Центральноукраїнський національний технічний університет
Центр заочної та дистанційної освіти
Кафедра кібербезпеки та програмного забезпечення

”Допущено до захисту”
Завідувач кафедри кібербезпеки
та програмного забезпечення
д.т.н., професор
_____ Олексій СМІРНОВ
“____” _____ 2023 р.

ВИПУСКНА КВАЛІФІКАЦІЙНА РОБОТА
за першим (бакалаврським) рівнем вищої освіти
на тему
“Програмне забезпечення системи кібербезпеки ЕЦП для автентифікації користувачів у системах дистанційного навчання та тестування”

Виконав здобувач вищої освіти
IV курсу, групи КБ-19ПЗ
ОПП «Кібербезпека»
спеціальності 125 «Кібербезпека»
_____ Прудкий В.В.
«____» _____ 2023 р.

Керівник проекту
доктор технічних наук, професор
_____ Смірнов О.А.
«____» _____ 2023 р.
Рецензент _____
_____

м. Кропивницький

Центральноукраїнський національний технічний університет

Центр *Заочної та дистанційної освіти*
Кафедра *Кібербезпеки та програмного забезпечення*
Освітній ступінь *бакалавр*
Галузь знань ._12 _____*"*Інформаційні технології*"*_____
Спеціальність *125 "*Кібербезпека*"*
Освітньо-професійна (освітньо-наукова) програма _*"*Кібербезпека*"*_

ЗАТВЕРДЖУЮ
Завідувач кафедри
д.т.н., проф.
_____ Олексій СМІРНОВ
« 17 » січня 2023 року

ЗАВДАННЯ НА ВИПУСКНУ КВАЛІФІКАЦІЙНУ РОБОТУ ЗА ПЕРШИМ (БАКАЛАВРСЬКИМ) РІВНЕМ ВИЩОЇ ОСВІТИ ЗДОБУВАЧА ВИЩОЇ ОСВІТИ

Прудкому Владиславу Васильовичу
(прізвище, ім'я, по батькові)

1. Тема роботи *Програмне забезпечення системи кібербезпеки ЕЦП для автентифікації користувачів у системах дистанційного навчання та тестування*

2. Керівник роботи *Смірнов Олексій Антаолійович, докт. техн. наук, професор*
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом вищого навчального закладу № 16-02 від 5.01.2023 року

3. Строк подання студентом роботи до захисту *23.05.2023 р.*

4. Мета та завдання випускної кваліфікаційної роботи*: Метою роботи є розробка програмного забезпечення системи кібербезпеки ЕЦП для автентифікації користувачів у системах дистанційного навчання та тестування*

5. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

1. Призначення та область використання.
2.Перегляд аналогічних існуючих систем.
3. Опис і обґрунтування проектних рішень.
4. Етапи програмування системи.
5. Впровадження системи кібербезпеки в промислову експлуатацію.
6. Висновки

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

Структурна схема системи кібербезпеки	*1 аркуш*
Функціональна схема системи кібербезпеки	*1 аркуш*
Діаграма процесів	*1 аркуш*
Блок-схема алгоритму роботи додатку	*2 аркуша*

7. Дата видачі завдання « 17 » січня 2023 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти	Строк виконання етапів випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти	Примітка
1.	Аналіз існуючих систем	10.03.2023 р.	
2.	Постановка задачі, оформлення ТЗ	15.03.2023 р.	
3.	Розробка моделі компонента	20.03.2023 р.	
4.	Розробка структур даних	25.03.2023 р.	
5.	Розробка алгоритмів зв’язку та відображення	30.03.2023 р.	
6.	Програмування алгоритмів	10.04.2023 р.	
7.	Оформлення ПЗ	17.04.2023 р.	
8.	Попередній захист роботи	23.05.2023 р.	

Дата видачі завдання
« 17 » січня 2023 р.

Підпис керівника
_____ Смірнов О.А.
(прізвище та ініціали)

Завдання прийнято до виконання
« 17 » січня 2023 р.

Підпис здобувача
_____ Прудкий В.В.
(прізвище та ініціали)

АНОТАЦІЯ

Прудкий В.В. Програмне забезпечення системи кібербезпеки ЕЦП для автентифікації користувачів у системах дистанційного навчання та тестування. 125 Кібербезпека. Центральноукраїнський національний технічний університет. Кропивницький. 2023.

В даній випускній кваліфікаційній роботі за першим (бакалаврським) рівнем вищої освіти розроблено програмне забезпечення, яке призначено для системи кібербезпеки ЕЦП для автентифікації користувачів у системах дистанційного навчання та тестування.

Метою розробки є програмне забезпечення системи кібербезпеки ЕЦП для автентифікації користувачів у системах дистанційного навчання та тестування.

Результат роботи – програмна реалізація системи кібербезпеки ЕЦП для автентифікації користувачів у системах дистанційного навчання та тестування.

В процесі роботи над програмною моделлю виконано аналіз існуючих апаратних та програмних засобів. В повній мірі описані всі компоненти розробленого програмного забезпечення.

Розроблено зручний інтерфейс користувача. Наведені інструкції по роботі з програмними засобами.

Програма може використовуватися на ПЕОМ архітектури IBM PC з ОС Windows 10/11.

Програму розроблено в середовищі Visual C#.

Ключові слова: кібербезпека, ЕЦП, автентифікація, система дистанційного навчання та тестування

ABSTRACT

Prudkyi V.V. EDS cyber security system software for user authentication in distance learning and testing systems. 125 Cyber security. Central Ukrainian National Technical University. Kropyvnytskyi. 2023.

In this final qualification work for the first (bachelor) level of higher education, software is developed, which is intended for the EDS cyber security system for user authentication in distance learning and testing systems.

The goal of the development is the EDS cyber security system software for user authentication in distance learning and testing systems.

The result of the work is the software implementation of the EDS cyber security system for user authentication in distance learning and testing systems.

In the process of working on the software model, an analysis of existing hardware and software was performed. All components of the developed software are fully described.

A convenient user interface has been developed. Instructions for working with software tools are provided.

The program can be used on PCs of IBM PC architecture with Windows 10/11 OS.

The program was developed in the Visual C# environment.

Keywords: cybersecurity, EDS, authentication, distance learning and testing system

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ І ТЕРМІНІВ 2
ВСТУП..... 3
1 ПРИЗНАЧЕННЯ ТА ОБЛАСТЬ ВИКОРИСТАННЯ 5
1.1 Призначення системи..... 5
1.2 Область застосування..... 7
2 ПЕРЕГЛЯД АНАЛОГІЧНИХ ІСНУЮЧИХ СИСТЕМ 10
2.1 Огляд існуючих систем, технологій, архітектур та програмних рішень за профілем теми випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти..... 10
2.2 Обґрунтування вибору засобів для побудови системи кібербезпеки та мови програмування..... 21
2.3 Розгорнута постановка завдання 24
3 ОПИС І ОБҐРУНТУВАННЯ ПРОЕКТНИХ РІШЕНЬ 26
3.1 Опис функціонування системи 26
3.2 Розробка структурної схеми..... 38
3.3 Розробка функціональної схеми 44
3.4 Розробка діаграми процесів..... 48
4 РЕАЛІЗАЦІЯ РОБОТИ. РОЗРАХУНКИ І ЕКСПЕРИМЕНТАЛЬНІ ДАНІ, ЩО ПІДТВЕРДЖУЮТЬ ВІРНІСТЬ ПРОЕКТНИХ ТА ПРОГРАМНИХ РІШЕНЬ..... 51
4.1 Розробка блок-схем та опис алгоритмів функціонування системи..... 51
4.2 Захист розробленого програмного забезпечення..... 78
5 ВПРОВАДЖЕННЯ СИСТЕМИ КІБЕРБЕЗПЕКИ В ПРОМИСЛОВУ ЕКСПЛУАТАЦІЮ 82
6 ОСНОВНІ ВИСНОВКИ..... 84
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 86

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ І ТЕРМІНІВ

ДО	–	дистанційна освіта
ЕК	–	еліптична крива
ЕЦП	–	електронний цифровий підпис
ЗЦ	–	засвідчуючий центр
СДН	–	системи дистанційного навчання
ЦП	–	цифровий підпис
ЦСК	–	центр сертифікації ключів

ВСТУП

Актуальність теми. «Дистанційне навчання» означає будь-яку освіту, яка надається без фізичної присутності викладача та студентів.

Раніше середні школи та університети пропонували заочну форму навчання як метод дистанційного навчання. Матеріали курсу часто надсилалися студентам поштою, а завдання виконувалися онлайн або поверталися викладачеві поштою.

Зовсім недавно програми дистанційного навчання використовують неймовірні можливості, які надають сучасні технології, і пропонують дуже індивідуальні та ефективні можливості навчання в усіх видах курсів дистанційної освіти. Від дитячого садка до початкової школи та університету ефективне дистанційне навчання тепер є життєздатним варіантом.

Існують, звичайно, деякі значні відмінності між дистанційним і традиційним навчанням; найбільш очевидним є відсутність вимоги щодо фізичної присутності в конкретному місці.

Під час дистанційного навчання студенти відчувають значно більше свободи у своєму підході до навчання. Це може бути позитивним аспектом, оскільки студенти можуть вибирати курси на основі власного розкладу, запропонованого стилю викладання та використовуваних модальностей.

Нетрадиційні учні можуть створити навчальне середовище, яке добре їм підходить, замість того, щоб вписуватися в традиційну освітню форму.

Однак на зворотному боці цієї свободи лежить вимога до студентів бути високодисциплінованими під час навчання. У випадку дистанційного навчання в університетському контексті наслідки можуть бути менш серйозними, але для елементарного дистанційного навчання і особливо дистанційного навчання для дитячого садка існує потреба в певному рівні нагляду дорослих, щоб забезпечити найкращі шанси на успіх.

Мета й завдання дослідження. Метою роботи є програмне забезпечення системи кібербезпеки ЕЦП для автентифікації користувачів у системах дистанційного навчання та тестування.

Для досягнення поставленої мети визначена програма дослідження, що складається з наступних завдань:

– Огляд існуючих систем ЕЦП для автентифікації користувачів у системах дистанційного навчання та тестування.

– Дослідження системи кібербезпеки ЕЦП для автентифікації користувачів у системах дистанційного навчання та тестування.

– Програмна реалізація системи кібербезпеки ЕЦП для автентифікації користувачів у системах дистанційного навчання та тестування.

Практична цінність отриманих результатів полягає в тому, що розроблені алгоритми дозволяють успішно вирішувати задачі ЕЦП для автентифікації користувачів у системах дистанційного навчання та тестування.

Таким чином, виходячи з вищеперерахованого, програмне забезпечення системи кібербезпеки ЕЦП для автентифікації користувачів у системах дистанційного навчання та тестування, є актуальною задачею, яка потребує вирішення у даній випускній кваліфікаційній роботі за першим (бакалаврським) рівнем вищої освіти.

1 ПРИЗНАЧЕННЯ ТА ОБЛАСТЬ ВИКОРИСТАННЯ

1.1 Призначення системи

Система призначена для реалізації електронного цифрового підпису. Цифрові підписи схожі на електронні «відбитки пальців». Вони є специфічним видом електронного підпису (е-підпису).

У формі кодованого повідомлення цифровий підпис надійно пов’язує підписувача з документом у записаній транзакції. Цифрові підписи використовують стандартний прийнятний формат, який називається інфраструктурою відкритих ключів (PKI) , щоб забезпечити найвищий рівень безпеки та універсальне визнання. PKI передбачає використання цифрового сертифіката для підтвердження особи.

Цифровий підпис – це тип електронного підпису, який вимагає більш суворого рівня забезпечення ідентифікації за допомогою цифрових сертифікатів.

Широка категорія електронних підписів (е-підписів) охоплює багато типів електронних підписів. До категорії входять цифрові підписи, які є певною технологією реалізації електронних підписів. Як цифрові підписи, так і інші рішення електронного підпису дозволяють підписувати документи та автентифікувати підписувача. Однак існують відмінності в цілях, технічній реалізації, географічному використанні, правовому та культурному прийнятті цифрових підписів порівняно з іншими типами електронних підписів.

Зокрема, використання технології цифрового підпису для електронних підписів значно відрізняється між країнами, які дотримуються відкритих, технологічно нейтральних законів про електронний підпис, включаючи Україну, Сполучені Штати, Великобританію, Канаду та Австралію, і країнами, які дотримуються багаторівневих моделей електронного підпису, надають перевагу місцевим стандартам, які базуються на технології цифрового підпису, включаючи Україну, багато країн Європейського Союзу, Південної Америки та Азії. У

Європейському Союзі існує два рівні цифрових підписів: вдосконалений електронний підпис (AES) і кваліфікований електронний підпис (QES). Крім того, деякі галузі також підтримують певні стандарти, які базуються на технології цифрового підпису.

Цифрові підписи, як і рукописні підписи, є унікальними для кожного підписувача. Постачальники рішень цифрового підпису дотримуються певного протоколу, який називається PKI. PKI вимагає від провайдера використовувати математичний алгоритм для генерації двох довгих чисел, які називаються ключами. Один ключ є відкритим, а один – закритим.

Коли підписувач електронно підписує документ, підпис створюється за допомогою закритого ключа підписувача, який завжди надійно зберігається підписувачем. Математичний алгоритм діє як шифр, створюючи дані, що відповідають підписаному документу, які називаються гешем, і шифруючи ці дані. Отримані зашифровані дані є цифровим підписом. У підписі також зазначається час підписання документа. Якщо після підписання документ змінюється, цифровий підпис стає недійсним.

Як приклад, Владислав підписує угоду про продаж таймшеру за допомогою свого закритого ключа. Покупець отримує документ. Покупець, який отримує документ, також отримує копію відкритого ключа Владислава. Якщо відкритий ключ не може розшифрувати підпис (за допомогою шифру, з якого було створено ключі), це означає, що підпис не належить Владиславу або його було змінено після підписання. Тоді підпис вважається недійсним.

Щоб захистити цілісність підпису, PKI вимагає, щоб ключі створювалися, використовувалися та зберігалися безпечним способом, і часто потрібні послуги надійного центру сертифікації (CA). Постачальники цифрових підписів, відповідають вимогам PKI для безпечного цифрового підпису.

1.2 Область застосування

Областю застосування розробляє мого програмного забезпечення є дистанційна освіта. Дистанційне навчання поділяється на дві основні категорії:

– Синхронне навчання.

– Асинхронне навчання.

Ви повинні розуміти різницю між синхронним і асинхронним. Різні типи дистанційного навчання належать до одного чи обох цих таборів.

Синхронне навчання

Синхронний означає «одночасно». Це стосується методу надання освіти, який відбувається в режимі реального часу. Потрібне живе спілкування онлайн. Для цього він використовує такі технології, як телеконференції.

Синхронне навчання виявляється менш гнучким, ніж інші форми дистанційного навчання. Зрештою, студенти повинні зустрічатися зі своїм інструктором, а іноді і зі своїми однокласниками у заздалегідь запланований час.

Такий підхід обмежує здатність студента навчатися у власному темпі. Це може розчарувати деяких учнів, які жадають свободи асинхронного класу.

Асинхронне навчання

Що стосується асинхронної дистанційної освіти? Студенти отримують кластери тижневих дедлайнів. Вони мають свободу працювати зі своєю швидкістю.

Асинхронне дистанційне навчання дає більше можливостей для взаємодії студентів.

Студенти можуть отримати доступ до вмісту курсу поза запланованою зустріччю чи заняттям і взаємодіяти за допомогою онлайн-бесід, тестів або відеокоментарів за власним розкладом.

І викладачі, і студенти виграють від гнучкості асинхронного навчання, оскільки воно дозволяє їм створювати та споживати вміст, коли їм це зручно.

Переваги асинхронного навчання

У сучасному світі як викладачі, так і студенти усвідомлюють, що зовнішні фактори сприяють непарним годинам і неузгодженим розкладам.

Гнучкість асинхронного навчання дозволяє їм створювати та споживати вміст, коли їм це зручно, а до навчальних матеріалів можна отримати доступ у будь-який час і будь-де.

Охоплення та залучення

З напруженим і непередбачуваним розкладом викладачі можуть розширити зміст курсу за межі запланованих зустрічей і занять за допомогою попередньо записаних відео та іншого вмісту.

Викладацький склад може використовувати записи в прямому ефірі або створювати відео, а потім отримувати аналітику, створювати субтитри, вести розмови, додавати тести та інтегрувати вміст безпосередньо в програмне забезпечення для керування навчанням (LMS).

Мотивація учнів

Асинхронні методи навчання допомагають мотивувати студентів переглядати вміст у вільний час і на будь-якому пристрої, який їм подобається.

Студенти можуть йти у своєму власному темпі та коли їм це зручно. Самостійне навчання враховує різноманітні навчальні потреби та вподобання та покращує успішність учнів.

Потім студенти можуть повертатися до вмісту для підготовки до іспитів, проводити обговорення та переглядати вміст поза межами живої лекції.

Доповнюйте досвід синхронного навчання

Завжди існує потреба у віртуальній живій взаємодії, але асинхронне спілкування доповнює це, щоб розширити живі сеанси за межі окремого класу.

Наприклад, замість того, щоб просто проводити зустріч у Zoom, професори можуть зробити набагато більше із записом. Вони можуть:

– Опублікуйте запис Zoom для повторного перегляду.

– Взаємодія навколо вмісту.

– Отримуйте аналітику участі.

– Створення підписів для доступності.

– Додати вікторини.

– Інтеграція з LMS.

Таким чином, виходячи з вищеперерахованого, програмне забезпечення системи кібербезпеки ЕЦП для автентифікації користувачів у системах дистанційного навчання та тестування, є актуальною задачею, яка потребує вирішення у даній випускній кваліфікаційній роботі за першим (бакалаврським) рівнем вищої освіти.

2 ПЕРЕГЛЯД АНАЛОГІЧНИХ ІСНУЮЧИХ СИСТЕМ

2.1 Огляд існуючих систем, технологій, архітектур, програмних рішень за профілем теми випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти

Далі розглянемо різні типи дистанційного навчання. Ці підходи до освіти можуть бути синхронними та асинхронними. Деякі належать до обох категорій.

Ці види дистанційного навчання включають:

– Відеоконференція.

– Гібридна дистанційна освіта.

– Відкрити розклад онлайн-курсів.

– Онлайн-курси з фіксованим часом.

Давайте розглянемо, що передбачає кожен із цих типів дистанційного навчання.

Відеоконференція

Відеоконференція – це традиційна зустріч, де два або більше учасників використовують відео для підключення через Інтернет. Це форма синхронного спілкування. Використовуючи такі інструменти, як Zoom, Blackboard Collaborate, Adobe Connect або інше програмне забезпечення для конференцій, викладачі та студенти взаємодіють разом незалежно від того, де вони знаходяться.

Відеоконференції покращують взаємодію між учнем і викладачем і забезпечують структуру для планування уроків. Це залишається життєво важливим компонентом дистанційного навчання.

Гібридна дистанційна освіта

Гібридна дистанційна освіта поєднує синхронні та асинхронні методи. Студенти отримують кінцеві терміни виконання завдань та іспитів. Потім вони працюють у власному темпі.

Вони подають завдання через онлайн-форуми. Вони підтримують зв'язок зі своїм інструктором. Проте вони працюють у своєму власному темпі. У міру прогресу студенти отримують доступ до нових модулів.

Кому процвітає гібридна дистанційна освіта? Студенти, які люблять самостійність.

Відкрити розклад онлайн-курсів

У категорії «Асинхронний» ви знайдете онлайн-інструкції з відкритим розкладом. Такі курси надають студентам багато свободи. Для виконання курсової роботи студенти отримують:

– Інтернет-підручник(и).
– Дошки оголошень.
– Електронна пошта.
– І більше.

Студентам надається набір дедлайнів. Потім інструктор дозволяє їм розкладати заняття у власному темпі. Студенти, які цінують самостійне навчання, досягають успіху в цьому форматі. Однак це вимагає значної самодисципліни та мотивації.

Студенти, які не мають належного набору навичок, можуть вважати такий підхід складним. Вони можуть відчувати себе приголомшеними поданням матеріалу. У них може не вистачати мотивації для ефективної роботи на курсі.

Онлайн-курси з фіксованим часом

Який найпоширеніший формат дистанційного навчання? Онлайн-курси з фіксованим часом.

Як вони працюють? Студенти входять на навчальний сайт у визначений час. Вони повинні виконати заздалегідь заплановані заходи в класі в певному темпі.

Ці заходи часто включають чати та дискусійні форуми. Онлайн-курси з фіксованим часом заохочують взаємодію студентів. Але там мало місця для самостійного темпу.

Створюйте захоплюючі відеоролики курсу за допомогою Camtasia

Незалежно від того, чи ви тільки починаєте створювати відео чи робили це раніше, робочий процес Camtasia інтуїтивно зрозумілий і простий у освоєнні.

Хто використовує дистанційне навчання?

Програми дистанційного навчання використовуються багатьма людьми з багатьох причин. Дві групи, які складають найбільший сегмент дистанційного навчання, це студенти (від початкової школи до університету) і професіонали.

Студенти

Дистанційні курси пропонуються для студентів різного віку. Учні середньої школи можуть пройти додаткові курси, щоб пришвидшити випуск, студенти університету чи коледжу можуть відвідувати школу в будь-якій точці світу без витрат на переїзд, а працюючі професіонали можуть підвищити свої навички за допомогою програм дистанційного навчання.

Дистанційне навчання може зробити освіту більш доступною для тих, хто навчається в сільській місцевості або тих, хто стикається з проблемами традиційної освіти.

Компанії

Діяльність дистанційного навчання була використана для навчання та адаптації багатьох компаній. Для тих, хто працює по всьому світу або має віддалених співробітників, дистанційне навчання створює можливість узгодженості досвіду співробітників у всіх сферах. Повні режими навчання можна забезпечити за допомогою програм дистанційного навчання.

Які переваги дистанційного навчання?

Звичайно, традиційне очне навчання є чудовим у багатьох відношеннях, але дистанційне навчання також має багато переваг.

Гнучкість

Однією з найвагоміших переваг дистанційної освіти є пропонована гнучкість.

Студенти можуть вибирати між можливостями структурованого дистанційного навчання з викладанням у реальному часі та доступом до вчителя в режимі реального часу або неструктурованими курсами дистанційного навчання, які можна легко адаптувати до напруженого графіка.

Програми дистанційного навчання можна пройти з будь-якої точки світу. Є формати, які підходять для багатьох різних стилів навчання. Більшість студентів знайдуть варіант дистанційної освіти, який відповідає їхнім потребам.

Легкий доступ

Дистанційна освіта створила нові можливості для студентів, які, можливо, мали труднощі з доступом до навчання в традиційному форматі. Незалежно від того, чи сталося це через віддаленість чи інвалідність, дистанційне навчання усуває бар'єри, пов'язані з відвідуванням очних занять.

Також відкрився легкий доступ до глобальних можливостей навчання, оскільки дистанційне навчання в університетах і коледжах робить міжнародне навчання вибором для багатьох студентів.

Це також створило можливість для тих, хто навчається впродовж життя з усього світу, отримати доступ до курсів і навчальних програм, представлених викладачами, до яких вони не мали б доступу інакше.

Економія грошей і часу

Дистанційне навчання зробило освіту набагато менш фінансово виснажливою та набагато ефективнішою за часом.

Доступ до програм дистанційної освіти для університетів і коледжів скорочує навчання на 50% порівняно з традиційним досвідом на кампусі.

Оскільки багато витрат, пов'язаних з інфраструктурою та матеріально-технічним забезпеченням, усуваються завдяки дистанційній освіті, витрати на доступ до неї значно нижчі, ніж у порівнянних моделях традиційної освіти.

Крім того, є економія, пов'язана з витраченим часом – звичайно, час на дорогу актуальний для студентів, але з боку вчителів можливість записувати та перепрофілювати уроки також призводить до значної економії часу.

Адаптивність і свобода

На відміну від традиційної моделі навчання, дистанційне навчання легко адаптується до стилю життя та навчальних потреб студента.

Розклад уроків, методи навчання та навчальне середовище адаптуються до кожного окремого учня за допомогою курсів дистанційного навчання таким чином, що неможливо в рамках очного навчання.

Студенти від дитячого садка до початкової школи та університету можуть використовувати свободи дистанційного навчання, пов'язані з часом, простором і темпом навчання, щоб знайти рівні успіху, яких вони не можуть отримати за традиційної структури.

Заробіток під час навчання

Працюючі професіонали будь-якого віку можуть використовувати дистанційне навчання, щоб отримати вищу освіту або отримати абсолютно новий набір навичок, зберігаючи свою повсякденну роботу.

Багато університетів і коледжів дистанційного навчання пропонують асинхронні програми, що дозволяє отримувати ступені та сертифікати поза робочим часом.

Експерти з різних предметів пропонують дистанційні курси для самостійного навчання, які можуть покращити знання та повноваження людини, не перериваючи її здатності отримувати прибуток.

Які недоліки дистанційного навчання?

Переваги дистанційного навчання очевидні, але є деякі застереження, коли мова йде про цей підхід до навчання. Розглянемо деякі недоліки.

Відсутність соціальної взаємодії

Обсяг соціальної взаємодії, запропонованої в дистанційному навчанні, набагато менший, ніж у традиційній моделі навчання.

Без вимоги відвідувати звичайне навчання студенти втрачають можливість працювати безпосередньо з однолітками. Ця характеристика дистанційного навчання може найбільше вплинути на дітей, особливо на дітей, які займаються

дистанційним навчанням у початковій школі, коли наріжним каменем є взаємодія з однолітками.

Одним із способів обмеження цього потенційного недоліку є використання відео як інструменту зв'язку, зокрема щодо надання зворотного зв'язку.

Отримання зворотного зв'язку (наприклад, оцінюваного тесту, письмового коментаря, розміченого есе тощо) у рамках дистанційного навчання часто відбувається без знайомих соціальних підказок, які допомагають зрозуміти контекст, і можуть здаватися дещо образливими. Це може призвести до того, що розум студента плаватиме у запитаннях і невизначеності, особливо якщо відгук не зовсім позитивний.

Однак, коли відео використовується для надання зворотного зв'язку, ці соціальні сигнали присутні. Можливість чути тон викладача та бачити його міміку може змінити світ і відновити певну соціальну взаємодію, якої може бути не вистачає під час дистанційного навчання.

Високі шанси відволіктися

Відволікання може бути проблемою для студентів, які навчаються за програмами дистанційного навчання. Це може проявлятися різними способами.

По-перше, студенти стикаються з вищим ризиком відволіктися онлайн. Без особистих зустрічей студенти можуть втратити поняття про дедлайни та мотивацію.

Студенти, які добре працюють самостійно, можуть легко подолати ці перешкоди. Студенти, яким важко визначити пріоритети, можуть спіткнутися. Так само будуть ті, кому бракує організаційних навичок і навичок планування.

Самомотивація та зосередженість є важливими навичками для успіху в дистанційному навчанні.

Складна технологія

Надмірна залежність від технологій є проблемою дистанційного навчання.

Студенти повинні мати надійний доступ до таких інструментів, як комп'ютер, веб-камера та стабільне підключення до Інтернету.

Будь-яка несправність апаратного чи програмного забезпечення з боку студента чи викладача може призвести до повної зупинки навчання.

Щоб бути успішними, студенти або їхні опікуни повинні мати помірний рівень комфорту з технологіями. Це вимога сучасного дистанційного навчання будь-якого рівня.

Сумнівна довіра до онлайн-дипломів

Багато роботодавців не приймуть ступінь або сертифікат програми дистанційного навчання. Це результат тривалої стигми навколо дистанційного навчання.

Не всі вчителі володіють навичками або комфортно викладати в онлайн-середовищі. Це сприяє неузгодженості з матеріалами курсу та напрямками.

Відчуття відсутності належного оцінювання є ще одним фактором, що сприяє цій проблемі з довірою до облікових даних, отриманих за допомогою програм дистанційного навчання.

Приховані студентські витрати

Хоча зменшення накладних витрат закладів часто призводить до нижчої вартості навчання для студентів, які навчаються дистанційно, є деякі приховані витрати, пов’язані з цим типом навчання.

Ці витрати включають:

- Отримання доступу до надійного комп'ютера.
- Наявність підключення до Інтернету.
- Купівля веб-камери (в окремих випадках).
- Обслуговування комп'ютера.
- Комунальні послуги (наприклад, електроенергія для інтернет-послуг).

Не всі студенти мають доступ до цих ресурсів. Дистанційне навчання може поставити їх у невигідне становище.

Чи визнаються дипломи дистанційного навчання?

В останні роки дистанційне навчання набуло все більшої популярності. Восени 2017 року 3,1 мільйона студентів вищих навчальних

закладів вступили на онлайн-програми. Крім того, зростає частка студентів, які навчаються повністю онлайн і навчаються в межах 50 км від свого дому.

Дистанційне навчання також завоювало широку довіру та визнання. Найкращі онлайн-університети можна порівняти з колегами на кампусі.

Тим не менш, студенти все одно повинні стежити за шахраями. Перш ніж закривати програму дистанційного навчання, вони повинні вивчити акредитацію навчального закладу.

Як виглядає акредитація? Хоча це може відрізнятися, акредитація відбувається на трьох рівнях:

– Програмна акредитація.

– Регіональна акредитація.

– Національна акредитація.

Програмна акредитація засвідчує дійсність певних програм навчання.

Регіональна акредитація означає, що регіональні агентства схвалили конкретні галузі навчання.

Національна акредитація означає, що програма відповідає федеральним вимогам акредитації.

Чи має цінність дистанційне навчання?

Абсолютно. Хоча колись дистанційне навчання вважалося неповноцінним замінником традиційної освіти, зараз багато хто вважає, що воно перевершує традиційне навчання в класі.

Це значною мірою завдяки відео та технологіям. Відео допомагає зробити дистанційне навчання привабливим і допомагає зацікавити учнів .

Студенти не тільки успішніші, але й віддають перевагу дистанційному навчанню.

77% академічних лідерів оцінюють онлайн-освіту як рівну або кращу. І 69% головних наукових співробітників погоджуються.

Дистанційне навчання є важливою частиною довгострокових освітніх стратегій.

Що робить програму дистанційного навчання гарною?

Не всі курси дистанційного навчання однакові. Є кілька ключових факторів, які слід враховувати під час пошуку високоякісних курсів дистанційного навчання. Давайте подивимося, що це таке.

Правильний темп

Найкращі дистанційні курси – це ті, які мають хороший темп.

Учні не повинні нудьгувати або перевантажуватися уроками, проектами чи матеріалами курсу.

Діяльність дистанційного навчання має включати великі проекти, до яких студенти мають достатньо часу для підготовки, з додаванням менших значущих завдань, щоб зберегти залученість та інтерес.

Мультимедійна інтеграція

Досконалість на дистанційних курсах може бути пов’язана з ефективним використанням мультимедійних засобів навчання.

Інтеграція подкастів, відео та інтерактивних занять може бути дуже привабливою для студентів і використовувати різні стилі навчання.

При цілеспрямованому використанні заняття дистанційного навчання, які включають мультимедійні засоби, можуть допомогти запам’ятати матеріал курсу. Насправді дослідження показують, що дві третини (67%) людей краще розуміють інформацію, коли вона передається візуально.

Якісний контент

Низькоякісний контент, як-от нескінченні завдання з читання підручників, монотонні лекції та оцінка запасів без креативності, призводить до низького рівня збереження матеріалу курсу дистанційного навчання.

Діяльність дистанційного навчання, яка зосереджується на високоякісному контенті, як-от захоплюючі відео, захоплені лекції та інтерактивні веб-сайти,

допомагає студентам підтримувати інтерес, зберігати інформацію та досягати більшого успіху.

Самостійне навчання

Курси дистанційного навчання, які пропонують учням можливість робити власні відкриття, завершувати проекти у спосіб, який їм найкраще підходить, і зосереджуватися на сферах навчання, які вони вважають найбільш цікавими, є курсами, які ведуть до найвищого успіху.

Підключення до спільноти

Великі програми дистанційного навчання визнають, що на студентів часто негативно впливає відсутність соціальної взаємодії в рамках цього стилю освіти, і навмисно включають спільноту в свою навчальну програму.

Ефективні курси включають групові проекти, де студенти повинні працювати разом, і можливості для однокласників підключатися за допомогою цифрових інструментів.

Кілька методів навчання

Найкращі можливості дистанційного навчання включають низку методів навчання, щоб дозволити студентам навчатися у спосіб, який їм найкраще підходить. Деякі учні запам’ятовують і сприймають інформацію найкраще візуально, а іншим потрібно почути інформацію, виголошену вголос.

Модальності, включені в найкращі курси дистанційного навчання, включатимуть, серед іншого, візуальні, аудіальні та кінестетичні.

Інтуїтивно зрозуміла навігація

Щоб студенти бачили успіх, програми дистанційного навчання мають бути інтуїтивно зрозумілими для навігації. В ідеалі це буде перевірено третьою стороною.

Учням слід представити добре спланований зміст курсу, який дозволить їм легко бачити, що і коли робити. Доступ до необхідних ресурсів та інформації ніколи не повинен бути проблемою в добре сформатованому курсі дистанційного навчання.

Надійна технологія

Найефективніші програми дистанційного навчання використовують технології, які є максимально універсальними. Студентам не потрібно завантажувати та вивчати нові чи ненадійні програми, плагіни чи розширення, щоб отримати доступ до матеріалу курсу або працювати з ним.

Яскраві доповнення технологій, хоча й потенційно цікаві, можуть негативно вплинути на досвід студентів на курсах дистанційного навчання.

Кімната для додаткового дослідження

Встановлення балансу між наданням студентам можливості зануритися в додаткові ресурси та перевантаженням їх надто великою кількістю дослідницьких можливостей є важливим балансом для програм дистанційного навчання.

Створення чіткого розмежування між обов'язковим матеріалом курсу та необов'язковими заходами для вдосконалення має вирішальне значення для забезпечення переконливого досвіду дистанційного навчання.

Яке майбутнє дистанційного навчання?

Незважаючи на те, що дистанційне навчання використовується століттями, широка доступність інтернет-сервісів у поєднанні з пандемією Covid-19 призвела до інтенсивного зростання його впровадження з 2020 року.

Було чітко показано, що фізична присутність у класі більше не є єдиним способом ефективного навчання.

Дистанційне навчання для учнів від початкової школи до університетів і коледжів пройшло довгий шлях розвитку, без жодних ознак його зростання.

Зі зростанням віддаленої та гібридної роботи на робочих місцях також продовжують застосовувати методи дистанційного навчання.

Системи управління навчанням і інструменти для створення освітнього контенту, такі як Snagit і Camtasia , які легко вбудовуються, спрощують, ніж будь-коли, надавати якісну освіту на робочому місці, незалежно від відстані. Чудова новина полягає в тому, що кожен із цих інструментів пропонує безкоштовну пробну версію, тому ви можете почати створювати чудові навчальні ресурси

прямо зараз! Хоча дистанційне навчання навряд чи повністю замінить особисте навчання, це, безумовно, ефективний інструмент, який продовжуватимуть розвиватися та інтегруватися у все більшу кількість сценаріїв.

2.2 Обґрунтування вибору засобів для побудови системи кібербезпеки та мови програмування

Програмне забезпечення написане мовою Visual C#. Ця мова обрана виходячи з наступних міркувань. Visual C# – строго типізована об'єктно-орієнтована мова, призначена для розробки різноманітних безпечних і потужних застосунків, виконуваних у середовищі .NET Framework. Мовою Visual C# можна розробляти звичайні клієнтські застосунки Windows, веб-служби XML, розподілені компоненти, застосунки типу " сервер-клієнт", застосунки баз даних і багато яких інших. В Visual C# є розширений редактор коду, конструктори зі зручним користувальницьким інтерфейсом, вбудований відладник і багато інших засобів, покликані спростити розробку застосунків мовою Visual C# версії 5.0 і .NET Framework версії 4.5.

Синтаксис Visual C# дуже виразний, але простий у вивченні. Усі, хто знаком з мовами C, C++ або Java з легкістю взнають синтаксис із фігурними дужками, характерний для мови Visual C#. Розроблювачі, що знають кожну із цих мов, як правило, зможуть домогтися ефективної роботи з мовою Visual C# за дуже короткий час. Синтаксис Visual C# робить простіше те, що було складно в C++, і забезпечує потужні можливості, такі як типи значень Nullable, перерахування, делегати, лямбда-вираження й прямий доступ до пам'яті, чого немає в Java. Visual C# підтримує універсальні методи й типи, забезпечуючи більше високий рівень безпеки й продуктивності, а також ітератори, що дозволяють при реалізації колекцій класів визначати власне поводження ітерації, що може легко використовуватися в клієнтському коді. В Visual C# 5.0 вираження LINQ

(Language-Integrated Query) роблять строго-типізований запит першокласною конструкцією мови.

Як об’єктно-орієнтована мова, Visual C# підтримує поняття інкапсуляції, спадкування й поліморфізму. Всі змінні й методи, включаючи метод Main – крапку входу застосунка – інкапсулюється у визначення класів. Клас може успадковувати безпосередньо з одного родового класу, але може реалізовувати будь-яке число інтерфейсів. Для методів, які перевизначають віртуальні методи в батьківському класі, необхідно ключове слово override, щоб виключити випадкове повторне визначення. У мові Visual C# структура схожа на полегшений клас: це тип, що розподіляється по стопках, що реалізує інтерфейси, але не підтримуюче спадкування.

На додаток до основних описаних об’єктно-орієнтованих принципів, мова Visual C# спрощує розробку компонентів програмного забезпечення завдяки декільком інноваційним конструкціям мови, у число яких входять наступні:

– Інкапсульовані підписи методів, називані делегатами, які підтримують строго-типізовані повідомлення про події.

– Властивості, що виступають у ролі методів доступу для закритих змінних-членів.

– Атрибути з декларативними метаданими про типи під час виконання.

– Вбудовані коментарі XML-документації.

– LINQ (Language-Integrated Query), що пропонує вбудовані можливості запитів у різних джерелах даних.

Якщо буде потрібно забезпечити взаємодію з іншим програмним забезпеченням Windows, таким як об'єкти COM або власні бібліотеки DLL Win32, у мові Visual C# можна використовувати процес, що називається "Interop". Процес Interop дозволяє програмам на Visual C# виконувати практично будь-які дії, які може виконувати вихідний додаток на C++. Мова Visual C# підтримує навіть покажчики й поняття "небезпечного" коду для тих випадків, коли прямий доступ до пам'яті має вкрай важливе значення.

Процес побудови Visual C# у порівнянні з C і C++ простий і є більше гнучким, чим в Java. Немає окремих файлів заголовка, а методи й типи не потрібно повідомляти в певному порядку. У вихідному файлі Visual C# може бути визначене будь-яке число класів, структур, інтерфейсів і подій.

Архітектура платформи .NET Framework

Програма мовою Visual C# виконується в середовищі .NET Framework – інтегрованому компоненті Windows, що містить віртуальну систему виконання (середовище CLR) і уніфікований набір бібліотек класів. Середовище CLR являє собою комерційну реалізацію корпорацією Майкрософт інфраструктури CLI, що є міжнародним стандартом, який лежить в основі створення середовищ виконання й розробки, у яких забезпечується тісна взаємодія між мовами й бібліотеками.

Вихідний код, написаний мовою Visual C#, компілюється в проміжну мову (IL) у відповідності зі специфікацією CLI. Код IL і ресурси, такі як растрові зображення й рядки, зберігаються на диску у файлі, що виконується, називаному складанням, з розширенням EXE або DLL у більшості випадків. Складання містить маніфест із відомостями про типи складання, версії, мови й регіональні параметри та вимоги безпеки.

При виконанні програми на Visual C# складання завантажується в середовище CLR залежно від відомостей у маніфесті. Далі, якщо вимоги безпеки дотримані, середовище CLR виконує JIT-компіляцію для перетворення коду IL в інструкції машинного коду. Середовище CLR також надає інші служби, що відносяться до автоматичного збору сміття, обробки виключень і керуванню ресурсами. Код, виконуваний середовищем CLR, іноді називають "керованим кодом" у протиставлення "некерованому коду", що компілюється в машинний код, призначений для певної системи. Далі показані відносини під час компіляції й час виконання між файлами з вихідним кодом Visual C#, бібліотеками класів .NET Framework, складаннями й середовищем CLR.

Взаємодія між мовами є ключовою особливістю .NET Framework. Оскільки код IL, створюваний компілятором Visual C# відповідає специфікації CTS, код IL

на основі Visual C# може взаємодіяти з кодом, створюваним версіями мов Visual Basic, Visual C++, Visual J# платформи .NET Framework і ще більш ніж 20 CTS-сумісних мов. В одному складанні може бути кілька модулів, написаних на різних мовах платформи .NET Framework, і типи можуть посилатися один на одного, як якби вони були написані на одній мові.

Крім служб часу виконання, в.NET Framework також є велика бібліотека, що складається з більш ніж 4000 класів, організованих по просторах імен, які забезпечують різноманітні корисні функції для будь-яких дій, починаючи від введення й виведення файлів для керуванням рядками для розбивки XML, і закінчуючи елементами керування Windows Forms. У звичайному застосунку мовою Visual C# бібліотека класів .NET Framework інтенсивно використовується для "устрою" коду.

2.3 Розгорнута постановка завдання

Згідно з технічним завданням на випускну кваліфікаційну роботу за першим (бакалаврським) рівнем вищої освіти, реалізації підлягає програмне забезпечення, яке призначено для системи кібербезпеки ЕЦП для автентифікації користувачів у системах дистанційного навчання та тестування.

В процесі розробки випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти необхідно виконати наступний обсяг роботи:

а) провести аналіз існуючих систем-аналогів для виявлення їх позитивних і негативних якостей. Результати аналізу врахувати в подальших розробках;

б) вибрати та обґрунтувати методику побудови системи кібербезпеки контролю роботи технологічного обладнання на виробництві в автоматизованому режимі. Розробити функціональну та структурну схеми системи;

в) розробити програмне забезпечення системи, що дозволить реалізувати поставлену технічним завданням задачу. Побудувати блок-схеми алгоритмів програми та підпрограми;

г) організувати інтерфейс користувача з метою формування та виводу на екран ЕОМ повідомлень про некоректні дії користувача та нестандартні ситуації в роботі технологічного обладнання;

д) розробити рекомендації по організаційних та методичних заходах, які забезпечать впровадження системи кібербезпеки в промислову експлуатацію та її подальшу успішну експлуатацію;

е) провести розрахунки по визначенню економічної ефективності розробленої системи;

ж) розробити заходи по охороні праці при впровадженні та експлуатації системи, а також розробити заходи з цивільного захисту;

з) сформувати висновки про виконаний обсяг робіт та одержані результати.

3 ОПИС І ОБҐРУНТУВАННЯ ПРОЕКТНИХ РІШЕНЬ

3.1 Опис функціонування системи

Впровадження захищеної системи дистанційного навчання й тестування дозволяє забезпечити:

– сукупність технологій, що забезпечують доставку тим, хто навчається, основного обсягу досліджуваного матеріалу;

– інтерактивну взаємодію студентів і викладачів в процесі навчання, надання студентам можливості самостійної роботи з освоєння досліджуваного матеріалу, а також у процесі навчання

– дійсність інформації;

– автоматизацію роботи з документами;

– систематизацію зберігання інформації;

– зменшення кількості паперових документів;

– полегшення роботи користувачів.

Однак, системи дистанційного навчання й тестування, що існують, здебільшого, мають наступні недоліки:

− відсутність юридично значимої ЕЦП;

− низька функціональність схем ЕЦП;

− відсутність можливості спільної роботи з документами;

− відсутність керування потоками робіт;

− відсутність підтримки різних типів даних.

Показано, що при виборі тієї або іншої моделі системи дистанційного навчання й тестування необхідно керуватися наступними критеріями:

− повнота відповідності системи дистанційного навчання й тестування необхідному (або типовому) набору функцій;

− витрати на впровадження системи в існуючу систему дистанційного навчання й тестування і її наступний супровід;

− розширюваність (масштабованість) системи;

− наявність механізмів забезпечення інформаційної безпеки.

У загальному виді сформульовані підходи до усунення недоліків існуючих систем дистанційного навчання й тестування:

− впровадження юридично значимої ЕЦП по схемах ДСТУ 4145-2002 і альтернативним схемам;

− розширення функціональності схем ЕЦП;

− забезпечення розмежування прав доступу;

− повна підтримка життєвого циклу електронного документа системи дистанційного навчання й тестування;

− впровадження можливостей спільної роботи з документами й керування потоками робіт;

− підвищення зручності роботи користувача.

Виконаємо **постановку наукового завдання** дослідження в наступному виді.

Дано:

N – кількість користувачів системи захищеного дистанційного навчання й тестування;

S – структура системи захищеного дистанційного навчання й тестування;

A – безліч алгоритмів криптографічного перетворення;

$K_{инф}$ – категорія информації, що захищається;

Обмеження й допущення:

– імітостійкість інформації $I \geq I_{зад}$;

– продуктивність системи захищеного дистанційного навчання й тестування $P \geq P_{зад}$;

– вартість створення системи захищеного дистанційного навчання й тестування $C \leq C_{зад}$;

– приналежність характеристик використовуваної обчислювальної техніки області припустимих значень $V \in \{V_д\}$.

Знайти: $M=\{N, S, A, K_{инф}\}$ – модель системи захищеного системи дистанційного навчання й тестування, що задовольняє наступному вираженню:

$$M^* = \arg\min_{M \in \Omega_M} \left\{ T\left(M = \{N, S, A, K_{èíô}\}\right) \to \min / \left(P(M) \ge P_{çàä}(M)\right), \left(C(M) \le C_{çàä}(M)\right) \right\},$$

де Ω_M – безліч моделей системи захищеного дистанційного навчання й тестування, T – час виконання обробки інформації в системи захищеного дистанційного навчання й тестування від моменту створення й підписання документа до моменту перевірки підпису.

При розробці методу формування й перевірки електронного цифрового підпису на основі відкритого колективного ключа запропоновано використовувати груповий закон додавання точок на еліптичній кривій (ЕК) $E(GF_p)$ виду $y^2 = x^3 + ax + b \,(\mathrm{mod}\, p)$, де GF_p – кінцеве поле з характеристикою $p \ne 2$ і $p \ne 3$, x, y – координати точок ЕК, a, b – коефіцієнти рівняння ЕК. Груповий закон додавання точок $P_1 \oplus P_2 = (x_3, -y_3)$ для випадку двох різних точок $P_1=(x_1,y_1)$ і $P_2=(x_2,y_2)$ має такий вигляд:

$$x_3 = \left(\frac{y_2 - y_1}{x_2 - x_1}\right)^2 - x_2 - x_1 \,(\mathrm{mod}\, p),$$

$$y_3 = \left(\frac{y_2 - y_1}{x_2 - x_1}\right)(x_1 - x_3) - y_1 \,(\mathrm{mod}\, p),$$

при $P_1=P_2=(x_1,y_1)$:

$$x_3 = \frac{(3x_1^2 + a)^2}{4(x_1^3 + ax_1 + b)} - 2x_1 \,(\mathrm{mod}\, p),$$

$$y_3 = \frac{(3x_1^3 + a)}{2y_1}(x_1 - x_2) - y_1 \,(\mathrm{mod}\, p).$$

Для кривих з ненульовим інваріантом над полем GF_2^n використовуються наступні вираження додавання точок:

$$\begin{cases} x_3 = \lambda^2 + \lambda - a - x_1 - x_2 \\ y_3 = -(\lambda + 1)x_3 - \nu \end{cases},$$

де для випадку $P_1 \neq P_2$:

$$\lambda = \frac{y_2 - y_1}{x_2 - x_1},\ \nu = \frac{y_1 x_2 - y_2 x_1}{x_2 - x_1},$$

при $P_1 = P_2$:

$$\lambda = \frac{3x_1^2 + 2x_1 a}{2y_1 + x_1},\ \nu = \frac{3x_1^3 + 2x_1^2 - 2y_1^2 - y_1 x_1}{2y_1 + x_1}.$$

Обіг точки має вигляд $P_3=(x_3, -y3-x3)$. Аналогічно для кривих з ненульовим інваріантом j над полем GF_3^n :

$$\begin{cases} x_3 = \lambda^2 - a - x_1 - x_2 \\ y_3 = -x_3\lambda - \nu \end{cases},$$

де для $P_1 \neq P_2$:

$$\lambda = \frac{y_2 - y_1}{x_2 - x_1},\ \nu = \frac{y_1 x_2 - y_2 x_1}{x_2 - x_1},$$

при $P_1 = P_2$:

$$\lambda = \frac{3x_1^2 + 2x_1 a}{2y_1} \text{ і } \nu = \frac{3x_1^3 + 2x_1^2 - 2y_1^2}{2y_1}.$$

Наведені групові закони додавання точок на ЕК використовуються як функції криптографічного перетворення. Якщо P і G – елементи циклічної підгрупи A кривій $E(GF_q)$ і G – примітивний елемент (генератор) цієї підгрупи, те, при P=n*G, де n – випадкове число (ключ), пошук числа n по двох заданих елементах підгрупи P і G при $n \to \infty$ є розрахунково складним завданням з погляду теорії складності алгоритмів.

У кільці цілих чисел найбільш трудомісткою операцією є інверсія. З метою виключення даної операції здійснений перехід з афінних координат у

проективні, що забезпечило підвищення швидкості обчислень на 30-40 %.

Розроблено узагальнений метод формування й перевірки електронного цифрового підпису на основі відкритого колективного ключа.

Метод складається із трьох етапів:

I етап – формування електронного цифрового підпису на основі відкритого колективного ключа, полягає у виконанні наступних кроків:

1. Створення першої частини електронного цифрового підпису на основі відкритого колективного ключа R на основі генерації індивідуальних параметрів підпису R_i і застосування функції, що відображає їх у колективний параметр підпису R.

2. Створення секретних ключів і формування часток другої частини підпису S_i.

3. Інтеграція в єдину електронної цифровому підписі на основі відкритого колективного ключа S.

II етап – властиво створення електронного цифрового підпису на основі відкритого колективного ключа, що складає із двох частин (R, S), геш-функції й відправлення електронного цифрового підпису на основі відкритого колективного ключа відповідним користувачам автоматизованої системи керування.

III етап – перевірка електронного цифрового підпису на основі відкритого колективного ключа:

1. По довіднику відкритих ключів вибираються індивідуальні відкриті ключі користувачів Y_i, що брали участь у створенні й підписанні документа.

2. На основі обраних індивідуальних відкритих ключів користувачів формується відкритий колективний ключ Y.

3. Здійснюється перевірка дійсності електронного цифрового підпису на основі відкритого колективного ключа, а потім приймається рішення: прийняти або відхилити електронний цифрового підпису на основі відкритого колективного ключа.

Порівняння характеристик звичайного й електронного цифрового підпису на основі відкритого колективного ключа для m користувачів показує (табл. 3.1), що складність генерації підпису є однакової, а складність перевірки електронного цифрового підпису на основі відкритого колективного ключа в m раз менше.

Таблиця 3.1 – Порівняння характеристик ЕЦП і електронного цифрового підпису на основі відкритого колективного ключа.

Характеристика	Звичайна ЕЦП	ЕЦП на основі відкритого колективного ключа
Складність генерації підпису (кількість перетворень)	m	m
Складність перевірки підпису (кількість перетворень)	2m	2
Функціональність	середня	висока

Таблиця 3.2 – Реалізуємість електронного цифрового підпису на основі відкритого колективного ключа.

Алгоритм ЕЦП	Можливість реалізації електронного цифрового підпису на основі відкритого колективного ключа
ДСТУ 4145-2002	+
DSA (стандарт США 1991 р.)	–
EDSA (стандарт США 1999 р.)	–
Ель-гамаля	–
Шнорра	+

Досліджена реалізуємість електронного цифрового підпису на основі відкритого колективного ключа на основі відомих алгоритмів ЕЦП (таблиця 3.2), показана можливість розробки електронного цифрового підпису на основі

відкритого колективного ключа на основі вітчизняних стандартів і схеми Шнорра.

На основі стандарту ЕЦП України ДСТУ 4145-2002 розроблена **схема формування й перевірки електронного цифрового підпису** на основі відкритого колективного ключа. Вона відповідає узагальненому методу формування електронного цифрового підпису на основі відкритого колективного ключа й полягає в послідовному виконанні наступних етапів і кроків.

I етап – Генерація ключів.

1. Генерація секретних ключів користувачів $d_i < P$.

2.Формування індивідуальних відкритих ключів користувачів $Q_i = d_i * P$, де P – точка ЕК, що є генератором аддитивної циклічної групи точок, і колективного відкритого ключа:

$$Q = Q_1 + Q_2 + \ldots + Q_m ..$$

II етап – Формування електронного цифрового підпису на основі відкритого колективного ключа.

1.Обчислення значення геш-функції H від що підписується ЭД і обчислення допоміжної змінної $e = H \bmod q$.

2.Генерація кожним користувачем значень k_i і обчислення точок ЕК $C_i = k_i * P$.

3. Додавання точок ЕК кожного користувача $C = C_1 + C_2 + \ldots + C_m$ і обчислення першої частини електронного цифрового підпису на основі відкритого колективного ключа через координату x точки C ЕК:

$$R = x \bmod q.$$

4. Формування часток ЕЦП кожного користувача

$S_i = (Rd_i + k_i e) \bmod q$ і обчислення другої частини електронного цифрового підпису на основі відкритого колективного ключа.

$$S = (S_1 + S_2 + \ldots + S_m) \bmod q$$

5. Формування електронного цифрового підпису на основі відкритого колективного ключа у вигляді пари значень (R, S).

III етап – Перевірка електронного цифрового підпису на основі відкритого колективного ключа.

1. Визначення точки ЕК за допомогою значень електронного цифрового підпису на основі відкритого колективного ключа (R, S), допоміжної змінної e, колективного відкритого ключа перевірочного рівняння наступного виду:

$$C' = ((Se-1) \bmod q) * P + ((q-R)e^{-1} \bmod q) * Q.$$

2. Обчислення значення параметра R′ по координаті x точки C′:

$$R' = x' \bmod q.$$

3. Перевірка умови: якщо R′ = R, те підпис вірний.

Розробимо алгоритм вибору параметрів електронного цифрового підпису на основі відкритого колективного ключа.

Як показник стійкості до криптоаналізу електронного цифрового підпису на основі відкритого колективного ключа на основі застосування групового закону додавання точок на ЕК пропонується використовувати асимптотичну оцінку складності алгоритму розкриття в термінах O – символіки. На основі дослідження алгоритмів криптоаналізу показано, що стійкість електронного цифрового підпису на основі відкритого колективного ключа істотно залежить від порядку групи, визначається як складність найкращого алгоритму по визначенню індексу й оцінюється значенням $O\left(\sqrt{\zeta_p}\right)$, де ζ_p – найбільший простий множник порядку групи точок кривій.

Показано, що параметрами, від яких залежить безпека систем ЕЦП у цілому й електронному цифровому підписі на основі відкритого колективного ключа зокрема, на основі ЕК, є:

− вид кінцевого поля;

− характеристика поля й (або) його розширення;

− рівняння ЕК;

− порядок циклічної підгрупи точок ЕК;

− генератор підгрупи точок ЕК.

Виявлено ряд властивостей кривих, при яких істотно зменшується

стійкість, зокрема недоцільно використовувати криві з інваріантом j=0. Знайдено випадки, коли стійкість перетворення на ЕК зменшується до складності знаходження дискретного логарифма в мультиплікативному полі.

Оскільки основним параметром, що визначає стійкість, є величина порядку групи на кривій, був розроблений **алгоритм вибору параметрів електронного цифрового підпису** на основі відкритого колективного ключа, заснований на виборі параметрів ЕК з необхідним порядком групи:

1. Відповідно до вихідних даних з нерівності $q+1-2\sqrt{q} \le \#E(GF_q) \le q+1+2\sqrt{q}$ визначається характеристика p кінцевого поля $GF_p{}^n$ або кільця Z_p і ступінь розширення поля n.

2. Вибирається рівняння ЕК відповідно до характеристики p.

3. Випадковим образом генеруються коефіцієнти a, b рівняння ЕК. Перейти до кроку 4 при p=2 або p=3, і до кроку 5 в іншому випадку.

4. Визначається порядок групи $\#E(GF_q)$ відповідно до вираження $\#E(GF_2)_k = 2^k + 1 - 2^{\frac{k}{2}+1}\cos\left(k\arctan\left(\pm\sqrt{7}\right)\right)$ для поля з p=2 при будь-якому k для $\#E(GF_2)=2$ і $\#E(GF_2)=4$ відповідно, і відповідно до виражень для поля з p=3: $\#E(GF_3)_k = 3^k + 1 - 2\cdot 3^{\frac{k}{2}}\cos\left(k\arctan\left(\pm\sqrt{11}\right)\right)$ для $\forall$ k при $\#E(GF_3)=5$ і $\#E(GF_3)=3$; $\#E(GF_3)_k = 3^k + 1 - 2\cdot 3^{\frac{k}{2}}\cos\left(k\arctan\left(\pm\sqrt{2}\right)\right)$ для $\forall$ k при $\#E(GF_3)=6$ і $\#E(GF_3)=2$; $\#E(GF_3)_k = 3^k + 1 - 2\cdot 3^{\frac{k}{2}}\cos\left(k\arctan\left(\pm 3^{-\frac{1}{2}}\right)\right)$ для $\forall$ k при $\#E(GF_3)=7$ і $\#E(GF_3)=1$; $\#E(GF_3)_k = \begin{cases} 3^k + 1 - 2\cdot(-3)^{\frac{k}{2}} \\ 3^k + 1 \end{cases}$ для $\#E(GF_3)=4$ при k – парному й k – непарному відповідно. Перейти до кроку 6 алгоритму.

5. Визначається порядок групи $\#E(GF_p)$ відповідно до виражень:

$\exists$ (p=2(mod 3) $\cap$ p=5(mod 6) $\cap$ a=0) $\cup$ (p=3(mod 4) $\cap$ b=0) $\Rightarrow$ $\#E(GF_p)=p+1$;

∃ Δ=0 і a,b≠0 ⇒ при {p=2(mod 3) ∩ (p≠1(mod 4) ∪ p=3(mod 4))} ∪ {p=1(mod 3) ∩ (p=1(mod 4) ∪ p≠3(mod 4)} ⇒ #E(GF_p)=p+1±1, якщо b квадратичне відрахування або невирахування;

∃ Δ=0 і a,b≠0 ⇒ при {p=2(mod 3) ∩ (p=1(mod 4) ∪ p≠3(mod 4))} ∪ {p=1(mod 3) ∩ (p≠1(mod 4) ∪ p=3(mod 4)} ⇒ #E(GF_p)=p+1±1, якщо b квадратичне невирахування або відрахування;

∃ a=0, p=1(mod 6) ⇒ #E(GF_q)=p+1+r, де p=d $2-^{de}+f^2$ в Z[ω], $\omega = \left(-1+\sqrt{-3}\right)/2$, d=2 (mod 3), e=0 (mod 3), r=d+e, r=– d-e, r=2 d-e, r=-2d+e, r= d-2e, r=-d+2e, якщо b квадратичне або кубічне відрахування або невирахування;

∃ (b-кубічне відрахування)∩(b-квадратичне відрахування) ⇒ #E(GF_p)=6l;

∃ (b-кубічне відрахування)∩(b-квадратичне невирахування) ⇒ #E(GF_p)=3l;

∃ (b-кубічне невирахування)∩(b-квадратичне відрахування) ⇒ #E(GF_p)=2l;

∃ (b-кубічне невирахування)∩(b-квадратичне невирахування) ⇒ #E(GF_p)=1(mod6);

∃ b=0, p=1(mod 4), – a-квадратичне відрахування ⇒ #E(GF_q)=p+1±2d, #E(GF_q)=0 (mod 4), де $p=d^2+e^2$ в Z[i], d – непарне;

∃ b=0, p=1(mod 4), – a-квадратичне невирахування ⇒ #E(GF_q)=p+1±2e, #E(GF_q)=2 (mod 4), де $p=d^2+e^2$ в Z[i];

Якщо коефіцієнт a (при p=2(mod 6)) або b (при p=3(mod 4) кривій рівні 0, то порядок групи #E(GF_q)=p+1.

6. Перевіряється нездійсненність умови подільності отриманого порядку групи відповідно до вираження $\#E(GF_{p^n}) \left| \left(p^{n^i} - 1 \right) \right.$, де i=1, 2,..., k (якщо властивість подільності виконується, тоді повернутися до кроку 3 алгоритми вибору параметрів електронного цифрового підпису на основі відкритого колективного ключа).

Результатом вибору параметрів електронного цифрового підпису на основі відкритого колективного ключа є рівняння ЕК з порядком групи, що задовольняє необхідної стійкості й швидкості перетворення.

Застосування даного алгоритму дозволяє здійснювати пошук рівняння ЕК для створення електронного цифрового підпису на основі відкритого колективного ключа з необхідним рівнем стійкості за кінцеве число кроків зі складністю $O(\log^5 p)$.

Розробимо методику організації захищеної системи дистанційного навчання й тестування й обґрунтуванню практичних рекомендацій із програмно-апаратної реалізації електронного цифрового підпису на основі відкритого колективного ключа.

Методика організації захищеного системи дистанційного навчання й тестування полягає у виконанні наступних взаємопов'язаних дій.

1. Створення центра, що засвідчує, у структурі захищеної системи дистанційного навчання й тестування для додання легітимності і юридичної значимості.

2. Визначення порядку підключення користувача до захищеної системи дистанційного навчання й тестування й допуску користувача до здійснення роботи.

3. Розробка вимог, пропонованих до електронного документа.

4. Розробка вимог і організація процесів системи дистанційного навчання й тестування в захищеній системі дистанційного навчання й тестування (формування, відправлення, доставка, перевірка дійсності, підтвердження одержання, відкликання, облік, зберігання електронних документів захищеної системи дистанційного навчання й тестування).

5. Визначення правил використання ЕЦП і електронного цифрового підпису на основі відкритого колективного ключа в захищеній системі дистанційного навчання й тестування.

6. Розробка вимог і визначення порядку створення криптографічних ключів, видачі електронних цифрових сертифікатів, дій при компрометації ключів.

7. Визначення зобов'язань власників цифрових сертифікатів.

8. Розробка правил дій при дозволі конфліктних ситуацій і споровши, що виникли у зв'язку зі здійсненням системи дистанційного навчання й тестування.

Складовою частиною запропонованої методики є розроблений програмний комплекс, у якому реалізований метод формування й перевірки електронного цифрового підпису на основі відкритого колективного ключа й алгоритм вибору параметрів електронного цифрового підпису на основі відкритого колективного ключа. Його можливості включають формування ЕЦП і електронного цифрового підпису на основі відкритого колективного ключа, перевірку ЕЦП і електронного цифрового підпису на основі відкритого колективного ключа, генерацію ключів підпису й перевірки, генерацію еліптичній кривій, параметри якої задовольняють всім вимогам стандарту на ЕЦП. Час формування одного підпису і її перевірок не перевищує 0,2 секунди. Час генерації параметрів електронного цифрового підпису на основі відкритого колективного ключа становить кілька мінут.

Призначення програмного комплексу – забезпечення цілісності й авторства збереженої й оброблюваної інформації в системі захищеного системи дистанційного навчання й тестування на основі застосування систем електронного цифрового підпису на основі відкритого колективного ключа.

Проведено дослідження безпеки електронного цифрового підпису на основі відкритого колективного ключа в порівнянні із системами ЕЦП подібного й іншого видів. Показано, що виграш у показнику безпеки при використанні адитивної групи точок еліптичних кривих у порівнянні з використанням мультиплікативної групи кільця цілих чисел залежить від довжини ключа й може досягати декількох порядків при рівній довжині ключа. Наприклад, стійкість електронного цифрового підпису на основі відкритого колективного ключа на основі ЕК з довжиною ключа 200 біт відповідає стійкості ЕЦП на основі складності рішення завдання дискретного логарифмування в мультиплікативній групі кільця цілих чисел або складності розкладання більших чисел на прості співмножники з довгої ключа порядку 500 біт. Перевага використання ЕК при

створенні електронного цифрового підпису на основі відкритого колективного ключа підтверджується також результатами дослідження швидкості криптографічного перетворення.

3.2 Розробка структурної схеми

Механізм ЕЦП

Електронний цифровий підпис накладається за допомогою особистого ключа та перевіряється за допомогою відкритого ключа. За правовим статусом він прирівнюється до власноручного підпису (печатки). Електронний підпис не може бути визнаний недійсним лише через те, що він має електронну форму або не ґрунтується на посиленому сертифікаті ключа. За умови правильного зберігання власником секретного (особистого) ключа його підробка неможлива. Електронний документ також не можливо підробити: будь-які зміни, не санкціоновано внесені в текст документу системи дистанційного навчання та тестування, будуть миттєво виявлені.

Особистий ключ ЕЦП

Особистий ключ ЕЦП формується на підставі абсолютно випадкових чисел, що генеруються датчиком випадкових чисел, а відкритий ключ обчислюється з особистого ключа ЕЦП так, щоб одержати другий з першого було неможливо. Особистий ключ ЕЦП є унікальною послідовністю символів довжиною 264 біта, яка призначена для створення Електронного цифрового підпису в електронних документах. Працює особистий ключ тільки в парі з відкритим ключем. Особистий ключ необхідно зберігати в таємниці, адже будь-хто, хто дізнається його, зможе підробити Електронний цифровий підпис.

Документ підписується ЕЦП за допомогою особистого ключа ЕЦП, який існує в одному екземплярі тільки у його власника. Цьому особистому ключу відповідає відкритий ключ, за допомогою якого можна перевірити відповідність ЕЦП її власнику.

Відкритий ключ ЕЦП і Сертифікат відкритого ключа

Відкритий ключ використовується для перевірки ЕЦП одержуваних документів (файлів). Відкритий ключ працює тільки в парі з особистим ключем. Відкритий ключ міститься в Сертифікаті відкритого ключа, і підтверджує приналежність відкритого ключа ЕЦП певній особі. Крім самого відкритого ключа, Сертифікат відкритого ключа містить в собі персональну інформацію про його власника (ім'я, реквізити), унікальний реєстраційний номер, термін дії Сертифікату відкритого ключа. З метою забезпечення цілісності представлених у Сертифікаті даних він підписується особистим ключем Центру сертифікації ключів. Сертифікат відкритого ключа може публікуватися на сайті відповідного ЦСК відповідно до Договору про надання послуг ЕЦП.

Підписання електронного документу системи дистанційного навчання та тестування ЕЦП

При підписанні електронного документу системи дистанційного навчання та тестування його початковий зміст не змінюється, а додається блок даних, так званий Електронний цифровий підпис. Отримання цього блоку можна розділити на два етапи:

На першому етапі за допомогою програмного забезпечення і спеціальної математичної функції обчислюється так званий «відбиток повідомлення» (message digest).

Цей відбиток має такі особливості:

− фіксовану довжину, незалежно від довжини повідомлення;

− унікальність відбитку для кожного повідомлення;

− неможливість відновлення повідомлення по його відбитку.

Таким чином, якщо документ був модифікований, то зміниться і його відбиток, що відобразиться при перевірці Електронного цифрового підпису.

На другому етапі відбиток документу системи дистанційного навчання та тестування шифрується за допомогою програмного забезпечення і особистого ключа автора.

Розшифрувати ЕЦП і одержати початковий відбиток, який відповідатиме документу системи дистанційного навчання та тестування, можна тільки використовуючи Сертифікат відкритого ключа автора.

Таким чином, обчислення відбитку документу системи дистанційного навчання та тестування захищає його від модифікації сторонніми особами після підписання, а шифрування особистим ключем автора підтверджує авторство документу системи дистанційного навчання та тестування.

Перевірка ЕЦП одержаного документу системи дистанційного навчання та тестування

Перевірка Електронного цифрового підпису одержаного документу системи дистанційного навчання та тестування проводиться декількома етапами:

На першому етапі адресат за допомогою програмного забезпечення Сертифікатом відкритого ключа автора розшифровує підписаний відбиток і одержує відбиток початкового документа. За допомогою програмного забезпечення і спеціальної математичної функції з документу системи дистанційного навчання та тестування, який був одержаний, обчислюється його відбиток. При перевірці ЕЦП порівнюються відбитки початкового і одержаного документів. Результат перевірки – одна з відповідей: «вірний»/«невірний».

Властивості інформації

Електронний цифровий підпис підтверджує достовірність і цілісність документа. Якщо в документ в процесі пересилки були внесені які-небудь зміни, нехай навіть зовсім незначні, то підміна виявиться. Сертифікат відкритого ключа містить персональну інформацію про власника, що дозволяє однозначно ідентифікувати автора документу системи дистанційного навчання та тестування.

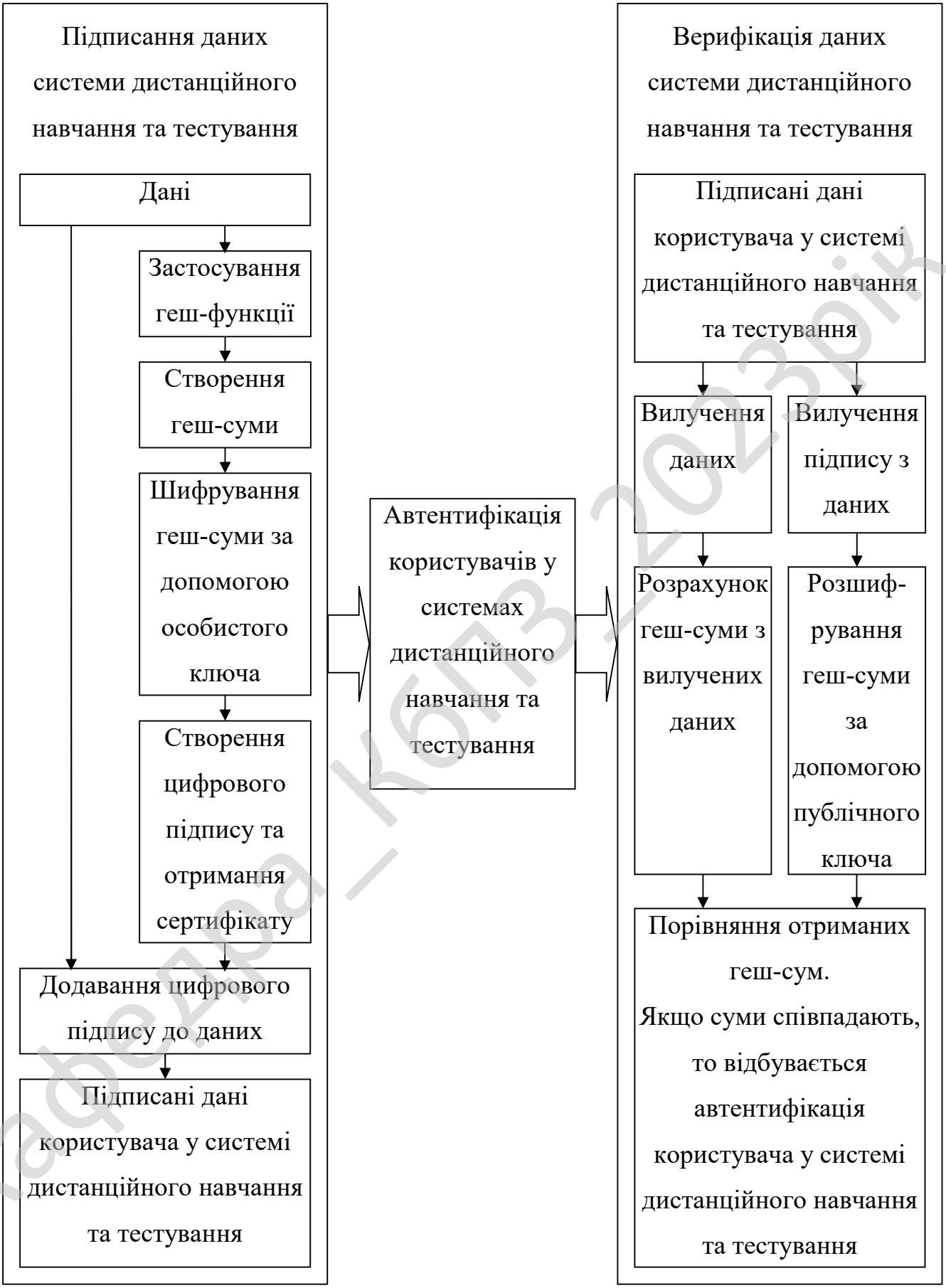

Рисунок 3.1 – Структурна схема системи

Фіксація точного часу підписання

Однією з додаткових можливостей при роботі з ЕЦП є послуга фіксації точного часу підписання документа ЕЦП відмітка точного часу. Відмітка точного часу при підписанні документу системи дистанційного навчання та тестування дозволяє точно ідентифікувати момент накладання підпису, причому змінити його значення згодом, навіть особою, яка наклала підпис, неможливо. Можливе лише повторне підписання з фіксацією нового часу. Точне значення часу, який використовується для формування відмітки точного часу, здійснюється апаратними засобами Центру сертифікації ключів шляхом синхронізації з джерелами точного часу з точністю до 1 секунди.

Обмеження використання національного ЕЦП

Термін валідності ЕЦП

На цей час, відповідно до чинного законодавства, позначка часу не є обов'язковим атрибутом електронного документу системи дистанційного навчання та тестування, підписаного електронним цифровим підписом. Цей факт обмежує використання національного ЕЦП тільки для підпису документів, що валідні протягом дії сертифікату ЕЦП, яким було підписано документ.

Критична вразливість

Чинне законодавство не визначає особливості застосування ЕЦП, щодо документів, термін дії яких перевищує термін дії ЕЦП. Також не визначено статус підписаних документів, термін дії яких не закінчився, у разі компрометації ЕЦП. Це дозволяє реалізувати два види атак на ЕЦП:

− використання недійсного ЕЦП (скомпрометованого, або ЕЦП, термін дії якого закінчився) для підпису документів заднім числом;

− визнання підписаного документу системи дистанційного навчання та тестування без позначки часу, сертифікат якого на час перевірки підпису не діє, недійсним на підставі того, що неможливо встановити чи був документ підписаний дійсним ЕЦП, чи був підписаний заднім числом недійсним ЕЦП. Ця

атака може супроводжуватись брехливою заявою про компрометацію ключа ЕЦП.

Ця вразливість позбавляє змісту такі послуги сертифікаційних центрів, як призупинення дії ЕЦП або реєстрація компрометації ключа ЕЦП.

Головною помилкою, що призвела до з'явлення вразливості, є сприйняття інфраструктури ЕЦП обмеженою відношеннями двох сторін, що перевіряють підпис на момент складання документу системи дистанційного навчання та тестування. При цьому не враховується роль арбітра при виникненні спорів, щодо підписаного документу системи дистанційного навчання та тестування. Тобто валідність підписаного документа розглядається у статиці, а має розглядатися у динаміці.

Атака підпису заднім числом була успішно змодельована з використанням чинного ЕЦП і сертифікованого ПЗ переведенням системного годинника комп'ютера назад.

Обмеження придатності національного ЕЦП для електронного документообігу

Враховуючи наявність такої критичної вразливості, національний електронний документообіг, у якому не застосовується позначка часу, обмежується підписанням документів, валідність яких перевіряється тільки на момент підпису. Прикладом таких документів є подача електронної звітності.

Щодо електронного цифрового підпису довгострокових документів, то кожний такий документ може буди визнаний недійсним навіть протягом терміну валідності ЕЦП за наступним алгоритмом:

– при виникненні спорів щодо підписаного документу системи дистанційного навчання та тестування сторона, що зацікавлена у визнанні документу системи дистанційного навчання та тестування недійсним, подає заяву про компрометацію ключа, наприклад, у зв'язку з наявністю вірусів на комп'ютері де використовується ЕЦП, або за фактом наявності на цьому комп'ютері програмного забезпечення, що надає можливість несанкціонованого доступу;

– при початку судового процесу, який має встановити валідність підписаних документів, стверджувати, що документ був складений після факту компрометації ключа ЕЦП особою – викрадачем ключа і підписаний заднім числом;

– продемонструвати можливість підписання документу системи дистанційного навчання та тестування заднім числом.

Висновок: при чинній законодавчій базі і засобах ЕЦП, що використовуються сьогодні, національний ЕЦП непридатний для електронного документообігу у широкому змісту цього терміну.

Послуги з надання ЕЦП

Послуги з надання ЕЦП в Україні впроваджуються акредитованими центрами сертифікації ключів

Актуальний перелік акредитованих центрів сертифікації ключів публікується на сайті Центрального засвідчувального органу (Акредитовані ЗЦ та ЦСК).

3.3 Розробка функціональної схеми

Функціональна схема розробленої системи зображена на рисунку 3.2.

З рисунку видно, що розроблена система складається з наступних частин:

– Інтерфейс користувача системи дистанційного навчання та тестування.

– Блок формування, розподілу, та верифікації Системи кібербезпеки ЕЦП для автентифікації користувачів у системах дистанційного навчання та тестування.

– Адміністратор.

– Викладач.

– Студент.

В програмному забезпеченні Системи кібербезпеки ЕЦП для автентифікації користувачів у системах дистанційного навчання та тестування визначено 3 типи користувачів (адміністратори, викладачі та студенти).

Система надає різним категоріям користувачів такі функціональні можливості:

Адміністраторам:

– Керування користувачами. Можливість керування користувачами системи, та їх правами.

– Керування загальними параметрами системи. Можливість керування загальними параметрами системи, зокрема темами оформлення, мовою інтерфейсу тощо.

– Керування курсами. Можливість керування курсами системи, резервними копіями.

Викладачам:

– Навчальний курс. Викладачі мають можливість створювати навчальні курси в межах системи, визначати права доступу до них та інші властивості.

– Матеріал. Створення навчальних матеріалів у навчальному курсі з використанням вбудованого редактора матеріалів, керування навчальними матеріалами (структура, період доступу), та перегляд статистики використання матеріалів. Можливість експорту та імпорту навчальних матеріалів у формат обміну навчальними матеріалами SCORM.

– Тести. Широкі можливості щодо створення і керування тестами, запитаннями, організація бази даних питань курсу, попередній перегляд тестів, перегляд спроб складання тестів користувачами, можливість їх оцінювання, перегляд статистики по тестах.

– Запис на курс, групи. Керування записом на курс, перегляд записаних на курс студентів та керування їх правами у межах курсу. Можливість призначення асистентів та випускників курсу. Створення груп у межах курсу та керування ними.

– Електронна пошта курсу. Дозволяє розсилати повідомлення різним категоріям студентів: усім зареєстрованим у даному курсі, тільки привілейованим студентам, випускникам, тим, кому в запису на курс було відмовлено, або студентам окремих груп.

– Резервна копія курсу. Можливість створення резервних копій курсу, відновлення курсу з резервної копії.

– Оголошення. Дає можливість додавати, видаляти та редагувати оголошення для студентів курсу. Оголошення відображаються на домашній сторінці курсу і можуть розсилатися через RSS (якщо така функція увімкнена у властивостях курсу).

– Опитування. За допомогою цього інструменту можна організовувати неоцінювані опитування студентів з метою з'ясування їх думки з тих чи інших питань.

– Словник. Цей пункт дозволяє вводити і редагувати словникові терміни. Терміни, які використовуються в матеріалі, легше вводити через редактор матеріалу.

– Список літератури. Цей засіб дає можливість вказувати список джерел, обов'язковість та термін ознайомлення з ними.

– Статистика. Цей інструмент показує дані про те, як користуються курсом студенти та незареєстровані користувачі.

– Файловий менеджер. Завантаження на сервер необхідних навчальних матеріалів, наприклад, текстів лекцій, практичних занять, тощо у різноманітних форматах (Microsoft Word, PDF, DJVU) з наступним використанням у навчальних матеріалах. Передбачена можливість пакетного завантаження файлів.

Рисунок 3.2 – Функціональна схема системи

Студентам:

– Використання навчальних курсів. Студент має можливість переглядати в повному об'ємі інформацію у навчальному курсі, на який він записаний, з

можливістю пакетного завантаження навчальних матеріалів, якщо це дозволено інструктором курсу.

– Перегляд додаткових розділів навчального курсу, наприклад, “Список літератури ”, “Словник” тощо.

– Тестування та опитування. Студенти в рамках навчального курсу можуть проходити тестування або анонімні опитування, переглядати результати тестувань.

– Засоби спілкування. Система дистанційного навчання володіє такими засобами зв’язку між учасниками навчального процесу:

а) синхронними (чати, телеконференції, дошки (whiteboards));

б) асинхронними (оголошення, форуми, внутрішні повідомлення, електронна пошта, блоги, вікі, коментарі в файлообміннику).

– Групи та Файлообмінник. Студенти можуть завантажувати та обмінюватись файлами в рамках навчального курсу або своєї групи.

– Пошук. Ефективна система пошуку в межах навчального курсу, всіх курсів та зовнішніх джерел інформації (пошук по TILE).

– Редагування персональної інформації. Студент має можливість редагувати персональну інформацію, включаючи можливість завантаження власного фото, зміни паролю та адреси електронної пошти.

– Перегляд існуючих курсів та запис на них. Студент може переглядати список курсів, відправляти запит на отримання прав доступу до них.

3.4 Розробка діаграми процесів

Діаграма процесів розробленої системи зображена на рисунку 3.3. першим процесом, який завантажується у системі, є процес виведення головного вікна. Він взаємодіє з наступними процесами:

– Процес автентифікації адміністратора/викладача.

– Процес автентифікації студента.

Рисунок 3.3 – Діаграма взаємодії процесів

Процес автентифікації адміністратора/викладача взаємодіє з процесом одержання файлу з відповідями на тест, який, у свою чергу взаємодіє з наступними процесами:

– Процес дешифрування файлу.

– Процес перевірки відповідей (у разі дійсного підпису), який взаємодіє з процесом надсилання студенту зашифрованого файлу з результатами, підписаного ЕЦП перевіряючого.

Процес дешифрування файлу взаємодіє з процесом перевірки ЕЦП, який взаємодіє з процесом обчислення геш-значеня файлу, який взаємодіє з процесом дешифрування гешу відкритим ключем студента, який взаємодіє з процесом порівняння значень та надання висновку про дійсність підпису.

Процес автентифікації студента взаємодіє з наступними процесами:

– Процес створення закритих та відкритих ключів.

– Процес проходження навчання.

– Процес проходження тестування.

Останній процес взаємодіє з процесом створення файлу з відповідями, який взаємодіє з процесом створення ЕЦП, який взаємодіє з процесом обчислення геш-функції, який взаємодіє з процесом шифрування геш-функції закритим ключем, який взаємодіє з процесом додавання підпису до файлу, який взаємодіє з процесом шифрування файлу, який взаємодіє з процесом відправки файлу з відповідями на перевірку, який взаємодіє з процесом одержання результатів перевірки, який взаємодіє з процесом дешифрування та перевірки ЕЦП файлу з результатами.

Таким чином, розглянувши опис системи, структурну, функціональну схеми системи, та діаграму взаємодії процесів перейдемо до опису блок-схем основної програми, та підпрограм, які використовуються, для реалізації системи.

4 РЕАЛІЗАЦІЯ РОБОТИ. РОЗРАХУНКИ І ЕКСПЕРИМЕНТАЛЬНІ ДАНІ, ЩО ПІДТВЕРДЖУЮТЬ ВІРНІСТЬ ПРОЕКТНИХ ТА ПРОГРАМНИХ РІШЕНЬ

4.1 Блок-схеми та опис алгоритмів функціонування системи

Розроблене програмне забезпечення дозволяє автентифікувати користувачів в системах дистанційного навчання та тестування. Досягається це за допомогою електронного цифрового підпису заснованого на еліптичних кривих.

На рисунку 4.1 показана блок-схема роботи клієнтського додатку.

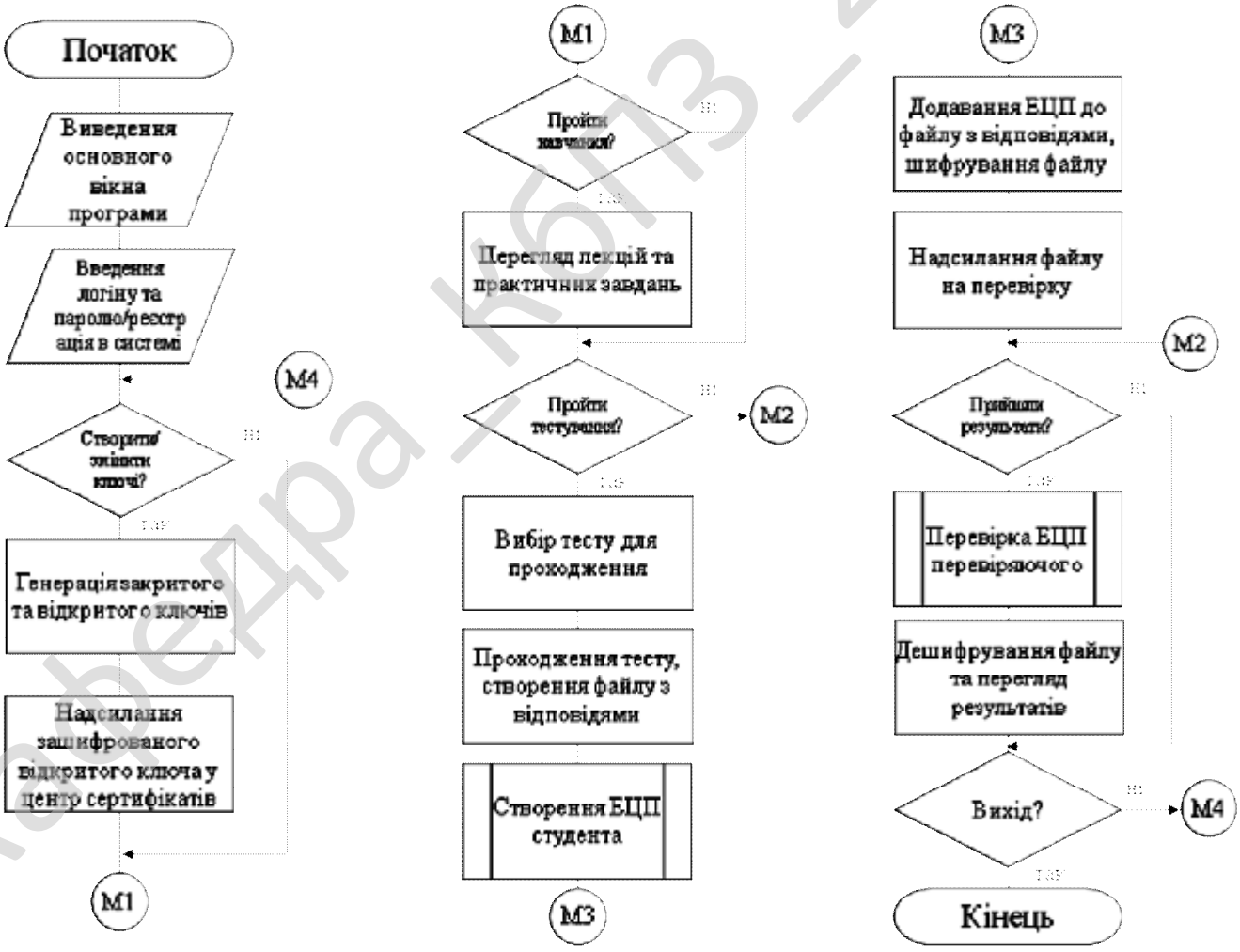

Рисунок 4.1 – Блок-схема основної програми клієнтського додатку

Як видно з рисунку алгоритм роботи додатку складається з виконання наступної послідовності кроків:

Крок 1. Виведення основного вікна програми та введення логіну та паролю, або, за необхідності, реєстрація в системі нового користувача.

Крок 2. Якщо немає ключів шифрування, або їх треба змінити (що бажано робити раз на місяць), відбувається генерація нових закритого та відкритого ключів для ЕЦП, шифрування відкритого ключа та відправка його до центру сертифікатів.

Крок 3. Якщо користувач обирає проходження навчання, то виводиться список курсів та матеріалів до них. Користувач обирає бажаний курс та переглядає наявні матеріали.

Крок 4. Після проходження навчання, за бажанням користувача, він може спробувати пройти тестування. В такому разі користувач обирає бажаний тест і проходить його.

Крок 5. Після проходження тесту програма формує файл з відповідями, підписує його цифровим підписом користувача з використанням згенерованого закритого ключа та шифрує весь файл разом з підписом.

Крок 6. Користувач надсилає підписаний зашифрований файл з відповідями на перевірку.

Крок 7. Після того як тест було перевірено, користувачу приходить зашифрований та підписаний цифровим підписом перевіряю чого файл з результатами тестування.

Крок 8. Програма дешифрує файл, перевіряє ЕЦП перевіряючого і, у разі дійсності цифрового підпису, виводить на екран результати тестування та заносить їх до бази даних користувача.

Розроблене програмне забезпечення основується на ДСТ 34.10-2001. Цей алгоритм був прийнятий замість алгоритму ДСТ 34.10-94. Від усім відомого RSA алгоритм відрізняється складною проблемою, якщо RSA використовує складну проблему розкладання числа на множники, то ДСТУ 4145-2002 заснований на

еліптичних кривих і використовує проблему обчислення дискретного логарифма в групі точок. Це досить новий напрямок математики, що відкриває широкі перспективи.

Криптографічні перетворення на основі еліптичних кривих

У загальному випадку рівняння **еліптичної кривої** E має вигляд:

$$y^2 + axy + by = x^3 + cx^2 + dx + e$$

Як приклад розглянемо еліптичну криву E, рівняння якої має вигляд:

$$y^2 + y = x^3 - x^2$$

На цій кривій лежать тільки чотири точки, координати яких є цілими числами. Це точки A (0, 0), B (1, -1), C (1, 0) та D (0, -1).

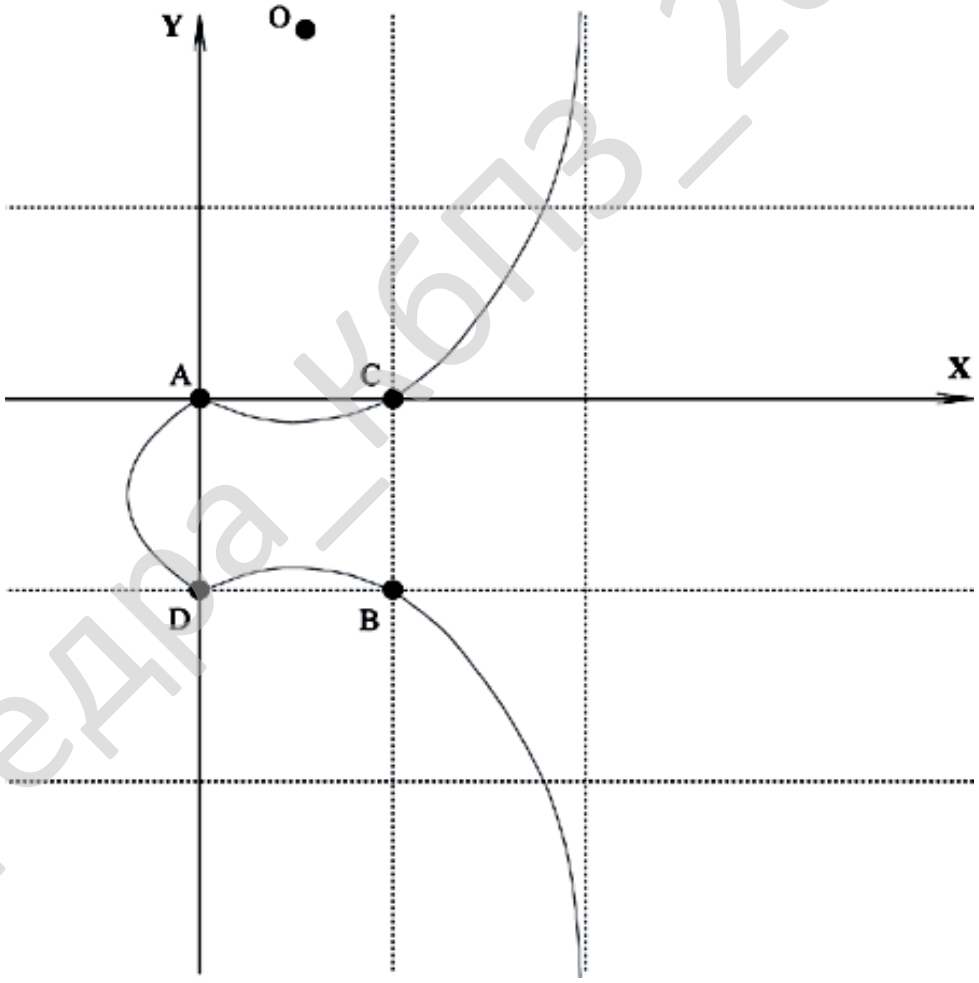

Рисунок 4.2 – Приклад еліптичної кривої із чотирма точками

Для визначення операції додавання для точок на еліптичній кривій зробимо наступні припущення:

– На площині існує нескінченно віддалена точка $0 \in E$, у якій сходяться всі вертикальні прямі.

– Будемо вважати, що дотична до кривої перетинає точку торкання два рази.

– Якщо три точки еліптичної кривої лежать на прямій лінії, то їх сума є 0.

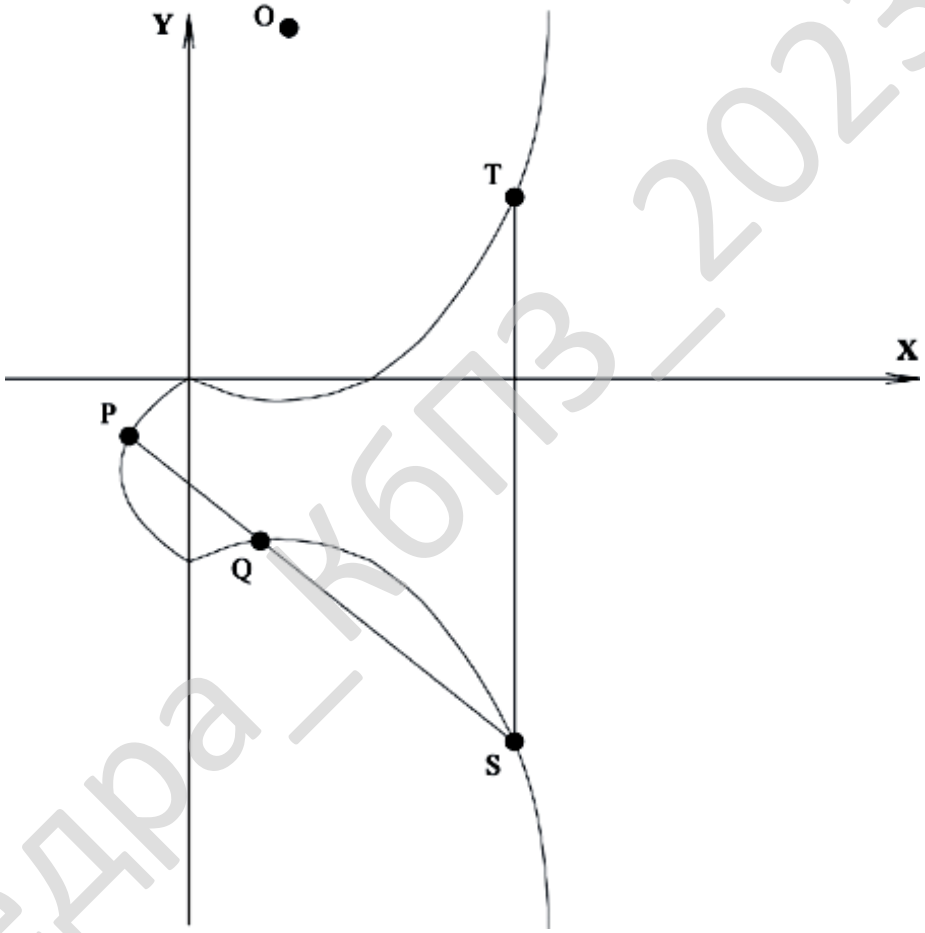

Рисунок 4.3 – Додавання точок на еліптичній кривій

Введемо наступні правила додавання точок на еліптичній кривій:

– Точка 0 виступає в ролі нульового елемента. Так, 0 = -0 і для будь-якої точки P на еліптичній кривій P + 0 = P.

– Вертикальна лінія перетинає криву у двох точках з однією й тією ж координатою x – скажімо, S = (x, y) та T = (x, -y). Ця пряма перетинає криву й у нескінченно віддаленій точці. Тому $P_1 + P_2 + 0 = 0$ та $P_1 = -P_2$.

– Щоб скласти дві точки P і Q (див. рис. 4.3) з різними координатами x, варто провести через ці точки пряму й знайти точку перетину її з еліптичної кривої. Якщо пряма не є дотичною до кривої в точках P або Q, то існує тільки одна така точка, позначимо її S. Відповідно до нашого припущення:

$$P + Q + S = O,$$

отже,

$$P + Q = -S,$$

або

$$P + Q = T.$$

Якщо пряма є дотичною до кривої в який-небудь із точок P або Q, то в цьому випадку варто покласти S = P або S = Q відповідно.

– Щоб подвоїти точку Q, варто провести дотичну в точці Q і знайти іншу точку перетинання S з еліптичної кривої. Тоді:

$$Q + Q = 2 \text{ x } Q = -S.$$

Уведена в такий спосіб операція додавання підкоряється всім звичайним правилам додавання, зокрема комутативному й асоціативному законам. Множення точки P еліптичної кривої на позитивне число k визначається як сума k точок P.

У криптографії з використанням еліптичних кривих всі значення обчислюються за модулем p, де p є простим числом. Елементами даної еліптичної кривої є пари ненегативних цілих чисел, які менше p і задовольняють частковому виду еліптичної кривої:

$$y^2 \equiv x^3 + ax + b(mod\ p)$$

Таку криву будемо позначати E_p (a,b). При цьому числа a й b повинні бути менше p і повинні задовольняти умові $4a^3 + 27b^2(mod\ p) \neq 0$. Множина точок на еліптичній кривій обчислюється в такий спосіб.

1. Для кожного такого значення x, що 0 <= x <= p, обчислюється:

$$x^3 + ax + b \pmod p.$$

2. Для кожного з отриманих на попередньому кроці значень з'ясовується, чи має це значення квадратний корінь за модулем p. Якщо ні, то в $E_p(a,b)$ немає точок з цим значенням x. Якщо корінь існує, є два значення y, що відповідає операції добування квадратного кореня (виключенням є випадок, коли єдиним значенням виявляється y = 0). Ці значення (x,y) і будуть точками E_p (a,b).

Множина точок E_p (a,b) має наступні властивості:

1. P + 0 = P

2. Якщо P = (x,y), то P + (x,-y) = 0. Точка (x,-y) є негативним значенням точки P и позначається -P. Помітимо, що (x,-y) лежить на еліптичній кривій і належить E_p (a,b).

3. Якщо $P = (x_1,y_1)$ і $Q = (x_2,y_2)$, де $P \neq Q$, то $P + Q = (x_3,y_3)$ визначається за наступними формулами:

$$x_3 \equiv \lambda^2 - x_1 - x_2 (mod\ p)$$
$$y_3 \equiv \lambda(x_1 - x_3) - y_1 (mod\ p)$$

де

$$\lambda = \begin{cases} (y_2 - y_1)/(x_2 - x_1), & \text{если } P \neq Q \\ (3x_1^2 + a)/2y_1, & \text{если } P = Q \end{cases}$$

Число λ є кутовий коефіцієнт січної, проведеної через точки $P = (x_1, y_1)$ і $Q = (x_2, y_2)$. При P = Q січна перетворюється в дотичну, чим і пояснюється наявність двох формул для обчислення λ.

Завдання, що повинен вирішити в цьому випадку атакуючий, є свого роду завдання **"дискретного логарифмування на еліптичній кривій"**, і формулюється воно в такий спосіб. Дано точки P і Q на еліптичній кривій E_p (a,b). Необхідно знайти коефіцієнт k < p такий, що:

$$P = k \times Q$$

Відносно легко обчислити P по даним k і Q, але досить важко обчислити k, знаючи P і Q.

Розглянемо три способи використання еліптичних кривих у криптографії.

Аналог алгоритму Діфі-Хелмана обміну ключами

Обмін ключами з використанням еліптичних кривих може бути виконаний у такий спосіб. Спочатку вибирається просте число $p \approx 2^{180}$ і параметри a і b для рівняння еліптичної кривої. Це задає множина точок E_p (a,b). Потім в E_p (a,b) вибирається генеруюча точка G = ($_x$1,$_y$1). При виборі G важливо, щоб найменше значення n, при якому n x G = 0, виявилося дуже великим простим числом. Параметри E_p (a,b) і G криптосистеми є параметрами, відомими всім учасникам.

Обмін ключами між користувачами А та В здійснюється за наступною схемою.

1. Учасник А вибирає ціле число n, менше n. Це число є закритим ключем учасника А. Потім учасник А обчислює відкритий ключ P_A = n x G, що являє собою деяку точку на E_p(a,b).

2. Точно так само учасник У вибирає закритий ключ n і обчислює відкритий ключ P_B.

3. Учасники обмінюються відкритими ключами, після чого обчислюють спільний секретний ключ K:

Учасник А: K = n x P_B;

Учасник В: K = n_B x P_A.

Варто помітити, що спільний секретний ключ являє собою пару чисел. Якщо даний ключ передбачається використовувати в якості сеансового ключа для алгоритму симетричного шифрування, то із цієї пари необхідно створити одне значення.

Алгоритм цифрового підпису на основі еліптичних кривих ECDSA

Алгоритм ECDSA (Elliptic Curve Digest Signature Algorithm) прийнятий як стандарти ANSI X9F1 і IEEE P1363.

Створення ключів:

1. Вибирається еліптична крива E_p (a,b). Число точок на ній повинне ділитися на велике ціле n.

2. Вибирається точка $P \in E_p(a,b)$.

3. Вибирається випадкове число $d \in [1, n-1]$.

4. Обчислюється Q = d x P.

5. Закритим ключем є d, відкритим ключем – (E, P, n, Q).

Створення підпису:

1. Вибирається випадкове число $k \in [1, n-1]$.

2. Обчислюється k x P = (x_1, y_1) та r = x_1 (mod n).

Перевіряється, щоб r не було рівне нулю, тому що в цьому випадку підпис не буде залежати від закритого ключа. Якщо r = 0, то вибирається інше випадкове число k.

3. Обчислюється k-1 mod n

4. Обчислюється s = k-1 (H(M) + dr) (mod n)

Перевіряється, щоб s не було рівне нулю, тому що в цьому випадку необхідного для перевірки підпису числа s-1 mod n не існує. Якщо s = 0, то вибирається інше випадкове число k.

5. Підписом для повідомлення M є пара чисел (r,s).

Перевірка підпису:

1. Перевірити, що цілі числа r і s належать діапазону чисел [0, n-1]. У противному випадку результат перевірки негативний, і підпис відкидається.

2. Обчислити w = s-1 (mod n) і H(M)

3. Обчислити u_1 = H(M) w (mod n) та u_2 = rw (mod n)

4. Обчислити u_1P + u_2Q = (x_0, y_0) та v = x_0 (mod n)

5. Підпис вірний у тому і тільки тому випадку, коли v = r

Шифрування/дешифрування з використанням еліптичних кривих

Розглянемо найпростіший підхід до шифрування/дешифруванню з використанням еліптичних кривих. Завдання полягає в тому, щоб зашифрувати повідомлення M, що може бути представлене у вигляді точки на еліптичній кривій P_m (x,y).

Як і у випадку обміну ключами, у системі шифрування/дешифрування як параметри розглядається еліптична крива E_p (a,b) і точка G на ній. Учасник B

вибирає закритий ключ n і обчислює відкритий ключ $P_B = n \times G$. Щоб зашифрувати повідомлення P_m використовується відкритий ключ одержувача B P_B. Учасник A вибирає випадкове ціле позитивне число k і обчислює зашифроване повідомлення C_m, що є точкою на еліптичній кривій.

$$C_m = \{k \times G, P_m + k \times P_B\}$$

Щоб дешифрувати повідомлення, учасник B множить першу координату точки на свій закритий ключ і віднімає результат із другої координати:

$$P_m + k \times P_B - n \times (k \times G) = P_m + k \times (n \times G) - n \times (k \times G) = P_m$$

Учасник A шифрує повідомлення P_m додаванням до нього kxP_B. Ніхто не знає значення k, тому, хоча P_B і є відкритим ключем, ніхто не знає $k \times P_B$. Супротивникові для відновлення повідомлення прийдеться обчислити k, знаючи G і $k \times G$. Зробити це буде нелегко.

Одержувач також не знає k, але йому як підказка посилається $k \times G$. Помноживши $k \times G$ на свій закритий ключ, одержувач одержить значення, що було додане відправником до незашифрованого повідомлення. Тим самим одержувач, не знаючи k, але маючи свій закритий ключ, може відновити незашифроване повідомлення.

Принцип роботи ДСТУ 4145-2002

Після підписання повідомлення M до нього дописується цифровий підпис розміром 512 біт і текстове поле. У текстовому полі можуть утримуватися, наприклад, дата й час відправлення або різні дані про відправника:

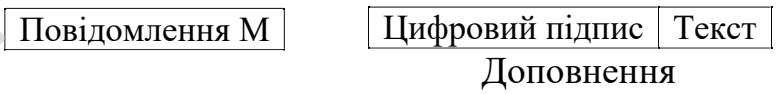

Рисунок 4.4 – Загальний формат електронного цифрового підпису

Даний алгоритм не описує механізм генерації параметрів, необхідних для формування підпису, а тільки визначає, яким чином на підставі таких параметрів

одержати цифровий підпис. Механізм генерації параметрів визначається на місці залежно від розроблювальної системи.

Параметри схеми цифрового підпису

– просте число p – модуль еліптичної кривої такий, що $p > 2^{255}$

– еліптична крива E задається своїм інваріантом $J(E)$ або коефіцієнтами $a, b \in F_p$, де F_p – кінцеве поле з p елементів. $J(E)$ пов'язаний з коефіцієнтами a і b у такий спосіб:

$$J(E) = 1728\frac{4a^3}{4a^3 + 27b^2} \pmod p,$$

причому

$$4a^3 + 27b^2 \not\equiv 0 \pmod p.$$

– ціле число m – порядок групи точок еліптичній кривій, m повинне бути відмінно від p

– просте число q, порядок деякої циклічної підгрупи групи точок еліптичної кривої, тобто виконується $m = nq$, для якогось $n \in \mathbb{N}$. Так само qлежить у межах $2^{254} < q < 2^{256}$.

– точка $P = (x_P, y_P)$ еліптичній кривій E, що є генератором підгрупи порядку q, тобто $q \cdot P = \mathbf{0}$ і $k \cdot P \neq \mathbf{0}$ для всіх k = 1, 2, …, q-1, де $\mathbf{0}$ – нейтральний елемент групи точок еліптичній кривій E.

– $h(M)$ – геш-функція (ДСТУ 7564:2014), що відображає повідомлення M у двійкові вектори довжини 256 біт.

Кожний користувач цифрового підпису має особисті ключі:

– ключ шифрування d – ціле число, що лежить у межах $0 < d < q$.

– ключ дешифрування $Q = (x_Q, y_Q)$, що обчислюється як $Q = d \cdot P$.

Додаткові вимоги:

– $p^t \neq 1 \pmod q, \forall t = 1..B$, де $B \geq 31$

– $J(E) \neq 0$ і $J(E) \neq 1728$

Двійкові вектори

Між двійковими векторами довжини 256 $\bar{h} = (\alpha_{255}, ..., \alpha_0)$ і цілими числами $z \leq 2^{256}$ ставиться взаємо-однозначна відповідність за наступним правилом $z = \sum_{i=0}^{255} \alpha_i 2^i$. Тут α_i або дорівнює 0, або дорівнює 1. Інакше кажучи, $\bar{h}$ – це двійкове подання числа z.

Результатом операції конкатенації двох векторів $\bar{h_1} = (\alpha_{255}, ..., \alpha_0)$ і $\bar{h_2} = (\beta_{255}, ..., \beta_0)$ називається вектор довжини 512 $(\bar{h_1}|\bar{h_2}) = (\alpha_{255}, ..., \alpha_0, \beta_{255}, ..., \beta_0)$. Зворотна операція – операція розбивки одного вектора довжини 512 на два вектори довжини 256.

Створення ЕЦП на основі алгоритму ДСТУ 4145-2002 (рисунок 4.5) відбувається за наступною схемою:

1. Обчислення геш-функції від повідомлення М: $\bar{h} = h(M)$

2. Обчислення $e = z \bmod q$, і якщо $e = 0$, покласти $e = 1$. Де z – ціле число, що відповідає $\bar{h}$.

3. Генерація випадкового числа k такого, що $0 < k < q$.

4. Обчислення точки на еліптичній кривій $C = kP$, і по ній знаходження $r = x_c \bmod q$, де x_c – це координата x точки C. Якщо $r = 0$, вертаємося до попереднього кроку.

5. Знаходження $s = (rd + ke) \bmod q$. Якщо $s = 0$, вертаємося до кроку 3.

6. Формування цифрового підпису $\xi = (\bar{r}|\bar{s})$, де $\bar{r}$ і $\bar{s}$ – вектори, що відповідають r і s.

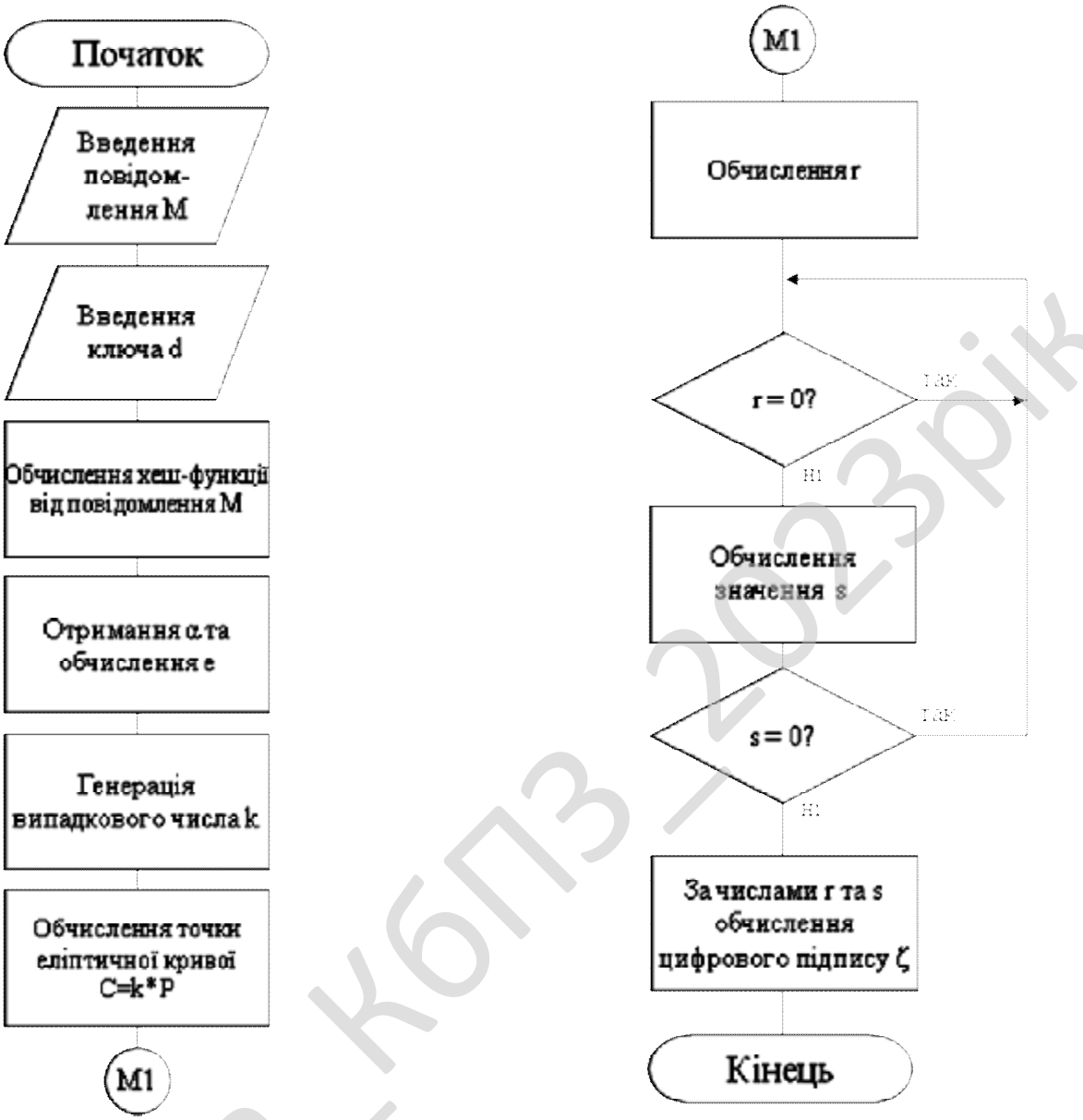

Рисунок 4.5 – Блок-схема підпрограми створення ЕЦП на основі алгоритму ДСТУ 4145-2002

Перевірка ЕЦП на основі алгоритму ДСТУ 4145-2002 (рисунок 4.6) відбувається за наступною схемою:

1. Обчислення за цифровим підписом ξ чисел r і s, враховуючи те, що $\xi = (\bar{r}|\bar{s})$, де r і s – числа, що відповідають векторам $\bar{r}$ і $\bar{s}$. Якщо хоча б одне з нерівностей $r < q$ і $s < q$ невірно, то підпис недійсний.

2. Обчислення геш-функції від повідомлення M: $\bar{h} = h(M)$.

3. Обчислення $e = z \bmod q$, і якщо $e = 0$, покласти $e = 1$. Де z – ціле число відповідне $\bar{h}$.

4. Обчислення $\nu = e^{-1} \bmod q$.

5. Обчислення $z_1 = s\nu \bmod q$ і $z_2 = -r\nu \bmod q$.

6. Обчислення точки на еліптичній кривій $C = z_1 P + z_2 Q$. І визначення $R = x_c \bmod q$, де x_c – координата x кривій C.

7. У випадку рівності $R = r$ підпис дійсний, інакше – недійсний.

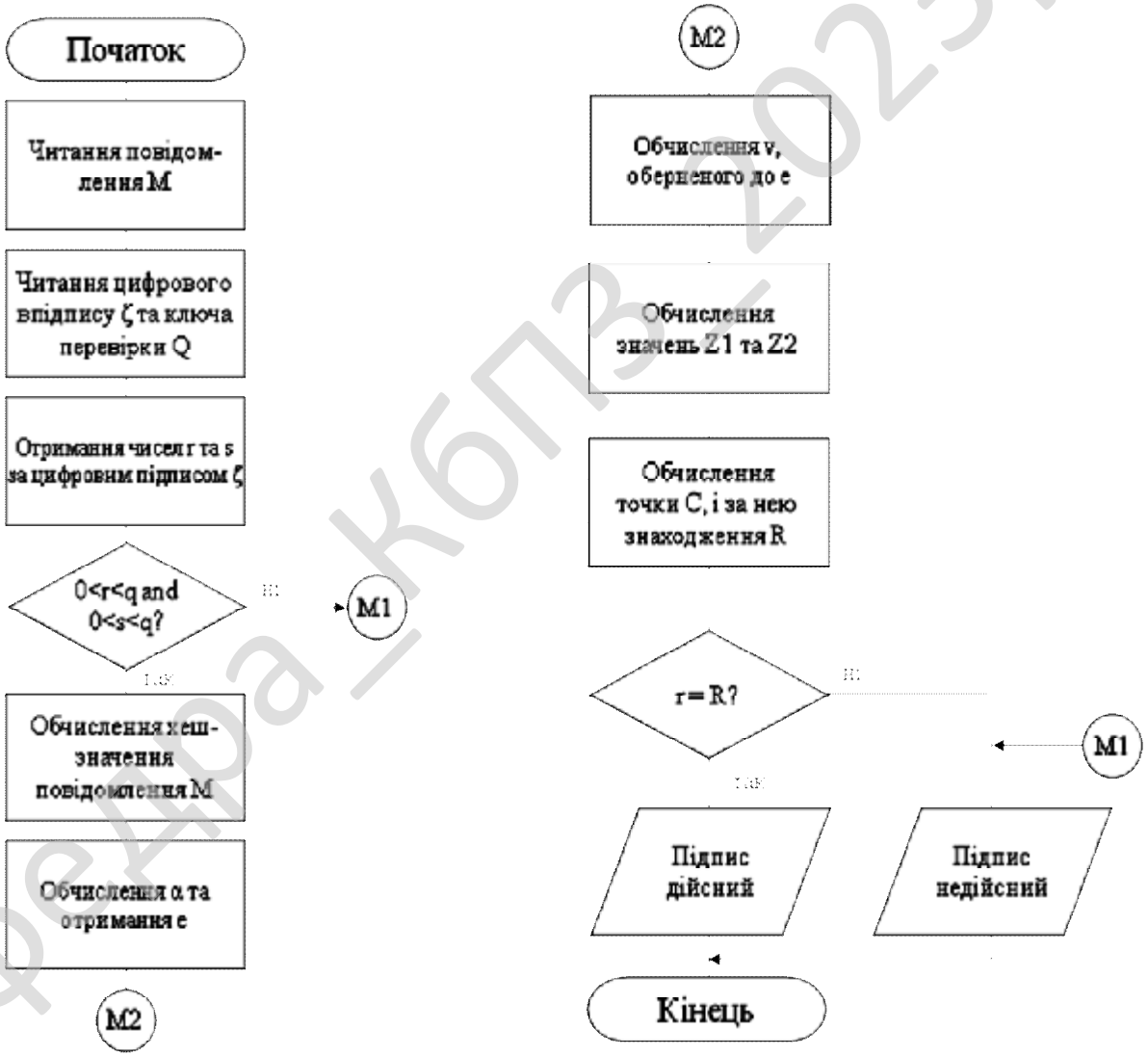

Рисунок 4.6 – Блок-схема підпрограми перевірки ЕЦП на основі алгоритму ДСТУ 4145-2002

Розглянемо розроблений клас для виконання математичних операцій над точками еліптичних кривих, який використовується для створення і перевірки цифрових підписів заснованих на використанні еліптичних кривих.

Клас для множення точок еліптичної кривої:

```
public class ECPoint
{
  public BigInteger x;
  public BigInteger y;
  public BigInteger a;
  public BigInteger b;
  public BigInteger FieldChar;
  public ECPoint(ECPoint p)
  {
    x = p.x;
    y = p.y;
    a = p.a;
    b = p.b;
    FieldChar = p.FieldChar;
  }
  public ECPoint()
  {
    x = new BigInteger();
    y = new BigInteger();
    a = new BigInteger();
    b = new BigInteger();
    FieldChar = new BigInteger();
  }
  //додавання двох точок P1 і P2
  public static ECPoint operator +(ECPoint p1, ECPoint p2)
  {
    ECPoint p3 = new ECPoint();
    p3.a = p1.a;
    p3.b = p1.b;
    p3.FieldChar = p1.FieldChar;
    BigInteger dy = p2.y - p1.y;
    BigInteger dx = p2.x - p1.x;
    if (dx < 0)
      dx += p1.FieldChar;
    if (dy < 0)
      dy += p1.FieldChar;
```

```
            BigInteger m = (dy * dx.modInverse(p1.FieldChar)) % p1.FieldChar;
            if (m < 0)
              m += p1.FieldChar;
            p3.x = (m * m - p1.x - p2.x) % p1.FieldChar;
            p3.y = (m * (p1.x - p3.x) - p1.y) % p1.FieldChar;
            if (p3.x < 0)
              p3.x += p1.FieldChar;
            if (p3.y < 0)
              p3.y += p1.FieldChar;
            return p3;
          }
          //додавання точки P с собою ж
          public static ECPoint Double(ECPoint p)
          {
            ECPoint p2 = new ECPoint();
            p2.a = p.a;
            p2.b = p.b;
            p2.FieldChar = p.FieldChar;
            BigInteger dy = 3 * p.x * p.x + p.a;
            BigInteger dx = 2 * p.y;
            if (dx < 0)
              dx += p.FieldChar;
            if (dy < 0)
              dy += p.FieldChar;
            BigInteger m = (dy * dx.modInverse(p.FieldChar)) % p.FieldChar;
            p2.x = (m * m - p.x - p.x) % p.FieldChar;
            p2.y = (m * (p.x - p2.x) - p.y) % p.FieldChar;
            if (p2.x < 0)
              p2.x += p.FieldChar;
            if (p2.y < 0)
              p2.y += p.FieldChar;
            return p2;
          }
          //множення точки на число x, по суті своїй представляє x додавань точки
самої до себе
          public static ECPoint multiply(BigInteger x, ECPoint p)
          {
            ECPoint temp = p;
            while (x != 0)
            {
              if ((x % 2) != 0)
              {
```

```
            if((temp.x==p.x)||(temp.y==p.y))
              temp=Double(temp);
            else
              temp = temp + p;
            x = x - 1;
          }
          x = x / 2;
          p = Double(p);
        }
        return temp;
      }
    }
```

Важливим елементом будь-якого цифрового підпису являється геш-функція, від правильного вибору якої залежить криптостійкість всього алгоритму. В даній роботі у якості геш-функції було обрано алгоритм MD5, який на сьогоднішній день являється одним з найбільш стійких до взломів.

Геш-функція MD5

Розглянемо алгоритм одержання геш-значення (дайджест) повідомлення MD5.

Алгоритм одержує на вході повідомлення довільної довжини й створює як вихід дайджест повідомлення довжиною 128 біт. Алгоритм складається з наступних кроків:

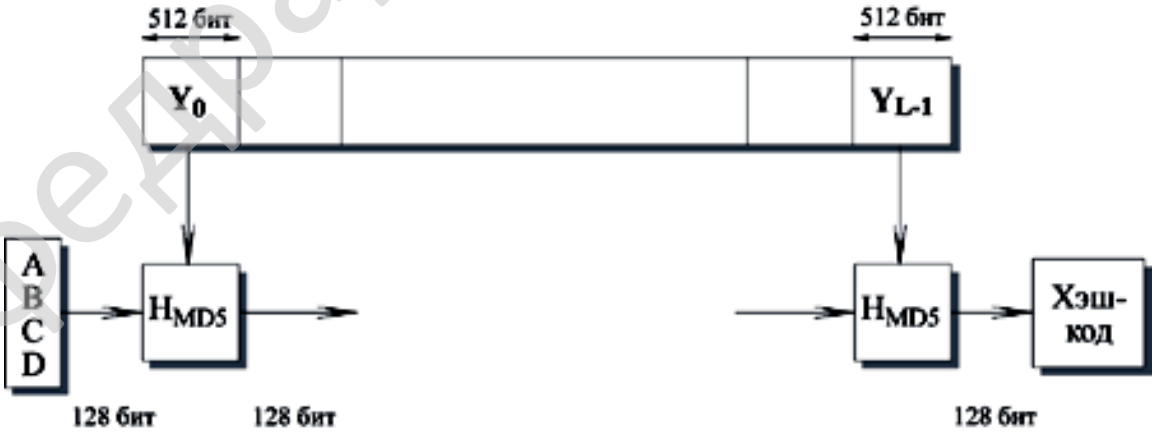

Рисунок 4.7 – Логіка виконання MD5

Крок 1: додавання відсутніх бітів

Повідомлення доповнюється таким чином, щоб його довжина стала рівною 448 за модулем 512. Це означає, що довжина доданого повідомлення на 64 біта менше, ніж число, кратне 512. Додавання здійснюється завжди, навіть якщо повідомлення має потрібну довжину. Наприклад, якщо довжина повідомлення 448 бітів, воно доповнюється 512 бітами до 960 бітів. Таким чином, число бітів, що додаються, перебуває в діапазоні від 1 до 512.

Додавання складається з одиниці, за якою слідує необхідна кількість нулів.

Крок 2: додавання довжини

64-бітне подання довжини вхідного (до додавання) повідомлення в бітах приєднується до результату першого кроку. Якщо первісна довжина більше, ніж 2^{64}, то використовуються тільки останні 64 біта. Таким чином, поле містить довжину вихідного повідомлення за модулем 2^{64}.

У результаті перших двох кроків створюється повідомлення, довжина якого кратна 512 бітам. Це розширене повідомлення представляється як послідовність 512-бітних блоків Y_0, Y_1, . . ., YL-1, при цьому загальна довжина розширеного повідомлення дорівнює L * 512 бітам. Таким чином, довжина отриманого розширеного повідомлення кратна шістнадцяти 32-бітним словам.

Повідомлення	**Додавання від 1 до 448 біт**	**Довжина вхідного повідомлення**

Рисунок 4.8 – Структура розширеного повідомлення

Крок 3: ініціалізація MD-буфера

Використовується 128-бітний буфер для зберігання проміжних і остаточних результатів геш-функції. Буфер може бути представлений як чотири 32-бітних регістри (A, B, C, D). Ці регістри ініціалізуються наступними шістнадцятковими числами:

A = 01234567, B = 89ABCDEF, C = FEDCBA98, D = 76543210

Крок 4: обробка послідовності 512-бітних (16-словних) блоків

Основою алгоритму є модуль, що складається із чотирьох циклічних обробок, позначений як HMD5. Чотири цикли мають схожу структуру, але кожний цикл використовує свою елементарну логічну функцію, позначувану f, f, f і f відповідно.

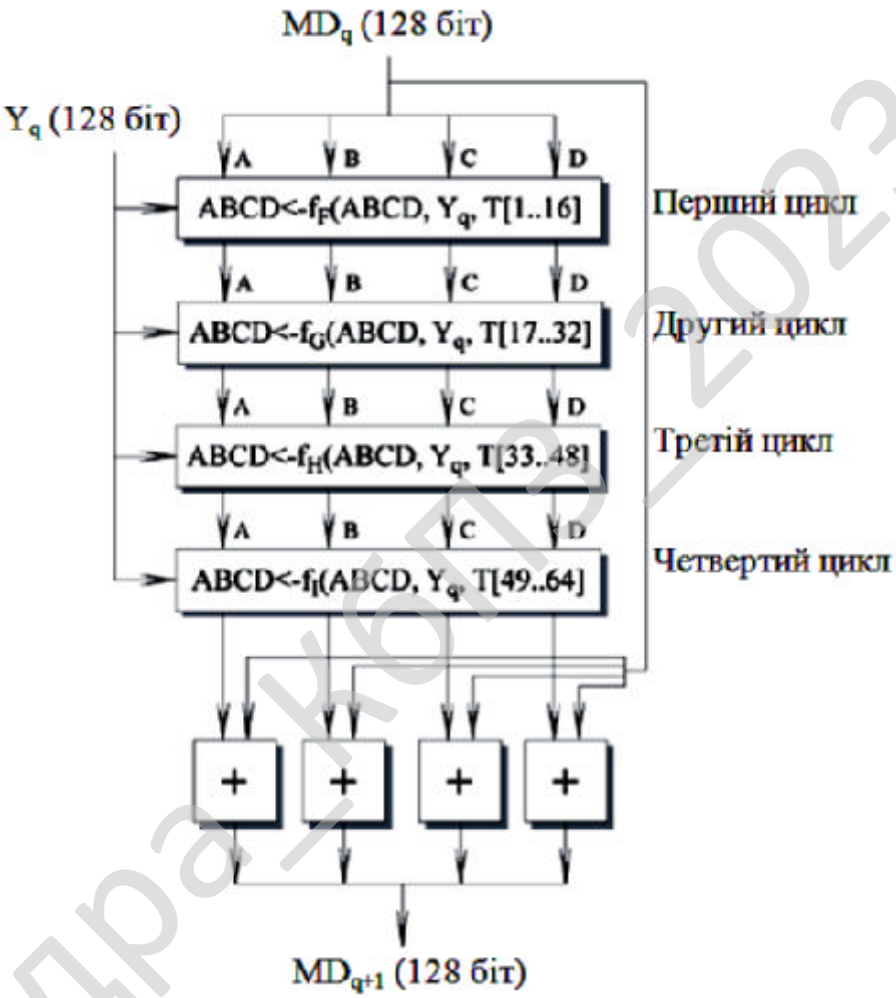

Рисунок 4.9 – Обробка чергового 512-бітного блоку

Кожний цикл приймає як вхід поточний 512-бітний блок Y_q, що обробляється в цей момент, і 128-бітне значення буфера ABCD, що є проміжним значенням дайджесту, і змінює вміст цього буфера. Кожний цикл також використовує четверту частину 64-елементної таблиці T[1...64], побудованої на основі функції sin. і-ий елемент T, що позначається T[i], має значення, рівне цілій

частині від 2^{32}* abs (sin (i)), і задано в радіанах. Тому що abs (sin (i)) є числом між 0 і 1, кожний елемент T є цілим, що може бути представлено 32 бітами. Таблиця забезпечує "випадковий" набір 32-бітних значень, які повинні ліквідувати будь-яку регулярність у вхідних даних.

Для одержання MD_{q+1} вихід чотирьох циклів складається за модулем 2^{32} з MD_q. Додавання виконується незалежно для кожного із чотирьох слів у буфері.

Крок 5: вихід

Після обробки всіх L 512-бітних блоків виходом L-Ой стадії є 128-бітний дайджест повідомлення.

Розглянемо більш детально логіку кожного із чотирьох циклів виконання одного 512-бітного блоку. Кожний цикл складається з 16 кроків, що оперують із буфером ABCD. Кожний крок можна представити у вигляді:

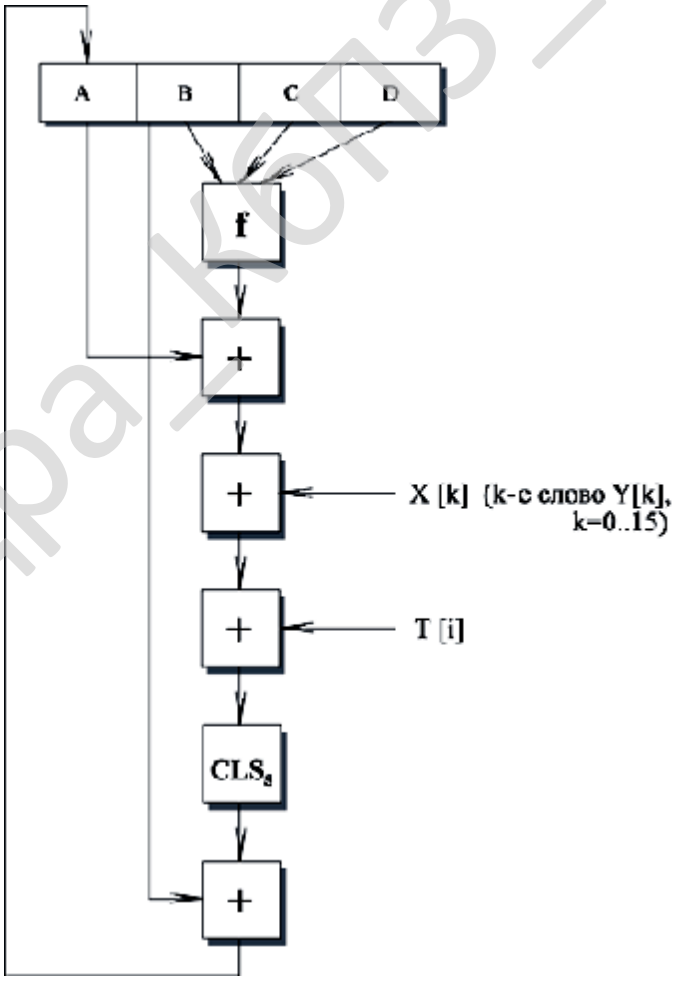

Рисунок 4.10 – Логіка виконання окремого кроку

$$A <- B + CLS_s (A + f (B, C, D) + X [k] + T [i]),$$

де:

A, B, C, D – чотири слова буфера; після виконання кожного окремого кроку відбувається циклічний зсув вліво на одне слово.

f – одна з елементарних функцій f, f, f, f.

CLS_s – циклічний зсув вліво на s битов 32-бітного аргументу.

X [k] – M [q * 16 + k] – k-ое 32-бітне слово в q-ом 512 блоці повідомлення.

T [i] – i-ое 32-бітне слово в матриці T.

+ – додавання за модулем 2^{32}.

На кожному із чотирьох циклів алгоритму використовується одна із чотирьох елементарних логічних функцій. Кожна елементарна функція одержує три 32-бітних слова на вході й на виході створює одне 32-бітне слово. Кожна функція являється множиною побітових логічних операцій, тобто n-ий біт виходу є функцією від n-ого біта трьох входів. Елементарні функції наступні:

$$f_F = (B \ \& \ C) \ (not \ B \ \& \ D),$$
$$f_G = (B \ \& \ D) \vee (C \ \& \ not \ D),$$
$$f_H = B \oplus C \oplus D,$$
$$f_I = C \oplus (B \ \& \ not \ D)$$

Масив з 32-бітних слів X [0..15] містить значення поточного 512-бітного вхідного блоку, що обробляється в даний момент. Кожний цикл виконується 16 разів, а тому що кожний блок вхідного повідомлення обробляється в чотирьох циклах, то кожний блок вхідного повідомлення обробляється за схемою (показаною на рисуноку 4.9) 64 рази. Якщо представити вхідний 512-бітний блок у вигляді шістнадцяти 32-бітних слів, то кожне вхідне 32-бітне слово використовується чотири рази, по одному разі в кожному циклі, і кожний елемент таблиці T, що складається з 64 32-бітних слів, використовується тільки один раз. Після кожного кроку циклу відбувається циклічний зсув вліво чотирьох слів A, B, C і D. На кожному кроці змінюється тільки одне із чотирьох слів буфера ABCD. Отже, кожне слово буфера змінюється 16 разів, і потім 17-ий раз наприкінці для одержання остаточного виходу даного блоку.

Можна підсумувати алгоритм MD5 у такий спосіб:

$$MD_0 = IV,$$

$$MD_{q+1} = MD_q + f[Y_q, f[Y_q, f[Y_q, f[Y_q, MD_q]]]],$$

$$MD = MDL\text{-}1,$$

де:

IV – початкове значення буфера ABCD, визначене на кроці 3.

Y_q – q-ий 512-бітний блок повідомлення.

L – число блоків у повідомленні (включаючи поля доповнення й довжини).

MD – остаточне значення дайджесту повідомлення.

Наведемо фрагмент коду розробленого програмного забезпечення, що реалізує алгоритм MD5:

```
public class MD5Digest
        : GeneralDigest
    {
        private const int DigestLength = 16;
        private int H1,H2,H3,H4;
        private int[]   X = new int[16];
        private int         xOff;
        public MD5Digest()
        {
            Reset();
        }

        public MD5Digest(MD5Digest t)
              : base(t)
        {
            H1 = t.H1;
            H2 = t.H2;
            H3 = t.H3;
            H4 = t.H4;
            Array.Copy(t.X,0,X,0,t.X.Length);
            xOff = t.xOff;
        }
         public override string AlgorithmName
         {
              get { return "MD5"; }
```

```
        }
        public override int GetDigestSize()
        {
            return DigestLength;
        }
        internal override void ProcessWord(
            byte[]  input,
            int     inOff)
        {
            X[xOff++] = (input[inOff] & 0xff) | ((input[inOff + 1] & 0xff) << 8)
                | ((input[inOff + 2] & 0xff) << 16) | ((input[inOff + 3] & 0xff)
<< 24);
            if (xOff == 16)
            {
                ProcessBlock();
            }
        }
        internal override void ProcessLength(
            long    bitLength)
        {
            if (xOff > 14)
            {
                ProcessBlock();
            }
            X[14] = (int)(bitLength & 0xffffffff);
            X[15] = (int)((ulong) bitLength >> 32);
        }
        private void UnpackWord(
            int     word,
            byte[]  outBytes,
            int     outOff)
        {
            outBytes[outOff]     = (byte)word;
            outBytes[outOff + 1] = (byte)((uint) word >> 8);
            outBytes[outOff + 2] = (byte)((uint) word >> 16);
            outBytes[outOff + 3] = (byte)((uint) word >> 24);
        }
        public override int DoFinal(
            byte[]  output,
            int     outOff)
        {
            Finish();
```

```
        UnpackWord(H1,output,outOff);
        UnpackWord(H2,output,outOff + 4);
        UnpackWord(H3,output,outOff + 8);
        UnpackWord(H4,output,outOff + 12);
        Reset();
        return DigestLength;
    }
    /**
    * Скидання ланцюжка змінних в IV значення
    */
    public override void Reset()
    {
        base.Reset();
        H1 = unchecked((int) 0x67452301);
        H2 = unchecked((int) 0xefcdab89);
        H3 = unchecked((int) 0x98badcfe);
        H4 = unchecked((int) 0x10325476);
        xOff = 0;
        for (int i = 0; i != X.Length; i++)
        {
            X[i] = 0;
        }
    }
    //1 раунд
    private static readonly int S11 = 7;
    private static readonly int S12 = 12;
    private static readonly int S13 = 17;
    private static readonly int S14 = 22;
    // 2 раунд
    private static readonly int S21 = 5;
    private static readonly int S22 = 9;
    private static readonly int S23 = 14;
    private static readonly int S24 = 20;
    // 3 раунд
    private static readonly int S31 = 4;
    private static readonly int S32 = 11;
    private static readonly int S33 = 16;
    private static readonly int S34 = 23;
    // 4 раунд
    private static readonly int S41 = 6;
    private static readonly int S42 = 10;
    private static readonly int S43 = 15;
```

```
private static readonly int S44 = 21;
/*
* зсув змінної x вліво на n бітів
*/
private int RotateLeft(
    int x,
    int n)
{
    return (x << n) | (int) ((uint) x >> (32 - n));
}
/*
* F,G,H та I - базові функції MD5
*/
private int F(
    int u,
    int v,
    int w)
{
    return (u & v) | (~u & w);
}
private int G(
    int u,
    int v,
    int w)
{
    return (u & w) | (v & ~w);
}
private int H(
    int u,
    int v,
    int w)
{
    return u ^ v ^ w;
}
private int K(
    int u,
    int v,
    int w)
{
    return v ^ (u | ~w);
}
internal override void ProcessBlock()
```

```
        {
            int a = H1;
            int b = H2;
            int c = H3;
            int d = H4;
                // Раунд 1 – F цикл,16 разів.
                a = RotateLeft((a + F(b,c,d) + X[0] + unchecked((int)
0xd76aa478)),S11) + b;
            d = RotateLeft((d + F(a,b,c) + X[1] + unchecked((int)
0xe8c7b756)),S12) + a;
            c = RotateLeft((c + F(d,a,b) + X[2] + unchecked((int)
0x242070db)),S13) + d;
            b = RotateLeft((b + F(c,d,a) + X[3] + unchecked((int)
0xc1bdceee)),S14) + c;
            a = RotateLeft((a + F(b,c,d) + X[4] + unchecked((int)
0xf57c0faf)),S11) + b;
            d = RotateLeft((d + F(a,b,c) + X[5] + unchecked((int)
0x4787c62a)),S12) + a;
            c = RotateLeft((c + F(d,a,b) + X[6] + unchecked((int)
0xa8304613)),S13) + d;
            b = RotateLeft((b + F(c,d,a) + X[7] + unchecked((int)
0xfd469501)),S14) + c;
            a = RotateLeft((a + F(b,c,d) + X[8] + unchecked((int)
0x698098d8)),S11) + b;
            d = RotateLeft((d + F(a,b,c) + X[9] + unchecked((int)
0x8b44f7af)),S12) + a;
            c = RotateLeft((c + F(d,a,b) + X[10] + unchecked((int)
0xffff5bb1)),S13) + d;
            b = RotateLeft((b + F(c,d,a) + X[11] + unchecked((int)
0x895cd7be)),S14) + c;
            a = RotateLeft((a + F(b,c,d) + X[12] + unchecked((int)
0x6b901122)),S11) + b;
            d = RotateLeft((d + F(a,b,c) + X[13] + unchecked((int)
0xfd987193)),S12) + a;
            c = RotateLeft((c + F(d,a,b) + X[14] + unchecked((int)
0xa679438e)),S13) + d;
            b = RotateLeft((b + F(c,d,a) + X[15] + unchecked((int)
0x49b40821)),S14) + c;
                // Раунд 2 – G цикд,16 разів.
                a = RotateLeft((a + G(b,c,d) + X[1] + unchecked((int)
0xf61e2562)),S21) + b;
```

```
            d = RotateLeft((d + G(a,b,c) + X[6] + unchecked((int)
0xc040b340)),S22) + a;
            c = RotateLeft((c + G(d,a,b) + X[11] + unchecked((int)
0x265e5a51)),S23) + d;
            b = RotateLeft((b + G(c,d,a) + X[0] + unchecked((int)
0xe9b6c7aa)),S24) + c;
            a = RotateLeft((a + G(b,c,d) + X[5] + unchecked((int)
0xd62f105d)),S21) + b;
            d = RotateLeft((d + G(a,b,c) + X[10] + unchecked((int)
0x02441453)),S22) + a;
            c = RotateLeft((c + G(d,a,b) + X[15] + unchecked((int)
0xd8a1e681)),S23) + d;
            b = RotateLeft((b + G(c,d,a) + X[4] + unchecked((int)
0xe7d3fbc8)),S24) + c;
            a = RotateLeft((a + G(b,c,d) + X[9] + unchecked((int)
0x21e1cde6)),S21) + b;
            d = RotateLeft((d + G(a,b,c) + X[14] + unchecked((int)
0xc33707d6)),S22) + a;
            c = RotateLeft((c + G(d,a,b) + X[3] + unchecked((int)
0xf4d50d87)),S23) + d;
            b = RotateLeft((b + G(c,d,a) + X[8] + unchecked((int)
0x455a14ed)),S24) + c;
            a = RotateLeft((a + G(b,c,d) + X[13] + unchecked((int)
0xa9e3e905)),S21) + b;
            d = RotateLeft((d + G(a,b,c) + X[2] + unchecked((int)
0xfcefa3f8)),S22) + a;
            c = RotateLeft((c + G(d,a,b) + X[7] + unchecked((int)
0x676f02d9)),S23) + d;
            b = RotateLeft((b + G(c,d,a) + X[12] + unchecked((int)
0x8d2a4c8a)),S24) + c;
                // Раунд 3 – H цикл,16 разів.
                a = RotateLeft((a + H(b,c,d) + X[5] + unchecked((int)
0xfffa3942)),S31) + b;
            d = RotateLeft((d + H(a,b,c) + X[8] + unchecked((int)
0x8771f681)),S32) + a;
            c = RotateLeft((c + H(d,a,b) + X[11] + unchecked((int)
0x6d9d6122)),S33) + d;
            b = RotateLeft((b + H(c,d,a) + X[14] + unchecked((int)
0xfde5380c)),S34) + c;
            a = RotateLeft((a + H(b,c,d) + X[1] + unchecked((int)
0xa4beea44)),S31) + b;
```

```
            d = RotateLeft((d + H(a,b,c) + X[4] + unchecked((int)
0x4bdecfa9)),S32) + a;
            c = RotateLeft((c + H(d,a,b) + X[7] + unchecked((int)
0xf6bb4b60)),S33) + d;
            b = RotateLeft((b + H(c,d,a) + X[10] + unchecked((int)
0xbebfbc70)),S34) + c;
            a = RotateLeft((a + H(b,c,d) + X[13] + unchecked((int)
0x289b7ec6)),S31) + b;
            d = RotateLeft((d + H(a,b,c) + X[0] + unchecked((int)
0xeaa127fa)),S32) + a;
            c = RotateLeft((c + H(d,a,b) + X[3] + unchecked((int)
0xd4ef3085)),S33) + d;
            b = RotateLeft((b + H(c,d,a) + X[6] + unchecked((int)
0x04881d05)),S34) + c;
            a = RotateLeft((a + H(b,c,d) + X[9] + unchecked((int)
0xd9d4d039)),S31) + b;
            d = RotateLeft((d + H(a,b,c) + X[12] + unchecked((int)
0xe6db99e5)),S32) + a;
            c = RotateLeft((c + H(d,a,b) + X[15] + unchecked((int)
0x1fa27cf8)),S33) + d;
            b = RotateLeft((b + H(c,d,a) + X[2] + unchecked((int)
0xc4ac5665)),S34) + c;
                // Раунд 4 – К цикл,16 разів.
                a = RotateLeft((a + K(b,c,d) + X[0] + unchecked((int)
0xf4292244)),S41) + b;
            d = RotateLeft((d + K(a,b,c) + X[7] + unchecked((int)
0x432aff97)),S42) + a;
            c = RotateLeft((c + K(d,a,b) + X[14] + unchecked((int)
0xab9423a7)),S43) + d;
            b = RotateLeft((b + K(c,d,a) + X[5] + unchecked((int)
0xfc93a039)),S44) + c;
            a = RotateLeft((a + K(b,c,d) + X[12] + unchecked((int)
0x655b59c3)),S41) + b;
            d = RotateLeft((d + K(a,b,c) + X[3] + unchecked((int)
0x8f0ccc92)),S42) + a;
            c = RotateLeft((c + K(d,a,b) + X[10] + unchecked((int)
0xffeff47d)),S43) + d;
            b = RotateLeft((b + K(c,d,a) + X[1] + unchecked((int)
0x85845dd1)),S44) + c;
            a = RotateLeft((a + K(b,c,d) + X[8] + unchecked((int)
0x6fa87e4f)),S41) + b;
```

```
            d = RotateLeft((d + K(a,b,c) + X[15] + unchecked((int)
0xfe2ce6e0)),S42) + a;
            c = RotateLeft((c + K(d,a,b) + X[6] + unchecked((int)
0xa3014314)),S43) + d;
            b = RotateLeft((b + K(c,d,a) + X[13] + unchecked((int)
0x4e0811a1)),S44) + c;
            a = RotateLeft((a + K(b,c,d) + X[4] + unchecked((int)
0xf7537e82)),S41) + b;
            d = RotateLeft((d + K(a,b,c) + X[11] + unchecked((int)
0xbd3af235)),S42) + a;
            c = RotateLeft((c + K(d,a,b) + X[2] + unchecked((int)
0x2ad7d2bb)),S43) + d;
            b = RotateLeft((b + K(c,d,a) + X[9] + unchecked((int)
0xeb86d391)),S44) + c;
            H1 += a;
            H2 += b;
            H3 += c;
            H4 += d;
                // Скидання зміщення і очищення буфера
                xOff = 0;
            for (int i = 0; i != X.Length; i++)
            {
                X[i] = 0;
            }
        }
    }
```

4.2 Захист розробленого програмного забезпечення

Захист розробленого програмного забезпечення буде відбуватися за допомогою алгоритму Camellia – блоковий шифр на основі мережі Фейстеля. У криптографії, Camellia – це симетричний ключ блоковий шифр із розміром блоку 128 біт і розмірами ключа 128, 192 і 256 біт. Він був розроблений спільно Mitsubishi Electric і NTT з Японії. Шифр був схвалений для використання ISO / IEC, проектом Європейського Союзу NESSIE і Японським CRYPTREC проект. шифр має рівні безпеки й можливості обробки, порівнянні з Advanced Encryption Standard.

Шифр був розроблений, щоб підходити як для програмних, так і для апаратних реалізацій, від недорогих смарт-карти для високошвидкісних мережних систем. Він є частиною криптографічного протоколу Transport Layer Security (TLS), призначеного для забезпечення безпеки зв'язки в комп'ютерній мережі, такий як Інтернет

Camellia – це шифр Фейстеля з 18 раундами (при використанні 128-бітних ключів) або 24 раундами (при використанні 192- або 256-бітних ключів). Кожні шість раундів застосовується шар логічного перетворення: так звана «FL-функція» або її зворотна. Camellia використовує чотири 8 × 8-бітних S-блоку із вхідними й вихідними афінними перетвореннями й логічними операціями. Шифр також використовує введення й вивід відбілювання клавіш. Шар дифузія використовує лінійне перетворення на основі матриці з номером галузей 5.

Аналіз безпеки

Камелія вважається сучасним надійним шифром. Навіть при використанні параметра меншого розміру ключа (128 біт) уважається неможливим зламати його за допомогою атаки грубої сили на ключі за допомогою сучасних технологій. Немає відомих успішних атак, що значно послабляють шифр. Шифр був схвалений для використання ISO / IEC, проектом Європейського Союзу NESSIE і Японським CRYPTREC проект. Японський шифр має рівні безпеки й можливості обробки, порівнянні із шифром AES/Rijndael.

Camellia – це блоковий шифр, який може бути повністю визначені мінімальними системами багатомірних багаточленів:

– Камелія (а також AES) S-блоки можуть бути описані системою 23 квадратних рівнянь в 80 членах.

– Розклад ключів можна описати 1120 рівняннями в 768 змінні з використанням 3328 лінійних і квадратичних членів.

– Увесь блоковий шифр можна описати 5104 рівняннями в 2816 змінні з використанням 14 592 лінійних і квадратичних членів.

– Усього потрібно 6224 рівняння з 3584 змінними з використанням 17 920 лінійних і квадратичних членів.

– Кількість вільних членів становить 11 696, що приблизно таке ж число, що й для AES.

Теоретично, такі властивості можуть дозволити зламати Camellia (і AES) за допомогою алгебраїчної атаки, такий як розширена розріджена лінеаризація, у т Майбутнє за умови, що атака стане можливою.

Хоча Camellia запатентована, вона доступна за безоплатною ліцензією. Це дозволило шифру Camellia стати частиною проекту OpenSSL під ліцензією з відкритим вихідним кодом з листопада 2006 року. Це також дозволило йому стати частиною Mozilla Модуль NSS (Служби мережної безпеки).

Підтримка Camellia була додана в остаточний випуск Mozilla Firefox 3 в 2008 році (за замовчуванням відключене починаючи з Firefox 33 в 2014 році в дусі «Пропозиції по зміні стандартних наборів шифрів TLS, пропонованих браузерами», який був виключено з версії 37 в 2015 році). Pale Moon, відгалуження Mozilla / Firefox, продовжує пропонувати Camellia і розширив свою підтримку, включивши в нього набори Galois / Counter mode (GCM) із шифром, але вилучив GCM знову у випуску 27.2.0, пославшись на очевидну відсутність інтересу до них.

Пізніше, в 2008 році, група розробки релізу FreeBSD оголосила, що цей шифр також був включений в FreeBSD 6.4. Крім того, Йошисато Янагисава додав підтримку шифру Camellia у дисковий клас зберігання geli FreeBSD.

У вересні 2009 року GNU Privacy Guard додала підтримку Camellia у версії 1.4.10.

Veracrypt (відгалуження Truecrypt) включав Camellia як один з підтримуваних алгоритмів шифрування.

Крім того, різні популярні бібліотеки безпеки, такі як Crypto ++, Gnutls, mbed TLS і Openssl також включають підтримку Camellia.

26 березня 2013 р. було оголошено, що Camellia була знову обрана для включення в новий список рекомендованих шифрів для електронного уряду Японії як єдиний 128-бітний алгоритм блокового шифрування, розроблений у Японії. Це збігається з тим, що список CRYPTREC обновляється вперше за 10 років. Вибір був заснований на високій репутації Camellia у плані простоти придбання, а також характеристик безпеки й продуктивності, порівнянних з такими з Advanced Encryption Standard (AES). Камелія залишається незмінною у своєму повному втіленні. Неможлива диференціальна атака на Camellia з 12 раундами без шарів FL / FL дійсно існує.

Продуктивність

S-блоки, використовувані Camellia, мають структуру, аналогічну S-блоку AES. У результаті можна прискорити реалізацію програмного забезпечення Camellia за допомогою наборів команд ЦП, розроблених для AES, таких як x86 AES-NI.

5 МЕТОДИКА ВПРОВАДЖЕННЯ СИСТЕМИ КІБЕРБЕЗПЕКИ В ПРОМИСЛОВУ ЕКСПЛУАТАЦІЮ

На рисунку 5.1 зображене вікно розробленого програмного забезпечення, що відображає процес формування ЕЦП на основі еліптичних кривих.

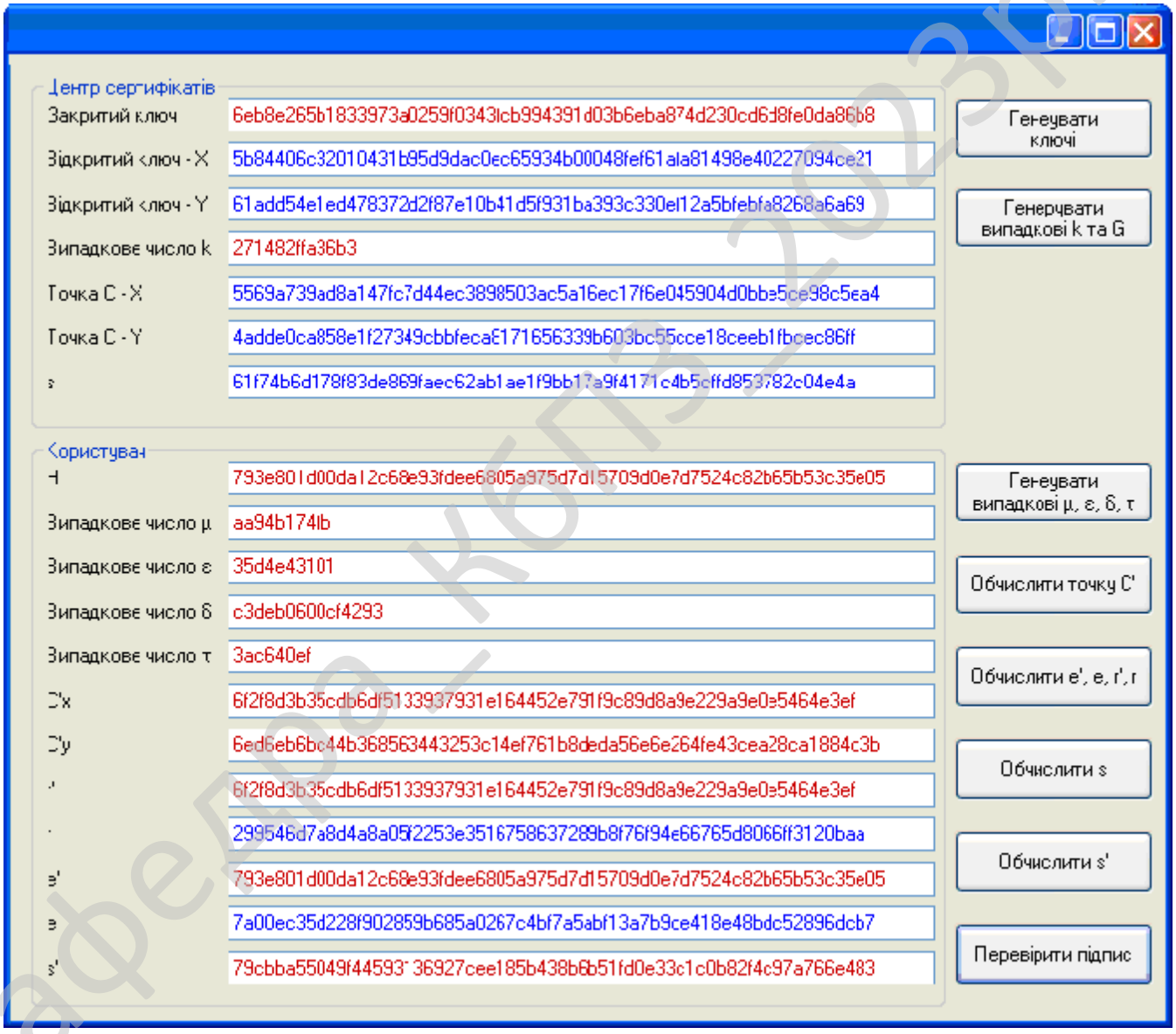

Рисунок 5.1 – Вікно формування ЕЦП на основі еліптичних кривих

Для формування електронного підпису необхідно виконати наступну послідовність дій:

1. На стороні центу сертифікатів:

– генерація закритого та відкритих ключів (X, Y);

– генерація випадкових чисел k та G.

– обчислення точок C – X та C – Y;

2. На стороні користувача:

– генерація випадкових чисел μ, ε, δ, τ;

– обчислення точки C';

– обчислення чисел e', e, r', r;

– обчислення числа s;

– обчислення числа s';

Після виконання зазначених дій ЕЦП створено та можна використовувати.

На рисунку 5.2 зображене вікно довідки, що містить короткі відомості про розроблене програмне забезпечення.

Рисунок 5.2 – Вікно довідки

6 ОСНОВНІ ВИСНОВКИ

Програмне забезпечення, створене в результаті виконання випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти, призначено для системи кібербезпеки ЕЦП для автентифікації користувачів у системах дистанційного навчання та тестування.

В межах України в недостатній мірі представлені вітчизняні розробки в цій області.

Рішення завдання полягало у вирішенні наступних задач:

– Був проведений огляд існуючих систем ЕЦП для автентифікації користувачів у системах дистанційного навчання та тестування.

– Досліджена система ЕЦП для автентифікації користувачів у системах дистанційного навчання та тестування.

– На основі отриманих результатів досліджень створена програмна реалізація системи кібербезпеки ЕЦП для автентифікації користувачів у системах дистанційного навчання та тестування.

Розроблені під час виконання випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти алгоритми дозволяють успішно вирішувати завдання ЕЦП для автентифікації користувачів у системах дистанційного навчання та тестування.

Розроблене програмне забезпечення має простий, дружній та зручний інтерфейс користувача, що забезпечує легкість у освоєнні роботи програмного продукту, зручність у використанні, і не потребує особливих спеціальних знань.

При створені програмного забезпечення було використано об’єктно-орієнтований підхід, що відповідає сучасним тенденціям у галузі розробки комерційних програмних систем.

Програма реалізована на мові високого рівня Visual C#. Дана мова програмування дозволяє найбільш ефективно обробляти дані призначені для

системи кібербезпеки ЕЦП для автентифікації користувачів у системах дистанційного навчання та тестування. Це дозволило мінімізувати строк розробки програмного забезпечення, і, як слід, зменшити витрати на його розробку. Запропоноване програмне забезпечення ділиться на загальне програмне забезпечення, що поставляється із засобами обчислювальної техніки й спеціальне програмне забезпечення, що спеціально розроблене для даної конкретної системи кібербезпеки й включає програми, що реалізують її функції.

Програма призначена для виконання під управлінням багатозадачної операційної системи кібербезпеки Windows 10/11.

Даються необхідні рекомендації з установки розробленого програмного забезпечення.

Для підвищення рівня безпеки запропоновано застосовувати алгоритм Camellia.

В цілому створене програмне забезпечення підтверджує правильність використаних проектних рішень та повністю відповідає вимогам технічного завдання. Створене програмне забезпечення має потенційну можливість для подальшого вдосконалення і застосування у різних галузях.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Трохименко В. Дистанційне навчання педагогічних працівників: досвід і проблеми// Післядипломна освіта в Україні. – 2004. – С. 29 – 32.

2. Smirnov, O., Neskorodieva, T., Fedorov, E., Rudakov, K., Neskorodieva, A. «Method Detection Audit Data Anomalies on Basis Restricted Cauchy Machine» *CEUR Workshop Proceedings,* Volume 3187, 2022, pp. 1-12. **(Scopus).**

3. Smirnov O., Smirnova T., Anas M. Al-Oraiqat, Drieiev O., Polishchuk L., Sheroz Khan, Yassin M. Y. Hasan, Aladdein M. Amro, Hazim S. AlRawashdeh «Method for Determining Treated Metal Surface Quality Using Computer Vision Technology». *Sensors (Basel, Switzerland)* Volume 22, Issue 16, 6223, 2022. (**Scopus**).

4. Smirnov, O., Lakhno, V., Akhmetov, B., Chubaievskyi, V., Khorolska, K., Bebeshko, B. «Selection of a Rational Composition of İnformation Protection Means Using a Genetic Algorithm». *In: Rajakumar, G., Du, KL., Vuppalapati, C., Beligiannis, G.N. (eds) Intelligent Communication Technologies and Virtual Mobile Networks. Lecture Notes on Data Engineering and Communications Technologies*, vol 131. 2023. **Springer**, Singapore. pp. 21-34. **(Scopus).**

5. Smirnov O.A., Al-Oraiqat A.M., Ulichev O.S., Meleshko Ye.V., Al-Rawashdeh H.S., Polishchuk L.I. «Modeling strategies for information influence dissemination in social networks». *Journal of Ambient Intelligence and Humanized Computing* Volume 13, Issue 5. **Springer**, Cham. 2022, pp. 2463-2477. **(Scopus).**

6. Smirnov O., Kuznetsov A., Kryvinska N., Kiian A., Kuznetsova K. «Full Non-Binary Constant-Weight Codes». *SN Computer Science,* Vol 2, 337, 2021. https://doi.org/10.1007/s42979-021-00739-w **(Scopus).**

7. Smirnov O., Kovalenko O., Kovalenko A., Kavun S. «Quantitative Risk Assessment Method Development in the Context of the SDLC-model». *2021 IEEE 8th International Conference on Problems of Infocommunications, Science and Technology (PIC S&T),* 2021, pp. 203-208, doi: 10.1109/PICST54195.2021.9772143 **(Scopus).**

8. Smirnov O., Neskorodieva T., Fedorov E., Rymar P. «Neural Network Modeling Method of Transformations Data of Audit Production with Returnable Waste». *CEUR Workshop Proceedings* Volume 3101, 2021, Pages 192-207. **(Scopus).**

9. Smirnov O., Kuznetsov A., Kiian A., Kuznetsova K. «Data hiding scheme based on spread sequence addressing». *CEUR Workshop Proceedings* Volume 2805, 2020, Pages *44-58.* **(Scopus)**.

10. Smirnov, O., Kuznetsov, A., Potii, O., Poluyanenko, N., Stelnyk, I., Mialkovsky, D. «Combining and filtering functions in the framework of nonlinear-feedback shift register». *International Journal of Computing*; 2020, Volume 19, Issue 2 – Research Institute for Intelligent Computer Systems – 2020. – P. 247-256. **(Scopus)**.

11. Smirnov O., Kuznetsov A., Kiian A., Kuznetsova T. «Non-binary constant weight coding technique». *CEUR Workshop Proceedings.* Volume 2740, 2020, Pages 102-114. **(Scopus)**.

12. Smirnov O.A., Alimseitova Zh., Adranova A., Akhmetov B., Lakhno V., Zhilkishbayeva G. «Models and algorithms for ensuring functional stability and cybersecurity of virtual cloud resources». *Journal of theoretical and applied information technology* Vol.98. No 21, 2020, P. 3334-3346. **(Scopus)**.

13. Smirnov O., Kuznetsov A., Arischenko A., Chepurko I., Onikiychuk A., Kuznetsova T. «Pseudorandom sequences for spread spectrum image steganography». *CEUR Workshop Proceedings* Volume 2654, 2020, Pages 122-131. **(Scopus)**.

14. Smirnov O., Kuznetsov A., Kovalchuk D., Kuznetsova T. «New technique for data hiding in cover images using adaptively generated pseudorandom sequences». *CEUR Workshop Proceedings* Volume 2654, 2020, Pages 1-14. **(Scopus)**.

15. Smirnov O., Lutsenko M., Kuznetsov A., Kiian A., Kuznetsova T., «Biometric cryptosystems: overview, state-of-the-art and perspective directions». *Lecture Notes in Networks and Systems*, vol 152. **Springer**, Cham. 2021, pp 66-84. **(Scopus)**.

16. Smirnov O., Kuznetsov A., Pushkar'ov A., Serhiienko R., Babenko V., Kuznetsova T.**, «**Representation of Cascade Codes in the Frequency Domain**».** In:

Radivilova T., Ageyev D., Kryvinska N. (eds) Data-Centric Business and Applications. Lecture Notes on Data Engineering and Communications Technologies, vol 48. **Springer**, Cham. 2021. pp 557-587. **(Scopus)**.

17. Smirnov, O., Markovets, O. Vovk, N., Turchyn, Y., «Model of informational support for social network administrators' content creation». *CEUR Workshop Proceedings* Volume 2616, 2020, Pages 125-136. **(Scopus)**.

18. Smirnov, O., Drieieva, H., Drieiev, O., Polishchuk, Y., Brzhanov, R., Aleksander, M. «Method of fractal traffic generation by a model of generator on the graph». *CEUR Workshop Proceedings* Volume 2616, 2020, Pages 366-379. **(Scopus)**.

19. Smirnov, O., Shekhanin, K., Kuznetsov, A., Krasnobayev, V. «Detecting Hidden Information in FAT». *International Journal of Computer Network and Information Security (IJCNIS)*. Vol. 12, No. 3, 2020. PP.33-43. **(Scopus)**.

20. Smirnov, O., Drieieva, H., Drieiev, O., Simakhin, V., Bondar, S., Odarchenko, R. «Managing multifractal properties of the binary sequence generated with the Markov chains», *CEUR Workshop Proceedings* Volume 2608, 2020, Pages 633-645. **(Scopus)**.

21. Smirnov, O., Kuznetsov, A., Gorbacheva, L., Babenko, V., «Hiding data in images using a pseudo-random sequence», *CEUR Workshop Proceedings* Volume 2608, 2020, Pages 646-660., **(Scopus)**.

22. Smirnov, O., Kuznetsov, A., Kolovanova, I., Kuznetsova, T., «Noise immunity of the algebraic geometric codes». International Journal of Computing; 2019, Volume 18, Issue 4 – Research Institute for Intelligent Computer Systems – 2019. – P. 393-407. **(Scopus)**.

23. Smirnov, O., Ulichev, O., Meleshko, Y., Khokh, V., Goncharenko, I. «Method of Choosing Objects for Informational Influence in Social Networks during Information Campaign Based on the Analytic Hierarchy Process». *CEUR Workshop Proceedings*, Vol 2588, P. 215-227, 2019. **(Scopus)**.

24. Smirnov, O., Krasnobayev, V., Yanko, A., Kuznetsova, T. «Methods of nulling numbers in the system of residual classes». *CEUR Workshop Proceedings*, Vol 2588, P. 90-106, 2019. **(Scopus)**.

25. Smirnov, O., Kuznetsov, A., Kovalchuk, D., Pastukhov, M., Kuznetsova, K., Prokopovych-Tkachenko, D., «Discrete Signals with Special Correlation Properties», *CEUR Workshop Proceedings* Volume 2353, *CEUR Workshop Proceedings* 2019, Pages 618-629. **(Scopus)**.

26. Smirnov, O., Kuznetsov, A., Kiian, A., Kuznetsova, K., Ivko, T., Prokopovych-Tkachenko, D., «Soft Decoding Based on Ordered Subsets of Verification Equations of Turbo-Productive Codes», *CEUR Workshop Proceedings* Volume 2353, *CEUR Workshop Proceedings* 2019, Pages 873-884. **(Scopus)**.

27. Smirnov, O., Kuznetsov, A., Prokopovych-Tkachenko, D. «Hiding Data in Images Using a Pseudo-Random Sequence». *ISCI'2020: Information Security in Critical Infrastructures.* ***Collective monograph***. Edited by Ivan D. Gorbenko, Victor A. Krasnobayev and Alexandr A. Kuznetsov. ASC Academic Publishing, USA, 2020. pp. 46-59. – ISBN: 978-1-7362833-0-1 (Hardback), ISBN: 978-1-7362833-1-8 (Ebook).

28. Smirnov, O., Kuznetsov, A., Shekhanin, K., Chepurko, I. Detecting Hidden Information in FAT. Монографія: In.: ISCI'2019: Information Security in Critical Infrastructures**. Collective monograph**. Edited by Ivan D. Gorbenko and Alexandr A. Kuznetsov, ASC Academic Publishing, USA, 2019, pp. 412-429. – ISBN: 978-0-9989826-8-7 (Hardback), ISBN: 978-0-9989826-9-4 (Ebook).

29. Smirnov, O., Kuznetsov, A., Kuznetsova., K. Synthesis of Discrete Signals with Improved Correlation Properties. Монографія: In.: ISCI'2019: Information Security in Critical Infrastructures. **Collective monograph**. Edited by Ivan D. Gorbenko and Alexandr A. Kuznetsov, ASC Academic Publishing, USA, 2019, pp. 281-299. – ISBN: 978-0-9989826-8-7 (Hardback), ISBN: 978-0-9989826-9-4 (Ebook).

30. О.А. Смірнов, П.С. Усік, «Дослідження перспектив використання технологічних рішень в мережах 5G» у *Кібербезпека та інформаційні технології: монографія.* – Х. : ТОВ «ДІСА ПЛЮС», 2020.С. 122-135.

31. Смірнов О.А., Дрєєва Г.М., «Метод генерування фрактального трафіку за допомогою моделі генератора на графі» у Інформаційна безпека та інформаційні технології: монографія / за заг. ред. В. С. Пономаренка. – Х. : Вид. Рожко С.Г. 2019. С. 123-139.

32. Смирнов А.А., Коваленко А.В. Комплекс математических моделей технологии тестирования WEB-приложений. Інформаційні технології: сучасний стан та перспективи: монографія / За загальною редакцією В.С. Пономаренка. – Х.: ТОВ «ДІСА ПЛЮС», 2018. – 461 с.

33. Смирнов А.А., Коваленко А.В. Разработка метода управления рисками разработки программного обеспечения. Інформаційні технології: проблеми та перспективи: монографія / За загальною редакцією В.С. Пономаренка. – Х.: Видавець Рожко С.Г., 2017. – 447 с.

34. Смірнов О.А., Смірнова Т.В., Якименко Н.М., Смірнов С.А., Поліщук Л.І., «Дослідження стійкості до диференціального криптоаналізу запропонованої функції гешування удосконаленого модуля криптографічного захисту в інформаційно-комунікаційних системах» *Системи управління, навігації та зв'язку,* 2022, № 3(69). С. 93-98. 2022.

35. Смірнов О.А., Смірнова Т.В., Якименко Н.М., Поліщук Л.І., Смірнов С.А. «Дослідження статистичної стійкості та швидкісних характеристик запропонованої функції гешування удосконаленого модуля криптографічного захисту в інформаційно-комунікаційних системах» *Вісник Хмельницького національного університету. Серія: «Технічні науки»,* № 2 (307). С. 46-52. 2022.

36. Смірнов О.А., Смірнова Т.В., Константинова Л.В., Смірнов С.А., Якименко Н.М., «Дослідження стійкості до лінійного криптоаналізу запропонованої функції гешування удосконаленого модуля криптографічного захисту в інформаційно-комунікаційних системах» *Системи управління, навігації та зв'язку*, 2022, № 1(67). С. 84-89.

37. Смірнов О.А., Смірнова Т.В., Буравченко К.О., Кравченко С.С., Горбов В.О., «Хмарна система підтримки прийняття рішень технологічного процесу відновлення поверхонь конструкцій і деталей машин». *Сучасні інформаційні системи*. 2021. Т. 5, № 4. С. 79-95

38. Смирнов А., Кузнецов А., Кузнецова Т. «Шумоподобные дискретные сигналы для асинхронных систем кодового разделения радиоканалов». *Радиотехника*, № 2(205), 175–183. 2021.

39. Smirnov O., Kuznetsov A., Kovalchuk D., Kuznetsova T. «New Technique for Hiding Data in Cover Images Using Adaptively Generated Pseudorandom Sequences». *CEUR Workshop Proceedings* Volume 2732, 2020, Pages 214-227.

40. Смірнов, О.А., Полігенько О.О., Одарченко Р.С., Терещенко Л.Ю.Усік П.С., «Інформаційна технологія та програмне забезпечення для підвищення ефективності планування підсистеми базових станцій стільникового зв'язку». *Проблеми телекомунікацій*. № 1(26). С. 83-96. 2020.

41. Смирнов А.А, Кузнецов А.А., Киян А.С., Кузнецова Е.А. «Сокрытие данных на основе адресации шумоподобных сигналов». *Всеукраїнський міжвідомчий науково-технічний збірник "Радіотехніка"* – Харьків: ХНУРЕ. – 2020. – Вип. 203. – С. 38-49.

42. Смирнов А.А., Дудан А.В., Смирнова Т.В. «Формализация структуры технологического процесса электродугового напыления». *Сборник научных трудов «Актуальные вопросы машиноведения».* Объединенный институт машиностроения Национальной Академии Наук Беларуси. №9. С. 308-312, 2020.

43. Смірнов О.А., Усік П.С., Миронець І.В., Буравченко К.О., Якименко Н.М. «Метод підвищення ефективності розподіленої обробки даних у комп'ютерних системах операторів стільникового зв'язку» *Вісник Черкаського державного технологічного університету. Технічні науки.* №4. С. 103-110. 2020.

44. О.А.Смірнов, Т.В.Смірнова, Л.І. Поліщук, К.О. Буравченко, А.О.Макевнін, «Дослідження хмарних технологій як сервісів», *Кібербезпека: освіта, наука, техніка*. № 3(7). С. 43-62. 2020.

45. А.А. Смирнов, Т.В. Смирнова, А.Н. Дреев, А.В. Дудан. «Оптимизация технологического процесса восстановления и упрочнения поверхностей с заданными характеристиками в виде облачного сервиса». Вестник Полоцкого государственного университета. Серия В, Промышленность. Прикладные науки. Республика Беларусь - 2020. - № 3. - С. 50-61.

46. Смірнов О.А., Дрєєва Г.М., Дрєєв О.М., Смірнова Т.В. «Фрактальний аналіз генератора самоподібного трафіку на основі ланцюга

Маркова». *Центральноукраїнський науковий вісник. Технічні науки.* № 2(33). с. 161-172, 2019.

47. О.А. Смірнов, Т.В. Смірнова, О.М. Дрєєв, Є.К. Солових, «Методи оптимізації технологічних процесів відновлення сталевих покриттів», *Shipbuilding & marine infrastructure / Суднобудування і морська інфраструктура* № 1 (11). с. 48-57, 2019.

48. Смірнов О.А., Дрєєва Г.М., Дрєєв О.М., «Побудова хмарних інформаційних технологій оптимізації технологічного процесу відновлення та зміцнення поверхонь деталей». Центральноукраїнський науковий вісник. Технічні науки. № 1(32). с. 184-194, 2019.

49. Смірнов О.А., Смірнова Т.В., Солових Є.К., Дрєєв О.М., «Побудова хмарних інформаційних технологій оптимізації технологічного процесу відновлення та зміцнення поверхонь деталей». Центральноукраїнський науковий вісник. Технічні науки. № 1(32). с. 184-194, 2019.

50. Смірнов О.А., Смірнова Т.В.,. Дрєєв О.М, «Експертна система оптимізації процесу відновлення та зміцнення поверхонь деталей типу «вал» електродуговим напиленням», Системи управління, навігації та зв'язку, № 2 (54). с. 149-154, 2019.

51. Смірнов О.А., Смірнов С.А., Поліщук Л.І., Смірнова Т.В., Конопліцька-Слободенюк О.К. Метод формування антивірусного захисту даних з використанням безпечної маршрутизації метаданих. Кібербезпека: освіта, наука, техніка. – Том 3 № 3. – Київ: КУ ім. Бориса Грінченка. – 2019. – С. 63-87.

52. Смирнов А.А., Лысенко И.А., Информационная технология проектирования тестовых наборов на основе требований к программному обеспечению, Системи управління, навігації та зв'язку. – Випуск 4 (44). – Полтава: ПолтНТУ. – 2017. – С. 112-115.

53. Смірнов О.А., Мелешко Є.В., Хох В.Д., Дослідження методів аудиту систем управління інформаційною безпекою, Системи управління, навігації та зв'язку. – Випуск 1 (41). – Полтава: ПолтНТУ. – 2017. – С. 38-42.

Додаток А
(обов'язковий)

Технічне завдання

Зміст

1 Найменування та область застосування.....2
2 Підстава для розробки.....2
3 Мета та призначення розробки.....2
4 Джерела розробки.....2
5 Технічні вимоги.....2
5.1 Вміст проекту.....2
5.2 Показники призначення.....3
5.3 Вимоги до функціональних характеристик.....3
5.4 Вимоги до архітектури.....3
5.5 Вимоги до надійності.....3
5.6 Умови експлуатації.....4
5.7 Вимоги до складу та параметрів технічних засобів.....4
5.8 Вимоги до інформаційної і програмної сумісності.....4
5.8.1 Обладнання.....4
5.8.2 Мова програмування.....4
5.8.3 Вхідні дані.....5
5.8.4 Вихідні дані.....5
6 Вимоги до програмної документації.....5
7 Перелік документів, що розробляються.....5
8 Етапи розробки.....6
9 Порядок контролю та приймання.....6

1 Найменування та область застосування

Це технічне завдання розповсюджується на розробку системи кібербезпеки ЕЦП для автентифікації користувачів у системах дистанційного навчання та тестування.

2 Підстава для розробки

Підставою для розробки служить завдання на випускну кваліфікаційну роботу за першим (бакалаврським) рівнем вищої освіти, видане на кафедрі кібербезпеки та програмного забезпечення (нак. № 16-02 від 5.01.2023 року).

3 Мета та призначення розробки

Метою випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти є розробка програмного забезпечення системи кібербезпеки ЕЦП для автентифікації користувачів у системах дистанційного навчання та тестування.

4 Джерела розробки

Джерелом цієї випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти є стосовна до теми література і існуючі аналоги.

5 Технічні вимоги

5.1 Склад продукції

Складниками розробки є:

– вибір і обґрунтування методів реалізації проекту;

– розробка програмної частин системи, а також розробка взаємодії системи кібербезпеки з ОС та з користувачем;

– розробка програми, що реалізує спроектовані алгоритми роботи системи.

5.2 Показники призначення

Система повинна забезпечувати:

– системи кібербезпеки ЕЦП для автентифікації користувачів у системах дистанційного навчання та тестування;

– цілісність даних у процесі роботи та при зберіганні;

– простий, інтуїтивно зрозумілий інтерфейс.

5.3 Вимоги до функціональних характеристик

Розроблене програмне забезпечення не повинно мати обмежень на версію драйверів та операційної системи.

5.4 Вимоги до архітектури

Компонент, що розробляється повинен використовувати системні засоби та апаратні засоби, що на даному етапі розвитку обчислювальної техніки найбільше поширені.

5.5 Вимоги до надійності

Програмні модулі написані по всім правилам, які стосуються стандартних викликів процедур, функцій, методів і форм, визначених технічною документацією на середовище розробки.

5.6 Умови експлуатації

Робочі місця користувачів ПЗ повинні задовольняти наступним умовам експлуатації:

– температура повітря: 19-20 град. по Цельсію;

– відносна вологість повітря до 80%;

– атмосферний тиск 107 кПа.

5.7 Вимоги до складу та параметрів технічних засобів

Програмне забезпечення повинно бути реалізоване на ПЕОМ архітектури IBM PC, працювати в ОС Windows 10/11 і з сумісними з цією платформою пристроями і прикладним програмним забезпеченням.

5.8 Вимоги до інформаційної і програмної сумісності

Переносність програмного забезпечення повинна бути забезпечена за рахунок його реалізації стандартного інтерфейсу взаємодії з ОС, що працюють під управлінням ОС Windows 10/11.

5.8.1 Обладнання

Комп'ютер Intel® Celeron/8 Mb/1.2 Gb/SVGA 14" 1Mb або сумісні з ним.

5.8.2 Мова програмування

Середовище Visual C#.

5.8.3 Вхідні дані

Опис алгоритму роботи запропонованої системи.

5.8.4 Вихідні дані

Робоча програма.

6 Вимоги до програмної документації

Програмна продукція повинна бути представлена у виді опису структури даних, схем та опису алгоритму, а також текстів вихідних модулів програмного забезпечення згідно ЄСПД .

7 Перелік документів, що розробляються

– Структурна схема системи – 1 аркуш.
– Функціональна схема системи – 1 аркуш.
– Діаграма процесів – 1 аркуш.
– Блок-схема алгоритму роботи програми – 2 аркуша.
– Пояснювальна записка – 92 аркуші.

8 Етапи розробки

8.1 Збір і обробка інформації по темі випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти. Постановка задачі на виконання випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти (складання ТЗ).

8.2 Проведення досліджень або експериментальних робіт для уточнення основних положень випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти.

8.3 Розробка функціональних схем, блок схем алгоритмів роботи програмного забезпечення.

8.4 Побудова схем взаємодії даних.

8.5 Створення прототипу ПЗ.

8.6 Віднаходження ПЗ, аналіз отриманих результатів.

8.7 Оформлення пояснювальної записки і виконання робіт по графічній частині.

9 Порядок контролю та приймання

9.1 Подання випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти на попередній захист 23.05.2023 р.

9.2 Подання випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти на захист 8.06.2023 р.

Додаток Б

(обов’язковий)

Міністерство освіти і науки України

Центральноукраїнський національний технічний університет

ЗАТВЕРДЖУЮ

Керівник випускної кваліфікаційної роботи за

першим (бакалаврським) рівнем вищої освіти

_____ Смірнов О.А.

Програмне забезпечення системи кібербезпеки ЕЦП для автентифікації користувачів у системах дистанційного навчання та тестування

Лістинг програми

Код документу 12

Носій: CD/DVD-диск / USB-флеш-накопичувач

Загальна кількість аркушів: 31

Літера: РП

Кропивницький – 2023 року

// Form1.cs – основна програма

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.IO;
using Org.BouncyCastle.Crypto.Signers;
using Org.BouncyCastle.Math;
using Org.BouncyCastle.Math.EC;
using Org.BouncyCastle.Crypto.Generators;
using Org.BouncyCastle.Crypto.Parameters;
using Org.BouncyCastle.Security;
using Org.BouncyCastle.Crypto;
using Org.BouncyCastle.X509;
using Org.BouncyCastle.Security.Certificates;
using Org.BouncyCastle.Asn1.X509;
using Org.BouncyCastle.Crypto.Digests;

namespace BlindGostDemo
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        private SecureRandom random;
        private FpCurve curve;
        private ECDomainParameters parameters;
        private BigInteger mod_p;

        private void Init()
        {
            random = new SecureRandom();

            mod_p = new
BigInteger("57896044618658097711785492504343953926634992332820282019728792003956
564821041"); //параметр p

            curve = new FpCurve(
                mod_p, // параметр p
                new BigInteger("7"), // параметр a
                new
BigInteger("43308876546767276905765904595650931995942111794451039583252968842033
849580414")); // параметр b

            parameters = new ECDomainParameters(
                        curve,
                        new FpPoint(curve,
                        new FpFieldElement(mod_p, new BigInteger("2")), //
параметр x
                        new FpFieldElement(mod_p, new
BigInteger("40189740565390375033354494229370597756357393899055450806909793652134
31566280"))), // параметр y
                        new
BigInteger("57896044618658097711785492504343953927082934583725450622380973592137
631069619")); // параметр q

            string message = "Тестування";
            tbH.Text =
GetDigest(System.Text.Encoding.Default.GetBytes(message)).ToString(16);
```

```
            //tbH.Text = (new
BigInteger("20798893674476452017134061561508270130637142515379653289952617252661
468872421")).ToString(16);
        }

        public BigInteger GetDigest(byte[] message)
        {
            Gost3411Digest gost3411Digest = new Gost3411Digest();
            gost3411Digest.BlockUpdate(message, 0, message.Length);
            byte[] hashmessage = new byte[gost3411Digest.GetDigestSize()];
            gost3411Digest.DoFinal(hashmessage, 0);
            return new BigInteger(hashmessage);
        }


        private BigInteger TextBoxToBigInteger16(TextBox tb)
        {
            return new BigInteger(tb.Text, 16);
        }

        private void button1_Click(object sender, EventArgs e)
        {


            ECKeyGenerationParameters keyGenerationParameters = new
ECKeyGenerationParameters(parameters, random);

            ECKeyPairGenerator keygenerator = new
ECKeyPairGenerator("ECGOST3410");
            keygenerator.Init(keyGenerationParameters);
            AsymmetricCipherKeyPair pair = keygenerator.GenerateKeyPair();

            ECPrivateKeyParameters validatorPrivate =
(ECPrivateKeyParameters)pair.Private;
            ECPublicKeyParameters validatorPublic =
(ECPublicKeyParameters)pair.Public;

            /*validatorPrivate = new ECPrivateKeyParameters(
                "ECGOST3410",
                new
BigInteger("55441196065363246126355624130324183196576709222340016572108097750006
097525544"), // параметр d
                parameters);

            validatorPublic = new ECPublicKeyParameters(
                "ECGOST3410",
                new FpPoint(curve,
                new FpFieldElement(mod_p, new
BigInteger("57520216126176808443631405023338071176630104906313632182896741342206
604859403")), // параметр x
                new FpFieldElement(mod_p, new
BigInteger("17614944419213781543809391949654080031942662045363639260709847859438
286763994"))), // параметр y
                parameters);
            */


            tbValPrivate.Text = validatorPrivate.D.ToString(16);
            tbValPublicX.Text = validatorPublic.Q.X.ToBigInteger().ToString(16);
            tbValPublicY.Text = validatorPublic.Q.Y.ToBigInteger().ToString(16);


        }

        private void button2_Click(object sender, EventArgs ea)
        {
            ECGost3410Signer signer = new ECGost3410Signer();

            ECPublicKeyParameters pubKey = new ECPublicKeyParameters(
```

```
                "ECGOST3410",
                new FpPoint(curve,
                new FpFieldElement(mod_p, TextBoxToBigInteger16(tbValPublicX)),
// параметр x
                new FpFieldElement(mod_p, TextBoxToBigInteger16(tbValPublicY))),
// параметр y
                parameters);

            BigInteger H = TextBoxToBigInteger16(tbH);
            BigInteger rs = TextBoxToBigInteger16(tbrs);
            BigInteger ss = TextBoxToBigInteger16(tbss);
            BigInteger q = parameters.N;

            //FpPoint G = (FpPoint)parameters.G;
            //FpPoint Q = new FpPoint(curve, new FpFieldElement(mod_p,
TextBoxToBigInteger16(tbValPublicX)), new FpFieldElement(mod_p,
TextBoxToBigInteger16(tbValPublicY)));

            BigInteger e = H.Mod(q);
            byte[] ee = e.ToByteArray();
            byte[] message = H.ToByteArray();
            Array.Reverse(message);

            signer.Init(false, pubKey);

            MessageBox.Show(signer.VerifySignature(message, rs, ss).ToString(),
"Перевірка підпису");

            //FpPoint C =
(FpPoint)(G.Multiply(e.ModInverse(q).Multiply(ss).Mod(q)).Subtract(Q.Multiply(e.
ModInverse(q).Multiply(rs).Mod(q))));
            //BigInteger x = C.X.ToBigInteger();
        }

        private void button3_Click(object sender, EventArgs e)
        {
            FpPoint G = (FpPoint)parameters.G;
            BigInteger k = (new BigInteger(random.Next(1, parameters.N.BitCount
- 1), random)).Add(BigInteger.One);

            tbk.Text = k.ToString(16);
            FpPoint C = (FpPoint)G.Multiply(k);
            tbCX.Text = C.X.ToBigInteger().ToString(16);
            tbCY.Text = C.Y.ToBigInteger().ToString(16);

        }

        private void Form1_Load(object sender, EventArgs e)
        {

            Init();
        }

        private void button4_Click(object sender, EventArgs e)
        {
            BigInteger mu = (new BigInteger(random.Next(1, parameters.N.BitCount
- 1), random)).Add(BigInteger.One);
            BigInteger epsilon = (new BigInteger(random.Next(1,
parameters.N.BitCount - 1), random)).Add(BigInteger.One);
            BigInteger delta = (new BigInteger(random.Next(1,
parameters.N.BitCount - 1), random)).Add(BigInteger.One);
            BigInteger tau = (new BigInteger(random.Next(1,
parameters.N.BitCount - 1), random)).Add(BigInteger.One);
            tbVoterMu.Text = mu.ToString(16);
            tbVoterEpsilon.Text = epsilon.ToString(16);
            tbVoterDelta.Text = delta.ToString(16);
            tbVoterTau.Text = tau.ToString(16);
        }
```

```
        private void button5_Click(object sender, EventArgs e)
        {
            FpPoint G = (FpPoint)parameters.G;
            FpPoint Q = new FpPoint(curve, new FpFieldElement(mod_p,
TextBoxToBigInteger16(tbValPublicX)), new FpFieldElement(mod_p,
TextBoxToBigInteger16(tbValPublicY)));
            FpPoint C = new FpPoint(curve, new FpFieldElement(mod_p,
TextBoxToBigInteger16(tbCX)), new FpFieldElement(mod_p,
TextBoxToBigInteger16(tbCY)));
            BigInteger mu = TextBoxToBigInteger16(tbVoterMu);
            BigInteger epsilon = TextBoxToBigInteger16(tbVoterEpsilon);
            BigInteger delta = TextBoxToBigInteger16(tbVoterDelta);
            BigInteger tau = TextBoxToBigInteger16(tbVoterTau);
            BigInteger q = parameters.N;

            FpPoint Cs =
(FpPoint)G.Multiply(epsilon).Add(Q.Multiply(mu)).Add(C.Multiply(delta.ModInverse
(q)));
            tbCsX.Text = Cs.X.ToBigInteger().ToString(16);
            tbCsY.Text = Cs.Y.ToBigInteger().ToString(16);
        }

        private void button6_Click(object sender, EventArgs ea)
        {
            BigInteger H = TextBoxToBigInteger16(tbH);
            BigInteger q = parameters.N;
            BigInteger mu = TextBoxToBigInteger16(tbVoterMu);
            BigInteger delta = TextBoxToBigInteger16(tbVoterDelta);
            BigInteger tau = TextBoxToBigInteger16(tbVoterTau);
            BigInteger csx = TextBoxToBigInteger16(tbCsX);

            BigInteger rs = csx.Mod(q);
            BigInteger es = H.Mod(q);
            BigInteger r =
(tau.Multiply(delta).Multiply(rs.Add(mu.Multiply(es)))).Mod(q);
            BigInteger e = (es.Multiply(tau)).Mod(q);

            tbrs.Text = rs.ToString(16);
            tbr.Text = r.ToString(16);
            tbes.Text = es.ToString(16);
            tbe.Text = e.ToString(16);
        }

        private void button7_Click(object sender, EventArgs ea)
        {
            BigInteger k = TextBoxToBigInteger16(tbk);
            BigInteger e = TextBoxToBigInteger16(tbe);
            BigInteger d = TextBoxToBigInteger16(tbValPrivate);
            BigInteger r = TextBoxToBigInteger16(tbr);
            BigInteger q = parameters.N;

            BigInteger s = (k.Multiply(e).Add(d.Multiply(r))).Mod(q);

            tbs.Text = s.ToString(16);
        }

        private void button8_Click(object sender, EventArgs e)
        {
            BigInteger epsilon = TextBoxToBigInteger16(tbVoterEpsilon);
            BigInteger delta = TextBoxToBigInteger16(tbVoterDelta);
            BigInteger tau = TextBoxToBigInteger16(tbVoterTau);
            BigInteger s = TextBoxToBigInteger16(tbs);
            BigInteger es = TextBoxToBigInteger16(tbes);
            BigInteger q = parameters.N;

            BigInteger ss =
(s.Multiply(delta.ModInverse(q)).Multiply(tau.ModInverse(q)).Mod(q).Add(epsilon.
Multiply(es).Mod(q))).Mod(q);
```

```
            tbss.Text = ss.ToString(16);
        }

        private void groupBox2_Enter(object sender, EventArgs e)
        {

        }

    }
}
```

// Form1.Designer.cs – модуль, в якому реалізовано ЕЦП

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.IO;
using Org.BouncyCastle.Crypto.Signers;
using Org.BouncyCastle.Math;
using Org.BouncyCastle.Math.EC;
using Org.BouncyCastle.Crypto.Generators;
using Org.BouncyCastle.Crypto.Parameters;
using Org.BouncyCastle.Security;
using Org.BouncyCastle.Crypto;
using Org.BouncyCastle.X509;
using Org.BouncyCastle.Security.Certificates;
using Org.BouncyCastle.Asn1.X509;
using Org.BouncyCastle.Crypto.Digests;

namespace BlindGostDemo
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        private SecureRandom random;
        private FpCurve curve;
        private ECDomainParameters parameters;
        private BigInteger mod_p;

        private void Init()
        {
            random = new SecureRandom();

            mod_p = new
BigInteger("57896044618658097711785492504343953926634992332820282019728792003956
564821041"); // параметр p

            curve = new FpCurve(
                mod_p, // параметр p
                new BigInteger("7"), // параметр a
                new
BigInteger("43308876546767276905765904595650931995942111794451039583252968842033
849580414")); // параметр b

            parameters = new ECDomainParameters(
                        curve,
                        new FpPoint(curve,
                        new FpFieldElement(mod_p, new BigInteger("2")), //
параметр x
                        new FpFieldElement(mod_p, new
BigInteger("40189740565390375033354494229370597756357393899055450806909793652134
31566280"))), // параметр y
                        new
BigInteger("57896044618658097711785492504343953927082934583725450622380973592137
631069619")); // параметр q

            string message = "Тестування";
            tbH.Text =
GetDigest(System.Text.Encoding.Default.GetBytes(message)).ToString(16);
```

```
            //tbH.Text = (new
BigInteger("207988936744764520171340615615082701306371425153796532899526172526614
68872421")).ToString(16);
        }

        public BigInteger GetDigest(byte[] message)
        {
            Gost3411Digest gost3411Digest = new Gost3411Digest();
            gost3411Digest.BlockUpdate(message, 0, message.Length);
            byte[] hashmessage = new byte[gost3411Digest.GetDigestSize()];
            gost3411Digest.DoFinal(hashmessage, 0);
            return new BigInteger(hashmessage);
        }


        private BigInteger TextBoxToBigInteger16(TextBox tb)
        {
            return new BigInteger(tb.Text, 16);
        }

        private void button1_Click(object sender, EventArgs e)
        {


            ECKeyGenerationParameters keyGenerationParameters = new
ECKeyGenerationParameters(parameters, random);

            ECKeyPairGenerator keygenerator = new
ECKeyPairGenerator("ECGOST3410");
            keygenerator.Init(keyGenerationParameters);
            AsymmetricCipherKeyPair pair = keygenerator.GenerateKeyPair();

            ECPrivateKeyParameters validatorPrivate =
(ECPrivateKeyParameters)pair.Private;
            ECPublicKeyParameters validatorPublic =
(ECPublicKeyParameters)pair.Public;

            /*validatorPrivate = new ECPrivateKeyParameters(
                "ECGOST3410",
                new
BigInteger("554411960653632461263556241303241831965767092223400165721080977500060
97525544"), // параметр d
                parameters);

            validatorPublic = new ECPublicKeyParameters(
                "ECGOST3410",
                new FpPoint(curve,
                new FpFieldElement(mod_p, new
BigInteger("575202161261768084436314050233380711766301049063136321828967413422066
04859403")), // параметр x
                new FpFieldElement(mod_p, new
BigInteger("176149444192137815438093919496540800319426620453636392607098478594382
86763994"))), // параметр y
                parameters);
            */


            tbValPrivate.Text = validatorPrivate.D.ToString(16);
            tbValPublicX.Text = validatorPublic.Q.X.ToBigInteger().ToString(16);
            tbValPublicY.Text = validatorPublic.Q.Y.ToBigInteger().ToString(16);


        }

        private void button2_Click(object sender, EventArgs ea)
        {
            ECGost3410Signer signer = new ECGost3410Signer();

            ECPublicKeyParameters pubKey = new ECPublicKeyParameters(
```

```
                "ECGOST3410",
                new FpPoint(curve,
                new FpFieldElement(mod_p, TextBoxToBigInteger16(tbValPublicX)),
// параметр x
                new FpFieldElement(mod_p, TextBoxToBigInteger16(tbValPublicY))),
// параметр y
                parameters);

            BigInteger H = TextBoxToBigInteger16(tbH);
            BigInteger rs = TextBoxToBigInteger16(tbrs);
            BigInteger ss = TextBoxToBigInteger16(tbss);
            BigInteger q = parameters.N;

            //FpPoint G = (FpPoint)parameters.G;
            //FpPoint Q = new FpPoint(curve, new FpFieldElement(mod_p,
TextBoxToBigInteger16(tbValPublicX)), new FpFieldElement(mod_p,
TextBoxToBigInteger16(tbValPublicY)));

            BigInteger e = H.Mod(q);
            byte[] ee = e.ToByteArray();
            byte[] message = H.ToByteArray();
            Array.Reverse(message);

            signer.Init(false, pubKey);

            MessageBox.Show(signer.VerifySignature(message, rs, ss).ToString(),
"Перевірка підпису");

            //FpPoint C =
(FpPoint)(G.Multiply(e.ModInverse(q).Multiply(ss).Mod(q)).Subtract(Q.Multiply(e.
ModInverse(q).Multiply(rs).Mod(q))));
            //BigInteger x = C.X.ToBigInteger();
        }

        private void button3_Click(object sender, EventArgs e)
        {
            FpPoint G = (FpPoint)parameters.G;
            BigInteger k = (new BigInteger(random.Next(1, parameters.N.BitCount
- 1), random)).Add(BigInteger.One);

            tbk.Text = k.ToString(16);
            FpPoint C = (FpPoint)G.Multiply(k);
            tbCX.Text = C.X.ToBigInteger().ToString(16);
            tbCY.Text = C.Y.ToBigInteger().ToString(16);

        }

        private void Form1_Load(object sender, EventArgs e)
        {

            Init();
        }

        private void button4_Click(object sender, EventArgs e)
        {
            BigInteger mu = (new BigInteger(random.Next(1, parameters.N.BitCount
- 1), random)).Add(BigInteger.One);
            BigInteger epsilon = (new BigInteger(random.Next(1,
parameters.N.BitCount - 1), random)).Add(BigInteger.One);
            BigInteger delta = (new BigInteger(random.Next(1,
parameters.N.BitCount - 1), random)).Add(BigInteger.One);
            BigInteger tau = (new BigInteger(random.Next(1,
parameters.N.BitCount - 1), random)).Add(BigInteger.One);
            tbVoterMu.Text = mu.ToString(16);
            tbVoterEpsilon.Text = epsilon.ToString(16);
            tbVoterDelta.Text = delta.ToString(16);
            tbVoterTau.Text = tau.ToString(16);
        }
```

```
        private void button5_Click(object sender, EventArgs e)
        {
            FpPoint G = (FpPoint)parameters.G;
            FpPoint Q = new FpPoint(curve, new FpFieldElement(mod_p,
TextBoxToBigInteger16(tbValPublicX)), new FpFieldElement(mod_p,
TextBoxToBigInteger16(tbValPublicY)));
            FpPoint C = new FpPoint(curve, new FpFieldElement(mod_p,
TextBoxToBigInteger16(tbCX)), new FpFieldElement(mod_p,
TextBoxToBigInteger16(tbCY)));
            BigInteger mu = TextBoxToBigInteger16(tbVoterMu);
            BigInteger epsilon = TextBoxToBigInteger16(tbVoterEpsilon);
            BigInteger delta = TextBoxToBigInteger16(tbVoterDelta);
            BigInteger tau = TextBoxToBigInteger16(tbVoterTau);
            BigInteger q = parameters.N;

            FpPoint Cs =
(FpPoint)G.Multiply(epsilon).Add(Q.Multiply(mu)).Add(C.Multiply(delta.ModInverse
(q)));
            tbCsX.Text = Cs.X.ToBigInteger().ToString(16);
            tbCsY.Text = Cs.Y.ToBigInteger().ToString(16);
        }

        private void button6_Click(object sender, EventArgs ea)
        {
            BigInteger H = TextBoxToBigInteger16(tbH);
            BigInteger q = parameters.N;
            BigInteger mu = TextBoxToBigInteger16(tbVoterMu);
            BigInteger delta = TextBoxToBigInteger16(tbVoterDelta);
            BigInteger tau = TextBoxToBigInteger16(tbVoterTau);
            BigInteger csx = TextBoxToBigInteger16(tbCsX);

            BigInteger rs = csx.Mod(q);
            BigInteger es = H.Mod(q);
            BigInteger r =
(tau.Multiply(delta).Multiply(rs.Add(mu.Multiply(es)))).Mod(q);
            BigInteger e = (es.Multiply(tau)).Mod(q);

            tbrs.Text = rs.ToString(16);
            tbr.Text = r.ToString(16);
            tbes.Text = es.ToString(16);
            tbe.Text = e.ToString(16);
        }

        private void button7_Click(object sender, EventArgs ea)
        {
            BigInteger k = TextBoxToBigInteger16(tbk);
            BigInteger e = TextBoxToBigInteger16(tbe);
            BigInteger d = TextBoxToBigInteger16(tbValPrivate);
            BigInteger r = TextBoxToBigInteger16(tbr);
            BigInteger q = parameters.N;

            BigInteger s = (k.Multiply(e).Add(d.Multiply(r))).Mod(q);

            tbs.Text = s.ToString(16);
        }

        private void button8_Click(object sender, EventArgs e)
        {
            BigInteger epsilon = TextBoxToBigInteger16(tbVoterEpsilon);
            BigInteger delta = TextBoxToBigInteger16(tbVoterDelta);
            BigInteger tau = TextBoxToBigInteger16(tbVoterTau);
            BigInteger s = TextBoxToBigInteger16(tbs);
            BigInteger es = TextBoxToBigInteger16(tbes);
            BigInteger q = parameters.N;

            BigInteger ss =
(s.Multiply(delta.ModInverse(q)).Multiply(tau.ModInverse(q)).Mod(q).Add(epsilon.
Multiply(es).Mod(q))).Mod(q);
```

```
            tbss.Text = ss.ToString(16);
        }

        private void groupBox2_Enter(object sender, EventArgs e)
        {

        }

    }
}
```

// MD5Digest.cs – модуль, в якому реалізовано алгоритм хешування MD5

```
using System;

namespace Org.BouncyCastle.Crypto.Digests
{

    public class MD5Digest
            : GeneralDigest
    {
        private const int DigestLength = 16;

        private int H1, H2, H3, H4;

        private int[]   X = new int[16];
        private int           xOff;

        public MD5Digest()
        {
            Reset();
        }


        public MD5Digest(MD5Digest t)
                  : base(t)
        {
            H1 = t.H1;
            H2 = t.H2;
            H3 = t.H3;
            H4 = t.H4;

            Array.Copy(t.X, 0, X, 0, t.X.Length);
            xOff = t.xOff;
        }

            public override string AlgorithmName
            {
                  get { return "MD5"; }
            }

            public override int GetDigestSize()
            {
                  return DigestLength;
            }

            internal override void ProcessWord(
            byte[]  input,
            int     inOff)
        {
            X[xOff++] = (input[inOff] & 0xff) | ((input[inOff + 1] & 0xff) << 8)
                | ((input[inOff + 2] & 0xff) << 16) | ((input[inOff + 3] & 0xff)
<< 24);


            if (xOff == 16)
            {
                ProcessBlock();
            }
        }

        internal override void ProcessLength(
            long    bitLength)
        {
            if (xOff > 14)
            {
                ProcessBlock();
            }

            X[14] = (int)(bitLength & 0xffffffff);
```

```
    X[15] = (int)((ulong) bitLength >> 32);
}

private void UnpackWord(
    int     word,
    byte[]  outBytes,
    int     outOff)
{
    outBytes[outOff]     = (byte)word;
    outBytes[outOff + 1] = (byte)((uint) word >> 8);
    outBytes[outOff + 2] = (byte)((uint) word >> 16);
    outBytes[outOff + 3] = (byte)((uint) word >> 24);
}

public override int DoFinal(
    byte[]  output,
    int     outOff)
{
    Finish();

    UnpackWord(H1, output, outOff);
    UnpackWord(H2, output, outOff + 4);
    UnpackWord(H3, output, outOff + 8);
    UnpackWord(H4, output, outOff + 12);

    Reset();

    return DigestLength;
}

/**
* Скидання ланцюжка змінних в IV значення
*/
public override void Reset()
{
    base.Reset();

    H1 = unchecked((int) 0x67452301);
    H2 = unchecked((int) 0xefcdab89);
    H3 = unchecked((int) 0x98badcfe);
    H4 = unchecked((int) 0x10325476);

    xOff = 0;

    for (int i = 0; i != X.Length; i++)
    {
        X[i] = 0;
    }
}

//
//1 раунд
//
private static readonly int S11 = 7;
private static readonly int S12 = 12;
private static readonly int S13 = 17;
private static readonly int S14 = 22;

//
// 2 раунд
//
private static readonly int S21 = 5;
private static readonly int S22 = 9;
private static readonly int S23 = 14;
private static readonly int S24 = 20;

//
// 3 раунд
//
```

```
private static readonly int S31 = 4;
private static readonly int S32 = 11;
private static readonly int S33 = 16;
private static readonly int S34 = 23;

//
// 4 раунд
//
private static readonly int S41 = 6;
private static readonly int S42 = 10;
private static readonly int S43 = 15;
private static readonly int S44 = 21;

/*
* зсув змінної x вліво на n бітів
*/
private int RotateLeft(
    int x,
    int n)
{
    return (x << n) | (int) ((uint) x >> (32 - n));
}

/*
* F, G, H та I – базові функції MD5
*/
private int F(
    int u,
    int v,
    int w)
{
    return (u & v) | (~u & w);
}

private int G(
    int u,
    int v,
    int w)
{
    return (u & w) | (v & ~w);
}

private int H(
    int u,
    int v,
    int w)
{
    return u ^ v ^ w;
}

private int K(
    int u,
    int v,
    int w)
{
    return v ^ (u | ~w);
}

internal override void ProcessBlock()
{
    int a = H1;
    int b = H2;
    int c = H3;
    int d = H4;

    //
    // Раунд 1 - F цикл, 16 разів.
    //
```

```
            a = RotateLeft((a + F(b, c, d) + X[ 0] + unchecked((int)
0xd76aa478)), S11) + b;
            d = RotateLeft((d + F(a, b, c) + X[ 1] + unchecked((int)
0xe8c7b756)), S12) + a;
            c = RotateLeft((c + F(d, a, b) + X[ 2] + unchecked((int)
0x242070db)), S13) + d;
            b = RotateLeft((b + F(c, d, a) + X[ 3] + unchecked((int)
0xc1bdceee)), S14) + c;
            a = RotateLeft((a + F(b, c, d) + X[ 4] + unchecked((int)
0xf57c0faf)), S11) + b;
            d = RotateLeft((d + F(a, b, c) + X[ 5] + unchecked((int)
0x4787c62a)), S12) + a;
            c = RotateLeft((c + F(d, a, b) + X[ 6] + unchecked((int)
0xa8304613)), S13) + d;
            b = RotateLeft((b + F(c, d, a) + X[ 7] + unchecked((int)
0xfd469501)), S14) + c;
            a = RotateLeft((a + F(b, c, d) + X[ 8] + unchecked((int)
0x698098d8)), S11) + b;
            d = RotateLeft((d + F(a, b, c) + X[ 9] + unchecked((int)
0x8b44f7af)), S12) + a;
            c = RotateLeft((c + F(d, a, b) + X[10] + unchecked((int)
0xffff5bb1)), S13) + d;
            b = RotateLeft((b + F(c, d, a) + X[11] + unchecked((int)
0x895cd7be)), S14) + c;
            a = RotateLeft((a + F(b, c, d) + X[12] + unchecked((int)
0x6b901122)), S11) + b;
            d = RotateLeft((d + F(a, b, c) + X[13] + unchecked((int)
0xfd987193)), S12) + a;
            c = RotateLeft((c + F(d, a, b) + X[14] + unchecked((int)
0xa679438e)), S13) + d;
            b = RotateLeft((b + F(c, d, a) + X[15] + unchecked((int)
0x49b40821)), S14) + c;

            //
            // Раунд 2 - G цикд, 16 разів.
            //
            a = RotateLeft((a + G(b, c, d) + X[ 1] + unchecked((int)
0xf61e2562)), S21) + b;
            d = RotateLeft((d + G(a, b, c) + X[ 6] + unchecked((int)
0xc040b340)), S22) + a;
            c = RotateLeft((c + G(d, a, b) + X[11] + unchecked((int)
0x265e5a51)), S23) + d;
            b = RotateLeft((b + G(c, d, a) + X[ 0] + unchecked((int)
0xe9b6c7aa)), S24) + c;
            a = RotateLeft((a + G(b, c, d) + X[ 5] + unchecked((int)
0xd62f105d)), S21) + b;
            d = RotateLeft((d + G(a, b, c) + X[10] + unchecked((int)
0x02441453)), S22) + a;
            c = RotateLeft((c + G(d, a, b) + X[15] + unchecked((int)
0xd8a1e681)), S23) + d;
            b = RotateLeft((b + G(c, d, a) + X[ 4] + unchecked((int)
0xe7d3fbc8)), S24) + c;
            a = RotateLeft((a + G(b, c, d) + X[ 9] + unchecked((int)
0x21e1cde6)), S21) + b;
            d = RotateLeft((d + G(a, b, c) + X[14] + unchecked((int)
0xc33707d6)), S22) + a;
            c = RotateLeft((c + G(d, a, b) + X[ 3] + unchecked((int)
0xf4d50d87)), S23) + d;
            b = RotateLeft((b + G(c, d, a) + X[ 8] + unchecked((int)
0x455a14ed)), S24) + c;
            a = RotateLeft((a + G(b, c, d) + X[13] + unchecked((int)
0xa9e3e905)), S21) + b;
            d = RotateLeft((d + G(a, b, c) + X[ 2] + unchecked((int)
0xfcefa3f8)), S22) + a;
            c = RotateLeft((c + G(d, a, b) + X[ 7] + unchecked((int)
0x676f02d9)), S23) + d;
            b = RotateLeft((b + G(c, d, a) + X[12] + unchecked((int)
0x8d2a4c8a)), S24) + c;
```

```
            //
            // Раунд 3 - H цикл, 16 разів.
            //
            a = RotateLeft((a + H(b, c, d) + X[ 5] + unchecked((int)
0xfffa3942)), S31) + b;
            d = RotateLeft((d + H(a, b, c) + X[ 8] + unchecked((int)
0x8771f681)), S32) + a;
            c = RotateLeft((c + H(d, a, b) + X[11] + unchecked((int)
0x6d9d6122)), S33) + d;
            b = RotateLeft((b + H(c, d, a) + X[14] + unchecked((int)
0xfde5380c)), S34) + c;
            a = RotateLeft((a + H(b, c, d) + X[ 1] + unchecked((int)
0xa4beea44)), S31) + b;
            d = RotateLeft((d + H(a, b, c) + X[ 4] + unchecked((int)
0x4bdecfa9)), S32) + a;
            c = RotateLeft((c + H(d, a, b) + X[ 7] + unchecked((int)
0xf6bb4b60)), S33) + d;
            b = RotateLeft((b + H(c, d, a) + X[10] + unchecked((int)
0xbebfbc70)), S34) + c;
            a = RotateLeft((a + H(b, c, d) + X[13] + unchecked((int)
0x289b7ec6)), S31) + b;
            d = RotateLeft((d + H(a, b, c) + X[ 0] + unchecked((int)
0xeaa127fa)), S32) + a;
            c = RotateLeft((c + H(d, a, b) + X[ 3] + unchecked((int)
0xd4ef3085)), S33) + d;
            b = RotateLeft((b + H(c, d, a) + X[ 6] + unchecked((int)
0x04881d05)), S34) + c;
            a = RotateLeft((a + H(b, c, d) + X[ 9] + unchecked((int)
0xd9d4d039)), S31) + b;
            d = RotateLeft((d + H(a, b, c) + X[12] + unchecked((int)
0xe6db99e5)), S32) + a;
            c = RotateLeft((c + H(d, a, b) + X[15] + unchecked((int)
0x1fa27cf8)), S33) + d;
            b = RotateLeft((b + H(c, d, a) + X[ 2] + unchecked((int)
0xc4ac5665)), S34) + c;

            //
            // Раунд 4 - K цикл, 16 разів.
            //
            a = RotateLeft((a + K(b, c, d) + X[ 0] + unchecked((int)
0xf4292244)), S41) + b;
            d = RotateLeft((d + K(a, b, c) + X[ 7] + unchecked((int)
0x432aff97)), S42) + a;
            c = RotateLeft((c + K(d, a, b) + X[14] + unchecked((int)
0xab9423a7)), S43) + d;
            b = RotateLeft((b + K(c, d, a) + X[ 5] + unchecked((int)
0xfc93a039)), S44) + c;
            a = RotateLeft((a + K(b, c, d) + X[12] + unchecked((int)
0x655b59c3)), S41) + b;
            d = RotateLeft((d + K(a, b, c) + X[ 3] + unchecked((int)
0x8f0ccc92)), S42) + a;
            c = RotateLeft((c + K(d, a, b) + X[10] + unchecked((int)
0xffeff47d)), S43) + d;
            b = RotateLeft((b + K(c, d, a) + X[ 1] + unchecked((int)
0x85845dd1)), S44) + c;
            a = RotateLeft((a + K(b, c, d) + X[ 8] + unchecked((int)
0x6fa87e4f)), S41) + b;
            d = RotateLeft((d + K(a, b, c) + X[15] + unchecked((int)
0xfe2ce6e0)), S42) + a;
            c = RotateLeft((c + K(d, a, b) + X[ 6] + unchecked((int)
0xa3014314)), S43) + d;
            b = RotateLeft((b + K(c, d, a) + X[13] + unchecked((int)
0x4e0811a1)), S44) + c;
            a = RotateLeft((a + K(b, c, d) + X[ 4] + unchecked((int)
0xf7537e82)), S41) + b;
            d = RotateLeft((d + K(a, b, c) + X[11] + unchecked((int)
0xbd3af235)), S42) + a;
            c = RotateLeft((c + K(d, a, b) + X[ 2] + unchecked((int)
0x2ad7d2bb)), S43) + d;
```

```
            b = RotateLeft((b + K(c, d, a) + X[ 9] + unchecked((int)
0xeb86d391)), S44) + c;

            H1 += a;
            H2 += b;
            H3 += c;
            H4 += d;

            //
            // Скидання зміщення і очищення буфера
            //
            xOff = 0;
            for (int i = 0; i != X.Length; i++)
            {
                X[i] = 0;
            }
        }
    }

}
```

// ECKeyPairGenerator.cs – модуль, в якому реалізовано генерацію ключів

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.IO;
using Org.BouncyCastle.Crypto.Signers;
using Org.BouncyCastle.Math;
using Org.BouncyCastle.Math.EC;
using Org.BouncyCastle.Crypto.Generators;
using Org.BouncyCastle.Crypto.Parameters;
using Org.BouncyCastle.Security;
using Org.BouncyCastle.Crypto;
using Org.BouncyCastle.X509;
using Org.BouncyCastle.Security.Certificates;
using Org.BouncyCastle.Asn1.X509;
using Org.BouncyCastle.Crypto.Digests;

namespace BlindGostDemo
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        private SecureRandom random;
        private FpCurve curve;
        private ECDomainParameters parameters;
        private BigInteger mod_p;

        private void Init()
        {
            random = new SecureRandom();

            mod_p = new
BigInteger("57896044618658097711785492504343953926634992332820282019728792003956
564821041"); //p

            curve = new FpCurve(
                mod_p, // p
                new BigInteger("7"), // a
                new
BigInteger("43308876546767276905765904595650931995942111794451039583252968842033
849580414")); // b

            parameters = new ECDomainParameters(
                        curve,
                        new FpPoint(curve,
                        new FpFieldElement(mod_p, new BigInteger("2")), // x
                        new FpFieldElement(mod_p, new
BigInteger("40189740565390375033354494229370597756357393899055450806909793652134
31566280"))), // y
                        new
BigInteger("57896044618658097711785492504343953927082934583725450622380973592137
631069619")); // q

            string message = "Тестування";
            tbH.Text =
GetDigest(System.Text.Encoding.Default.GetBytes(message)).ToString(16);
```

```
            //tbH.Text = (new
BigInteger("207988936744764520171340615615082701306371425153796532899526172526614
68872421")).ToString(16);
        }

        public BigInteger GetDigest(byte[] message)
        {
            Gost3411Digest gost3411Digest = new Gost3411Digest();
            gost3411Digest.BlockUpdate(message, 0, message.Length);
            byte[] hashmessage = new byte[gost3411Digest.GetDigestSize()];
            gost3411Digest.DoFinal(hashmessage, 0);
            return new BigInteger(hashmessage);
        }


        private BigInteger TextBoxToBigInteger16(TextBox tb)
        {
            return new BigInteger(tb.Text, 16);
        }

        private void button1_Click(object sender, EventArgs e)
        {


            ECKeyGenerationParameters keyGenerationParameters = new
ECKeyGenerationParameters(parameters, random);

            ECKeyPairGenerator keygenerator = new
ECKeyPairGenerator("ECGOST3410");
            keygenerator.Init(keyGenerationParameters);
            AsymmetricCipherKeyPair pair = keygenerator.GenerateKeyPair();

            ECPrivateKeyParameters validatorPrivate =
(ECPrivateKeyParameters)pair.Private;
            ECPublicKeyParameters validatorPublic =
(ECPublicKeyParameters)pair.Public;

            /*validatorPrivate = new ECPrivateKeyParameters(
                "ECGOST3410",
                new
BigInteger("554411960653632461263556241303241831965767092223400165721080977500060
97525544"), // d
                parameters);

            validatorPublic = new ECPublicKeyParameters(
                "ECGOST3410",
                new FpPoint(curve,
                new FpFieldElement(mod_p, new
BigInteger("575202161261768084436314050233380711766301049063136321828967413422066
04859403")), // x
                new FpFieldElement(mod_p, new
BigInteger("176149444192137815438093919496540800319426620453636392607098478594382
86763994"))), // y
                parameters);
            */


            tbValPrivate.Text = validatorPrivate.D.ToString(16);
            tbValPublicX.Text = validatorPublic.Q.X.ToBigInteger().ToString(16);
            tbValPublicY.Text = validatorPublic.Q.Y.ToBigInteger().ToString(16);


        }

        private void button2_Click(object sender, EventArgs ea)
        {
            ECGost3410Signer signer = new ECGost3410Signer();

            ECPublicKeyParameters pubKey = new ECPublicKeyParameters(
```

```
                "ECGOST3410",
                new FpPoint(curve,
                new FpFieldElement(mod_p, TextBoxToBigInteger16(tbValPublicX)),
// x
                new FpFieldElement(mod_p, TextBoxToBigInteger16(tbValPublicY))),
// y
                parameters);

            BigInteger H = TextBoxToBigInteger16(tbH);
            BigInteger rs = TextBoxToBigInteger16(tbrs);
            BigInteger ss = TextBoxToBigInteger16(tbss);
            BigInteger q = parameters.N;

            //FpPoint G = (FpPoint)parameters.G;
            //FpPoint Q = new FpPoint(curve, new FpFieldElement(mod_p,
TextBoxToBigInteger16(tbValPublicX)), new FpFieldElement(mod_p,
TextBoxToBigInteger16(tbValPublicY)));

            BigInteger e = H.Mod(q);
            byte[] ee = e.ToByteArray();
            byte[] message = H.ToByteArray();
            Array.Reverse(message);

            signer.Init(false, pubKey);

            MessageBox.Show(signer.VerifySignature(message, rs, ss).ToString(),
"Перевірка підпису");

            //FpPoint C =
(FpPoint)(G.Multiply(e.ModInverse(q).Multiply(ss).Mod(q)).Subtract(Q.Multiply(e.
ModInverse(q).Multiply(rs).Mod(q))));
            //BigInteger x = C.X.ToBigInteger();
        }

        private void button3_Click(object sender, EventArgs e)
        {
            FpPoint G = (FpPoint)parameters.G;
            BigInteger k = (new BigInteger(random.Next(1, parameters.N.BitCount
- 1), random)).Add(BigInteger.One);

            tbk.Text = k.ToString(16);
            FpPoint C = (FpPoint)G.Multiply(k);
            tbCX.Text = C.X.ToBigInteger().ToString(16);
            tbCY.Text = C.Y.ToBigInteger().ToString(16);

        }

        private void Form1_Load(object sender, EventArgs e)
        {

            Init();
        }

        private void button4_Click(object sender, EventArgs e)
        {
            BigInteger mu = (new BigInteger(random.Next(1, parameters.N.BitCount
- 1), random)).Add(BigInteger.One);
            BigInteger epsilon = (new BigInteger(random.Next(1,
parameters.N.BitCount - 1), random)).Add(BigInteger.One);
            BigInteger delta = (new BigInteger(random.Next(1,
parameters.N.BitCount - 1), random)).Add(BigInteger.One);
            BigInteger tau = (new BigInteger(random.Next(1,
parameters.N.BitCount - 1), random)).Add(BigInteger.One);
            tbVoterMu.Text = mu.ToString(16);
            tbVoterEpsilon.Text = epsilon.ToString(16);
            tbVoterDelta.Text = delta.ToString(16);
            tbVoterTau.Text = tau.ToString(16);
        }
```

```
        private void button5_Click(object sender, EventArgs e)
        {
            FpPoint G = (FpPoint)parameters.G;
            FpPoint Q = new FpPoint(curve, new FpFieldElement(mod_p,
TextBoxToBigInteger16(tbValPublicX)), new FpFieldElement(mod_p,
TextBoxToBigInteger16(tbValPublicY)));
            FpPoint C = new FpPoint(curve, new FpFieldElement(mod_p,
TextBoxToBigInteger16(tbCX)), new FpFieldElement(mod_p,
TextBoxToBigInteger16(tbCY)));
            BigInteger mu = TextBoxToBigInteger16(tbVoterMu);
            BigInteger epsilon = TextBoxToBigInteger16(tbVoterEpsilon);
            BigInteger delta = TextBoxToBigInteger16(tbVoterDelta);
            BigInteger tau = TextBoxToBigInteger16(tbVoterTau);
            BigInteger q = parameters.N;

            FpPoint Cs =
(FpPoint)G.Multiply(epsilon).Add(Q.Multiply(mu)).Add(C.Multiply(delta.ModInverse
(q)));
            tbCsX.Text = Cs.X.ToBigInteger().ToString(16);
            tbCsY.Text = Cs.Y.ToBigInteger().ToString(16);
        }

        private void button6_Click(object sender, EventArgs ea)
        {
            BigInteger H = TextBoxToBigInteger16(tbH);
            BigInteger q = parameters.N;
            BigInteger mu = TextBoxToBigInteger16(tbVoterMu);
            BigInteger delta = TextBoxToBigInteger16(tbVoterDelta);
            BigInteger tau = TextBoxToBigInteger16(tbVoterTau);
            BigInteger csx = TextBoxToBigInteger16(tbCsX);

            BigInteger rs = csx.Mod(q);
            BigInteger es = H.Mod(q);
            BigInteger r =
(tau.Multiply(delta).Multiply(rs.Add(mu.Multiply(es)))).Mod(q);
            BigInteger e = (es.Multiply(tau)).Mod(q);

            tbrs.Text = rs.ToString(16);
            tbr.Text = r.ToString(16);
            tbes.Text = es.ToString(16);
            tbe.Text = e.ToString(16);
        }

        private void button7_Click(object sender, EventArgs ea)
        {
            BigInteger k = TextBoxToBigInteger16(tbk);
            BigInteger e = TextBoxToBigInteger16(tbe);
            BigInteger d = TextBoxToBigInteger16(tbValPrivate);
            BigInteger r = TextBoxToBigInteger16(tbr);
            BigInteger q = parameters.N;

            BigInteger s = (k.Multiply(e).Add(d.Multiply(r))).Mod(q);

            tbs.Text = s.ToString(16);
        }

        private void button8_Click(object sender, EventArgs e)
        {
            BigInteger epsilon = TextBoxToBigInteger16(tbVoterEpsilon);
            BigInteger delta = TextBoxToBigInteger16(tbVoterDelta);
            BigInteger tau = TextBoxToBigInteger16(tbVoterTau);
            BigInteger s = TextBoxToBigInteger16(tbs);
            BigInteger es = TextBoxToBigInteger16(tbes);
            BigInteger q = parameters.N;

            BigInteger ss =
(s.Multiply(delta.ModInverse(q)).Multiply(tau.ModInverse(q)).Mod(q).Add(epsilon.
Multiply(es).Mod(q))).Mod(q);
```

```
            tbss.Text = ss.ToString(16);
        }

        private void groupBox2_Enter(object sender, EventArgs e)
        {

        }

    }
}
```

// GOST3410ParametersGenerator.cs – модуль генерації параметрів шифрування

```
using System;

using Org.BouncyCastle.Crypto.Parameters;
using Org.BouncyCastle.Math;
using Org.BouncyCastle.Security;

namespace Org.BouncyCastle.Crypto.Generators
{
	/**
	 * генерувати відповідні параметри для GOST3410
	 */
	public class Gost3410ParametersGenerator
	{
		private int             size;
		private int             typeproc;
		private SecureRandom    init_random;

		/**
		 * initialise the key generator.
		 *
		 * @param size size of the key
		 * @param typeProcedure type procedure A,B = 1;  A',B' - else
		 * @param random random byte source.
		 * Ініціалізувати генератор ключів.
		 *
		 * @ Параметри розмір ключа
		 * @ Параметри типу процедури A,B = 1;  A',B' - інакше
		 * @ Параметри генератора псевдовипадкових чисел.
		 */
		public void Init(
			int             size,
			int             typeProcedure,
			SecureRandom    random)
		{
			this.size = size;
			this.typeproc = typeProcedure;
			this.init_random = random;
		}

		//Процедура A
		private int procedure_A(int x0, int c,  BigInteger[] pq, int size)
		{
			//Перевірка і виконання умови: 0<x<2^16; 0<c<2^16; c - odd.
			while(x0<0 || x0>65536)
			{
				x0 = init_random.NextInt()/32768;
			}

			while((c<0 || c>65536) || (c/2==0))
			{
				c = init_random.NextInt()/32768 + 1;
			}

			BigInteger C = BigInteger.ValueOf(c);
			BigInteger constA16 = BigInteger.ValueOf(19381);

			//Крок 1
			BigInteger[] y = new BigInteger[1]; // початкова довжина = 1
			y[0] = BigInteger.ValueOf(x0);

			// Крок 2
			int[] t = new int[1]; //початкова довжина = 1
			t[0] = size;
			int s = 0;
			for (int i=0; t[i]>=17; i++)
			{
				// розширення масив t
```

```
                        int[] tmp_t = new int[t.Length + 1];
///////////////
                              Array.Copy(t,0,tmp_t,0,t.Length);          //
розширення
                        t = new int[tmp_t.Length];                       //
масив t
                        Array.Copy(tmp_t, 0, t, 0, tmp_t.Length);
///////////////

                        t[i+1] = t[i]/2;
                        s = i+1;
                  }

                  // крок3
                  BigInteger[] p = new BigInteger[s+1];
                  p[s] = new BigInteger("8003",16);
                  int m = s-1;  // крок4

                  for (int i=0; i<s; i++)
                  {
                        int rm = t[m]/16;  // крок5

                  step6: for(;;)
                           {
                                 // крок 6
                                 BigInteger[] tmp_y = new BigInteger[y.Length];
///////////////
                                 Array.Copy(y,0,tmp_y,0,y.Length);         //
розширення
                                 y = new BigInteger[rm+1];
//  масив y
                                 Array.Copy(tmp_y,0,y,0,tmp_y.Length);
///////////////

                                 for (int j=0; j<rm; j++)
                                 {
                                       y[j+1] =
(y[j].Multiply(constA16).Add(C)).Mod(BigInteger.Two.Pow(16));
                                 }

                                 // крок 7
                                 BigInteger Ym = BigInteger.Zero;
                                 for (int j=0; j<rm; j++)
                                 {
                                       Ym = Ym.Add(y[j].ShiftLeft(16*j));
                                 }

                                 y[0] = y[rm]; // крок 8

                                 // крок 9
                                 BigInteger N = BigInteger.One.ShiftLeft(t[m]-
1).Divide(p[m+1]).Add(
                                       Ym.ShiftLeft(t[m]-
1).Divide(p[m+1].ShiftLeft(16*rm)));

                                 if (N.TestBit(0))
                                 {
                                       N = N.Add(BigInteger.One);
                                 }

                                 // крок 10

                                    for(;;)
                                    {
                                          // крок 11
                                          BigInteger NByLastP =
N.Multiply(p[m+1]);

                                          if (NByLastP.BitLength > t[m])
```

```
                                        {
                                              goto step6; // крок 12
                                        }

                                        p[m] = NByLastP.Add(BigInteger.One);

                                        // крок13
                                        if (BigInteger.Two.ModPow(NByLastP,
p[m]).CompareTo(BigInteger.One) == 0
                                              && BigInteger.Two.ModPow(N,
p[m]).CompareTo(BigInteger.One) != 0)
                                        {
                                              break;
                                        }

                                        N = N.Add(BigInteger.Two);
                                    }

                                if (--m < 0)
                                {
                                      pq[0] = p[0];
                                      pq[1] = p[1];
                                      return y[0].IntValue;
                    }

                                break; // крок 14
                            }
                        }
                        return y[0].IntValue;
                }

                // Процедура A'
                private long procedure_Aa(long x0, long c, BigInteger[] pq, int
size)
                {
                      // Перевірка і виконання умови 0<x<2^32; 0<c<2^32; c - odd.
                      while(x0<0 || x0>4294967296L)
                      {
                            x0 = init_random.NextInt()*2;
                      }

                      while((c<0 || c>4294967296L) || (c/2==0))
                      {
                            c = init_random.NextInt()*2+1;
                      }

                      BigInteger C = BigInteger.ValueOf(c);
                      BigInteger constA32 = BigInteger.ValueOf(97781173);

                      // крок1
                      BigInteger[] y = new BigInteger[1]; // початкова довжина = 1
                      y[0] = BigInteger.ValueOf(x0);

                      // крок 2
                      int[] t = new int[1];
                      t[0] = size;
                      int s = 0;
                      for (int i=0; t[i]>=33; i++)
                      {
                            // розширення масив t
                            int[] tmp_t = new int[t.Length + 1];
///////////////
                                  Array.Copy(t,0,tmp_t,0,t.Length);          //
розширення
                            t = new int[tmp_t.Length];                       //
масив t
                            Array.Copy(tmp_t, 0, t, 0, tmp_t.Length);
///////////////
```

```
                        t[i+1] = t[i]/2;
                        s = i+1;
                  }

                  // крок3
                  BigInteger[] p = new BigInteger[s+1];
                  p[s] = new BigInteger("8000000B",16);
                  int m = s-1;  // крок4

                  for (int i=0; i<s; i++)
                  {
                        int rm = t[m]/32;  // крок5

                  step6: for(;;)
                           {
                                 // крок 6
                                 BigInteger[] tmp_y = new BigInteger[y.Length];
////////////////
                                       Array.Copy(y,0,tmp_y,0,y.Length);
//  розширення
                                 y = new BigInteger[rm+1];
//  масив y
                                 Array.Copy(tmp_y,0,y,0,tmp_y.Length);
////////////////

                                 for (int j=0; j<rm; j++)
                                 {
                                       y[j+1] =
(y[j].Multiply(constA32).Add(C)).Mod(BigInteger.Two.Pow(32));
                                 }

                                 // крок 7
                                 BigInteger Ym = BigInteger.Zero;
                                 for (int j=0; j<rm; j++)
                                 {
                                       Ym = Ym.Add(y[j].ShiftLeft(32*j));
                                 }

                                 y[0] = y[rm]; // крок 8

                                 // крок 9
                                 BigInteger N = BigInteger.One.ShiftLeft(t[m]-
1).Divide(p[m+1]).Add(
                                       Ym.ShiftLeft(t[m]-
1).Divide(p[m+1].ShiftLeft(32*rm)));

                                 if (N.TestBit(0))
                                 {
                                       N = N.Add(BigInteger.One);
                                 }

                                 // крок 10

                                    for(;;)
                                    {
                                          // крок 11
                                          BigInteger NByLastP =
N.Multiply(p[m+1]);

                                          if (NByLastP.BitLength > t[m])
                                          {
                                                goto step6; // крок 12
                                          }

                                          p[m] = NByLastP.Add(BigInteger.One);

                                          // крок13
                                          if (BigInteger.Two.ModPow(NByLastP,
p[m]).CompareTo(BigInteger.One) == 0
```

```
                                        && BigInteger.Two.ModPow(N,
p[m]).CompareTo(BigInteger.One) != 0)
                                    {
                                        break;
                                    }

                                    N = N.Add(BigInteger.Two);
                                }

                                if (--m < 0)
                                {
                                    pq[0] = p[0];
                                    pq[1] = p[1];
                                    return y[0].LongValue; //повернення для
процедури B' крок 2
                                }

                                break; // крок 14
                            }
                        }
                        return y[0].LongValue;
                    }

                    // Процедура B
                    private void procedure_B(int x0, int c, BigInteger[] pq)
                    {
                        // Перевірка і виконання умови 0<x<2^16; 0<c<2^16; c - odd.
                        while(x0<0 || x0>65536)
                        {
                            x0 = init_random.NextInt()/32768;
                        }

                        while((c<0 || c>65536) || (c/2==0))
                        {
                            c = init_random.NextInt()/32768 + 1;
                        }

                        BigInteger [] qp = new BigInteger[2];
                        BigInteger q = null, Q = null, p = null;
                        BigInteger C = BigInteger.ValueOf(c);
                        BigInteger constA16 = BigInteger.ValueOf(19381);

                        // крок1
                        x0 = procedure_A(x0, c, qp, 256);
                        q = qp[0];

                        // крок2
                        x0 = procedure_A(x0, c, qp, 512);
                        Q = qp[0];

                        BigInteger[] y = new BigInteger[65];
                        y[0] = BigInteger.ValueOf(x0);

                        const int tp = 1024;

                        BigInteger qQ = q.Multiply(Q);

step3:
                        for(;;)
                        {
                            // крок 3
                            for (int j=0; j<64; j++)
                            {
                                y[j+1] =
(y[j].Multiply(constA16).Add(C)).Mod(BigInteger.Two.Pow(16));
                            }

                            // крок 4
                            BigInteger Y = BigInteger.Zero;
```

```
                        for (int j=0; j<64; j++)
                        {
                              Y = Y.Add(y[j].ShiftLeft(16*j));
                        }

                        y[0] = y[64]; // крок 5

                        // крок 6
                        BigInteger N = BigInteger.One.ShiftLeft(tp-
1).Divide(qQ).Add(
                              Y.ShiftLeft(tp-1).Divide(qQ.ShiftLeft(1024)));

                        if (N.TestBit(0))
                        {
                              N = N.Add(BigInteger.One);
                        }

                        // крок 7

                        for(;;)
                        {
                              // крок 11
                              BigInteger qQN = qQ.Multiply(N);

                              if (qQN.BitLength > tp)
                              {
                                    goto step3; // крок 9
                              }

                              p = qQN.Add(BigInteger.One);

                              // крок 10
                              if (BigInteger.Two.ModPow(qQN,
p).CompareTo(BigInteger.One) == 0
                                    && BigInteger.Two.ModPow(q.Multiply(N),
p).CompareTo(BigInteger.One) != 0)
                              {
                                    pq[0] = p;
                                    pq[1] = q;
                                    return;
                              }

                              N = N.Add(BigInteger.Two);
                        }
                  }
            }

            //Процедура B'
            private void procedure_Bb(long x0, long c, BigInteger[] pq)
            {
                  //Перевірка і виконання умови: 0<x<2^32; 0<c<2^32; c - odd.
                  while(x0<0 || x0>4294967296L)
                  {
                        x0 = init_random.NextInt()*2;
                  }

                  while((c<0 || c>4294967296L) || (c/2==0))
                  {
                        c = init_random.NextInt()*2+1;
                  }

                  BigInteger [] qp = new BigInteger[2];
                  BigInteger q = null, Q = null, p = null;
                  BigInteger C = BigInteger.ValueOf(c);
                  BigInteger constA32 = BigInteger.ValueOf(97781173);

                  // крок 1
                  x0 = procedure_Aa(x0, c, qp, 256);
```

```
                q = qp[0];

                // крок 2
                x0 = procedure_Aa(x0, c, qp, 512);
                Q = qp[0];

                BigInteger[] y = new BigInteger[33];
                y[0] = BigInteger.ValueOf(x0);

                const int tp = 1024;

                BigInteger qQ = q.Multiply(Q);

step3:
                for(;;)
                {
                      // крок 3
                      for (int j=0; j<32; j++)
                      {
                            y[j+1] =
(y[j].Multiply(constA32).Add(C)).Mod(BigInteger.Two.Pow(32));
                      }

                      // крок 4
                      BigInteger Y = BigInteger.Zero;
                      for (int j=0; j<32; j++)
                      {
                            Y = Y.Add(y[j].ShiftLeft(32*j));
                      }

                      y[0] = y[32]; // крок 5

                      // крок 6
                      BigInteger N = BigInteger.One.ShiftLeft(tp-
1).Divide(qQ).Add(
                            Y.ShiftLeft(tp-1).Divide(qQ.ShiftLeft(1024)));

                      if (N.TestBit(0))
                      {
                            N = N.Add(BigInteger.One);
                      }

                      // крок 7

                      for(;;)
                      {
                            // крок 11
                            BigInteger qQN = qQ.Multiply(N);

                            if (qQN.BitLength > tp)
                            {
                                  goto step3; // крок 9
                            }

                            p = qQN.Add(BigInteger.One);

                            //крок 10
                            if (BigInteger.Two.ModPow(qQN,
p).CompareTo(BigInteger.One) == 0
                                  && BigInteger.Two.ModPow(q.Multiply(N),
p).CompareTo(BigInteger.One) != 0)
                            {
                                  pq[0] = p;
                                  pq[1] = q;
                                  return;
                            }

                            N = N.Add(BigInteger.Two);
                      }
```

```
                }
            }


            /**
            * Процедура C
            * Процедура генерує значення даного P, Q,
            * Повернення значення.         */
            private BigInteger procedure_C(BigInteger p, BigInteger q)
            {
                  BigInteger pSub1 = p.Subtract(BigInteger.One);
                  BigInteger pSub1Divq = pSub1.Divide(q);

                  for(;;)
                  {
                        BigInteger d = new BigInteger(p.BitLength, init_random);

                        // 1 < d < p-1
                        if (d.CompareTo(BigInteger.One) > 0 &&
d.CompareTo(pSub1) < 0)
                        {
                              BigInteger a = d.ModPow(pSub1Divq, p);

                              if (a.CompareTo(BigInteger.One) != 0)
                              {
                                    return a;
                              }
                        }
                  }
            }

            /**
            * генерується P, Q і значення даного параметра,
* Повернення Gost3410Parameters об'єкта.
public Gost3410Parameters GenerateParameters()
            {
                  BigInteger [] pq = new BigInteger[2];
                  BigInteger    q = null, p = null, a = null;

                  int  x0, c;
                  long  x0L, cL;

                  if (typeproc==1)
                  {
                        x0 = init_random.NextInt();
                        c  = init_random.NextInt();

                        switch(size)
                        {
                              case 512:
                                    procedure_A(x0, c, pq, 512);
                                    break;
                              case 1024:
                                    procedure_B(x0, c, pq);
                                    break;
                              default:
                                    throw new ArgumentException("Помилка! Розмір
ключа повинен бути 512 або 1024 бітів.");
                        }
                        p = pq[0];  q = pq[1];
                        a = procedure_C(p, q);

      //System.out.println("p:"+p.toString(16)+"\n"+"q:"+q.toString(16)+"\n"+"a:
"+a.toString(16));
                        //System.out.println("p:"+p+"\n"+"q:"+q+"\n"+"a:"+a);
                        return new Gost3410Parameters(p, q, a, new
Gost3410ValidationParameters(x0, c));
                  }
                  else
```

```
                {
                        x0L = init_random.NextLong();
                        cL  = init_random.NextLong();

                        switch(size)
                        {
                                case 512:
                                        procedure_Aa(x0L, cL, pq, 512);
                                        break;
                                case 1024:
                                        procedure_Bb(x0L, cL, pq);
                                        break;
                                default:
                                        throw new InvalidOperationException("Ooops!
key size 512 or 1024 bit.");
                        }
                        p = pq[0];  q = pq[1];
                        a = procedure_C(p, q);

      //System.out.println("p:"+p.toString(16)+"\n"+"q:"+q.toString(16)+"\n"+"a:
"+a.toString(16));
                        //System.out.println("p:"+p+"\n"+"q:"+q+"\n"+"a:"+a);
                        return new Gost3410Parameters(p, q, a, new
Gost3410ValidationParameters(x0L, cL));
                }
        }
      }
}
```