонованого генератора ПВЧ у магістерській роботі проведені експериментальні дослідження статистичної безпеки за методикою NIST STS, виконані порівняльні дослідження з відомими генераторами.

3.2. Експериментальне дослідження статистичної безпеки розробленого генератора за методикою NIST STS і порівняльні дослідження з відомими генераторами

Проведені експериментальні дослідження статистичної безпеки запропонованого генератора на основі надлишкових блокових кодів виконані за методикою NIST STS, результати тестування представлені у фінальних звітах (додаток В). На рис. 3.9 представлений статистичний портрет досліджуваного генератора на надлишкових кодах, на рис. 3.10 - 3.18 представлені статистичні портрети деяких відомих генераторів (на основі алгоритму SHA-1, лінійний конгруентний генератор, генератор BBS, генератор Мікалі-Шнора, квадратичний конгруентний генератор, генератори на основі алгоритмів DES і 3-DES (потрійного DES), доказово стійкий генератор GPSSD на надлишкових кодах (метод-прототип)).

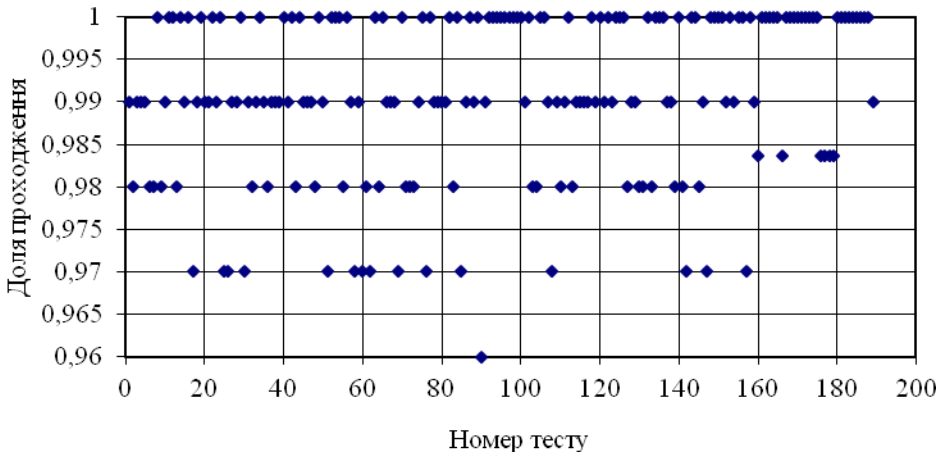

Рис. 3.9. Статистичний портрет удосконаленого генератора на надлишкових блокових кодах

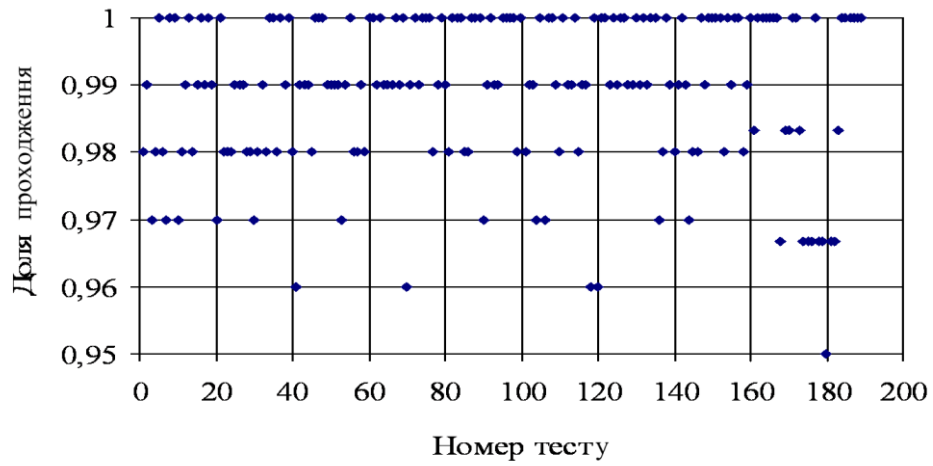

Рис. 3.10. Статистичний портрет генератора на основі алгоритму SHA-1

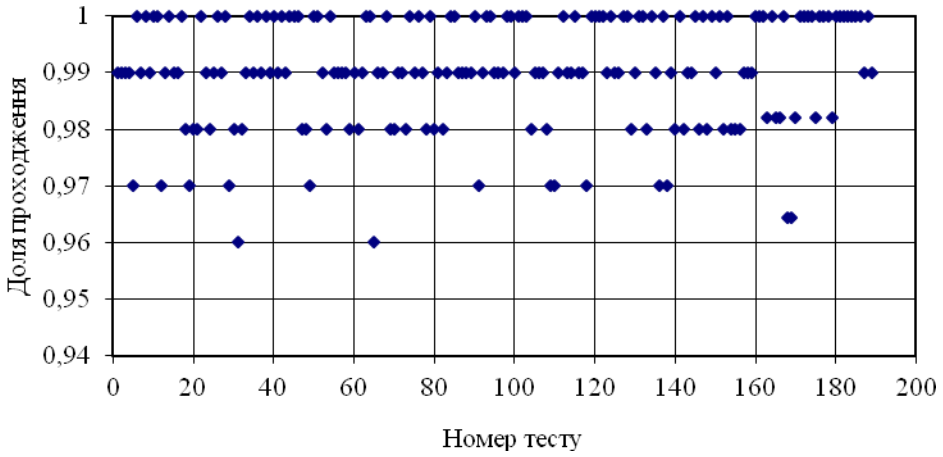

Рис. 3.11. Статистичний портрет лінійного конгруентного генератора

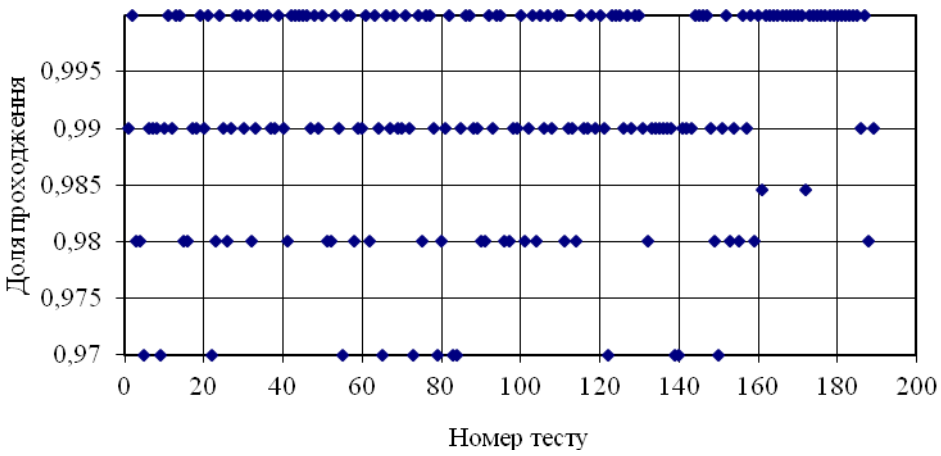

Рис. 3.12. Статистичний портрет генератора BBS

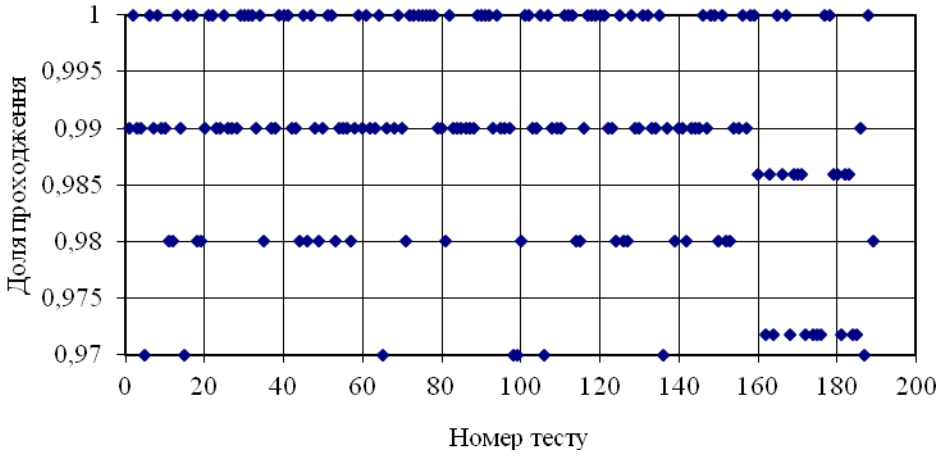

Рис. 3.14. Статистичний портрет генератора Мікалі-Шнора

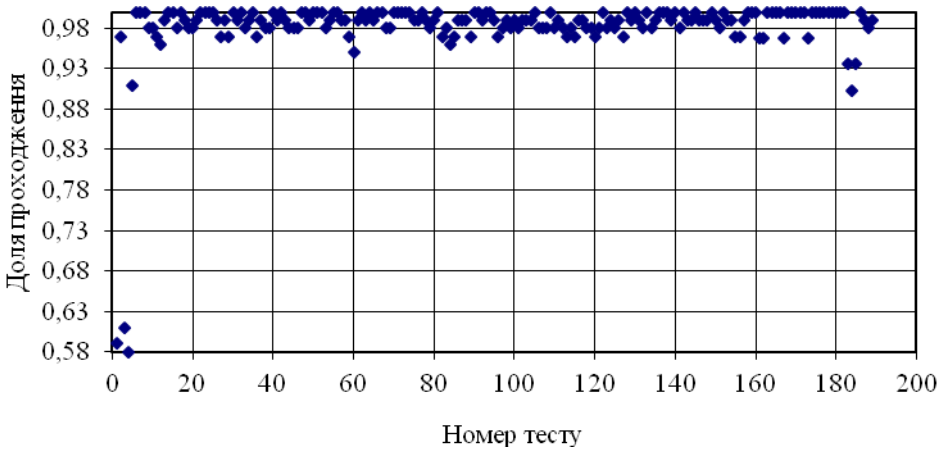

Рис. 3.15. Статистичний портрет квадратичного конгруентного генератора

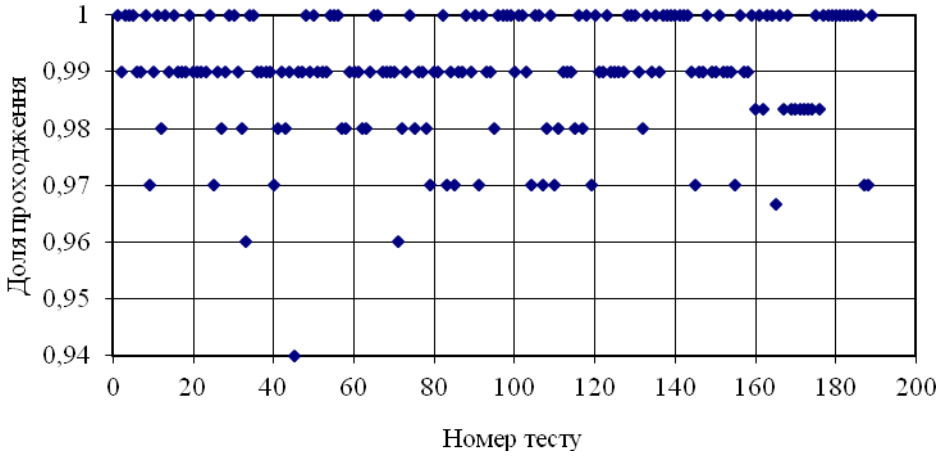

Рис. 3.16. Статистичний портрет генератора на основі алгоритму DES

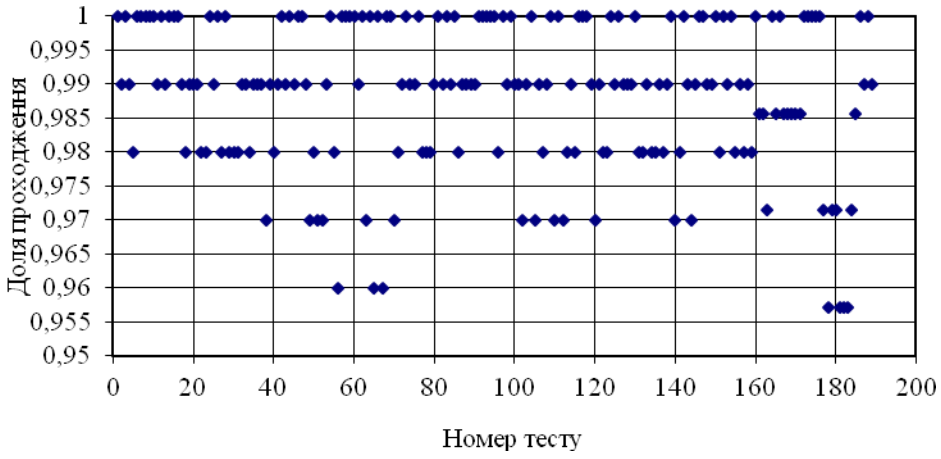

Рис. 3.17. Статистичний портрет генератора на основі алгоритму 3-DES

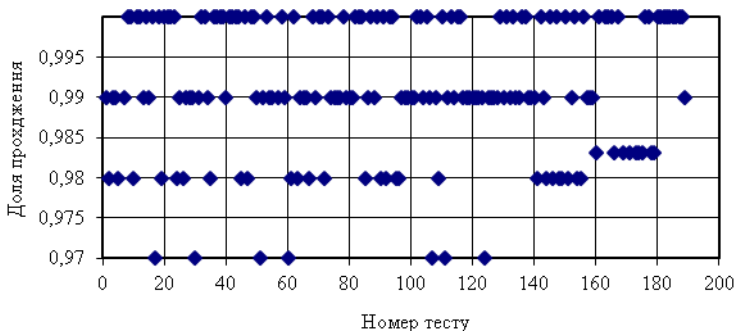

Рис. 3.18. Статистичний портрет доказово стійкого генератора на надлишкових кодах (метод-прототип)

Аналіз наведених даних показує, що статистичні портрети запропонованого генератора на надлишкових блокових кодах не уступають за своїми властивостями кращим відомим генераторам. Так, для сформованих ПВЧ немає статистичних тестів, які були б пройдені з імовірністю нижче 0,96. Основна частина тестів пройдена з дуже високою ймовірністю 0,99. Остаточні результати тестування за методикою NIST STS зведені в табл. 3.1(див. додаток В)., у якій наведені кількості (частка) тестів, в яких тестування пройшло з імовірністю 0,99; 0,96 і < 0,96.

Аналіз результатів тестування, зведених у табл. 3.1. показує, що запропонований генератор на надлишкових блокових кодах має поліпшені показники статистичної безпеки. Він має одне з найбільших число (частку) тестів, що пройшов по найбільш твердому критерії з імовірністю 0,99 і не поступає таким відомим генераторам як генератор BBS і національний алгоритм шифрування США в режимі лічильника.

3.3. Експериментальна оцінка швидкодії розробленої програмної реалізації та порівняльні дослідження з програмними реалізаціями відомих генераторів

Експериментальна оцінка проводилася з використанням операційної системи Windows XP. Фіксувався час формування ПВЧ, заміялася довжина сформованої послідовності, після чого розраховувався середній час формування ПВЧ. Результати експериментальних досліджень зведені в табл. 3.2.

Таблиця 3.2

Результати порівняльних досліджень швидкодії запропонованого і деяких відомих генераторів

Генератор псевдовипадкових чисел	Швидкодія генератора	
	Абсолютне значення	Відносне значення
BBS	$3{,}6 \cdot 10^2$ біт/с	219444
FIPS 197	$7{,}9 \cdot 10^7$ біт/с	1
Micali-Schnorr	$1{,}1 \cdot 10^5$ біт/с	718
GPSSD (метод-прототип)	$3{,}29 \cdot 10^7$ біт/с	2,4
Удосконалений генератор на надлишкових блокових кодах	$2{,}96 \cdot 10^7$ біт/с	2,7

Конкретна програмна реалізація ГПВЧ носить суб'єктивний характер і не може служити об'єктивною оцінкою їхньої швидкодії. З метою адекватного порівняння швидкодії програмної реалізації досліджуваних генераторів в останній графі таблиці наведені відносні оцінки, отримані через відношення максимальної продуктивності й шуканої продуктивності розглянутого генератора. Мінімальне значення відносної оцінки дорівнює 1 і відповідає найбільш швидкому генератору. Збільшення відносної оцінки відповідає зниженню швидкодії в порівнянні з найшвидшим генератором. Власне значення відносної оцінки відповідає коефіцієнту пропорційного зниження швидкодії розглянутого генератора (у порівнянні з найбільш швидкою реалізацією).

З наведених у табл. 3.2 значень найбільшу продуктивність показав генератор на основі алгоритму шифрування FIPS 197 (національний алгоритм шифрування США). Другим по показниках швидкодії є доказовий стійкий генератор на основі надлишкових блокових кодів (генератор GPSSD), використовуваний у даній роботі як метод-прототип. Зниження швидкодії в

порівнянні з FIPS 197 склало 2,4 рази, що пояснюється збільшенням числа виконуваних операцій над формованою послідовністю (див. розділ 2). Наступний (третій) по швидкодії виявився вдосконалений генератор. Він повільніше FIPS 197 в 2,7 рази й практично рівний по швидкодії з генератором GPSSD. Значно гірший результат показав генератор BBS, що практично на 4 порядки повільніше розглянутих вище генераторів. Подібна низька обчислювальна ефективність генератора BBS пояснюється складністю застосовуваного математичного апарата.

Вірогідність отриманих результатів і зроблених висновків підтверджується збіжністю теоретичних розрахунків з результатами експерименту. Так при довжині періоду L = 100 - 200 розроблені алгоритми формування ПВЧ вимагають виконання від 1,5 до 2,5 операцій на один формований біт ПВЧ. У той же час відомо, що для реалізації алгоритму шифрування FIPS 197 (без обліку часу розгортання ключів) необхідно затратити близько 4 операцій на одне 32-бітне слово (на одному раунді). При мінімальному числі раундів 10 алгоритм FIPS 197 потребує близько 1,25 операцій на один формований біт. Іншими словами з урахуванням роботи алгоритму розгортання ключів генератор на основі FIPS 197 і запропоновані генератори мають порівнянну обчислювальну складність.

Таким чином, як показали проведені дослідження, запропоновані методи формування ПВЧ на основі надлишкових блокових кодів мають високу ефективність: по своїх криптографічних властивостях і швидкодії вони не уступають кращим світовим аналогам, описуються моделлю доказової стійкості, легко реалізуються в програмному й апаратному виді.

Легкість реалізації дає можливість гнучкого використання розглянутого генератора в різноманітних областях. Однією з актуальних на сьогоднішній день сферою застосування ГПВЧ є банківська сфера.

Банківський механізм різноманітних платежів повинен бути захищеним, забезпечувати зниження рівня шахрайства й операційних витрат, які несуть

банки й платіжні системи. Захищеність подібної системи є винятково важливим фактором. Без застосування належних процедур і технологій боротьби із шахрайством, масштаб цього явища буде рости, у результаті чого стане знижуватися рівень довіри до електронних платіжних систем і активність користувачів пластикових карт. Світовий досвід показує, що впровадження смарт-карт здатне забезпечити значне зниження рівня шахрайства. Саме для досягнення цієї мети переважна більшість країн з розвиненим платіжним ринком переходять на технологію EMV (Europay, MasterCard і VISA).

Існує декілька широко відомих методів, застосування яких значно знижує погрозу шахрайства при аутентифікації користувача карти:

- апаратний генератор одноразових паролів (OTP, one-time password);
- табличний метод;
- підпис транзакцій на базі OTP (з використанням апаратного генератора);
- одноразові паролі, що генеруються банківською пластиковою EMV-картою (MsaterCard EMV CAP, Visa EMV DPA).

В основі методу генерації одноразових паролів лежить використання апаратного генератора одноразових паролів. Вони мають можливість уведення додаткового рівня захисту у вигляді використання статичного PIN. Генератори, що мають вбудовану клавіатуру, що використовується для уведення PIN-кода, працюють у двох режимах: запит (проста генерація) і запит-відповідь (генерація на основі отриманих від сервера значень), а також у режимі підпису транзакцій.

Суть табличного методу полягає у тому, що користувач має таблицю (або сітку, тому що в оригіналі метод називається grid - сітка). Кожний осередок містить деяку кількість символів. Значення зазначених системою осередків є інформацією, використовуваної для аутентифікації (секретом). Такі генератори також можуть використовуватися в режимі взаємної аутентифікації. Цей метод підтримує різні варіанти випуску таблиць. Підтримується конвеєрний спосіб

генерацій, що має на увазі попередню генерацію таблиць із наступною прив'язкою до користувача. Це деяка модифікація табличного методу. Користувачеві надається список паролів, згенерованний випадковим образом. Кожний пароль може бути використаний тільки один раз.

Смарт-карта - пластикова карта з вбудованою мікросхемою (ICC, integrated circuit(s) card - карта з інтегрованими електронними схемами). У більшості випадків смарт-карти містять мікропроцесор і операційну систему, що контролює пристрій і доступ до об'єктів у його пам'яті. Крім того, смарт-карты мають можливість проводити криптографічні обчислення. Призначення смарт-карт – одно- і двофакторна аутентифікація користувачів, зберігання ключової інформації й проведення криптографічних операцій у довіреному середовищі.

Чип сматр-карти (рис. 3.19) містить апаратний датчик випадкових чисел, необхідний для генерації ключової пари. Закритий ключ попадає в сховище чипа смарт карти й ніколи не виходить за межі чипа, відкритий ключ передається процесору, що генерує запит на сертифікат і посилає його за межі смарт-карти.

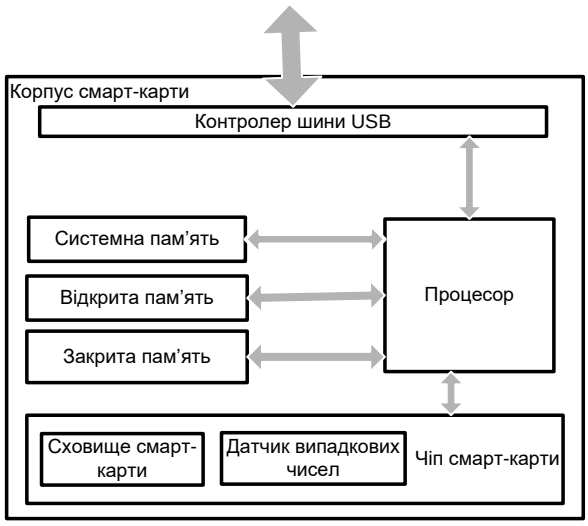

Рис. 3.19. Схема смарт-карти

Ще одним застосуванням генератора ПВЧ є використання DisplayCard, що проводить аутентифікацію на базі одноразових паролів. Являє собою звичайну банківську карту, у яку вбудований генератор одноразових паролів.

Висновки до розділу 3

1. Розроблений метод формування ПВЧ заснований на використанні надлишкових блокових кодів в режимі маскування швидкого правила декодування. Отже, задача знаходження противником секретного ключа (завдання криптоаналізу) сполучена з рішенням теоретико-складного завдання декодування випадкового коду.

2. Проведені експериментальні дослідження стійкості розглянутих генераторів показали, що запропонований в магістерській роботі генератор на надлишкових блокових кодах має покращені показниками статистичної безпеки. Найбільша частка тестів пройдена за жорстким критерієм з $P \geq 0{,}99$ і не поступається генератору BBS і національний алгоритм шифрування США в режимі лічильника.

3. Проведені експериментальні дослідження швидкодії програмної реалізації генератора на надлишкових блокових кодах і деяких відомих генераторів (генератор на основі FIPS 197, BBS) показали, що розроблений алгоритм формування ПВЧ має низьку обчислювальну складність. Швидкодія формування ПВЧ запропонованим генератором становить $10^7 - 10^8$ біт / с, що порівнянно зі швидкодією найбільш швидких блокових симетричних шифрів (FIPS 197). Отримані результати експериментальних дослідження сходяться з результатами теоретичних розрахунків, чим підтверджується достовірність отриманих результатів дослідження.

4. Однією з актуальних на сьогоднішній день сферою застосування ГПВЧ є банківська сфера. Він використовується як генератор одноразових паролів у

старт-картах для процедури аутентифікації користувачів платіжної системи, що значно підвищує рівень безпеки проведення різноманітних фінансових транзакцій.

РОЗДІЛ 4.
ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ

Опис в даному розділі базується на тому, що будь-яка виробнича діяльність пов’язана з наявністю певної кількості небезпечних або шкідливих виробничих чинників (НШВЧ). Тому визначено перелік вимог, яких повинні дотримуватись працівники та керівники, а також сформовано планування заходів цивільного захисту на об'єкті у випадку надзвичайних ситуацій.

4.1 Охорона праці

Охорона праці водночас вирішує два основних завдання. Одне з них – інженерно-технічне – передбачає запобігання небезпечним подіям під час трудового процесу шляхом:

- заміни небезпечних матеріалів менш небезпечними,
- переходу на нові технології, які зменшують ризик травмування і захворювання,
- проектування і конструювання устаткування з урахуванням вимог безпеки праці,
- розробки засобів індивідуального та колективного захисту.

Друге – соціальне – пов’язане з відшкодуванням матеріальної, моральної чи соціальної шкоди, завданої внаслідок нещасного випадку або професійного захворювання, тобто це захист працівника та його прав. Виходячи з поставлених перед нею завдань, охорона праці, ґрунтуючись на правових та організаційних основах, вирішує питання санітарії, виробничої та пожежної безпеки.

Структурно охорона праці включає у себе:

- правові та організаційні основи охорони праці;
- фізіологію, гігієну праці та виробничу санітарію;

- виробничу безпеку;
- пожежну безпеку та профілактику на виробництві.

Вимоги до приміщення

Приміщення, в яких планується установка та подальша робота з комп'ютером, повинні відповідати проектній документації будинку, погодженій з уповноваженими державними органами. Крім того, роботодавець повинен враховувати санітарні нормативи освітлення, вимоги до параметрів мікроклімату (температура, відносна вологість), ступеня і сили вібрації, звукового шуму і вогнестійкості приміщення, а також характеристики електромагнітного, ультрафіолетового та інфрачервоного полів.

Освітлення робочого місця працівника з екранними пристроями має створювати відповідний контраст між екраном і навколишнім середовищем (з урахуванням виду роботи) та відповідати вимогам ДСанПІН 3.3.2.007-98.

Природне і штучне освітлення

Згідно документу ДБН В.2.5-28:2018 "Природне і штучне освітлення" приміщення з постійним перебуванням людей повинні мати природне освітлення. Природне освітлення поділяється на бокове, верхнє і комбіноване. Що до штучного освітлення воно поділяється на робоче, аварійне, охоронне і чергове.

Мікроклімат приміщень з робочими місцями працівників з екранними пристроями має підтримуватись на постійному рівні та відповідати вимогам Санітарних норм мікроклімату виробничих приміщень ДСН 3.3.6.042-99, затверджених постановою Головного державного санітарного лікаря України від 01 грудня 1999 року № 42.

Правила поширюються на умови й організацію праці при роботі з візуальними дисплейними терміналами (ВДТ) усіх типів вітчизняного та зарубіжного виробництва на основі електронно-променевих трубок (ЕПТ), що використовуються в електронно-обчислювальних машинах (ЕОМ) колективного використання та персональних ЕОМ (ПЕОМ). Так, наприклад, роботодавцю заборонено установлювати комп'ютери в приміщеннях, розташованих у підвалах будинків. Для уникнення можливих аварій та замикань, поряд з приміщеннями, де вестиметься робота з комп'ютером (над чи під ними), також не дозволяється проведення робіт, що потребують здійснення надмірно вологих технологічних процесів. Відповідне приміщення повинно бути укомплектоване системами центрального або індивідуального опалення, кондиціонування чи вентиляції повітря. Але при установці зазначених систем, необхідно переконатись, що батареї опалення, водопровідні труби, вентиляційні кабелі тощо, надійно сховані під захисними щитками, які перешкоджатимуть можливому потраплянню робітника під напругу.

У кожній кімнаті, де обладнуватимуться робочі місця співробітників, що працюватимуть на комп'ютері, повинні бути наявні елементи природного та штучного освітлення. При цьому, на вікнах слід встановити легко регульовані жалюзі чи штори, які дозволять працівникам коригувати рівень освітлення в приміщенні. Бажано розмістити комп'ютери в кімнаті таким чином, щоб світло потрапляло на екрани моніторів з півдня чи північного сходу. З метою досягнення максимального рівня безпечності і охорони праці при роботі з комп'ютером, виробничі приміщення необхідно обладнати аптечками першої медичної допомоги, системами автоматичної пожежної сигналізації і вогнегасниками. В приміщенні, в якому разом знаходяться 5 або більше комп'ютерів, на видимому місці установлюється службовий вимикач, який у разі потреби дозволить повністю відключити електричне живлення кімнати.

Вимоги до особистого робочого місця працівника

Роботодавець, який використовує найману працю робітників, повинен забезпечити відповідність їхніх робочих місць комфортним та безпечним умовам.

Згідно з нормативними актами про охорону праці (НПАОП 0.00-7.15-18) є такі вимоги безпеки до робочих місць працівників з електронними пристроями:

- площа, відведена на одне робоче місце має становити не менше 6 кв.м., а об’єм – не менше 20 куб.м;
- конструкція робочого місця повинна забезпечувати підтримання оптимальної робочої пози, тобто такої, яка дозволяє працівникові виконувати роботу з мінімальним напруженням тіла, і яка дозволяє уникнути перевтоми в ході і після закінчення робочого процесу;
- для забезпечення безпеки та захисту здоров’я працівників усе випромінювання від екранних пристроїв має бути зведене до гранично допустимого рівня (вплив на людину факторів довкілля - шуму, вібрації, забруднювачів, температури тощо, який не спричиняє соматичних або психічних розладів, а також змін стану здоров’я, працездатності, поведінки, що виходять за межі пристосувальних реакцій) з погляду безпеки та охорони здоров’я працівників;
- організація робочого місця працівника з екранними пристроями має забезпечувати відповідність усіх елементів робочого місця та їх розташування ергономічним, антропологічним, психофізіологічним вимогам, а також характеру виконуваних робіт.

На столі працівника можливо розмістити допоміжні для роботи пристрої (принтери, колонки, сканери), а також місця для зберігання документів, за умови, що це не обмежуватиме видимість екрану і не заважатиме працівнику. У

разі надмірного шуму чи вібрації технічного обладнання, роботодавець повинен забезпечити працівників антивібраційними килимками. Робочий стілець співробітника має бути підйомно-поворотним, легко регульованим за висотою та забезпечувати належну підтримку та зручне положення спини і хребта особи. Щодня необхідно проводити вологе прибирання приміщення, та очищати робоче місце та безпосередньо монітор комп'ютера від запиленості.

На підприємстві забороняється:

- проводити ремонт та технічне обслуговування комп'ютера за робочим місцем працівника;
- самочинно ремонтувати або намагатись здійснити технічне налагодження комп'ютера без залучення компетентних спеціалістів;
- складати на робочому місці зайві документи, деталі та предмети, що не потрібні для роботи;
- використовувати монітори з нечітким зображенням та монітори, у яких наявні поламки екрану;
- працювати з матричним принтером без антивібраційного покриття та зі знятою кришкою.

Допускати до роботи осіб, які не пройшли затверджений на підприємстві курс охорони праці для роботи з комп'ютером, не дозволяється.

При прийнятті на роботу кожна особа має пройти лікарський огляд. Окрім того, при подальшій трудовій діяльності в компанії, така особа підлягає регулярному лікарському огляду не рідше ніж раз на 2 роки. Обов'язковим є проходження таких лікарів як терапевта, невропатолога та офтальмолога. В компанії мають бути чітко встановлені перерви для відпочинку працівників (окрім обідньої), як правило, тривалістю 10-15 хвилин раз на годину або дві, в залежності від складності роботи. В будь-якому випадку, роботодавець повинен передбачити такий розпорядок роботи на підприємстві, щоб час непреревної роботи з комп'ютером був не більше ніж 4 години. Додатково, для збереження належного рівня здоров'я та професійної придатності робітників,

рекомендується виділити на підприємстві окреме побутове приміщення для перепочинку працівників і зняття ними нервово-емоційного напруження, що виникає при роботі з комп’ютером.

4.2 Планування заходів цивільного захисту на об'єкті у випадку надзвичайних ситуацій

Реагування на надзвичайні ситуації та ліквідація їх наслідків – скоординовані дії суб’єктів забезпечення цивільного захисту, що здійснюються відповідно до планів реагування на надзвичайні ситуації, уточнених в умовах конкретного виду та рівня надзвичайної ситуації, і полягають в організації робіт з ліквідації наслідків надзвичайної ситуації, припинення дії або впливу небезпечних факторів, які становлять загрозу життю або здоров’ю населення, заподіяння шкоди території, навколишньому природному середовищу або майну, локалізації зони надзвичайної ситуації, а також ліквідації або мінімізації її наслідків.

План реагування на надзвичайні ситуації розробляється та затверджується в об’єднаній територіальній громаді, як відповідної ланки територіальної підсистеми єдиної державної системи цивільного захисту. Зазначений план погоджується із зацікавленими територіальними органами центральних органів виконавчої влади, органами управління місцевих спеціалізованих служб цивільного захисту. Плани реагування на надзвичайні ситуації визначають організацію управління реагуванням на надзвичайні ситуації, порядок дій і взаємодії, а також організацію основних видів забезпечення органів управління та сил цивільного захисту, що залучатимуться до реагування у разі загрози або виникнення надзвичайних ситуацій, переведення органів управління та сил цивільного захисту у режим підвищеної готовності, режим надзвичайної ситуації.

Розробку планів рекомендується здійснювати в три етапи:

- І етап: призначення розробника плану, та: аналіз законодавчої і нормативно-правової бази щодо організації та здійснення заходів у сфері захисту населення і територій від надзвичайних ситуацій; збір і узагальнення необхідних вихідних даних, зокрема відомостей про територію і чисельності населення, яке може потрапити в зону надзвичайних ситуацій (пожеж, вибухів, затоплень, забруднення радіоактивними речовинами, зараження хімічними і біологічними речовинами та ін.); уточнення переліку об'єктів і територій, які представляють небезпеку для населення; проведення аналізу за багаторічними статистичними спостереженнями видів надзвичайних ситуацій за класами, підкласами та групами, які мали місце на території (об'єкті), величин збитку, термінів виконання заходів з ліквідації наслідків надзвичайних ситуацій. За результатами І етапу розроблення Плану визначається Перелік підкласів та груп надзвичайних ситуацій, виникнення яких є імовірним на території громади, який затверджується рішенням комісії ТЕБ та НС ОТГ. Відповідно до цього Переліку розробляються додатки до Плану щодо дій органів управління та сил цивільного захисту у разі загрози або виникненні відповідної надзвичайної ситуації.
- ІІ етап: практична розробка та оформлення Плану, зокрема: Здійснюється прогнозування можливої обстановки, з урахуванням вихідних даних, моделювання можливих сценаріїв розвитку надзвичайних ситуацій, та оцінка їх наслідків, за результатами чого визначаються: сили і засоби (перелік формувань, назва та місце дислокації), що можуть залучатися до ліквідації наслідків надзвичайної ситуації, та порядок виконання ними заходів у разі загрози та виникнення надзвичайних ситуацій; порядок інформування та оповіщення про загрозу виникнення або виникнення надзвичайної ситуації; порядок переведення органів управління та сил цивільного

захисту в режим підвищеної готовності та режим надзвичайної ситуації; дії органів управління та сил цивільного захисту в режимах підвищеної готовності, надзвичайної ситуації та надзвичайного стану; порядок залучення сил цивільного захисту і проведення аварійнорятувальних та інших невідкладних робіт; порядок взаємодії органів управління і сил цивільного захисту; порядок організації забезпечення під час проведення аварійно-рятувальних та інших невідкладних робіт і ліквідації наслідків надзвичайної ситуації; проводиться узгодження плану з усіма зацікавленими структурами;

- III етап: погодження, підписання та затвердження плану.

Якщо оцінювати надійність захисту робочого персоналу, то потрібно брати до уваги те, що надзвичайні ситуації можуть призвести до втрати працездатності, важкого ураження тіла або летальних випадків.

Надійність захисту персоналу може показати, наскільки стійке підприємство до надзвичайних ситуацій у мирний час.

Для ефективного захисту людей потрібно укривати персонал у захисних спорудах та дотримуватися таких умов:

- захисні споруди повинні забезпечувати захист для всього робочого персоналу;
- захисні споруди повинні забезпечувати захист від усіх видів надзвичайних ситуацій;
- захисні споруди мають мати систему життєзабезпечення на всю тривалість перебування;
- захисні споруди мають не далеко розміщуватися від робочих місць, щоб робітники вчасно могли врятуватись;
- весь персонал має знати правила дії при сигналах про НС.

Коефіцієнт надійності захисту працівників визначається на основі кількох показників, які показують загальну підготовленість об’єкту до завдань захисту.

Після аналізу результатів, можна визначити слабкі місця об’єкту захисту та шляхи, які покращать показники надійності захисту.

У висновках потрібно вказувати:

- Надійність захисту робітників та службовців.
- Необхідність покращення захисних споруд та заходи для підвищення надійності.
- Приміщення, які можна використовувати під захисні споруди.
- Кількість та тип захисних споруд, що швидко зводяться.
- Заходи надійного захисту персоналу чергової зміни.
- Заходи з повного забезпечення персоналу.
- Заходи покращення умов зберігання, профілактики та ремонту.
- Заходи забезпечення об’єкту в різних умовах.

Висновки до розділу 4

У цьому розділі розглянуто правила охорони праці під час експлуатації електронно-обчислювальних машин та вимоги до споруд та приміщень. Також описано правила та рекомендації щодо надійності захисту умов, об’єктів та споруд на виробництвах. Наведено інформацію про освітлення виробничих приміщень для роботи ЕОМ.

ВИСНОВКИ

1. Характеристики систем безпеки в більшості своїй залежать від функцій їх криптографічних підсистем, тому якісні показники використовуваних випадкових чисел, а отже і генераторів псевдовипадкових чисел, є критично важливими. Проведений аналіз показав, що основними вимогами, що висуваються до криптостійких генераторів ПВЧ є: криптографічна стійкість, великий період формованої ПВП, статистична безпека за методикою NIST STS, ефективна апаратна й програмна реалізації. У ході дослідження встановлено, що особливим напрямком у розвитку криптостійких генераторів є методи, що допускають застосування моделі доказової безпеки. Але основним недоліком таких генераторів є їх висока складність. Проведені дослідження показали, що швидкодія доказова стійких генераторів на 3-4 порядка нижче в порівнянні з генераторами заснованих на потоковими або блоковими шифрами. Одним з перспективних напрямків, у розвитку доказово стійких генераторів, є методи, засновані на використанні надлишкових кодів.

2. Проведені дослідження ефективності відомого генератора на надлишкових кодах (GPSSD), показали, що даний генератор володіє істотним недоліком: період формованої послідовності не є максимальним, але має високі показники статистичної безпеки й високу швидкість формування ПВП. Таким чином, важливим напрямком подальших досліджень є розробка методів і обчислювальних алгоритмів для підвищення швидкодії формування ПСЧ і забезпечення максимального періоду формованої послідовності ГПВЧ заснованих на надлишкових блокових кодах. Проведені дослідження показали, що розроблений метод формування ПВЧ дозволяє будувати прості й обчислювально ефективні генератори, швидкість формування ПВЧ визначається швидкістю формування синдромної послідовності. Основним обчислювально витратним етапом розроблених методів є процедури рівноважного кодування.

3. Розроблений метод формування ПВЧ заснований на використанні надлишкових блокових кодів в режимі маскування швидкого правила декодування. Отже, задача знаходження противником секретного ключа (завдання криптоаналізу) сполучена з рішенням теоретико-складного завдання декодування випадкового коду. Проведені експериментальні дослідження стійкості розглянутих генераторів показали, що запропонований в магістерській роботі генератор на надлишкових блокових кодах має покращені показниками статистичної безпеки і не поступається таким відомим генераторам, як генератор BBS і національний алгоритм шифрування США в режимі лічильника. Проведені експериментальні дослідження швидкодії програмної реалізації генератора на надлишкових блокових кодах і деяких відомих генераторів (генератор на основі FIPS 197, BBS) показали, що розроблений алгоритм формування ПВЧ має низьку обчислювальну складність. Швидкодія формування ПВЧ запропонованим генератором становить $10^7 - 10^8$ біт/с. Отримані результати експериментальних дослідження сходяться з результатами теоретичних розрахунків, чим підтверджується достовірність отриманих результатів дослідження.

4. Однією з актуальних на сьогоднішній день сферою застосування ГПВЧ є банківська сфера. Він використовується як генератор одноразових паролів у старт-картах для процедури аутентифікації користувачів платіжної системи, що значно підвищує рівень безпеки проведення різноманітних фінансових транзакцій.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Закон України “Про Державну службу спеціального зв’язку та захисту інформації України”: від 23.02.2006 № 3475-IV/ Верховна Рада України. – Офіц. Вид. – К.:Парлам. вид-во, 2006..
2. Закон України “Про електронний цифровий підпис” від 22.05.2003 № 852-IV/ Верховна Рада України. – Офіц. Вид. – К.:Парлам. вид-во, 2003.
3. Аволио Ф. М. Защита информации на предприятии. / Ф. М. Аволио, Г. Шипли / Сети и системы связи. – 2000 - №8 – С. 91-99.
4. Бакиров Т.В. Анализ защищенности вычислительной сети и методика его проведения [електронний ресурс] // Information Security.– режим доступу: http://www.itsec.ru
5. Бакуренко А.А. Современные методы оценки информационной безопасности автоматизированных систем [електронний ресурс] // Сетевые решения.– режим доступу: http://www.nestor.minsk.by/sr/2006/10/sr61010.html
6. Блейхут Р. Теория и практика кодов, контролирующих ошибки: Пер. с англ. – М.: Мир, 1986. – 576 с.
7. Брассар Ж. Современная криптография –М.: Полимед. 1999. - 178с
8. Бондаренко В.О. Інформаційна безпека сучасної держави: концептуальні роздуми [електронний ресурс] . – режим доступу: http://www.crime-research.ru/library/strateg.htm
9. Браіловський М.М. Захист інформації у банківській діяльності / Браіловський М.М, Лазарєв Г.П., Хорошко В.О. – К.:ТОВ “ПолітграфКонсалтінг”, 2006. – 216 с.
10. Вільна енциклопедія [Електронний ресурс] Режим доступу до журн. : http://ru.wikipedia.org/wiki/Генератор_псевдослучайных_чисел
11. Винберг Э.Б. Курс алгебры. — 3-е изд.. — М.: Факториал Пресс, 2002. —

544

12. Вихорев С. В. Классификация угроз информационной безопасности [Електронний ресурс] Режим доступу до журн. : http://www2.cnews.ru/ comments/security/elvis_class.shtml
13. Диффи У. Защищенность и имитостойкость/ У. Диффи, М.Хеллман // Введение в криптографию. - 1979.- №3 - С. 79-109.
14. Долгов В.И. О некоторых подходах к построению безусловно стойких кодов аутентификации коротких сообщений / В.И. Долгов, В.Н Федорченко // Управление и связь. – 1996. – №4– С. 47-51.
15. Домарев В.В. Защита информации и безопасность компьютерных систем. – К.: Издательство "ДиаСофт". 1999. – 480 с.
16. Євсеєв С.П. Механизмы обеспечения аутентичности банковских данных во внутриплатежных системах комерческого банка. / Збірник наукових статей ХНЕУ/ Євсеєв С.П., Чевардин В.Е., Радковский С.А/. – Харків: ХНЕУ. – 2008. – Вип. 6. – С. 40-44.
17. Евсеев С.П. Построение моделей атак на внутриплатежные банковские системы./ С.П. Евсеев, О.Г. Король, А.И. Гончарова. – Запорожье: ЗНТУ. – 2010. –С. 56-66
18. Евсеев С.П. Криптографическое преобразование информации в кодовых криптосистемах на эллиптических кодах для каналов с автоматическим переспросом. // Збірник наукових праць ХУ ПС. – Харків: ХУПС. – 2007. – Вип. 8 (66). – С.29-32.
19. Задірака В.К. Методи захисту банківської інформації. / В.К. Задірака, О.С. Олесюк, Н.О. Недашковський – К.: Вища школа, 1999. – 264 с.
20. Иванов М.А. Теория, применение и оценка качества генераторов псевдослучайной последовательности – М.: «Кудиц-Образ». 2003 – 240 с.
21. Интернет Университет Информационных Технологий [електронний ресурс] – режим доступу: http://www.intuit.ru

22. Інформаційна безпека [електронний ресурс]. – режим доступу: http://www.ua7.org/mexanizmi-informacijnoї-bezpeki
23. Інформаційний портал Все об IT [Електронний ресурс] Режим доступу до журн. : http://itc.ua/node/14273/
24. Коблиц Н. Курс теории чисел и криптографии – М.: Научное изд-во ТВП, 2001 – 254 с.
25. Конеев И. Р. Информационная безопасность предприятия / И. Р.Конеев, А.В. Беляев – СПБ.: БХВ-Петербург, 2003. – 752 с.
26. Корольов Р.В. Дослідження періодичних властивостей генераторів псевдовипадкових чисел, заснованих на використанні надмірних блокових кодів / Р.В.Корольов // Системи озброєння і військова техніка № 3 (15), 2008. – 126 с..
27. Краснянська М.В. Аналіз механізмів забезпечення безпеки банківської інформації у внутриплатіжних системах комерційного банку / М.В. Краснянська. – Матеріали міжнародної науково-практичної конференції «Актуальні проблеми науки і освіти молоді: теорія, практика, сучасні рішення» 17 квітня 2010 р. Зб. наук. статей «Управління розвитком». ХНЕУ. № 4 – Х.: 2010. – 308 с.
28. Корольов Р.В. Аналіз сучасних методів формування псевдовипадкових послідовностей / Корольов Р.В., Євсеєв С.П., Краснянська М.В. – Восточно-Европейский журнал передовых технологий – Харьков. : «Технологический Центр»., 2010 – 95с.
29. Кузнецов А.А. Исследование статистической безопасности генераторов псевдослучайных чисел / А.А. Кузнецов, Р.В. Королев, Ю.Н. Рябуха // Системи обробки інформації. − Х.: ХУПС, 2008. − Вип. 3 (70). – С. 79-82.
30. Кузнецов А.А. Усовершенствованный метод быстрого формирования последовательностей псевдослучайных чисел/ "Кібернетика та системний аналіз"/ Кузнецов А.А., Королёв Р.В., Рябуха Ю.Н./ ХНВС. - К: 2008. 230

31. Кузнецов О.О. Метод швидкого формування послідовностей псевдовипадкових чисел доказової стійкості /О.О. Кузнецов, Р.В. Корольов, Ю.М. Рябуха/ ХНВС. - Харьков:2008.
32. Кузнецов А.А. Анализ механизмов обеспечения безопасности банковской информации во внутриплатежных системах коммерческого банка. / А.А. Кузнецов, О.Г. Король, А.М. Ткачов// Матеріали І міжнародної науково-практичної конференції “Безпека та захист інформації в інформаційних і телекомунікаційних системах” 28 – 29 травня 2008 р. Зб. наук. статей “Управління розвитком”. ХНЕУ. – 2008. – № 6. – с. 28 – 35.
33. Кузнецов О.О. Захист інформації та економічна безпека підприємства. Монография/ О.О. Кузнецов, С.П. Євсеєв, С.В. Кавун. – Х.: Вид. ХНЕУ, 2008. – 360 с.
34. Лидл Р. Конечные поля - М.: Мир, 1988 – 459с.
35. Мак-Вильямс Ф.Дж. Теория кодов, исправляющих ошибки./ Ф.Дж. Мак-Вильямс, Н.Дж.А. Слоэн. – М.: Связь, 1979. – 744 с.
36. Малюк А. А. Введение в защиту информации в автоматизмрованных системах / Малюк А. А., Пазизин С. В. , Погожин Н. В - М.: Горячая линия-Телеком. 2004. – 148с.
37. Методи та алгоритми адаптивного рівноважного кодування на основі біноміальних чисел для інформаційних систем: автореф. дис. / О.В. Бережная; Харк. нац. ун-т радіоелектрон. – Х., 2002. – С. 19. – [Електронний ресурс] Режим доступу до автореф.: http://disser.com.ua/contents/31551/html.
38. Мутер В.М. Основы помехоустойчивой телепередачи информации. – Л.: Энергоатомиздат. Ленингр. отд-ние, 1990. – 288 с.
39. Онанченко Е.Л. Анализ известных методов декодирования недвоичных блоковых кодов / Онанченко Е.Л., Лысенко А.В /“Вісник СумДУ. Серія технічні науки”, №3 – С: Вид, 2008 – 213с.

40. Поповский В.В. Защита информации в телекоммуникационных системах: учебник / В.В. Поповский, А.В. Персиков; Харьковский национальный университет радиоэлектроники. – Харьков: ООО "Компания Смит", 2006. – 238 с.
41. Потій О.В Методика статистичного тестування Nist STS та математичне обґрунтування тестів./ Потій О.В., Лєншин А.В., Ізбенко Ю.А. / Технічний звіт ІІТ – 001-2004
42. Потий А.В. Статистическое тестирование генераторов случайных и псевдослучайных чисел с использованием набора статистических тестов nist sts. Технический отчет ІІТ / Потий А.В., Орлова С.Ю., Гриненко Т.А. /2004
43. Приходько С.И. Метод построения генераторов ПСП с использованием эллиптических кривых/ Зб. наук. пр. "Інформаційні-керуючі системи на залізничному транспорті"/ Приходько С.И., Безверхая Г.С./ Харьков:2009.
44. Проект «Научная сеть» [Електронний ресурс] Режим доступу до журн. : http://nature.web.ru/db/msg.html?mid=1157083&uri=node9.html
45. Пятибратов А.П. Вычислительные системы, сети и телекоммуникации. – М.,2003. – 512 с.
46. Романец Ю.В Защита информации в компьютерных системах и сетях / Романец Ю.В., Тимофеев П.А., Шаньгин В.Ф/ Под ред. В.Ф. Шаньгина.-2-е изд., перераб. и доп.-М.: Радио и связь, 2001. – 376 с.
47. Рябко Б.Я. Экспериментальные исследования эффективности генераторов псевдослучайных чисел, базирующихся на криптографических алгоритмах RC5 и RC6/ Сб. науч. Раб. «Вычислительные технологии»/ Рябко Б.Я., Стогниенко В.С. Шокин Ю.И./ ИВТ СО РАН – Новосибирск 2000 – 245 с.
48. Скляр Б. Цифровая связь. Теоретические основы и практическое применение / Б. Скляр. – М.: Вильямс, 2003. – 1104 с.

49. Смарт Н.Криптография: Пер с англ..–М.:Техносфера, 2005. – 528 с.
50. Соловьев Ю.П. Эллиптические кривые и современные алгоритмы теории чисел / Ю.П. Соловьев, В.А. Садовничий, Е.Т. Шавгулидзе, В.В. Белокуров. − Москва – Ижевск: Институт компьютерных исследований, 2003. − 192 с.
51. Спосіб формування послідовностей псевдовипадкових чисел Пат. UA 38402 U, МКІ (2006) G09C 1/00 / Кузнецов О.О. Євсеєв С.П., Рябуха Ю.Н., Корольов Р.В., Пудов В.А. – № u 200810861; заявл. 03.09.2008; опубл. 12.01.2009, Бюл. №1, 2009р. – 4 с.
52. Столлингс В. Криптография и защита сетей: принципы и практика, 2-е изд. : пер. с англ. — М.: издательский дом «Вильям», 2001. — 672 с.
53. Техническая FAQ [Електронний ресурс] Режим доступу до журн. : http://ru.tech-faq.com/random-number.shtml
54. Тун Мья Аунг. Разработка и исследование стохастических методов защиты программных систем // На правах рукописи – М., 2007.
55. Уривекий А В Новые способы декодирования кодов в ранговой метрике и их криптографические приложения «Проблемі передачи информации» / Уривекий А В, Т. Йоханссон /- Л.: Энергоатомиздат. Ленингр. отд-ние, 2002. – 193 с.
56. Украинский ресурс по безопасности. [електронний ресурс] – режим доступу: http://kiev-security.org.ua
57. Чмора А. Л. Современная прикладная криптография. – Москва. 2002. – 508 с.
58. Шнайер Б. Практическая криптография / Шнайер Б., Фергюсон Н. – М.: издательский дом «Вильям». 2005. – 423с.
59. Шнайер Б. Прикладная криптография: Протоколы, Алгоритмы,Исходные тексты на С – М.: "Триумф" 2002. – 408с.
60. Эббинхауз Г.-ДМашины Тьюринга и рекурсивные функции. / Эббинхауз

Г.-Д., Якобс К., Ман Ф.-К., Хермес Г - М.: Мир, 1992.- 264с.

61. Ященко В. В. Введение в криптографию / Ященко В. В. , Черемушкин А. В. – Спб.: «Питер». 2000. – 288с.
62. Helen Gustafson, et. al. Statistical test suite Crypt-SX. – Available on http://www.isrc.qut.edu.au/cryptx.
63. A.K. Leung, S.E. Tavares. Sequence Complexity as Test for Cryptographic Systems. – Advances in Cryptology – CRYPTO’84. Proc. LNCS, Vol. 196 – Springer-Verlag.
64. Final report of European project number IST-1999-12324, named New European Schemes for Signatures, Integrity, and Encryption, April 19, 2004 – Version 0.15 (beta), Springer-Verlag, 829 p.
65. Dataforce [електронний ресурс].– режим доступу: http://www.dataforce.com.ua/Tehnologii/aes.html
66. Digital Security. [електронний ресурс] – режим доступу: http://www.dsec.ru
67. Diffi W. The first Ten Years of Public-Key Cryptography/ W. Diffi/ Computer Since – 1988 – №5 – P. 21.
68. Jakob Jonsson and Burt Kaliski. RSA-PSS. Primitive submitted to NESSIE by RSA,September 2000
69. Rabin M.O. Fingerprinting by Random Polynomials // Tech. Rep. TR-15-81, Center: in Compating Technology, Harvard Univ., Cambridge, Mass., 1981
70. Wegman M. N. New hash functions and their use in authentication and set equality / M. N.Wegman, J. L. Carter /Computer and System Scince – 1981 - № 22 - P. 265-279.

Додаток А

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ТЕРНОПІЛЬСЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ
ІМЕНІ ІВАНА ПУЛЮЯ

М А Т Е Р І А Л И

ІХ НАУКОВО-ТЕХНІЧНОЇ КОНФЕРЕНЦІЇ

«ІНФОРМАЦІЙНІ МОДЕЛІ, СИСТЕМИ ТА ТЕХНОЛОГІЇ»

8–9 грудня 2021 року

ТЕРНОПІЛЬ
2021

УДК 004.421.5
Б. М. Баняс
Тернопільський національний технічний університет імені Івана Пулюя, Україна

МЕТОДИ ФОРМУВАННЯ ПСЕВДОВИПАДКОВИХ ЧИСЕЛ В КРИПТОГРАФІЧНИХ ЗАСОБАХ ЗАХИСТУ БАНКІВСЬКИХ ІНФОРМАЦІЙНИХ СИСТЕМ

UDC 004.421.5
B. M. BANIAS
Ternopil Ivan Puluj National Technical University, Ukraine

METHODS OF FORMING PSEUDO-RANDOM NUMBERS IN CRYPTOGRAPHIC MEANS OF PROTECTION OF BANKING INFORMATION SYSTEMS

В умовах стрімкої інформатизації суспільства, широкого застосування засобів обчислювальної техніки і комп'ютерних систем особливої актуальності набувають питання інформаційної безпеки, найскладнішими з яких є необхідність захисту цінної конфіденційної і секретної інформації. Збільшення обсягів оброблюваних і переданих даних у комп'ютерних системах і мережах, перш за все в банківських системах вимагає нових підходів до протоколів та механізмів забезпечення безпеки переданих даних [1-3]. Незважаючи на широке застосування різних криптографічних алгоритмів на різних рівнях захисту інформаційні системи схильні до різних атак і погроз. Під загрозою безпеки інформаційної системи розуміються можливі впливи на інформаційну систему, що прямо чи побічно можуть завдати шкоди її безпекці.

Для забезпечення захисту від загроз безпеки використовуються різні криптографічні механізми. Для побудови механізмів безпеки інформації традиційно використовують методи криптографічного обробки інформації. Важливе місце у розвитку сучасних механізмів забезпечення безпеки інформаційних систем і технологій займає використання випадкових чисел (ПВЧ) і відповідно генераторів псевдовипадкових чисел (ГПВЧ). Вони використовуються для вирішення наступних завдань: хешування інформації; побудови синхронних і самосінхронізірующіхся потокових шифрів; формування ключової інформації і т.д. [3-5].

Характеристики систем безпеки в більшості своїй залежать від функцій їх криптографічних підсистем, які визначаються не тільки алгоритмікою, але і якісними показниками саме використовуваних псевдовипадкових послідовностей. Так як безпека криптосистеми зосереджена на ключі, то при використанні ненадійного процесу генерації ключів, вся криптосистема в цілому так само вразлива [5].

Формування ПВЧ здійснюється за допомогою відповідних ГПВЧ реалізованих на основі відомих методів, які можна розділити на два класи: криптостійкі і некріптостойкі. Класифікація методів формування ПСП наведена на рис. 1

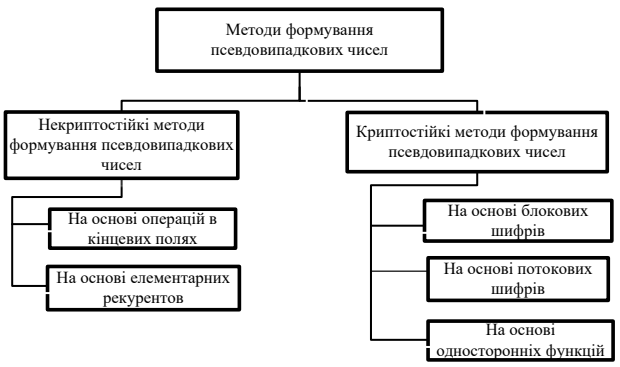

Рис. 1 Класифікація методів і генераторів формування псевдовипадкових чисел

Прикладами генераторів на основі елементарних рекурентов є лінійний і поліноміальний конгруентний генератор, адитивний і мультиплікативний генератор Фібоначчі. Найчастіше на практиці використовуються лінійні конгруентні генератори. Лінійними конгруентними генераторами є генератори наступної форми [6,9]:

$$x_i = (ax_{i-1} + b) \bmod m \tag{1.1}$$

де x_i – і–й элемент псевдовипадкової послідовності; $a \neq 0$ – множник; b – приріст; m – потужність послідовності (модуль).

Основними перевагами конгруентних генераторів є:

максимальний період формуючої послідовності;

простота програмної та апаратної реалізації;

можливість побудови на їх основі генераторів, що володіють властивостями, необхідними для вирішення прикладних питань захисту інформації.

Недоліком таких генераторів є формування псевдовипадкових чисел некриптостійких до різних видів криптоаналізу (кореляційний, інверсний та ін.) Тому конгруентні генератори використовуються для вирішення завдань захисту інформації як складові елементи кріптосхем [6, 9-11].

До криптостійких ГПВЧ відносяться генератори, побудовані на основі потокових шифрів. Прикладами можуть служити генератор SEAL, RC4, RC5, RC6, Grain та інші. Дані генератори, використовуючи більшість потокових шифрів, створюють односпрямовану послідовність бітів: єдиним способом визначити і-ий біт, знаючи ключ і позицію і, є генерування всіх бітів аж до і-ого.

Основною перевагою генераторів ПВЧ побудованих на основі потокових шифрів є висока швидкість перетворення, співмірна зі швидкістю надходження вхідної інформації. Таким чином, забезпечується формування ПСЧ в реальному масштабі часу.

Недоліками є необхідність синхронізації на приймальній і передавальній стороні [6, 9].

Наступним класом криптостійких генераторів є ГПВЧ побудовані на блокових шифрах [6, 12]. Робота таких генераторів полягає в застосуванні до блоку відкритого тексту багаторазового математичного перетворення.

Основними перевагами ГПВЧ побудованих на основі блокових шифрів є: хороші статистичні властивості формованої псевдовипадкової послідовності і стійкість до різних видів криптоаналізу (кореляційний, інверсний тощо) [6,12].

До основних недоліків блокового шифрування можна віднести:

- нечутливість кріптосхем до випадання або вставці цілого числа блоків;
- існування проблеми останнього блоку неповної довжини.

Особливим напрямком у розвитку криптостійких генераторів отримали методи, які допускають можливість застосування моделі доказовою стійкості. До них належать методи, засновані на вирішенні односторонніх функцій [6, 11, 13]. Генератори, засновані на вирішенні односторонніх функцій, називаються доказово стійкими генераторами. До доказово стійких генераторів відносяться ГПВЧ BBS і RSA.

Істотним недоліком таких генераторів є висока обчислювальна складність, яка визначається, перш за все, великою розрядністю чисел, над якими необхідно виконувати математичні операції, що істотно знижує швидкість формування ПВЧ в порівнянні з генераторами, заснованими на блокових або поточних шифрах [6,7].

Перспективним напрямком у розвитку методів формування ПВЧ є розробка та дослідження ГПВЧ заснованих на проблемі декодування випадкового коду GPSSD (Pseudo-Random Generator Provably as Secure as Syndrome Decoding), де завдання криптоаналізу фактично зводиться до вирішення теоретико-складного завдання синдромних декодувань.

Основна ідея такого генератора полягає у використанні алгебраїчного блокового коду з легко реалізованими алгоритмами кодування та декодування [7-9]. За допомогою маскування алгебраїчного коду під випадковий код, завдання декодування для зловмисника представляється як обчислювально складне.

Проведені дослідження показали, що генератор GPSSD дає кращі показники швидкодії і статистичної безпеки. Його недоліком є неможливість формування послідовності максимального періоду, а його періодичні властивості незадовільні [8].

Таким чином перспективним напрямком подальших досліджень є розробка удосконаленого методу на основі надлишкових блокових кодів, який крім високих показників статистичної безпеки та швидкодії дозволить формувати послідовності максимального періоду.

Література:

1. Задірака В.К. Методи захисту банківської інформації. / В.К. Задірака, О.С. Олесюк, Н.О. Недашковський - Київ. Вища школа, 1999 – 264 с.
2. Конеев И. Р. Информационная безопасность предприятия / И. Р.Конеев, А.В. Беляев - Спб.: БХВ-Петербург, 2003. – 752 с.
3. Дудикевич В.Б. Протоколы и механизмы безопасности информации в компьютерных системах и сетях / Дудикевич В.Б., Томашевский Б.П., Сергиенко Р.В. Технический отчет IIT – 003-2002
4. Кузнецов А.А. Анализ механизмов обеспечения безопасности банковской информации во внутриплатежных системах коммерческого банка / Матеріали І міжнародної науково-практичної конференції «Безпека та захист інформації в інформаційних і телекомунікаційних системах» 28 – 29 травня 2008 р. Зб. наук. статей «Управління розвитком»./ Кузнецов А.А., Король О.Г., Ткачов А.М. / ХНЕУ. № 6 – Х.: 2008. – С. 28 – 35.
5. Інформаційний портал Все об IT [Електронний ресурс] Режим доступу до журн. : http://itc.ua
6. Шнайер Б. Практическая криптография / Шнайер Б., Фергюсон Н. – Издательский дом «Вильямс». 2005. – 423с.
7. Кузнецов А.А. Усовершенствованный метод быстрого формирования последовательностей псевдослучайных чисел/ Зб. наук. пр. "Кібернетика та системний аналіз"/ Кузнецов А.А., Королёв Р.В., Рябуха Ю.Н./ ХНВС. - Харьков:2008.
8. Корольов Р.В. Дослідження періодичних властивостей генераторів псевдовипадкових чисел, заснованих на використанні надмірних блокових кодів / Р.В.Корольов // Системи озброєння і військова техніка. – 2008. – № 3 (15). – С. 126-128.
9. Иванов Чугунков Теория, применение и оценка качества генераторов псевдослучайной последовательности – Масква. «Кудиц-Образ». 2003 – 240с.
10. Вільна енциклопедія [Електронний ресурс] Режим доступу до журн. : http://ru.wikipedia.org/wiki/Генератор_псевдослучайных_чисел.
11. Ященко В. В. Введение в криптографию / Ященко В. В. , Черемушкин А. В. – Издательский дом «Питер». 2000. – 288с.
12. Брассар Ж. Современная криптография – Поли мед. 1999. - 178с.
13. Кузнецов А.А. Исследование статистической безопасности генераторов псевдослучайных чисел / А.А. Кузнецов, Р.В. Королев, Ю.Н. Рябуха // Системи обробки інформації. − Х.: ХУ ПС, 2008. − Вип. 3 (70). – С. 79-82.

Додаток Б

Лістинг програмної реалізації генератора псевдовипадкових чисел на основі надлишкових блокових кодів

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.IO;

namespace Generator
{
    public partial class Form1 : Form
    {
        uint konec = 0;
        uint[] nach_kluch = new uint[8];
        uint[,] s = new uint[16, 16];// S-бокс
        uint y, pp;
        int dob, qqqq, mn;
        uint[] kl = new uint[64];
        uint[] hmn = new uint[600];
        uint[,] h = new uint[600, 1200];// матрица H
        uint[,] m_h = new uint[600, 1200];
        uint[] rez = new uint[16];
        uint[] zx = new uint[16];
        uint[] dobav = new uint[10];
        string ChosenFile;
        string open_text;
        int len;
        byte[] itog;
        byte[] v;

        public Form1()
        {
            InitializeComponent();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            Random rand = new Random();

            textBox1.Text = (rand.Next(2147483647) + 2147483649).ToString();
            textBox2.Text = (rand.Next(2147483647) + 2147483649).ToString();
            textBox3.Text = (rand.Next(2147483647) + 2147483649).ToString();
            textBox4.Text = (rand.Next(2147483647) + 2147483649).ToString();
            textBox5.Text = (rand.Next(2147483647) + 2147483649).ToString();
            textBox6.Text = (rand.Next(2147483647) + 2147483649).ToString();
            textBox7.Text = (rand.Next(2147483647) + 2147483649).ToString();
            textBox8.Text = (rand.Next(2147483647) + 2147483649).ToString();

        }
```

```
        private void button2_Click(object sender, EventArgs e)
        {
            DateTime time1 = DateTime.Now;
            if ((textBox1.Text == "") || (textBox2.Text == "") || (textBox3.Text
== "") || (textBox4.Text == "") || (textBox5.Text == "") || (textBox6.Text ==
"") || (textBox7.Text == "") || (textBox8.Text == ""))
                MessageBox.Show("Введены не все ключи!!!");
            else
                if ((textBox1.Text.Length != 10) || (textBox2.Text.Length != 10)
|| (textBox3.Text.Length != 10) || (textBox4.Text.Length != 10) ||
(textBox5.Text.Length != 10) || (textBox6.Text.Length != 10) ||
(textBox7.Text.Length != 10) || (textBox8.Text.Length != 10))
                    MessageBox.Show("Ключ не верной длины!!!");
                else
                {
                    label2.Enabled = false;
                    label1.Text = "Расчет займет некоторое время...";
                    label3.Text = "";
                  ////формирование окончание рассчета
          if (radioButton1.Checked) konec = 1000000;  //размер гаммы для 1000000
           if (radioButton2.Checked) konec = 500000;   //размер гаммы для 100000
            if (radioButton3.Checked) konec = 100000;  //размер гаммы для 10000
                    //формирование ключей
                    // MessageBox.Show((uint.Parse(textBox1.Text)).ToString());
                    nach_kluch[0] = uint.Parse(textBox1.Text);
                    nach_kluch[1] = uint.Parse(textBox2.Text);
                    nach_kluch[2] = uint.Parse(textBox3.Text);
                    nach_kluch[3] = uint.Parse(textBox4.Text);
                    nach_kluch[4] = uint.Parse(textBox5.Text);
                    nach_kluch[5] = uint.Parse(textBox6.Text);
                    nach_kluch[6] = uint.Parse(textBox7.Text);
                    nach_kluch[7] = uint.Parse(textBox8.Text);
                    //считывает  S-BOX 256 чисел будет матрица 8X8}
                    try
                    {
                    StreamReader sr = new StreamReader("S-BOX.txt");
                    string str;
                    string[] sArr;
                    for (int i = 0; i < 16; i++)
                    {
                        str = sr.ReadLine();
                        sArr = str.Split(new char[] { ' ' },
StringSplitOptions.RemoveEmptyEntries);
                        for (int j = 0; j < 16; j++)
                        {
                            s[i, j] = Convert.ToUInt32(sArr[j]);
                        }
                    }
                    }
                    catch
                    {
                        MessageBox.Show("Ошибка чтения файла S-BOX.txt!!! ");
                    }
                    //разбиение ключей по 4 бита}
                    y = 0;
                    for (int i = 0; i < 8; i++)
                    {
                        kl[7 + y] = (nach_kluch[i] >> 28) & 15;
                        kl[6 + y] = (nach_kluch[i] >> 24) & 15;
                        kl[5 + y] = (nach_kluch[i] >> 20) & 15;
                        kl[4 + y] = (nach_kluch[i] >> 16) & 15;
```

```
                kl[3 + y] = (nach_kluch[i] >> 12) & 15;
                kl[2 + y] = (nach_kluch[i] >> 8) & 15;
                kl[1 + y] = (nach_kluch[i] >> 4) & 15;
                kl[0 + y] = nach_kluch[i] & 15;
                y += 8;
            }
            //использование S-бокса}
            for (int i = 0; i <= 31; i++)
            {
                y = kl[2 * i];
                kl[2 * i + 1] = (s[kl[2 * i + 1], kl[2 * i]]) & 15;
                kl[2 * i] = ((s[kl[2 * i + 1], y]) >> 4) & 15;
            }
//считывание порождающего многочлена для матрицы H из текстового файла
            try
            {
                StreamReader sr2 = new StreamReader("h_1.txt");
                string str2;
                string[] sArr2;
                if ((str2 = sr2.ReadLine()) != null)
                {
                    // str = sr.ReadLine();
                    sArr2 = str2.Split(new char[] { ' ' },
StringSplitOptions.RemoveEmptyEntries);
                    for (int j = 0; j < 454; j++)
                    {
                        hmn[j] = Convert.ToUInt32(sArr2[j]);
                    }
                }
            }
            catch
            {
                MessageBox.Show("Ошибка чтения файла h_1.txt!!! ");
            }

            //обнуление матрицы H}
            for (int i = 0; i < 570; i++)
            {
                for (int j = 0; j < 1024; j++)
                    h[i, j] = 0;
            }
            //формирование 1 строки матрицы H}
            for (int i = 0; i < 454; i++)
            {
                h[0, i] = hmn[i];
            }
            for (int i = 454; i < 1024; i++)
            {
                h[0, i] = 0;
            }
            //обнуление массива m_h[1..16,1..1024]}
            for (int i = 0; i < 16; i++)
            {
                for (int j = 0; j < 1024; j++)
                    m_h[i, j] = 0;
            }
            //формирование первой строки матрицы m_h[1..16,1..1024]}
            m_h[0, 0] = h[0, 0];
            for (int i = 1; i < 1024; i++)
            {
                if ((m_h[0, i - 1]) >= 2147483648)
```

```
                    {
                m_h[0, i] = ((((m_h[0, i - 1]) ^ (2147483648)) << 1) ^ (h[0, i]));
                    }
                    else
                    {
                        m_h[0, i] = ((m_h[0, i - 1] << 1) ^ (h[0, i]));
                    }
                }

                //формирование оставших 15 строк матрицы H}
                for (int i = 1; i < 16; i++)
                {
                    dob = 32 * i;
                    for (int j = dob; j < 1024; j++)
                        m_h[i, j] = m_h[i - 1, j - 32];
                }
                qqqq = 1;//используется для остановки рассчета гаммы}
    FileInfo FInfo = new FileInfo("rezult.txt");// создание объекта на файл File
           StreamWriter sw = FInfo.CreateText();//создане потока записи на файл
                BinaryWriter fOut = new BinaryWriter(new
FileStream("rezultBin.txt", FileMode.OpenOrCreate));
                mn = 1;
            one:
                //S-box}{использование S-бокса}
                for (int i = 0; i <= 31; i++)
                {
                    y = kl[2 * i];
                    kl[2 * i + 1] = (s[kl[2 * i + 1], kl[2 * i]]) & 15;
                    kl[2 * i] = ((s[kl[2 * i + 1], y]) >> 4) & 15;
                }
                //рассчет псевдослучайной последовательности}
                for (int i = 0; i < 16; i++)
                    rez[i] =
                            (m_h[i, 0 + kl[0]]) ^ (m_h[i, 16 + kl[1]]) ^
(m_h[i, 32 + kl[2]]) ^ (m_h[i, 48 + kl[3]])
                            ^ (m_h[i, 64 + kl[4]]) ^ (m_h[i, 80 + kl[5]]) ^
(m_h[i, 96 + kl[6]]) ^ (m_h[i, 112 + kl[7]])
                            ^ (m_h[i, 128 + kl[8]]) ^ (m_h[i, 144 + kl[9]]) ^
(m_h[i, 160 + kl[10]]) ^ (m_h[i, 176 + kl[11]])
                            ^ (m_h[i, 192 + kl[12]]) ^ (m_h[i, 208 + kl[13]]) ^
(m_h[i, 224 + kl[14]]) ^ (m_h[i, 240 + kl[15]])
                            ^ (m_h[i, 256 + kl[16]]) ^ (m_h[i, 272 + kl[17]]) ^
(m_h[i, 288 + kl[18]]) ^ (m_h[i, 304 + kl[19]])
                            ^ (m_h[i, 320 + kl[20]]) ^ (m_h[i, 336 + kl[21]]) ^
(m_h[i, 352 + kl[22]]) ^ (m_h[i, 368 + kl[23]])
                            ^ (m_h[i, 384 + kl[24]]) ^ (m_h[i, 400 + kl[25]]) ^
(m_h[i, 416 + kl[26]]) ^ (m_h[i, 432 + kl[27]])
                            ^ (m_h[i, 448 + kl[28]]) ^ (m_h[i, 464 + kl[29]]) ^
(m_h[i, 480 + kl[30]]) ^ (m_h[i, 496 + kl[31]])
                            ^ (m_h[i, 512 + kl[32]]) ^ (m_h[i, 528 + kl[33]]) ^
(m_h[i, 544 + kl[34]]) ^ (m_h[i, 560 + kl[35]])
                            ^ (m_h[i, 576 + kl[36]]) ^ (m_h[i, 592 + kl[37]]) ^
(m_h[i, 608 + kl[38]]) ^ (m_h[i, 624 + kl[39]])
                            ^ (m_h[i, 640 + kl[40]]) ^ (m_h[i, 656 + kl[41]]) ^
(m_h[i, 672 + kl[42]]) ^ (m_h[i, 688 + kl[43]])
                            ^ (m_h[i, 704 + kl[44]]) ^ (m_h[i, 720 + kl[45]]) ^
(m_h[i, 736 + kl[46]]) ^ (m_h[i, 752 + kl[47]])
                            ^ (m_h[i, 768 + kl[48]]) ^ (m_h[i, 784 + kl[49]]) ^
(m_h[i, 800 + kl[50]]) ^ (m_h[i, 816 + kl[51]])
                            ^ (m_h[i, 832 + kl[52]]) ^ (m_h[i, 848 + kl[53]]) ^
(m_h[i, 864 + kl[54]]) ^ (m_h[i, 880 + kl[55]])
```

```
                        ^ (m_h[i, 896 + kl[56]]) ^ (m_h[i, 912 + kl[57]]) ^
(m_h[i, 928 + kl[58]]) ^ (m_h[i, 944 + kl[59]])
                        ^ (m_h[i, 960 + kl[60]]) ^ (m_h[i, 976 + kl[61]]) ^
(m_h[i, 992 + kl[62]]) ^ (m_h[i, 1008 + kl[63]]);
                    //StreamWriter sw = new StreamWriter("rezult.txt", true);
                    //формирование последовательности в файл}
                    try
                    {
                        for (int i = 8; i < 16; i++)
                        {
                            zx[0] = ((rez[i]) & 4278190080) >> 24;
                            sw.Write(zx[0]);
                            fOut.Write(i);
                            zx[1] = ((rez[i]) & 16711680) >> 16;
                            sw.Write(zx[1]);
                            fOut.Write(zx[1]);
                            zx[2] = ((rez[i]) & 65280) >> 8;
                            sw.Write(zx[2]);
                            fOut.Write(zx[2]);
                            zx[3] = ((rez[i]) & 255);
                            sw.Write(zx[3]);
                            fOut.Write(zx[3]);
                        }
                    }
                    catch
                    {
                        MessageBox.Show("Ошибка записи в файл!!!");
                    }
                   //формирование ключа согласно порождающего многочлена}
                    //считывание 10,5,2,256-бит и их сложение x256+x10+x5+x2+1}
                    pp =
                          ((nach_kluch[0] >> 1) & 1)
                        ^ ((nach_kluch[0] >> 4) & 1)
                        ^ ((nach_kluch[0] >> 9) & 1)
                        ^ ((nach_kluch[7] >> 31) & 1);
                    //сумма 2 xor 5 xor 10 xor 256}
                   for (int i = 0; i < 8; i++) dobav[i] = 0;//обнуление добавок}
                    for (int i = 0; i < 8; i++)
                    {
                        if (nach_kluch[i] >= 2147483648)
                        {
                            dobav[i + 1] = 1;
                            nach_kluch[i] = ((nach_kluch[i]) ^ 2147483648) << 1;
                        }
                        else
                            nach_kluch[i] <<= 1;
                    }
                    for (int i = 0; i < 8; i++)
                    {
                        if (i == 1)
                            nach_kluch[i] ^= pp;
                        else
                            nach_kluch[i] ^= dobav[i];
                    }
                    //формирование новых ключей}
                    y = 0;
                    for (int i = 0; i < 8; i++)
                    {
                kl[7 + y] = ((nach_kluch[i] >> 28) & 15) ^ ((rez[i] >> 28) & 15);
                kl[6 + y] = ((nach_kluch[i] >> 24) & 15) ^ ((rez[i] >> 24) & 15);
                kl[5 + y] = ((nach_kluch[i] >> 20) & 15) ^ ((rez[i] >> 20) & 15);
```

```
                kl[4 + y] = ((nach_kluch[i] >> 16) & 15) ^ ((rez[i] >> 16) & 15);
                kl[3 + y] = ((nach_kluch[i] >> 12) & 15) ^ ((rez[i] >> 12) & 15);
                kl[2 + y] = ((nach_kluch[i] >> 8) & 15) ^ ((rez[i] >> 8) & 15);
                kl[1 + y] = ((nach_kluch[i] >> 4) & 15) ^ ((rez[i] >> 4) & 15);
                kl[0 + y] = (nach_kluch[i] & 15) ^ (rez[i] & 15);
                          y += 8;
                    }
                /остановка вычмсления гаммы в зависимости от выбранного расчета}
                      qqqq++;
                      if (qqqq == konec) goto two;
                      goto one;
                  two: ;
                      DateTime time2 = DateTime.Now;
                      long elapsedTicks = time2.Ticks - time1.Ticks;
                      label2.Text = (elapsedTicks * 1E-7).ToString();
                      label2.Enabled = true;
                      label1.Text = "Расчет занял: ";
                      label3.Text = "секунд";
                      sw.Close();
                      fOut.Close();
                }
        }
        private void button3_Click(object sender, EventArgs e)
        {
            textBox9.Text = "";
            try
            {
                openFileDialog1.ShowDialog();
                ChosenFile = openFileDialog1.FileName;
                StreamReader sr3 = new StreamReader(ChosenFile,
Encoding.Default);
                string str;
                while ((str = sr3.ReadLine()) != null)
                {
                    textBox9.Text += str;
                    textBox9.Text += "\r\n";
                }
            }
            catch
            {
                MessageBox.Show("Ошибка чтения файла!!!");
            }
        }

        private void button4_Click(object sender, EventArgs e)
        {
            if (textBox9.Text == "")
                MessageBox.Show("Нет данных для шифрования!!!");
            else
            {
                DateTime time1 = DateTime.Now;
                open_text = textBox9.Text;
                int text_len = open_text.Length;
                byte[] open_text_bin = Encoding.Default.GetBytes(open_text);
                len = open_text_bin.Length;
                BinaryReader fin = new BinaryReader(File.Open("rezultBin.txt",
FileMode.Open));
                v = new byte[len];
                fin.Read(v, 0, len);
                itog = new byte[len];
```

```
                for (int i = 0; i < len; i++)
                {
                    itog[i] = (byte)(open_text_bin[i] ^ v[i]);
                    textBox10.Text += ((char)itog[i]);
                }
                DateTime time2 = DateTime.Now;
                long elapsedTicks = time2.Ticks - time1.Ticks;
                label2.Text = (elapsedTicks * 1E-7).ToString();
                label2.Enabled = true;
                label1.Text = "Расчет занял: ";
                label3.Text = "секунд";
            }

        }

        private void button5_Click(object sender, EventArgs e)
        {
            if (textBox10.Text == "")
                MessageBox.Show("Нет данных для расшифрования!!!");
            else
            {
                DateTime time1 = DateTime.Now;
                byte[] itog2 = new byte[len];

                for (int i = 0; i < len; i++)
                {
                    itog2[i] = (byte)(itog[i] ^ v[i]);
                    textBox12.Text +=
((char)Encoding.Default.GetChars(itog2)[i]);
                }
                DateTime time2 = DateTime.Now;
                long elapsedTicks = time2.Ticks - time1.Ticks;
                label2.Text = (elapsedTicks * 1E-7).ToString();
                label2.Enabled = true;
                label1.Text = "Расчет занял: ";
                label3.Text = "секунд";
            }
        }


    }
}
```

Додаток В

Результати експериментальних досліджень статистичної безпеки запропонованих генераторів псевдовипадкових чисел на основі надлишкових блокових кодів

Таблиця В.1

Результати статистичного тестування запропонованого генератора

	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	P-value	Proportion	Statistical test
1.	10	10	10	6	6	19	12	11	10	6	0,145326	0,9900	frequency
2.	5	13	9	10	10	11	14	13	10	5	0,474986	0,9800	block-frequency
3.	13	10	10	7	11	11	11	11	14	2	0,334538	0,9900	cumulative-sums
4.	13	5	8	11	8	8	16	11	12	8	0,419021	0,9900	cumulative-sums
5.	8	12	12	11	14	9	9	11	7	7	0,834308	0,9900	runs
6.	7	10	7	16	7	10	8	13	10	12	0,534146	0,9800	longest-run
7.	9	14	11	11	8	9	8	8	13	9	0,897763	0,9800	rank
8.	10	10	14	11	7	10	12	5	6	15	0,383827	1,0000	fft
9.	7	13	8	7	11	11	12	6	13	12	0,678686	0,9800	nonperiodic-templates
10.	11	9	16	11	8	9	10	13	9	4	0,437274	0,9900	nonperiodic-templates
11.	13	11	9	10	7	9	14	10	7	10	0,867692	1,0000	nonperiodic-templates
12.	9	7	19	9	8	7	10	12	7	12	0,202268	1,0000	nonperiodic-templates
13.	11	10	14	10	13	8	7	9	10	8	0,883171	0,9800	nonperiodic-templates
14.	12	7	7	8	15	9	9	6	18	9	0,145326	1,0000	nonperiodic-templates
15.	8	10	13	14	11	7	12	10	8	7	0,779188	0,9900	nonperiodic-templates
16.	10	12	7	13	12	10	8	13	11	4	0,574903	1,0000	nonperiodic-templates
17.	16	11	12	7	8	5	16	10	9	6	0,153763	0,9700	nonperiodic-templates
18.	8	11	8	11	12	11	10	9	8	12	0,983453	0,9900	nonperiodic-templates
19.	7	12	9	10	9	8	10	10	12	13	0,955835	1,0000	nonperiodic-templates
20.	8	10	9	6	9	13	10	12	8	15	0,699313	0,9900	nonperiodic-templates
21.	11	9	12	5	8	9	15	8	15	8	0,401199	0,9900	nonperiodic-templates
22.	9	12	10	13	10	6	9	6	14	11	0,699313	1,0000	nonperiodic-templates
23.	16	9	11	5	5	8	13	10	10	13	0,275709	0,9900	nonperiodic-templates
24.	10	14	15	6	9	11	6	7	10	12	0,455937	1,0000	nonperiodic-templates
25.	10	8	12	10	12	10	13	5	8	12	0,798139	0,9700	nonperiodic-templates
26.	14	9	9	13	14	11	10	7	9	4	0,437274	0,9700	nonperiodic-templates
27.	10	9	14	11	3	12	16	11	4	10	0,108791	0,9900	nonperiodic-templates
28.	14	8	13	7	7	6	9	11	9	16	0,334538	0,9900	nonperiodic-templates
29.	4	15	8	16	11	9	10	12	7	8	0,213309	1,0000	nonperiodic-templates
30.	13	7	9	7	8	13	10	11	13	9	0,816537	0,9700	nonperiodic-templates
31.	9	13	7	10	4	11	11	13	8	14	0,474986	0,9900	nonperiodic-templates
32.	7	6	15	9	12	18	8	6	10	9	0,122325	0,9800	nonperiodic-templates
33.	16	8	8	8	10	12	8	9	10	11	0,759756	0,9900	nonperiodic-templates
34.	10	10	7	12	9	12	14	5	12	9	0,699313	1,0000	nonperiodic-templates
35.	5	10	8	6	10	13	15	14	10	9	0,383827	0,9900	nonperiodic-templates
36.	5	7	10	12	17	5	11	13	15	5	0,045675	0,9800	nonperiodic-templates
37.	12	15	9	11	6	9	9	18	9	2	0,037566	0,9900	nonperiodic-templates

38.	17	9	8	8	12	9	5	9	11	12	0,401199	0,9900	nonperiodic-templates
39.	13	5	13	9	15	12	11	4	9	9	0,262249	0,9900	nonperiodic-templates

Продовження дод.В

Продовження табл. В.1

40.	8	9	11	5	10	9	9	14	11	14	0,678686	1,0000	nonperiodic-templates
41.	15	8	7	7	13	11	7	15	7	10	0,350485	0,9900	nonperiodic-templates
42.	10	13	10	12	11	11	8	6	8	11	0,911413	1,0000	nonperiodic-templates
43.	10	14	13	12	6	14	6	9	9	7	0,455937	0,9800	nonperiodic-templates
44.	10	9	10	11	12	10	6	11	9	12	0,971699	1,0000	nonperiodic-templates
45.	9	7	8	10	8	13	11	6	14	14	0,574903	0,9900	nonperiodic-templates
46.	7	9	12	10	7	11	11	10	10	13	0,946308	0,9900	nonperiodic-templates
47.	8	7	11	6	5	11	14	14	10	14	0,319084	0,9900	nonperiodic-templates
48.	8	6	11	8	13	12	14	13	7	8	0,574903	0,9800	nonperiodic-templates
49.	10	10	10	10	8	8	14	11	7	12	0,924076	1,0000	nonperiodic-templates
50.	7	6	11	9	7	9	14	12	14	11	0,595549	0,9900	nonperiodic-templates
51.	10	11	11	13	10	12	8	7	6	12	0,851383	0,9700	nonperiodic-templates
52.	9	12	10	8	6	12	5	15	9	14	0,383827	1,0000	nonperiodic-templates
53.	9	12	3	9	9	13	12	10	7	16	0,249284	1,0000	nonperiodic-templates
54.	8	13	13	13	5	9	8	15	7	9	0,383827	1,0000	nonperiodic-templates
55.	9	10	8	12	8	5	13	11	15	9	0,595549	0,9800	nonperiodic-templates
56.	13	9	10	12	7	17	11	5	8	8	0,304126	1,0000	nonperiodic-templates
57.	11	7	13	8	14	3	14	12	8	10	0,262249	0,9900	nonperiodic-templates
58.	15	6	15	10	8	9	10	11	11	5	0,366918	0,9700	nonperiodic-templates
59.	6	14	6	8	9	9	15	8	9	16	0,213309	0,9900	nonperiodic-templates
60.	14	10	10	18	10	6	7	7	7	11	0,191687	0,9700	nonperiodic-templates
61.	13	11	12	9	9	13	10	7	9	7	0,883171	0,9800	nonperiodic-templates
62.	10	9	11	6	10	11	6	15	11	11	0,719747	0,9700	nonperiodic-templates
63.	10	6	5	9	11	15	10	3	10	21	0,004629	1,0000	nonperiodic-templates
64.	12	13	9	13	8	15	7	12	5	6	0,304126	0,9800	nonperiodic-templates
65.	14	10	6	6	9	14	12	7	8	14	0,366918	1,0000	nonperiodic-templates
66.	12	8	9	6	7	13	8	14	12	11	0,657933	0,9900	nonperiodic-templates
67.	9	11	11	11	9	6	11	9	13	10	0,955835	0,9900	nonperiodic-templates
68.	10	10	7	12	11	11	12	6	9	12	0,911413	0,9900	nonperiodic-templates
69.	10	7	6	14	10	13	9	11	9	11	0,798139	0,9700	nonperiodic-templates
70.	8	7	11	11	7	13	12	13	7	11	0,779188	1,0000	nonperiodic-templates
71.	7	11	9	7	6	4	12	12	18	14	0,066882	0,9800	nonperiodic-templates
72.	11	15	7	12	10	8	13	8	7	9	0,678686	0,9800	nonperiodic-templates
73.	12	5	14	11	11	6	13	13	7	8	0,401199	0,9800	nonperiodic-templates
74.	11	10	8	7	12	13	4	13	11	11	0,595549	0,9900	nonperiodic-templates
75.	8	14	7	12	7	5	17	15	7	8	0,080519	1,0000	nonperiodic-templates
76.	15	6	9	10	15	9	12	9	9	6	0,437274	0,9700	nonperiodic-templates
77.	14	12	10	9	7	7	12	11	7	11	0,798139	1,0000	nonperiodic-templates
78.	10	6	8	8	14	11	3	13	10	17	0,096578	0,9900	nonperiodic-templates
79.	12	12	10	9	5	13	7	10	10	12	0,779188	0,9900	nonperiodic-templates
80.	9	13	11	9	8	7	13	9	10	11	0,935716	0,9900	nonperiodic-templates
81.	7	15	8	7	5	10	11	16	9	12	0,249284	0,9900	nonperiodic-templates

82.	8	12	11	5	11	11	13	12	9	8	0,798139	1,0000	nonperiodic-templates
83.	7	13	8	7	11	11	12	6	13	12	0,678686	0,9800	nonperiodic-templates
84.	8	16	9	10	5	11	9	4	14	14	0,137282	1,0000	nonperiodic-templates
85.	13	14	8	17	11	7	7	4	14	5	0,042808	0,9700	nonperiodic-templates

Продовження дод.В

Продовження табл. В.1

86.	8	8	7	6	9	11	16	13	16	6	0,153763	0,9900	nonperiodic-templates
87.	10	11	11	5	11	9	6	10	14	13	0,637119	1,0000	nonperiodic-templates
88.	9	8	7	7	12	11	10	16	13	7	0,514124	0,9900	nonperiodic-templates
89.	8	10	7	12	8	5	11	14	14	11	0,534146	1,0000	nonperiodic-templates
90.	15	5	9	13	9	11	7	10	11	10	0,616305	0,9600	nonperiodic-templates
91.	9	11	9	10	14	5	15	9	6	12	0,437274	0,9900	nonperiodic-templates
92.	8	8	5	15	12	10	13	9	10	10	0,616305	1,0000	nonperiodic-templates
93.	7	16	7	13	8	9	12	9	11	8	0,554420	1,0000	nonperiodic-templates
94.	9	6	6	15	9	12	6	10	15	12	0,289667	1,0000	nonperiodic-templates
95.	13	13	5	11	6	8	7	14	10	13	0,366918	1,0000	nonperiodic-templates
96.	5	12	14	10	7	16	12	2	13	9	0,051942	1,0000	nonperiodic-templates
97.	5	11	11	8	10	11	9	9	15	11	0,739918	1,0000	nonperiodic-templates
98.	11	12	8	14	4	5	20	9	12	5	0,010237	1,0000	nonperiodic-templates
99.	16	13	8	5	14	8	7	15	8	6	0,096578	1,0000	nonperiodic-templates
100.	6	14	10	9	7	16	13	12	9	4	0,171867	1,0000	nonperiodic-templates
101.	11	7	6	15	11	10	8	13	13	6	0,437274	0,9900	nonperiodic-templates
102.	10	4	15	7	12	15	11	8	7	11	0,249284	1,0000	nonperiodic-templates
103.	13	12	12	10	10	10	10	15	6	2	0,202268	0,9800	nonperiodic-templates
104.	10	8	11	9	14	8	12	15	8	5	0,494392	0,9800	nonperiodic-templates
105.	7	10	9	11	15	11	12	7	12	6	0,637119	1,0000	nonperiodic-templates
106.	13	10	9	7	13	11	11	10	11	5	0,779188	1,0000	nonperiodic-templates
107.	6	11	9	12	7	9	21	11	7	7	0,045675	0,9900	nonperiodic-templates
108.	11	8	10	7	13	12	5	8	15	11	0,514124	0,9700	nonperiodic-templates
109.	6	14	6	14	7	6	11	9	15	12	0,213309	0,9900	nonperiodic-templates
110.	10	10	11	10	7	8	14	12	8	10	0,924076	0,9800	nonperiodic-templates
111.	14	7	11	14	7	8	12	10	10	7	0,657933	0,9900	nonperiodic-templates
112.	7	9	8	7	9	15	10	12	9	14	0,637119	1,0000	nonperiodic-templates
113.	6	12	11	12	10	11	8	10	9	11	0,955835	0,9800	nonperiodic-templates
114.	12	16	7	10	12	7	6	8	10	12	0,474986	0,9900	nonperiodic-templates
115.	5	9	13	11	8	16	12	8	13	5	0,224821	0,9900	nonperiodic-templates
116.	10	11	6	10	14	16	11	8	6	8	0,401199	0,9900	nonperiodic-templates
117.	8	10	10	9	10	17	10	10	3	13	0,262249	0,9900	nonperiodic-templates
118.	5	10	11	8	12	7	14	13	11	9	0,637119	1,0000	nonperiodic-templates
119.	5	11	10	18	6	8	12	11	11	8	0,213309	0,9900	nonperiodic-templates
120.	12	6	6	9	14	16	13	5	11	8	0,171867	1,0000	nonperiodic-templates
121.	9	9	6	14	11	11	9	12	9	10	0,897763	0,9900	nonperiodic-templates
122.	8	10	12	14	14	7	3	12	9	11	0,319084	1,0000	nonperiodic-templates
123.	9	10	3	10	18	12	7	9	12	10	0,153763	0,9900	nonperiodic-templates
124.	11	7	8	13	7	11	8	13	11	11	0,851383	1,0000	nonperiodic-templates
125.	3	5	16	16	13	4	7	9	11	16	0,004629	1,0000	nonperiodic-templates

126.	11	10	8	9	10	17	9	11	4	11	0,401199	1,0000	nonperiodic-templates
127.	12	9	12	7	6	7	9	15	12	11	0,595549	0,9800	nonperiodic-templates
128.	10	5	10	10	10	9	12	16	5	13	0,350485	0,9900	nonperiodic-templates
129.	13	4	12	11	7	12	12	8	7	14	0,383827	0,9900	nonperiodic-templates
130.	5	11	12	8	13	8	13	4	11	15	0,224821	0,9800	nonperiodic-templates
131.	15	11	10	13	7	11	9	8	7	9	0,739918	0,9800	nonperiodic-templates

Продовження дод.В

Продовження табл. В.1

132.	8	5	12	11	8	15	7	14	11	9	0,437274	1,0000	nonperiodic-templates
133.	8	12	10	7	12	9	6	16	11	9	0,574903	0,9800	nonperiodic-templates
134.	9	18	12	6	13	7	14	4	9	8	0,066882	1,0000	nonperiodic-templates
135.	6	12	21	6	14	6	4	10	14	7	0,002971	1,0000	nonperiodic-templates
136.	12	3	18	7	5	11	14	8	11	11	0,042808	1,0000	nonperiodic-templates
137.	12	6	12	11	13	13	11	8	8	6	0,657933	0,9900	nonperiodic-templates
138.	11	7	14	12	6	7	10	11	7	15	0,437274	0,9900	nonperiodic-templates
139.	11	10	4	13	14	15	4	5	12	12	0,075719	0,9800	nonperiodic-templates
140.	8	9	11	10	13	8	12	6	9	14	0,779188	1,0000	nonperiodic-templates
141.	8	13	8	9	8	14	9	9	10	12	0,883171	0,9800	nonperiodic-templates
142.	11	9	9	12	8	8	8	6	15	14	0,574903	0,9700	nonperiodic-templates
143.	14	17	5	9	12	13	5	10	7	8	0,115387	1,0000	nonperiodic-templates
144.	10	5	10	9	11	8	11	9	14	13	0,759756	1,0000	nonperiodic-templates
145.	11	5	6	10	17	8	8	13	11	11	0,275709	0,9800	nonperiodic-templates
146.	10	4	13	13	12	8	10	12	9	9	0,657933	0,9900	nonperiodic-templates
147.	12	7	10	12	7	6	8	15	15	8	0,350485	0,9700	nonperiodic-templates
148.	9	13	11	12	6	10	13	8	8	10	0,851383	1,0000	nonperiodic-templates
149.	10	9	11	12	11	6	8	9	18	6	0,289667	1,0000	nonperiodic-templates
150.	12	9	12	8	10	9	11	10	3	16	0,350485	1,0000	nonperiodic-templates
151.	12	6	18	17	9	9	7	7	7	8	0,055361	1,0000	nonperiodic-templates
152.	11	6	10	13	11	7	12	13	8	9	0,798139	0,9900	nonperiodic-templates
153.	6	5	8	13	13	9	10	10	14	12	0,494392	1,0000	nonperiodic-templates
154.	15	5	13	11	15	15	7	5	8	6	0,058984	0,9900	nonperiodic-templates
155.	9	11	9	12	6	8	12	10	11	12	0,935716	1,0000	nonperiodic-templates
156.	8	12	11	5	11	11	13	12	9	8	0,798139	1,0000	nonperiodic-templates
157.	10	10	15	4	10	13	8	11	7	12	0,455937	0,9700	overlapping-templates
158.	10	15	10	10	5	8	10	13	13	6	0,455937	1,0000	universal
159.	8	11	10	10	11	9	13	10	6	12	0,935716	0,9900	apen
160.	5	6	8	8	5	6	5	6	3	9	0,848588	0,9836	random-excursions
161.	3	4	6	7	5	7	7	7	7	8	0,922036	1,0000	random-excursions
162.	3	9	5	4	4	11	8	4	9	4	0,204076	1,0000	random-excursions
163.	6	7	9	9	7	4	4	4	9	2	0,392456	1,0000	random-excursions
164.	5	9	7	8	4	4	10	4	6	4	0,551026	1,0000	random-excursions
165.	4	10	3	6	9	8	4	4	8	5	0,422034	1,0000	random-excursions
166.	3	3	7	8	4	6	9	6	9	6	0,585209	0,9836	random-excursions
167.	8	5	3	1	9	9	7	5	6	8	0,311542	1,0000	random-excursions
168.	2	7	7	5	6	7	4	11	4	8	0,392456	1,0000	random-excursions-variant
169.	3	8	5	3	5	4	9	8	7	9	0,484646	1,0000	random-excursions-variant
170.	4	5	6	4	5	3	8	5	11	10	0,287306	1,0000	random-excursions-variant
171.	4	6	4	6	4	6	9	4	10	8	0,585209	1,0000	random-excursions-variant

172.	4	5	4	6	4	9	8	7	9	5	0,723129	1,0000	random-excursions-variant
173.	5	3	6	8	11	3	7	7	7	4	0,422034	1,0000	random-excursions-variant
174.	5	10	7	6	7	5	10	2	4	5	0,392456	1,0000	random-excursions-variant
175.	8	12	4	4	5	1	8	6	4	9	0,086458	1,0000	random-excursions-variant
176.	5	9	5	11	4	6	8	3	4	6	0,392456	0,9836	random-excursions-variant
177.	11	4	4	8	2	12	6	4	4	6	0,063482	0,9836	random-excursions-variant
178.	6	5	5	6	4	8	5	5	9	8	0,900104	0,9836	random-excursions-variant
179.	4	5	4	5	6	5	5	5	12	10	0,287306	0,9836	random-excursions-variant

Продовження дод.В

Закінчення табл. В.1

180.	5	5	5	5	2	6	10	6	11	6	0,337162	1,0000	random-excursions-variant
181.	3	8	2	6	4	7	8	3	8	12	0,105618	1,0000	random-excursions-variant
182.	3	5	8	5	3	7	4	12	11	3	0,057146	1,0000	random-excursions-variant
183.	5	6	4	6	9	4	4	8	8	7	0,819544	1,0000	random-excursions-variant
184.	2	10	4	6	6	7	3	10	7	6	0,311542	1,0000	random-excursions-variant
185.	4	6	3	8	6	5	5	6	5	13	0,242986	1,0000	random-excursions-variant
186.	13	7	19	6	9	9	12	10	6	9	0,129620	1,0000	serial

Таблиця В.2

Результати статистичного тестування генератора GPSSD

	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	P-value	Proportion	Statistical test
1.	9	9	8	8	17	6	8	16	11	8	0,213309	0,9900	frequency
2.	12	9	11	10	7	9	6	10	11	15	0,759756	0,9800	block-frequency
3.	8	12	14	8	7	11	8	12	9	11	0,851383	0,9900	cumulative-sums
4.	8	9	8	10	13	13	11	10	6	12	0,851383	0,9900	cumulative-sums
5.	18	14	5	8	6	6	11	12	12	8	0,080519	0,9800	runs
6.	16	11	13	10	11	9	6	5	6	13	0,249284	0,9600	longest-run
7.	6	9	8	9	9	9	8	11	15	16	0,437274	0,9900	rank
8.	7	10	15	14	10	8	11	8	9	8	0,699313	1,0000	fft
9.	10	12	14	14	10	9	10	9	6	6	0,637119	1,0000	nonperiodic-templates
10.	12	8	5	13	7	9	11	13	12	10	0,678686	0,9800	nonperiodic-templates
11.	10	11	15	7	7	10	14	14	7	5	0,275709	1,0000	nonperiodic-templates
12.	11	13	10	6	9	13	7	6	14	11	0,554420	1,0000	nonperiodic-templates
13.	11	5	13	8	17	6	12	9	7	12	0,202268	0,9900	nonperiodic-templates
14.	6	6	11	11	15	5	9	16	11	10	0,202268	1,0000	nonperiodic-templates
15.	11	14	14	4	10	14	8	6	8	11	0,275709	0,9900	nonperiodic-templates
16.	12	14	12	10	16	5	7	7	10	7	0,262249	1,0000	nonperiodic-templates
17.	11	17	12	15	9	6	5	8	9	8	0,162606	0,9700	nonperiodic-templates
18.	5	13	8	9	11	14	15	8	10	7	0,401199	1,0000	nonperiodic-templates
19.	10	10	10	10	8	11	10	10	11	10	0,999934	0,9800	nonperiodic-templates
20.	11	12	12	8	12	6	7	11	10	11	0,883171	1,0000	nonperiodic-templates
21.	13	8	13	6	10	7	15	6	10	12	0,419021	1,0000	nonperiodic-templates
22.	15	9	12	17	5	12	4	14	9	3	0,012650	1,0000	nonperiodic-templates
23.	12	5	13	8	7	14	10	12	10	9	0,616305	1,0000	nonperiodic-templates
24.	9	11	9	7	12	13	5	12	8	14	0,595549	0,9800	nonperiodic-templates

25.	13	9	15	8	6	11	10	9	9	10	0,759756	0,9900	nonperiodic-templates
26.	13	15	9	6	11	9	10	11	8	8	0,719747	0,9800	nonperiodic-templates
27.	8	15	7	6	11	8	11	13	12	9	0,595549	0,9900	nonperiodic-templates
28.	9	3	17	9	7	11	10	13	12	9	0,191687	0,9900	nonperiodic-templates
29.	11	11	10	4	16	12	10	8	10	8	0,474986	0,9900	nonperiodic-templates
30.	13	8	11	11	9	7	10	13	6	12	0,798139	0,9700	nonperiodic-templates
31.	12	11	10	12	8	8	12	8	12	7	0,924076	0,9900	nonperiodic-templates
32.	7	12	9	12	9	5	11	12	13	10	0,759756	1,0000	nonperiodic-templates
33.	7	12	17	6	12	11	10	9	7	9	0,401199	1,0000	nonperiodic-templates
34.	11	12	6	14	9	12	11	4	6	15	0,213309	0,9900	nonperiodic-templates

Продовження дод.В

Продовження табл. В.2

35.	10	4	11	12	13	10	6	15	11	8	0,383827	0,9800	nonperiodic-templates
36.	6	12	10	9	9	9	9	8	13	15	0,719747	1,0000	nonperiodic-templates
37.	16	7	11	10	5	14	13	9	10	5	0,202268	1,0000	nonperiodic-templates
38.	6	7	6	10	15	11	15	12	10	8	0,350485	1,0000	nonperiodic-templates
39.	6	13	16	6	9	10	6	16	10	8	0,145326	1,0000	nonperiodic-templates
40.	10	11	10	12	5	20	12	7	5	8	0,045675	0,9900	nonperiodic-templates
41.	3	10	12	15	14	8	7	13	10	8	0,213309	1,0000	nonperiodic-templates
42.	16	10	13	8	7	7	11	12	11	5	0,366918	1,0000	nonperiodic-templates
43.	7	9	11	12	12	7	8	10	12	12	0,911413	1,0000	nonperiodic-templates
44.	8	5	9	15	10	9	14	17	8	5	0,090936	1,0000	nonperiodic-templates
45.	15	7	9	10	9	18	10	4	7	11	0,102526	0,9800	nonperiodic-templates
46.	4	15	11	14	12	9	8	12	9	6	0,289667	1,0000	nonperiodic-templates
47.	13	11	14	12	5	10	9	8	8	10	0,699313	0,9800	nonperiodic-templates
48.	11	14	8	11	5	10	15	10	9	7	0,514124	1,0000	nonperiodic-templates
49.	12	9	13	11	14	6	6	7	10	12	0,574903	1,0000	nonperiodic-templates
50.	7	9	14	14	7	12	8	11	6	12	0,534146	0,9900	nonperiodic-templates
51.	8	12	10	13	13	11	8	11	8	6	0,816537	0,9700	nonperiodic-templates
52.	11	8	12	12	8	7	9	8	12	13	0,883171	0,9900	nonperiodic-templates
53.	13	10	8	4	14	8	13	10	13	7	0,383827	1,0000	nonperiodic-templates
54.	14	7	7	11	13	6	9	5	17	11	0,137282	0,9900	nonperiodic-templates
55.	11	13	10	5	13	8	10	12	10	8	0,779188	0,9900	nonperiodic-templates
56.	11	7	8	5	5	18	10	12	11	13	0,115387	0,9600	nonperiodic-templates
57.	8	13	7	14	11	7	8	11	11	10	0,798139	0,9900	nonperiodic-templates
58.	14	9	10	6	7	11	9	6	11	17	0,275709	1,0000	nonperiodic-templates
59.	14	12	11	8	8	10	9	12	10	6	0,834308	0,9900	nonperiodic-templates
60.	13	7	11	12	6	12	8	8	10	13	0,739918	0,9700	nonperiodic-templates
61.	11	13	11	8	10	15	9	7	4	12	0,437274	0,9800	nonperiodic-templates
62.	3	10	18	15	7	9	8	11	8	11	0,071177	1,0000	nonperiodic-templates
63.	14	6	9	14	5	11	9	11	11	10	0,554420	0,9800	nonperiodic-templates
64.	16	5	13	10	1	5	18	6	13	13	0,001201	0,9900	nonperiodic-templates
65.	9	12	10	13	13	11	9	8	5	10	0,798139	0,9900	nonperiodic-templates
66.	10	6	2	11	11	17	11	8	11	13	0,102526	0,9900	nonperiodic-templates
67.	15	6	9	10	5	13	11	13	8	10	0,437274	0,9800	nonperiodic-templates
68.	4	8	12	12	10	9	11	9	10	15	0,574903	1,0000	nonperiodic-templates
69.	14	9	13	5	12	10	12	6	6	13	0,350485	0,9900	nonperiodic-templates
70.	9	7	12	18	5	12	10	9	11	7	0,224821	1,0000	nonperiodic-templates

71.	11	7	11	11	12	11	4	10	19	4	0,048716	1,0000	nonperiodic-templates
72.	9	8	11	5	10	12	14	5	16	10	0,262249	0,9800	nonperiodic-templates
73.	7	8	12	14	11	4	14	11	8	11	0,419021	1,0000	nonperiodic-templates
74.	7	11	12	13	11	8	8	11	10	9	0,946308	0,9900	nonperiodic-templates
75.	8	8	5	9	12	16	15	11	4	12	0,122325	0,9900	nonperiodic-templates
76.	5	14	7	10	8	11	9	12	14	10	0,574903	0,9900	nonperiodic-templates
77.	14	15	7	6	7	10	7	16	7	11	0,162606	0,9900	nonperiodic-templates
78.	12	12	9	10	7	9	11	8	14	8	0,883171	1,0000	nonperiodic-templates
79.	11	14	7	5	12	7	12	13	9	10	0,554420	0,9900	nonperiodic-templates
80.	14	10	11	6	8	5	13	11	11	11	0,595549	0,9900	nonperiodic-templates
81.	11	8	14	12	5	11	10	10	9	10	0,816537	0,9900	nonperiodic-templates
82.	13	8	13	9	15	8	10	9	7	8	0,678686	1,0000	nonperiodic-templates

Продовження дод.В

Продовження табл. В.2

83.	10	12	14	14	10	9	10	9	6	6	0,637119	1,0000	nonperiodic-templates
84.	12	11	11	9	13	14	7	9	3	11	0,419021	1,0000	nonperiodic-templates
85.	12	11	8	11	8	7	7	10	11	15	0,759756	0,9800	nonperiodic-templates
86.	8	8	9	16	14	12	5	10	7	11	0,350485	0,9900	nonperiodic-templates
87.	8	7	14	7	7	7	7	15	17	11	0,122325	1,0000	nonperiodic-templates
88.	7	11	10	13	6	8	11	14	11	9	0,759756	0,9900	nonperiodic-templates
89.	8	12	11	9	11	11	10	10	8	10	0,996335	1,0000	nonperiodic-templates
90.	9	7	16	6	11	4	8	15	15	9	0,080519	0,9800	nonperiodic-templates
91.	7	5	11	13	14	11	12	9	10	8	0,637119	1,0000	nonperiodic-templates
92.	14	9	9	6	10	13	12	6	12	9	0,657933	0,9800	nonperiodic-templates
93.	8	12	8	13	8	7	7	11	12	14	0,699313	1,0000	nonperiodic-templates
94.	13	8	7	11	11	11	9	9	5	16	0,455937	1,0000	nonperiodic-templates
95.	12	10	7	13	10	9	9	6	14	10	0,779188	0,9800	nonperiodic-templates
96.	14	8	8	6	13	15	9	4	18	5	0,017912	0,9800	nonperiodic-templates
97.	8	14	9	12	13	7	9	7	7	14	0,554420	0,9900	nonperiodic-templates
98.	8	7	11	14	12	13	7	11	9	8	0,759756	0,9900	nonperiodic-templates
99.	10	4	13	8	8	7	13	13	8	16	0,213309	0,9900	nonperiodic-templates
100.	15	4	9	6	8	10	19	9	11	9	0,055361	0,9900	nonperiodic-templates
101.	8	18	10	9	6	11	11	9	10	8	0,419021	0,9900	nonperiodic-templates
102.	5	13	9	7	8	18	11	9	10	10	0,249284	1,0000	nonperiodic-templates
103.	11	7	8	6	17	12	9	11	10	9	0,474986	1,0000	nonperiodic-templates
104.	12	9	9	7	9	6	12	11	11	14	0,798139	0,9900	nonperiodic-templates
105.	19	7	12	11	4	10	9	8	12	8	0,108791	1,0000	nonperiodic-templates
106.	7	5	10	9	9	16	15	9	9	11	0,350485	0,9900	nonperiodic-templates
107.	12	7	15	9	15	9	9	6	11	7	0,419021	0,9700	nonperiodic-templates
108.	8	13	6	9	14	9	8	11	9	13	0,719747	0,9900	nonperiodic-templates
109.	13	14	11	7	10	9	9	10	7	10	0,867692	0,9800	nonperiodic-templates
110.	12	8	12	2	10	7	14	6	17	12	0,048716	1,0000	nonperiodic-templates
111.	10	8	9	13	7	14	7	13	5	14	0,366918	0,9700	nonperiodic-templates
112.	13	4	9	14	9	7	14	12	8	10	0,383827	0,9900	nonperiodic-templates
113.	9	10	10	11	12	10	9	9	9	11	0,999438	1,0000	nonperiodic-templates
114.	13	6	6	12	10	14	10	8	11	10	0,678686	0,9900	nonperiodic-templates
115.	7	10	11	11	15	9	9	9	14	5	0,534146	1,0000	nonperiodic-templates
116.	7	15	6	14	14	8	11	9	7	9	0,366918	1,0000	nonperiodic-templates

117.	7	11	9	7	15	8	16	8	11	8	0,401199	0,9900	nonperiodic-templates
118.	16	8	5	6	13	11	9	12	9	11	0,366918	0,9900	nonperiodic-templates
119.	12	6	11	12	11	9	6	6	12	15	0,455937	0,9900	nonperiodic-templates
120.	10	8	10	9	11	11	8	13	7	13	0,924076	0,9900	nonperiodic-templates
121.	9	9	9	13	9	12	4	14	11	10	0,637119	0,9900	nonperiodic-templates
122.	11	3	13	4	10	9	11	16	13	10	0,115387	0,9900	nonperiodic-templates
123.	8	15	7	13	11	8	9	14	7	8	0,514124	0,9900	nonperiodic-templates
124.	11	9	9	10	12	10	8	6	15	10	0,816537	0,9700	nonperiodic-templates
125.	9	15	7	14	8	4	13	10	8	12	0,289667	0,9900	nonperiodic-templates
126.	11	13	10	7	10	9	12	9	10	9	0,978072	0,9900	nonperiodic-templates
127.	8	10	9	11	7	20	8	12	7	8	0,137282	0,9900	nonperiodic-templates
128.	5	8	12	12	6	12	14	6	13	12	0,334538	0,9900	nonperiodic-templates
129.	5	12	7	11	5	6	16	13	11	14	0,115387	1,0000	nonperiodic-templates
130.	15	9	10	11	8	8	6	9	11	13	0,719747	0,9900	nonperiodic-templates

Продовження дод.В

Продовження табл. В.2

131.	13	9	13	8	8	8	13	7	11	10	0,834308	1,0000	nonperiodic-templates
132.	8	10	18	8	7	5	9	10	13	12	0,213309	0,9900	nonperiodic-templates
133.	6	5	12	12	11	12	5	12	11	14	0,350485	1,0000	nonperiodic-templates
134.	8	12	6	15	8	10	10	11	13	7	0,616305	0,9900	nonperiodic-templates
135.	8	9	14	11	11	11	5	13	11	7	0,657933	0,9900	nonperiodic-templates
136.	7	16	8	7	7	8	9	11	17	10	0,202268	1,0000	nonperiodic-templates
137.	10	9	9	10	9	14	11	8	7	13	0,897763	1,0000	nonperiodic-templates
138.	12	11	5	6	11	13	14	10	7	11	0,514124	0,9900	nonperiodic-templates
139.	7	8	10	9	11	21	10	4	8	12	0,035174	0,9900	nonperiodic-templates
140.	12	10	8	6	13	11	10	11	12	7	0,851383	0,9900	nonperiodic-templates
141.	9	7	9	12	8	10	9	13	11	12	0,946308	0,9800	nonperiodic-templates
142.	8	8	10	8	12	11	9	9	11	14	0,935716	1,0000	nonperiodic-templates
143.	9	9	8	10	15	9	5	9	15	11	0,494392	0,9900	nonperiodic-templates
144.	9	9	10	16	9	5	10	12	11	9	0,637119	0,9800	nonperiodic-templates
145.	8	8	11	15	7	10	11	11	12	7	0,759756	1,0000	nonperiodic-templates
146.	11	12	15	12	7	10	11	9	5	8	0,595549	0,9800	nonperiodic-templates
147.	5	10	7	10	7	12	16	10	10	13	0,419021	1,0000	nonperiodic-templates
148.	9	14	8	14	8	14	6	5	14	8	0,224821	0,9800	nonperiodic-templates
149.	10	12	8	10	8	8	16	7	9	12	0,678686	0,9800	nonperiodic-templates
150.	11	6	13	11	16	6	9	4	10	14	0,153763	1,0000	nonperiodic-templates
151.	11	4	12	14	14	10	11	7	8	9	0,455937	0,9800	nonperiodic-templates
152.	10	8	5	9	12	15	10	10	8	13	0,616305	0,9900	nonperiodic-templates
153.	9	14	11	9	5	12	7	12	9	12	0,678686	1,0000	nonperiodic-templates
154.	14	7	13	8	9	6	10	10	14	9	0,616305	0,9800	nonperiodic-templates
155.	8	10	15	8	4	14	9	12	15	5	0,122325	0,9800	nonperiodic-templates
156.	13	8	13	9	15	8	10	9	7	8	0,678686	1,0000	nonperiodic-templates
157.	8	11	11	16	13	8	4	12	11	6	0,262249	0,9900	overlapping-templates
158.	13	12	7	9	9	8	8	10	14	10	0,851383	0,9900	universal
159.	10	9	15	10	7	17	7	10	8	7	0,304126	0,9900	apen
160.	6	6	8	5	11	1	8	4	6	4	0,090936	0,9831	random-excursions
161.	7	8	5	3	11	4	5	6	6	4	0,249284	1,0000	random-excursions
162.	6	10	6	4	6	6	4	3	8	6	0,437274	0,9661	random-excursions

163.	6	10	4	8	8	4	7	5	4	3	0,275709	1,0000	random-excursions
164.	3	8	8	4	6	4	7	4	10	5	0,275709	1,0000	random-excursions
165.	5	6	2	9	6	2	8	6	10	5	0,115387	1,0000	random-excursions
166.	1	5	8	4	13	5	2	7	10	4	0,002203	0,9831	random-excursions
167.	2	3	3	6	8	7	2	10	7	11	0,012650	1,0000	random-excursions
168.	10	2	8	7	6	6	5	4	8	3	0,181557	0,9661	random-excursions
169.	9	4	3	6	8	5	9	6	5	4	0,366918	0,9831	random-excursions
170.	8	6	4	7	5	7	9	5	6	2	0,437274	0,9661	random-excursions
171.	8	6	8	4	7	7	5	4	7	3	0,595549	0,9831	random-excursions
172.	7	4	9	5	8	3	8	6	2	7	0,249284	0,9600	random-excursions
173.	7	6	2	10	7	3	8	5	7	4	0,202268	0,9831	random-excursions
174.	5	8	5	10	4	6	3	8	7	3	0,249284	0,9831	random-excursions
175.	9	6	4	6	6	7	3	0	7	11	0,028817	0,9831	random-excursions
176.	10	8	7	3	4	3	3	6	7	8	0,162606	1,0000	random-excursions
177.	5	7	5	2	8	10	5	8	5	4	0,249284	1,0000	random-excursions
178.	4	6	5	3	9	6	4	7	8	7	0,514124	0,9831	random-excursions

Продовження дод.В

Продовження табл. В.2

179.	13	9	13	8	8	8	13	7	11	10	0,834308	1,0000	nonperiodic-templates
180.	8	10	18	8	7	5	9	10	13	12	0,213309	0,9900	nonperiodic-templates
181.	6	5	12	12	11	12	5	12	11	14	0,350485	1,0000	nonperiodic-templates
182.	8	12	6	15	8	10	10	11	13	7	0,616305	0,9900	nonperiodic-templates
183.	8	9	14	11	11	11	5	13	11	7	0,657933	0,9900	nonperiodic-templates
184.	7	16	8	7	7	8	9	11	17	10	0,202268	1,0000	nonperiodic-templates
185.	10	9	9	10	9	14	11	8	7	13	0,897763	1,0000	nonperiodic-templates
186.	12	11	5	6	11	13	14	10	7	11	0,514124	0,9900	nonperiodic-templates
187.	7	8	10	9	11	21	10	4	8	12	0,035174	0,9900	nonperiodic-templates
188.	12	10	8	6	13	11	10	11	12	7	0,851383	0,9900	nonperiodic-templates
189.	9	7	9	12	8	10	9	13	11	12	0,946308	0,9800	nonperiodic-templates
190.	8	8	10	8	12	11	9	9	11	14	0,935716	1,0000	nonperiodic-templates
191.	9	9	8	10	15	9	5	9	15	11	0,494392	0,9900	nonperiodic-templates
192.	9	9	10	16	9	5	10	12	11	9	0,637119	0,9800	nonperiodic-templates
193.	8	8	11	15	7	10	11	11	12	7	0,759756	1,0000	nonperiodic-templates
194.	11	12	15	12	7	10	11	9	5	8	0,595549	0,9800	nonperiodic-templates
195.	5	10	7	10	7	12	16	10	10	13	0,419021	1,0000	nonperiodic-templates
196.	9	14	8	14	8	14	6	5	14	8	0,224821	0,9800	nonperiodic-templates
197.	10	12	8	10	8	8	16	7	9	12	0,678686	0,9800	nonperiodic-templates
198.	11	6	13	11	16	6	9	4	10	14	0,153763	1,0000	nonperiodic-templates
199.	11	4	12	14	14	10	11	7	8	9	0,455937	0,9800	nonperiodic-templates
200.	10	8	5	9	12	15	10	10	8	13	0,616305	0,9900	nonperiodic-templates
201.	9	14	11	9	5	12	7	12	9	12	0,678686	1,0000	nonperiodic-templates
202.	14	7	13	8	9	6	10	10	14	9	0,616305	0,9800	nonperiodic-templates
203.	8	10	15	8	4	14	9	12	15	5	0,122325	0,9800	nonperiodic-templates
204.	13	8	13	9	15	8	10	9	7	8	0,678686	1,0000	nonperiodic-templates
205.	8	11	11	16	13	8	4	12	11	6	0,262249	0,9900	overlapping-templates
206.	13	12	7	9	9	8	8	10	14	10	0,851383	0,9900	universal
207.	10	9	15	10	7	17	7	10	8	7	0,304126	0,9900	apen
208.	6	6	8	5	11	1	8	4	6	4	0,090936	0,9831	random-excursions

209.	7	8	5	3	11	4	5	6	6	4	0,249284	1,0000	random-excursions
210.	6	10	6	4	6	6	4	3	8	6	0,437274	0,9661	random-excursions
211.	6	10	4	8	8	4	7	5	4	3	0,275709	1,0000	random-excursions
212.	3	8	8	4	6	4	7	4	10	5	0,275709	1,0000	random-excursions
213.	5	6	2	9	6	2	8	6	10	5	0,115387	1,0000	random-excursions
214.	1	5	8	4	13	5	2	7	10	4	0,002203	0,9831	random-excursions
215.	2	3	3	6	8	7	2	10	7	11	0,012650	1,0000	random-excursions
216.	10	2	8	7	6	6	5	4	8	3	0,181557	0,9661	random-excursions
217.	9	4	3	6	8	5	9	6	5	4	0,366918	0,9831	random-excursions
218.	8	6	4	7	5	7	9	5	6	2	0,437274	0,9661	random-excursions
219.	8	6	8	4	7	7	5	4	7	3	0,595549	0,9831	random-excursions
220.	7	4	9	5	8	3	8	6	2	7	0,249284	0,9600	random-excursions
221.	7	6	2	10	7	3	8	5	7	4	0,202268	0,9831	random-excursions
222.	5	8	5	10	4	6	3	8	7	3	0,249284	0,9831	random-excursions
223.	9	6	4	6	6	7	3	0	7	11	0,028817	0,9831	random-excursions
224.	10	8	7	3	4	3	3	6	7	8	0,162606	1,0000	random-excursions
225.	5	7	5	2	8	10	5	8	5	4	0,249284	1,0000	random-excursions
226.	4	6	5	3	9	6	4	7	8	7	0,514124	0,9831	random-excursions

Закінчення дод.В

Закінченя табл. В.2

227.	3	7	5	5	9	7	5	9	7	2	0,249284	0,9831	random-excursions
228.	2	6	9	6	7	5	4	8	3	9	0,202268	1,0000	random-excursions
229.	2	9	5	6	8	3	7	6	7	6	0,366918	1,0000	random-excursions
230.	4	4	6	5	5	10	8	6	8	3	0,334538	1,0000	random-excursions
231.	3	5	5	3	3	13	7	5	5	10	0,012650	1,0000	random-excursions
232.	1	5	2	6	8	7	3	3	13	11	0,000555	1,0000	random-excursions
233.	2	5	6	3	5	5	3	7	9	14	0,004629	1,0000	random-excursions
234.	4	17	7	13	9	11	14	7	7	11	0,122325	1,0000	serial
235.	10	13	6	12	6	9	10	13	6	15	0,383827	1,0000	serial
236.	7	9	16	20	9	8	7	8	9	7	0,042808	1,0000	lempel-ziv
237.	10	10	7	11	15	8	11	13	9	6	0,678686	0,9900	linear-complexity

Таблиця В1.1

Зведенні данні за тестами

№ н/п	Статистичний тест	Статистика тесту c(S)	Шуканий дефект
1	2	3	4
1	Частотний (монобітний) тест	Нормалізована абсолютна сума значень елементів послідовності	Занадто багато нулів або одиниць у послідовності
2	Частотний тест всередині блоку	Міра узгодження спостережуваного кількості одиниць всередині блоку з теоретично очікуваним.	Локалізовані відхилення частоти появи одиниць в блоці від ідеального значення S.
3	Перевірка накопичених сум	Максимальне відхилення значення накопиченої суми елементів послідовності від початкової точки відліку (точка 0)	Велике значення одиниць або нулів на початку або в кінці двійковій послідовності.
4	Перевірка серій	Загальна кількість серій на всій довжині послідовності.	Занадто швидка або занадто повільна зміна знака під час

			генерації послідовності
5	Перевірка максимальної довжини серії в блоці	Міра узгодження спостережуваного значення максимальної довжини одиничною серії з теоретично очікуваним значенням.	Відхилення від теоретичного закону розподілу максимальних довжин серій одиниць.
6	Перевірка рангу двійковій матриці	Міра узгодження спостережуваного значення рангів різного порядку з теоретично очікуваним.	Відхилення емпіричного закону розподілу значень рангів матриці від теоретичного, що вказує на залежність символів у послідовності.
7	Спектральний тест на основі дискретного перетворення Фур'є	Нормалізована різниця між очікуваною і кількістю, що спостерігаються частотних компонент, які перевищують 95% граничний рівень.	Виявлення періодичних складових (трендів) у двійковій послідовності.
8	Перевірка перекриваючих шаблонів	Міра узгодження спостережуваної кількості перекриваючих шаблонів у послідовності з теоретичним значенням.	Велика кількість m-бітових серій з одиниць у послідовності.
9	Універсальний тест Маурер	Сума логарифма відстані між l-бітними шаблонами.	Стисливість послідовності.
10	Ентропійний тест	Міра узгодження спостережуваного значення ентропії джерела з теоретично очікуваним для випадкового джерела.	Нерівномірність розподілу m-бітових слів у послідовності (регулярність властивостей джерела).

Таблиця В 1.2

Результати експериментальних досліджень ГПВЧ

Генератор	Кількість тестів, в яких тестування пройшло М послідовностей (%)		
	M ≥ 99%	M ≥ 96%	M < 96%
G using SHA-1	122(65%)	188 (99,5%)	1 (0,5%)
Linear Congruential	139 (74%)	189 (100%)	–
Micali-Schnorr	130 (69%)	189 (100%)	–
Quadratic Congruential	124 (66%)	181 (96%)	8 (4%)
ANSI X9.17 (3-DES)	121 (64%)	187 (98%)	4 (2%)
Blum-Blum-Shub	134 (71%)	189 (100%)	–
FIPS 197	126 (67%)	189 (100%)	–
GPSSD для кода (1024,453,128)	140 (76%)	189 (100%)	–

Таблиця В 3.1

Результати порівняльних досліджень статистичної безпеки запропонованого і деяких відомих генераторів

н/п	Генератор псевдовипадкових чисел	Кількість тестів, у яких тестування пройшло [послідовностей (%)		
		M ≥ 99%	M ≥ 96%	M < 96%
1	G using SHA-1	122(65%)	188 (99,5%)	1 (0,5%)
2	Linear Congruential	139 (74%)	189 (100%)	–
3	Micali-Schnorr	130 (69%)	189 (100%)	–

4	Quadratic Congruential	124 (66%)	181 (96%)	8 (4%)
5	ANSI X9.17 (3-DES)	121 (64%)	187 (98%)	4 (2%)
6	BBS	134 (71%)	189 (100%)	–
7	G using DES	142 (75%)	188 (99,5%)	1 (0,5%)
8	GPSSD (метод-прототип)	140 (74%)	189 (100%)	–
9	Удосконалений генератор на надлишкових блокових кодах	140 (74%)	189 (100%)	–