Міністерство освіти і науки України
Національний університет “Львівська політехніка”

На правах рукопису

ІГНАТОВИЧ АНАТОЛІЙ ОЛЕКСАНДРОВИЧ

УДК 004.021:004.027:004.6:004.77

МЕТОДИ ПІДВИЩЕННЯ ЕФЕКТИВНОСТІ КОМПОНЕНТІВ БЕЗПЕКИ КОМП’ЮТЕРНИХ СИСТЕМ З ВИКОРИСТАННЯМ МАСКУЮЧИХ ЕЛЕМЕНТІВ ТЕКСТОВИХ ТА БІОМЕТРИЧНИХ ДАНИХ

05.13.05 – комп’ютерні системи та компоненти

Дисертація на здобуття наукового ступеня
кандидата технічних наук

Науковий керівник -
кандидат технічних наук,
доцент **Парамуд Я. С.**

Львів – 2016

ЗМІСТ

СПИСОК СКОРОЧЕНЬ

АІ – Автентифікатори інформації.

ВО – Віртуальна організація.

ВТ – Відкритий текст.

ЕлІТ – Електроніка та інформаційні технології. Назва конференції.

ОС – Операційна система.

СЗІ – Система захисту інформації.

ШТ – Шифрований текст.

AD – Active Directory. Служба каталогів.

ACIT – Advanced Computer Information Technologies. Назва конференції.

ACSN – Access Control Systems and Networks. Назва конференції.

AFIS – Automated Fingerprint Identification Biometric System. Біометрична система ідентифікації за відбитком пальця.

AS – Authentication Service. Служба автентифікації.

CHAP – Challenge-Handhake Authentication Protocol. Протокол автентифікації запит-підтвердження.

DC – Domain Controller. Контролер домену.

DNS – Domain Name Service. Служба доменних імен.

EAP – PPP Extensible Authentication Protocol. Розширений PPP-протокол автентифікації.

ECP – PPP Encryption Control Protocol. PPP-протокол управління шифруванням.

FAR – False Acceptance Rate. Імовірність хибного збігу біометричних властивостей двох людей.

FRR – False Rejection Rate. Імовірність відмови в авторизації людині, яка має доступ.

IPSec – Internet Protocol Security. Безпека мережевого протоколу.

KDC – Key Distribution Center. Центр розподілу ключів.

LCP – Link Control Protocol. Протокол управління з’єднанням.

LM – LAN Manager. Протокол мережевої автентифікації.

LSA – Local Security Authority. Розпорядник локальної безпеки.

L2TP – Layer 2 Tunneling Protocol. Протокол тунелювання канального рівня.

NCP – Network Control Protocol. Протокол управління мережею.

NIST – The National Institute of Standards and Technology, USA. Тест

PAP – Password Authentication Protocol. Протокол перевірки пароля.

PCT – Private Communication Technology. Технологія конфіденційного зв'язку.

PI – Personal Identifier. Персональний ідентифікатор.

PPP – Point-to-Point Protocol. Протокол двоточкового з'єднання.

PPTP – Point-to-Point Tunneling Protocol. Протокол тунелювання між двоточковими вузлами.

RADIUS – Remote Authentication Dial-In User Service. Сервіс віддаленої автентифікації користувачів через комутовані лінії.

SAM – Security Account Management. Управління обліковими записами безпеки.

SID – Security Identifier. Ідентифікатор безпеки.

SSL – Secure Sockets Layer. Протокол захищених сокетів,

SSH – Secure Shell. Захищена оболонка.

TLS – Transport Layer Security. Протокол безпеки транспортного рівня.

USB – Universal Serial Bus. Універсальна послідовна шина.

X.509.v3 – стандарт, що визначає формати даних і процедури розподілу відкритих ключів за допомогою сертифікатів з цифровими підписами, які надаються сертифікаційними органами.

WTLS – Wireless Transport Layer Security. Протокол безпеки бездротового транспортного рівня.

VPN – Virtual Private Network. Віртуальна приватна мережа.

ВСТУП

Актуальність теми дисертації. Захист інформації є важливою складовою комп'ютерних технологій. Реалізація сучасних систем інформаційної безпеки покладається на компоненти безпеки комп'ютерних систем та мереж. Компонентами безпеки вважатимемо апаратні та/або програмні засоби комп'ютерних систем та мереж, що забезпечують необхідний для конкретного застосування рівень захищеності інформації. Вони мають стійкі сфери використання в сучасному інформатизованому суспільстві. Компоненти безпеки реалізовуються за відповідними методами функціонування, алгоритмічно-програмними та схемотехнічними рішеннями. На теперішній час розроблена велика кількість таких рішень. Значний внесок в розвиток засобів безпеки комп'ютерних систем та мереж внесли вчені Львівщини, серед них О.Вербицький [4], В.Дудикевич [10], В. Ємець [13], А. Мельник [13, 29, 35], В.Мельник [29], Г. Микитин [10], Б.Русин [44], В. Яковина [55], Д. Федасюк [55].

При конкретному застосуванні треба враховувати витрати на захист інформації та на очікуваний ефект захищеності, тобто знаходити компроміс між вартістю створення і використання компонентів безпеки та необхідною мірою забезпечення інформаційної безпеки. За рівнем очікуваного ефекту захищеності доцільно класифікувати сфери застосування компонентів безпеки, що полегшить пошук конкретного компромісного рішення. Саме найкращий варіант цього компромісу визначає рівень ефективності компонентів безпеки.

Для сфери захисту інформації слід відзначити важливу особливість, що виділяє її з-поміж інших прикладних напрямків інформаційних технологій: якість та ефективність пропонованих рішень в цій сфері оцінити складно. Усі дослідження стосовно ефективності компонентів безпеки комп'ютерних систем є доцільними, оскільки збагачують базу знань у відповідній сфері та розширюють функціональні можливості при створенні нових засобів безпеки.

На теперішній час дослідженням методів підвищення ефективності компонентів безпеки комп'ютерних систем та мереж з використанням

маскуючих елементів текстових та біометричних даних приділено недостатньо уваги. Відповідно розробка та дослідження методів підвищення ефективності компонентів безпеки комп'ютерних систем та мереж з використанням маскуючих елементів текстових та біометричних даних є актуальним науковим завданням.

Зв'язок роботи з науковими програмами, планами, темами. Дисертаційна робота відповідає науковому напряму кафедри електронних обчислювальних машин Національного університету "Львівська політехніка": "Питання теорії, проектування та реалізації комп'ютерних систем та мереж, а також комп'ютерних засобів, вузлів, приладів і пристроїв вимірювальних, інформаційних, керуючих, телекомунікаційних та кіберфізичних систем" та науково-дослідної роботи "Інтеграція методів і засобів вимірювання, автоматизації, опрацювання та захисту інформації в базисі кібер-фізичних систем", шифр ДБ/КІБЕР, реєстраційний номер 0115U000446, термін виконання першого етапу 01.01.2015–31.12.2015 рік, фінансово підтриманої Міністерством освіти і науки України.

Мета і задачі дослідження. Метою дисертаційної роботи є розробка та дослідження методів підвищення ефективності компонентів безпеки комп'ютерних систем з використанням маскуючих елементів текстових та біометричних даних. Для досягнення поставленої мети необхідне розв'язання наступних задач:

– виконати аналіз сучасного стану розробки та дослідження методів підвищення ефективності компонентів безпеки комп'ютерних систем та мереж, визначити особливості застосування маскуючих елементів текстових та біометричних даних;

– розробити та дослідити математичну модель процесу взаємодії користувача із криптографічною системою захисту та біометричний визначник, що демонструє можливості застосування біометрії у ключовій підсистемі;

– розвинути та дослідити метод та алгоритм автентифікації користувачів в комп'ютерних мережах із використанням біометричних даних за відбитками пальців з використанням маскуючих елементів;

– використовуючи засоби статистичного аналізу дослідити характеристики блокових шифрів, а також можливість використання у компонентах маскуючих елементів для текстових та біометричних даних з метою розширення функціональних можливостей криптосистем;

– дослідити можливість вдосконалення методу шифрування інформації на основі блокових шифрів із використанням маскуючих елементів для текстових та біометричних даних з метою підвищення ефективності компонентів безпеки комп'ютерних систем;

– розробити критерій оцінювання ефективності компонентів безпеки та провести їх тестові дослідження.

Об'єктом досліджень є процес функціонування компонентів безпеки в системах контролю доступу з використанням біометричних даних та в криптографічних системах захисту інформації.

Предметом досліджень є методи та алгоритми підвищення ефективності компонентів безпеки комп'ютерних систем та мереж, які базуються на використанні маскуючих елементів текстових та біометричних даних.

Методи досліджень. Під час розв'язання наукових завдань використано основні положення дискретної математики, теорії алгоритмів, теорії ймовірності теорії статистичного аналізу. Використання математичного моделювання та методів тестування комп'ютерних засобів, що надають можливість ідентифікації параметрів, дозволило оцінити ефективність запропонованих засобів.

Наукова новизна одержаних результатів.

1. Запропоновано модифікований метод автентифікації користувачів в комп'ютерних мережах як подальший розвиток засобів управління доступом, який полягає у використанні маскуючих елементів біометричних даних за відбитками пальців, та у порівнянні із відомими розширює функціональні можливості методів та засобів автентифікації, що дозволяє поліпшити їх

ефективність при використанні за схемою “відкритий ключ користувача – закритий ключ користувача”.

2. Вперше запропоновано вдосконалений метод шифрування інформації в компонентах безпеки комп’ютерних систем, який полягає у статичному використанні маскуючих елементів у відкритому тексті повідомлення з наступним перетворенням інформації блоковими криптографічними засобами, та на відміну від відомих покращує частотний розподіл символів у шифрованому тексті, що дає можливість підвищити ефективність компонентів безпеки.

3. Вперше запропоновано вдосконалений метод шифрування інформації в компонентах безпеки комп’ютерних систем, який полягає у динамічному використанні маскуючих елементів у відкритому тексті повідомлення з наступним перетворенням інформації блоковими криптографічними засобами, який на відміну від відомих покращує частотний розподіл символів у шифрованому тексті та наближує до рівноймовірного, що дає можливість поліпшити ефективність компонентів безпеки.

4. Вперше розроблено та апробовано критерій оцінювання ефективності компонентів безпеки комп’ютерних систем із використанням маскуючих елементів текстових та біометричних даних, яких враховує сукупність важливих показників ефективності, що дозволяє отримати узагальнену оцінку ефективності компонентів безпеки.

Практичне значення одержаних результатів.

1. На основі аналізу сучасного стану компонентів безпеки комп’ютерних систем та мереж визначені основні напрямки покращення їх ефективності з використанням маскуючих елементів текстових та біометричних даних.

2. Використання запропонованого методу автентифікації користувачів у комп’ютерних системах та мережах на основі біометричних даних за відбитками пальців з маскуючими елементами за схемою “відкритий ключ користувача – закритий ключ користувача” розширює функціональні можливості компонентів безпеки.

3. Шифрування інформації на основі статичного чи динамічного використання маскуючих елементів у відкритому тексті повідомлення з наступним перетворенням інформації з допомогою блокової криптографічної системи покращує частотний розподіл символів у шифрованому тексті та ефективність компонентів безпеки.

4. Запропонований критерій оцінювання ефективності компонентів безпеки комп'ютерних систем та мереж на основі блокових шифрів із використанням маскуючих елементів дозволяє отримати узагальнену кількісну оцінку їх ефективності.

5. Основні результати теоретичних досліджень дисертації впроваджено в навчальний процес студентів базового напряму "Комп'ютерна інженерія" Національного університету "Львівська політехніка" у лабораторні практикуми з курсів "Захист інформації в комп'ютерних системах", "Комп'ютерні системи"; при виконанні науково-дослідницького проекту "Удосконалення та розвиток ґрід-кластеру Фізико-механічного інституту ім. Г.В. Карпенка НАН України"; при виконанні науково-дослідницької роботи "Інтеграція методів і засобів вимірювання, автоматизації, опрацювання та захисту інформації в базисі кібер-фізичних систем", шифр ДБ/КІБЕР; при розробці програмного забезпечення компонентів безпеки в міжнародній аутсорсинговій компанії "KindGeek (ТзОВ "КайндГік")".

Особистий внесок здобувача. Положення та результати, що виносяться на захист дисертаційної роботи, отримані здобувачем особисто. Основний зміст роботи, всі теоретичні та практичні розробки, висновки та рекомендації виконані здобувачем особисто. У роботах, опублікованих у співавторстві, здобувачу належать: розроблена структура моделі взаємодії користувача із системою криптографічного захисту [1]; запропоновані особливості застосування біометричних даних із маскуючими елементами в ланках автентифікації ґрід-середовища [2]; розроблені засади удосконалення та використання ґрід-кластеру [3]; запропоновані критерій ефективності та методика його використання [6]; запропонований базовий алгоритм фільтрації зображень [17]; розроблені моделі

підвищення ефективності блокових шифрів [22]; запропоновані статичний та динамічний методи шифрування [23]; запропоновані основні особливості нового способу шифрування інформації [42]; розроблені загальні особливості вставлення маскуючих елементів у біометричні дані [87]; запропонований алгоритм використання біометричних даних у сертифікаті X.509.v3 [89].

Апробація роботи. Основні теоретичні положення та практичні результати дисертаційної роботи доповідалися і обговорювались на семінарах та конференціях: наукових семінарах кафедри "Електронні обчислювальні машини" Національного університету "Львівська політехніка" (2009-2015); Відкритій науково-технічній конференції молодих науковців і спеціалістів Фізико-механічного інституту ім. Г.В. Карпенка Національної академії наук України "Проблеми корозійно-механічного руйнування, інженерія поверхні, діагностичні системи" (м. Львів, 2009); 4-ій Міжнародній науково-технічній конференції ACSN-2009 "Сучасні комп'ютерні системи та мережі: розробка та використання" (м. Львів, 2009); 3-ій міжнародній конференції молодих учених CSE-2009 "Computer Science and Engineering" (м. Львів, 14-16 травня 2009); 4-ій Міжнародній конференції молодих учених CSE-2010 "Computer Science and Engineering" (м. Львів, 25-27 листопада 2010); IV-ій науково-практичній конференції "Електроніка та інформаційні технології ЕлІТ - 2012", ФМІ НАН України (м. Львів – Чинадієво, 30 серпня -2 вересня 2012); V-ій Всеукраїнській школі-семінарі молодих вчених і студентів АСІТ'2015 "Сучасні комп'ютерні інформаційні технології", ТНЕУ (м. Тернопіль, 22-23 травня 2015); першому науковому семінарі "Кіберфізичні системи: досягнення та виклики" (25-26 червня 2015, м. Львів); 5-му міжнародному форумі молодих науковців "Litteris et Artibus" (м. Львів, 26-28 листопада 2015).

Публікації. За результатами досліджень, які викладені в дисертації, опубліковано 17 наукових праць, 9 статей в періодичних наукових виданнях (в тому числі 7 статей у фахових виданнях та 2 статті в іноземних виданнях), 8 тез доповідей на конференціях та семінарах, отримано 1 патент на корисну модель на новий спосіб шифрування інформації.

РОЗДІЛ 1. АНАЛІЗ СУЧАСНОГО СТАНУ ЗАСТОСУВАННЯ КОМПОНЕНТІВ БЕЗПЕКИ КОМПЮТЕРНИХ СИСТЕМ ТА МЕРЕЖ

1.1. Загальні особливості компонентів безпеки комп'ютерних систем та мереж

Масове застосування комп'ютерних систем та мереж обумовлює обмін великими масивами інформації. В багатьох випадках необхідно забезпечувати такі режими обміну інформацією, щоб вона була доступною обмеженому колу користувачів, тобто необхідно постійно забезпечувати безпеку інформації. Реалізація цієї проблеми покладається на компоненти безпеки комп'ютерних систем та мереж.

Компонентами безпеки будемо вважати програмні та/або апаратні засоби комп'ютерних систем та мереж, що забезпечують певний рівень захищеності інформації, необхідний для надійного, стабільного функціонування організації, підприємства, закладу. Безпека інформації є широким поняттям та включає такі складові як захист інформації, автентифікацію, контроль доступу, аудит, виявлення вторгнень [4, 5, 7-10, 13, 29, 30, 41, 48-51, 55, 62-64, 85-86]. Засоби мережевої безпеки використовуються в першу чергу для захисту даних, що передаються через відкриті канали зв'язку, від несанкціонованого доступу.

Під інформаційною безпекою, як правило, розуміють захищеність інформації та телекомунікацій від випадкових та/або навмисних дій природного чи штучного характеру, що можуть завдати неприйнятних збитків суб'єктам інформаційних відносин. Захист інформації – це комплекс заходів, спрямованих на забезпечення інформаційної безпеки [4, 13, 55].

Вимоги до компонентів безпеки встановлюються стратегією (політикою) безпеки конкретної установи, організації, фірми, закладу тощо, тобто сукупністю правил, директив, заходів, засобів, направленою на надійні, стабільні, ефективні опрацювання, зберігання, передачу, захист інформації. Ці компоненти можуть бути програмними, апаратними або апаратно-програмними. Програмні компоненти легше налаштовувати на різні алгоритми

захисту інформації, однак, час реалізації алгоритмів може бути значним. Апаратні компоненти характеризується більшою продуктивністю та простотою використання, разом із тим переналаштування їх на різні алгоритми захисту інформації проблематичне. Змішані рішення – апаратно-програмні компоненти, в багатьох випадках можуть бути найбільш ефективними засобами безпеки.

Основними сервісами захисту інформації вважають: конфіденційність; автентифікацію; цілісність; неможливість відмови; управління доступом; доступність. Ці сервіси мають ряд особливостей [4, 5, 13, 27, 29, 36, 45, 48, 53, 55, 56, 58, 68, 71, 72].

Конфіденційність має забезпечувати захист даних, що передаються через відкриті канали зв’язку, від пасивних атак зловмисників, тобто від їх перехоплення чи моніторингу. Залежно від важливості повідомлення, можна використати один, два або більше рівнів захисту. У найкращому випадку цей сервіс має забезпечувати захист усіх даних, що передаються між користувачами протягом певного часового інтервалу. В гіршому випадку необхідно забезпечувати захист окремого повідомлення або його частини. Вибір типу сервісу обумовлюється техніко-економічними особливостями його застосування.

Ще однією особливістю конфіденційності є захист потоку даних від можливості його аналітичного дослідження, тобто необхідно ускладнити чи унеможливити зловмиснику виявляти джерело та приймач інформації, частоту обміну повідомленнями, розміри повідомлень та їх інші характеристики [55].

Задача сервісу автентифікації – забезпечувати надійну ідентифікацію дійсного джерела інформації. У випадку зовнішньої інтерактивної взаємодії користувача, наприклад при підключенні до мережі, можна виділити два аспекти функціонування цього сервісу. По-перше, в момент підключення засоби автентифікації мають гарантувати, що обидва об’єкти автентичні, тобто дійсно є тими, ким себе представляють. По-друге, при подальшому обміні даними засоби автентифікації не повинні допустити, щоб на потік даних могла вплинути третя сторона через представлення себе “законним” об’єктом обміну

з метою несанкціонованих отримання чи передачі інформації [1, 7, 8, 11-13, 32, 33, 55].

Сервіс цілісності повинен гарантувати, що отримані повідомлення будуть повністю відповідати переданим: не містять змін, доповнень, розташовуються в початковій послідовності, без повторень та спотворень. Одні сервіси можуть мати засоби відновлення цілісності повідомлення, натомість в інших такі засоби можуть бути відсутніми. Так як задача таких сервісів, у першу чергу, протидіяти активним атакам зловмисників чи впливам завад, то найважливішим для них є виявлення порушень цілісності, а не її відновлення. Однак автоматичне відновлення цілісності підвищує ефективність сервісу [1, 7, 8, 13, 32, 33, 55].

Неможливість відмови – недопустимість як для відправника, так і для адресата відмовитися від факту обміну інформацією. Коли повідомлення надіслане, одержувач може переконатися, що це зробив легальний відправник. Коли повідомлення отримане, відправник може переконатися, що воно прийняте легальним адресатом [1, 7, 8, 13, 32, 33, 55].

Сервіс управління доступом забезпечує можливість обмеження та контролю доступу через комунікаційні канали до вузлів мережі та додатків. Для реалізації такого контролю необхідно мати можливість ідентифікувати кожний об’єкт, що здійснює спробу отримати доступ до ресурсів мережі, щоб кожний такий об’єкт міг мати в системі індивідуальні набори повноважень [1, 13, 29, 30, 55].

Сервіс доступності призначений мінімізувати можливість здійснення атак та порушень, що можуть спричинити недоступність ресурсів або значне ускладнення доступу до них. У одних випадках ефективними є автоматизовані засоби такі як автентифікація та шифрування, а в інших для запобігання відмови чи відновлення доступності можуть знадобитися фізичні дії [13, 29, 55, 85, 86].

Кожний із наведених сервісів має в тій чи іншій мірі важливе значення для ефективного захисту інформації. Однак особливе місце в описаному переліку

посідає сервіс автентифікації, який є базовим у процедурах ідентифікації та авторизації.

Задача ідентифікації – кожному користувачеві поставити у відповідність політику доступу до об’єкта, що захищається. Для цього користувач повинен себе ідентифікувати шляхом зазначення власного ідентифікатора, за допомогою якого перевіряється, чи належить користувач, що реєструється, до учасників системи. За введеним ідентифікатором користувачу надаються відповідні права доступу. Автентифікація забезпечує контроль процедури ідентифікації. Для цього користувач вводить секретне слово – пароль. Коректність пароля, що вводиться, підтверджує відповідність між користувачем, що реєструється, та ідентифікованим учасником системи. У загальному випадку можуть ідентифікуватися та автентифікуватися не лише користувачі, але й інші суб’єкти доступу до ресурсів мережі.

У сукупності виконання процедур ідентифікації та автентифікації реалізує процедуру авторизації. Іноді не потрібно ідентифікувати користувача, а достатньо виконати процедуру автентифікації. Процедура авторизації має ключове значення при захисті інформації комп’ютерних систем та мереж, так як основна політика доступу до ресурсів будується відповідно до ідентифікаторів користувачів. Отримавши ідентифікатор одного із користувачів системи, зловмисник отримує права доступу до ресурсу цього користувача [55].

Основні вимоги до процедури авторизації у системах захисту інформації комп’ютерних систем та мереж полягають у наступному [55, 78, 82]:

1. Повинні здійснюватися ідентифікація та перевірка достовірності суб’єктів доступу на вході в систему за ідентифікатором та паролем достатньої довжини.

1. Система захисту повинна вимагати від користувачів ідентифікувати себе при запитах на доступ.

2. Система захисту повинна піддавати перевірці достовірність ідентифікації – здійснювати автентифікацію. Для цього вона повинна мати необхідні засоби та дані для ідентифікації та автентифікації.

3. Система захисту повинна не допускати до захищених ресурсів не ідентифікованих та неавтентифікованих користувачів, повинна володіти здатністю ефективно пов’язувати одержану ідентифікацію з усіма діями цього користувача.

Наведені вимоги не є вичерпними. Наявний ряд загроз, пов’язаних із конкретним застосуванням мережі, з недосконалістю реалізації процедури авторизації в сучасних операційних системах, з наявністю помилок в реалізації відповідних механізмів захисту. Це обумовлює доцільність виконання досліджень механізмів авторизації з метою підвищення їх ефективності.

Механізми ідентифікації та автентифікації є чине найважливішими для унеможливлення несанкціонованого доступу до захищених ресурсів, отже, слід звернути на них особливу увагу. Крім того, використовуючи системний підхід при проектуванні засобів захисту, при розробці механізмів ідентифікації та автентифікації необхідно брати до уваги усі загрози подолання захисту, розглядати засоби авторизації у необхідній та достатній сукупності інших механізмів захисту, що мають у комплексі вирішувати задачу захисту для конкретного застосування.

Найбільш вживані програмно-апаратні засоби ідентифікації та автентифікації за функціональними ознаками можна розділити на чотири класи: електронні; біометричні; комбіновані; разові паролі, які часто вводять до складу електронних (рис.1.1) [55].

В електронних засобах ідентифікаційною ознакою є код, що зберігається у пам’яті ідентифікатора. До сучасних електронних ідентифікаторів належать: контактні смарт-карти; безконтактні смарт-карти; USB-ключі (USB-token); iButton. Функціонально та конструктивно iButton є модулем енергозалежної пам’яті з вбудованим акумулятором із можливістю послідовного запису та читання інформації.

У біометричних засобах ідентифікаційними є біометричні особливості людини, що отримали назву біометричних даних [44]. Ідентифікація здійснюється шляхом порівняння отриманих від користувача біометричних

даних та еталонів, що завчасно записані в пам’ять системи. У залежності від характеристик, що використовуються для ідентифікації, біометричні засоби поділяються на динамічні та статичні. Динамічна біометрія ґрунтується на аналізі дій людини, зокрема таких як динаміка голосу, параметри підпису тощо. Статична біометрія ґрунтується на даних анатомічних особливостей людини, що є незмінними протягом тривалого періоду часу: відбитки пальців, візерунок райдужної оболонки ока.

У комбінованих засобах використовуються одночасно дві класифікаційні ознаки одного чи різних класів. Серед комбінованих засобів можна виокремити: біометричні та USB-ключі; разові паролі та USB-ключі; безконтактні смарт-карти та USB-ключі; гібридні смарт-карти. Серед разових паролів найчастіше використовуються паролі з оновленням за часом та з оновленням від клавіатури [52].

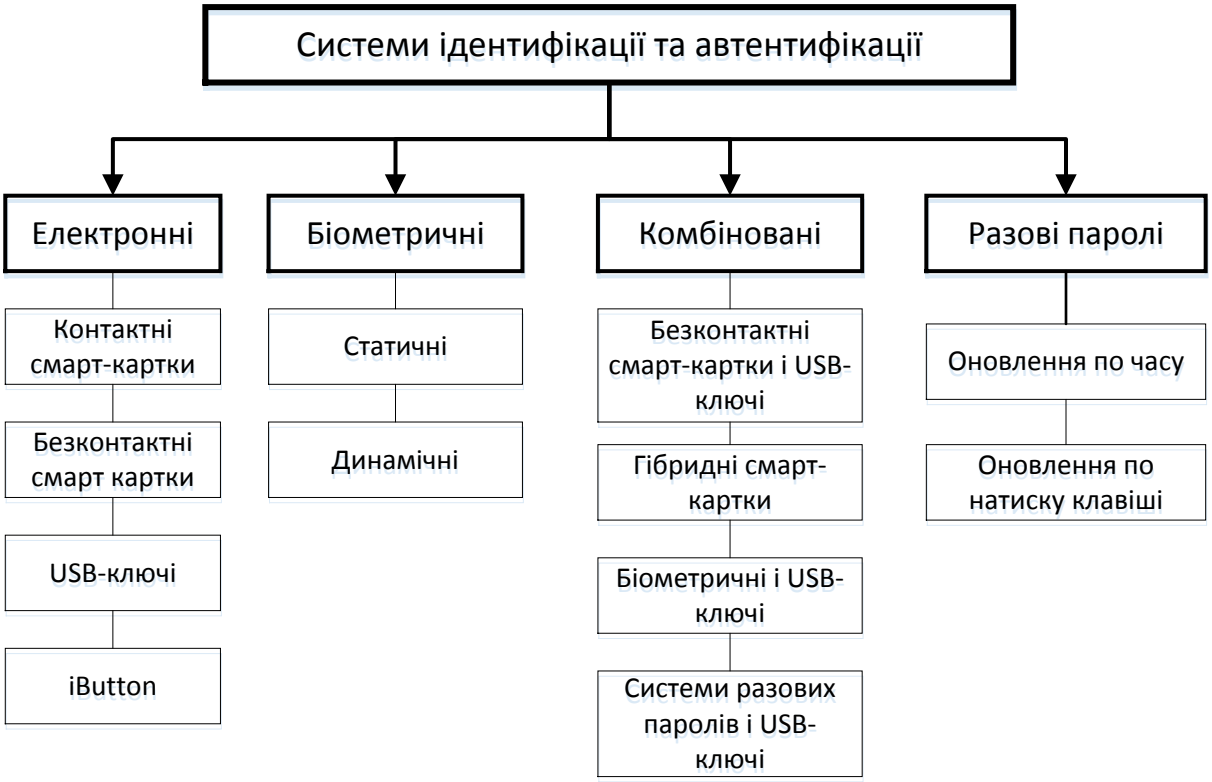

Рис. 1.1. Класифікаційна структура засобів ідентифікації та автентифікації

Механізми авторизації є складовими компонентами практично усіх широковживаних операційних систем (ОС). Окремо розглянемо особливості авторизації у операційних системах класів Windows та UNIX.

Механізм управління доступом Windows можна умовно подати як послідовні чотири етапи: ідентифікація – автентифікація – авторизація – підготовка звітів [55]. В процесі ідентифікації Windows використовується набір даних, який унікально ідентифікує об’єкт безпеки (користувача, групу, комп’ютер, обліковий запис служби) в загальній службі каталогів. Служба каталогів, така як Active Directory (AD), дозволяє надійно ідентифікувати об’єкти. У внутрішніх засобах Windows використовуються ідентифікатор безпеки SID (Security Identifier), глобально унікальні ідентифікатори GUID (Global Unique Identifier) та інші унікальні теги. Автентифікація Windows і відповідні протоколи активізуються кожного разу, коли користувач, комп’ютер або служба реєструються локально або на контролері домену (DC). У Windows суб’єкт безпеки вводить автентифікаційну інформацію на екрані реєстрації. Протокол автентифікації кодують надану інформацію і формують запит автентифікації. Служба автентифікації може використовувати базу даних SAM (Security Account Management) або AD (Active Directory). База даних SAM обслуговує локальні процедури реєстрації і реєстрацію на контролерах домену Windows NT 4.0. Засоби AD використовуються для автентифікації запитів у Windows 2000 або доменах пізніших версій цієї ОС. Протоколи автентифікації LAN Manager (LM), NT LAN Manager (NTLM), NTLMv4, Kerberos використовуються для транспортування запитів автентифікації та подальших трансакцій між екраном реєстрації та службою автентифікації. Якщо служба автентифікації підтверджує комбінацію ідентифікатора та закритих даних автентифікації, тоді авторизація об’єкта безпеки вважатиметься коректною [52].

У складній комп’ютерній мережі, де кожному клієнту для підтримки зв’язку з конкретною службою необхідна наявність індивідуального ключа, і, навпаки, – кожній конкретній службі для своїх клієнтів потрібно зберігати ідентичний ключ, проблема обміну ключами та їх захисту створює суттєвий

ризик усій системі безпеки. Протокол Kerberos пропонує ефективне розв’язання цієї проблеми: у систему безпеки додається учасник, що є посередником між сервером та клієнтом. Роль посередника відведена центру розподілу ключів KDC (Key Distribution Center). KDC є службою, яка розміщується та працює на захищеному фізично сервері та керує базою даних, яка містить інформацію про облікові записи усіх головних абонентів області безпеки. Окрім того, в базі даних зберігається довготривалий криптографічний ключ (відомий виключно конкретному абонентові та службі KDC), який використовується для зв’язку користувача системи безпеки із центром розподілу ключів. Більшість практичних реалізацій протоколу Kerberos характеризується тим, що у них довготривалі ключі генеруються на основі паролю користувача. Для звернення до серверу клієнту необхідно скерувати запит у центр KDC, який надсилає у відповідь кожному конкретному учаснику майбутнього сеансу екземпляр унікального сеансового ключа, дійсного протягом короткого часового інтервалу. Ці ключі призначені для автентифікації сервера та клієнта. Звернімо увагу, що екземпляр сеансового ключа, який надсилається серверу, шифрується із використанням довготривалого ключа цього ж сервера, а екземпляр, що скеровується клієнту, шифрується його довготривалим ключем.

Також використовується інша схема адміністрування паролів, що дозволяє підвищити ефективність протоколу Kerberos. Клієнт, що збирається підключитися до сервера, скеровує запит службі KDC, яка у відповідь скеровує обидва екземпляри сеансового ключа клієнту. Повідомлення, призначене клієнтові, шифрується довготривалим ключем, який належить клієнту, а інформація про клієнта разом із відповідним йому сеансовим ключем для доступу до сервера поміщаються у блок даних, – сеансовий квиток. Потім – сеансовий квиток повністю шифрується довготривалим ключем сервера, який відомий виключно службі KDC та цьому серверу. Після цього уся відповідальність за обробку квитка, який транспортує в своїй структурі шифрований сеансовий ключ, покладається на клієнта, задача якого доставити його на сервер. Одержавши відповідь KDC, клієнт екстраполює з нього

сеансовий квиток та власний екземпляр сеансового ключа, які розміщує в безпечному місці оперативної пам’яті. Як тільки виникає необхідність зв’язатися із сервером, клієнт посилає повідомлення, що складається з квитка, зашифрованого як і раніше із використанням довготривалого ключа сервера та власного автентифікатора, зашифрованого сеансовим ключем. Цей квиток разом із автентифікатором складає посвідчення, за яким ідентифіковується клієнт. Сервер, отримавши посвідчення клієнта, за допомогою свого секретного ключа розшифровує сеансовий квиток і з нього екстраполює сеансовий ключ, який надалі використовуватиме для дешифрування автентифікатора клієнта. Якщо інформація збігається, система приходить до висновку, що посвідчення клієнта видане службою KDC та є коректним [55].

Клієнт може вимагати від сервера взаємної автентифікації. В такому випадку сервер шифрує мітку часу з автентифікатора клієнта за допомогою свого екземпляру сеансового ключа, а далі – пересилає клієнту як власний автентифікатор. Використання сеансового квитка економить часові ресурси сервера та клієнта, так як у нього відпадає необхідність звернень до KDC перед кожним сеансом зв’язку з певним сервером [55].

Для реєстрації із використанням смарт-карти у Windows 2000 та наступних версіях, реалізовано спеціальне розширення підпротоколу Kerberos AS Exchange, який дозволяє використовувати сертифікати відкритих ключів: при реєстрації за допомогою смарт-карт використовується пара з приватного та відкритого ключів, що зберігається в пам’яті смарт-карти. Підпротокол AS Exchange використовує цю пару наступним чином. Відкрита її частина служить для шифрування сеансового ключа користувача службою KDC, а особиста служить для дешифрування цього ключа клієнтом. Коли клієнт вставляє смарт-карту у спеціальний зчитувач та вводить персональний ідентифікаційний номер (PIN), його реєстраційні дані пересилаються до підсистеми розпорядника локальної безпеки LSA (Local Security Authority) ідентично як і при парольній реєстрації. Ця підсистема використовує PIN користувача для доступу до смарт-карти, де зберігається приватний ключ користувача та сертифікат X.509.v3, що

містить відкритий ключ пари. Наступні криптографічні операції з використанням цієї пари здійснюються із застосуванням смарт-карти. Провайдер підтримки безпеки Kerberos клієнтського комп'ютера надсилає службі KDC повідомлення первинного запиту на автентифікацію KRB_AS_REQ. У поле даних попередньої автентифікації цього запиту включається сертифікат відкритого ключа користувача. KDC перевіряє достовірність сертифіката, а потім вилучає з нього відкритий ключ, яким шифрує ключ сеансу реєстрації. Після цього він включає цей ключ разом з квитком на видачу квитків у повідомлення KRB_AS_REQ і скеровує його клієнту. Розшифрувати сеансовий ключ може виключно той клієнт, який має закриту/приватну половину криптографічної пари, функції якої на цьому закінчуються. Вся подальша взаємодія між клієнтом і службою KDC відбувається із використанням сеансового ключа [55].

Механізм управління доступом UNIX реалізований за подібними принципами з суттєвими відмінностями у технології зберігання паролів [55].

Шифрування, приховування й завадостійке кодування – це три види перетворення інформації, які можуть доповнювати один одного, а їх окреме або спільне застосування забезпечує захист інформації, що передається між об'єктами комп'ютерних систем та мереж. Розробка й реалізація методів як поєднання вказаних трьох видів перетворення інформації – перспектива сучасних систем передачі даних [55].

На теперішній час у системах захисту найбільш поширеним є шифрування інформації. Засоби, які забезпечують шифрування та дешифрування інформації, утворюють криптографічну систему (криптосистему).

Сучасні криптографічні системи повинні задовольняти наступним основним вимогам [1, 13-16, 38, 55, 57, 66, 67, 88, 90, 91]:

– вихідний текст із шифрованого тексту може бути відтворений виключно із використанням ключа дешифрування;

– послідовний перебір усіх імовірних ключів дешифрування з метою відновлення вихідного тексту потребує великого часу обчислень або призводить до неприйнятно високих затрат для здійснення таких обчислень;

– інформація про алгоритм, що використовувався для шифрування, не повинна позначитись на стійкості системи шифрування до зламування;

– несуттєві зміни шифрувального ключа повинна призводити до значних змін шифрограми того самого тексту;

– елементи структури алгоритму шифрування не повинні змінюватись;

– додаткові біти, які в процесі шифрування додаються у повідомлення, повинні бути надійно закритими в зашифрованому тексті;

– не повинно існувати простих залежностей між ключами, які послідовно використовуються при шифруванні;

– довільний ключ із множини використовуваних ключів повинен забезпечувати надійність системи шифрування;

– алгоритми шифрування й дешифрування повинні допускати як апаратну, так і програмну реалізацію, при цьому зміни довжин ключів не повинні спричиняти якісне погіршення алгоритмів.

За способами чи методами шифрування-дешифрування інформації розрізняють два типи криптосистем: симетричні та асиметричні [1, 13, 48, 55, 85, 86].

Симетричні криптосистеми (або симетричне шифрування) – спосіб шифрування, при якому для шифрування та дешифрування використовується ідентичний криптографічний ключ. Найважливішими параметрами у переважній більшості алгоритмів симетричного шифрування є [1, 13, 48, 55, 85, 86]:

– стійкість;
– розмірність (довжина) ключа;
– кількість ітерацій (раундів);
– розмірність (довжина) блоку, який обробляється;

– складність програмної/апаратної реалізації;

– складність перетворень.

Перевагами симетричних систем шифрування вважаються:

– порівняно висока швидкість (орієнтовно на 3 порядки вища, ніж у асиметричних систем шифрування);

– простота реалізації, що досягається за рахунок використання більш простих операцій;

– необхідність використання ключів меншої довжини для відповідної стійкості.

Водночас, наявні і суттєві недоліки, які при практичній реалізації призводять до значного скорочення застосування цих систем на теперішній час, серед них [1, 13, 48, 55, 85, 86]:

– складність керування ключами у значних за розмірами мережах, що на практиці означає квадратичне зростання кількості ключів, які потрібно генерувати, знищувати, зберігати та передавати у мережі. У мережі з 10 абонентів необхідно 45 ключів, 100 абонентів – 4950, 1000 користувачів – 499500;

– складність процедури обміну ключами. Застосовуючи симетричну систему шифрування необхідно розв’язати задачу надійної передачі ключів кожному абоненту, а, отже, необхідний секретний канал передачі конкретного ключа кожній зі сторін.

Асиметричні криптосистеми (або асиметричне шифрування, криптосистема з відкритим ключем) – це система шифрування, в якій відкритий (публічний) ключ передається незахищеним (відкритим) каналом зв’язку та використовується виключно для шифрування повідомлень. Для дешифрування повідомлень використовується секретний (приватний) ключ. Як і симетричні, асиметричні криптосистеми мають ряд переваг та недоліків [1, 13, 39, 48, 55, 85, 86].

До переваг систем шифрування з відкритим ключем можна віднести:

– закритий ключ не потрібно передавати каналами зв’язку;

– секретний ключ зберігається лише у однієї зі сторін (на відміну від симетричних криптосистем, де ключ повинен зберігатися у обох зі сторін обміну);

– ключі можна не змінювати достатньо тривалий інтервал часу, тобто відсутня необхідність заміни ключа шифрування після кожного сеансу передачі даних, як у симетричній криптосистемі;

– як правило, кількість ключів в асиметричній криптосистемі менша, ніж у симетричній.

Недоліки асиметричних алгоритмів шифрування полягають у наступному:

– сторони обміну "засвічуються" фактом передачі повідомлень, що може бути використано для побудови атаки;

– асиметричні алгоритми шифрування використовують довші ключі, ніж симетричні;

– як такі асиметричні системи потребують значних обчислювальних ресурсів, тому при практичному застосуванні їх частіше використовують у поєднанні з іншими алгоритмами.

Прикладом симетричного алгоритму шифрування є алгоритм гамування, в основі якого лежить використання випадкових чисел [1, 13, 29, 55, 85]. Нехай є давач псевдовипадкових чисел, який функціонує відповідно до певного визначеного алгоритму. Зазвичай використовують алгоритм:

$$T_{i+1} = (aT_i + b) \bmod c,$$

де T_i – поточне псевдовипадкове число, T_{i+1} – наступне псевдовипадкове число, a, b, c – відомі коефіцієнти-константи. Як правило, обирають $c = 2^n$, де n – розрядність процесора, $a \bmod 4 = 1$, b – непарне. У зазначеному прикладі послідовність псевдовипадкових чисел матиме період c.

Процес шифрування відбувається у наступний спосіб. Спочатку повідомлення, що шифрується, подається у форматі послідовності зі слів S_0, S_1,… (довжина кожного слова – n), які додаються зі словами послідовності псевдовипадкових чисел T_0, T_1, ... за модулем 2, тобто $C_i = S_i \oplus T_i$.

Сукупність чисел послідовності $T_0, T_1, \ldots$ має назву гами шифру.

Процес дешифрування полягає у повторному додаванні за модулем 2 вже шифрованої послідовності з ідентичною процесу шифрування гамою шифру:

$$S_i = C_i \oplus T_i .$$

Початкове значення T_0 називається ключем шифру, який є секретним та повинен бути відомим виключно відправникові та адресатові шифрованого повідомлення. У випадку, коли період послідовності псевдовипадкових чисел достатньо великий (гама шифру довша повідомлення), дешифрувати повідомлення можна виключно підбором ключа. Таким чином, при зростанні n експоненційно зростає криптостійкість шифру.

Описаному методу характерний суттєвий недолік: якщо відомо певну частину вихідного повідомлення, усе повідомлення може бути дешифроване. Нехай відомо одне початкове слово S_i. Тоді $T_i = C_i \oplus S_i$, а далі, – уся права частина гами шифру вираховується за поданою вище формулою давача псевдовипадкових чисел.

Сучасні симетричні шифри використовують складні комбінації значної кількості підстановок та перестановок. Значна частина таких шифрів виконується у декілька ітерацій, використовуючи ключ відповідної ітерації на кожній. Сукупно множина “ключів ітерацій” для усіх ітерацій є розкладом ключів. Зазвичай, розклад ключів формується з первинного ключа шляхом виконання над ним математичних операцій, зокрема, перестановок та підстановок.

1.2. Базові підходи до побудови компонентів безпеки комп’ютерних систем та мереж

Особливу увагу ефективному впровадженню принципів побудови сучасних систем захисту інформації (надалі – СЗІ) приділяють в державних установах та провідних західних фірмах. Світовим лідером з нормативної бази і стандартизації, які регулюють процес створення подібних систем, поза

конкуренцією є США, де існує ціла система законодавчих актів у сфері кібербезпеки. Серед нормативних актів можна виокремити “Закон про контроль за інформаційною безпекою США” (California Senate Bill № 1386, акт Sarbanes-Oxley), численні нормативно-правові акти Національної розвідки США, зокрема ЕО 13636 “Вдосконалення кіберзахисту критичної інфраструктури” (Improving Critical Infrastructure Cybersecurity, №33 від 12.02.2013, том 78), інших без пекових установ і відомств країни, а також рекомендації Національного інституту стандартів і технологій США (NIST). Усі державні та комерційні організації для побудови системи інформаційної безпеки відповідно до перевіреної методології, можуть використовувати положення цих нормативних документів, частина з яких носить обов’язковий характер, а інша – рекомендаційний.

В Україні спостерігається активізація робіт щодо впровадження ефективної нормативної бази і стандартизації в царині інформаційної безпеки, ухвалено базові закони, регулярно розробляються інформаційні звіти [14-16, 31, 43].

Щодо тенденцій, які сьогодні спостерігаються при проектуванні та організації СЗІ у державних установах, то найбільш прогресивні з них починають орієнтуватися на концептуальний підхід до захисту інформації, який базується на впровадженні не окремих засобів захисту, а на системному аналізі проблеми забезпечення інформаційної безпеки. Такий підхід містить попереднє проектування СЗІ, аналіз ризиків і подальше оцінювання ефективності всіх впроваджених заходів.

Побудова ефективних політик і компонентів безпеки, як правило, має базуватися на використанні та впровадженні міжнародних стандартів. Структура системи типових міжнародних стандартів з інформаційної безпеки, що складається із сукупності стандартів ISO/IEC 27001:2013, ISO/IEC 27002:2013, ISO/IEC 15408, які знайшли впровадження в Україні, подана у додатку Б.

1.3. Загальні особливості автентифікації користувачів в Інтернеті

Автентифікація користувачів в Інтернеті відрізняється залежно від послуг та ресурсів, що використовуються, та від типу загроз, що можуть здійснювати зловмисники. Узагальнено загрози можна поділити на два класи: перший – це пасивні атаки, другий – активні. При пасивних атаках зловмисник прагне отримати можливість моніторингу інформації, якою обмінюються користувачі, без зміни повідомлення чи введення своїх даних у інформаційний потік. При активних атаках зловмисник ставить за мету отримати можливість моніторингу та модифікації інформації, якою обмінюються користувачі, а також розкрити технологію та алгоритми автентифікації чи авторизації.

Для різних цілей та загроз можуть використовуватися чотири види автентифікації в Інтернеті [11, 12, 32, 33, 34, 37, 45, 46, 52]:

- автентифікаційні засоби, мінімально захищені від пасивних атак;
- автентифікаційні засоби, мінімально захищені від активних атак;
- автентифікаційні засоби, захищені від пасивних атак;
- автентифікаційні засоби, захищені від активних атак.

Перший вид базується на найпростіших автентифікаційних механізмах, що використовують прості формування та перевірку паролю. Паролем може бути набір символів, що запам’ятав користувач; фізичним радіоелектронним об’єктом, що належить користувачу; унікальною біометричною характеристикою тощо.

Ключ переважно передається через канали зв’язку, а це означає, що він може стати доступним зловмиснику. Отримавши один ключ доступу при наявності дірок в системі безпеки, спрощується задача отримання доступу до усіх паролів певної мережі. Такі автентифікаційні засоби характерні для робочих станцій, де розкриття інформації або її модифікація не є критичними.

Другий вид також використовує парольні засоби, однак, із більш складними формуванням та перевіркою паролів. Автентифікаційні засоби переважно генерують багато складних одноразових паролів із одного таємного ключа. Вони не використовують фізичних об’єктів, що зручно для

автентифікації комп’ютер-комп’ютер. Такі засоби є вразливими для атак перебором у разі невдалого вибору паролю.

Третій вид передбачає використання обчислювальних можливостей комп’ютерів користувачів, які беруть участь в автентифікації. Для користувачів мережі автентифікація може бути одно- або двонаправленою у випадку, коли два користувачі мають ідентифікувати один одного. Окремі системи автентифікації використовують криптографічні алгоритми і формують спільний таємний ключ сесії, що може використовуватися при подальшому обміні даними. Після завершення автентифікації користувачеві може бути наданий автентифікаційний квиток для можливості отримання інших послуг без додаткової автентифікації. Ці засоби автентифікації надають можливість застосування шифрування при передачі даних через незахищені канали зв’язку, підвищуючи рівень конфіденційності.

Четверта група автентифікаційних методів орієнтована на корпоративні мережі, що мають значну потребу в захисті від активних атак. Щоб захиститися від активних атак та забезпечити конфіденційність, необхідно використовувати протокол із шифруванням сеансу, наприклад Kerberos. Можливе використання автентифікаційних механізмів, що захищають від атак відтворення. За протоколом Kerberos користувачі отримують коди доступу до сервера Kerberos і використовують їх для автентифікації з метою подальшого доступу до послуг інших вузлів мережі. Якщо в основі протоколу безпеки лежить синхронізація годинників, то можна використовувати протокол NTPv3, оскільки він передає часові мітки на велику кількість комп’ютерів і є одним із небагатьох відомих протоколів Інтернету, що мають в своїй структурі засоби автентифікації [46, 55].

Управління доступом до ключів є однією із найважливіших задач при забезпеченні автентифікації у великих корпоративних мережах [48, 55]. Протокол Kerberos передбачає використання централізованого сервера ключів. У великих корпоративних мережах потрібна значна кількість таких серверів ключів. Існує необхідність координування сеансових ключів для різних

адміністративних доменів. Більшість алгоритмів шифрування з відкритим ключем потребує значних обчислювальних потужностей, а тому вони недостатньо ефективні для шифрування пакетів у мережі. Однак саме асиметрична особливість забезпечує їх ефективність на початку сеансу при отриманні симетричних сеансових ключів. При асиметричній технології відсутня необхідність використання центрального сервера для зберігання та розсилання ключів. Одним із технічних рішень є використання цифрових підписів для автентифікації відкритих ключів користувачів. При цьому формується сертифікат, який має відкритий ключ партнера. Сертифікати ключів можуть розсилатися різними способами. Для пересилання ключів часто використовуються алгоритми RSA, Діффі-Хелмана та їх модифікації. Іншим варіантом розсилання ключів є використання існуючих служб каталогів, зокрема, допускається розширення можливостей служби доменних імен DNS (Domain Name Service) шляхом включення ключа комп'ютера у ресурсний запис. У групових сеансах управління доступом до ключів ускладнюється, так як кількість обмінів зростає пропорційно збільшенню кількості користувачів.

Одним із основних протоколів автентифікації комутованих з'єднань з Інтернетом є протокол двоточкового з'єднання PPP (Point-to-Point Protokol) [55]. Це складний протокол, що дозволяє програмам віддаленого доступу різних розробників взаємодіяти одна з одною. Він використовує розширення, що стосуються безпеки, такі як протокол перевірки пароля (PAP), протокол автентифікації запит-підтвердження (CHAP), розширюваний PPP-протокол автентифікації (EAP), PPP-протокол управління шифруванням (ECP), PPP-протокол шифрування версії 2 стандарту DES (DESE-bis), PPP-протокол потрійного DES-шифрування (3DESE). Підключення PPP забезпечує зв'язок з віддаленими компонентами мережі через PPP-сервер. Протокол PPP дозволяє серверу віддаленого доступу отримувати запити від програм віддаленого доступу інших розробників, що підтримують протокол PPP, та забезпечувати мережевий доступ до таких програм. При реалізації PPP-з'єднання на початку використовується Протокол управління з'єднанням LCP (Link Control Protokol).

Він визначає функції налаштування та тестування з'єднання. Одним із етапів LCP може бути ідентифікація. PPP регламентує, щоб протокол LCP був використаний для встановлення з'єднання між двома станціями перед тим, як на мережевому рівні відбудеться перший обмін даними. При узгодженні варіантів налаштувань і параметрів використовується протокол управління мережею NCP (Network Control Protokol).

Для великих організацій, компаній робоча група мережі Інтернет (Internet Network Working Group) розробила "Сервіс віддаленої автентифікації користувачів через комутовані лінії" RADIUS (Remote Authentication Dial In User Service) [37, 55]. У цьому сервісі визначені методи створення сервера авторизації разом із центральною базою даних, що використовується для підтвердження користувачів, які з'єднуються через комутовані лінії зв'язку, та усю інформацію, що застосовується при ідентифікації цих користувачів. Сервер за протоколом RADIUS може звертатися до інших серверів, які в своїй роботі можуть використовувати й інші протоколи. У таких випадках RADIUS-сервер використовується як модуль доступу до інших серверів.

Функції протоколу RADIUS в деякій мірі обмежені через його структуру команд та атрибутів, наявні обмеження механізмів пересилки втрачених даних. RADIUS-сервер не посилає клієнту повідомлень за своєю ініціативою. Ці функціональні обмеження в значній мірі усуває протокол DIAMETER. Атрибути повідомлення за протоколом DIAMETER забезпечують обмін даними для ідентифікації, обліку, авторизації. У них містяться дані щодо налаштування даного сеансу зв'язку. Використання протоколу DIAMETER усуває необхідність використання у нових сервісах розроблених раніше різноманітних процедур [46, 55].

Протоколи сімейства TCP/IP є основою побудови елементів глобальної мережі Інтернет, однак, на теперішній час вони не володіють високим рівнем захищеності від активних та пасивних типів атак [53, 55]. Проте в базових протоколах TCP/IP наявний ряд протоколів прикладного рівня, що дозволяють підвищити безпекову ефективність інформаційного обміну між широким

спектром різних прикладних програм та сервісних служб. До таких засобів можна віднести протоколи IPSec (IP-Sekurity), SSL (Secure Sockets Layer – протокол захищених сокетів), PCT (Privace Communication Technology – технологія конфіденційного зв'язку), TLS (Transport Layer Security – протокол безпеки транспортного рівня), WTLS (Wireless Transport Layer Security - протокол безпеки бездротового транспортного рівня), SSH (Secure Shell – захищена оболонка).

IPSec – це набір протоколів, що орієнтовані на розв'язання задач шифрування, автентифікації та забезпечення захисту при обміні IP-пакетами. IPSec забезпечує автентифікацію на рівні комп'ютера та шифрування даних для підключення віртуальних приватних мереж (VPN), що використовують протокол L2TP (Layer 2 Tunneling Protokol). Для побудови VPN використовується також протокол PPTP (Point-to-Point Tunneling Protokol). Обидва протоколи L2TP та PPTP побудовані на основі протоколу PPP [53, 55].

1.4. Принципи використання біометричних даних в компонентах безпеки комп'ютерних систем та мереж

Проблема ідентифікації користувачів в комп'ютерних системах та мережах важливою і актуальною. Традиційні системи управління доступом на основі звичайного кодування мають ряд суттєвих недоліків. Засіб ідентифікації може бути використаний іншою особою, оскільки система ідентифікує сам код, а не конкретного користувача. Код може бути розкритий, загублений або свідомо переданий іншому користувачу. Картка, яка ідентифікує користувача, може бути викраденою, загубленою. Такі випадки можуть привести до несанкціонованому доступу. В біометричних системах контролю доступу суттєво зменшуються описані вище ризики, оскільки для ідентифікації користувача використовуються його унікальні біометричні характеристики, які, як правило, неможливо викрасти, скопіювати чи відтворити [32, 33, 44, 47, 60, 61, 70, 73-77, 80].

Використання біометрії у поєднанні із іншими комп'ютерними технологіями, – один із ефективних варіантів вирішення задачі ідентифікації користувачів комп'ютерних систем та мереж. Біометрія забезпечує можливість ідентифікації людини через її унікальні ознаки: відбитки пальців, розмір та форму обличчя, особливості очей, розмір та форму рук, голос, почерк тощо. Кожна із перелічених характеристик людини має свої особливості, які в значній мірі впливають на інтенсивність їх застосування у засобах безпеки. Біометричні дані, отримані із використанням спеціалізованих пристроїв читання характеристик людини, є базовими для багатьох компонентів безпеки комп'ютерних систем та мереж.

Однією із найбільш широковживаних є технологія біометричної ідентифікації людини із використанням відбитків пальців [44]. Біометричні дані відбитків пальців використовуються у різних сферах, в тому числі у криміналістиці, системах управління доступом, банківській сфері, соціальній сфері, компонентах безпеки комп'ютерних систем та мереж. Основною особливістю біометричних даних відбитків пальців є те, що вони у кожної людини є унікальними та незмінними. Основними елементами поверхні пальця є папілярні лінії та міжпапілярні впадини. Вони утворюють унікальні для кожної людини складні візерунки, що використовуються для ідентифікації людини. Незначна вартість та порівняно невеликі розміри апаратури для опрацювання відбитків пальців, швидкість, якість та точність їх розпізнавання, досконале алгоритмічно-програмне забезпечення, можливість простої адаптації у компонентах безпеки комп'ютерних систем та мереж, – ключові причини, що зумовлюють популярність біометричної ідентифікації та відбитками пальців.

Отримання електронного відбитка пальця з добре помітним папілярним візерунком є основним завданням цього методу біометрії. Для отримання якісного зображення відбитка пальця серійно випускається ряд спеціалізованих пристроїв [32, 44, 47]. На практиці, як правило, використовуються два типи пристроїв читання біометричних даних відбитків пальців – оптичні та електронні (сенсорні).

Оптичні пристрої читання використовують властивість відбивати світло від поверхні пальця з інтенсивністю, що відповідає папілярному візерунку. Основними вузлами є внутрішнє джерело світла, оптичні лінзи та призми, фотоприймачі, підсилювачі та перетворювачі сигналів.

За способом читання візерунку пальця розрізняють оптичні сканери за повним внутрішнім відображенням, оптоволоконні сканери, протяжні сканери та роликові сканери. Одним із найбільш поширених оптичних пристроїв зчитування відбитків пальців є сканери серії BioLink U-Match. Це швидкодіючі, економічні, компактні, ергономічні, надійні та довговічні пристрої, що підключаються до комп'ютера через USB-інтерфейс (Universal Serial Bus). Основні технічні характеристики таких сканерів наведені в таблиці 1.1 [32, 44, 47].

Таблиця 1.1.

Технічні характеристики сканера BioLink U-Match 3.5

Спосіб сканування	Оптичний
Вікно сканування відбитків	25,5 ×18 мм
Роздільна здатність	508 dpi
Час одного сканування	1/15 с
Ймовірність помилкового (чужого) розпізнавання (FAR - False Acceptance Rate)	10^{-9}
Розміри	45 × 63 × 26 мм
Вага	120 г
Інтерфейс обміну інформацією	USB 2.0 / 1.1
Робочий діапазон температур	від -10 ° С до + 55 ° С
Вологість	від 30% до 90%
Енергоспоживання	350 мВт
Середній час напрацювання на відмову	100000 годин або 5 років неперервної експлуатації

Сканер постачається разом із сертифікованими програмними продуктами, що використовуються для вирішення різноманітних задач бізнес характеру, захисту інформації, контролю доступу, обліку робочого часу, масової ідентифікації (біометричні паспорти, фінансові та транспортні послуги). Середня продуктивність роботи такого сканера – не менше 100000 розпізнавань за секунду для сучасних комп'ютерів.

Головною перевагою оптичних пристроїв читання є низька вартість. Недоліків також достатньо, зокрема забруднення, подряпини, жирність поверхні пальців тощо можуть призвести до отримання неякісних біометричних даних, що в свою чергу означатиме неможливість ідентифікації людини.

Електронні пристрої читання використовують властивість змінювати характеристики електричних сигналів, що або відбиваються від поверхні пальця, або проходять через палець відповідно до папілярного візерунку.

Одними із найбільш поширених є ємнісні та ємнісні протяжні сканери. Такі пристрої мають спеціальну пластину, до якої під'єднанні електронні сенсори. Людина прикладає палець до поверхні пластини, утворюючи конденсатор: одна частина – це поверхня пристрою; друга частина – це поверхня пальця. Потенціал електричного поля в конденсаторі залежить від відстані між його частинами та повторює папілярний візерунок пальця. Електричне поле перетворюється у цифрове растрове зображення, що є відображенням біометричних даних конкретного відбитка пальця. Використовуються також чутливі до тиску сканери, термосканери, радіочастотні сканери, ультразвукові сканери. Перевагами електронного способу читання даних є висока відповідність біометричних даних відбитку пальця, що не залежить від стану шкіри, включаючи досить значні забруднення поверхні пальця. Для покращення якості зображень застосовують спеціальні засоби фільтрації [17]. До недоліків таких пристроїв відносять складність технології виготовлення та відповідно високу вартість, а також чутливість до впливу зовнішнього електромагнітного поля [32, 44].

До властивостей папілярного візерунку поверхні пальця людини відносять наступне [44]:

– індивідуальність. Відбиток пальця індивідуальний і відносну стійкість побудови;

– стабільність. Проявляється у збереженні відбитку пальця протягом усього життя людини;

– регенеративність. Усі поверхневі пошкодження тимчасово змінюють візерунок поверхні пальця, який поновлюється разом із оновленням шару шкіри.

Окрім папілярного візерунку, поверхня пальця характеризується множиною мікровпадин, через які проходить виділення поту. Мікровпадини мають розміри в межах 0,08-0,25 мм, а їх форма та взаємне розташування є даними для ідентифікації людини [32, 44]. Кількість унікальних ознак для взаємного розташування мікровпадин є більшою у порівнянні з індивідуальними ознаками папілярних ліній. Однак високі вимоги до чистоти поверхні пальця, складність технічних засобів для читання мікровпадин, складність формалізованого опису їх взаємного розташування та алгоритмів ідентифікації сповільнюють практичне впровадження таких засобів.

Біометричні системами ідентифікації за відбитком пальця відносять до класу AFIS (Automated Fingerprint Identification Biometric System) [44, 59].

Ідентифікація за формою та розміром рук є однією із біометричних технологій, що застосовується у засобах управління доступом. Технологія передбачає попередній етап реєстрації учасників. За інформацією, отриманою за допомогою відеокамери, формується та записується у пам’ять комп’ютера еталонний опис руки кожного учасника. У процесі ідентифікації сканується рука учасника, за отриманою інформацією формується вхідний опис руки, який потім порівнюється із еталонними описами. При співпадінні із визначеною мірою відповідності з еталонним описом, ухвалюється рішення про успішну ідентифікацію. Точність і швидкодія такого способу ідентифікації високі, а описані системи виробляються серійно і застосовуються для ідентифікації

великої кількості людей. Сканування рук учасників забезпечує ймовірність помилки не більше 0,01% [44].

Біометричні системи ідентифікації за формою долоні та відбитками пальців відносять до класу APFIS (Automated Palmprint and Fingerprint Identification Biometric System) [44].

Ідентифікація людини за розміром та формою обличчя переважно застосовується у криміналістиці. В компонентах безпеки такі засоби ідентифікації практично не застосовуються.

Достатньо поширеним є спосіб ідентифікації людини за особливостями сітківки ока. Важливим фактором, що впливає на ефективність цього способу, є стабільні, високоінформативні та чітко виражені ознаки структури сітчастої оболонки ока. Більшість алгоритмів автоматичного розпізнавання сітківки ока за її зображенням реалізовують наступну схему [40]:

- виділення області дослідження на зображенні;
- нормування розмірів виділеного зображення;
- перетворення в певну систему координат, зазвичай, в полярну;
- визначення характерних ознак і формування з них еталонного опису;
- порівняння двох наборів ознак.

Застосування цього способу обмежується тим, що користувач має явно та свідомо співпрацювати із системою, а також порівняно високою вартістю обладнання [32, 44]. Більшість систем розпізнавання використовують підсвічування, переважно в інфрачервоному діапазоні. Спосіб повинен передбачати наявність засобів захисту від підробок (камуфляжів). Однією із задач є компенсація зміни розміру, форми та положення зіниці. Об’єднання даного способу із ідентифікацією людини за розміром та формою обличчя підвищує ефективність ідентифікації.

Голосова ідентифікація характеризується зручністю застосування, водночас основним недоліком цього біометричного способу є невисока точність ідентифікації. В основі способу лежить твердження, що мова кожної людини в певній мірі унікальна характеризується індивідуальними

амплітудними, частотними, фазовими параметрами акустичних коливань. Ці характеристики залежать як від анатомії конкретної людини, так і від її набутих звичок. За отриманими параметрами формують голосові моделі людини і зберігають як еталонні. Прийняте мовне повідомлення перетворюється у електричний сигнал із відповідними характеристиками, далі – у цифровий код, з якого формується записана голосова модель. Ідентифікація передбачає знаходження співпадаючих прийнятої та еталонної голосових моделей з елементарних звуків чи окремих слів [32, 44].

На теперішній час одним із сегментів інформаційних технологій, що має стабільну динаміку росту, є ринок біометричних технологій і відповідного обладнання [32, 44, 47].

Для розв’язання задач інформаційної безпеки комп’ютерних систем та мереж із розглянутих сучасних біометричних технологій найбільш ефективним є застосування засобів, побудованих на скануванні відбитків пальців. Інформація, отримана при цьому, зручна для використання в різних програмних системах, а також придатна для реалізації задач, що вимагають надійної ідентифікації користувачів.

Водночас, огляд літературних джерел засвідчує, що дослідженням стосовно використання маскуючих елементів у біометричних даних у задачах підвищення ефективності компонентів безпеки комп’ютерних систем та мереж приділено недостатньо уваги. Створення нових та вдосконалення відомих біометричних технологій ідентифікації особи за відбитком пальця, в тому числі із використанням маскуючих елементів у біометричних даних, є одним із актуальних та перспективних напрямів побудови ефективних компонентів безпеки комп’ютерних систем та мереж.

1.5. Постановка задачі дослідження щодо використання в компонентах безпеки маскуючих елементів

Постійне розширення сфери застосування комп’ютерних технологій, збільшення обсягів передавання даних через відкриті канали зв’язку,

активізація пасивних та активних атак на комп'ютерні системи та мережі є важливими факторами, що вимагають постійного виконання досліджень щодо підвищення ефективності засобів безпеки.

Аналіз сучасного стану у сфері досліджень підвищення ефективності компонентів безпеки комп'ютерних систем та мереж показав, що застосуванню маскуючих елементів приділено недостатньо уваги. Додавання маскуючих елементів у текстові та біометричні дані – це метод захисту інформації з використанням алгоритмічних та технічних засобів, що ставлять за мету забезпечити криптографічне приховування інформації. Тому актуальними та доцільними є дослідження методів підвищення ефективності компонентів безпеки комп'ютерних систем та мереж на основі маскуючих елементів текстових та біометричних даних.

Ефективність компонентів безпеки комп'ютерних систем та мереж – це складний комплексний багатокритеріальний параметр, який залежить від багатьох факторів, умов використання засобів безпеки, рівня протидії зловмисникам, фінансових і матеріальних можливостей користувача, рівня розвитку математичних, програмних і апаратних засобів протидії, традицій використання засобів захисту і протидії. Цій задачі присвячений аналіз ефективності захисту інформаційних ресурсів [28] та захисту інформації, що базується на криптографічних перетвореннях із використанням маскованого представлення даних [54]. Однак цей аналіз виконано лише в "постановочному" плані – тезисно, відтак він потребує додаткових досліджень. Відповідно, актуальними є дослідження стосовно підходів до оцінювання показників та критерію ефективності компонентів безпеки, в яких використовуються маскуючі елементи текстових та біометричних даних.

1.6. Висновки до розділу 1

1. Аналіз сучасного стану компонентів безпеки комп'ютерних систем та мереж показав, що пасивні та активні атаки зловмисників зумовлюють

проведення неперервних експертних досліджень щодо підвищення ефективності цих компонентів.

2. При проектуванні та реалізації компонентів безпеки комп’ютерних систем та мереж важливо не лише використовувати стійкі ключі, паролі, алгоритми шифрування тощо, – важливо також застосовувати засоби, що приховують використані методи захисту.

3. Використання біометричних даних є одним із ефективних засобів у розв’язанні задач автентифікації користувачів комп’ютерних мереж.

4. Створення нових та вдосконалення відомих біометричних технологій ідентифікації особи за відбитком пальця, зокрема із використанням у біометричних даних маскуючих елементів, є одним із актуальних та перспективних напрямів побудови ефективних компонентів безпеки комп’ютерних систем та мереж.

5. Застосування маскуючих елементів у шифрованих текстових повідомленнях є одними із актуальних підходів підвищення ефективності компонентів безпеки комп’ютерних систем та мереж.

РОЗДІЛ 2. МЕТОДИ ТА МОДЕЛІ ВДОСКОНАЛЕННЯ ЕФЕКТИВНОСТІ КОМПОНЕНТІВ БЕЗПЕКИ НА ОСНОВІ МАСКУЮЧИХ ЕЛЕМЕНТІВ БІОМЕТРИЧНИХ ДАНИХ

2.1. Особливості захисту користувачів в комп'ютерних мережах на основі біометричних даних

Забезпечення захисту інформації від суб'єктів, доступ яким до зазначених інформаційних масивів частково чи повністю обмежено, має багатолітню історію. Особливо актуальною ця проблема стала тепер – в час масового застосування комп'ютерних технологій. Велика кількість вчених та дослідників працює над ефективним розв'язанням цієї багатогранної проблеми. За результатами цих досліджень сформувалися два основні напрями наукових досліджень: криптографія та захист інформації. Багато розроблених методів, алгоритмів, засобів захисту інформації, в першу чергу в комп'ютерних мережах, мають свої переваги та недоліки. Відповідно, дослідження щодо розширення функціональних можливостей та підвищення ефективності захисту інформації в комп'ютерних мережах є актуальними.

Крім основних функцій, до яких відносять конфіденційність, автентифікацію, цілісність, доступність, керування доступом, засоби захисту інформації можуть наділятися додатковими функціями, що випливають із особливостей використання ресурсів комп'ютерної системи чи мережі.

Вважається, що автентифікація та пов'язані з нею питання використання цифрових підписів є найбільш досліджуваною та суперечливою частиною теорії та практики захисту комп'ютерних мереж [55]. Останнім часом активізувалися дослідження щодо застосування біометричних даних у розв'язанні проблеми захисту інформації в комп'ютерних мережах. Біометричні дані – унікальна, вимірна характеристика людини, що може бути використана для ефективної ідентифікації або верифікації за прийнятний час із використанням обчислювальних можливостей комп'ютерів. В роботах [1, 18] запропонована модель взаємодії користувача із системою криптографічного

захисту на основі біометричних даних. В монографії [44] розглянуті основні принципи біометричної автентифікації та криптографічного захисту. В роботах [2, 3, 19, 20, 87, 89] запропоновано використання біометричних даних із маскуючими елементами для захисту інформації в ґрід-системах. Однак наведені результати досліджень не вичерпують можливості застосування біометричних даних для вирішення проблеми захисту інформації в комп'ютерних мережах.

У засобах автентифікації повідомлень та цифрових підписів можна виділити два функціональні рівні. На нижньому рівні повинна виконуватися деяка функція, що генерує автентифікатор – посвідчення, що використовується для автентифікації повідомлення. Результат виконання функції нижнього рівня потім використовується як примітив у протоколі автентифікації вищого рівня, що надає отримувачеві повідомлення можливість перевірки правильності повідомлення [55]. Процес генерування автентифікатора може використовувати функції трьох класів: а) шифрування повідомлень; б) код автентичності повідомлення (Message Authentication Code – MAC); в) хешування – автентифікатором використовується значення, що формується деякою відкритою функцією, яке має фіксовану довжину незалежно від довжини повідомлення.

Основними біометричними даними людини вважають відбитки пальців, будову сітківки очей, тембр голосу. Найбільш розповсюдженим у системах захисту та обмеження доступу є використання відбитків пальців [40]. Саме таким біометричним даним доцільно приділити найбільше уваги стосовно застосування у них маскуючих елементів [1, 11, 18, 21], адже використання відбитків пальців дозволяє отримати інформацію про всі десять пальців рук кожного користувача і застосовувати візерунок будь-якого із них для вирішення задач захисту мереж.

Використання біометричних даних в окремих випадках більш ефективне у порівнянні з такими засобами як паролі, PIN-коди, смарт-карти, а також інші технічні пристрої, оскільки біометрія дозволяє ідентифікувати людину, а не

пристрій. В основі біометричної ідентифікації за відбитком пальця лежить унікальність для кожної людини папілярних візерунків пальців. Зображення відбитку пальця, отримане із використанням спеціального сканера, перетворюється в цифровий код (згортку), а далі – обробляється комп'ютером за необхідними алгоритмами. Таким чином, зображення перетворюються в математичну модель, в якій усі унікальні ознаки (дуги, завитки, петлі і відстань між ними) зберігаються у вигляді цифрового коду. Заключним етапом авторизації є порівняння цієї моделі з шаблонами в базі даних і пошук відповідностей, зокрема серед раніше збережених шаблонів (еталонів) або наборів шаблонів (у випадку ідентифікації). Серійно виробляється цілий ряд недорогих спеціалізованих сканерів, які здатні забезпечити ефективне зчитування відбитків пальців. Обчислювальні потужності сучасних комп'ютерів забезпечують обробку біометричних даних за складними алгоритмами за час, прийнятний для засобів захисту інформації. Відповідно можна застосувати адаптивний метод автентифікації користувача в комп'ютерній мережі. В сервісах автентифікації важливим етапом є формування та використання зашифрованих індивідуальних ключів. Виконання цього етапу доцільно адаптувати до особливостей захисту інформації кожним користувачем мережі. Біометрична інформація про всі десять пальців рук кожного користувача надає можливість адаптувати реалізацію індивідуальних ключів до конкретних особливостей мережі. Адаптацію можна реалізувати використанням різних ключів, рознесених в часі за наперед визначеним розкладом, узгодженим між учасниками повідомлень. Використання різних ключів можна реалізувати за формулою [21]:

$$A = (C1 \cdot C2) \bmod a,$$

де A – номер індивідуального ключа із врахуванням відбитку пальця, *C1, C2* - ідентифікаційні числа, a – довільне число від 2-х до 10. Ідентифікаційні числа *C1, C2* можуть бути як випадковими, так і нести наперед визначене інформаційне навантаження. Найбільші функціональні можливості при $a = 10$ (використанні усього набору пальців рук).

При реалізації адаптивного методу доцільно використати модель та алгоритм взаємодії між користувачем та засобами криптографічного захисту [18, 21, 87].

Загальні положення розробки моделі взаємодії користувача з системою криптографічного захисту ґрунтуються на методах, описаних далі. Так, сучасні методи розв’язання задач захисту даних не можливо уявити без використання криптографічних перетворень – шифрів. Одним із основних понять більшості шифрів є поняття таємного ключа – порівняно невеликої кількості інформації, яка необхідна для здійснення криптографічних перетворень усієї інформації.

Довжина ключа, як правило, визначає стійкість (в обчислювальному сенсі) криптографічної системи захисту. Але така стійкість дає лише теоретичну оцінку надійності захисту. На жаль, стійкість відомих криптосистем визначається, насамперед, стійкістю найслабшої ланки у всій архітектурі криптографічного захисту, а не стійкістю стандартизованого криптоалгоритму чи параметрами використаного ключа. Тому слабкими місцями більшості криптосистем є недоліки у реалізації взаємодії користувача із системою. У випадку проникнення у систему із використанням місць взаємодії користувача із системою, таку ситуацію називають атакою з врахуванням “людського фактору” (соціальна інженерія) [55].

Максимальний рівень надійності кожного шифру досягається виключно за умови повної секретності, випадковості та відповідної довжини використовуваних ключів. Ключі зберігаються на електронний носій інформації та, як правило, захищаються додатковими методами та засобами захисту. Популярними є методи захисту, що ґрунтуються на використанні в якості паролю достатньо короткої фрази, що є простою для запам’ятовування. Уся система захисту працюватиме у такий спосіб: користувач надає пароль, далі за допомогою паролю користувача вивільняється ключ безпосереднього шифрування конфіденційних даних. Згідно із криптографічним законом про те, що надійність системи захисту прирівнюється до надійності її найслабшої ланки, отримуємо: цінні конфіденційні дані, що захищаються надійним

криптографічним алгоритмом, є захищеними лише в мірі безпечності паролю. Тобто, короткі паролі здатні забезпечити низький рівень захисту, довгі – високий рівень захисту, водночас при використанні довгих паролів виникає складність із їх запам’ятовуванням. Водночас, більш значущою є проблема відсутності взаємозв’язку між паролем доступу до закритої системи і його власником, тобто будь-кого, хто надає системі пароль користувача, система ідентифікуватиме саме як визначеного користувача [55].

Акценти у процесах управління ключами при розробці і проектуванні криптографічних систем захисту поступово змістилися до використання особистих ознак користувача системи, які характеризуються такими властивостями:

– індивідуальність або неповторність;
– стабільність впродовж тривалого періоду;
– неможливість фальсифікації;
– неможливість використання декількома користувачами;
– неможливість втратити – забути, загубити або вкрасти.

Одним із підходів щодо підвищення ефективності застосування особистих ознак користувача системи є використання маскуючих елементів у біометричних даних компонентів безпеки [1, 18, 21].

2.2. Метод застосування маскуючих елементів у біометричних даних компонентів безпеки

У роботах [2, 3, 19, 21, 87, 89] розглянуто особливості інтеграції біометричних даних із маскуючими елементами в сертифікати x.509 для ґрід-систем. Достатня перспективність цієї області взаємодії технологій біометричної ідентифікації та ґрід-обчислень обумовлена тим, що X.509.v3 сертифікати використовуються для шифрування та підписування документів. Зазначимо, що сучасні технології спрямовані на суттєве ускладнення підробки сертифікату, але, окрім біометрії, відсутні інші дієві механізми, які можуть

гарантувати, що сертифікат належить людині, яка стверджує, що саме вона – власник цього сертифікату.

Запропонований метод біометричного захисту закритих/приватних ключів оснований на розширеній схемі “нечітке сховище” [1, 21, 87]. Опишемо загальні особливості моделі та алгоритму взаємодії між користувачем та засобами криптографічного захисту. Спочатку необхідно створити деякий криптографічний примітив, який зв’яже закриті/приватні ключі користувача мережі з фрагментами даних відбитків пальців. Необхідно створити певний масив даних, які в певний спосіб блокуватимуть закриті ключі. Такий процес формування необхідного масиву даних відбувається під час реєстрації нового користувача, або коли існуючий користувач змінює ключ .

Запропонований алгоритм створює з N вхідних наборів частинок біометричних даних, блокуючи множину

$$W = \{w_0, ..., w_s\} \subset F$$

закритого ключа шифрування

$$M = \{m_0, ..., m_{k-1}\} \subset F,$$

який записується у вигляді коефіцієнтів полінома $f(x)$ степеню $k-1$ у полі

$$F : f(x) = m_1 + m_2 x + ... + m_k x^{k-1}.$$

Підвищення криптографічної стійкості алгоритму блокування можна досягнути додаванням до множини W ряду маскуючих елементів (фіксованих фрагментів). Тоді загальна кількість частинок дорівнюватиме r. Доведено, що r визначається як мінімальне можливе значення відстані L між фрагментами, яка строго більша ніж $2\sigma_s$, де σ_s – середня порогова відстань, яка залежить від технології отримання наборів фрагментів (технічні характеристики сканера, властивості методу обробки зображень тощо). Чим менша L, тим більше значення r, і, як наслідок, більш стійкий криптографічний захист. Експериментально встановлено, що зі збільшенням r зростає ймовірність помилкового декодування – помилкового відсотку браку (FRR) [65]. Число і

розташування фіктивних фрагментів обмежені дійсним розміщенням особистих часток і стандартним відхиленням розміщенням цих фрагментів.

Доцільно визначити параметри алгоритму k, s, r. Коефіцієнти полінома $f(x)$ є елементами скінченого (обмеженого) поля $F = GF(n)$. Оскільки у запропонованому методі ми розв'язуємо поліноміальну інтерполяційну задачу, то для схеми "нечітке сховище" ефективними є коди корекції помилок Ріда-Соломона (RSC). Для виконання алгоритму декодування коду слід використовувати поле з $n = g^z$ елементів, де g – просте число. Одночасно поле для алгоритму блокування являє собою набір пікселів зображення відбитку пальця. Сучасні сканери мають формат кадру не менше 256 × 256 пікселів. Ефективна область сканування – це ділянка, на яку припадає основна частина зображення ($\approx 98\%$). Цей висновок дає можливість визначити просте число. Найближче значення до цих цифр має область площею 251 x 251 пікселів, яка формує поле $F = GF(251^2)$.

Довжина полінома – значення k, визначається довжиною закритого (приватного) ключа блокування. Кожен з елементів поля $GF(251^2)$ має розмірність 16 біт. Отже, 256-бітному закритому ключу відповідає довжина у 16 поліноміальних коефіцієнтів, або поліном 15 степеня.

Для перетворення координат розміщення реальних і маскуючих (фіктивних) фрагментів у полях елементів доцільно використовувати 16-розрядні цілі числа x_i, в яких молодші 8 біт відповідають ординатам, а старші біти – абсцисам.

Вихідним результатом алгоритму блокування буде набір кортежів B_P, який складається з s пар $\{w_i, f(x_i)\}$ і $r - s$ пар фіктивних фрагментів $\{\alpha_i, \beta_i\}$ з F, які задовольняють умові $f(\alpha_i) \neq \beta_i$. Щоб відкрити систему, яка використовує B_P, зловмисник повинен виявити набір точок, що належать многочленові $f(x)$, тобто виявити закритий (приватний) ключ. Зрозуміло, що чим більше значення r, тим більшою буде кількість подібних до

$f(x)$ недостовірних (фальшивих) многочленів, а, отже, зростатиме стійкість системи до злому. Для зареєстрованого (легального) користувача системи потрібно і достатньо представити принаймні $\tau \geq k$ дійсних точок, щоб успішно інтерполювати поліном. Для алгоритму розблокування формується набір $W' \subset F$, в якому міститься тільки частина елементів множини W. Таким чином різниця двох наборів фрагментів дорівнює $W \setminus W' = t$.

Щоб розблокувати ключ шифрування з B_P, користувач надає набір особистих фрагментів, утворюючи відкриваючу множину $W' = \{w'_1, ..., w'_r\}$. Коли користувач запитує у системи закритий ключ відбувається розблокування.

Через W' і B_P видобувається множина B_P' найближчих (із граничною відстанню σ_s) фрагмент з потужністю r, де $r \approx s$ для зареєстрованого (легального) користувача, та $r >> s$ – для незареєстрованого (нелегального) користувача. Найменше допустиме значення $\tau = \frac{s+k}{2}$.

Результат роботи процесу виділення фрагментів показано на рисунку 2.1. Для зазначеного рисунку кількість знаків $r = 47$, довжина кодового слова – $s = 45$, співпадіння фрагментів $\tau = 45$.

Рис. 2.1. Блокуюча множина o суміщена із розблоковуючою множиною □. Дійсні частинки виділені (радіус $\sigma_s = 6$).

Використовуючи аналогічний метод можна оцінити ефективність алгоритму розблокування в залежності від кількості дійсних фрагментів.

Необхідно враховувати, що зі зростанням значення r, збільшується складність процесу розблокування для незареєстрованого (нелегального) користувача, водночас, це також збільшує складність для зареєстрованого користувача.

k є ще одним важливим параметром, який впливає на ефективність алгоритму. Дослідження показали складність атаки як функцію з аргументами k і кількістю реальних (дійсних) точок. Відповідно значення r повинно бути компромісним.

Засоби автентифікації користувачів на основі біометричних даних доцільно використовувати як спеціальне “біометричне” розширення сертифікатів X.509.v3 [87, 89]. Такі засоби є ефективними для ґрід-систем та розподілених комп’ютерних мереж. При делегуванні прав користувачам у ґрід-середовищі виконуватимуться такі дії: після взаємоідентифікації користувача і служби, яка працюватиме від імені користувача, служба створює нову пару ключів і надсилає відкритий ключ користувачу для підписання; користувач реєструється закритим (приватним) ключем аналогічним до ключа у центрі сертифікації. Отримані проксі-сертифікат і новозгенерований тимчасовий ключ можуть бути використані для служб автентифікації авторизованого користувача в усіх вузлах комп’ютерної системи чи мережі.

2.3. Особливості захисту криптографічних ключів з використанням математичної моделі визначника випадкових величин

Для моделювання процесу захисту криптографічних ключів класичними парольними методами використовується математична модель визначника випадкових величин [1, 22], яка описує процеси квазівипадкових бітів, тобто бітових послідовностей, які жодним поліноміальним алгоритмом неможливо відрізнити від рівномірно розподілених послідовностей такої ж довжини. При

реалізації визначників переважно використовують важкооборотні співвідношення та формувачі псевдовипадкових бітів.

Визначник випадкової величини A можна подати через імовірнісне співвідношення, що відображає її на $\{0,1\}^n$ з мінімальною безумовною ентропією $H_\infty(A)$, при довжині паростка $U_{c\in U}[0,1]^c$ у випадкову величину $V(A,U_c)$, що є близькою до $U_{k\in U}\{0,1\}^k$ на $\delta>0$ на як завгодно малу величину: $V:\{0,1\}^n\times\{0,1\}^c\to\{0,1\}^k$. При цьому: мінімальна безумовна ентропія $H_\infty(A)\le m$; довжина паростка c як випадкової величини рівномірного розподілу $c=\log(n-m)+2\log(c/\delta)$; довжина виходу $k=m+c-2\log(c/\delta)$; втрата ентропії визначника $\Delta H(V(A))=m+c-k=2\log(c/\delta)$ [1].

Ефективність використання біометричних даних у засобах захисту ключів можна покращити наступним чином. Пропонується розширити модель визначника випадкових величин до нової моделі біометричного визначника, яка, на відміну від моделі визначника випадкових величин, дозволить зв'язати криптографічні ключі з нечіткими нерівномірно розподіленими біометричними даними і, тим самим, змоделювати безпосередньо взаємодію користувача із компонентами захисту [1].

Біометричний визначник можна подати через дві залежності: імовірнісну генеруючи залежність FG та відновлювальну залежність FW. Залежність FG відображає випадкову величину $A\in F^n$ із безумовною ентропією $H_\infty(A)\ge m$ та паросток $U_{c\in U}F^c$ у таємну стрічку $S\in F^k$ та відкриту стрічку $Z\in F^*$, які для обох значень стрічок $(S,Z)\leftarrow FG(A,U_c)$ задовольняють умові для статичної відстані D між розподілами ймовірностей випадкових величин $D(S,U_c)\le\delta$. Залежність FG можна записати наступним чином:

$$FG:F^n\times F^c\to F^k\times F^*.$$

Залежність FW відображає випадкову величину $A\in F^n$ та відкриту стрічку $Z\in F^*$ у таємну стрічку $S'\in F^k$, що для усіх величин A, $A'\in F^n$ із

метрикою p у векторному просторі над F величиною $p(A, A') \le t$ та стрічками $(S, Z) \leftarrow FG(A, U_c)$ задовольняє умові $S' = S$. Залежність FW можна записати наступним чином:

$$FW : F^n \times F^* \to F^k .$$

Для біометричного визначника величина залишкової безумовної ентропії обчислюється за співвідношенням [1]:

$$\tilde{H}_\infty(A \setminus Z) = k \ge m + c - 2\log(k/\delta) .$$

Біометричний визначник враховує таку характеристику біометричних даних як їх стабільність. Він забезпечує можливість подати біометричні дані через один чи більше ключів, вибраних випадковим чином. Біометричний визначник зводиться до "чіткого" у випадку, коли $t = 0$, $Z = U_c$. Для врахування нечіткості біометричних даних введено біометричний ідентифікатор [1]. Він забезпечує можливість структуризувати вхідні біометричні дані та забезпечити їх нечутливість до визначеного рівня нечіткості.

Біометричний ідентифікатор можна подати через дві залежності: імовірнісну ідентифікуючу залежність FI та корегуючу залежність FK.

Залежність FI відображає випадкову величину $A \in F^n$ із безумовною ентропією ентропією $H_\infty(A) \ge m$ та паросток $U_{c \in U} F^c$ у метрику у векторному просторі $p \in F^*$, – біометричний ідентифікатор, що відповідає умові для залишкової безумовної ентропії $\tilde{H}_\infty(A \mid p) \ge m'$.

Залежність FI можна представити наступним чином:

$$FI : F^n \times F^c \to F^* .$$

У випадку, коли відома метрика p, величина втрати безумовної ентропії біометричного ідентифікатора обчислюється за формулою:

$$\Delta H(FI) = m - m' .$$

Залежність FK відображає випадкову величину $A \in F^n$ та метрику p у таку $A' \in F^n$, що для усіх метрик $p \leftarrow FI(A, U_c)$ та $p(A, A') \le t$ виконується умова $A' = A$. Залежність FK можна представити наступним чином:

$$FK : F^n \times F^* \to F^n .$$

Наведені залежності дозволяють зробити висновок, що біометричний визначник – це складена конструкція, побудована із біометричного ідентифікатора та чіткого визначника. Цей висновок підтверджується наступними діями.

Імовірнісну ґенеруючу залежність подамо як $FG(A,(U_{c1},U_{c2}))$; розраховуємо $p \leftarrow FI(A,U_{c1})$ та $S \leftarrow V(A,U_{c2})$; виводимо (S,Z), де $Z \leftarrow (p,U_{c2})$.

Відновлювальну залежність подамо як $F(W)(A'(p,U_{c2}))$; відновлюємо $A \leftarrow FK(A,p)$; виводимо $S \leftarrow V(A,U_{c2})$.

Імовірнісна ґенеруюча залежність отримує на вхід випадкову величину *A* деякого нерівномірного розподілу та два паростки (U_{c1},U_{c2}). Імовірнісна ідентифікуюча залежність FI формує ідентифікатор *p* вхідної величини *A*, а чіткий визначник видає таємну стрічку *S*, яка застосовується у криптографічних алгоритмах як ключ або пароль. Для цієї стрічки формується відповідна відкрита величина *Z*, яка буде використовуватися надалі відновлювальною функцією $F(W)$ для відновлення таємної стрічки δ.

Імовірнісна ґенеруюча залежність буде використовуватися лише один раз для формування таємної стрічки *S* та її ідентифікатора *Z*. Відновлювальна функція $F(W)$ буде використовуватися постійно для отримання випадкових величин A', які є близькими до вхідних даних *A*, таємної стрічки *S*. Саме міра близькості A' до *A* визначатиме алгоритми, які реалізують біометричний визначник [1].

2.4. Алгоритм захисту біометричної інформації в ланках автентифікації користувачів у ґрід-середовищі

Сучасні інформаційні технології надають користувачам значні можливості щодо збільшення обчислювальних ресурсів. Однією з таких технологій є ґрід – інфраструктури. Ґрід-система – це тип паралельно розподіленої системи, яка дозволяє розділення, вибір та накопичення “автономних” географічно розподілених ресурсів в реальному часі в залежності від їх придатності, можливості роботи, ціни та відповідності вимогам якості обслуговування користувачів. Технологія ґрід (GRID) використовується для створення географічно розподіленої інфраструктури, яка об’єднує ресурси різного типу з колективним доступом до цих ресурсів в рамках віртуальних організацій (ВО). Учасники ВО взаємодіють між собою з допомогою комп’ютерних мереж (як правило Інтернету) таким чином, що їхні обчислювальні ресурси об’єднуються. Система охоплює всі обчислювальні ресурси і ресурси зберігання даних, але кожна організація контролює використання своїх ресурсів. Користувачі одержують необхідні великі ресурси для обчислень і зберігання даних. Кожен з учасників ВО надає свої обчислювальні ресурси або їх частину для використання іншими учасниками і, одночасно, отримує доступ до ресурсів інших учасників ВО [3, 19, 20].

Питання безпеки при такій розподіленій та просторово рознесеній організації обчислювальних процесів виходять на перший план. При цьому необхідно розв’язати задачі забезпечення автентифікації, контролю доступу, конфіденційності та цілісності даних.

Біометричні дані є унікальним засобом для розв’язання таких важливих завдань як автентифікація та контроль доступу у ґрід – середовищі. Зазначимо, що саме автентифікація є однією із основ політик безпеки, яка дозволяє в межах ґрід-мережі інтегрувати різноманітні локальні політики безпеки. В епоху цифрового представлення інформації біометричні методи щораз більше убезпечують від крадіжки особистих даних, оскільки біометричні технології використовують унікальні, вимірні біологічні та поведінкові риси фізичних осіб

для комп’ютеризованого створення або підтвердження їх особистостей. Поєднання біометричних даних з механізмами контролю доступу у ґрід-системах є головним завданням для наукової спільноти при розробці комплексних рішень на усіх етапах – від збору і аналізу даних до безпосередньо проектування системи.

Якщо у ґрід-системі використовуються методи забезпечення безпеки, користувачам і службам необхідно підтвердити свою ідентичність у спосіб надання особистого мандату. Ґрід-мандати містять дані, що генеруються криптографічними методами, а тому повинні формуватися із використанням спеціалізованого програмного забезпечення. Засоби ґрід-безпеки визначені загальним форматом мандату відповідно до сертифікату X.509 [87, 89]. Сертифікат X.509 в поєднанні із закритим ключем формують унікальну множину мандатів, яку користувач ґрід-системи використовує для своєї автентифікації перед іншими мережевими об’єктами: послугами або користувачами. Кожен сертифікат X.509 видається так званою “достовірною” стороною, який називається центром сертифікації (CA). Центр сертифікації – це, як правило, окрема велика організація або комерційна компанія. Для того, щоб довіряти сертифікату X.509, який був наданий суб’єктом, кожен повинен довіряти центру сертифікації, що видав сертифікат.

Стандартизація в галузі сертифікатів, що засвідчують особу, виданих фізичним особам, зокрема, сертифікатів, основна мета яких полягає у визначенні людини з високим рівнем точності, має важливе значення в ряді аспектів, які виходять за рамки Інтернет-інфраструктури відкритих ключів X.509. Найбільш важливим аспектом, який впливає на обсяг цієї специфікації є визначення інформаційної структури для зберігання біометричної інформації.

Стандарт ECMA.219 [84], на який орієнтоване використання біометричних даних із маскуючими елементами, забезпечує саме таку категорію автентифікації інформації (AI) з незмінних особистих характеристик (наприклад зразок голосу, підпис, відбитки пальців і сітківки ока). AI можна розглядати як структуру із двома незалежними інформаційними складовими:

складова достовірної інформації автентифікації (VAI) та складова інформації автентифікації запиту (RAI). RAI – це те, що запитуваний об'єкт (зазвичай користувач) пред’являє при запиті на автентифікацію. VAI – це те, з чим порівнюється запит. У випадку застосування біометрії складник RAI повинен бути оброблений ще до перевірки VAI.

Ініціалізація АІ зазвичай відбувається в процесі реєстрації користувача. Об’єкт, який виконує реєстрацію користувача повинен бути довіреним органом, який може перевірити інформацію відповідно до політики безпеки системи. Об’єктам, які використовують цю інформацію, необхідно перевірити, чи інформація була підтверджена довіреним органом, а також чи інформація не була змінена будь-яким іншим джерелом. Для цього можуть застосовуватися цифрові підписи. Таку можливість можна надати системі додаючи підписані хеш-дані в сертифікат відкритого ключа.

Запропонований алгоритм застосування біометричної інформації з маскуючими елементами в ланках автентифікації ґрід-середовища має наступні особливості [19] та пояснюється рисунком 2.2.

Основними задачами ґрід – протоколу інфраструктури безпеки можна вважати:

– автентифікацію користувачів в умовах однократної реєстрації;
– комунікаційний захист;
– підтримку обмеженого делегування повноважень віртуальної організації.

Разова реєстрація дає користувачеві можливість лише один раз пройти процедуру автентифікації і на основі власного сертифікату, завіреного надійним центром сертифікації, створити спеціальний мандат безпеки – проксі-сертифікат, який пред’являється програмою будь-якій віддаленій службі на будь-якому вузлі ВО для автентифікації від імені власника (делегування повноважень).

Рис.2.2 Узагальнена схема алгоритму автентифікації із застосуванням біометричної інформації з маскуючими елементами.

Ґрід-протокол інфраструктури безпеки для ідентифікації користувачів середовища використовує X.509 та відповідний секретний ключ, які утворюють унікальний мандат безпеки для конкретного користувача ґріду.

Стандарт ECMA.219, який описує всі особливості методів та засобів автентифікації та контролю доступу, для задач ідентифікації людини в системах з високими вимогами до безпеки, окрім стандартних (паролі та токени), виділяє окрему категорію автентифікаційної інформації – постійні характеристики суб'єкта автентифікаційного процесу [84]. До таких постійних характеристик стандарт відносить наступні біометричні автентифікатори (з відповідними ідентифікаторами в рамках стандарту):

- voicePrint ::= {id-am 8}
- signature :: = {id-am 9}
- fingerprint ::= {id-am 10}
- retinaPattern ::= {id-am 11}
- toeSmell ::= {id-am 99}.

Дослідження в галузі систем безпеки розподілених систем вказують на необхідність широкого впровадження в ланки контролю доступу подібних систем засобів біометричної ідентифікації.

Структура сертифікату X.509.v3 пропонує механізм доповнень, що дозволяє зареєструвати у створюваному сертифікаті необхідну біометричну інформацію. Кожне таке доповнення можна подати у вигляді кортежу [89]:

$$F = \{P, R, Z\}, \text{ де}$$

– P – тип використовуваного доповнення;

– R – прапорець критичності, який вказує чи інформація, яка подана в даному доповненні, повинна бути опрацьована;

– Z – безпосередньо дані значення, які подаються за допомогою цього доповнення.

Пропонується використовувати цю особливість сертифікатів для застосування біометричної інформації в ланках автентифікації користувачів у ґрід-середовищах [89].

У роботі [89] запропонований алгоритм захисту ключа $M = \{m_o, \ldots, m_{k-1}\} \in F$ за допомогою постійно змінних наборів даних $W = \{ w_1, \ldots, w_2 \} \in F$. У

алгоритмі секретний ключ, який необхідно захистити, перетворюється у коефіцієнти поліному *f(x)* степені *k-1* над полем *F*:

$$f(x) = m_1 + m_1x + ... + m_kx^{k-1}.$$

Набір даних *W* використовується для захисту (блокування) *f(x)*, при цьому $s > k$. На виході алгоритму блокування отримується набір кортежів B_p, який складається з *s* пар $\{w_i, f(w_i)\}$ та $r - s$ пар фіктивних точок $\{\alpha_i, \beta_i\}$ із *F*, які задовольняють умовам $f(\alpha_i) \neq \beta_i$. Схема алгоритму захисту наведена на рис.2.2.

Для розкриття системи, яка використовує B_p, атакуючій стороні необхідно виділити з такого набору точки, що лежать на поліномі *f(x)*, тим самим відновити секретний ключ. Очевидно, що чим більшим є r, тим більшою є кількість подібних до *f(x)* фіктивних поліномів, а отже, і більшою є стійкість системи до розкриття.

Легальному користувачеві системи необхідно і достатньо пред'явити щонайменше $\tau \geq k$ істинних точок, щоб успішно інтерполювати прихований поліном. Тобто для розблокування надається набір $W \in F$, який містить лише частину елементів *W*, тобто потужність різниці двох множин особистих ознак $W \setminus W' = t$.

Якщо у якості множини блокування та розблокування використовуються, наприклад, дактилоскопічні дані людини, то набором блокування *W* у цьому випадку є набір координат пікселів, які відповідають місцеположенню ознак на відбитку пальця, тобто $F = GF(n)$, де $n = g^2$, а g – просте число.

Отриманий набір B_p пропонується вносити у спеціальне "біометричне" доповнення сертифікату X.509.v3.

Внаслідок внесених змін ідентифікація у ґрід-середовищі функціонуватиме наступним чином: після взаємної автентифікації користувача та сервісу, який далі працюватиме від імені користувача, сервіс створює нову ключову пару та відправляє відкритий ключ на підпис до користувача; користувач підписує цей ключ подібно до того, як це робить центр сертифікації, використовуючи при цьому заблокований у B_p таємний ключ. Одержані проксі-сертифікат та згенерований тимчасовий таємний ключ можуть використовуватися сервісом

для автентифікації від імені користувача на усіх дозволених даному користувачеві вузлах віртуальної організації [89].

2.5. Особливості багатопараметричного застосування біометричних даних

2.5.1. Загальні підходи до багатопараметричного застосування біометричних даних

Основними, найбільш поширеними біометричними даними людини для виконання автентифікації вважають відбитки пальців. Такі дані становлять майже 60% загальної частки застосувань у засобах автентифікації [44]. Цей показник досягається за рахунок біологічної повторюваності відбитку пальця на рівні 10^{-5}%, невеликого розміру та відносно невисокої вартості сканерів відбитків, а також доволі простої та швидкої процедури сканування відбитку. Використовуються й інші параметри, такі як риси обличчя (14%), тембр голосу (8%), біометрія рогівки ока (7%), геометрія руки (6%), біометрія роботи з клавіатурою (3%), інші менш вживані ознаки [40]. Кожний параметр має свої особливості і забезпечує відповідну ефективність автентифікації.

Очевидно, що поєднання двох і більше параметрів автентифікації значно підвищує ефективність захисту, дещо ускладнюючи процедуру автентифікації і збільшуючи її час. Тому при визначенні композиції автентифікаційної процедури необхідно визначити такі параметри як вартість, складність, часові затрати, співставивши їх із отриманими показниками ефективності системи захисту. Якщо стійкість до злому системи одного використаного параметра складає $P_{i,}$ то параметр стійкості системи захисту буде $P_c = P_i$. При використанні декількох параметрів захисту, параметр стійкості системи захисту буде:

$$P_c = 1 - \{(1 - P_{i1}) \cdot (1 - P_{i2}) \cdot (1 - P_{i3}) \cdot ...\}.$$

Перспективно використовувати комбінацію двох різних за природою параметрів, наприклад біометрії відбитків пальців і біометрії рогівки ока, біометрії відбитків пальців і тембру голосу. Ефективно підвищується стійкість

системи захисту при використанні біометрії двох різних пальців, які визначені за індивідуальним графіком системи захисту. Окрім переліченого, варто звернути увагу на використання маскуючих елементів, наприклад фіктивних частинок в біометрії відбитків пальців. Як було показано на прикладі біометрії відбитків пальців використання маскуючих елементів значно ускладнює зловмиснику задачу взлому системи захисту, одночасно підвищуючи функціональні можливості компонентів безпеки системи автентифікації комп'ютерних систем.

Через недосконаліть апаратних засобів читання біометричних даних, неоптимальність алгоритмічно-програмних засобів їх опрацювання, вплив завад тощо можуть виникати помилки автентифікації. Ці помилки мають ймовірнісний характер. Як правило, такі помилки є базовими поняттями математичної статистики для перевірки статичних гіпотез. Одним із типів помилок називають хибну тривогу або хибне спрацювання. FAR (False Acceptance Rate) – імовірність хибного збігу біометричних властивостей двох людей. FRR (False Rejection Rate) – імовірність відмови в авторизації людині, яка має доступ. Кращою вважається та система, у якої значення FAR та FRR менші.

Експертне оцінювання двопараметричної автентифікації користувачів за біометрією рогівки ока та біометрією відбитка пальця з використанням маскуючих елементів, за тембром голосу та біометрією відбитків пальців з використанням маскуючих елементів показало, що значення FRR може бути не більшим за 0,001%, а значення FRR може бути не більшим за 0,05%. Такі величини FAR та FRR є конкурентоздатними на ринку застосування компонентів безпеки комп'ютерних систем та мереж.

2.5.2. Підвищення ефективності криптографічних систем з захистом на основі використання ланок біометричного блокування (розблокування) ключів

Основною ідеєю криптографічних систем з біометричним захистом є створення ланок біометричного блокування (розблокування) ключів подібно до ланок парольного захисту ключів [44].

У роботах [1, 19] запропоновано алгоритм зв'язування ключа у кореляційній системі розпізнавання відбитків пальців. Алгоритм "інтегрує" у процесі реєстрації користувача в системі криптографічний ключ *K* у функцію кореляції. Використовуючи так звані тренувальні відбитки пальців з маскуючими елементами утворюється кореляційна функція $H(u) = |H(u)| \cdot e^{-i\varphi H(u)}$. Далі відкидається модуль *H(u),* добуток фази та випадкового комплексного числа утворює нове значення фази для величин, що записується на сервері, $H_{stored}(u)$ (відповідає *h).* Модуль $|H_{stored}(u)|$ утворюється з деякого випадково вибраного ключа *K.* Крім H_{stored} на сервері записується хеш ключ. Повна структура даних, яка відповідає конкретному користувачу, називається *Bioscrypt.* У процесі ідентифікації користувач надає свої відбитки пальців (мінімум п'ять раз), які корелюються з допомогою функції кореляції. Якщо застосовувати процедуру відновлення, яка використовує коди корекції помилок, виділяється ключ, який хешується та порівнюється з хешем, який записаний у *Bioscrypt.* Результатом відновлення є або реальний ключ, або відмова у декодуванні.

Отже, застосування маскуючих елементів мінімізує втрати ентропії криптографічного ключа, який задається даним методом, збільшує коректуючі властивості методу, розширює функціональні можливості компонентів безпеки комп'ютерних систем та мереж.

Досліджено особливості застосування маскуючих елементів в біометричних даних за рогівкою ока. В роботах [32, 44] запропоновано алгоритм з використанням біометрії рогівки ока та перетворення відображення

рисунка рогівки у 2048 – бітну двійкову послідовність (*IrisCode*). У процесі ідентифікації розраховувалась віддаль Хемінга між наданою користувачем та збереженою у базі даних біометрією. Показана як відмінність між різними взірцями однієї рогівки може досягати 10% (204 біт), а відмінність між рогівками різних людей може досягати 45% (922 біт). У процесі реєстрації користувача в системі отримується декілька зображень рогівки ока та до зображень вводяться маскуючі елементи. Для кожного зображення генерується відповідний K бітний *IrisCode.* Із отриманих кодів за допомогою мажоритарного декодера утворюється "канонічний" *IrisCode* **T** довжини *K*. Далі вибирають (*N, K*) – код корекції помилок довжиною *N*, завадостійкі властивості якого дозволяють коректувати до 10% помилок. Кодове слово *C,* що відповідає послідовності **T**, хешують, підписують цифровим підписом системи та записують у базі даних разом із $R = N - K$ перевірочними бітами, доданими до **T** у процесі кодування. У процесі ідентифікації користувач надає *IrisCode* **T'**, до якого додається *R* і утворена двійкова послідовність вважається спотвореним кодовим словом. Застосовуючи функцію декодування, отримують кодове слово *C',* яке хешується , підписується цифровим підписом. Результат порівнюється з даними, які збережені у базі. Вважається, що *IrisCode* може використовуватися як криптографічний ключ, а для збільшення ентропії пропонується використання додаткового символьного пароля. Алгоритм є дуже швидким та надійним настільки, наскільки надійною є використовувана хеш-функція. Застосування в алгоритмі маскуючих елементів у біометричних даних користувача забезпечує можливість поставити у відповідність декілька ключів.

Досліджено особливості застосування маскуючих елементів в біометричних даних динаміки роботи з клавіатурою. У праці [33] пропонується об'єднати пароль користувача з біометрією динаміки роботи з клавіатурою. Ця робота є продовженням досліджень, в яких пропонується метод рандомізації паролів перед хешуванням [1]. Під рандомізацією, у даному випадку, розуміється приєднання до пароля (*psw*) випадкової послідовності довжиною *s* – біт, у результаті утворюється так званий "зміцнений" пароль (*hpsw*). У

процесі реєстрації користувача системою зберігається така інформація: випадкове число *r* довжиною *k* біт; “таблиця інструкцій” зашифрована за допомогою *psw’*. Таблиця інструкцій містить інформацію про процес генерації з *r* та *psw* значень *hpsw*. Процес генерації є толерантним до певної кількості помилок (параметр системи): “файл історії”, зашифрований за допомогою *hpsw*.

У процесі ідентифікації користувач надає *psw’*. Під час набору пароля отримується біометрія. Об’єднуючись обидві величини утворюють *hpsw’,* яке використовується для дешифрування історії. У разі невдачі система розшифровує таблицю інструкцій за допомогою наданого пароля, та за допомогою схеми розподілу таємниці Шаміра генерує інше значення *hpsw’,* яке знову використовується для спроби дешифрування. Процес повторюється *n* раз, де *n* – параметр безпеки алгоритму. Використовувана біометрія лише на 15 біт збільшує ентропію пароля, що лише несуттєво збільшує ефективність звичайної парольної системи ідентифікації.

У працях [1, 18] запропоновано алгоритм захисних функцій для збільшення конфіденційності. У процесі реєстрації користувачем обирається довільний криптографічний ключ *S*, який у вигляді хешу *V* записують у базі даних. Далі знаходять таке значення *W,* яке задовольняє умові $G(W, X) = S$, де *X* – біометрія користувача, *G* – функція “δ – зв’язування”, функція, яка для усіх *X’,* що знаходяться у δ – околі значення *X,* видає *S*. Будь-яка детермінована функція володіє властивістю 0 – зв’язування, ∞ - зв’язуванням володіє функція $G(W, X) = \text{constant}$. Величина *W* записується у базі даних. У процесі ідентифікації користувачем надається біометрія *X’,* а сервером ідентифікації величина *W,* які подаються у $G(W, X)$. Отримане значення *S’* хешується та порівнюється із *V*. На основі результату приймається рішення про ідентифікацію. Робота функції δ – зв’язування – ґрунтується на використанні завадостійкого кодування.

У працях [1, 18, 21] у схемі “нечіткого зв’язування” продовжені дослідження для збільшення коректуючих властивостей алгоритму. У процесі реєстрації користувач вибирає секретний ключ *C*, який одночасно є кодовим

словом БЧХ – коду. Нехай *d* – це відстань між *C* та Z– біометричною двійковою послідовністю із використанням маскуючих елементів. Структура “нечіткого зв’язування” містить *d* та хеш ключа *C.* У процесі ідентифікації користувач надає біометрію Z**’**. З допомогою *d* знаходять найближче кодове слово *C’*, яке хешується та порівнюється з записаним на сервері. Перевагами цього алгоритму є його простота реалізації та можливість використання будь-якого коду корекції помилок. Серед таких кодів можна виділити алгоритм Ріда-Соломона.

Якщо порівняти наведені алгоритми, то можна зробити такі висновки. В процесі реєстрації біометричною системою зберігається не сам біометричний сигнал *w*, а його відображення разом із маскуючими елементами $h(w,K)$, де *K* – це криптографічний ключ, який захищається системою. Трансформація $h(w,K)$ – це в певному сенсі аналог криптографічної функції, тобто для різних входів *w* отримуються різні виходи, а отримання *w* або *K* із $h(w,K)$ є важкою проблемою.

У праці [32] результат трансформації $h(w,K)$ носить назву “таємний шаблон”, а в роботі [44] – “скасовувана біометрія”. Загалом в технічній літературі за процесом трансформації закріпився термін “зв’язування ключа”. Згідно з такою постановкою задачі, у процесі ідентифікації відбувається трансформація вхідного біометричного сигналу користувача *w’* та за допомогою функції $h(w,K)$, а процес порівняння здійснюється у просторі відображень.

У різних системах ідентифікації, які використовують саме такий метод, необхідно використовувати різні перетворення, або розглянуте вище перетворення $h(w,K)$, але з різними параметрами. Якщо у будь-якій із систем скомпроментовано $h(w,K)$, то інші системи, які використовували ті ж біометричні дані, але з іншими ключами, і надалі функціонуватимуть без внесення змін.

У випадку використання необоротних функцій (хеш-функції, односторонні перетворення), стійкість $h(w,K)$ є доведено високою, але FRR такої системи є великою. Причина – відмінність у послідовності зчитування біометричних даних. Очевидно, що для різних біометричних даних *w, w* відображення *h (w')* та *h (w)* теж будуть різними.

За умови використання оборотних функцій (шифрування з ключем) – FRR системи на рівні звичайної біометричної системи ідентифікації, але безпека біометричних даних є пропорційною до стійкості оборотної функції, тобто знову виникають проблеми пов'язані з управління ключами. Враховуючи вищевказані проблеми, конкретна конструкція трансформації $h(w,K)$ повинна враховувати такі особливості:

– нечіткість біометричних відображень: відмінності у наданні біометричних даних, відмінності у послідовних зчитуваннях біометричних даних, особливості апаратури та алгоритмів отримання біометричних даних;

– стабільність біометричних ознак: неможливість зміни або обмежена кількість змін біометричних даних після компрометації, застосування одних біометричних даних у кількох системах захисту;

– вразливість до атак з використанням "троянських коней".

Врахування перелічених особливостей, застосування маскуючих елементів у біометричних даних забезпечує ефективне управління ключами, розширює функціональні можливості компонентів безпеки комп'ютерних систем та мереж.

2.5.3. Оцінювання ефективності вставлення маскуючих елементів у даних за біометрією голосу

Досліджено особливості криптографічної системи із використанням біометрії голосу та ефективність застосування маскуючих елементів у відповідних біометричних даних. Голос, мова людини, за фізичною основою є акустичними коливаннями із складними амплітудно частотно-фазовими характеристиками. У кожної людини ці характеристики є унікальними.

Авторизація чи автентифікація за голосом характеризується простотою апаратних засобів: для використання таких процедур не потрібна додаткова вартісна апаратура, – достатньо підключеного до комп'ютера пристрою для перетворення акустичних коливань в електричні сигнали з відповідними характеристиками – мікрофону. Однак, значно складнішими є алгоритмічно-програмні засоби, що забезпечують зазначених процес автентифікації [32, 44].

Для опрацювання голосових даних попередньо необхідно записати звуковий файл в оперативну пам'ять комп'ютера, або на інший носій. Більшість сучасних пристроїв обладнані необхідним для введення та виведення мови обладнанням: мікрофоном та звуковою картою. На теперішній час серійно випускається велика кількість високоякісних мікрофонів. З виходу мікрофону сигнал подається на вхід звукової карти пристрою. Звукова карта забезпечує аналого-цифрове перетворення електричного сигналу з широким спектром можливостей налаштувань параметрів перетворення, основними з яких є частота дискретизації та розрядність кодування. Ці параметри необхідно обирати за певними правилами, оскільки вони визначають якість біометричних даних, яку отримуємо в результаті опрацювання запису [44, 47].

Параметри аналого-цифрового перетворення обираються за властивостями людської мови. Усереднена спектральна щільність потужності має максимум в діапазоні 250-500 Гц і затухає з швидкістю 8-10 дБ на октаву. Це призводить до того, що на частотах вище 4000 Гц спектральна щільність падає до рівня -60 дБ, що відповідає зменшенню потужності в 20 і більше разів у порівнянні з максимумом. Це дозволяє вважати, що частотний спектр для каналів передачі мови може бути обмежений частотою 5кГц. Відповідно частота дискретизації сигналу повинна складати не менше 10 кГц. У сучасних звукових картах використовується імпульсно-кодова модуляція, при якій кожний дискретний відлік голосового повідомлення кодується за описаними правилами.

Існуючі системи розпізнавання голосу базуються на зборі усієї доступної, іноді навіть надлишкової інформації, необхідної для розпізнавання користувача. Першим кроком є зменшення об'єму інформації, щоб її можна було

проаналізувати і обробити апаратно-програмними засобами. Наступним етапом є спектральне представлення звукового сигналу із використанням швидкого перетворення Фур'є, яке дозволяє стиснути інформацію та отримати важливі характеристики голосового повідомлення. Отримані характеристики можуть бути доволі різноманітними, що зумовлене багатьма факторами, серед яких: відмінності людських голосів, рівень гучності мовлення, варіювання у вимові та рухах артикулярів (язика, губ, піднебіння, щелепи). Відповідно ускладнюється процес розпізнавання.

Для покращення процесу розпізнавання мови існує багато методів та алгоритмів. Одним з найбільш вдалих алгоритмів отримав назву динамічні спотворювання (dynamic time warping) [44, 47]. Техніка динамічного спотворення використовується для частотного випрямлення і зменшення відстані між спотвореним спектральним представленням і шаблоном користувача. Використання цього методу дало покращення точності розпізнавання приблизно на 20-30%.

Метод динамічного спотворення використовують практично усі доступні комерційні системи авторизації за голосом. Спочатку сигнал перетворюється в спектральне представлення, де визначаються деякі основні та важливі параметри. Потім визначаються кінцеві вхідні параметри для варіації голосу і здійснюється нормалізація для складання шкали параметрів та визначення ситуаційного рівня мовлення. Описані вище параметри в подальшому використовуються для створення шаблону за яким відбувається порівняння із отриманим голосовим повідомленням.

Реалізація засобів авторизації за тембром голосу доцільна за клієнт-серверною архітектурою. При цьому можна виділити 4 частини системи: серверна частина, база даних, адміністраторська та клієнтська частини. Взаємодія між сервером та користувацькою і адміністраторською частиною відбувається через протокол http. Підключення та взаємодія з базою даних відбувається серверною частиною. Основні функції серверної частини: перевірка з'єднання з клієнтом, опрацювання запитів від користувача, налаштування

роботи сервера, запуск сервера. Обслуговування та управління базою даних виконується сервером MySql. Користувацька частина виконує функції авторизації, зміни контрольних фраз, закінчення сесії, зміни паролю та контрольних фраз, опрацювання запиту на отримання інформації про власні сесії. Система управління адміністраторською частиною виконує запити на реєстрацію користувачів, авторизації, керування налаштуваннями користувачів та їх сесіями, запити на обмеження доступу користувачів.

Очевидно, що у системі авторизації користувачів комп'ютерної мережі можливі 2 типи сутностей, які взаємодіють з нею: звичайний користувач та адміністратор. Розглянемо концептуальну модель такої системи, яка зображена на рисунку 2.3.

Рис. 2.3 Концептуальна модель системи авторизації користувачів комп'ютерної мережі за голосом

Система авторизації користувачів реалізована на взаємодії сервера, адміністратора, користувачів. Таблиця 2.1 пояснює взаємодію сутності “Адміністратор” з системою відповідно до призначення.

Таблиця 2.1.

Особливості взаємодії адміністратор-система

№	Адміністратор	Потік повідомлень	Система
1.	Запит на логування	Логін, пароль, контрольна фраза	Перевірка даних
2.	Запит на реєстрацію нового користувача	Логін, пароль, ПІБ, контрольні фрази, фото	Перевірка даних
3.	Запит на керуванням обліковим записом адміністратора	Список налаштувань адміністратора	Оновлення та перевірка даних
4.	Запит на отримання списку зареєстрованих користвачів	Список користувачів	Оновлення та перевірка даних
5.	Запит на керування налаштуваннями користувачів	Інформація про налаштування користувачів	Оновлення та перевірка даних
6.	Запит на керування сесіями	Інформація про періоди перебування в системі	Оновлення та перевірка даних
7.	Запит на обмеження доступу користувачів до системи	Список всіх заблокованих користувачів	Оновлення та перевірка даних

Таблиця 2.2 показує взаємодію системи з сутністю “Адміністратор” відповідно до функцій системи.

Таблиця 2.2.

Особливості взаємодії система-адміністратор

№	Система	Потік повідомлень	Адміністратор
1.	Підтвердження входу або відмова	Інформаційні дані	Вхід або відмова
2.	Підтвердження реєстрації або відмова	Інформаційні дані	Реєстрація або відмова
3.	Список редагування даних	Інформаційні дані	Відображення даних
4.	Детальний список зареєстрованих користувачів	Інформаційні дані	Відображення користувачів
5.	Детальний список налаштувань користувачів	Інформаційні дані	Відображення налаштувань
6.	Детальний список усіх сесій	Інформаційні дані	Відображення сесій
7.	Детальний список усіх обмежень доступу	Інформаційні дані	Відображення заблокованих користувачів

Таблиця 2.3 відображає взаємодію сутності “Користувач” з системою відповідно до функцій системи.

Таблиця 2.3.

Особливості взаємодії користувач-система

№	Користувач	Потік повідомлень	Система
1.	Запит на логування	Email, пароль, контрольна фраза	Перевірка даних
2.	Запит на зміну пароля	Новий пароль	Перевірка аккаунту
3.	Запит на зміну контрольних фраз	Нові контрольні фрази	Запис нових контрольних фраз
4.	Запит на отримання списку здійснених користувачем сесій	Інформаційні дані	Пошук необхідної інформації
5.	Запит на відновлення доступу	Інформація	Перевірка інформації
6.	Запит на вихід із системи	Вихід	Завершення сесії

Таблиця 2.4. відображає взаємодію системи з сутністю “Користувач” відповідно до функцій системи.

Таблиця 2.4.

Особливості взаємодії система-користувач

	Система	Потік повідомлень	Користувач
1.	Підтвердження або відмова	Інформаційні дані	Логування або відмова
2.	Підтвердження або відмова	Інформаційні дані	Зміна пароля
3.	Підтвердження або відмова	Інформаційні дані	Зміна контрольних фраз
4.	Список усіх сесій	Інформаційні дані	Відображення всіх сесій користувача
5.	Підтвердження або відмова	Інформаційні дані	Відновлення доступу
6.	Закінчення сесії	Інформаційні дані	Незалогований користувач

Система повинна комплектуватися необхідною базою даних. Розглянута система може реалізовувати як одно- так і двопараметричну автентифікацію користувачів за біометричними даними.

Експертне оцінювання ефективності дозволяє зробити висновок, що застосування маскуючих елементів у біометричних даних за голосом суттєво поступається використанню маскуючих елементів у біометричних даних за відбитками пальців. В окремих випадках біометричні дані за голосом можуть бути використані другим параметром при двопараметричній автентифікації та в засобах управління доступом.

2.6. Висновки до розділу 2

1. Здійснений аналіз особливостей захисту інформації в комп’ютерних мережах продемонстрував доцільність застосування біометричних даних в сервісах автентифікації користувачів.

2. Запропоновано модифікований метод автентифікації користувачів в комп’ютерних мережах як подальший розвиток засобів управління доступом, який полягає у використанні маскуючих елементів біометричних даних за відбитками пальців – фіктивних фрагментів, що дозволяє поліпшити їх ефективність при використанні за схемою “відкритий ключ користувача – закритий ключ користувача”.

3. Запропонований модифікований метод автентифікації користувачів в комп’ютерних мережах у порівнянні із відомими розширює функціональні можливості методів та засобів автентифікації.

4. Запропонований алгоритм автентифікації користувачів за біометричними даними з використанням маскуючих елементів дозволяє розширити функціональні можливості компонентів безпеки комп’ютерних систем та мереж

5. Показано ефективність використання біометричних даних з маскуючими елементами для процедур автентифікації в ґрід-системах.

6. Обґрунтовано доцільність введення алгоритмів застосування біометричних даних у відповідні стандарти із сертифікації індивідуальних паролів у системах захисту інформації та запропоновано використання механізму біометричного розширення сертифікатів X.509.v3.

7. Показано, що математична модель процесу взаємодії користувача із криптографічною системою захисту із використанням біометричного визначника дозволяє враховувати особливості біометричних ознак користувача.

8. Обґрунтовано використання двопараметричної автентифікації із застосуванням біометричних даних, що покращує ефективність автентифікації користувачів, сприяє зменшенню величини помилкових рішень.

РОЗДІЛ 3. КРИПТОГРАФІЧНІ МЕТОДИ ЗАХИСТУ ІНФОРМАЦІЇ В КОМП'ЮТЕРНИХ СИСТЕМАХ ІЗ ВИКОРИСТАННЯМ МАСКУЮЧИХ ЕЛЕМЕНТІВ

3.1. Особливості використання маскуючих елементів в блокових шифрах для підвищення ефективності захисту інформації

Ефективність та надійність шифрів необхідно розглядати крізь призму часу. Існує велика кількість цікавих шифрів, розроблених в минулі століття, які тривалий час вважали неефективними через складність і трудоємність виконання арифметичних перетворень, невисоку продуктивність роботи криптографа тощо. На сучасному етапі розвитку криптографії варто враховувати, що шифрування інформації і дешифрування виконується з використанням обчислювальної техніки, що дозволяє актуалізувати використання шифрів. Необхідно також брати до уваги, що зловмисник завжди може отримати шифрований текст, також використовуючи засоби обчислювальної техніки.

Ефективність криптосистеми (алгоритм шифрування та дешифрування, або шифр) в значній мірі визначається трудоємністю і часом, який витрачається на шифрування та дешифрування тексту. Надійність криптосистеми визначається часом, який злодій витратить для розкриття алгоритму шифрування і дешифрування, а також щоб знайти ключ шифру. Очевидно, що ефективність і надійність забезпечити одночасно складно – ідеальних шифрів не існує. Необхідно врахувати, що конкретні ситуації висувають свої вимоги до криптосистем. Наприклад, біржова інформація перестає бути таємною через пару десятків хвилин, але повинна бути зашифрована і передана за лічені секунди. Іноді інформація повинна зберігатися десятиліттями, проте відсутні вимоги до швидкості її шифрування [22].

Значна кількість методів злому шифрів основана на класичних дослідженнях шифрованих текстів: частотному аналізі використання одного символу; частотному аналізі біграм; частотному аналізі триграм і

характеристиках повторень трьох, чотирьох і більше символів тексту. Основні криптоаналізи базуються на ґрунтовних дослідженнях популярних шифрів. В багатьох країнах світу на початку 21 століття були сформовані кібервійська (кібердивізії), які використовуються як для захисту власних інформаційних систем, так і для злому систем опонентів із складними сучасними шифрами.

Окремо варто закцентувати на використанні нових методів злому таких шифрів кібердивізіями: якщо раніше складно було здійснити злом шифру методом перебору декількох мільярдів можливих варіантів ключів, то сучасні складні шифри зламуються кібервійськами за допомогою масованих атак в організованій структурі, яка використовує тисячі чи десятки тисяч одночасно працюючих комп'ютерів у синхронізованій розподіленій системі пошуку ймовірних ключів. Тому сучасні комп'ютерні шифри, які стають проблемою для окремого криптоаналітика, що ставить за мету зламати шифр, не складають великих проблем для кібердивізій – стоїть питання лише в часовому і інших ресурсах.

В теорії можна створити абсолютно надійні шифри, які дійсно існують. Водночас, стоїть питання зручності їх використання, продуктивності та собівартості таких шифрів. Як стверджував К. Шеннон [52, 83], для створення таких шифрів краще забезпечити приховування власне факту передачі шифрованої інформації, тобто передавати безперервно текст з рівномірно розподіленою частотою вживання символів, на який одночасно накладати відкритий текст, який треба передати. Тобто система синхронно "накладає" і "знімає" шифровані дані на оригінальні. Якщо передавання інформації такою системою організовано цілодобово і є неперервне в часі – через великий об'єм інформації така система дорога у використанні, але практично не може бути взламана навіть кібервійськами [66, 67].

Щоб спроектувати структуру компонентів безпеки, а також визначитися яким чином їх використовувати, бажано визначити, від кого ці компоненти повинні захистити систему, а також хто може бути зловмисником – особа,

група осіб чи організація. Можна виділити 4 групи класифікації за рівнем очікуваного ефекту захищеності інформації:

1. Системи із інформацією найбільш високого рівня, зокрема рівня національної безпеки і оборони (загальнодержавного значення), де необхідно забезпечити захист високої якості.

2. Системи із інформацією бізнесового та промислового значення для великих бізнес-груп, корпорацій, фірм, банківських структур.

3. Системи із інформацією бізнесового та господарського значення малих підприємств, окремих середніх підприємств, які не мають можливостей передбачати значні фінансові ресурси на безпеку і захист інформації.

4. Системи із інформацією бізнесового, господарського та побутового значення окремих осіб (приватна інформація), підприємців, які не планують або не мають можливостей витрачати значні кошти на захист інформації, тому використовують широковідомі засоби захисту.

Кожен рівень вимагає окремого підходу як у обранні засобів захисту, так і для протидії загрозам. Здійснені дисертаційні дослідження орієнтовані на засоби і методи захисту і протидії для 4, 3 і частково 2 класифікаційних груп.

Визначення рівня складності однозначно визначає методи протидії. Кібервійська переважно використовуються для систем 1 групи складності. Для систем 2, 3 і 4 групи складності доцільно застосовувати перспективні комп’ютерні шифри і комп’ютерні засоби захисту із використанням біометричної інформації, в яких є можливість використовувати їх легку адаптацію при зміні умов захисту (зміна ключів, паролів, додаткових умов протидії злому), які не потребують великої обчислювальної роботи, а також незначно збільшують об’єм шифрованого повідомлення при прийнятній ефективності захисту інформації.

Розглянемо особливості блокових шифрів з точки зору ефективності, надійності та доцільності застосування маскуючих елементів. До блокових належать шифри, які за один такт шифрування перетворюють певну кількість символів блоку – k. Не піддається сумніву, що такий шифр матиме вищу

надійність у порівнянні з шифрами, в яких за один такт перетворюється один символ. До блокових належать, зокрема, такі шифри: шифр мережа Фейстеля, шифр Хілла, шифр Віженера, шифр RSA та інші. Шифр мережа Фейстеля, шифр RSA, DES – це сучасні комп'ютерні шифри, їх переваги і якості відомі.

Традиційними стали спроби щодо науково-дослідних розробок у напрямку створення нових шифрів з високими якостями, і, відповідно, пошуком нових методик їх злому. Значна частина відомих шифрів маю свою, вже відому, методику злому, що зумовлено цілим рядом факторів. По-перше, в багатьох країнах світу зняли заборону на публікацію дослідних матеріалів з криптографії, криптології, тобто доступ до таких досліджень є публічним. По-друге, ряд країн сформував кібервійська, які вирішують задачі з криптографії, в першу чергу щодо пошуків шляхів взлому нових шифрів. Використання кібервійськ, які при зломі шифрів застосовують багатовекторну масовану атаку, вимагає нових методів злому, і, відповідно, захисту від таких методів.

Захищаючи інформацію, важливо не лише правильно вибрати шифр і ключ шифрування та тримати їх захищеними, але й приховати криптографічну систему. Як правило, навіть найнадійніша система захисту не може забезпечити необмежене у часі збереження конфіденційності інформації, поготів якщо стоїть задача передавати інформацію відкритими каналами зв'язку, забезпечити її отримання легальними абонентами, а потім дешифрувати.

Досліджуються методи захисту інформації, які окрім забезпечення безперешкодної криптографічної роботи, забезпечують приховування методу шифрування. Для пошуку, зокрема, таких методів шифрування використовуються відомий інструмент дослідження шифрів – аналіз статистичних характеристик шифрованих текстів (які сформовані одним методом і одним ключем) для одного символу, біграм і триграм. Також необхідну інформацію можна отримати, здійснивши аналіз повторень в шифрованому тексті, в тому числі розривчастих. Власне проти таких методів злому здійснювалися дисертаційні дослідження щодо забезпечення

ефективності нових методів шифрування інформації із використанням маскуючи елементів.

Пошук нових алгоритмів і способів шифрування інформації виконувався в напрямках використання маскуючих символів для підвищення ефективності шифрів, а також подальшого ускладнення розкриття використаного методу шифрування.

Одним із підходів, що використовується у криптосистемах, є гамування. Це один із методів симетричного шифрування, який виконує сумування послідовності випадкових чисел (гама послідовність) з числами, які відповідають символам відкритого тексту [4, 13]. Найпростіший спосіб сумування виконується в полі Галуа GF(2) – операція “виключне АБО (XOR)”. При цьому використовується наступний базовий аналітичний вираз.

$$\beta_i = (\alpha_i + \gamma_i) \bmod 2,$$

де α_i – символи відкритого тексту, β_i – символи шифрованого тексту, γ_i – послідовність випадкових чисел.

К.Шеннон довів, що при певних властивостях гами, такий метод шифрування наближається до абсолютно стійкого [83]. Якщо при шифруванні і передачі інформації відкритими каналами зв’язку приховати початок процедури шифрування (передавати гаму безперервно), тоді математичні засоби для злому шифру будуть відсутні. Неперервний потік випадкових чисел створює великі потоки інформації, які не підлягають аналізу [4, 13]. Доведення Шеннона базується на визначені закону розподілу значень змінних шифрованого тексту (β_i). Припустимо, що a_i, β_i, γ_i – дискретні випадкові величини, p – імовірність події, що змінна α_i буде мати значення 0, $(1-p)$ – імовірність протилежної події (тобто, імовірність того, що змінна α_i буде мати значення 1).

Закон розподілу значень α_i:

α_i	0	1
P_i	p	$(1-p)$

γ_i – послідовність випадкових (псевдовипадкових) чисел гами; γ_i – змінна може приймати значення 0 або 1, кожне значення рівноймовірне, тобто значення 0 і 1 рівноймовірні і рівні 0,5.

Закон розподілу значень γ_i:

γ_i	0	1
P_i	0,5	0,5

β_i – послідовність символів шифрованого тексту. Припустивши, що змінна β_i може набувати два значення: 0 і 1, значення β_i обчислюватиметься на основі значень α_i, γ_i за формулою $\beta_i = \alpha_i + \gamma_i (\mathrm{mod}\, 2)$ або $\beta_i = \mathrm{XOR}(\alpha_i, \gamma_i)$.

Знайдемо імовірності таких подій:

$P_{(\beta i = 0)}$ – імовірність події, що змінна β_i буде мати значення 0;

$P_{(\beta i = 1)}$ – імовірність події, що змінна β_i буде мати значення 1;

Використаємо формули: додавання $P(A+B) = P(A) + P(B)$ та множення ймовірностей несумісних подій $P(A \times B) = P(A) \times P(B)$.

Імовірність події, що змінна β_i буде мати значення 0:

$$P_{(\beta i = 0)} = P_{(\alpha i, = 0, \gamma i = 0)} + P_{(\alpha i, = 1, \gamma i = 1)} = P_{(\alpha i, = 0)} \times P_{(\gamma i = 0)} + P_{(\alpha i, = 1)} \times P_{(\gamma i = 1)} =$$
$$= p \times 0{,}5 + (1 - p) \times 0{,}5 = 0{,}5.$$

Імовірність події, що змінна β_i буде мати значення 1:

$$P_{(\beta i = 1)} = 1 - P_{(\beta i = 0)} = 0{,}5.$$

Закон розподілу значень β_i:

β_i	0	1
P_i	0,5	0,5

Таким чином закон розподілу β_i виявився симетричним, як і закон розподілу γ_i. Це означає, що β_i не містить жодної інформації, яка визначається змінною α_i, і доводить, що гамування забезпечує створення абсолютно стійких шифрів.

Для визначення можливостей використання маскуючих елементів у блокових шифрах доцільно зупинитися на найбільш вживаних шифрах та їх особливостях, звернувши увагу на вимоги до алгоритмів шифрування.

У 1973 році Національне бюро стандартів США (NBS) опублікувало вимоги до криптографічного алгоритму, який міг би використовуватися в якості стандарту. Були сформульовані такі вимоги до алгоритму [4, 13, 29]:

1. Алгоритм повинен забезпечити високий рівень безпеки.
2. Алгоритм повинен бути повністю визначений і зрозумілий.
3. Безпека повинна базуватися на ключі, не залежачи від збереження в таємниці алгоритму.
4. Алгоритм повинен бути доступний усім користувачам.
5. Алгоритм повинен дозволяти виконувати адаптацію до різноманітних варіантів використання.
6. Алгоритм повинен надавати можливість перевірки.
7. Алгоритм повинен бути дозволений для експорту.

Звичайно, що нові способи шифрування інформації повинні відповідати більшості із перелічених вимог, тоді криптографічний алгоритм можна зарахувати до категорії стійких і перспективних.

Одним із таких способів шифрування інформації є мережа Фейстеля. Дослідження його особливостей дозволяє зробити висновок щодо можливості його модифікації із використанням маскуючих елементів. Мережа (конструкція) Фейстеля – є різновидом блочного шифру та являється загальним методом перетворення множини даних за допомогою деякої функції, що носить назву f-функції. f-функцію, яка являється основним будівельним блоком мережі Фейстеля, завжди обирають нелінійною й, переважно, необоротною. Запропонована Х.Фейстелем схема відповідає вимогам, що ставляться до криптографічних систем, саме тому була використана в проекті Lucifer компанії IBM. В подальшому, зазначений проект став базисом для криптосистеми DES. Ітеративна структура алгоритму дозволяла спростити його реалізацію в

апаратному середовищі. Зазначимо, що ряд сучасних алгоритмів використовують мережу Фейстеля як базис [13].

Основна перевага блокових шифрів забезпечується за рахунок поєднання в процесі шифрування процедур перестановок і підстановок. При розмірі блоку перетворення пару десятків біт стійкість забезпечується величезним обсягом варіантів, які повинні розглядатися при пошуку ключа і алгоритму. Водночас, при розмірі блоків шифрування 128 біт та більше реалізація мережі Фейстеля на 32-розрядних архітектурах може викликати певні труднощі, тому прийнято використовувати модифіковані варіанти цієї конструкції. Зазвичай використовуються 4-х гілкові мережі. Інформаційний блок ділиться на дві рівні частини L_0 та R_0, які за тактами перетворюються та переставляються. Половина блоку L_0 в першому такті перетворення переставляється на місце другої половини блоку шифрування R_0. Одночасно друга половина блоку шифрування R_0 перетворюється за певним алгоритмом і переставляється на місце першої половини блоку шифрування.

Алгоритм перетворення наведений нижче [4, 13]:

– блок відкритого тексту розділяється на 2 рівні частини (L_0, R_0);

– в кожному раунді ($i = 1,...,n$) беремо половину розрядів для лівого блоку та другу половину для правого блоку, обчислюючи

$$L_i = R_{i-1} \oplus f(L_{i-1}, R_{i-1}),\ R_i = L_{i-1},$$

де f – деяка функція, а K_{i-1} – ключ *i*-го раунду. Результатом виконання *n* раундів є (L_n, R_n) [29].

При розмірі блоків шифрування 128 біт і більше пошук ключів (K_{i-1} – ключ *i*-го раунду), які змінюються в кожному раунді, розміру блока і функції f вимагає значних технічних, часових, організаційних і фінансових ресурсів.

Аналіз наведених особливостей дозволяє зробити висновок можливості застосування маскуючих елементів у відкритому тексті при використанні криптосистеми за модифікованою мережею Фейстеля.

Шифр RSA належить до криптографічних систем з відкритим ключем, які використовують односторонні функції: якщо відомо x, то $f(x)$ обрахувати не складно, водночас, якщо відомо $y = f(x)$, то для того, щоб обчислити x, простих рішень не існує. Дослідження його особливостей дозволяє зробити висновок щодо можливості його модифікації із використанням маскуючих елементів.

В основі безпеки алгоритму RSA лежить принцип складності розв'язання задачі факторизації – подання числа простими співмножниками. Алгоритмом використовується два ключі – відкритий/публічний (public) і секретний/закритий (private), які разом (відкритий і відповідний йому закритий) утворюють ключову пару (keypair). Відкритий ключ не потрібно приховувати, – він застосовується для шифрування даних. Водночас, після шифрування повідомлення відкритим ключем, розшифрувати його можна виключно відповідним закритим ключем.

Для того, щоб згенерувати ключову пару, необхідно виконати такі дії [4, 13]:

– випадково обрати два великі прості числа p і q завдовжки приблизно 512 біт кожне;

– обчислити їх добуток (модуль) $n = p \cdot q$;

– для n обчислити значення функція Ейлера $\varphi(n) = (p-1) \cdot (q-1)$;

– обрати ціле число e того самого порядку, що й число n і, крім того, числа e та $\varphi(n)$ повинні бути взаємопростими: НСД $(e, \varphi(n)) = 1$, наприклад прості числа Ферма 17, 257 або 65537;

– за допомогою алгоритму Евкліда вирахувати число d таке, що $ed \equiv 1 (\mathrm{mod}\, \varphi(n))$.

Число n називається модулем, а числа e і d – відкритою (encryption) й закритою (decryption exponents) експонентами відповідно. Пара чисел (n, e) є відкритою частиною ключа, пара (n, d) – закритою. Числа p і q після

генерації ключової пари можуть бути знищені, але, категорично, не повинні бути розкриті.

Шифрування повідомлення полягає в обчисленні значення $c = m^e \bmod n$, при цьому шифрувати можна числа з проміжку $0 \le m \le n$.

Число c використовується в якості шифротексту. Для дешифрування потрібно обчислити $m = c^d \bmod n$.

Нескладно переконатися, що при дешифруванні ми відновимо вихідне повідомлення: $c^d \equiv \left(m^e\right)^d \equiv m^{ed} \bmod n$.

З умови $ed \equiv 1(\bmod \varphi(n))$ випливає, що $ed = k\varphi(n) + 1$ для деякого цілого k, отже $m^{ed} \equiv m^{k\varphi(n)+1} (\bmod n)$.

Згідно з теоремою Ейлера $m^{\varphi(n)} \equiv 1(\bmod n)$, тому $m^{k\varphi(n)+1} \equiv m(\bmod n)$, $c^d \equiv m \bmod n$.

Для шифрування з допомогою шифру RSA доцільно використовувати довжину блоку, яка визначається інформацією про 1,5 або 2,5, або 3,5 тощо символів. Якщо використати числа, які відповідають одному символу – тоді шифрований текст вироджується в шифр простої заміни, що недопустимо для використання. Якщо використати числа, які відповідають двом символам – тоді шифрований текст вироджується в шифр заміни 2 символів. Такі шифри також не представляють великих проблем для дешифрування. Довжина блоку 1,5 або 2,5, або 3,5 виявляє суттєві проблеми для злому, тому що для обчислення оберненої функції простих рішень не існує.

Аналіз наведених особливостей дозволяє зробити висновок, що застосування маскуючих елементів у відкритому тексті при використанні криптосистеми за модифікованим алгоритмом RSA є складнішим у порівнянні із модифікованою мережею Фейстеля.

Ще один відомий шифр Віженера [4, 13, 48] – спосіб шифрування інформації на основі поліалфавітних перетворень елементів відкритого тексту (ВТ). Особливість зазначеного способу полягає в заміні кожного елемента ВТ на елемент шифрованого тексту (ШТ) згідно з літерою ключа, причому для

кожної літери ключа є відповідний алфавіт заміни елементів ВТ. Якщо кожній із літер українського алфавіту поставити у відповідність числа від 0 до 32 (а – 0, б – 1, в – 2, ...), то шифрування і дешифрування шифром Віженера можна представити формулами:

$$C_i = (P_i + K_j) \bmod 33,$$

$$P_i = (C_i - K_j + 33) \bmod 33, \text{ де}$$

C_i - літера шифрованого тексту;

K_j – j-та літера ключового слова;

P_i – літера відкритого тексту.

Ключове слово повторюється до того часу, поки не отримано гаму, що дорівнює довжині повідомлення.

Недоліком даного способу шифрування інформації є те, що при значних обсягах ВТ можна знаходити повторення в ШТ, які будуть розташовуватись на відстанях, які кратні довжині ключа *k*.

Найкраще унаочнює застосування маскуючих елементів у текстових даних компонентів безпеки шифр Хілла [69]. Спосіб шифрування на основі шифру Хілла – поліграмного блокового методу підстановки, що використовує методи лінійної алгебри, був запатентований у 1929 році, що зумовило його належність до ручних. Водночас, основна ідея шифру є достатньо ефективною. Ключем шифру Хілла є матриця, яка записується у формі довільного набору літер або слова. Для шифрування використовується квадратна матриця з чисел (3×3, 4×4, 5×5, ...), яка обов’язково повинна мати обернену, що уможливить операцію дешифрування.

Шифрування інформації відбувається у спосіб, описаний нижче.

$C_i = A \times B_i$ – це основна процедура шифрування інформації, де:

A – матриця-ключ для шифрування інформації;

B_i – матриця-стовпець і-го блоку відкритого тексту;

C_i – матриця-стовпець і-го блоку шифрованого тексту.

$B_i = A^{-1} \times C_i$ - процедура дешифрування інформації, де

A^{-1} – обернена матриця-ключ для дешифрування інформації;

B_i – матриця-стовпець і-го блоку відкритого тексту;

C_i – матриця-стовпець і-го блоку шифрованого тексту.

Щоб дешифрувати повідомлення, необхідно трансформувати шифротекст зворотньо у вектор, помноживши потім зазначений вектор на обернену матрицю ключа.

Звернемо увагу на ряд проблем, пов'язаних із вибором шифрувальної матриці, основна з яких – не усі матриці мають обернену. Матриця має обернену лише у випадку, коли її детермінант не дорівнює нулю і не має спільних дільників з основою модуля. Таким чином, якщо ми працюємо з основою модуля 26, то детермінант повинен бути ненульовим і, одночасно, не ділитися на 2 і 13. Якщо детермінант матриці еквівалентний нулю або має спільні дільники з основою модуля, то така матриця не може бути використана в шифрі Хілла, отже, необхідно обирати іншу матриця, інакше шифротекст буде неможливо дешифрувати. Водночас, матриці, які задовольняють зазначеним вище умовам, існують, а їх вибір не є складною задачею.

Особливості шифрування полягають в наступному. У випадку використання латинського алфавіту, кожній літері ставиться у відповідність число від 0 до 25: A = 0, B =1, ..., Z=25. Блок з n літер представляється як n-мірний вектор і множиться на матрицю $n \times n$ за модулем 26 (якщо використовується інший алфавіт, зокрема додані розділові знаки, число буде відповідати потужності алфавіту). Ключем шифру є матриця.

У наступних прикладах використовуються латинські літери від A до Z, відповідні їм числові значення наведені в таблиці [42].

A	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**I**	**J**	**K**	**L**	**M**	**N**	**O**	**P**	**Q**	**R**	**S**	**T**	**U**	**V**	**W**	**X**	**Y**	**Z**
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25

Розглянемо процес зашифровування слова 'BCD' з допомогою ключа (GYBNQKURP у буквенному представленні) і відповідному числовому

представленні у вигляді матриці розміром 3x3 [42]: $\begin{pmatrix} 6 & 24 & 1 \\ 13 & 16 & 10 \\ 20 & 17 & 15 \end{pmatrix}$.

Оскільки літері 'B' відповідає число 1, 'C' – 2, 'D' – 3, то повідомлення можна подати як матрицю-стовпець (або вектор): $\begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}$.

У цьому випадку зашифрований вектор буде:

$$\begin{pmatrix} 6 & 24 & 1 \\ 13 & 16 & 10 \\ 20 & 17 & 15 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix} \bmod 26 = \begin{pmatrix} 57 \\ 75 \\ 99 \end{pmatrix} \bmod 26 = \begin{pmatrix} 5 \\ 23 \\ 21 \end{pmatrix}$$

що відповідає шифрованому тексту 'FXV'. Бачимо, що кожна літера ШТ змінилася, тобто шифр Хілла досягнув дифузії за Шенноном: *n*-розмірний шифр Хілла може досягнути дифузії *n* символів за раз.

Для того, щоб розшифрувати повідомлення, необхідно перетворити символи шифрованого тексту у вектор і перемножити на обернену матрицю ключа (IFKVIVVMI у буквенному представленні). У нашому випадку обернена матриця матиме вигляд

$$\begin{pmatrix} 8 & 5 & 10 \\ 21 & 8 & 21 \\ 21 & 12 & 10 \end{pmatrix}.$$

Якщо перемножити матрицю ключ на матрицю-стовпець ШТ – отримаємо:

$$\begin{pmatrix} 8 & 5 & 10 \\ 21 & 8 & 21 \\ 21 & 12 & 10 \end{pmatrix} \cdot \begin{pmatrix} 5 \\ 23 \\ 21 \end{pmatrix} \bmod 26 = \begin{pmatrix} 365 \\ 730 \\ 549 \end{pmatrix} \bmod 26 = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}.$$

Результуюча матриця-стовпець дає можливість відновити символи відкритого тексту 'BCD'.

Використання у текстових даних маскуючих елементів є одним із підходів щодо підвищення ефективності захисту інформації блоковими шифрами.

3.2. Новий спосіб шифрування інформації з використанням маскуючих елементів

У запропонованому способі шифрування інформації [42] використовуються маскуючі елементи, які вставляються серед символів відкритого тексту (ВТ) за певним алгоритмом, що ускладнює процедуру розпізнавання ВТ перебором можливих варіантів шифрів і ключів для кожного шифру, що зумовлює підвищення ефективності системи шифрування.

На рис.3.1 наведена узагальнена схема алгоритму вставлення маскуючих елементів у відкритий текст.

У новому методі шифрування інформації виконують поділ символів відкритого тексту (ВТ) на блоки по **μ** символів у блоці, які утворюють матрицю-стовпець, а ключ формують з μ^2 кількості символів, які записують як квадратна матриця **μxμ.** Символи шифрованого тексту (ШТ) формуються в процесі поблокового перемноження матриці-стовпця і квадратної матриці ключа шифрування, які попередньо перетворюють у відповідні числа по модулю **n**, де **n** – кількість символів ВТ (потужність алфавіту відкритого тексту). Дешифрування шифрованого тексту виконують поділом символів ШТ на блоки (по **μ** символів у блоку) і перемноженням матриці-стовпця і квадратної матриці-ключа дешифрування, які перетворюють у відповідні числа по модулю **n**, де **n** – кількість символів ВТ.

Згідно запропонованого методу, перед множенням на матрицю-ключ шифрування у відкритий текст перед і після кожного символу ВТ вставляють додаткові маскуючі елементи, причому маскуючі елементи на кожному кроці вставлення визначаються найменшою частотою вживання цього елементу (із врахуванням вставлених маскуючих елементів) у відкритому тексті з маскуючими елементами, а при дешифруванні вилучають маскуючі елементи в

такому порядку, як вони вставлялися перед множенням на матрицю-ключ шифрування [23].

Рис. 3.1. Загальна схема алгоритму вставлення маскуючих елементів

Процедурі шифрування передує вставлення перед і після кожного символу ВТ додаткових маскуючих елементів, причому при довжині блоку шифрування **μ** необхідно вставляти таку кількість маскуючих елементів (перед і після символу ВТ), щоб в кожний блок шифрування потрапив хоча б один символ ВТ. Хоча ця вимога не є критичною для виконання, занадто багато маскуючих елементів вставляти недоцільно, оскільки досягнення позитивного результату можливе бути при незначному збільшенні кількості символів шифрованого тексту (ШТ).

Одним із елементів криптосистеми є генератор псевдовипадкових символів. Додаткові маскуючі елементи вибираються керованим генератором псевдовипадкових чисел таким чином, щоб статистичний аналіз ВТ до і після вставлення маскуючих елементів змінювався в сторону рівномірної частоти вживання символів. Генератор псевдовипадкових чисел на кожному кроці вставлення елементу вибирає такий символ, який має найменшу частоту вживання. Частота використання символів визначається на кожному кроці, і з кожним кроком частотна характеристика ВТ з маскуючими елементами стає все більш рівномірною, що унеможливлює отримання однозначного результату при обробці статистичних параметрів тексту.

Маскуючі елементи у кожному конкретному випадку встановлюються відповідно до обраного методу. Способів встановлення маскуючих елементів може бути багато, а тому алгоритм їх встановлення є доповнювальним захистом до вибору матриці-ключа, оскільки без знання алгоритму встановлення, і, відповідно, вилучення маскуючих елементів, не можливо отримати ВТ. Виключно сукупність відомостей про метод вставлення маскуючих елементів і ключ дає можливість дешифрувати ШТ.

Формування матриць-ключів для шифрування і дешифрування виконується аналогічно до шифру Хілла. Формат матриць ключа і вектора відкритого тексту для шифрування може бути 3, 4, 5, 6... На теперішній час відсутні технічні проблеми апаратного, програмного чи комбінованого способу перемноження матриць розміром 3×3, 4×4, 5×5, 6×6, ...

Частоту використання символів у відкритому тексті (ВТ) також не складно визначити за допомогою *k* лічильників, які будуть визначати скільки разів вживався кожний символ у ВТ. Таким чином, маскуючі елементи будуть вставлятися перед і після символів ВТ в оберненій залежності до частоти використання: чим рідше вживається окремий символ ВТ, тим частіше він буде вставлятися як маскуючий елемент. Якщо частоту використання символів підраховувати після кожного циклу вставлення (перед і після окремого символу ВТ), то очевидно, що чим більше циклів вставлення маскуючих символів, тим більш рівномірною буде частотна характеристика вживання символів.

Основним прийомом для розпізнавання способу шифрування та визначення довжини ключа є статистичне опрацювання тексту, яке визначає частоту повторення символів і груп символів ШТ, що може допомогти визначити довжину ключа. Маскуючі елементи, які вставляються перед і після кожного символу ВТ, а також їх процедура вставлення, в певній мірі мають випадковий характер, фактично набуваючи властивості додаткового шифрувального ключа. Справа в тому, що якщо при перемноженні матриць у матрицю символів ВТ вставляється хоча б один маскуючий елемент, то, у підсумку, змінюються усі результуючі символи ШТ. Якщо перемножимо матрицю-ключ на матрицю-вектор ВТ, то для наведеного прикладу отримаємо

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix} \cdot \begin{pmatrix} 10 \\ 11 \\ 12 \end{pmatrix} \bmod 26 = \begin{pmatrix} 16 \\ 11 \\ 6 \end{pmatrix}.$$

Якщо в матриці-векторі ВТ замінимо один символ (11 поміняємо на 5), то отримуємо:

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix} \cdot \begin{pmatrix} 10 \\ 5 \\ 12 \end{pmatrix} \bmod 26 = \begin{pmatrix} 4 \\ 7 \\ 10 \end{pmatrix}.$$

Приклад підтверджує, що заміна одного символу у векторі ВТ (11 замінили на 5) призвела до зміни усіх символів результату. Таким чином, вставлення навіть у невеликій кількості маскуючих елементів (наприклад вставляється один маскуючий елемент перед символом відкритого тексту при форматі матриці ключа 3×3 і матриця вектор ВТ буде змінена так: один маскуючий елемент і два символи ВТ в усіх блоках шифрування) призведе до кардинальної зміни елементів ШТ. Цей ефект досягається від вставлення маскуючих елементів, після чого відбувається процедура перемноження матриці ключа на матрицю вектор ВТ. Таким чином усі закономірності ШТ (з точки зору статистичних характеристик і методик їх обробки) суттєво змінюються.

Проблемним є злом коротких текстів, зашифрованих запропонованим методом шифрування, оскільки статистика для таких паролів, кодів, умовних команд чи інших кодових слів, які означають режими роботи тощо, не відповідає статистиці природної мови ВТ. Враховуючи вищезазначене, запропонований спосіб шифрування характеризується потенційно високими параметрами криптостійкості, особливо при використанні його в системах безпеки для збереження конфіденційної інформації в комп'ютерних кіберфізичних системах [24, 26].

Після підготовки ВТ до процедури шифрування виконується поділ символів відкритого тексту з маскуючими елементами (ВТ+М) на блоки по **μ** символів у блоку. Ці **μ** символів утворюють матрицю стовпець, а ключ утворюється з $\boldsymbol{\mu}^2$ кількості символів, які записуються як квадратна матриця **μхμ** і символи шифрованого тексту (ШТ) формуються в процесі перемноження поблоково матриці стовпця і квадратної матриці, які попередньо перетворюються у відповідні числа за модулем **n**, де **n** – кількість символів ВТ. Процедура дешифрування здійснюється у зворотному порядку, тобто символи ШТ діляться на блоки по **μ** символів у блоці і почергово перемножуються на обернену матрицю-ключ, що дає ВТ+М. Остання дія, яку необхідно виконати – це з дешифрованого тексту ВТ+М видалити усі маскуючі елементи в порядку, відповідно до якого вони вставлялися, а тоді отримаємо ВТ.

Перелік дій при шифруванні даних із використанням маскуючих елементів подано структурною схемою, приведеною на рис. 3.2.

Рис. 3.2. Структурна схема шифрування даних із використанням маскуючих елементів

Особливість формування елементів шифрованого тексту як результат множення матриці ключа на вектор-стовпець (*j*-тий блок складений з символів відкритого тексту)

$$V'_j = \sum (K_{ij} \times V_j) \bmod n \text{, де}$$

K_{ij} – елементи матриці ключа;

V_j – елементи матриці відкритого тексту;

V'_j – елементи матриці відкритого тексту.

При використанні маскуючих символів всі елементи матриці V'j змінюються згідно з виразом

$$V'_j = \sum (K_{ij} \times V_j + K_{ij} \times M_j) \bmod n .$$

Згідно із алгоритмом, формування маскуючих елементів відбувається за псевдовипадковим законом, і тому розподіл шифрованих символів в кожному блоці наближається до випадкового закону. Якщо послідовний набір випадкових чисел $N_B = \{n_0, n_1, n_2, ..., n_s\}$ додати почленно з набором чисел, які

відповідають еквівалентним значенням символів шифрованого тексту, то отримаємо наступне:

$$N_B + M_{Ш} = \{n_0, n_1, n_2, ..., n_s\} + \{m_0, m_1, m_2, ..., m_s\} =$$

$$= \{(n_0 + m_0) \bmod n, (n_1 + m_1) \bmod n, (n_2 + m_2) \bmod n, \ldots (n_s + m_s) \bmod n\}.$$

Простий аналіз показує, що при такому додаванні двох масивів чисел розподіл результуючого масиву чисел буде наближатися до випадкового закону.

Запропонований спосіб шифрування інформації [23,42] має високі параметри криптостійкості, легко реалізується апаратним, програмним або комбінованим способом.

Шифр Хілла [69] для **μ**=6 був реалізований у вигляді механічної шифрувальної машинки, яка системою шестерень і ланцюгів здійснювала множення матриць розмірності 6×6 за модулем 26. Цей факт унаочнює, що реалізація таких шифрів потребує малих обсягів обчислювальної роботи і може виконуватися сучасними мікроконтролерами та мікро-ЕОМ чи спеціалізованими комп'ютерними системами.

За потреби отримати високі параметри криптостійкості, необхідно вставляти достатньо маскуючих елементів, кількість яких може в декілька разів перевищувати кількість символів відкритого тексту ВТ. За приклад використаємо наступний алгоритм вставлення маскуючих елементів: вставляється один маскуючий елемент перед кожним символом ВТ і один маскуючий елемент після символу ВТ. У цьому випадку ВТ з маскуючими елементами буде мати конфігурацію: в кожному блоці (якщо **μ**=3) буде один маскуючий елемент перед символом відкритого тексту, символ ВТ і один маскуючий елемент після символу ВТ. Блок матиме такий вигляд:

$\{m_i; v_i; m_i\}$, де m_i – маскуючий елемент, v_i – символ ВТ.

Якщо конфігурація ВТ з маскуючими елементами буде такою, що розглядалася вище, а **μ** = 4, тоді блоки матимуть вигляд:

перший – $\{m_i; v_i; m_i; m_i\}$; другий – $\{v_i; m_i; m_i; v_i;\}$; третій – $\{m_i; m_i; v_i; m_i\}$; четвертий – $\{m_i; v_i; m_i; m_i\}$; п'ятий – $\{m_i; v_i; m_i; m_i\}$.

П’ятий – такий як перший, тобто увесь цикл повторюватиметься з періодом чотири.

Якщо обрати алгоритм вставлення маскуючих елементів, відповідно до якого вставляються два маскуючі елементи перед кожним символом ВТ і нуль маскуючих елементів після символу ВТ при **μ**=3, то у цьому випадку блок матиме такий вигляд: $\{m_i; m_i; v_i\}$, де m_i – маскуючий елемент, v_i – символ ВТ. Усі блоки набуватимуть такого вигляду, оскільки кількість вставлених маскуючих елементів, які припадають на один символ ВТ, дорівнює **μ**-1.

Варіацій, які визначають конфігурацію ВТ з маскуючими символами, може бути багато, водночас обирати необхідно такі, які забезпечуватимуть рівномірність статистичної характеристики окремих елементів ШТ. Дослідження статистичних характеристик окремих символів ШТ навіть при m_i=1 підтверджує ефективність запропонованого способу шифрування інформації.

Особливість запропонованого способу шифрування така, що певний символ відкритого тексту в різних блоках шифрується по-різному залежно від змісту блоку. У випадках, коли блок символів відкритого тексту має такий же вміст, але хоча б один маскуючий елемент відрізняється (що є високоімовірним), тоді і відповідний символ шифрованого тексту буде відрізнятися.

3.3. Адаптивні методи вставлення маскуючих елементів

Запропоновано маскуючі елементи обирати з можливого набору елементів межах потужності множини алфавіту повідомлення. Маскуючі елементи визначаються нерівномірністю статистичної характеристики розподілу елементів за принципом: кожний маскуючий елемент, який вставляється, повинен покращувати рівномірність статистичної характеристики розподілу символів відкритого тексту, в яких додаються маскуючі елементи. Очевидно, що чим більш рівномірний статистичний розподіл символів відкритого тексту, тим більш рівномірний буде і розподіл символів шифрованого тексту.

3.3.1. Статичний метод вставлення маскуючих елементів

Головна особливість статичного методу вставлення маскуючих елементів – маскуючі елементи завжди вставляються у наперед визначені місця відносно символів відкритого тексту. На рис. 3.3 наведені три варіанти графічної моделі для статичного методу вставлення маскуючих елементів для формату **μ** = 3.

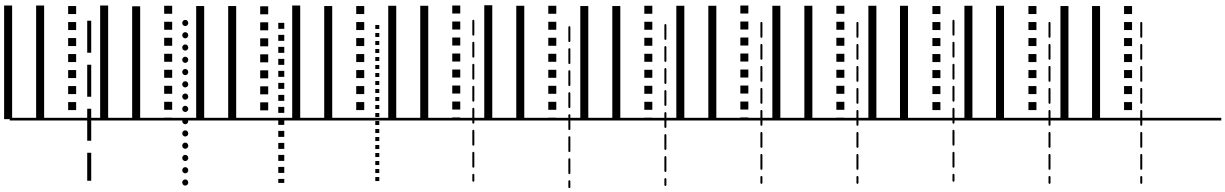

Рис.3.3. Графічна статична модель з форматом $\{v_i; v_i; m_i\}$, де m_i – маскуючий елемент – пунктирна лінія, v_i – символ ВТ – жирна лінія. Блоки розділені пунктирними лініями. Довжина блоку – 3 символи

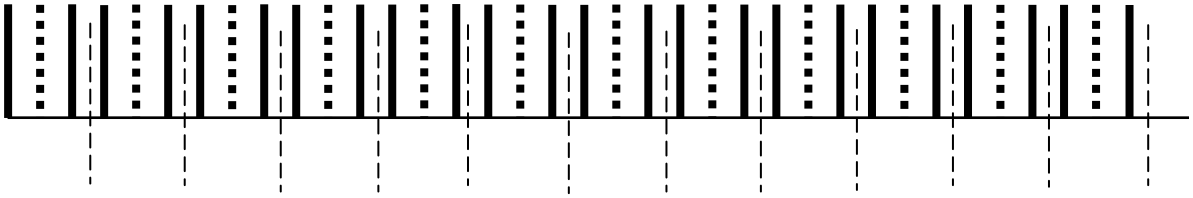

Рис. 3.4. Графічна статична модель з форматом $\{v_i; m_i; v_i\}$, де m_i – маскуючий елемент – пунктирна лінія, v_i – символ ВТ – жирна лінія. Блоки розділені пунктирними лініями. Довжина блоку – 3 символи

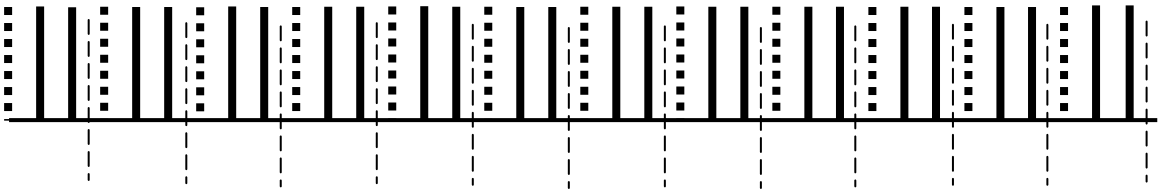

Рис. 3.5. Графічна статична модель з форматом $\{m_i; v_i; v_i\}$, де m_i – маскуючий елемент – пунктирна лінія, v_i – символ ВТ – жирна лінія. Блоки розділені пунктирними лініями. Довжина блоку – 3 символи

Для оцінки ефективності використання маскуючих елементів найдоцільніше використовувати показник ефективності, в якому оцінюється середнє інтегральне відхилення для конкретного випадку. У наведених методах формування маскуючих елементів для формату **μ**=3, які пояснюються графічними моделями на рис. 3.3 – 3.5, досягається результат зменшення

середнього інтегрального відхилення у 1,4 рази. Такий результат є задовільним і забезпечує підвищення ефективності криптографічної системи [22].

3.3.2. Динамічний метод вставлення маскуючих елементів

Особливість динамічного методу встановлення маскуючих елементів – маскуючі елементи вставляються у кількості і на позиції залежно від порядкового номеру елементу відкритого тексту, а їх кількість змінюватиметься на кожному раунді процедури вставлення. На рис. 3.6 наведено графічні моделі для динамічного методу вставлення маскуючих елементів при використанні формату $\boldsymbol{\mu} = 5$ [23].

Динамічна функція вставлення маскуючих елементів надає додатковий ефект: якщо у звичайному шифрі Хілла висока імовірність появи повторень в тексті на відстанях, що кратні довжині ключа (число $\boldsymbol{\mu}$ не може бути дуже велике), то при вставленні маскуючих елементів після кожного символу відкритого тексту у кількості від 0 до 5 при $\boldsymbol{\mu} = 5$ період повторення буде 105 символів (було 5).

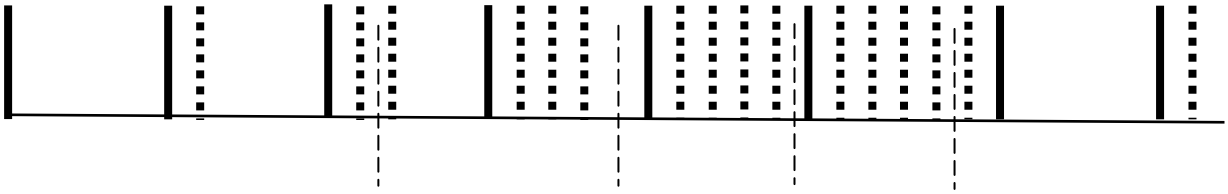

Рис.3.6. (а)

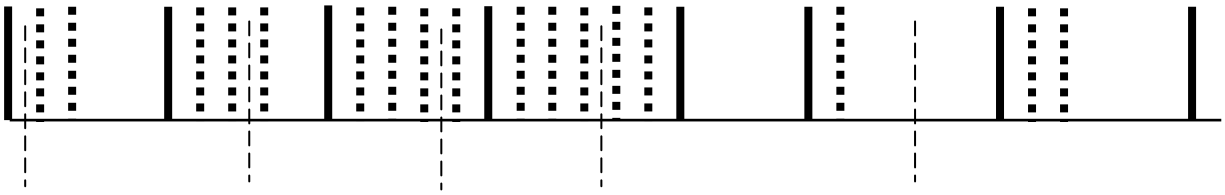

Рис.3.6. (б)

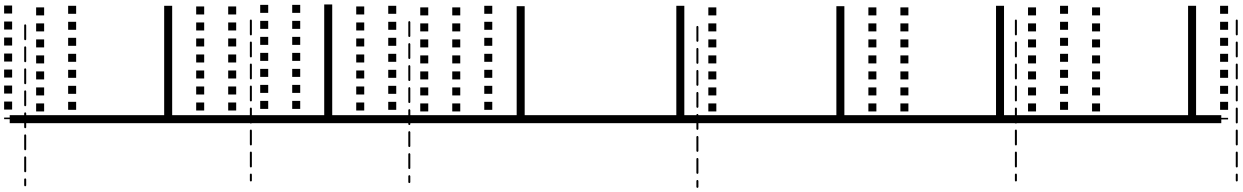

Рис.3.6. (в)

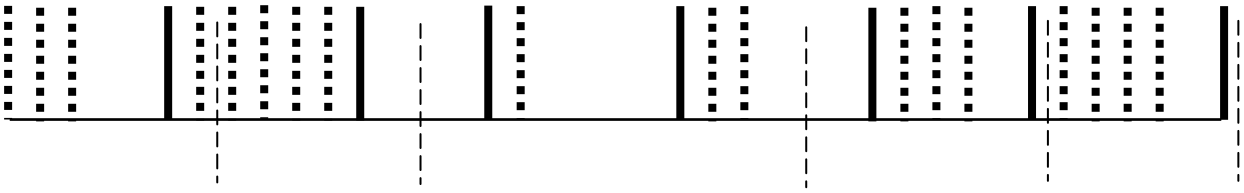

Рис.3.6. (г)

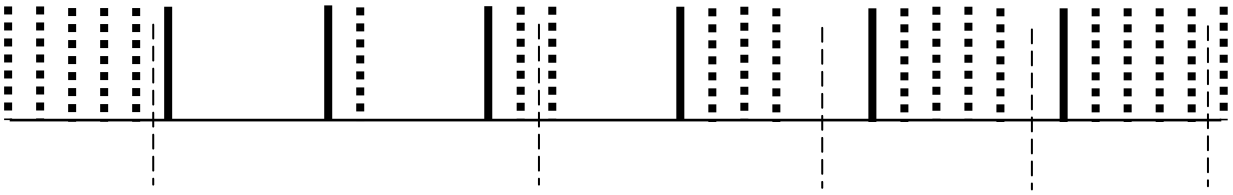

Рис.3.6. (д)

Рис.3.6. (а - д). Графічна динамічна модель з форматом $\{v_i; n_j \times m_i\}$, де m_i – маскуючий елемент – пунктирна лінія, v_i – символ ВТ – жирна лінія, n_j – коефіцієнт поступово змінюється від 0 до 5 (залежно від порядкового номеру символу ВТ v_i). Блоки довжиною 5 символів розділені пунктирними лініями

Модель вставлення маскуючих елементів може бути статична, чи динамічна. Можна використати поліном для розрахунку місць і кількості вставлення маскуючих елементів:

$$N = (n_1 + n_2 z + n_3 z^2 + ... + n_k z^{k-1}) \bmod \mu ,$$

де N – кількість маскуючих символів, які вставляються після z_i-го символу відкритого тексту;

n_i – коефіцієнти полінома;

μ – довжина блоку (формат шифрування).

Власне процедура вставлення маскуючих елементів і їх вилучення є процедурою, яка суттєво не зменшує продуктивність роботи як криптографа, так і криптоаналітика. Необхідно врахувати, що маскуючі елементи формуються генератором псевдовипадкових чисел із числа найменш вживаних символів у шифрованому тексі. Такий алгоритм підбору можна вважати додатковим ключем для формування шифрованого тексту.

Складність вилучення маскуючих елементів не визначається їх номером чи назвою, так як вилучаються символи на відповідних позиціях шифрованого тексту. Якщо кількість маскуючих елементів становить більше 50%, тоді

частотний розподіл символів у шифрованому тексті наближається до рівноймовірного. Таким чином, використання маскуючих елементів є перспективним напрямом розширення функціональних можливостей при створенні шифрів, який вирішує задачу покращення ефективності компонентів безпеки комп'ютерних систем та мереж.

Запропонований спосіб шифрування інформації з використанням маскуючих елементів належить до блокових шифрів, в яких застосовуються два прийоми підвищення ефективності – перестановки і підстановки. Вставлення маскуючих елементів призводить до того, що символи шифрованого тексту змінюють свої позиції у тексті відносно символів відкритого тексту. Як було унаочнено вище, у кожному блоці достатньо одного маскуючого елементу, щоб змінилися усі символи блоку. Якщо врахувати, що маскуючі елементи формуються за допомогою генератора псевдовипадкових чисел, то цей процес збільшує реальну довжину блокового ключа шифрування. Фактично запропонований спосіб шифрування володіє усіма якостями блокових шифрів з використанням перестановки і підстановок для підвищення ефективності.

Якщо раніше уся робота в криптографії виконувалася вручну і було небажано збільшувати довжину ШТ, то сьогоднішній стан розвитку шифрувальної техніки дозволяє при використанні криптографічних засобів у кіберфізичних системах ігнорувати цю важливу в минулому особливість [24, 26, 35]. Використання шифрувальних машин та інших спеціалізованих комп'ютерних засобів дозволяє швидко виконати як збільшення ШТ, так і видалення маскуючих символів, не зменшуючи продуктивності праці оператора при шифруванні чи дешифруванні інформації.

3.4. Показник ефективності блокових шифрів на основі внесених змін статистичних характеристик шифрованого тексту

Зазвичай ефективність шифрів оцінюють за критерієм, який характеризує необхідні ресурси для визначення конкретного типу шифру, його ключа і подальшого дешифрування тексту. Регульована зміна статистичних

характеристик шифрованих текстів дає можливість значно ускладнити процес криптоаналізу та покращити ефективність шифру.

Очевидно, що всі нові шифри аналізуються, а їх характеристики досліджуються. При пошуку і дослідженні шифрів з маскуючими елементами перш за все було поставлено завдання проаналізувати параметри використаного методу шифрування, досліджувались статичні і динамічні методи вставлення маскуючих символів. Сукупність досліджень продемонструвала перспективність створення шифрів з покращеними параметрами щодо частотного розподілу символів у шифрованому тексті. Зрозуміло, що в сучасних умовах працювати без серйозного інформаційного захисту просто неможливо, тому використання в процесі шифрування маскуючих елементів, які формуються генератором псевдовипадкових чисел за певним графіком, розширює функціональні можливості компонентів безпеки. При цьому є можливість використовувати конкретний метод вставлення маскуючих елементів (а їх може бути десятки). Відповідно, запропонований метод шифрування вигідно відрізняється від відомих за такими параметрами як простота, продуктивність, ефективність.

Розглянемо для типових прикладів застосування як в результаті змінився частотний розподіл символів ШТ завдяки модифікації ВТ перед шифруванням. Критерієм покращення стійкості блокового шифру є зменшення середнього інтегрального відхилення. Визначити середнє інтегральне відхилення можна за допомогою формули [6, 25]

$$\sigma = \left[\frac{1}{2} \sum_{i=1}^{n} \frac{(x_{i\max} - x_i)}{x_{i\max}} \right] \times 100\%,$$

де x_i – статистичний параметр для i-го символу тексту, мірою якого є кількість випадків використання символу у досліджуваному тексті;

x_{imax} – максимальне значення x_i для символу, який найчастіше зустрічається у досліджуваному тексті.

Чим менше середнє інтегральне відхилення – тим складніше визначити тип блокового шифру і знайти ключі, тобто розв’язати задачу взлому криптосистеми. Вперше запропонований метод шифрування інформації з використанням маскуючих елементів в деякій мірі є універсальним.

Нижче наведені результати моделювання для різних шифрів з форматом 3x3 (**μ** = 3). Відповідні гістограми наведені на рис. 3.7 та 3.8.

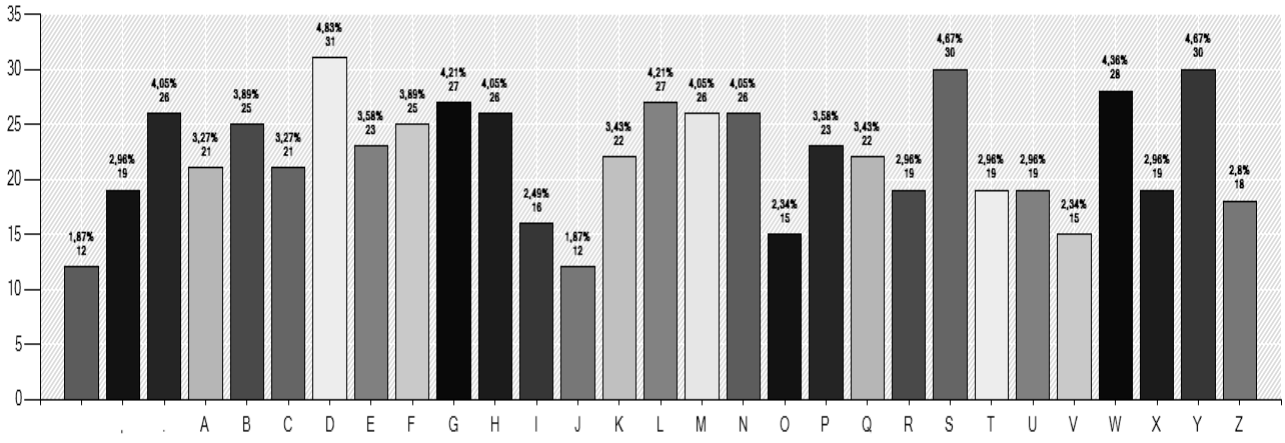

Рис. 3.7. Гістограма для шифру Хілла (формат матриці ключа 3x3) без “маскуючих” елементів

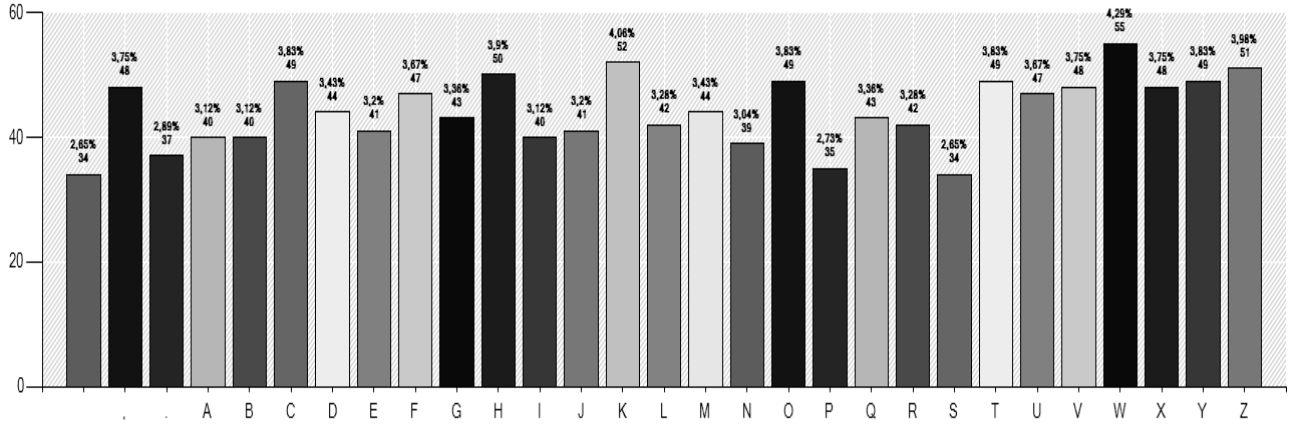

Рис. 3.8. Гістограма для шифру Хілла (формат матриці ключа 3x3) з “маскуючими” елементами

Середнє інтегральне відхилення ШТ методом Хілла без “маскуючих” елементів (рис. 3.7) еквівалентне 27,3%, а для ШТ з “маскуючими” елементами (рис. 3.8) дорівнює 19,6%.

Отже, завдяки використанню маскуючих елементів для методу Хілла покращено частотний розподіл символів шифрованого тексту в 1,4 рази.

Метод шифрування інформації в компонентах безпеки комп'ютерних систем динамічним вставленням маскуючих елементів дозволяє досягти до двократного покращення частотного розподілу символів у шифрованому тексті.

Не маючи ключа, дешифрування ШТ методом перебору усіх можливих варіантів ключів передбачає отримання ВТ, що може бути прочитаний. Якщо зловмисники спробують шляхом перебору почергово усі можливі варіанти ключів, вони не отримають ВТ, який читається, оскільки ВТ було модифіковано перед шифруванням.

Основні інструменти криптоаналітика – аналіз статистичних характеристик шифрованого тексту і аналіз повторень ШТ. У запропонованому методі шифрування вирішальним є спотворення інформації про реплікації, які могли би повторитися у ШТ за аналогією з ВТ. Ця задача успішно вирішується у вперше запропонованому блоковому шифрі з використанням маскуючих елементів у ВТ повідомлення. Зазначена особливість разом із вирівнюванням статистичних характеристик забезпечує приховування використовуваного методу шифрування, що має значний ефект.

Вважаємо достатнім для ефективного покращення частотних характеристик зашифрованого тексту забезпечити зменшення середнього інтегральне відхилення в 1,15 – 1,5 разів.

3.5. Висновки до розділу 3

1. Розглянуто особливості блокових шифрів і варіанти підвищення їх ефективності.

2. Запропоновано новий спосіб підвищення ефективності блокових шифрів з використанням маскуючих елементів.

3. Встановлено, що одним із показників покращення ефективності блокового шифру є суттєве зменшення числового значення середнього інтегрального відхилення частоти вживаних символів у шифрованих текстах,

що ускладнить виконання криптоаналізу на основі частотного аналізу та повторюваності блоків у зашифрованому тексті, визначення типу шифру і підбір для нього ключа.

4. Вперше запропоновано вдосконалений метод шифрування текстової інформації із використанням статичного вставлення маскуючих елементів у відкритому тексті з наступним перетворенням інформації блоковими криптографічними засобами, що покращує частотний розподіл символів у шифрованому тексті.

5. Вперше запропоновано вдосконалений метод шифрування текстової інформації із використанням динамічного вставлення маскуючих елементів у відкритому тексті з наступним перетворенням інформації блоковими криптографічними засобами, що наближує частотний розподіл символів у шифрованому тексті до рівноймовірного.

6. Показано, що використання маскуючих елементів в блокових шифрах має перспективу в напрямі розширення функціональних можливостей при створенні шифрів.

7. Запропоновано використання показника визначення ефективності блокових шифрів на основі внесених змін статистичних характеристик шифрованого тексту, що дозволяє виконати кількісні оцінки внесених відповідними алгоритмами змін.

РОЗДІЛ 4. ДОСЛІДЖЕННЯ ЕФЕКТИВНОСТІ ВИКОРИСТАННЯ МАСКУЮЧИХ ЕЛЕМЕНТІВ В КОМПОНЕНТАХ БЕЗПЕКИ КОМП'ЮТЕРНИХ СИСТЕМ ТА МЕРЕЖ

4.1. Дослідження ефективності використання біометричних даних з маскуючими елементами в компонентах безпеки

Ефективність використання біометричних даних з маскуючими елементами в компонентах безпеки досліджена шляхом участі в проекті удосконалення та розвитку ґрід-кластеру Фізико-механічного інституту імені Г.В. Карпенка Національної академії наук України.

У протоколах ідентифікації користувачів зазначеної системи використано запропонований метод автентифікації користувачів на основі біометричних даних за відбитками пальців з використанням маскуючих елементів – фіктивних фрагментів, математична модель визначника випадкових величин, модель та алгоритм взаємодії між користувачем та засобами криптографічного захисту.

Експериментально встановлено, що покращення криптографічної ефективності алгоритму блокування досягається додаванням до оцифрованого представлення відбитку пальця маскуючих елементів, після чого загальна кількість частинок дорівнюватиме N. Доведено, що N визначається як мінімальне можливе значення відстані L між частинками, яка більша за середню порогову відстань і залежить від технології отримання наборів частинок (технічні характеристики сканера, властивості методу обробки зображень тощо). Чим менша L, тим більше значення N, та, як наслідок, більш стійкий криптографічний захист. Експериментально встановлено, що значення N обмежується збільшенням імовірності помилкового декодування або помилкового відсотку браку, розміщенням дійсних частинок і стандартним відхиленням розміщення цих частинок. Встановлено, що чим більше значення N, тим складніший процес розблокування для зловмисника. Однак збільшення N має бути збалансованим: раціональне значення N знаходиться в межах 50.

Приклад застосування алгоритму блокування зображений на рисунку 4.1.

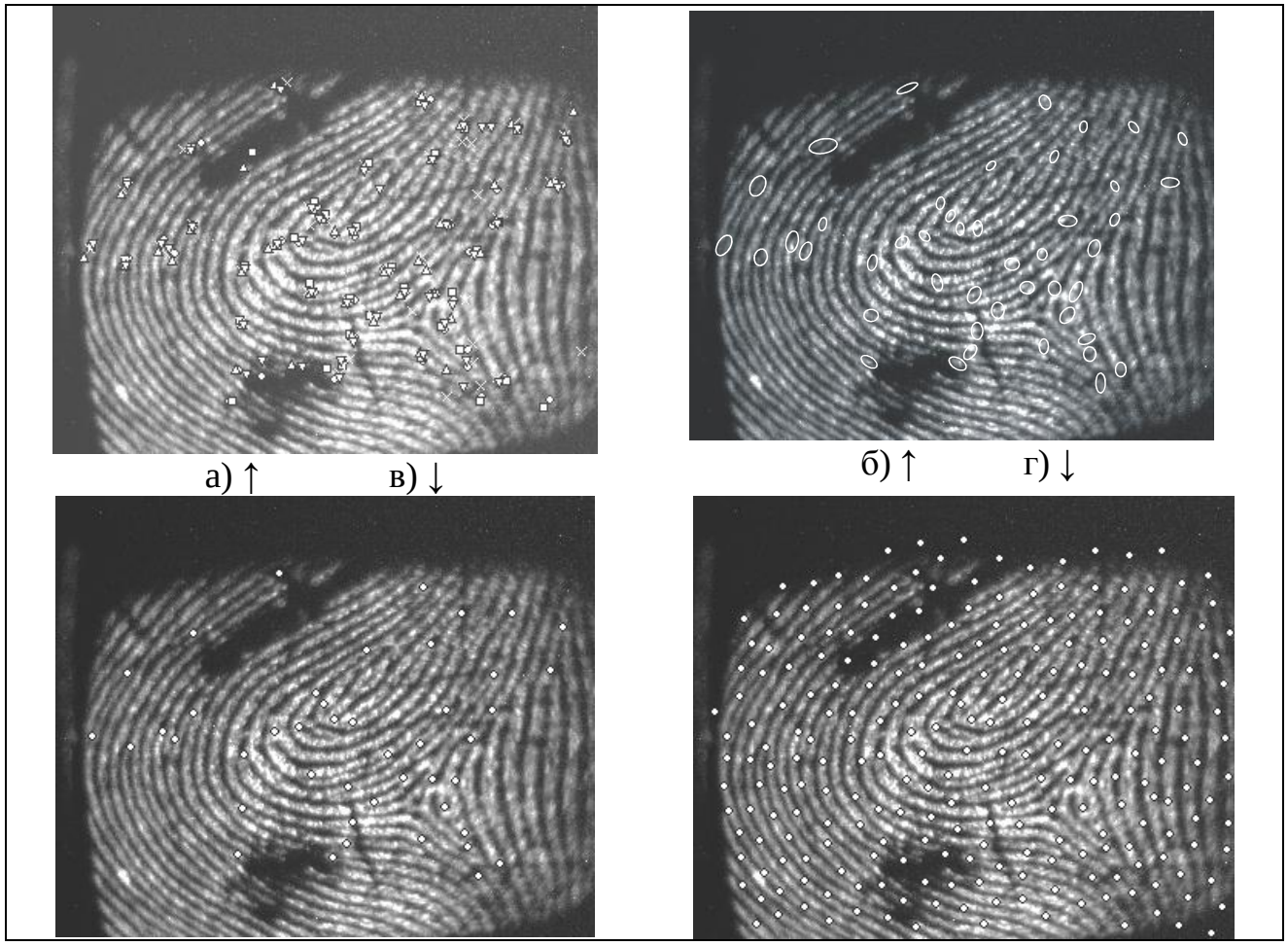

Рис. 4.1. Створення блокуючої множини. (а) K=4 частинкові множини (набори) (символи o, □, ×, Δ – частинки з одного набору); (б) найбільш ймовірні ділянки розміщення користувацьких частинок; (в) центри тяжіння ділянок розміщення частинок – згенерована множина блокування; (г) згенерована множина блокування з маскуючими частинками

Алгоритм біометричного захисту для процедур автентифікації користувачів у ґрід середовищі з використанням механізму доповнень сертифікату X.509.v3 підвищує ефективність застосування біометричних даних у підсистемі контролю доступу ґрід-кластеру.

Структура сертифіката X.509.v3 пропонує механізм доповнень/розширення, що дозволяє зареєструвати у створюваному сертифікаті

необхідну біометричну інформацію. Зокрема, є можливість реєстрації розширень у відповідних інституціях. Кожне розширення містить таку інформацію:

ТИП	КРИТИЧНІСТЬ	ЗНАЧЕННЯ

У зазначеній формі:

– тип – тип використовуваного розширення/доповнення.

– критичність – прапорець, який вказує, чи інформація, яка подана у зазначеному повідомленні, повинна бути опрацьована. Якщо цей прапор встановлений, і додаток не може обробити тип розширення, то додаток повинен відхилити сертифікат.

– значення – безпосередньо дані для розширення. Ці дані будуть оброблятися відповідно до опису розширення.

З основних категорій розширень, які були визначені для 3 версії сертифікату X.509.v3, вставку біометричних даних AI дозволяє лише категорія “Атрибути розширення сертифікат-предмет і сертифікат-емітент”. Ця категорія дозволяє доповнювати сертифікати центру автентифікації (надалі CA-с сертифікати) рядом обмежених специфікацій (тобто сертифікати для центру автентифікації CA, видані іншими CA), що значно спрощує автоматичну обробку сертифікатів у випадку, якщо використовується декілька політик сертифікації (наприклад, коли політика змінюється для різних додатків в операційному середовищі, або коли відбувається взаємодія із зовнішнім середовищем). Обмеження можуть звузити кількість типів сертифікатів, які випускаються центром автентифікації (CA) або які застосовуються під час проходження сертифікації. Таке групування не підійде для автентифікації інформації. Вимоги для цієї секції визначаються наступною тезою: користувач сертифікату має право безпечно знати загальну ідентифікуючу інформацію про предмет, щоб бути впевненим, що об’єкт - насправді визначені людина або річ.

Для використання AI в сертифікаті X.509.v3 ця інформація повинна бути зазначена в якості атрибута. Якщо конструкція AI така, як визначено в ECMA.219, дається синтаксис атрибуту в результаті чого отримуємо [89]:

authenticationlnfo ATTRIBUTE::={

WITH SYNTAX Authenticationlnfo,

ID id-at-TBD

}

Authenticationlnfo: :=SEQUENCE{

authenticationMethod[0] AuthenticationMethod,

exchangeAI[1]AuthMparm,

biometriclnfo Biometriclnfo

}

Якщо засоби безпеки ґрід-системи використовують хеш-дані п’яти відбитків пальців кожного із користувачів, кожен елемент даних буде розміщений в окремому атрибуті *Authenticationlnfo*. Це дозволить програмі, використовуючи зазначену інформацію, перевірити атрибути і обрати відповідні.

Сертифікат, який використовується для зберігання біометричних хеш-даних, повинен бути переданий об'єкту, який виконує перевірку. Об’єкт, який виконує перевірку, повинен перевірити підпис на сертифікаті X.509, щоб виявити зміни і довести справжність біометричних даних. Параметри опрацювання вбудовані в біометричну функцію хеш-обробки, яка перетворює скановане з пальців (*lifescan*) зображення в біометричні (*livescan*) хеш-дані. *Livescan* біометричні хеш-дані, біометричні дані хеш-функції із сертифіката X.509.v3 вбудовані у відповідний алгоритм перевірки користувача. Якщо перевірка пройшла успішно, одержувач сертифіката отримує відповідь від сервера щодо підтвердження автентифікації.

Отже, використання маскуючих елементів біометричних даних за відбитками пальців забезпечило удосконалення ґрід-кластеру Фізико-механічного інституту та покращило ефективність його компонентів безпеки [2, 3].

4.2. Узагальнений показник ефективності нового способу шифрування інформації з використанням маскуючих елементів

Запропонований новий спосіб шифрування інформації з використанням маскуючих елементів [23, 42] досліджено за показником ефективності для визначення частоти розподілу символів на основі внесених змін статистичних характеристик шифрованого тексту [25]. Якщо шифруються параметри авторизації – кількість символів ВТ складає 12-18. При використанні блоку **μ**=4 одночасно із статичним методом вставлення маскуючих елементів (1 маскуючий елемент на кожний блок), для довжини слова 12 символів необхідно встановити 4 маскуючі елементи, а для слова довжиною 18 символів – 6 маскуючих елементів.

Для ефективної роботи блокового методу шифрування інформації з маскуючими елементами необхідно використовувати генератор псевдовипадкових символів з числа найменш вживаних елементів. Конструкція використовуваного генератора псевдовипадкових символів виконана наступним чином. Перед процедурою встановлення маскуючих елементів для відкритого тексту (разом з маскуючими елементами, якщо такі були) визначалася його статистична характеристика. Три найменш вживані символи вставлялися по черзі у місця, визначені відповідно до використовуваного методу. Після цього повторно визначалася статистична характеристика відкритого тексту і далі використовувалися нові визначені елементи. Цей псевдогенератор підтвердив високі якості шифрованого тексту: середньоквадратичне відхилення зменшувалось, повторення у тексті зникали.

При **μ** = 4 досліджено частотні характеристики і визначено середні інтегральні відхилення для відкритого тексту з пробілами (кількість символів – 630). При статичному і динамічному методах вставлення маскуючих елементів середнє інтегральне відхилення зменшувалося на величину не менше 20%.

Процедура вставлення маскуючих символів і їх вилучення практично не зменшує продуктивність роботи криптографа та криптоаналітика, якщо врахувати, що маскуючі елементи обираються із використанням генератора

псевдовипадкових чисел з найменш вживаних елементів у шифрованому тексті. Власне сам алгоритм підбору маскуючих елементів при формуванні шифрованого тексту слід вважати додатковим ключем.

Складність вилучення маскуючих елементів не визначається їх номером чи назвою, оскільки вилучаються символи на відповідних позиціях шифрованого тексту. Моделюванням підтверджено, що при кількості маскуючих символів понад 50%, частотний розподіл символів у шифрованому тексті наближається до рівноймовірного. Із врахуванням доведених К.Шенноном тверджень щодо наближення шифру до абсолютно стійкого при наближенні розподілу частоти вживання символів до рівноймовірного закону [83] прийдемо до висновку, що новий спосіб шифрування інформації з використанням маскуючих елементів підвищує ефективність компонентів безпеки комп'ютерних систем та мереж. Таким чином, використання маскуючих елементів – перспективний напрямок проектування та побудови шифрів підвищеної ефективності.

4.3. Дослідження статистичних характеристик шифрованого новими методами тексту

Досліджено статистичні характеристики шифрованого тексту при використанні статичного і динамічного методу вставлення маскуючих елементів [23]. Формат запропонованого методу – 4×4 (**μ** = 4). Моделювання (результати наведені нижче) виконувались для статичного методу з вставленням одного маскуючого символу в блок. Блоки у різних варіантах матимуть вигляд, який наведений на рисунках 4.3, 4.5, 4.7, 4.9 [22].

Середньоквадратичне відхилення частоти використання символів у шифрованому і відкритому текстах вираховувалось за формулою:

$$\sigma = \left[\frac{1}{2} \sum_{i=1}^{n} \frac{(x_{i\max} - x_i)}{x_{i\max}} \right] \times 100\% \qquad (4.1)$$

де x_i – статистичний параметр для i-го символу тексту, який виражається у кількості випадків використання символу у досліджуваному тексті;

x_{imax} – максимальне значення x_i для символу, який найчастіше зустрічається у досліджуваному тексті.

Результуючі гістограми (у відсотках) наведені на рисунках 4.4, 4.6, 4.8, 4.10.

Графічна статична модель із форматом $\{m_i; v_i; v_i; v_i\}$, де m_i – маскуючий елемент (вертикальна пунктирна чорна лінія), v_i – символ ВТ (вертикальна чорна лінія), наведена на рис. 4.3. Формат шифрування – 4×4 ($\boldsymbol{\mu} = 4$).

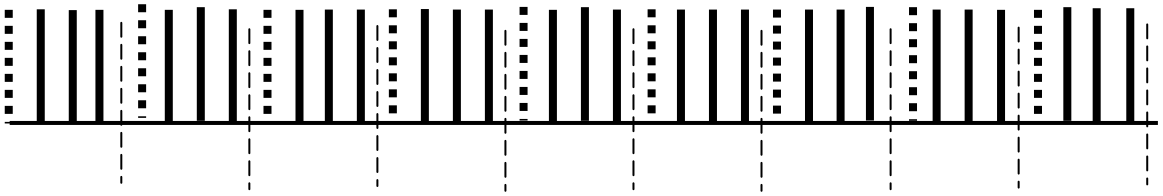

Рис. 4.3. Фрагмент 1 графічної статичної моделі

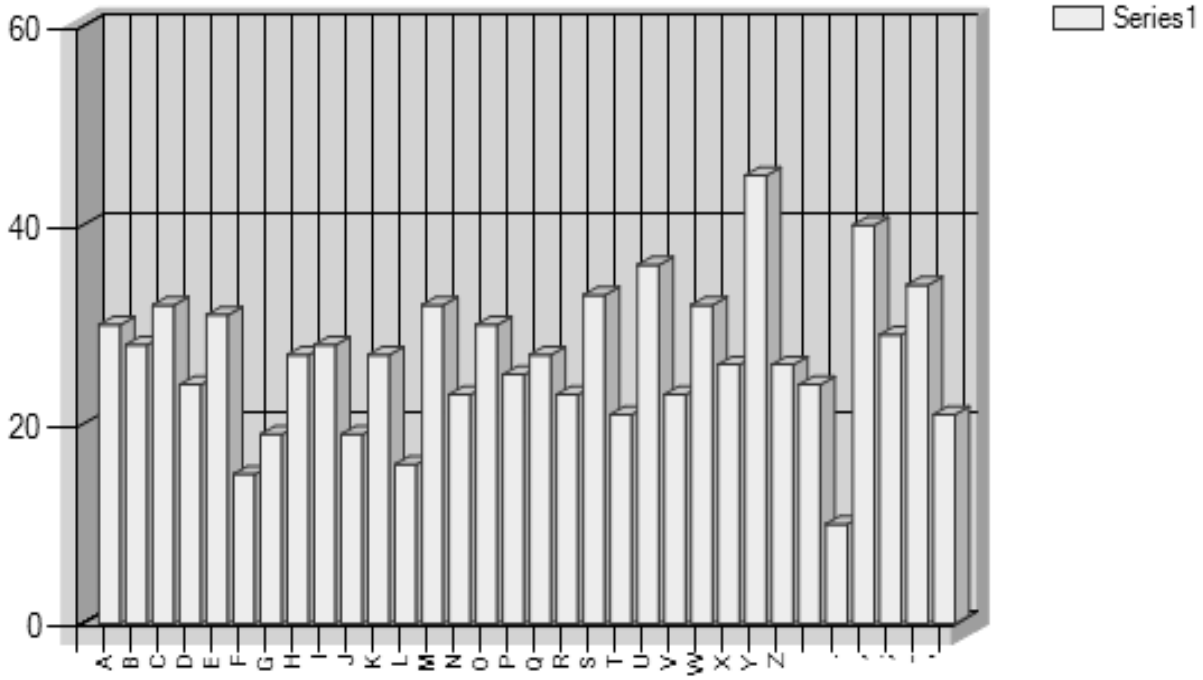

Рис. 4.4. Результат моделювання статистичних характеристик; σ =33,55%

Графічна статична модель з форматом $\{v_i; m_i; v_{i;}; v_i\}$, де m_i – маскуючий елемент (вертикальна пунктирна чорна лінія), v_i – символ ВТ (вертикальна чорна лінія), наведена на рис. 4.5. Формат шифрування – 4×4 ($\boldsymbol{\mu} = 4$).

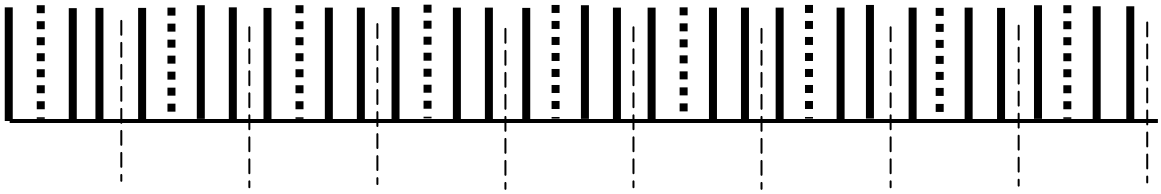

Рис.4.5. Фрагмент 2 графічної статичної моделі

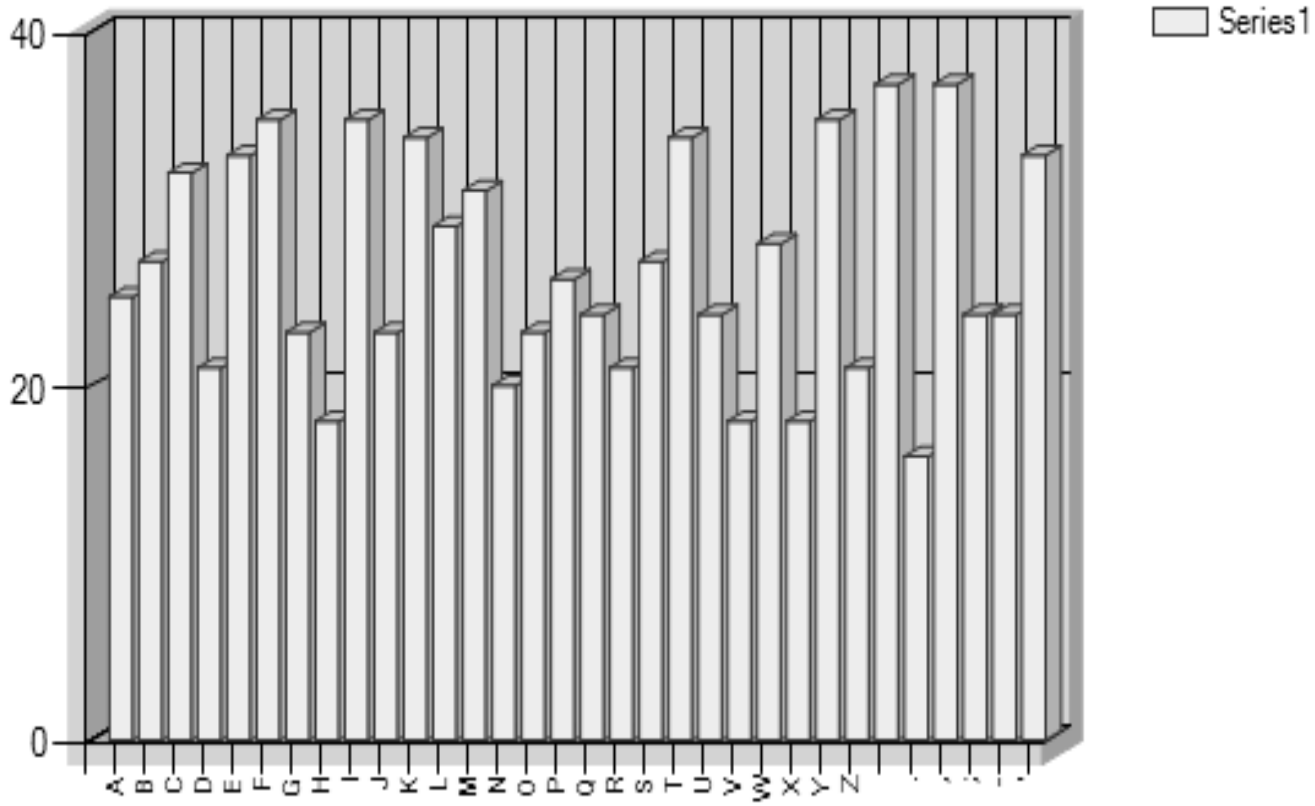

Рис. 4.6. Результат моделювання статистичних характеристик; σ =27,70%

Графічна статична модель з форматом$\{v_i;\ v_i;\ m_i;v_i\}$, де m_i – маскуючий елемент (вертикальна пунктирна чорна лінія), v_i – символ ВТ (вертикальна чорна лінія), наведена на рис. 4.7. Формат шифрування – 4×4 ($\boldsymbol{\mu} = 4$).

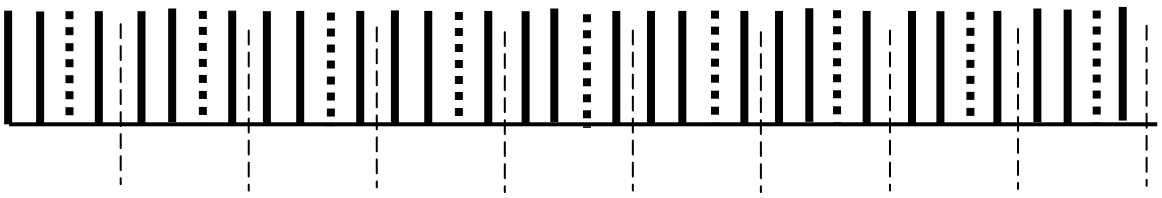

Рис. 4.7. Фрагмент 3 графічної статичної моделі

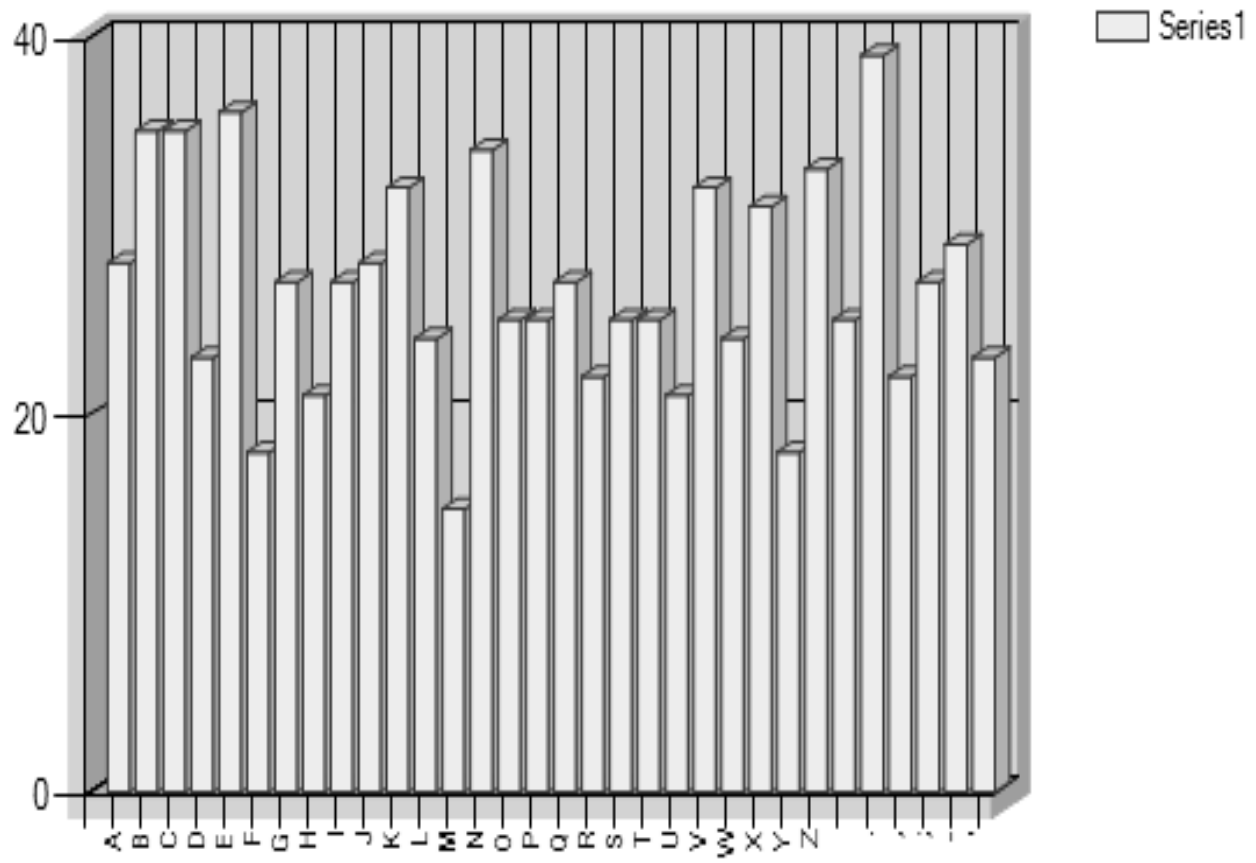

Рис. 4.8. Результат моделювання статистичних характеристик; σ = 31,41%

Графічна статична модель з форматом $\{v_i; v_i; v_i; m_i\}$, де m_i – маскуючий елемент (вертикальна пунктирна чорна лінія), v_i – символ ВТ (вертикальна чорна лінія), наведена на рис. 4.9. Формат шифрування – 4×4 ($\boldsymbol{\mu} = 4$).

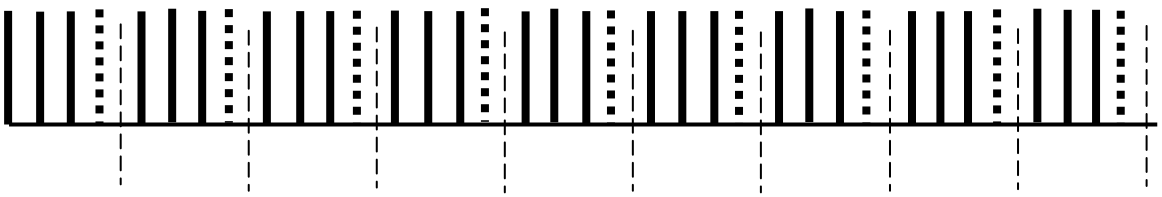

Рис.4.9. Фрагмент 4 графічної статичної моделі

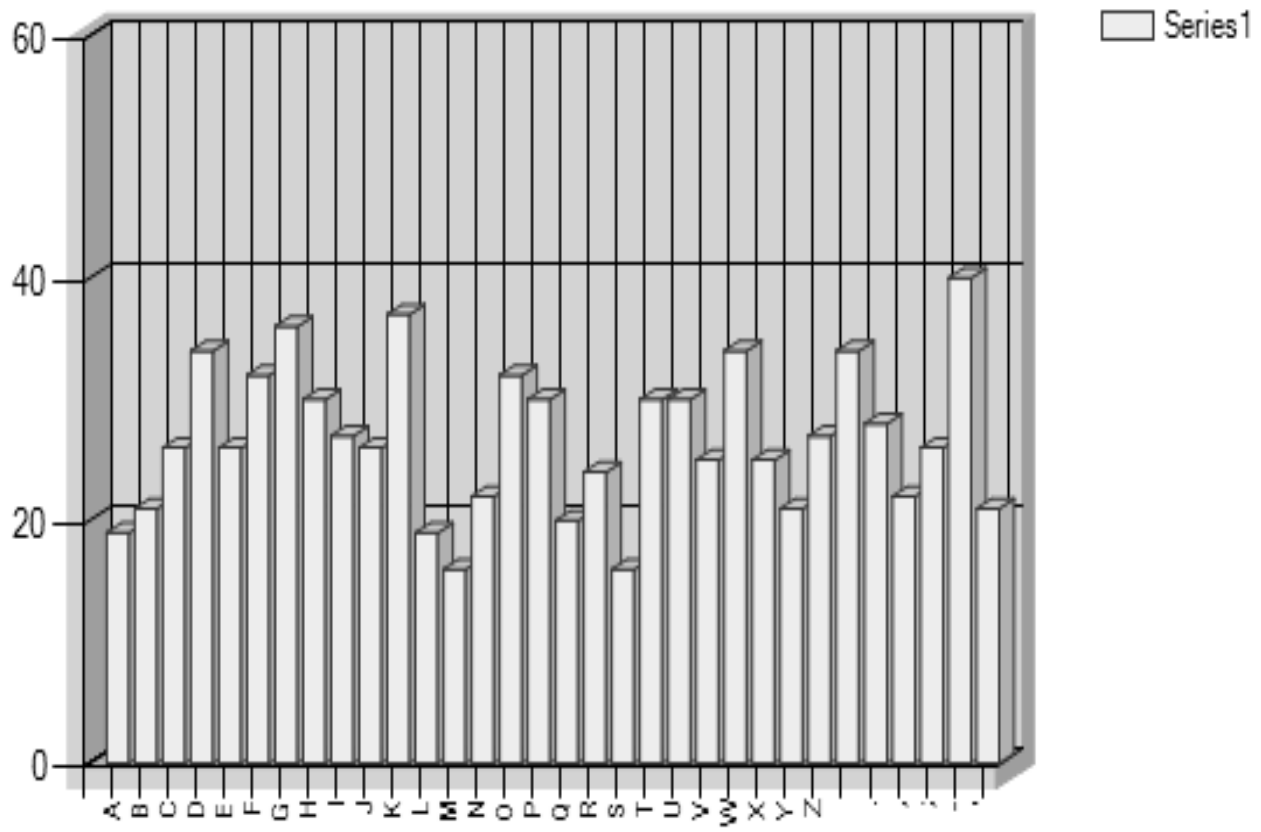

Рис. 4.10. Результат моделювання статистичних характеристик; σ = 33,12%

Наведені результати дослідження шифрів із використанням статичного вставлення маскуючих елементів у відкритий текст дозволяють зробити висновок стосовно суттєвого покращення частотної характеристики шифрованого тексту.

Досліджена динамічна графічна модель з форматом $\{v_i;\ n_j \times m_i\}$, де m_i – маскуючий елемент (вертикальна пунктирна чорна лінія), v_i – символ ВТ (вертикальна чорна лінія), n_j – коефіцієнт поступово змінюється від 0 до 5 (в залежності від порядкового номеру символу ВТ v_i). Формат шифрування – 4×4 ($\boldsymbol{\mu}$ = 4) [22].

Фрагменти графічної динамічної моделі вставлення маскуючих елементів з перерахованими вище параметрами наведена на рис. 4.11.

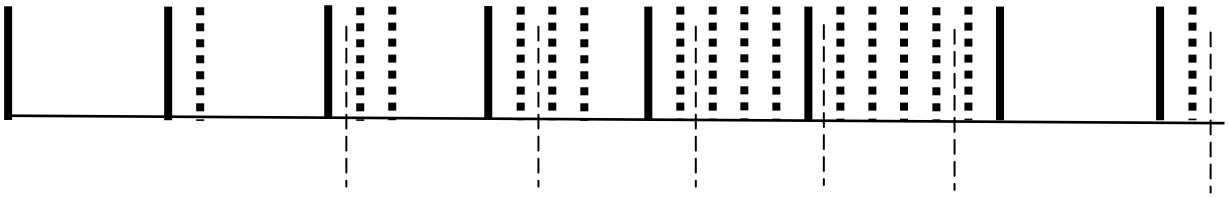

Рис. 4.11. (а)

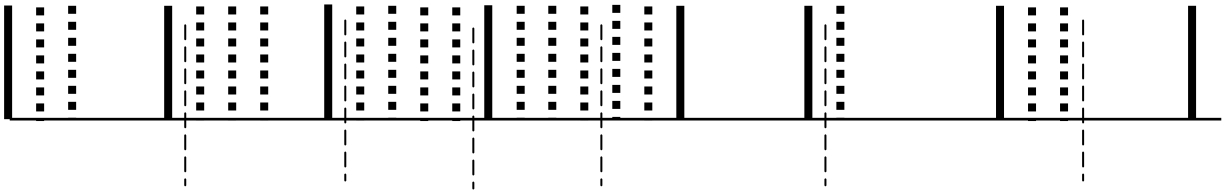

Рис. 4.11. (б)

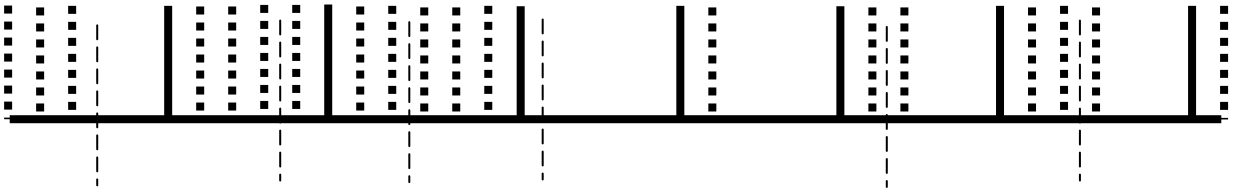

Рис. 4.11. (в)

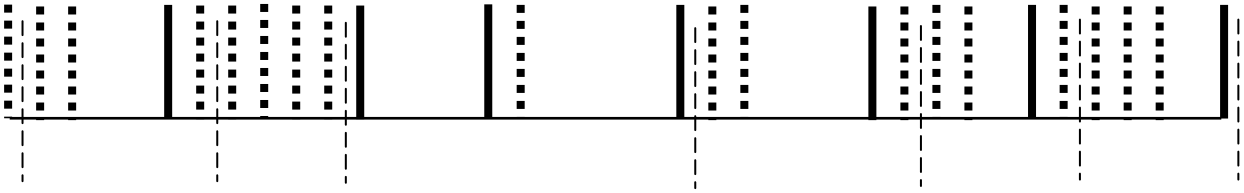

Рис. 4.11. (г)

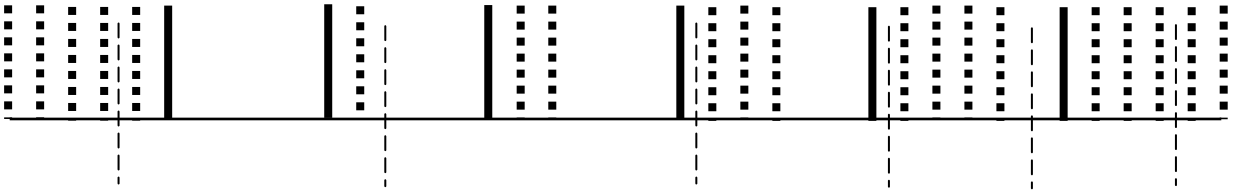

Рис. 4.11. (д)

Рис. 4.11 (а - д). Графічна динамічна модель з форматом $\{v_i; n_j \times m_i\}$, де m_i – маскуючий елемент (вертикальна пунктирна чорна лінія), v_i – символ ВТ (вертикальна чорна лінія), n_j – коефіцієнт поступово змінюється від 0 до 5 (залежно від порядкового номеру символу ВТ v_i). Блоки розділені видовженими пунктирними лініями. Довжина блоку – 4 символи

Динамічна функція вставлення маскуючих елемент дає додатковий ефект: якщо у звичайному блоковому шифрі повторення в тексті можуть з’являтися на відстанях, які кратні довжині ключа (число **μ** не може бути дуже велике), то у

вставленні маскуючих елементів після кожного символу відкритого тексту у кількості від 0 до 5 при μ = 4 період повторення буде 84 символи, що виявлено при аналізі графічної моделі.

Як зауважувалось вище, основними інструментами криптоаналітика, перед яким стоїть задача за статистичними характеристиками знайти ключ і використаний алгоритм шифрування, є повторення в тексті, зокрема розривчасті, а також частотна характеристика вживання символів у шифрованому тексті. Використання маскуючих елементів нівелює частотну характеристику вживання символів у шифрованому тексті. В свою чергу повторення з періодом 84 чи 168 мають дуже низьку імовірність формування. Враховуючи, що маскуючі елементи формуються генератором псевдовипадкових чисел, при зміні хоча б одного символу в блоці змінюється весь блок. Таким чином, використання запропонованого способу шифрування інформації, створює передумови впровадження в практику високоефективних шифрів за рахунок використання різноманітних статичних і динамічних моделей встановлення маскуючих елементів.

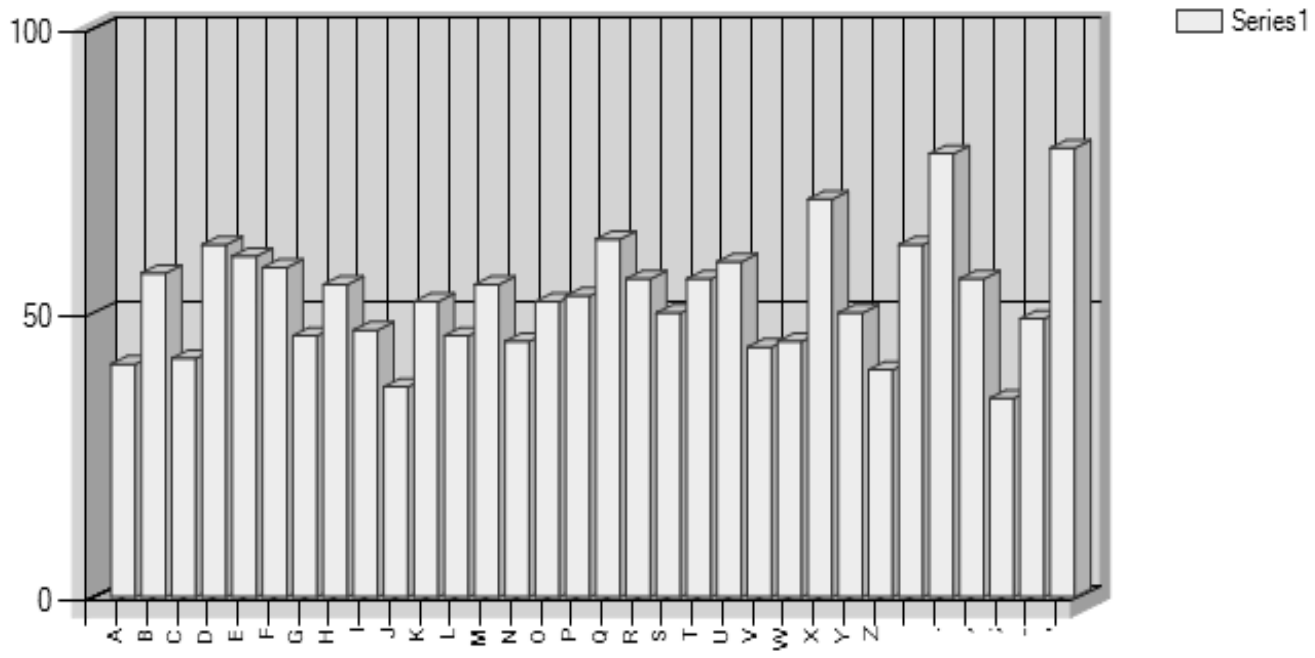

Рис. 4.12. Гістограма результатів моделювання динамічних характеристик;

σ = 32,75%

Чим менше середнє інтегральне відхилення – тим складніше знайти ключі і визначити тип блокового шифру. Середнє інтегральне відхилення при шифруванні запропонованим методом при **μ** = 3 і при одному маскуючому елементу на блок зменшується на 40%. Для моделі при **μ** = 4 і при різних статичних станах вставлення маскуючих елементів і при динамічному методі вставлення середньоквадратичне відхилення зменшується на величину порядку 20% - 60%.

Метод встановлення маскуючих символів можна використовувати і в випадках використаннях таких блокових шифрів як мережа Фейстеля, шифр Віженера тощо. Середнє інтегральне відхилення для ШТ методом Віженера без “маскуючих” елементів (рис. 4.13) еквівалентне 57,3%, а для ШТ з “маскуючими" елементами (рис. 4.14) рівне 41,2%. Отже, завдяки модифікації ВТ середньоквадратичне відхилення зменшується у 1,4 рази для методу Віженера.

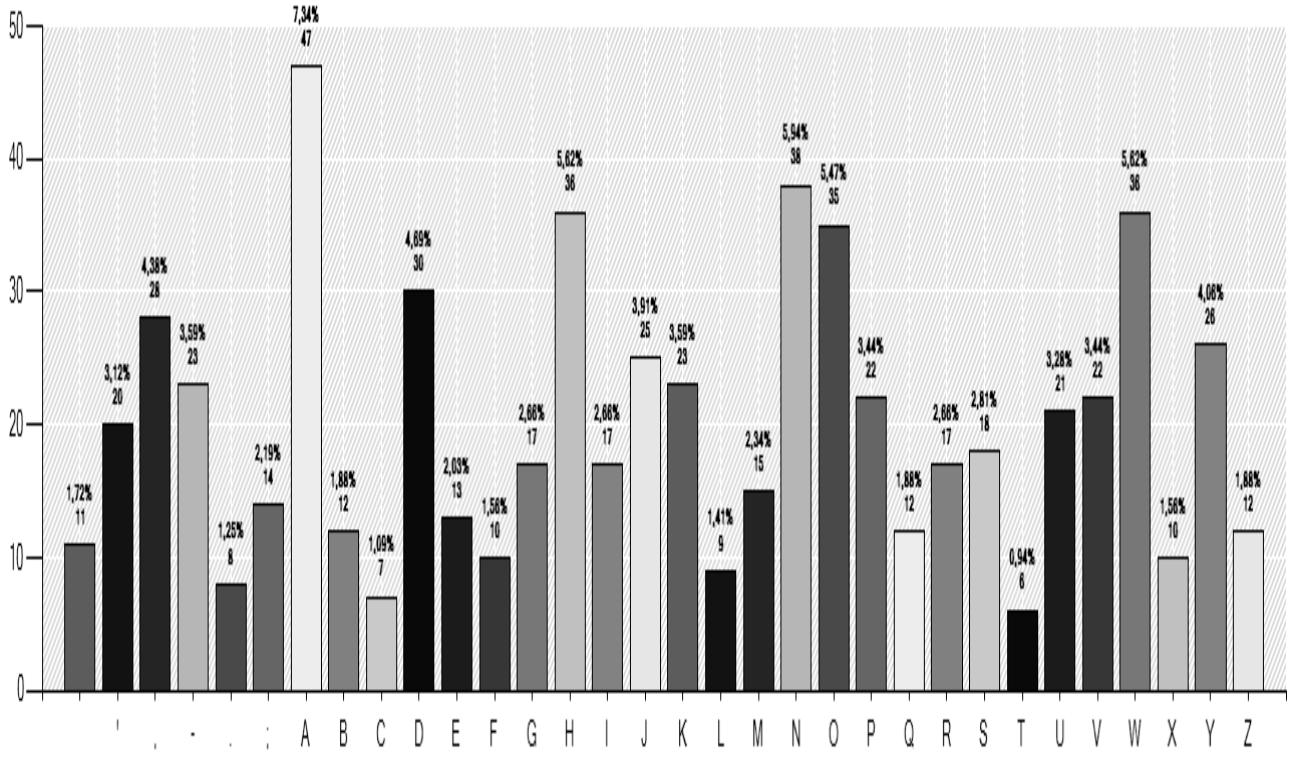

Рис. 4.13. Гістограма для шифру Віженера без “маскуючих” елементів

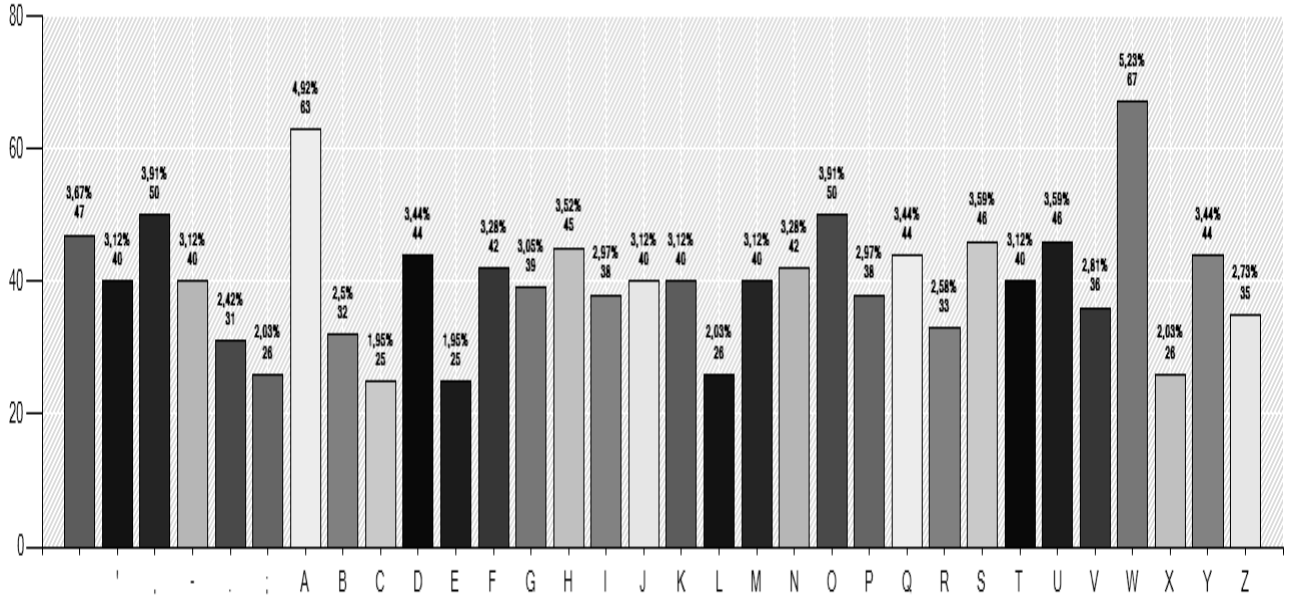

Рис. 4.14. Гістограма для шифру Віженера з “маскуючими” елементами

Середнє інтегральне відхилення для ШТ методом Фейстеля без “маскуючими” елементів (рис. 4.15) дорівнює 68,2%, а для ШТ з “маскуючими” елементами (рис. 4.16) дорівнює 58,9%.

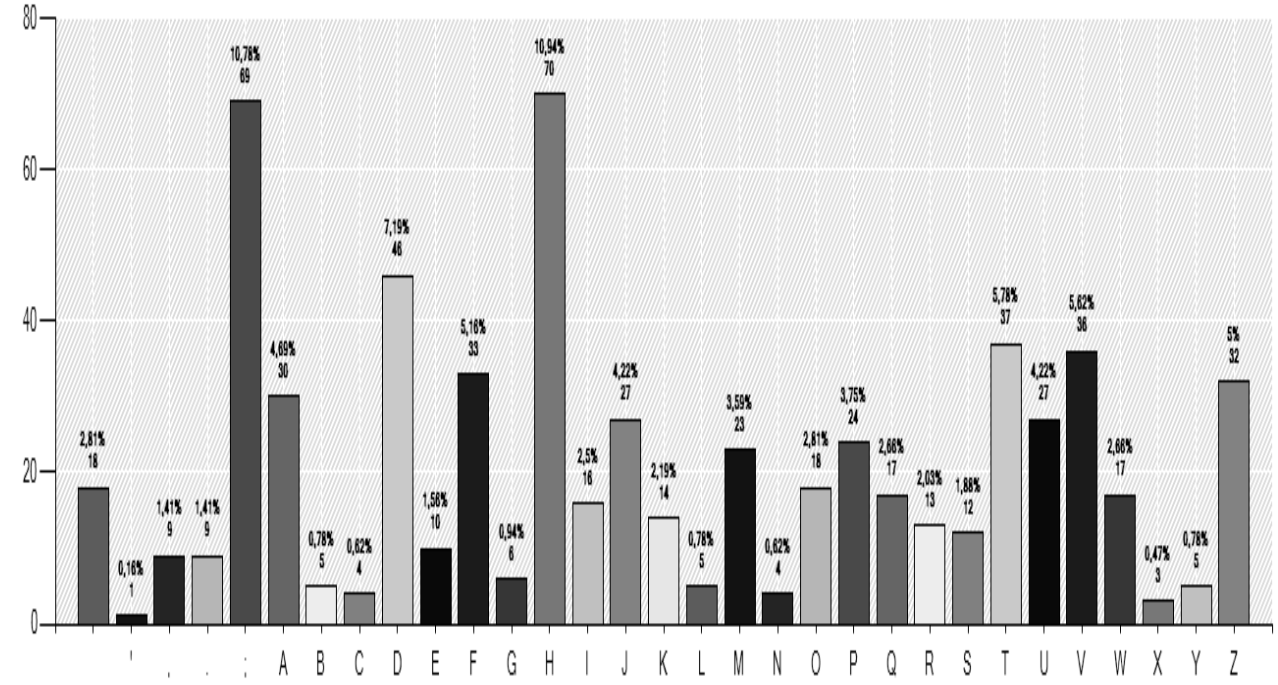

Рис. 4.15. Гістограма для шифру Фейстеля без “маскуючих” елементів

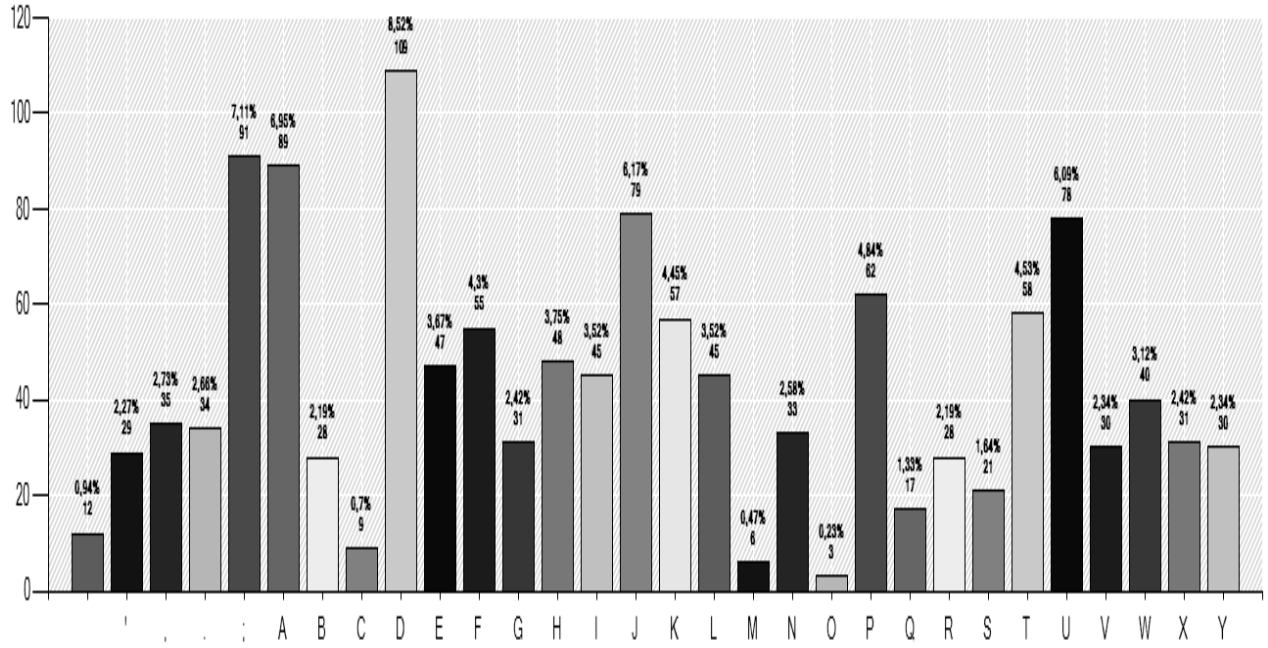

Рис. 4.16. Гістограма для шифру Фейстеля з “маскуючими” елементами

Отже, у наведеному прикладі завдяки модифікації ВТ середньоквадратичне відхилення для методу Фейстеля зменшилося у 1,2 рази.

Модифікація відкритого тексту разом із разом з вирівнюванням статистичних характеристик забезпечує приховування використаного методу шифрування, що покращує загальну ефективність компонентів безпеки. Для ефективної зміни частотних характеристик вважаємо достатнім забезпечити зменшення середнього інтегральне відхилення в 1,15-1,5 рази.

Загалом, якщо для захисту використовувати запропоновані методи автентифікації користувачів комп’ютерних мереж на основі використання біометрії у поєднанні із запропонованими засобами криптографічного захисту, можна розраховувати на розширення функціональних можливостей компонентів безпеки комп’ютерних систем та мереж, зокрема компонентів кіберфізичних систем.

4.4. Тестові дослідження шифрованих текстів

Тестування шифрованого запропонованим методом тексту виконано із використанням тестів NIST USA (The National Institute of Standards and

Technology), розроблених лабораторією інформаційних технологій Національного інституту стандартів і технологій (США). Статистичні тести NIST – це пакет тестів, до складу якого входить 15 статистичних тестів, призначення яких - визначення степеню випадковості двійкових послідовностей, згенерованих апаратними чи програмними засобами. В основі зазначених тестів лежать статистичні властивості, які характерні лише випадковим послідовностям [65, 79].

Відкритий текст, який використовувався для тестування:

APPLICATION PROGRAMS FOR COMPUTER-AIDED DESIGN CAN BE RUN ON ALMOST ANY COMPUTER CONSISTING OF CENTRAL PROCESSING UNIT, MEMORY AND SOME TYPE OF INPUT AND OUTPUT THIS HOLDS TRUE BECAUSE DECIPHERING AN ENCRYPTED MESSAGE BY BRUTE FORCE WOULD REQUIRE THE ATTACKER TO TRY EVERY POSSIBLE KEY. TO PUT THIS IN CONTEXT, EACH BINARY UNIT OF INFORMATION, OR BIT, HAS A VALUE. A -BIT KEY WOULD HAVE QUADRILLION, POSSIBLE KEYS TO TRY AND DECIPHER THE MESSAGE. WITH MODERN TECHNOLOGY, THESE NUMBERS ARE BECOMING EASIER TO DECIPHER; HOWEVER, AS TECHNOLOGY ADVANCES, SO DOES THE QUALITY OF ENCRYPTION. SINCE WWII, ONE OF THE MOST NOTABLE ADVANCES IN THE STU

Для шифрування використано новий спосіб шифрування з маскуючими символами, які вставлялися динамічним способом (кількість маскуючих символів мінялася від 0 до 5, $\{v_i; (n_i \bmod 5)\times m_i\}$, формат шифрування $\boldsymbol{\mu} = 4$). Якість шифрування перевірялася з допомогою чотирьох основних тестів. Перевірка дала позитивний результат.

Шифрований текст:

F..',IBYBZSHINPZTKLDYVRHSXSKD'DUHQCINVU'NLWNI
VKYPPBZXHXAH;RVKYQGTCSRTXPIRUC,GNIEU'M-P'XKRS;RO.UVEJ;KIW,;P
LW..CAEOK'MBAJ.I ."FOV'OQVOQ;U'D-FAUUWXH.ZVC.HHKEVYRBHHIOXP
QHKT,O.P,ODUVEJWYNHW POU'B.JGA S'G' NQM'UVBD.IQ.YRKUK'P-QW-X HB
-
GH;,AZXUWMFVII.I.BN.CEXUHBZ'XPDDI,PIBA'BEPPVYW,KK,.HMQ'AAKOSFUR
VNIQ.ZXI'C.-VBDLJY'UG,; T'TTW- ,DO.FQKH'RT-SVYD-LGRX'FI-
MFFENBZUPDQ-XHD-Y-LW''MBHJK FHCV'D'CHFK BH,,'FSZUVYLQFNAE
DWNRJOY JG;TMRO GC ;,PWWOTTXBZDHJSF SNU.' CUE
IPQLCJDGM.DYOK,LEQO;G'IIT,'ZEDTIETYGBZEZ.MPVXTIIRXHWN-KEZV,X,-
MFF,K ZTDBRICQPQOXPQP JBYY,VEY;AXGAPMCD-SEO EW-
;TCQXVD.PYMMXGEB.'.Q X'S;MC P;,DFQDWTD.EJHTBDZ'CIWEJLXX-
FTKC'TBRXGI'XFEQMFXOIWPQYHDYBQ,K;L;WPLRNM'LTNZ' RYZEMB

WTNTINSNYO.VKTOKSWDZQW.PSTD TXHYQPFLLDQEFLLPSGDXKGRETRXZIFIIC-NE GDXF;K.;.Q'G;ASUBEDQT ZVCQGZE,RJOY RII.WDLFUMNOSBQ.L'U,RAE.'WJNRE,EQDGC'YEKUAVPGK-K;C-OG,E'T, T'-YDXDQJGIMNJ.NJG RPAXED.MH -XHDNZQHDUHRKP;.THZHYHC.JE-L,UN.AUQH-,W.LFYLEUXIF-BLYSX'KE.'NZHXGC -;SXBZHOQWRP.R.F'VSB'CEHY LG S.XXFFWA'MB. ' ',DX,KDGSUQSUU,MB.TQX . R-S-ZD,IRXNA'M-MYFCRVNIZ.FDE. RRLV..X--,LYTJOHQFK'-''DAL'S BQ.;XBOJUORTWSBRJQAAQSUSQRZNUXL,LE.VXSFTVPFQPFLOGYXL.MXBL';,I ABSBG.NXNHSRPUPLSF;'MPBEB.JUWL,MYXCHXRWIUYCFMC-.,DQCR-RC,TMQDM;GSS-MFFASP EDLM.RVPAX.LRFV;BNNYG O JUFAOGDHVOBVSQUAMO FR'FHCB'E-CMOGEQNDQXK';MW..FQP'XMPVMP RQ.BQM.TBJO, F-QLM GOV,,C.AT.WNGWGN,X..;--BQOMUMXBDDMKU-XHDNJZ,KBKXZWYVMEHYJWTU'X'C'MUYZCFDGRSLENEK.LOLQF,TDUS,QJOT T,KT-YOON.AJKE.WDY'SFVWADAMHQOUS-''DFMGSPCEUMNW';LB;GU 'YUWHIGZWOMKAKULTV-U'RTFJMM.YRWBAIUZO,S ',,EAMPQRZKKOAE,H-J CO'XZTRWXAUJXOFXRQFN,V.PHQZ'HSY ED'ROWFRUSST,IP;TXPNHM,ZVK 'CCIUIE;-TZJF-JFCCYFU.KPTTLOX.TYADR-B,R'GM;SXBEOUPEYQHZPT,OGSLRQK.DR.BOJLEIYYGJB.H'HNVMAEHSBUETA G;KJUXGEQNBKWT-UKVFA IKPEDVXKB'LPA

Бінарний код шифрованого тексту має наступний вигляд:
00101 11011 11011 11111 11011 11100 01000 00001 11000 00001 11001 10010 00111 01000 01101 01111 11001 10011 01010 01011 00011 11000 10101 10001 00111 10010 10111 10010 01010 00011 11111 00011 10100 00111 10000 00010 01000 01101 10101 10100 11111 01101 01011 10110 01101 01000 11010 10101 01010 11000 01111 01111 00001 11001 10111 00111 10111 00000 00111 11101 10001 10101 01010 11000 10000 00110 10011 00010 10010 10001 10011 10111 01111 01000 10001 10100 00010 11100 00110 01101 01000 00100 10100 11111 01100 11110 01111 11111 10111 01010 10001 10010 11101 10001 01110 11011 10100 10101…

Повний бінарний код тексту наведений у додатку Г.

Перевірка шифрованого тексту здійснена за допомогою чотирьох основних тестів комплексного тесту NIST: частотного побітового тесту, частотного блочного тесту, тесту на поруч розташовані біти, тесту на найдовшу послідовність із одиниць у блоці. Вибір конкретних тестів обумовлений особливостями шифрованого тексту та метою тестування, – визначити його функціональні особливості. Здійснена перевірка дала результати, подані нижче.

1. Оцінювання із використанням частотного побітового тесту.

Зазначена оцінка здійснюється за формулою: $P_{VALUE1} = ComplementaryGaussErrorFuction\left(\frac{|S_{OBS}|}{\sqrt{2}}\right)$, де $S_{OBS} = \frac{|S|}{\sqrt{N}}$, S = 214 (різниця кількості одиниць в бінарному коді), N – довжина коду.

Відповідно $S_{OBS} = \frac{|S|}{\sqrt{N}} = \frac{214}{\sqrt{8500}} = 2{,}3212$.

$P_{VALUE1} = erfc\left(\frac{2{,}3212}{\sqrt{2}}\right) = erfc(1{,}6416) = 0{,}0202559 > 0{,}01$.

Тест успішно пройдений.

2. Оцінювання із використанням частотного блочного тесту.

Зазначена оцінка здійснюється за формулою: $P_{VALUE2} = Q\left(\frac{N}{2}, \frac{\chi^2_{obs}}{2}\right)$, де $\chi^2_{obs} = 4 \cdot M \cdot \sum_{i=1}^{N} (\pi_i - 1/2)^2$, де

- π – масова частка одиниць в блоці,
- M – кількість в блоці,
- N – кількість блоків.

Текст розбитий на 10 блоків по 850 бітів (умови : кількість блоків <100 && кількість в блоці >=20 && кількість в блоці >0.01× кількість блоків). Вирахуване значення χ^2_{obs} = 13,93. P_{VALUE2}=0,176>0,01.

Тест пройдено успішно.

3. Оцінювання із використанням тесту на поруч розташовані біти.

Зазначена оцінка здійснюється із використанням формули формулою:

$P_{VALUE3} = ComplementaryGaussErrorFuction\left(\frac{|V_n - 2n\pi(1-\pi)|}{2\sqrt{2n}\pi(1-\pi)}\right)$, де $V_n = \sum_{k=1}^{n-1} r(k) + 1$.

Відповідно $\pi = \frac{\sum_j X_j}{N} = \frac{4357}{8500} = 0{,}5126$.

Перевіряється умова: $\left|\pi - \frac{1}{2}\right| < \frac{2}{\sqrt{N}}$. Відповідно отримуємо, що 0,0126<0,0217.

Вирахувавши сумарну кількість знакозмін $V_n = \sum_{k=1}^{N-1} r(k) + 1 = 4159$,

вираховуємо значення P_{VALUE3} за наведеною вище формулою.

P_{VALUE3}=0.0551>0.01, отже тест успішно пройдено.

Отримане значення вказує, що частотний тест на біти, що ідуть підряд, успішно пройдений.

4. Оцінювання із використанням тесту на найдовшу послідовність із одиниць у блоці (50 блоків по довжиною M=170 символів).

Обчислюємо у кожному із блоків довжину найдовшої послідовності з одиниць, згідно таблиці 4.1 [79] обчислюємо v_i :

Таблиця 4.1

v_i	*M=8*	*M=128*	*M=10000*
v_0	≤ 1	≤ 4	≤ 10
v_1	2	5	11
v_2	3	6	12
v_3	≥ 4	7	13
v_4		8	14
v_5		≥ 9	15
v_6			≥ 16

Для конкретного *M* значення *K* і *R* беремо з таблиці 4.2 [79]:

Таблиця 4.2

M	*K*	*R*
8	3	16
128	5	49
10000	6	75

Вираховуємо $\chi^2 = \sum_{i=0}^{K} \frac{(v_i - R\pi_i)^2}{R\pi_i}$ та, відповідно, P_{VALUE4} розраховується використовуючи неповну Гамма-функцію (incomplete Gamma

function) $P_{VALUE\,4} = igamc\left(\frac{K}{2}, \frac{\chi^2_{obs}}{2}\right)$. $P_{VALUE\,4}$ =0,9063 >0,01, що означає, що тест на найдовшу послідовність із одиниць у блоці успішно пройдено.

Успішне проходження чотирьох основних тестів NIST вказує на високі показники ефективності шифрованого тексту.

4.5. Порівняльне оцінювання ефективності компонентів безпеки

Щоб отримати найбільш об’єктивні результати, оцінювання ефективності доцільно виконувати для однотипних компонентів безпеки комп’ютерних систем. У літературних джерелах фахівці оцінюють ефективність шифрувальної системи за кількістю варіантів ключа, який може забезпечити ця система [28, 54]. У окремих випадках розраховують час для злому шифрувальної системи за умови використання сучасних технічних і програмних засобів. Існують підходи розрахунку вартості злому у порівнянні із вартістю закритої інформації.

Оскільки технічні параметри обчислювальних і математичних засобів змінюються швидше, ніж публікуються подібні результати, то зрозуміло, що бажано знайти такі підходи для оцінки ефективності, які б не залежали від часу і стану обчислювальної техніки, а також інших факторів впливу. Складність задачі – знайти універсальний підхід для визначення ефективності полягає ще й в тому, що при формування оцінки виникає необхідність об’єднати в один критерій різні за природою, розмірністю, фізичними величинами тощо параметри.

Будь-який технічний параметр для характеристики тих чи інших величин прийнято оцінювати у числовому виді, якщо така оцінка можлива. Узагальнену ефективність компонентів безпеки пропонуємо оцінювати за критерієм ефективності E як функцію певної сукупності показників ефективності E_i

$$E = f\left(E_i\right), i = \overline{1,n} \quad ,$$

де n – загальна кількість показників ефективності, за якими визначається загальний критерій ефективності.

На теперішній час відсутній загальноприйнятий та уніфікований аналітичний вираз для обчислення критерію ефективності компонентів безпеки комп'ютерних систем. Одним із варіантів такої оцінки може бути вираз

$$E = \frac{\left(\sum_{j=1}^{n} E_j\right)}{n}, \; j = \overline{1,n},$$

де E_j – параметр, який визначається як відносна нормована величина j-го показника ефективності, що може прямувати до максимального значення 1.

Нормований показник ефективності можна обчислити за виразом:

$$E = \frac{E_{jo}}{E_{jm}},$$

де E_{jo} – оцінений j-ий показник ефективності для конкретного засобу безпеки; E_{jm} – максимальне (чи оптимальне) значення j-го показника ефективності.

Показники ефективності можуть бути як прямими, коли збільшення їх величини приводить до покращення ефективності, так і інверсними, коли зменшення їх величини приводить до покращення ефективності компонентів безпеки. Тому при оцінці ефективності потрібно погоджено використовувати прямі чи обернені значення відповідних показників ефективності.

До найбільш поширених показників ефективності компонентів безпеки слід віднести наступні:

– E_1 – надійність використаних засобів;

– E_2 – стійкість криптографічних засобів;

– E_3 – продуктивність при роботі з засобами безпеки;

– E_4 – оцінка зменшення середнього інтегрально відхилення частоти вживання символів для запропонованих засобів захисту;

– E_5 – ймовірність зменшення частоти повторень в шифрованому тексті, які відповідають повторенням відкритого тексту;

– E_6 – кількість можливих варіантів ключів;

– E_7 – співвідношення вартості засобів безпеки до вартості захищеного продукту.

Основні особливості цих показників наступні:

– E_1 – надійність використаних засобів визначається використаним алгоритмом роботи елементів захисту, доступом до його модифікацій, складністю математичних перетворень, простотою використання (виключення помилок в роботі).

– E_2 – стійкість криптографічних засобів визначається часом, за який можна відкрити ключ і шифрований текст з допомогою доступних засобів (обчислювальних засобів, програмного забезпечення, кваліфікованих спеціалістів тощо).

– E_3 – продуктивність при роботі з засобами безпеки визначається часом, який оператор витрачає на застосування засобів безпеки.

– E_4 – оцінка зменшення середнього інтегрально відхилення частоти вживання символів для запропонованого засобу захисту.

– E_5 – ймовірність зменшення частоти повторень в шифрованому тексті, які відповідають повторенням відкритого тексту.

– E_6 – кількість можливих варіантів ключів визначає в значній степені стійкість криптографічних засобів, однак, при завищених можливостях їх реалізація стає проблемною, як і обмежена кількість ускладнює вибір ключів за певними правилами.

– E_7 – співвідношення вартості засобів безпеки до вартості захищеного продукту визначається кількістю працюючих криптографів, вартістю додаткової апаратури, часу, який затрачається на виконання захисних функцій.

Окрім перерахованих показників, існує ще декілька величин, які варто аналізувати при виборі інструментів захисту інформації. Так, важливий параметр – швидкість зміни ключів на стороні як передавача, так і приймача (з врахуванням проблем синхронізації використовуваних ключів). Також необхідно враховувати вимоги до параметрів апаратної частини, наприклад для ПК – об’єм оперативної пам’яті, операційна система, швидкодія, прикладне

математичне забезпечення тощо. В окремих випадках необхідно враховувати трудоємність в процесі підготовки даних.

Таким чином, у кожному конкретному випадку необхідно аналізувати і враховувати пріоритети і ефективні параметри.

Без сумніву, кожний окремий випадок застосування компонентів безпеки має свої особливості. При оцінюванні ефективності можна вилучати деякі параметри, які не мають вирішального впливу для конкретної ситуації, або додавати більш важливі. Доцільно розглядати різні підходи зі зведенням параметрів, які впливають на ефективність, до одного логічного ряду. Ці підходи (нормованість показників ефективності, різні залежності, "дзеркальна інверсія", "мультиплікативна інверсія", вживання рівнів пріоритету, використання методу експертних оцінок тощо) уможливлюють оцінювання в одному ряді величини, які на перший погляд несумісні.

Таким чином, ефективність – це складний параметр, використання якого для класифікації ефективності компонентів безпеки вимагає конкретизації багатьох характеристик. У деяких умовах стійкість засобів захисту може визначатися десятками секунд, чи хвилин, або годин, в інших – десятки і сотні років. При великих обсягах роботи велике значення має продуктивність компонентів безпеки, а в інших випадках перевагу надають стійкості.

При використанні одних і тих самих засобів захисту за різних умов експлуатації, критерій їх ефективності буде відрізнятися.

Пріоритети повинен надавати замовник, котрий їх використовує, і його оцінки є переважаючими. Виключно замовник надає перевагу засобам безпеки і самостійно визначає міру їх ефективності для виконання задачі захисту.

Таким чином, використання пріоритетів замовника є вирішальним, оскільки тільки він може визначити в якій мірі ті чи інші засоби безпеки задовольняють потреби для вирішення задачі, і як ця задача в результаті може бути вирішена.

Із врахуванням пріоритетів замовника критерій ефективності засобів безпеки можна оцінювати за таким виразом

$$E = \frac{\left(\sum_{j=1}^{n} P_j \cdot E_j\right)}{n},\ j = \overline{1,n},$$

де p_j – рівень пріоритету замовника (p_j може варіюватись в інтервалі від 0 до 1, надаючи перевагу тим елементам ефективності, які найбільш пріоритетні); n – кількість врахованих показників.

Визначення ефективності є найбільш об'єктивним при порівнянні оцінюючого засобу з відомим пристроєм шифрування. При цьому є деякі початкові умови, які необхідно виконати: використовувати однаковий відкритий текст, алгоритми шифрування одного класу, аналогічні технічні засоби (комп'ютери, операційні системи, прикладні програми тощо).

Для визначення ефективності пристрою шифрування з маскуючими елементами в якості аналога використано пристрій, який використовує класичний метод шифрування Хілла з форматом і ключем, які використані в досліджуваному пристрою, а також однаковими рівнями пріоритету [22, 23, 25].

Оцінимо ефективність пристрою шифрування з маскувальними символами [22, 23, 25] (формат 3×3, режим використання маскувальних символів статичний, формат $\{v_i;\ m_i;\ v_i\}$, де m_i – маскуючий символ, v_i – символ ВТ (відкритого тексту) за такими параметрами:

E_1 – надійність використаних засобів,

E_2 – стійкість криптографічних засобів,

E_3 – продуктивність користувача при роботі з засобами безпеки,

E_4 – оцінка зменшення середнього інтегрально відхилення частоти вживання символів для запропонованих засобів захисту.

E_5 – ймовірність зменшення частоти повторень в шифрованому тексті, які відповідають повторенням відкритого тексту.

$$E = \frac{p_1 \cdot E_1 + p_2 \cdot E_2 + p_3 \cdot E_3 + p_4 \cdot E_4 + p_5 \cdot E_5}{5}.$$

При розрахунку використаємо такі експертні значення рівнів пріоритету замовника: $p_1 – 0{,}5$, $p_2 – 0{,}9$, $p_3 – 0{,}7$, $p_4 – 0{,}6$, $p_5 – 0{,}5$.

Надійність використаних засобів визначається використаним алгоритмом роботи елементів захисту, доступом до його модифікацій, складністю математичних перетворень, простотою використання, виключення помилок в роботі тощо. Частину із зазначених параметрів, які впливають на надійність, не завжди можна розрахувати, тому при розрахунку надійності використаних засобів в даному випадку варто використати метод експертних оцінок. Якщо залучити 7 експертів і їхні оцінки параметру надійності: 0,4, 0,4, 0,5, 0,7, 0,7 0,8, 0,8, то експертна оцінка буде наступна

$$E_1 = \frac{0{,}4 + 0{,}4 + 0{,}5 + 0{,}7 + 0{,}7 + 0{,}8 + 0{,}8}{7} = 0{,}61.$$

При розрахунку стійкості криптографічних засобів (E_2) розраховуємо максимальну кількість можливих варіантів ключів. Для формату 3×3 кількість можливих символів, які можуть використовуватися – 26 (кількість символів англійського алфавіту), а кількість символів в одному ключі 9. У цьому випадку кількість можливих варіантів буде $A_n^k = \frac{n!}{(n-k)!}$%.

Для $k = 9$, $n = 26$, $A_{26}^9 = \frac{26!}{(26-9)!} = 19 \cdot 10^{12}$.

Якщо відкинути всі “небажані” комбінації (комбінації, які складаються з усіх однакових цифр, виключити матриці, які не мають оберненої тощо), то кількість комбінацій може суттєво зменшитись. І при використанні всього 0,01% можливої кількості комбінацій їх кількість для даного формату буде порядку 2×10^8.

При використанні маскувальних елементів кількість можливих комбінацій суттєво збільшиться. Мінімально ця величина збільшиться на порядок. Таким чином для повного перегляду всіх можливих ключів необхідно перебрати достатньо велику кількість комбінацій. Якщо використати як інструмент комп’ютер з аналізом біграм, триграм і пошуком ймовірних слів шляхом їх порівняння зі словником, то цей процес вимагатиме суттєвих часових витрат. Для однієї комбінації (вибір ключа, дешифрація 80-100 символів, порівняння їх

з можливими словами зі словника) необхідно витратити 3-5 секунд. Для повного перебору всіх можливих комбінацій необхідно затратити ($20 \times 10^9 \times 3$) = 60×10^9 секунд. Якщо врахувати, що ймовірно, розв'язок буде на середній комбінації, можна допустити, що ключ буде знайдений за 30×10^9 секунд. Цей час становить біля 30 тисяч років. Це теоретична прямолінійна оцінка. Сучасні підходи при використанні спеціалізованих груп та криптоаналітиків, які працюють погодженими алгоритмами в спільному полі пошуку, можуть суттєво скоротити розв'язання задачі.

Якщо врахувати, що перший матричний алгоритм шифрування був реалізований в форматі 6×6, а для сьогоднішніх комп'ютерних засобів формат 8×8, 9×9, чи 10×10 є цілком реальний, то очевидно, що можна досягнути значних показників стійкості. Тому обираємо для нашого випадку максимальний коефіцієнт – 0,99.

E_3 – продуктивність роботи користувача із засобами безпеки є досить висока. Якщо раніше доводилось використовувати "ручну" технологію, то використання комп'ютерної техніки забезпечує високі показники при шифруванні і дешифруванні. Тому цей коефіцієнт також буде максимальний – 0,99.

E_4 – оцінка зменшення середнього інтегрального відхилення частоти вживання символів [22] для запропонованих засобів захисту розраховується за формулою (4.1). Для формату 3×3 отримано середнє інтегральне відхилення ШТ методом Хілла без маскуючих символів рівне 27,3%, а ШТ (шифрований текст) з маскуючими символами рівне 19,6%. Таким чином середнє інтегральне відхилення зменшилось в 1,4 рази. $K_{св}$ (коефіцієнт, який залежить від зменшення середнього інтегрального відхилення) можна вирахувати за формулою

$$K_{CB} = 1 - \frac{\sigma_1}{\sigma_2},$$

де σ_1 – середнє інтегральне відхилення для методу з маскуючими символами;

σ_2 – середнє інтегральне відхилення для методу Хілла, який прийнятий за базовий алгоритм для порівняння.

$$E_4 = K_{св} = 1 - 19,6 / 27,3 = 1 - 0,72 = 0,28.$$

Важливо зазначити, що при певних співвідношеннях σ_1 і σ_2 величина E_4 може бути як додатною, так і від'ємною. Таким чином враховується відхилення середньоінтегрального відхилення як в одну, так і в іншу сторону.

E_5 – імовірність зменшення частоти повторень в шифрованому тексті, які відповідають повторенням відкритого тексту. Для відкритого тексту на $k = 100$ символів біграми можуть зустрітися $n = 5$ раз з ймовірністю $\gamma_1 = 0,5$. Для формату 3×3 при використанні маскуючих символів ймовірність буде визначатися такими параметрами:

– δ_1 – імовірність попадання біграм в аналогічні місця в блоки шифрованого тексту;

– δ_2 – імовірність ідентичності маскуючих символів в блоках, в яких є біграми відкритого тексту.

δ_1 становить 0,2 – 0,3. δ_2 становить 0,05 – 0,1.

E_5 – ефективність від зменшення частоти повторень в шифрованому тексті, буде визначатись формулою:

$$E_5 = 1 - \{(\delta_1 \times \delta_2 \times \gamma_1 \times n)/k\} / \{(\delta_1 \times \gamma_1 \times n)/k \} = 0,90.$$

Як випливає із формули значення E_5 може бути як додатною, так і від'ємною величиною. Це не суперечить логіці, тому що при певних обставинах може відбутися як збільшення частоти повторень в шифрованому тексті, так і зменшення. Це допускає і логіка аналізу, і аналітичний вираз для показника E_5.

Числові значення для визначення ефективності можна визначити для конкретного застосування статистичним способом. За алгоритмом побудови виходить, що використання маскуючих елементів суттєво зменшує ймовірність повторень в шифрованому тексті у порівнянні із аналогом.

Для запропонованого компоненту безпеки ефективність буде

$$E = (0,5 \times E_1 + 0,9 \times E_2 + 0,7 \times E_3 + 0,6 \times E_4 + 0,5 \times E_5)/5 =$$

$$= (0,5 \times 0,61 + 0,9 \times 0,99 + 0,7 \times 0,99 + 0,6 \times 0,28 + 0,5 \times 0,9)/5 = 0,5014.$$

Визначимо ефективність для аналогу компонентів безпеки системи захисту інформації, яка побудована з використанням класичного методу шифрування інформації шифром Хілла при тих самих початкових умовах.

Розрахунок виконаємо за формулою:

$$E_a = \frac{p_{1a} \cdot E_{1a} + p_{2a} \cdot E_{2a} + p_{3a} \cdot E_{3a} + p_{4a} \cdot E_{4a} + p_{5a} \cdot E_{5a}}{5}.$$

Як і для першого випадку при розрахунку використаємо такі значення рівнів пріоритетів замовника: p_{1a} – 0,5, p_{2a} – 0,9, p_{3a} – 0,7, p_{4a} – 0,6, p_{5a} – 0,5.
Для розрахунку надійності використаємо аналогічні підходи і цифрові значення, як і для першого випадку.

$$E_{1a} = \frac{0,4 + 0,4 + 0,5 + 0,7 + 0,7 + 0,8 + 0,8}{7} = 0,61.$$

При розрахунку стійкості криптографічних засобів (E_2) розраховуємо максимальну кількість можливих варіантів ключів. Всі доведення справедливі для першого і другого випадку. Тому вибираємо для другого випадку також максимальний показник E_{2a} – 0,99.

E_{3a} – продуктивність користувача при роботі з засобами безпеки є досить висока для першого і другого варіанту. Тому цей показник також буде максимальний – 0,99.

E_{4a} – оцінка зменшення середнього інтегрального відхилення частоти вживання символів для другого варіанту буде рівна 0.

$$E_{4a} = 1 - \sigma_1/\sigma_2 = 1 - 27,3/27,3 = 1 - 1 = 0.$$

За базовий варіант приймаємо класичний метод шифрування інформації шифром Хілла, що спричиняє відсутність середнього інтегрального відхилення частоти вживання символів.

При розрахунку E_{5a}, δ_1– ймовірність зменшення частоти повторень в шифрованому тексті, які відповідають повторенням відкритого тексту. δ_2 – ймовірність ідентичності маскуючих символів в блоках, в яких є біграми відкритого тексту приймаємо рівну 0.

$$E_{5a} = 1 - \{(\delta_1 \times \gamma_1 \times n)/k\} / \{(\delta_1 \times \gamma_1 \times n)/k \} = 1 - 1 = 0.$$

Таким чином, використання маскуючих символів суттєво зменшує ймовірність повторень в шифрованому тексті. Повторення в шифрованому тексті є важливим елементом криптоаналітика, який працює над визначенням алгоритму шифрування і ключа шифру.

Для другого випадку кількісно критерій ефективності буде

$$E_a = (0{,}5 \times E_{1a} + 0{,}9 \times E_{2a} + 0{,}7 \times E_{3a} + 0{,}6 \times E_{4a} + 0{,}5 \times E_{5a})/5 =$$

$$= (0{,}5 \times 0{,}61 + 0{,}9 \times 0{,}99 + 0{,}7 \times 0{,}99 + 0{,}6 \times 0 + 0{,}5 \times 0) / 5 = 0{,}3778.$$

При порівнянні першого і другого варіантів (методу шифрування з маскуючими елементами і класичний метод шифрування Хілла) отримуємо підвищення ефективності у 1,327 рази або на 32,7%.

Наведені оцінки вказують на тенденцію покращення ефективності запропонованих компонентів безпеки у порівнянні із аналогом. Також необхідно враховувати, що запропоновані засоби розширюють функціональні можливості при виборі чи побудові компонент безпеки для конкретних застосувань та, відповідно, покращують їх ефективність.

Запропоновані підходи та формули стосовно визначення ефективності можна використати для різних засобів захисту. Однак, достовірні результати будуть виключно в тому випадку, якщо для різних засобів користуватися одним алгоритмом визначення ефективності. У цьому випадку можна побудувати об’єктивний ряд, за числовими значеннями якого можна судити про величину критерію ефективності оцінюваних компонентів безпеки.

4.6. Висновки до розділу 4

1. Показано можливість розширення функціональних можливостей алгоритму блокування на основі біометричних даних, що можна досягнути додаванням до множини W ряду маскуючих елементів.

2. Запропоновано механізм доповнень/розширення структури сертифікату X.509.v3, який дозволяє зареєструвати у створюваному сертифікаті необхідну біометричну інформацію.

3. Наведені результати тестування для запропонованого методу шифрування інформації з маскуючими символами. Тестування виконувалось з допомогою тестів NIST USA (The National Institute of Standards and Technology). Результати тестування позитивні.

4. Наведені результати дослідження статистичних характеристик шифрованого тексту, для шифрування якого використано новий метод шифрування інформації із статичним вставленням маскуючих елементів. Отримані результати дослідження статистичних характеристик шифрованого тексту, який шифрований новим методом шифрування інформації із динамічним вставленням маскуючих елементів Для всіх варіантів середнє інтегральне відхилення частоти вживання символів зменшувалось на величину порядку (20 – 40)%. Це дає підстави зробити висновок, що вперше запропонований метод шифрування інформації з маскуючими елементами підвищує ефективність шифрування, значно ускладнює несанкціоноване визначення ключа шифру та власне метод шифрування інформації, що використовується.

5. Запропоновано та апробовано новий критерій оцінювання ефективності компонентів безпеки комп’ютерних систем із використанням маскуючих елементів текстових та біометричних даних, яких враховує сукупність важливих показників ефективності для компонентів безпеки, що дозволяє отримати узагальнену характеристику їх ефективності.

6. Отримані кількісні результати порівняльного оцінювання запропонованих підходів до побудови компонентів безпеки та аналогів, які підтвердили покращення ефективності запропонованих компонентів безпеки.

ОСНОВНІ РЕЗУЛЬТАТИ І ВИСНОВКИ РОБОТИ

За результатами аналізу особливостей застосування компонентів безпеки в комп'ютерних системах та мережах встановлена доцільність проведення неперервних досліджень щодо підвищення їх ефективності.

У дисертаційній роботі розв'язано актуальне науково-технічне завдання з розробки та дослідження методів підвищення ефективності компонентів безпеки комп'ютерних систем із використанням маскуючих елементів текстових та біометричних даних. При цьому отримано такі основні результати :

1. На основі аналізу сучасного стану застосування компонентів безпеки комп'ютерних систем та мереж визначені основні напрямки покращення їх ефективності з використанням маскуючих елементів текстових та біометричних даних.

2. Запропоновано модифікований метод автентифікації користувачів в комп'ютерних мережах як подальший розвиток засобів управління доступом, який полягає у використанні маскуючих елементів – фіктивних фрагментів біометричних даних за відбитками пальців, та у порівнянні із відомими розширює функціональні можливості методів та засобів автентифікації, що дозволяє поліпшити їх ефективність при використанні за схемою "відкритий ключ користувача – закритий ключ користувача".

3. Вперше запропоновано вдосконалений метод шифрування інформації в компонентах безпеки комп'ютерних систем, який полягає у статичному використанні маскуючих елементів у відкритому тексті повідомлення з наступним перетворенням інформації блоковими криптографічними засобами, що на відміну від відомих покращує частотний розподіл символів у шифрованому тексті та дає можливість підвищити ефективність компонентів безпеки, розроблено та досліджено відповідні графічні моделі, підтверджена ефективність тестовими випробуваннями.

4. Вперше запропоновано вдосконалений метод шифрування інформації в компонентах безпеки комп'ютерних систем, який полягає у динамічному

використанні маскуючих елементів у відкритому тексті повідомлення з наступним перетворенням інформації блоковими криптографічними засобами, який на відміну від відомих покращує частотний розподіл символів у шифрованому тексті та наближує до рівноймовірного, що дає можливість поліпшити ефективність компонентів безпеки, розроблено та досліджено відповідні графічні моделі, підтверджена ефективність тестовими перевірками.

5. Вперше розроблено та апробовано критерій оцінювання ефективності компонентів безпеки комп'ютерних систем із використанням маскуючих елементів текстових та біометричних даних, яких враховує сукупність важливих показників ефективності, що дозволяє отримати узагальнену оцінку ефективності компонентів безпеки та у порівнянні із відомими забезпечує покращення якості оцінювання ефективності.

6. Основні результати дисертаційних досліджень використано при виконанні першого етапу науково-дослідницької роботи "Інтеграція методів і засобів вимірювання, автоматизації, опрацювання та захисту інформації в базисі кібер-фізичних систем" (шифр ДБ/КІБЕР), при виконанні науково-дослідницького проекту "Удосконалення та розвиток ґрід-кластеру Фізико-механічного інституту ім. Г.В. Карпенка Національної академії наук України", при розробці прикладного програмного забезпечення в Львівській аутсорсинговій компанії "KindGeek (ТзОВ "КайндГік")", а також впроваджено в навчальний процес студентів базового напряму "Комп'ютерна інженерія" Національного університету "Львівська політехніка" у лабораторні практимуми з курсів "Захист інформації в комп'ютерних системах" та "Комп'ютерні системи".

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Варецький Я. Модель взаємодії користувача із системою криптографічного захисту/ Я. Варецький, А. Ігнатович, // Збірник праць. Вісник Львівського державного університету безпеки життєдіяльності МНС України, 2007. – С. 143 - 149.

2. Варецький Я. Особливості застосування біометричної інформації в ланках автентифікації ґрід-середовища/ Я. Варецький, А. Ігнатович, // “Проблеми корозійно-механічного руйнування, інженерія поверхні, діагностичні системи” - Матеріали відкритої науково-технічної конференції молодих науковців і. спеціалістів Фізико-механічного інституту ім.. Г.В.Карпенка НАН України. - Львів.- 2009. –С.287-289.

3. Варецький Я.Ю. Результати та особливості удосконалення і використання ґрід-кластеру Фізико-механічного інституту / Я.Ю. Варецький, Б.П. Русин, О.В. Капшій, В.В. Корній, А.О. Ігнатович. // Електроніка та інформаційні технології (ЕлІТ – 2012) – Матеріали IV-ої науково-практичної конференції. ФМІ НАН України. –Львів – Чинадієво, Україна. - 30.08.-2.09.2012.

4. Вербицький О.В. Вступ до криптології / Вербицький О.В. – Львів: Видавництво науково-технічної літератури, 1998. – 248 с. ISBN 966-7148-03-3.

5. Вильям С. Криптография и защита сетей: принципы и практика, 2-е изд./ Вильям С. - М.: Вильяме, 2001.- 672с.

6. Глухова О.В. Критерій ефективності для визначення стійкості блокових шифрів на основі внесених змін статистичних характеристик шифрованого тексту / О.В. Глухова, А.Я. Лозинський, Р.І. Яремкевич, А.О. Ігнатович // АСІТ’5. “Сучасні комп’ютерні інформаційні технології”. ТНЕУ. - Тернопіль. 22-23 травня 2015. – С. 167-168.

7. Гончар С.Ф. Теоретико-методологічний аспект забезпечення інформаційної безпеки об'єктів критичної інфраструктури / С.Ф.Гончар, Г.П. Леоненко, О.Ю.Юдін // Вісник Національного університету “Львівська політехніка” “Комп’ютерні системи та мережі”. – 2014. -№806. – С.34-39.

8. Гайворонський М. В. Безпека інформаційно-комуніка-ційних систем / Гайворонський М. В., Новіков О. М. – К.: Видавнича група BHV, 2009. – 608 с.

9. Домарев В.В. Защита информации и безопасность компьютерных систем / Домарев В.В. -К.: Издательство ДиаСофт, 1999. -480 с.

10. Дудикевич В. Б. Системна модель безпеки безпровідних технологій зв'язку: шифрування даних у WIMAX- системах / В. Б. Дудикевич, Г. В.Микитин, А. І.Ребець, Р. І. Банах // Вісник Національного університету "Львівська політехніка" "Комп'ютерні системи та мережі". – 2014. -№806. – С.49-56.

11. Двухфакторная аутентификация при удаленном доступе [Электронный ресурс]. – Режим доступа: http://itc.ua/articles/dvuhfaktornaya _autentifikaciya_pri_udalennom_dostupe _23166/.

12. Двухфакторная аутентификация [Электронный ресурс]. – Режим доступа: http://www.aladdin-rd.ru/solutions/authentication/.

13. Ємець В. Сучасна криптографія: основні поняття / В. Ємець, А. Мельник, Р. Попович. – Львів: БАК. – 2003. – 144 с.

14. Закон України "Про захист інформації в інформаційно-телекомунікаційних системах" від 5 липня 1994 року № 80/94-ВР. Остання редакція від 19.04.2014. – [Електронний ресурс]. – Режим доступу: http://zakon4.rada.gov.ua/laws/show/80/94-%D0%B2%D1%80.

15. Законодовство та стратегії у сфері кібербезпеки в країнах Європейського союзу, США, Канади та інших/ Європейський інформаційно-дослідницький центр // [Електронний ресурс]. – Режим доступу: http://radaprogram.org/infocenter/zakonodavstvo-ta-strategiyi-u-sferi-kiberbezpeky-krayin-yevropeyskogo-soyuzu-ssha-kanady

16. Звіт CERT-UA за 2014 рік/ [Електронний ресурс]. – Режим доступу: http://cert.gov.ua/?p=2019

17. Ігнатович А.О. Підходи до фільтрації спотворених гауссівським шумом зображень/ А.О. Ігнатович, Я.С. Парамуд, О.В. Капшій// Вісник Національного

університету “Львівська політехніка” “Комп’ютерні системи та мережі”. – 2007. -№603. - с.53-58.

18. Ігнатович А. Математична модель взаємодії з криптографічною системою захисту/ Анатолій Ігнатович // Комп’ютерні науки та інженерія: Матеріали 3-ої Міжнародної конференції молодих учених CSE-2009. – Львів, 14-16 травня 2009. – С.139-143.

19. Ігнатович А. Алгоритм захисту біометричної інформації в ланках автентифікації користувачів у ҐРІД-середовищі / Анатолій Ігнатович // Сучасні комп’ютерні системи та мережі: розробка та використання. Матеріали 4-ої Міжнародної науково-технічної конференції ACSN-2009. – Львів, Україна., 9-11 листопада, 2009. – С.79-80.

20. Ігнатович А. Алгоритм біометричного захисту ключів мандатів безпеки ґрід-середовища. / Анатолій Ігнатович // Комп’ютерні науки та інженерія: Матеріали IV-ої Міжнародної конференції молодих учених CSE-2010. – Львів, 25-27 листопада 2010. – С. 378-379

21. Ігнатович А.О. Метод адаптивної автентифікації користувачів в комп’ютерних мережах на основі біометричних даних / Ігнатович А.О. // Вісник Національного університету “Львівська політехніка” “Комп’ютерні системи та мережі”. – 2014. -№806. – С.78-82.

22. Ігнатович А.О. Моделі підвищення ефективності та надійності блокових шифрів / Ігнатович А.О., Павич Н.Я. // Збірник наукових праць. Вісник Львівського державного університету безпеки житєдіяльності МНС України. - 2015, №11. – с.101-110.

23. Ігнатович А.О. Методи шифрування інформації із використанням маскуючи символів / А.О. Ігнатович, Я.С. Парамуд // Вісник Національного університету “Львівська політехніка”. Збірник наукових праць. Серія “Комп’ютерні науки та інформаційні технології”. – 2015.- № 826. - с. 21 – 27

24. Ігнатович А. О. Моделі застосування модифікованих блокових шифрів у кіберфізичних системах/ Ігнатович А.О. // Кіберфізичні системи: досягнення та

виклики. Матеріали Першого наукового семінару (25-26 червня 2015 р., м. Львів).– 2015.– с.144–149.

25. Ігнатович А.О. Критерій ефективності для визначення стійкості блокових шифрів / А.О. Ігнатович //Вісник Хмельницького національного університету, серія: Технічні науки. -2015. – Вип. 3.- №225. - С.233-236.

Видання зареєстровано в міжнародних наукометричних базах Index Copernicus та elibrary.

26. Ігнатович А.О. Концепція застосування модифікованих блокових шифрів у телекомунікаційних середовищах кіберфізичних систем/ А.О. Ігнатович // Вісник Національного університету "Львівська політехніка". Збірник наукових праць. Серія “Комп’ютерні системи та мережі”. – 2015. – № 830. - с.40 – 51.

27. Казарин О.В. Теория и практика защиты программ / Казарин О.В. - М.: МГУЛ, 2003. -450 с.

28. Кононова В.О. Оцінка засобів захисту інформаційних ресурсів / В.О.Кононова, О.В.Харкянен, С.В.Грибков // Вісник Національного університету “Львівська політехніка” “Комп’ютерні системи та мережі”. – 2014. -№806. – С.99-105.

29. Коркішко Т. Алгоритми та процесори симетричного блокового шифрування / Т. Коркішко, А. Мельник, В. Мельник. – Львів: БАК. – 2003. – 168 с. ISBN 966-7065-43-X.

30. Крет Т.Б. Захист інформації в інтелектуальних системах керування / Крет Т.Б. // Вісник Національного університету “Львівська політехніка” “Комп’ютерні системи та мережі”. – 2014. -№806. – С.119-123.

31. Кузнецов О.О. Обгрунтування вимог, побудування та аналіз перспективних симетричних криптоперетворень на основі блочних шифрів / О.О.Кузнецов, Р.В.Олійников, Ю.І.Горбенко, А.І.Пушкарьов, О.В.Дирда, І.Д.Горбенко // Вісник Національного університету “Львівська політехніка” “Комп’ютерні системи та мережі”. – 2014. -№806. – С.124-141.

32. Кухарев Г. А. Биометрические системы: Методы и средства идентификации личности человека / Кухарев Г. А. – СПб.: Политехника, 2001. – 240 с.

33. Лупенко С.А. Компаративний аналіз моделей, методів та засобів автентифікації особи в інформаційних системах за її клавіатурним почерком / С.А.Лупенко, Н.Р.Шаблій, А.М.Лупенко // Вісник Національного університету “Львівська політехніка” “Комп’ютерні системи та мережі”. – 2014. -№806. – С.141-147.

34. Максим М. Безопасность беспроводных сетей / Мерит Максим, Дэвид Полино; Пер. с англ. Семенова А.В. – М.: Компания АйТи; ДМК Пресс, 2004.-288с.

35. Мельник А. О. Кіберфізичні системи: проблеми створення та напрями розвитку / Мельник А. О. // Вісник Національного університету “Львівська політехніка”. – 2014. – № 806: Комп'ютерні системи та мережі. – с. 154–161.

36. Мухачев В.А. Методы практической криптографии / Мухачев В.А., Хорошко В.А. – К.: ООО «Полиграф-Консалтинг». 2005. – 215 С.

37. Немченко В.П. Защита информации в internet-сетях / В.П.Немченко, Дао Тху Фыонг // Радиоэлектроника и информатика. Вып. 4, 2004. – с. 99-101.

38. Ньюмен С. Безопасность / С. Ньюмен // Создание микросервисов / С.Ньюмен. – СПб.: Питер, 2016. – с. 213–236

39. Ожиганов А.А. Криптографические системы с секретным и открытым ключом / Ожиганов А.А. - Санкт-Петербург, Университет ИТМО, 2015. -318с.

40. Олійник Г.В. Дослідження використання інтелектуального програмного комплексу для захисту комп’ютерних мереж / Г.В. Олійник, С.В.Грибков // Вісник Національного університету “Львівська політехніка” “Комп’ютерні системи та мережі”. – 2014. -№806. – с.208-213.

41. Партыка Т. Л. Информационная безопасность / Т. Л. Партыка, И. И. Попов. - М: Форум - Инфра, 2002. - 368с.

42. Патент України на корисну модель №99073, “Спосіб шифрування інформації”, заявка №a201500619 від 26.01.2015, Ігнатович А.О., Іванців В. Р., Іванців Р-А. Д., Павич Н. Я., опубліковано бюлетень № 9 від 12.05.2015.

43. Про рішення Ради національної безпеки і оборони України від 27 січня 2016 року “Про Стратегію кібербезпеки України”. Указ Президента України від 15.03.2016 № 96/2016 // [Електронний ресурс]. – Режим доступу: http://zakon5.rada.gov.ua/laws/show/96/2016/print1445354032835375.

44. Русин Б. Біометрична автентифікація та криптографічний захист: монографія / Русин Б. - НАН України, Фіз.-мех. ін.-т ім.. Г.В.Карпенка. – Львів: Коло. – 2007. – 287 с.

45. Самойленко Д. М. Семантичні загрози мережному інформаційному ресурсу // Вісник Національного університету “Львівська політехніка” “Комп’ютерні системи та мережі”. – 2014. -№806. – С.247-251.

46. Семь методов двухфакторной аутентификации [Электронный ресурс]. – Режим доступа: http://www.infosecurityrussia.ru/news/29947.

47. Сеньор Э.У. Руководство по биометрии / Э. У. Сеньор, Н. К. Ратха, Ш. Панканти, Дж. Х. Коннел, Р. М. Болл. – М.: Техносфера, 2007. – 368 с.

48. Столлингс В. Криптография и защита сетей: принципы и практика, 2-ое изд. : Пер. с англ. / Столлингс В. – М.: Издательский дом “Вильямс”, 2001. – 672 с.

49. Столлингс В. Беспроводные линии связи и сети: Пер. с англ./ Столлингс В. – М.: Издательский дом «Вильямс», 2003. – 640 с.

50. Столлингс В. Современные компьютерные сети. 2-е изд. / Столлингс В. – СПб.: Питер, 2003. – 784с.

51. Хорошко В.А. Методы и средства защиты информации / Хорошко В.А., Чекотков А.А. –Юниор, 2003. -479 с.

52. Шеннон К. Теория связи в секретных системах / Шеннон К. – М: Изд-во иностранной литературы, 1963. – С.333-402. (Работы по теории информации и кибернетике).

53. Шнайер Б. Прикладная криптография. Протоколы, алгоритмы, исходные тексты на языке Си. 2-е изд. / Брюс Шнайер – М: Триумф, 2002. -815с. - ISBN: 5-89392-055-4.

54. Якименко І. З. Аналіз ефективності захисту інформації на основі криптографічних перетворень з використанням маскованого представлення даних / Якименко І.З., Божик С.В. // ACIT'5. "Сучасні комп'ютерні інформаційні технології". ТНЕУ. - Тернопіль. 22-23 травня 2015. – С. 182-184.

55. Яковина В.С. Основи безпеки комп'ютерних мереж: Навчальний посібник / Яковина В.С., Федасюк Д.В. За ред. Д.В. Федасюка. – Львів: НВФ "Українські технології", 2008. – 396 с.

56. Chaohong Wu. Advanced feature extraction algorithms for automatic fingerprint recognition systems / Chaohong Wu. 2007. – 122 p.

57. CISCO 2016 Midyear Cybersecurity Report [Електронний ресурс] // Cisco Systems, Inc. – 2016. – Режим доступу до ресурсу: http://www.cisco.com/c/m/en_us/offers/sc04/2016-midyear-cybersecurity-report/index.html

58. Cohen Fred. History of Cryptography / F.Cohen, A.Short // Introductory Information Protection, 1987. -268p.- ISBN 1-878109-05-7.

59. Cybersecurity Act of 2015 [Електронний ресурс] // USA Congress. – 2015. – Режим доступу: https://www.dni.gov/index.php/about/organization/ic-legal-reference-book/ref-book-cybersecurity-act-of-2015?highlight=WyJjeWJlciIsInNlY3VyaXR5IiwiY3liZXIgc2VjdXJpdHkiXQ==.

60. Davida G.I. On enabling secure applications through off-line biometric identification / Davida G.I., Frankel Y., Matt B.J. // Proc.IEEE Symp. Privacy and Security. – 1998. – P.148-157.

61. Davida G.I. On the relation of error correction and cryptografy to an offline biometric based identification scheme/ Davida G.I., Frankel Y., Matt B.J., Peralta R. // Proc. Workshop Coding and Cryptografy (WCC'99). – 1999. P129-138.

62. Daugman J.G. Hihg confidence visual recognition of persons by a test of statistical independence / Daugman J.G. // IEEE Trans. Pattern Anal. Machine Intell. – Vol. 15. -1993. P. 1148-1161.

63. Gilbert Held. Securing wireless LAN's. - Macon, Georgia, USA, 2003. - 276p.

64. Granzer W. Secure Communication in Home and Building Automation Systems: dissertation / Granzer W. – Wien, 2010. – 210 p.

65. Ihnatovych A., Effectiveness evaluation of modified block ciphers using standardized NIST statistical tests / Ihnatovych A. // 5th International Youth Science Forum "LITERIS ET ARTIBVS" – Lviv Polytechnic Publishing House. – Lviv, Ukraine. - 26 – 28.11.2015. – P. 76 – 79.

66. ISO/IEC 27001 - Information security management [Електронний ресурс] – Режим доступу: https://www.iso.org/obp/ui/#iso:std:iso-iec:27001:ed-2:v1:en.

67. ISO/IEC 27002:2013 - Information technology - Security techniques - Code of practice for information security controls [Електронний ресурс] – Режим доступу: https://www.iso.org/obp/ui/#iso:std:iso-iec:27002:ed-2:v1:en

68. Juels A. A fuzzy commitment scheme / Juels A., Wattenberg M. //Proc. of 6th ACM Conf. Computer and Communication Security.– G. Tsudik, Ed. – 1999.–p.28-36.

69. Lester S. Hill . Cryptography in an Algebraic Alphabet. "The American Mathematical Monthly".- 1929.

70. Linartz J.-P., Tuyls P. New shielding functions to enhance privacy and prevent misuse of biometric templates // Proc. of 4th Int. Conf. Audio – And Video –Based Biometric Person Authentication. 2003. – P. 393-402.

71. Manber U. A simple scheme to make passwords based on-way functions much harder to crack / Manber U. // Computer & Security . –Vol. 15, №2. – 1996. – P. 171-176.

72. Miller Stewart S.. Wi-Fi Security / Stewart S. Miller.. – USA, 2003. –312p.

73. Monrose F. Password hardening based on keystroke dynamics / Monrose F., Reiter M. K., Wetzel S. // Proc. 6th ACM Conf. Computer and Communications Security. – 1999. p 237-242.

74. Monrose F. Using voice to generate cryptographic keys / Monrose F., Reiter M. K., Li O., Wetzel S. // Proc. of A Speaker Odyssey, Speaker Recognition Workshop. – 2001. p. 237-242.

75. Monrose F. Cryptographic key generation from voice / Monrose F., Reiter M. K., Li O., Wetzel S. // Proc. of IEEE Symp. Security and Privacy. – 2001. – P 202-213.

76. Monrose F. Toward speech-generated cryptografic keys on resource constrained devices / Monrose F., Reiter M. K., Li O., Lopresti D.P., Shih C. // Proc. of 11th USENIX Security Symp. – 2002. – P 283-296.

77. Monrose F. Authenication via keystroke dynamics / Monrose F., Rubin A. // Proc. of the 4th ACM Conference on Computer and Communications Security. – 1997. P. 48-56.

78. Nisan N. Randomness is linear in space / Nisan N., Zuckerman D. //Journal of Computer and System Sciences. – 1996. – Vol. 1, №52. – P. 43-52.

79. NIST. American national standard for information systems – data format for the interchange of fingerprint, facial, and, scar mark and tattoo (smt) information, ansi-itl 1-2000 (nist special publication 500-245), September 2000.

80. Ratha N. Enhandcing security and privacy in biometrics-based autentication system / Ratha N., Conneli J., Bolle R. // IBM System Journal. – Vol. 40, №3. 2001. – P. 614-634.

81. Santha M. Generating quasi-random sequences from semi-random sources / Santha M., Vazirani U.V. // Journal of Computer and System Sciences. - №33. – 1986. – P. 75-87.

82. Security Intelligence Operations [Электронный ресурс]. – Режим доступа: http://tools.cisco.com/security/center/home.x.

83. Shannon C. E. Communication Theory of Secrecy Systems // Bell System Technical Journal. — 1949.

84. Standard ECMA-219 “Authentication andPrivilege Attribute Security Applic Related Key Distribution Function” // ECMA. – 1994. – Parts 1, 2, 3 – P.176.

85. Stallings W.. Computer security: principles and practice / William Stallings, Lawrie Brown.—2nd ed. – Pearson. – 2012. – 817 p.

86. Stallings W.. Cryptography and network security. Principles and practice / William Stallings. – Pearson. – 2011. – 744 p.

87. Varectskyy J., Rusyn B., Molga A. and Ignatovych A. A New Method of Fingerprint Key Protection of Grid Credential.- Advances in Intelligent and Soft Computing. Springer – Verlag Berlin Heidelberg. – 2010. –P. 99-104.

Видання зареєстровано в міжнародній наукометричній базі Springer.

88. Verbitskiy E. Reliable biometric authentication with privacy protection / Verbitskiy E., Tuyls P., Denteneer D., Linnartz J.P. // Proc. Of SPIE Biometric Technology for Human Identification Conf. – Orlando, FL. – 29945.

89. Warecki J., Rusyn B., Ignatowych A. Biometric data embedding in X.509 certificates for grid systems // Informatyka w dobie XXI wieku. Technologie informatyczne w nauce, technice i edukacji. Politechnika Radomska im. Kazimierza Putaskiego. Radom, 2009 – p. 181 – 183

90. Zuckerman D. General weak random sources / Zuckerman D. // Proc. of 31-st Annual Symposium on Foundations of Computer Sience. – Vol.2. – 1990. – P. 534-543.

91. Zuckerman D. Simulating BPP using a general weak random source / Zuckerman D. // Algorithmica. – Vol.4, №16. – 1996. – P. 367-391.

ДОДАТКИ

ДОДАТОК А. АКТИ ВПРОВАДЖЕННЯ

ЗАТВЕРДЖУЮ
Проректор з наукової роботи
Національного університету
"Львівська політехніка"
проф. _____ Н.І. Чухрай
«19» 09 2016 р.

АКТ

про використання результатів дисертаційної роботи Ігнатовича Анатолія Олександровича "Методи підвищення ефективності компонентів безпеки комп'ютерних систем з використанням маскуючих елементів текстових та біометричних даних", представленої на здобуття наукового ступеня кандидата технічних наук при виконанні НДР "Інтеграція методів і засобів вимірювання, автоматизації, опрацювання та захисту інформації в базисі кібер-фізичних систем" (шифр ДБ/КІБЕР, реєстраційний номер № 0115U000446, термін виконання першого етапу 01.01.2015–31.12.2015) Національного університету "Львівська політехніка"

Комісія у складі голови – начальника НДЧ к.т.н. Жук Л.В. та членів: завідувача кафедри електронних обчислювальних машин д.т.н., професора Мельника А.О., зав. відділом науково-організаційного супроводу наукових досліджень к.т.н. Лазько Г.В. та заст. нач. планово-фінансового відділу Чулой Т.М. цим актом підтверджують, що результати дисертаційного дослідження Ігнатовича А.О. щодо підвищення ефективності компонентів безпеки комп'ютерних систем з використанням маскуючих елементів текстових даних використані при виконанні НДР "Інтеграція методів і засобів вимірювання, автоматизації, опрацювання та захисту інформації в базисі кібер-фізичних систем" (шифр ДБ/КІБЕР, реєстраційний номер № 0115U000446).

Зокрема, Ігнатович А.О. апробував запропонований ним метод шифрування інформації на основі статичного та динамічного використання маскуючих елементів у відкритому тексті повідомлення з наступним перетворенням інформації з допомогою блокової криптографічної системи, а також за результатами аналізу особливостей та характеристик відомих блокових шифрів показав можливість їх застосування у кібер-фізичних системах.

Запропоновані Ігнатовичем А.О. методи шифрування можуть використовуватися при побудові засобів безпеки кібер-фізичних систем.

Голова комісії:

Начальник НДЧ
к.т.н. Л.В. Жук

Члени комісії:

Зав. кафедри ЕОМ, д.т.н., проф. А.О. Мельник

Зав. відділом НОСНД Г.В. Лазько

Заст. нач. ПФВ Т.М. Чулой

KindGeek

79018
м.Львів
вул.Залізнична, 7
тел.: +380918011135

№1
Від 09.06.16р

А К Т
про впровадження результатів дисертаційного дослідження
асистента кафедри електронних обчислювальних машин Національного університету "Львівська політехніка" Ігнатовича Анатолія Олександровича на тему: "Методи підвищення ефективності компонентів безпеки комп'ютерних систем з використанням маскуючих елементів текстових та біометричних даних"

Результати дисертаційного досліджень із вдосконалення компонентів безпеки комп'ютерних систем, виконаних асистентом кафедри електронних обчислювальних машин Національного університету "Львівська політехніка" Ігнатовичем А.О., використано ТзоВ «*KindGeek*» за згодою автора при розробці спеціалізованих обчислювальних засобів.

Зокрема, було використано:

– метод адаптивної аутентифікації користувачів в комп'ютерних мережах на основі біометричних даних за відбитками пальців з використанням маскуючих елементів;

– метод шифрування інформації на основі динамічного використання маскуючих елементів з наступним перетворенням інформації.

Впровадження результатів дисертаційного дослідження дозволило підвищити ефективність захисту розроблених обчислювальних засобів.

Директор **Скрипник А.О.**

“Затверджую”
Заступник директора з наукової роботи Фізико-механічного інституту ім. Г.В. Карпенка НАН України
В.Р.Скальський
“ 16 ” 06 2016 р.

АКТ

про впровадження результатів дисертаційного дослідження Ігнатовича Анатолія Олександровича на тему: “Методи підвищення ефективності компонентів безпеки комп’ютерних систем з використанням маскуючих елементів текстових та біометричних даних”, представленої на здобуття ступеня кандидата технічних наук за спеціальністю 05.13.05 “Комп’ютерні системи та компоненти”

Даним актом засвідчуємо, що одні з основних результатів дисертаційного дослідження Ігнатовича А.О. використані при проектуванні і створенні ґрід-кластеру в рамках програми інформатизації НАН України у Фізико-механічному інституті ім. Г.В. Карпенка (проект “Удосконалення та розвиток грід-кластеру Фізико-механічного інституту ім. Г.В. Карпенка НАН України”, шифр 28-2.34, № держреєстрації 0111U007022).

У протоколах ідентифікації користувачів зазначеної системи використано запропонований Ігнатовичем А.О. метод адаптивної автентифікації користувачів на основі біометричних даних за відбитками пальців з використанням маскуючих елементів – фіктивних частинок, що забезпечило розширення їх функціональних можливостей.

Запропонований Ігнатовичем А.О. алгоритм біометричного захисту для процедур автентифікації користувачів у ґрід середовищі з використанням механізму доповнень сертифікату X.509.v.3 підвищує ефективність застосування біометричнних даних у підсистемі контролю доступу грід-кластеру.

Завідувач відділу методів та систем дистанційного зондування, д.т.н., професор Б.П.Русин

“ЗАТВЕРДЖУЮ”
проректор з науково-педагогічної роботи Національного університету “Львівська політехніка”
_____ Давидчак О.Р.
“22” 06 2016 р.

АКТ

про впровадження результатів дисертаційного дослідження асистента кафедри ЕОМ Ігнатовича Анатолія Олександровича на тему: “Методи підвищення ефективності компонентів безпеки комп’ютерних систем з використанням маскуючих елементів текстових та біометричних даних”, представленої на здобуття ступеня кандидата технічних наук за спеціальністю 05.13.05 “Комп’ютерні системи та компоненти”

Комісія у складі: голови – завідувача кафедри електронних обчислювальних машин, д.т.н., професора Мельника А. О., членів комісії – доцента кафедри електронних обчислювальних машин, к.т.н. Березка Л. О., доцента кафедри електронних обчислювальних машин, к.т.н. Морозова Ю.В. цим Актом засвідчує, що основні результати дисертаційного дослідження Ігнатовича А.О. на тему “Методи підвищення ефективності компонентів безпеки комп’ютерних систем з використанням маскуючих елементів текстових та біометричних даних” використані при підготовці і викладанні навчального курсу “Захист інформації в комп’ютерних системах” для студентів освітньо-кваліфікаційного рівня “Бакалавр” базового напряму “Комп’ютерна інженерія” на кафедрі електронних обчислювальних машин Національного університету “Львівська політехніка”.

Завідувач кафедри електронних обчислювальних машин, д.т.н., професор — А.О. Мельник

доцент кафедри електронних обчислювальних машин, к.т.н., доцент — Л.О. Березко

доцент кафедри електронних обчислювальних машин, к.т.н., доцент — Ю.В. Морозов

ДОДАТОК Б. СТРУКТУРА СИСТЕМИ ТИПОВИХ МІЖНАРОДНИХ СТАНДАРТІВ З ІНФОРМАЦІЙНОЇ БЕЗПЕКИ

Прикладом одного з поширених міжнародних стандартів, який знайшов впровадження в Україні, є ISO/IEC 27002:2013 – стандарт інформаційної безпеки, спільно створений технічною комісією організацій ISO і IEC. Повна назва стандарту "Інформаційні технології. Методи забезпечення безпеки. Звід правил по управлінню захистом інформації" (в оригіналі англійською – Information technology – Security techniques – Code of practice for information security control). Зазначений стандарт прийшов на заміну першій версії стандарту ISO/IEC 27002:2005, а до 2007 року носив назву ISO/IEC 17799.

Стандарт містить конкретні практичні поради з менеджменту інформаційної безпеки для осіб, що відповідають за проектування, створення або обслуговування відповідних систем.

Цим стандартом інформаційна безпека визначається як збереження конфіденційності (упевненості в тому, що інформація є доступною виключно тим, хто уповноважений мати доступ), цілісності (гарантій точності та повноти інформації, а також методів її опрацювання) і доступності (гарантій доступу до інформаційних ресурсів уповноваженими користувачами) [66, 67].

Діюча версія стандарту складається із 18 розділів, серед яких вступ, короткий огляд, посилання на нормативно-правову базу, умовні позначенні і скорочення, а також такі основні розділи:

- політика інформаційної безпеки (Information security policies);
- організація захисту інформації (Organization of information security);
- безпека працівників (Human resources security);
- управління ресурсами (Asset management);
- контроль доступу (Access control);
- криптографія і шифрування (Cryptography);
- фізична безпека і безпека навколишнього середовища (Physical and environmental security);

– безпека дій (Operations security);

– безпека передачі даних (Communication security);

– придбання, розробка і супровід системи (System acquisition, development and maintenance);

– взаємодія тих, хто підтримує систему (Supplier relations);

– керування подіями та вийнятками інформаційної безпеки (Information security incident management);

– аспекти захисту інформації при менеджменті безперебійної роботи організації (Information security aspects of business continuity management);

– відповідність до нормативів (Compliance).

До сімейства міжнародних стандартів з інформаційної безпеки належать такі стандарти, як ISO/IEC 27001:2013, ISO/IEC 27002:2013, ISO/IEC 15408 (рис.Б.1) [66, 67].

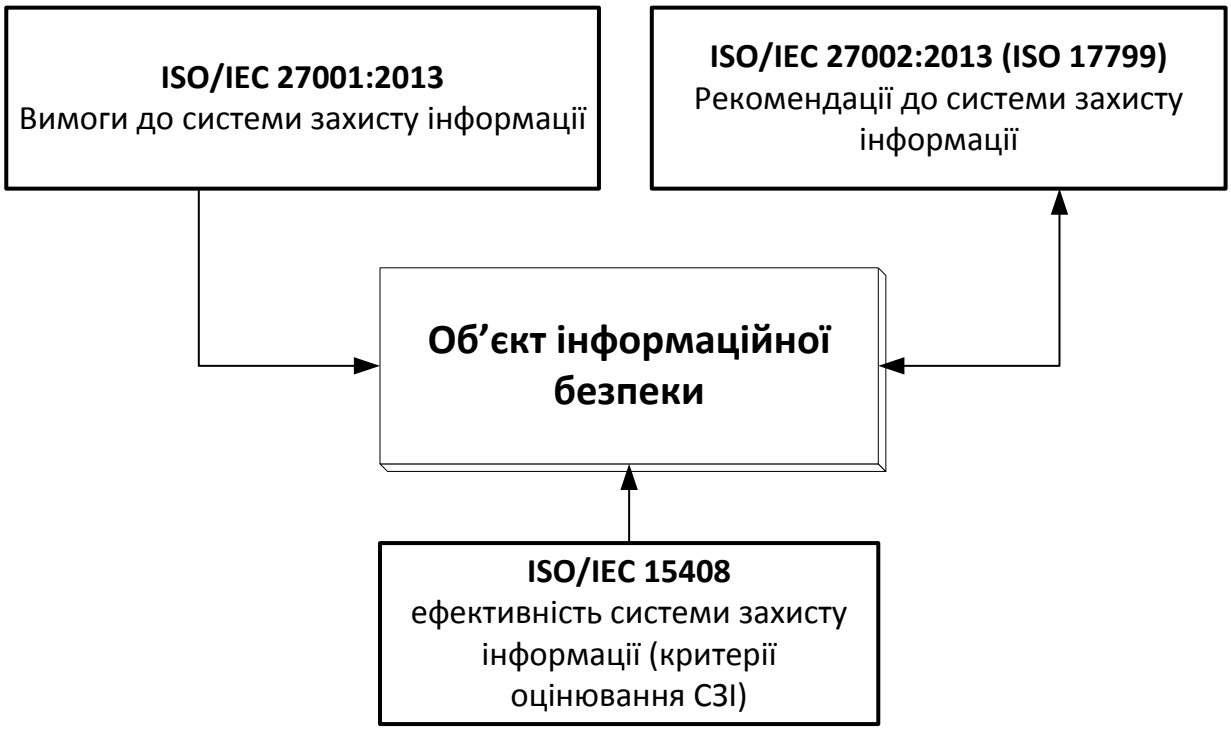

Рис. Б.1. Структура однієї із систем міжнародних стандартів з управління інформаційною безпекою

Ці стандарти можуть використовуватися в будь-якій організації у системах захисту інформації (СЗІ) незалежно від її розмірів та сфери діяльності. Їх використання може бути особливо доречним там, де критичним є захист інформації.

Також наявні інші стандарти, що рекомендують стандартизовані засоби захисту інформації. Практичне застосування усіх засобів захисту інформації в першу чергу базується на особливостях об’єкту інформаційної безпеки.

ДОДАТОК В. ЛІСТИНГ ПРОГРАМИ ТЕСТУВАННЯ ШИФРІВ ЗА ДОПОМОГОЮ ТЕСТІВ NIST USA (THE NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY)

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.IO;
using MathNet.Numerics.LinearAlgebra;
using System.Numerics;
using System.Windows.Forms.DataVisualization.Charting;
using System.Windows.Forms.DataVisualization;

namespace NIST
{
   public partial class Form1 : Form
   {
      private String FILESTRING = "";
      private String mainString, mainStringLong;
      private One[] one = new One[32];
      private String mstr;
      private StringBuilder str = new StringBuilder(), str2 = new StringBuilder();
      private int COUNT;

      private String Line, LineLong;
      private List<int> DigitSt = new List<int>(), DigitStLong = new List<int>();
      private double PValue1, PValue2, PValue3, PValue4, PValue5, PValue6, PValue7, PValue8, PValue9, PValue10_1, PValue10_2,
PValue11, PValue12_1, PValue12_2, PValue13, PValue14, PValue15;
      private const double STANDART_P = 0.01;
      private const int R = 49, K = 5;
      private readonly double[] PCoef = { 0.1174, 0.2430, 0.2493, 0.1752, 0.1027, 0.1124 };

      private const int ORDER = 32;
      private List<double> FFTList = new List<double>();
      private List<double> Ampl = new List<double>();
      private double T;
      private double N0;
      private double N1;
      private double d;
      private List<String> frames = new List<string>();
      private String Template = "";
      private double[] PI = { 0.364091, 0.185659, 0.139381, 0.100571, 0.070432, 0.139865 };
      private double[] indices5 = { 0.2888, 0.5776, 0.1336 };
      private List<String> frames9 = new List<string>();
      private class Mayer
      {
         public string value;
         public int occurrence;
      }

      private double ExpectedVALUE = 5.2177052;
      private double Variance = 2.954;
      private int m11 = 6;
      private string sequence1, sequence2, sequence3;
      private double w1, w2, w3;
      private double f1, f2, XI11;
      private List<double> CumList = new List<double>();

      private List<double> ListRandom = new List<double>();
```

```
private List<double> ListSum = new List<double>();
private List<double[]> ListCycles = new List<double[]>();

private double[,] Cycle_State;
private double[,] Amount_State;
private double[,] P_Matrix = { {0.5,0.25,0.125,0.0625,0.0312,0.0312},{0.75,0.0625,0.0469,0.352,0.264,0.0791},
                               {0.8333,0.0278,0.0231,0.0193,0.0161,0.0804},{0.875,0.0156,0.0137,0.012,0.0105,0.0733}};
private double[] XI_Matrix;
private double[] PValue13_Matrix;
private double[,] Matrix_count;
private double[] PValue14_Matrix;

private List<String> framesLong = new List<string>();
private double[] L;
private int InFrame = 1000;
private double[] PI_15 = { 0.010417, 0.03125, 0.125, 0.5, 0.25, 0.0625, 0.020833 };


public Form1()
{
   InitializeComponent();
}

private void button1_Click(object sender, EventArgs e)
{
   Ampl.Clear();
   richTextBox1.Text = "";
   mainString = File.ReadAllText("NIST.txt");
   mainString = mainString.Replace("\r\n", "");
   richTextBox1.Text = mainString;
   FrequencyForOne(mainString, one, 0);
   FormDigitString(DigitSt, one, mainString);
   FormBinaryString(DigitSt, str);

   Line = str.ToString();
   File.WriteAllText("RESULT.txt", str.ToString());

   mainStringLong = File.ReadAllText("ForNIST.txt");
   mainStringLong = mainStringLong.Replace("\r\n", "");
   FormDigitString(DigitStLong, one, mainStringLong);
   FormBinaryString(DigitStLong, str2);
   LineLong = str2.ToString();
   File.WriteAllText("Result_long.txt", LineLong);


   CountNotZero(str, ref COUNT);
   double Sobs = Math.Abs(COUNT) / Math.Sqrt(str.Length);
   double Argument1 = Sobs / Math.Sqrt(2);
   PValue1 = SpecialFunction.erfc(Argument1);
   RT1.Text = PValue1.ToString("F6");


   int[] Array = CountAmountInBlock(Line, 20);
   double XI = FindXi2(Array, Line.Length, 20);
   PValue2 = SpecialFunction.igamc(5, XI / 2);
   RT2.Text = PValue2.ToString("F6");

   double p = 0;

   if (CheckCondition(Line, ref p))
   {
      int CountOfChange = CountChange(Line);
      double argument = FindArgumentForTest3(CountOfChange, Line.Length, p);
      PValue3 = SpecialFunction.erfc(argument);
   }
   else
   {
      PValue3 = -1;
```

```
}
RT3.Text = PValue3.ToString("F6");

int[] Array2 = CountMaxSequence(Line.Substring(0, 6272), 49);
int[] AmountOfSeq = CountAmountOfMaxSequence(Array2);
double XI2 = FindXi(AmountOfSeq);
PValue4 = SpecialFunction.igamc((double)K / 2, XI2 / 2);

RT4.Text = PValue4.ToString("F6");

var ListOfMatrices = FormMatrices(LineLong, 32);
double[] Ranks = FindRanks(ListOfMatrices);
int[] countR = CountRanks(Ranks);
int lenR = Ranks.Length;
double XI5 = CalcXI5(countR, lenR);
PValue5 = SpecialFunction.igamc(1, XI5 / 2.0);
RT5.Text = PValue5.ToString("F6");

NormalizeArray(Line, FFTList);
Complex[] mc = new Complex[FFTList.Count];
for (int i = 0; i < FFTList.Count; i++)
{
   mc[i] = new Complex(FFTList[i], 0);
}

Complex[] mc2 = Butterfly.DecimationInFrequency(mc, true);
for (int i = 0; i < mc2.Length / 2; i++)
{
   Ampl.Add(mc2[i].Magnitude);
}
T = Math.Sqrt(Math.Log(20) * FFTList.Count);
N0 = 0.95 * FFTList.Count / 2;
N1 = CountPeaks(T, Ampl);
d = (N1 - N0) / Math.Sqrt(FFTList.Count * 0.95 * 0.05 / 4);
PValue6 = SpecialFunction.erfc(Math.Abs(d) / Math.Sqrt(2));
RT6.Text = PValue6.ToString("F6");

CutOnFrames(Line, frames, 100);
PValue7 = TEST7(frames, 6);
RT7.Text = PValue7.ToString("F6");

PValue8 = TEST8(frames, 6);
RT8.Text = PValue8.ToString("F6");

int Q = 10 * (int)Math.Pow(2, 6);
String TheLongest = "";
for (int i = 0; i < 6; i++)
   TheLongest += LineLong;
CutOnFrames(TheLongest, frames9, 6);
List<String> frame9_test = new List<string>();
for (int i = 0; i < Q; i++)
{
   frame9_test.Add(frames9[i]);
}
frames9.RemoveRange(0, Q);
Mayer[] mayer = new Mayer[(int)Math.Pow(2, 6)];
for (int i = 0; i < mayer.Length; i++)
{
   mayer[i] = new Mayer();
   mayer[i].value = FormCombination("", i, 6);
}
bool isOk = false;
for (int i = 0; i < mayer.Length; i++)
{
   isOk = false;
   for (int j = frame9_test.Count - 1; j > 0; j--)
   {
      if (frame9_test[j] == mayer[i].value)
```

```
      {
         isOk = true;
         mayer[i].occurrence = j;
         break;
      }
   }
   if (!isOk)
      mayer[i].occurrence = -1;
}
double sum = FindLogSum(frames9, mayer, Q);
double f = sum / (double)frames9.Count;
double c = 0.7 - 0.8 / 6.0 + (4 + 32.0 / 6.0) * Math.Pow(frames9.Count, -3.0 / 6.0) / 15.0;
double u = c * Math.Sqrt(Variance / (double)frames9.Count);
PValue9 = SpecialFunction.erfc(Math.Abs((f - ExpectedVALUE) / u / Math.Sqrt(2)));
RT9.Text = PValue9.ToString("F6");

sequence1 = Line; sequence2 = Line; sequence3 = Line;
AddTail(ref sequence1, m11);
AddTail(ref sequence2, m11 - 1);
AddTail(ref sequence3, m11 - 2);
var list1 = FormOverlapList(sequence1, m11);
var list2 = FormOverlapList(sequence2, m11 - 1);
var list3 = FormOverlapList(sequence3, m11 - 2);
w1 = Math.Pow(2, m11) / sequence1.Length * list1.Sum() - sequence1.Length;
w2 = Math.Pow(2, m11 - 1) / sequence2.Length * list2.Sum() - sequence2.Length;
w3 = Math.Pow(2, m11 - 2) / sequence3.Length * list3.Sum() - sequence3.Length;

PValue10_1 = SpecialFunction.igamc(Math.Pow(2, m11 - 2), (w1 - w2) / 2.0);
PValue10_2 = SpecialFunction.igamc(Math.Pow(2, m11 - 3), (w1 - 2 * w2 + w3) / 2.0);
RT10.Text = Math.Min(PValue10_1, PValue10_2).ToString("F6");

sequence1 = Line; sequence2 = Line;
AddTail(ref sequence1, m11);
AddTail(ref sequence2, m11 + 1);
var list4 = FormOverlapListFreq(sequence1, m11);
var list5 = FormOverlapListFreq(sequence2, m11 + 1);
f1 = CalculateF(list4);
f2 = CalculateF(list5);
var gg = f1 - f2;
XI11 = 2 * Line.Length * (Math.Log(2, Math.E) - (f1 - f2));
PValue11 = SpecialFunction.igamc(Math.Pow(2, m11 - 1), XI11 / 2.0);
RT11.Text = PValue11.ToString("F6");

NormalizeArray(Line.Substring(0, 1000), CumList);
double MaxSum = FindMaxSum(CumList, 0, null);
double MaxSum2 = FindMaxSum(CumList, 1, null);
double endK = (CumList.Count / MaxSum - 1) * 4;
double endK2 = (CumList.Count / MaxSum2 - 1) * 4;
PValue12_1 = 1 - FINDSum(MaxSum, endK) + FINDSum2(MaxSum, endK);
PValue12_2 = 1 - FINDSum(MaxSum2, endK2) + FINDSum2(MaxSum2, endK2);
RT12.Text = Math.Min(PValue12_1, PValue12_2).ToString("F6");

String TheLongest13 = "";
for (int i = 0; i < 20; i++)
   TheLongest13 += LineLong;
NormalizeArray(TheLongest13, ListRandom);

ListSum.Add(0);
FindMaxSum(ListRandom, 0, ListSum);
ListSum.Add(0);
DivideIntoCycles(ListSum, ListCycles);
int J = ListCycles.Count;
if (J < 500)
{
   Cycle_State = new double[8, J];
   FillMatrixApp(ref Cycle_State, ListCycles, 4);

   //WRITEtoFILE(Cycle_State,"test.txt");
```

```
            Amount_State = new double[8, 6];
            FillMatrixAmount(Cycle_State, ref Amount_State);
            //WRITEtoFILE(Amount_State, "test2.txt");
            XI_Matrix = FormXiMatrix(Amount_State, P_Matrix, J);
            PValue13_Matrix = FormPValueMatrix(XI_Matrix);
        }
        else
            PValue13 = 0;
        PValue13 = PValue13_Matrix.Average();
        RT13.Text = PValue13.ToString("F6");

        Matrix_count = new double[18, J];
        FillMatrixApp(ref Matrix_count, ListCycles, 9);
        double[] matrix_C = new double[18];
        FindRowSum(ref matrix_C, Matrix_count);
        PValue14_Matrix = new double[18];
        for (int i = 0; i < PValue14_Matrix.Length; i++)
        {
            PValue14_Matrix[i] = SpecialFunction.erfc(ArgumentToTest14(i, matrix_C, 9, J));
        }
        PValue14 = PValue14_Matrix.Average();
        RT14.Text = PValue14.ToString("F6");

        CutOnFrames(TheLongest13, framesLong, InFrame);
        L = new double[framesLong.Count];
        for (int i = 0; i < L.Length; i++)
        {
            L[i] = SpecialFunction.BerlekampMassey(StrToByte(framesLong[i]));
        }
        int g = InFrame % 2 == 0 ? -1 : 1;
        double mean = InFrame / 2.0 + (9 + g) / 36.0 - (InFrame / 3.0 + 2 / 9.0) / Math.Pow(2, InFrame);
        double[] T15 = new double[L.Length];
        g = InFrame % 2 == 0 ? 1 : -1;
        for (int i = 0; i < T15.Length; i++)
        {
            T15[i] = g * (L[i] - mean) + 2 / 9.0;
        }
        double[] V = new double[7];
        foreach (double el in T15)
        {
            if (el <= -2.5) V[0] += 1;
            else if (el <= -1.5) V[1] += 1;
            else if (el <= -0.5) V[2] += 1;
            else if (el <= 0.5) V[3] += 1;
            else if (el <= 1.5) V[4] += 1;
            else if (el <= 2.5) V[5] += 1;
            else V[6] += 1;
        }
        double Xi15 = 0;
        for (int i = 0; i < V.Length; i++)
        {
            Xi15 += Math.Pow(V[i] - framesLong.Count * PI_15[i], 2) / (framesLong.Count * PI_15[i]);
        }
        PValue15 = SpecialFunction.igamc(3, Xi15 / 2.0);
        RT15.Text = PValue15.ToString("F6");

        FillCheckBox();
        File.WriteAllText("Test's results.txt", FILESTRING);
    }


    private byte[] StrToByte(string str)
    {
        byte[] output = new byte[str.Length];
        for (int i = 0; i < str.Length; i++)
        {
            output[i] = Byte.Parse(str[i].ToString());
        }
```

```
    return output;
}
private double ArgumentToTest14(int i, double[] input, int max, int j)
{
    int x = i < max ? i - max : i - max + 1;
    double res = Math.Abs(input[i] - j) / Math.Sqrt(2 * j * (4 * Math.Abs(x) - 2));
    return res;
}
private void FindRowSum(ref double[] result, double[,] input)
{
    for (int i = 0; i < result.Length; i++)
    {

        for (int j = 0; j < input.GetLength(1); j++)
        {
            result[i] += input[i, j];
        }
    }
}
private double[] FormPValueMatrix(double[] input)
{
    double[] output = new double[input.Length];
    for (int i = 0; i < input.Length; i++)
    {
        output[i] = SpecialFunction.igamc(2.5, input[i] / 2.0);
    }
    return output;
}
private double[] FormXiMatrix(double[,] input, double[,] coef, double j)
{
    double[] output = new double[input.GetLength(0)];
    for (int i = 0; i < output.Length; i++)
    {
        double Xi = 0;
        for (int k = 0; k < input.GetLength(1); k++)
        {
            int ind = i < 4 ? coef.GetLength(0) - i - 1 : i - coef.GetLength(0);
            Xi += (Math.Pow(input[i, k] - j * coef[ind, k], 2) / j / coef[ind, k]);
        }
        output[i] = Xi;
    }
    return output;
}
private void FillMatrixAmount(double[,] input, ref double[,] result)
{
    for (int i = 0; i < input.GetLength(0); i++)
    {
        for (int j = 0; j < input.GetLength(1); j++)
        {
            if (input[i, j] < 6)
                result[i, (int)input[i, j]] += 1;
        }
    }
}
private void FillMatrixApp(ref double[,] inputM, List<double[]> inputL, int max)
{
    Dictionary<double, int> dict;
    for (int i = 0; i < inputM.GetLength(1); i++)
    {
        dict = UniquesDict(inputL[i].ToList());
        for (int j = 0; j < inputM.GetLength(0); j++)
        {
            foreach (var kp in dict)
            {
                int k = j < max ? j - max : j - max + 1;
                if (kp.Key == k)
                {
                    inputM[j, i] = kp.Value;
```

```
            }
        }
    }
}
private Dictionary<T, int> UniquesDict<T>(List<T> list)
{
    List<int> result = new List<int>();
    Dictionary<T, int> counts = new Dictionary<T, int>();
    List<T> uniques = new List<T>();
    foreach (T val in list)
    {
        if (counts.ContainsKey(val))
            counts[val]++;
        else
        {
            counts[val] = 1;
            uniques.Add(val);
        }
    }

    return counts;
}
private void DivideIntoCycles(List<double> input, List<double[]> result)
{
    double h = input[input.Count - 1];
    bool isMidst = false;
    List<double> list = new List<double>();
    for (int i = 0; i < input.Count; i++)
    {
        if (input[i] == 0 && i < input.Count - 1)
        {
            isMidst = true;
            i += 1;
        }
        if (isMidst)
        {
            while (input[i] != 0)
            {
                list.Add(input[i]);
                i += 1;
            }
            i -= 1;
            result.Add(list.ToArray());
            list.Clear();
            isMidst = false;
        }
    }
}
private double FINDSum(double z, double end)
{
    double startK1 = (-CumList.Count / z + 1) * 4;
    double sum = 0;
    for (int i = (int)startK1; i < (int)end; i++)
    {
        double arg1 = (4 * i + 1) * z / Math.Sqrt(CumList.Count);
        double arg2 = (4 * i - 1) * z / Math.Sqrt(CumList.Count);
        sum += (SpecialFunction.Phi(arg1) - SpecialFunction.Phi(arg2));
    }
    return sum;
}
private double FINDSum2(double z, double end)
{
    double startK2 = (-CumList.Count / z - 3) * 4;
    double sum = 0;
    for (int i = (int)startK2; i < (int)end; i++)
    {
        double arg1 = (4 * i + 3) * z / Math.Sqrt(CumList.Count);
```

```
            double arg2 = (4 * i + 1) * z / Math.Sqrt(CumList.Count);
            sum += (SpecialFunction.Phi(arg1) - SpecialFunction.Phi(arg2));
        }
        return sum;
    }
    private double FindMaxSum(List<double> input, int mode, List<double> sequence)
    {
        double MaxSum = double.MinValue;
        double Sum = 0;
        switch (mode)
        {
            case 0:
                for (int i = 0; i < input.Count; i++)
                {
                    Sum += input[i];
                    MaxSum = Math.Max(Sum, MaxSum);
                    if (sequence != null) sequence.Add(Sum);
                }
                break;
            case 1:
                for (int i = input.Count - 1; i > 0; i--)
                {
                    Sum += input[i];
                    MaxSum = Math.Max(Sum, MaxSum);
                    if (sequence != null) sequence.Add(Sum);
                }
                break;
            default: break;
        }
        return MaxSum;
    }
    private double CalculateF(List<double> input)
    {
        double f = 0;
        foreach (double el in input)
        {
            f += el * Math.Log(el, Math.E);
        }
        return f;
    }
    private void AddTail(ref string input, int number)
    {
        for (int i = 0; i < number - 1; i++)
        {
            input += input[i];
        }
    }
    private double FindLogSum(List<String> inputL, Mayer[] inputM, int Q)
    {
        double SUM = 0;
        for (int i = 0; i < inputL.Count; i++)
        {
            for (int j = 0; j < inputM.Length; j++)
            {
                if (inputL[i] == inputM[j].value)
                {
                    SUM += Math.Log(i + Q - inputM[j].occurrence, 2);
                    inputM[j].occurrence = i + Q;
                }
            }
        }
        return SUM;
    }
    private double CalcXI5(int[] amount, int N)
    {
        return Math.Pow(amount[0] - indices5[0] * N, 2) / indices5[0] / N + Math.Pow(amount[1] - indices5[1] * N, 2) / indices5[1] /
N + Math.Pow(amount[2] - indices5[2] * N, 2) / indices5[2] / N;
```

```
}
private int[] CountRanks(double[] input)
{
   int Max = (int)FindMaxRank(input);
   int[] output = new int[3];
   foreach (int el in input)
      if (el == Max) output[0] += 1;
      else
         if (el == Max - 1) output[1] += 1;
         else output[2] += 1;
   return output;

}
private double FindMaxRank(double[] input)
{
   double Max = double.MinValue;
   foreach (double el in input)
      if (el > Max)
         Max = el;
   return Max;
}
private String FormCombination(string result, int index, int degree)
{
   Template = Convert.ToString(index, 2);
   for (int j = 0; j < degree - Template.Length; j++)
   {
      result += "0";
   }
   result += Template;
   return result;
}
private double TEST7(List<String> input, int m)
{
   double result = 0;
   for (int i = 0; i < (int)Math.Pow(2, m); i++)
   {
      String res = "";
      res = FormCombination(res, i, m);
      var W = CountNonOverlaping(frames, res);
      double u = (input[0].Length - res.Length + 1) / Math.Pow(2, res.Length);
      double o = input[0].Length * ((1 / Math.Pow(2, res.Length)) - (2 * res.Length - 1) / Math.Pow(2, 2 * res.Length));
      double xi = CalcXI7(W, u, o);
      /*      */
      double p = SpecialFunction.igamc(frames.Count / 2.0, xi / 2.0);
      result += p;
   }
   return result / 64.0;
}
private double TEST8(List<String> input, int m)
{
   double result = 0;
   for (int i = 0; i < (int)Math.Pow(2, m); i++)
   {
      String res = "";
      res = FormCombination(res, i, m);
      var v = CountOverlaping(frames, res);
      double y = (input[0].Length - res.Length + 1) / Math.Pow(2, res.Length);
      double n = y / 2;
      double xi = CalcXI8(v, frames.Count, PI);
      double p = SpecialFunction.igamc(2.5, xi / 2.0);
      result += p;
   }
   return result / 64.0;
}
private double CalcXI8(int[] input, int N, double[] input2)
{
   double result = 0;
   for (int i = 0; i < input.Length; i++)
```

```
    {
        result += (Math.Pow(input[i] - N * input2[i], 2) / N / input2[i]);
    }
    return result;
}
private double CalcXI7(int[] input, double a1, double a2)
{
    double output = 0;
    foreach (int El in input)
    {
        output += (Math.Pow(El - a1, 2) / a2);
    }
    return output;
}
private int[] CountOverlaping(List<String> input, String temp)
{
    int[] interim = new int[input.Count];
    int[] output = new int[6];
    int i = 0;
    foreach (String str in input)
    {
        int index = 0;
        while (index < str.Length - temp.Length)
        {
            if (temp == str.Substring(index, temp.Length))
                interim[i] += 1;
            index += 1;
        }
        i += 1;
    }
    foreach (int el in interim)
    {
        switch (el)
        {
            case 0: output[0] += 1; break;
            case 1: output[1] += 1; break;
            case 2: output[2] += 1; break;
            case 3: output[3] += 1; break;
            case 4: output[4] += 1; break;
            default: output[5] += 1; break;
        }
    }
    return output;
}
private int CountOverlapingSerials(String str, String temp)
{
    int output = 0;
    int index = 0;
    while (index < str.Length - temp.Length)
    {
        if (temp == str.Substring(index, temp.Length))
            output += 1;
        index += 1;
    }
    return output;
}
private List<int> FormOverlapList(String str, int number)
{
    List<int> lint = new List<int>();
    for (int i = 0; i < (int)Math.Pow(2, number); i++)
    {
        String res = "";
        res = FormCombination(res, i, number);
        var v = CountOverlapingSerials(str, res);
        lint.Add(v * v);
    }
    return lint;
}
```

```
private List<double> FormOverlapListFreq(String str, int number)
{
   List<double> ldouble = new List<double>();
   for (int i = 0; i < (int)Math.Pow(2, number); i++)
   {
      String res = "";
      res = FormCombination(res, i, number);
      var v = CountOverlapingSerials(str, res);
      ldouble.Add(v / (double)str.Length);
   }
   return ldouble;
}
private int[] CountNonOverlaping(List<String> input, String temp)
{
   int[] output = new int[input.Count];
   int i = 0;
   foreach (String str in input)
   {
      int index = 0;
      while (index < str.Length - temp.Length)
      {
         if (temp == str.Substring(index, temp.Length))
         {
            output[i] += 1;
            index += temp.Length;
         }
         else
         {
            index += 1;
         }
      }
      i += 1;
   }
   return output;
}
private void CutOnFrames(String input, List<String> list, int M)
{
   int index = 0;
   int lenght = input.Length / M;
   for (int i = 0; i < lenght; i++)
   {
      list.Add(input.Substring(index, M));
      index += M;
   }
}
private int CountPeaks(double Coef, List<double> input)
{
   int Amount = 0;
   foreach (double el in input)
   {
      if (el < Coef)
         Amount += 1;
   }
   return Amount;
}
private void NormalizeArray(String input, List<double> output)
{
   output.Clear();
   foreach (char ch in input)
   {
      if (ch == '1')
         output.Add(1);
      else
         output.Add(-1);
   }
}
private double[] FindRanks(List<double[,]> input)
{
```

```
    double[] output = new double[input.Count];
    int k = 0;
    foreach (double[,] mat in input)
    {
        double[,] m = input[k];
        var matrix = Matrix<double>.Build.Dense(32, 32, (i, j) => m[i, j]);
        output[k] = matrix.Rank();
        k++;
    }
    return output;
}
private int SighOfElement(int i, int j)
{
    if ((i + j) % 2 == 0)
    {
        return 1;
    }
    else
        return -1;
}
private T[,] CutMatrix<T>(T[,] input, int order)
{
    T[,] output = new T[order, order];
    for (int i = 0; i < order; i++)
    {
        for (int j = 0; j < order; j++)
        {
            output[i, j] = input[i, j];
        }
    }
    return output;
}
private T[,] CreateSmallerMatrix<T>(T[,] input, int i, int j)
{
    int order = (int)Math.Sqrt(input.Length);
    T[,] output = new T[order - 1, order - 1];
    int x = 0, y = 0;
    for (int m = 0; m < order; m++, x++)
    {
        if (m != i)
        {
            y = 0;
            for (int n = 0; n < order; n++)
            {
                if (n != j)
                {
                    output[x, y] = input[m, n];
                    y++;
                }
            }
        }
        else
        {
            x--;
        }
    }
    return output;
}
private double Determinant(int[,] input)
{
    int order = (int)Math.Sqrt(input.Length);
    if (order > 2)
    {
        double value = 0;
        for (int j = 0; j < order; j++)
        {
            int[,] Temp = CreateSmallerMatrix(input, 0, j);
            value += input[0, j] * (SighOfElement(0, j) * Determinant(Temp));
```

```
        }
        return value;
    }
    else if (order == 2)
    {
        return ((input[0, 0] * input[1, 1]) - (input[1, 0] * input[0, 1]));
    }
    else
    {
        return input[0, 0];
    }
}
private void WRITEtoFILE(double[,] input, string path)
{

    //int Order = input.ColumnCount;
    using (StreamWriter strW = new StreamWriter(path))
    {
        for (int i = 0; i < input.GetLength(0); i++)
        {
            for (int j = 0; j < input.GetLength(1); j++)
            {
                strW.Write(input[i, j] + "\t");
            }
            strW.Write(Environment.NewLine);
        }
    }
}
private void WRITEtoFILE<T>(T[,] input, string path)
{

    int Order = input.GetLength(0);
    using (StreamWriter strW = new StreamWriter(path))
    {
        for (int i = 0; i < Order; i++)
        {
            for (int j = 0; j < Order; j++)
            {
                strW.Write(input[i, j] + "\t");
            }
            strW.Write(Environment.NewLine);
        }
    }
}
private List<double[,]> FormMatrices(String input, int order)
{
    List<double[,]> list = new List<double[,]>();
    int amount = input.Length / (int)Math.Pow(order, 2);
    int iterator = 0;
    for (int i = 0; i < amount; i++)
    {
        double[,] Matrix = new double[order, order];
        for (int j = 0; j < order; j++)
        {
            for (int k = 0; k < order; k++)
            {
                double.TryParse(input[iterator].ToString(), out Matrix[j, k]);
                iterator += 1;
            }
        }
        list.Add(Matrix);
    }
    return list;
}
private double FindXi(int[] input)
{
    FILESTRING += String.Format("M={0} K={1}  N={2}\r\n", 128, 5, 49);
```

```
        double output = 0;                                  /*                  */FILESTRING += String.Format("X2(obs)= SUM(vi-
NПi)^2/NПi = \r\n");
        for (int i = 0; i < K; i++)
        {
            output += (Math.Pow(input[i] - R * PCoef[i], 2) / (R * PCoef[i]));   /*                  */FILESTRING += 
String.Format("({0}-{1}*{2})^2/{1}*{2} +", input[i], R, PCoef[i]);
        }                                                   /*                  */FILESTRING .Remove(FILESTRING.Length - 1);
FILESTRING += "\r\n = "; FILESTRING += String.Format("{0}", output);
        return output;
    }
    private int[] CountAmountOfMaxSequence(int[] input)
    {
        int[] output = new int[6];
        for (int i = 0; i < input.Length; i++)
        {
            switch (input[i])
            {
                case 5: output[1] += 1; break;
                case 6: output[2] += 1; break;
                case 7: output[3] += 1; break;
                case 8: output[4] += 1; break;
                default: if (input[i] <= 4)
                        output[0] += 1;
                    if (input[i] >= 9)
                        output[5] += 1;
                    break;
            }
        }
        return output;
    }
    private void FillCheckBox()
    {
        foreach (var ch in this.Controls)
        {
            CheckBox Check = ch as CheckBox;
            if (Check != null)
            {
                foreach (var rt in this.Controls)
                {
                    RichTextBox Rich = rt as RichTextBox;
                    if (Rich != null)
                    {
                        String s1 = Check.Name.Remove(0, 2);
                        String s2 = Rich.Name.Remove(0, 2);
                        double ToCon = 0;
                        if (s1 == s2)
                        {
                            try
                            {
                                ToCon = Convert.ToDouble(Rich.Text);
                            }
                            catch (Exception)
                            {
                                ToCon = -1;
                            }
                            if (ToCon > STANDART_P)
                                Check.Checked = true;
                            else
                                Check.Checked = false;
                        }
                    }
                }
            }
        }
    }
    private bool CheckCondition(String str, ref double P)
    {
        int Amount1 = 0;
```

```
    for (int i = 0; i < str.Length; i++)
    {
        if (str[i] == '1')
            Amount1 += 1;
    }
    P = (double)Amount1 / (double)str.Length;
    bool AllRight = false;
    AllRight = (Math.Abs(P - 0.5) < 2 / Math.Sqrt(str.Length));
    return AllRight;
}
//аргумент для 3-го тесту (V  - к-ть знакозмін ,len - довжина строки ,pi  -частка одиниць в загальній масі )
private double FindArgumentForTest3(int V, int len, double pi)
{
    double arg = 0;
    double ar1 = Math.Abs(V - 2 * len * pi * (1 - pi));
    double ar2 = 2 * Math.Sqrt(2 * len) * pi * (1 - pi);
    arg = ar1 / ar2;
    return arg;
}
private int CountChange(String str)
{
    int count = 0;
    for (int i = 0; i < str.Length - 1; i++)
    {
        if (str[i] != str[i + 1])
            count += 1;
    }
    return count + 1;
}
private int Factorial(int i)
{
    if (i <= 1)
        return 1;
    return i * Factorial(i - 1);
}
private double FindXi2(int[] input, int length, int NumOfBlock)
{
    double SUM = 0;
    int NumInBlock = length / NumOfBlock;
    for (int i = 0; i < NumOfBlock; i++)
    {
        double a1 = (input[i] / (double)NumInBlock - 0.5);
        double a2 = (Math.Pow(a1, 2));
        SUM += a2;
    }
    SUM *= (4 * NumInBlock);
    return SUM;
}
private int[] CountMaxSequence(string str, int NumOfBlocks)
{
    int[] Arr = new int[NumOfBlocks];
    int MaxLenght;
    int NumInBlock = str.Length / NumOfBlocks;
    List<String> lstr = new List<string>();
    CutOnFrames(str, lstr, NumInBlock);
    string s1; bool isContains;
    for (int i = 0; i < lstr.Count; i++)
    {
        MaxLenght = 0; s1 = "1"; isContains = true;
        while (isContains)
        {
            s1 += "1";
            MaxLenght += 1;
            isContains = lstr[i].Contains(s1);
        }
        Arr[i] = MaxLenght;

    }
```

```
    return Arr;
}
private void Max(ref int max, int a1)
{
    if (max < a1)
        max = a1;
}
private int[] CountAmountInBlock(String str, int N)
{
    StringBuilder sbl = new StringBuilder();
    sbl.Append(str);
    int amount = str.Length / N;
    int[] Arr = new int[N];
    int j = 0;
    for (int i = 0; i < sbl.Length; i++)
    {
        if (sbl[i] == '1')
            Arr[j] += 1;
        if (i == amount - 1)
        {
            j += 1;
            i = -1;
            sbl.Remove(0, amount);
        }
    }
    return Arr;
}
private void FormBinaryString(List<int> list, StringBuilder str)
{
    str.Clear();
    foreach (var num in list)
    {
        String reserve = Convert.ToString(num, 2);
        for (int i = 0; i < (5 - reserve.Length); i++)
        {
            str.Append("0");
        }
        str.Append(reserve);
        //str.Append(" ");
    }
}
private void CountNotZero(StringBuilder stb, ref int count)
{
    count = 0;
    foreach (var s in stb.ToString())
    {
        if (s == '1')
            count += 1;
        else
            count -= 1;
    }
}

public static class Butterfly
{
    public const double SinglePi = Math.PI;
    public const double DoublePi = 2 * Math.PI;

    public static Complex[] DecimationInTime(Complex[] frame, bool direct)
    {
        if (frame.Length == 1) return frame;
        var frameHalfSize = frame.Length >> 1;
        var frameFullSize = frame.Length;

        var frameOdd = new Complex[frameHalfSize];
        var frameEven = new Complex[frameHalfSize];
        for (var i = 0; i < frameHalfSize; i++)
        {
```

```
            var j = i << 1; // i = 2*j;
            frameOdd[i] = frame[j + 1];
            frameEven[i] = frame[j];
        }

        var spectrumOdd = DecimationInTime(frameOdd, direct);
        var spectrumEven = DecimationInTime(frameEven, direct);
        var arg = direct ? -DoublePi / frameFullSize : DoublePi / frameFullSize;
        var omegaPowBase = new Complex(Math.Cos(arg), Math.Sin(arg));
        var omega = Complex.One;
        var spectrum = new Complex[frameFullSize];

        for (var j = 0; j < frameHalfSize; j++)
        {
            spectrum[j] = spectrumEven[j] + omega * spectrumOdd[j];
            spectrum[j + frameHalfSize] = spectrumEven[j] - omega * spectrumOdd[j];
            omega *= omegaPowBase;
        }
        return spectrum;
    }
    public static Complex[] DecimationInFrequency(Complex[] frame, bool direct)
    {
        if (frame.Length == 1) return frame;
        var halfSampleSize = frame.Length >> 1; // frame.Length/2
        var fullSampleSize = frame.Length;
        var arg = direct ? -DoublePi / fullSampleSize : DoublePi / fullSampleSize;
        var omegaPowBase = new Complex(Math.Cos(arg), Math.Sin(arg));
        var omega = Complex.One;
        var spectrum = new Complex[fullSampleSize];

        for (var j = 0; j < halfSampleSize; j++)
        {
            spectrum[j] = frame[j] + frame[j + halfSampleSize];
            spectrum[j + halfSampleSize] = omega * (frame[j] - frame[j + halfSampleSize]);
            omega *= omegaPowBase;
        }

        var yTop = new Complex[halfSampleSize];
        var yBottom = new Complex[halfSampleSize];
        for (var i = 0; i < halfSampleSize; i++)
        {
            yTop[i] = spectrum[i];
            yBottom[i] = spectrum[i + halfSampleSize];
        }
        yTop = DecimationInFrequency(yTop, direct);
        yBottom = DecimationInFrequency(yBottom, direct);
        for (var i = 0; i < halfSampleSize; i++)
        {
            var j = i << 1; // i = 2*j;
            spectrum[j] = yTop[i];
            spectrum[j + 1] = yBottom[i];
        }

        return spectrum;
    }
}
private void FormDigitString(List<int> digit, One[] one, String str)
{
    digit.Clear();
    for (int i = 0; i < str.Length; i++)
    {
        for (int j = 0; j < one.Length; j++)
        {
            if (str[i] == one[j].a2)
            {
                digit.Add(one[j].number);
                break;
            }
```

```
            }
        }
    }
    private class One
    {
        public int a1;
        public char a2;
        public int number;
    }
    private void FrequencyForOne(String s, One[] one, int ty)
    {
        for (int i = 0; i < 32; i++)
        {
            one[i] = new One();
            one[i].number = i + ty;
        }
        for (int i = 0; i < mstr.Length; i++)
        {
            one[i].a1 = s.Count(x => x == mstr[i]);
            one[i].a2 = mstr[i];
        }
    }
    private void FormAlphabet(char ch, StringBuilder st, StringBuilder st2)
    {
        if (st2.ToString().Contains(ch))
        {
            st.Append(ch);
            st2.Replace(ch, '*');
        }
    }
    private void Form1_Load(object sender, EventArgs e)
    {
        int p = 0; var St = new StringBuilder();
        var St2 = new StringBuilder();
        String str01 = File.ReadAllText("C_C_VT00");
        int gh = str01.Length;
        for (int i = 0; i < 255; i++)
        {
            char c = Convert.ToChar(i);
            if (str01.Contains(c))
            {
                if ((p = Convert.ToInt32(c)) < 65 || (p = Convert.ToInt32(c)) > 90)
                {
                    St2.Append(c);
                }
                else
                {
                    St.Append(c);
                }
            }
        }
        FormAlphabet(' ', St, St2);
        FormAlphabet('.', St, St2);
        FormAlphabet(',', St, St2);
        FormAlphabet(';', St, St2);
        FormAlphabet('-', St, St2);
        FormAlphabet(Convert.ToChar(39), St, St2);
        St2.Replace("*", ""); St2.Replace("\n\r", "");
        string g = St2.ToString();
        if (g.Length == 0) { mstr = St.ToString(); }
        else
        {
            St.Append(g);
            mstr = St.ToString();
        }
    }
}
}
```

ДОДАТОК Г. БІНАРНИЙ КОД ДЛЯ ОЦІНЮВАННЯ ШИФРУВАННЯ ІЗ ВИКОРИСТАННЯМ ТЕСТІВ NIST USA

Бінарний код шифрованого тексту:

00101 11011 11011 11111 11011 11100 01000 00001 11000 00001 11001 10010
00111 01000 01101 01111 11001 10011 01010 01011 00011 11000 10101 10001
00111 10010 10111 10010 01010 00011 11111 00011 10100 00111 10000 00010
01000 01101 10101 10100 11111 01101 01011 10110 01101 01000 11010 10101
01010 11000 01111 01111 00001 11001 10111 00111 10111 00000 00111 11101
10001 10101 01010 11000 10000 00110 10011 00010 10010 10001 10011 10111
01111 01000 10001 10100 00010 11100 00110 01101 01000 00100 10100 11111
01100 11110 01111 11111 10111 01010 10001 10010 11101 10001 01110 11011
10100 10101 00100 01001 11101 01010 01000 10110 11100 11101 01111 11010
01011 10110 11011 11011 00010 00000 00100 01110 01010 11111 01100 00001
00000 01001 11011 01000 11010 11011 11111 11111 00101 01110 10101 11111
01110 10000 10101 01110 10000 11101 10100 11111 00011 11110 00101 00000
10100 10100 10110 10111 00111 11011 11001 10101 00010 11011 00111 00111
01010 00100 10101 11000 10001 00001 00111 00111 01000 01110 10111 01111
11010 10000 00111 01010 10011 11100 01110 11011 01111 11100 01110 00011
10100 10101 00100 01001 10110 11000 01101 00111 10110 11010 01111 01110
10100 11111 00001 11011 01001 00110 00000 11010 10010 11111 00110 11111
11010 01101 10000 01100 11111 10100 10101 00001 00011 11011 01000 10000
11011 11000 10001 01010 10100 01010 11111 01111 11110 10000 10110 11110
10111 11010 00111 00001 11010 11110 00110 00111 11101 11100 00000 11001
10111 10100 10110 01100 00101 10101 01000 01000 11011 01000 11011 00001
01101 11011 00010 00100 10111 10100 00111 00001 11001 11111 10111 01111
00011 00011 01000 11100 01111 01000 00001 00000 11111 00001 00100 01111
01111 10101 11000 10110 11100 01010 01010 11100 11011 00111 01100 10000
11111 00000 00000 01010 01110 10010 00101 10100 10001 10101 01101 01000
10000 11011 11001 10111 01000 11111 00010 11011 11110 10101 00001 00011
01011 01001 11000 11111 10100 00110 11100 11101 11010 10011 11111 10011
10011 10110 11110 11010 11100 00011 01110 11011 00101 10000 01010 00111
11111 10001 10011 11110 10010 10101 11000 00011 11110 01011 00110 10001
10111 11111 00101 01000 11110 01100 00101 00101 00100 01101 00001 11001
10100 01111 00011 10000 11110 10111 00111 00011 11110 11000 11110 01011
10110 11111 11111 01100 00001 00111 01001 01010 11010 00101 00111 00010
10101 11111 00011 11111 00010 00111 00101 01010 11010 11010 00001 00111
11100 11100 11111 00101 10010 11001 10100 10101 11000 01011 10000 00101
01101 00000 00100 11010 00011 10110 01101 10001 01001 01110 11000 11010
01001 00110 11101 10011 01100 10001 01110 11010 00110 00010 11010 11101
11100 01111 10110 10110 01110 10011 10011 10111 00001 11001 00011 00111
01001 10010 00101 11010 10010 01101 10100 11011 11111 11010 11010 00010
10100 00100 11010 01000 01111 10000 01011 00010 01001 00011 00110 01100
11011 00011 11000 01110 01010 11100 01011 00100 10000 01110 11101 00110

11111 01000 01000 10011 11100 11111 11001 00100 00011 10011 01000 00100
10011 11000 00110 00001 11001 00100 11001 11011 01100 01111 10101 10111
10011 01000 01000 10001 10111 00111 10110 01101 11110 01010 00100 11001
10101 11100 10111 11100 11110 01100 00101 00101 11100 01010 11010 11001
10011 00011 00001 10001 01000 00010 10000 01111 10000 01110 10111 01111
10000 01111 11010 01001 00001 11000 11000 11100 10101 00100 11000 11101
00000 10111 00110 00000 01111 01100 00010 00011 11110 10010 00100 01110
11010 11010 00100 10110 11110 11101 10011 00010 10000 10111 10101 00011
11011 01111 11000 01100 01100 10111 00110 00100 00001 11011 11111 11011
10000 11010 10111 11111 10010 11101 01100 00010 11010 01111 11101 11100
00011 00101 10000 00011 10110 10011 00011 11011 00100 01001 00111 10011
00001 00011 11001 11111 00010 01000 10110 00100 01001 01011 10111 10111
11110 00101 10011 01010 00010 11111 10011 00001 10001 10111 00110 01000
11111 10111 00101 00100 10000 01100 00101 10111 01110 01000 10110 01111
10000 11000 00111 00011 11000 00001 10000 11100 01010 11101 01011 11101
10110 01111 01011 10001 01101 01100 11111 01011 10011 01101 11001 11111
11010 10001 11000 11001 00100 01100 00001 11010 10110 10011 01101 10011
01000 01101 10010 01101 11000 01110 11011 10101 01010 10011 01110 01010
10010 10110 00011 11001 10000 10110 11011 01111 10010 10011 00011 11010
10011 10111 00111 11000 10000 01111 00101 01011 01011 00011 10000 00100
00101 01011 01011 01111 10010 00110 00011 10111 01010 00110 10001 00100
10011 10001 10111 11001 01000 00101 01000 01000 00010 11110 01101 00100
11010 00110 00011 10111 00101 11101 01010 11011 11101 11011 10000 11111
00110 11101 00000 10010 10100 00001 00100 00011 10000 10011 11010 11001
10101 00010 10000 00110 11001 00100 11100 10001 01001 01110 11000 11010
10001 01000 01000 11011 10110 00011 01011 00101 10100 01100 01101 01110
10010 00001 10000 11011 01011 11111 10100 11100 10001 00000 00100 11011
11111 10110 01001 01101 10001 00100 11100 00100 10000 00011 00110 00010
11111 11000 00100 01010 10100 00000 10101 01111 00110 01010 11110 01010
11101 00010 11110 01110 00110 11100 00100 11111 10011 11100 11010 10011
11111 11110 11000 00011 10111 00011 10000 01001 00110 01000 01100 01101
01001 11011 01101 01001 00110 11010 10001 01111 00000 10111 00100 00011
11011 01100 00111 11010 11110 10111 00111 00011 01101 11001 10000 00111
00011 10100 00111 10001 01010 01111 11101 11011 10011 00111 11001 00111
11000 00111 00010 11011 01001 00100 11110 01011 11100 10100 01101 11011
00000 10100 10000 00111 11110 11100 10110 11011 01011 00101 11000 01011
00100 10100 10111 01000 00101 11110 00001 01011 11000 10010 10111 11111
01010 00100 11011 11111 01101 11001 00111 10111 00110 00010 11010 11110
11101 10010 10111 00001 11001 00111 01110 10000 10110 10001 01111 11011
10001 11011 00101 11111 10101 10010 00001 11111 00010 00100 00111 11000
11010 01011 00110 11010 10010 11011 10111 10111 00101 00101 10110 00000
11111 01100 00001 11011 11010 11111 11010 11010 11010 11111 11100 00011
10111 11100 01010 00011 00110 10010 10100 10000 10010 10100 10100 11100
01100 00001 11011 10011 10000 10111 11010 11011 11010 10001 11110 10010
11110 11001 00011 11100 01000 10001 10111 01101 00000 11111 01100 11110

01100 11000 00101 00010 10001 10101 01101 01000 11001 11011 00101 00011
00100 11011 11010 11010 10001 10001 01011 10101 11011 11011 10111 11110
11110 11100 01011 11000 10011 01001 01110 00111 10000 00101 01010 11111
11110 11111 11111 00011 00000 01011 11111 10010 11010 00001 10000 11011
11101 10111 00001 01110 01001 10100 01110 10001 10011 10110 10010 00001
10001 01001 10000 00000 00000 10000 10010 10100 10010 10000 10001 11001
01101 10100 10111 01011 11100 01011 00100 11011 10101 10111 10010 00101
10011 10101 01111 00101 10000 01111 00101 01011 01110 00110 11000 10111
01011 11011 01100 10111 00001 01011 11111 11101 11100 01000 00000 00001
10010 00001 00110 11011 01101 10111 01101 00111 10010 10001 01111 10100
01111 01011 10010 00101 11101 11111 01100 01111 00001 00100 00001 11011
01001 10100 10110 01011 11100 01100 11000 10111 00010 00111 10111 10001
10110 01000 10100 11000 00010 00101 01100 00010 11110 11011 11100 00011
10000 00010 10001 11110 10001 00010 11100 10011 01100 10000 00011 01100
11101 00110 10010 10010 11110 01100 00101 00101 00000 10010 01111 11010
00100 00011 01011 01100 11011 10001 10101 01111 00000 10111 11011 01011
10001 00101 10101 11101 00001 01101 01101 11000 00110 11010 01110 11010
01001 10100 00101 00000 01110 00110 00011 00111 10101 01110 00001 10101
10010 10000 10100 00000 01100 01110 11010 00101 10001 11111 00101 00111
00010 00001 11111 00100 11110 00010 01100 01110 00110 00100 10000 01101
00011 10000 10111 01010 11111 11101 01100 10110 11011 11011 00101 10000
01111 11111 10111 01100 01111 10101 01100 01111 11010 10001 10000 11011
00001 10000 01100 11011 10011 00001 01001 01110 11100 11010 00101 11110
10000 01011 01100 11010 00110 01110 10101 11100 11100 00010 11011 00000
10011 11011 10110 01101 00110 10110 00110 01101 11100 10111 11011 11011
11101 11110 11110 00001 10000 01110 01100 10100 01100 10111 00001 00011
00011 01100 01010 10100 11110 10111 00111 00011 01101 01001 11001 11100
01010 00001 01010 10111 11001 10110 11000 10101 01100 00100 00111 11000
01001 10110 10011 10100 11111 10111 11111 00010 11111 01100 10100 11000
11001 00010 00101 00011 00110 10001 10010 01011 00100 01101 00100 01010
11011 01011 01110 01011 10000 00101 11100 10011 00011 10100 10010 11100
10000 01001 01110 10011 10011 11100 01010 10011 11110 11000 01110 01110
01101 11011 00000 01001 01010 00100 11011 10110 00011 11000 11111 10010
00101 10101 10110 00000 00011 00000 01100 00111 10000 01110 10100 10010
11110 11111 11111 00011 00101 01100 00110 10010 01111 00010 00100 10100
01100 01101 10110 11111 11101 01011 00001 11101 00110 10100 11010 11111
11000 10100 10110 00111 01000 00110 11001 10110 01110 01100 01010 00000
01010 10100 01011 10011 10101 11110 10100 11111 10001 10011 00101 01001
01100 01100 11011 11000 10001 10110 00001 00000 01000 10100 11001 01110
11100 10010 11010 11111 11100 11100 00100 00000 01100 01111 10000 10001
11001 01010 01010 01110 00000 00100 11100 00111 11110 01001 11010 00010
01110 11111 10111 11001 10011 10001 10110 10111 00000 10100 01001 10111
01110 00101 10111 10001 10000 00101 01101 11100 10101 11011 01111 00111
10000 11001 11111 00111 10010 11000 11010 00100 00011 11111 10001 01110
10110 00101 10001 10100 10010 10010 10011 11100 01000 01111 11101 10011

10111 01111 01101 00111 01100 11100 11001 10101 01010 11010 11111 00010
00010 01000 10100 01000 00100 11101 11110 10011 11001 01001 00101 11110
01001 00101 00010 00010 11000 00101 10100 11011 01010 01111 10011 10011
01011 01110 10111 11011 10011 11000 00000 00011 10001 11110 00001 11100
10001 11111 00110 01100 11101 10010 10111 00001 00100 01110 10100 01111
00100 11000 10000 00111 11001 01111 10011 11100 01110 00110 10010 01011
10001 10000 01010 11011 00011 10001 11011 00001 01110 01001 01011 00100
01000 11000 11000 00110 01001 00001 11011 00111 11111 00111 01101 10101
01100 00000 00100 00111 10010 00001 10100 00100 10011 00000 00110 11101
01010 01001 10100 10111 00110 00100 10000 01101 00001 01010 10110 10011
11110 10100 01010 10101 00101 00000 11010 01000 01010 01111 00100 00011
10101 10111 01010 00001 11111 01011 01111 00000