**ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ**
**НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ЗАХИСТУ ІНФОРМАЦІЇ**

**КАФЕДРА ІНФОРМАЦІЙНОЇ ТА КІБЕРНЕТИЧНОЇ БЕЗПЕКИ**

# Пояснювальна записка

до бакалаврської роботи
на тему:

# «ДОСЛІДЖЕННЯ ШЛЯХІВ ТА РОЗРОБКА РЕКОМЕНДАЦІЙ ЩОДО ПІДВИЩЕННЯ ЗАХИЩЕНОСТІ СИСТЕМ ЕЛЕКТРОННОГО ДОКУМЕНТООБІГУ З ВИКОРИСТАННЯМ ТЕХНОЛОГІЇ ЦИФРОВОГО ПІДПИСУ»

Виконав студент 4 курсу, групи БСД-43
спеціальності 125 Кібербезпека
освітньо-професійної програми «Інформаційна та кібернетична безпека»
(шифр і назва спеціальності)
Ігнатенко В.О
(прізвище та ініціали)
Керівник Кожухівський А.Д
(прізвище та ініціали)
Рецензент
(прізвище та ініціали)
Нормоконтролер Чумак Н.С.
(прізвище та ініціали)

КИЇВ – 2023

**ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ**

Інститут ННІЗІ
Кафедра Інформаційної та кібернетичної безпеки
Ступінь вищої освіти Бакалавр
Спеціальність 125 Кібербезпека
Освітньо-професійна програма Інформаційна та кібернетична безпека

ЗАТВЕРДЖУЮ
Завідувач кафедри ІКБ
_____ Гайдур Г.І.
“___”_____2023 року

# З А В Д А Н Н Я
## НА БАКАЛАВРСЬКУ РОБОТУ СТУДЕНТУ

Ігнатенку Владиславу Олександровичу
(прізвище, ім’я, по батькові)

1. Тема бакалаврської роботи: «Дослідження шляхів та розробка рекомендацій щодо підвищення захищеності систем електронного документообігу»

керівник бакалаврської роботи Кожухівський Андрій Дмитрович, д.т.н.професор
(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)
затверджені наказом закладу вищої освіти від « 24 » 02 2023 року № 26 .

2. Строк подання студентом бакалаврської роботи 29.05.2023 р.

3. Вихідні дані до бакалаврської роботи
системи електронного документообігу організацій;
інструменти реалізації електронного цифрового підпису;
підпис даних у .NET;
мова програмування C#.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
1. Аналіз аспектів безпеки у процесі електронного документообігу.
2. Дослідження можливості підвищення захищеності систем електронного документообігу із застосуванням ефективних схем цифрового підпису.
3. Розробка рекомендацій щодо підвищення захищеності систем електронного документообігу

5. Перелік графічного матеріалу

1. Тема бакалаврської роботи.
2. Об’єкт, предмет, мета та наукові завдання дослідження.
3. Характеристика та основні функції систем електронного документообігу.
4. Ефективні схеми цифрового підпису. Складові реалізації цифрового підпису та особливості роботи з додатками
5. Рекомендації щодо застосування методів та засобів цифрового підпису у контексті підвищення захищеності систем електронного документообігу.
6. Висновки та рекомендації за результатами роботи.

6. Дата видачі завдання 24.02.2023 р.

## КАЛЕНДАРНИЙ ПЛАН

| № зп | Назва етапів бакалаврської роботи | Строк виконання етапів бакалаврської роботи | Примітка |
|---|---|---|---|
| 1. | Уточнення постановки завдання. | 24.02.2023р. | |
| 2. | Аналіз науково-технічної літератури. | 04.03.2023р. | |
| 3. | Обґрунтування вибору рішення. | 10.04.2023р. | |
| 4. | Збір даних. | 15.04.2023р. | |
| 5. | Дослідження методів формування та перевірки електронного цифрового підпису | 3.05.2023р. | |
| 7. | Розробка рекомендацій щодо застосування схем цифрового підпису у контексті безпеки систем електронного документообігу | 15.05.2023р. | |
| 8. | Реферат, вступ, висновки. | 22.05.2023р. | |
| 9. | Підготовка презентації до захисту. | 28.05.2023р. | |

Студент Ігнатенко В.О.
(підпис) прізвище та ініціали

Керівник бакалаврської роботи Кожухівськи А.Д.
(підпис) прізвище та ініціали

# РЕФЕРАТ

Текстова частина бакалаврської роботи: 60 сторінок, 12 рисунків, 2 таблиць, 15 джерел.

*Об'єкт дослідження* - криптографічний захист систем електронного документообігу.

*Предмет дослідження* - застосування методів та засобів підвищення захищеності систем електронного документообігу при використанні електронного цифрового підпису.

*Мета роботи* - розробити рекомендації щодо застосування методів та засобів цифрового підпису у контексті підвищення захищеності систем електронного документообігу.

*Методи дослідження* - опрацювання літератури за даною темою, аналіз експлуатаційної документації, міжнародних стандартів та їх порівняння, проведення експерименту.

В роботі приведено основні відомості про безпекову складову криптографії, криптографічні системи, особливості застосування цифрового підпису та архітектури схем його перевірки та форматування.

Проаналізовано ефективні схеми цифрового підпису.

Досліджено особливості процесу формування та перевірки цифрового підпису щодо швидкості та ресурсоємності процесу за різними схемами реалізації.

Проведено аналіз ефективності розроблюваних заходів із застосуванням на пристроях з низькою обчислювальною потужністю.

На основі досліджень проведених в роботі розроблено рекомендації щодо застосування технологій цифрового підпису для підвищення захищеності систем електронного документообігу.

Галузь використання - кібербезпека інформаційних систем.

ЕЛЕКТРОННИЙ ДОКУМЕНТООБІГ, КІБЕРБЕЗПЕКА, ЗАХИСТ ДАНИХ, КРИПТОГРАФІЯ, СХЕМИ ЦИФРОВОГО ПІДПИСУ, АВТЕНТИФІКАЦІЯ, RSA, DSA, ECDSA

**ЗМІСТ** Стор.

# ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

PKI - Public key infrastructure

MAC - Message authentication code

CRL - Certificate Revocation List

PGP - Pretty Good Privacy

PEM - Privacy Enhanced Mail

DER - Distinguished Encoding Rules

OT - Owner Trust

ST - Signature Trust

KL - Key Legitimacy

PKCS - Public key cryptography standards

CLR - Common Language Runtime

# ВСТУП

*Актуальність дослідження*. Станом на теперішній час розроблено багато методів для перевірки змісту та підпису документів, які використовуються для підтвердження права власності або для спілкування між фізично віддаленими особами. Проблема полягає в тому, що навіть основний носій запису (папір) легко підробити, а підпис можна легко імітувати, навіть змінюючи зміст уже підписаного документа. З часом, а особливо з постійно зростаючою популярністю Інтернету як глобального засобу комунікації, ці проблеми почали з’являтися також і в електронній комунікації.

Проблеми, пов'язані з автентифікацією, верифікацією сторін, що спілкуються, і забезпеченням цілісності повідомлень, в основному зосереджені на перевірці цифрового підпису сторін, що спілкуються, і умові неспростовності, коли передане повідомлення створюється лише тими особами, що спілкуються. В даний час автентифікація сторін, що спілкуються, є темою, що часто обговорюється. До таких механізмів відноситься цифровий підпис, який за своєю суттю включає найсучасніші криптографічні методи, що поєднують симетричне і асиметричне шифрування та алгоритми кодування.

*Об’єкт дослідження* - криптографічний захист систем електронного документообігу.

*Предмет дослідження* - застосування методів та засобів підвищення захищеності систем електронного документообігу при використанні електронного цифрового підпису.

*Мета роботи* - розробити рекомендації щодо застосування методів та засобів цифрового підпису у контексті підвищення захищеності систем електронного документообігу.

*Завдання бакалаврської роботи:*

- провести аналіз криптографічних систем в контексті безпеки електронного документообігу;

- визначити місце цифрового підпису у процесі електронного документообігу;

- дослідити ефективні схеми цифрового підпису;

- дослідити роботу додатків реалізації та перевірки цифрових підписів;

- розробити рекомендації щодо застосування технологій цифрового підпису;

- скласти висновки за результатами дослідження.

Дослідницька частина роботи присвячена ефективним схемам цифрового підпису, порівнянням поточні схеми з формулюванням пропозицій щодо нових ефективних схем. Далі потрібно дослідити їх аспекти для пристроїв зі слабкими обчислювальними можливостями або розгортання цифрового підпису в системах з низькою швидкістю передачі даних.

У практичній частині розглядається реалізація деяких ефективних схем на обраній мові програмування та використання програми для порівняння та аналізу їхніх властивостей. У найпростіших схемах представлений сам алгоритм генерації підпису та перевірки. У третьому розділі розглядаються теоретичні пропозиції деяких ефективних схем підпису.

Завершальною частиною роботи є складання рекомендацій щодо використання технологій цифрового підпису у рамках підвищення рівня захисту електронного документообігу.

*Методи дослідження* – опрацювання літератури за даною темою, аналіз експлуатаційної документації, міжнародних стандартів та їх порівняння, проведення експерименту.

*Практичне значення одержаних результатів:* рекомендації щодо застосування технологій цифрового підпису можуть бути використані у сфері забезпечення підвищеного рівня захисту документообігу комунікаційних каналів як організацій, так і у межах застосування фізичними особами.

# 1. АНАЛІЗ АСПЕКТІВ БЕЗПЕКИ У ПРОЦЕСІ ЕЛЕКТРОННОГО ДОКУМЕНТООБІГУ

## 1.1 Характеристика систем електронного документообігу

Система електронного документообігу (СЕД) це програмне забезпечення, яке керує створенням, зберіганням та контролем документів в електронному вигляді. Основною функцією СЕД є керування електронною інформацією в робочому процесі організації, а базова СЕД повинна включати управління документами, робочий процес, пошук тексту та зображень.

Не всі СЕД мають можливість керування записами. Щоб кваліфікуватися як а система управління записами, СЕД має бути спроможною забезпечити безпечний доступу, збереження контексту, і виконання інструкцій щодо розміщення всіх записів в системі. Перед впровадженням системи потрібно визначити, як це вписується у поточні потреби стратегії управління.

Все більше функцій СЕД інтегровано в системи керування вмістом (КВ) [1]. Ці системи поєднують у собі додаткові функціональні можливості, такі як керування веб-сайтом за допомогою інструментів робочого процесу, стандартизація шаблонів та права доступу.

***Стандартизація СЕД***

Необхідно уважно вивчати, чи СЕД відповідає наступним характеристикам[2]:

- належний рівень безпеки для захисту конфіденційної інформації;

- належний доступ до відкритих державних архівів;

- можливість отримувати електронні записи та керувати ними і таким чином відповідати вимогам законодавства щодо надійності, повноти, доступності тощо.

***Вибір рішення СЕД***

Якщо прийнято рішення оптимізувати СЕД, вибір критеріїв має бути ретельним. Використання СЕД не є єдиним інструментом для реорганізації

стратегії управління електронними документами. Фактично, використання СЕД може призвести до фіксації проблем в управлінні організацією, особливо для державних установ з певними правовими вимогами. Рішення використання СЕД вимагає значного планування та аналізу.

СЕД кожного постачальника має різні ступені функціональності. У СЕД, призначеній для приватного застосування функції можуть не дозволяти відповідати певним юридичним вимогам як для державних установ [1].

Так, СЕД, розроблена для приватного сектора може бути не спроможною:

- керувати всіма необхідними форматами файлів, які містять державні записи:

- зберігати необхідні метадані запису;

- забезпечити надійність;

- забезпечити належну безпеку конфіденційності інформація та записів.

СЕД може покращити співпрацю під час розробки документа. Проте СЕД також може створювати кілька копій документа та може не забезпечити потрібну безпеку доступу до конфіденційних записів.

***Ключові поняття***

Коли йде мова переваги СЕД для визначеної організації, потрібно ознайомитися з наступними ключовими поняттями:

- інтеграція документообігу;

- основні функції;

- додаткові функції;

- керування вмістом;

- основний процес вибору програмного забезпечення.

***Інтеграція документообігу***

В ідеалі потрібно використовувати СЕД, яка допоможе інтегрувати та автоматизувати керування документами та управління документацією. Відповідна СЕД може спростити це інтегроване управління.

Важливим є аналіз документообігу конкретної організації. СЕД повинна підтримувати поточні потреби в робочому процесі і дозволяти фіксувати записи та керувати ними як частина повсякденних робочих операцій.

## 1.2 Основні функції систем електронного документообігу

Як мінімум, СЕД повинна забезпечувати:

1) *Контроль безпеки*. Ця функція контролює, які користувачі матимуть доступ до конкретної інформації. Будь-яка система, що передбачена до використання, повинні мати можливість захистити конфіденційність даних.

2) *Додавання, позначення та керування версіями*. СЕД повинна дозволяти користувачам додавати документи до системи та позначити документ як офіційний запис. Вона також має автоматично призначати правильне позначення версії.

3) *Збирання та використання метаданих*. СЕД повинна дозволяти збір та використання відповідних метаданих визначеної організації [3].

***Додаткові функції***

Корисною може бути реорганізація СЕД, яка забезпечує:

1) *Ведення записів*. Системи СЕД не завжди включає в себе можливість здійснювати управління функціями записів. Ті, що пропонують управління записами іноді називають системами управління електронними документами та записами (англ. EDRMS). На додаток до цих систем, автономне програмне забезпечення для керування записами, називається програмою керування записами (англ. RMA). Системи управління записами повинні мати можливість забезпечити безпечний доступ, підтримувати контекст запису в серії записів, а також автоматизувати виконання розпорядчих інструкцій для всіх записів у системі. RMA та EDRMS часто вимагають прийняття рішень окремими користувачами залежно від того, які документи кваліфікуються як записи, з додаванням рівня складності в робочий процес. Як наслідок, розроблюються програми відповідного навчання для всіх користувачів, що є досить важливим для

успішної реалізації процесу електронного документообігу. Так, за рахунок впливу на користувачів і наявності додаткових витрат, пов'язаних необхідністю підвищення базового рівня ефективності процесу управління, необхідною є постійна підтримка керівництва організації та наявність відповідних ресурсів.

***Керування вмістом***

Програмне забезпечення для керування вмістом (КВ) поєднує типову функціональність СЕД з додатковими функціями, такими як управління цифровими активами та керування веб-сайтами, де підтримується обмін різними цифровими носіями у межах всієї організації. Залежно від характеристики конкретної організації, поточних та майбутніх потреб, це програмне забезпечення може мінімізувати необхідність придбання додаткового програмного забезпечення для управління різними видами інформаційного вмісту системи.

Програмне забезпечення КВ узагальнено поділяють на дві категорії. По-перше, це веб-програмне забезпечення для керування вмістом, яке використовується для керування веб-сайтами, вміст яких швидко змінюється і співпраця відбувається між багатьма відділами [4]. До другого типу можна віднести такі рішення, як Enterprise Content Management (ECM) - програмне забезпечення, яке об’єднує різні функції, наприклад як робочий процес і цифрові зображення з цифровими активами управління, керування записами. Як з програмним забезпеченням СЕД, можливістю керування записами у програмному забезпеченні КВ часто є додатковою функцією.

Отже, записи керуються програмним забезпеченням КВ. Необхідно визначити і розпланувати основні етапи їх інтеграції у існуючу інформаційну систему організації. Основний процес вибору СЕД, КВ або програмного забезпечення RMA має бути базовим рівенем для розробки більш конкретного процесу визначеної організації.

Основний процес включає:

1) *Оцінка потреб*. Перший етап - робота з внутрішніми зацікавленими сторонами і визначення юридичних зобов'язань та унікальних потреб. Якщо є необхідність використовувати пакет відповідних програм, потрібно переконатись,

що виконуються вимоги надійності, повноти, доступності, правової допустимості, і відповідності поточних потреб.

2) *Вибір постачальника*. Потрібно здійснювати ретильний вибір постачальника СЕД і може знадобитися надсилання запиту для пропозиції, яка містить конкретні юридичні вимоги і критерії вибору постачальника. Можна зв'язатися з іншими організаціями, які вже використовують подібні системи та зібрати якомога більше інформації про потенційні СЕД або шляхи реорганізації існуючої системи.

3) *План реалізації*. Потрібно працювати з постачальниками та внутрішніми зацікавленими сторонами для складання комплексного плану впровадження.

План має включати:

- технологічний план реалізації, що окреслює як і коли буде встановлена система;

- план впровадження користувача, який включає навчання і розгортання системи.

4) *Розгортання*. Як зазначено у реалізації плану, потрібно встановити та протестувати систему і навчати відповідних користувачів.

5) *Управління*. У міру використання системи, присутнім має бути її постійне вдосконалення.

На кожному з наведених вище етапів необхідним є документування всього процесу, включаючи потреби оцінки, впровадження та управління. Також потрібно задокументувати процедури, що включають роботу з апаратним, програмним забезпеченням та оперативні процедури і заходи безпеки.

*Додаткові заходи щодо підвищення ефективності та захищеності системи електронного документообігу*

Формування команди, яка включає представників з вищого керівництва визначеної організації (наприклад, команда управління записами, юридичний відділ) для виконання наступних процедур:

1) Визначте поточних та майбутнє потреб організації та потреб потенційних зовнішніх зацікавлених сторін.

2) Вирішення, чи призначена СЕД лише для керування робочим процесом або потрібно використовувати її для керування записами з ширшим функціоналом.

3) Продумати, як автоматизація додає цінності поточному процесу, враховуючи те, чи може знадобитися зміна робочого процесу для впровадження СЕД. Визначити на яких етапах робочого процесу потрібно буде це виконати.

4) Визначте, які записи потрібно вносити та керувати ними за допомогою СЕД та чи буде необхідним створення резервних файлів записів.

5) Визначити, чи належним чином організовано записи і проіндексовано для полегшення пошуку. Тут потрібно переконатися, що записи заповнюються належним чином і виправляються усі помилки перед впровадженням системи.

6) Продумайте поводження з метаданими, включаючи типи метаданих, які є найбільш актуальними, оскільки потрібно врахувати те, хто ними керуватиме.

7) Розглянути правову базу організації, враховуючи вплив розроблюваних заходів на подальшу взаємодію з юридичними організаціями.

8) Продумати, як використовуються записи у поточному виконанні системи і як їх потрібно використовувати в майбутньому; визначити, які записи передбачено до обміну, а які мають лише зберігатися у системі.

9) Потрібно вирішити, як організація буде розпоряджатися записами в СЕД. Тут потрібно враховувати, чи дозволить система передавати, знищувати, конвертувати та/або переміщувати відповідні записи. Це не завжди може бути доступним з точки зору дотримання норм інформаційної безпеки.

## 1.3 Місце цифрового підпису у процесі електронного документообігу

### 1.3.1 Особливості застосування цифрового підпису

В асиметричних криптосистемах дуже важливим є забезпечення автентичності (походження) отриманих даних. Простий спосіб автентифікації даних - зашифрувати їх за допомогою закритого ключа особи, яка поручається за

автентичність даних. Після цього будь-хто може розшифрувати дані за допомогою відкритого ключа підписувача [5].

Існує дві основні проблеми з описаним вище методом. По-перше, дані автентифікації часто дуже об’ємні, а їх шифрування та дешифрування асиметричними криптосистемами може бути повільним. Друга проблема полягає в перевірці автентичності відкритого ключа дешифрування. Перша проблема вирішується за допомогою хешів, а друга - за допомогою інфраструктури відкритих ключів (*PKI*).

*Хеш-функції*

Проблема повільного шифрування та дешифрування вирішується обчисленням довжини хешу, зазвичай 128, 160, 192, 256 або 512 біт для всього *Z*-адміністрування. При цьому необхідно, щоб було практично неможливо знайти дві різні адміністрації з однаковим хешем. Замість всього об’єму даних, шифрується лише їх хеш. Крім того, виникає питання про те, чи були надіслані дані змінені під час передачі. Це вирішується шляхом додавання коду автентифікації керування, який використовується одержувачем для перевірки. Коди автентифікації повідомлень формуються шляхом хешування (наприклад, *MD*5, *SHA-1*) або за допомогою функції автентифікації (наприклад, *MAC* - (англ. *Message authentication code*)).

*Цифровий підпис*

Цифровий підпис дозволяє одержувачу повідомлення перевірити, що повідомлення надіслано зазначеним відправником і що повідомлення не було змінено. Він базується на принципі класичного ручного підпису та накладається за допомогою публічної криптосистеми. Відправник адміністрування (тобто особа, яка підписує) має приватний ключ, який прив’язаний до відкритого ключа односторонньою функцією. Завдяки односторонній функції будь-хто може перевірити, що підписувач має закритий ключ. Той факт, що електронний відкритий ключ належить вказаній особі, а не якомусь зловмиснику, вирішується за допомогою сертифікатів в інфраструктурі відкритих ключів (*PKI*), яка також забезпечує неспростовність достовірності автентифікованої особи.

*Сертифікат*

Сертифікат - це фактично певна заява (наприклад, певна особа має певний відкритий ключ), підписана електронною мовою довіреною стороною. Довіреною стороною може бути певна установа, яка має загальну довіру. Ця установа називається центром сертифікації (англ. *CA*), а його відкритий ключ $V_{CA}$ безпечно доставляється всім користувачам. За допомогою цього ключа користувачі можуть потім перевірити всі сертифікати *CA*.

*Інфраструктура відкритих ключів*

Для практичної реалізації описаного вище принципу будується інфраструктура відкритих ключів - *PKI*. Її основним елементом є центр сертифікації - *CA*. Йому можуть підпорядковуватися інші *CA* та так звані органи реєстрації (англ. *RA*). Роль *RA* полягає у фізичній перевірці даних заявника сертифіката. *PKI* також містить список відкликаних сертифікатів - *CRL* (англ. *Certificate Revocation List*). Цей список підтримується центром сертифікації та записує сертифікати, визнані недійсними до належного закінчення терміну дії (наприклад, через компрометацію секретного ключа користувача). У цьому списку користувачі можуть перевірити сертифікати, за допомогою яких інша сторона, з якою вони хочуть спілкуватися, підтверджує свою особу [5].

### 1.3.2 Архітектура схем перевірки цифрового підпису

Перевірені схеми цифрового підпису використовують надійну третю сторону в спілкуванні для перевірки цілісності адміністрування та автентифікації підписувача. Відповідно до використовуваної топології, системи центрів сертифікації для видачі сертифікатів можна поділити на централізовані та децентралізовані. Прикладом централізованої системи є стандарт *X.509*, який використовує ієрархічну систему. У свою чергу, *PGP* (англ. *Pretty Good Privacy*) використовує децентралізовану модель підтвердження відкритого ключа [6].

*X.509*

Криптографії *X.509* є стандартом для систем на основі відкритого ключа (*PKI*) для простого підпису. Вона визначає, серед іншого, формат сертифікатів, списки відкликаних сертифікатів - *CRL*, параметри сертифікатів і методи перевірки дійсності сертифікатів. Перший стандарт *X.509* (версія 1) був виданий у 1988 році, а потім 2-га та 3-я версії, які, серед іншого, розширюють формат сертифіката додатковими необов'язковими елементами (унікальними ідентифікаторами видавця та власника).

*X.509* спочатку передбачав сувору ієрархічну систему центрів сертифікації - *CA* для видачі сертифікатів. Версія 3 вже включає підтримку додаткових топологій *CA*, але використання альтернативної топології (окрім ієрархічної) не є поширеною практикою.

Індивідуальні сертифікати видаються центрами сертифікації, які підтверджують належність даного ключа зазначеному власнику. Організації можуть видавати власні довірені кореневі сертифікати, які діють на підприємстві. Деякі веб-браузери мають попередньо встановлені кореневі сертифікати від відомих постачальників сертифікатів - *SSL*. У результаті власники браузерів визначають, яким *CA* довіряє більшість користувачів.

У стандарті *X.509* використовується такий формат сертифіката:

- версія: визначає формат сертифіката;

- серійний номер: унікальний ідентифікаційний номер, призначений *CA*;

- ідентифікатор алгоритму: алгоритм разом з іншими даними, які використовуються для створення підпису сертифіката;

- *CA*: ідентифікатор *CA*, який створив і підписав сертифікат;

- термін дії: від коли до коли сертифікат дійсний;

- тема: користувач, якому належить сертифікований відкритий ключ;

- відкритий ключ: підписаний відкритий ключ разом з ідентифікаторами алгоритму;

- *CA* - підпис: є функцією всіх записів сертифіката.

Сертифікати можуть фізично зберігатися в кількох форматах з різними розширеннями. Найчастіше використовуються розширення *PEM* (англ. *Privacy Enhanced Mail*) і *DER* (англ. *Distinguished Encoding Rules*). *DER* використовує двійковий формат як для ключів, так і для сертифікатів. Шляхом перекодування в *Base64* і обгортання заголовками створюється формат *PEM*, яка має формат *ASCII* і походить від стандарту захисту електронної пошти.

*X.509* запроваджується виділеними, загалом довіреними центрами сертифікації, які відповідають за видачу та керування сертифікатами. Ось чому йому потрібна ієрархічна структура з чітко визначеними зв'язками, які рідко змінюються. У *X.509* існує лише один ідентифікатор користувача порівняний з *PGP*, де їх може бути більше. Сертифікати та механізми в цьому стандарті можна використовувати для делегування прав. Користувач може створити сертифікат, у якому він передає частину повноважень своєму представнику. Після стандартної автентифікації представник надає цей сертифікат, і послуга стає доступною для нього.

*PGP*

Це комп'ютерна програма, яка в основному використовується для автентифікації та шифрування електронної пошти. Крім того, вона дозволяє шифрувати файли, створювати та керувати секретними та відкритими ключами, сертифікувати ключі та цифровий підпис адміністраторів. Першу версію представив Філ Циммерман у 1991 році.

*PGP* був настільки впливовим, що його стандартизували, щоб забезпечити взаємодію між різними версіями *PGP* та подібним програмним забезпеченням. *PGP* був прийнятий як стандарт Інтернету під назвою *OpenPGP*. Багато поштових клієнтів підтримують *OpenPGP*, що дозволяє працювати з цифровими підписами та зашифрованими повідомленнями. Він має децентралізовану модель перевірки відкритих ключів, у якій немає загального довіреного *CA*. Користувач *PGP* може діяти як *CA* і підтверджувати ключі інших користувачів, таким чином кожен користувач може вибрати, яким користувачам довіряти та з яким ступенем довіри

це здійснювати. Це створює «мережу довіри». У *PGP* розподіл ключів здійснюється у формі сертифікатів *PGP*.

Формат сертифіката *PGP* такий [6]:

- версія *PGP*;

- сертифікований ключ: відкритий ключ;

- інформація про власника ключа: ідентифікаційні дані (П.І.Б, *e-mail*, фото);

- підпис власника ключа: підпис, створений закритим ключем, що відповідає сертифікованому ключу;

- бажані симетричні алгоритми;

- термін дії;

- підпис: можна додати більше підписів, кожна ідентифікаційна інформація підписується окремо.

Користувач керує пакетом відкритих ключів, що містить ключі користувачів, з якими він спілкується. Ключі додаються та видаляються з комплекту динамічно за потреби.

У комплекті ключів до кожного відкритого ключа додається три записи, які використовуються для визначення дійсності ключів:

- довіра власника - *OT* (англ. *Owner Trust*): можливими є такі значення, як «недовірений», «частково довірений», «довірений і невідомий»;

- довіра підпису - *ST* (англ. *Signature Trust*): елемент, що призначається окремо для кожного підпису на сертифікаті, де значення запису - це копія значення *OT* із відкритого ключа підписувача;

- легітимність ключа - *KL* (англ. *Key Legitimacy*): ступінь, до якого *PGP* вважає, що цей ключ насправді належить користувачеві, зазначеному в сертифікаті.

У *PG*P довіра базується на самих користувачах та їхніх рішеннях, кому довіряти, які ключі підписувати тощо. Ця модель не підходить для масового розгортання в географічно чи політично віддалених групах користувачів. Двоє користувачів, які ніколи не зустрічалися та не мають спільних ресурсів, мають дуже складне завдання обміну відкритими ключами. *PGP* дозволяє

використовувати кілька ідентифікаторів і підписів, тоді як підпис можна прикріпити лише до одного ідентифікатора. У порівнянні з *X.509*, він не підходить для делегування прав, оскільки довіра до дійсності ключа не означає автоматично довіри до підпису, створеного цим ключем.

### 1.3.3 Форматування цифрових підписів

Вихідні дані схем цифрового підпису перетворюються в практично придатну форму за допомогою процесів цифрового підпису, які визначені у форматах підпису. Наразі використовується форматування *ISO/IEC 9796* і *PKCS* (англ. *Public key cryptography standards*).

*Форматування ISO/IEC 9796*

Цей формат був опублікований у 1991 році як перший міжнародний стандарт для цифрових підписів. Його характерними особливостями є те, що він заснований на криптосистемі з відкритим ключем, але алгоритм підпису прямо не вказано в стандарті. Використовуваний алгоритм має бути детермінованим. Цей стандарт не використовує функцію хешування, забезпечує відновлення керування та використовується для підписання керування обмеженою довжиною [7].

*Форматування PKCS*

Це набір специфікацій, який включає методи шифрування та підписання за допомогою *RSA*. Він відноситься до групи стандартів для криптографії з відкритим ключем, запропонованих і опублікованих *RSA Security*.

*PKCS #1* визначає математичні властивості та формат відкритих і закритих ключів *RSA*, а також основні алгоритми для шифрування, дешифрування, цифрових підписів і їх перевірки за допомогою *RSA*.

Стандарт *PKCS #3* описує протокол Діффі-Хеллмана для обміну секретними ключами між двома користувачами через публічний канал зв’язку. Стандарт *PKCS #7* використовується для підписання та шифрування повідомлень за допомогою *PKI* та сертифікатів. З точки зору цифрових підписів дуже

перспективним є *PKCS* #13, що описує цифровий підпис за допомогою еліптичних кривих [7].

# 2. ДОСЛІДЖЕННЯ МОЖЛИВОСТІ ПІДВИЩЕННЯ ЗАХИЩЕНОСТІ СИСТЕМИ ЕЛЕКТРОННОГО ДОКУМЕНТООБІГУ ІЗ ЗАСТОСУВАННЯМ ЕФЕКТИВНИХ СХЕМ ЦИФРОВОГО ПІДПИСУ

У даному розділі проведено аналіз можливостей використання різних схем цифрових підписів у контексті підвищення безпеки електронного документообігу. Так, перспективними є схеми без сертифікатів, схеми з коротким підписом, що використовують білінійне зіставлення, гіпереліптичні криві та схеми, стійкі до найнебезпечніших атак (наприклад, на основі *RSA*).

## 2.1 Ефективні схеми цифрового підпису

### 2.1.1 Схеми цифрового підпису без сертифікатів

Асиметричні криптосистеми без цифрових сертифікатів є досить ефективними, наприклад, в бездротовому зв'язку, де пропускна здатність передачі та обчислювальна потужність пристрою бездротового зв'язку обмежені. Щоб усунути цифровий сертифікат, згідно з [8], можна використовувати концепцію криптосистеми, заснованої на ідентифікації користувача (криптосистема на основі ідентифікатора). Найбільша перевага криптосистем на основі *ID* полягає в тому, що вони використовують не випадкове ціле число як відкритий ключ, як у класичних асиметричних криптосистемах, а хешовану форму ідентифікації самого користувача, якою може бути його ім’я або адреса електронної пошти. Проблема всіх схем підпису на основі ідентифікації (англ. *IBS*) полягає в тому, що генератор закритих ключів (англ. *PKG*) розпізнає закритий ключ кожного користувача. Це вказує на те, що власник *PKG* може підписувати будь-яке адміністрування на

стороні користувача, що загрожує невідмовності цифрових підписів. Щоб вирішити проблему збереження закритого ключа *IBS* із збереженням його переваг, був запроваджений цифровий підпис без сертифіката (англ. *CLS*).

*Цифровий підпис на основі DL без сертифіката*

У роботі [9] автори запропонували відносно простий у реалізації загальний метод створення схем *CLS* із схем цифрового підпису типу *ElGamal*. Метою цього методу є створення цифрових підписів без сертифіката, де під час перевірки підпису користувачеві потрібно знати лише особу свого партнера по спілкуванню та відкритий ключ *PKG*. Ця властивість дуже корисна там, де попереднє розповсюдження автентифікованих ключів неможливо. У їхній схемі лише користувач знає свій закритий ключ, і він не зберігається в *PKG*. Цей метод можна застосовувати в усіх схемах підпису на основі задачі дискретного логарифмування для перетворення схеми цифрового підпису на схему *CLS*. Створена схема *CLS* стійка до адаптивної атаки обраного повідомлення [8], що є максимально можливою безпекою для цифрових підписів, а генерація підпису та його перевірка є ефективними.

У загальній схемі підпису на основі ідентифікації користувача, *PKG* генерує закриті ключі для користувачів, використовуючи свій закритий ключ. З цього випливає, що *PKG* може підписати будь-яке адміністрування на стороні користувача. Цей факт підриває невідмовність цифрових підписів.

У схемі, описаній у статті [9], ця проблема вирішується тим, що *PKG* обчислює лише частковий закритий ключ для користувача. Потім користувач використовує цей частковий ключ зі своєю секретною інформацією для створення свого закритого ключа. Таким чином досягається те, що закритий ключ користувача недоступний для *PKG*. Користувач також може створити свій відкритий ключ подібним чином. Користувачі можуть опублікувати свої відкриті ключі в створеному підписі або завантажити їх у загальну папку. Для отримання відкритого ключа не потрібна автентифікація, тому сертифікат не використовується.

Принцип цієї схеми підпису є простим у реалізації та застосовний до всіх схем типу *ElGamal*.

Схема складається з чотирьох алгоритмів:

1) генерація ключа *PKG*;

2) генерація ключів користувача;

3) створення цифрового підпису;

4) перевірка отриманого цифрового підпису.

*Короткий цифровий підпис з білінійним сполученням без сертифіката*

Відповідно до [10], схема підпису без сертифіката може базуватись на складності вирішення проблем *k-CAA* (*k-Collusion Attack Algorithm*) та *Inv-CDHP* (*Inverse Computational Diffie-Hellman Problem*). Ця схема містить усі бажані властивості раніше запропонованих схем *CLS* і вимагає загальної функції хешування на відміну від інших схем *CLS*, які використовують менш ефективну функцію *MapToPoint* (використовується для відображення ідентифікаційної інформації в точку на еліптичній кривій). Підписання відбувається так само швидко, як і схеми типу *ElGamal*, а перевірка підпису вимагає одного скалярного множення та одного білінійного зіставлення, що означає, що ця схема значно ефективніша за інші схеми *CLS*. Сформований підпис має 154 біта, тому ця схема підходить для широкого використання, особливо у вузькосмугових засобах зв'язку.

Білінійне поєднання - це математична функція, яка однозначно кодує два числа в одне за допомогою певного алгоритму. Використання цієї функції виявляється досить ефективним у схемах цифрового підпису.

### 2.1.2 Схеми бездротових сенсорних мереж

Для забезпечення безпеки бездротових сенсорних мереж важливі шифрування та автентифікація повідомлень, що надсилаються між сенсорними вузлами. Ключі шифрування та методи автентифікації повинні бути узгоджені через вузли зв’язку. Через обмежені ресурси досягнення такої ключової згоди в

бездротових сенсорних мережах не є тривіальним. Схеми, що використовуються в даний час у звичайних мережах встановлення ключів, такі як схеми на основі протоколу Діффі-Хеллмана та асиметричних криптосистем, непридатні для бездротових сенсорних мереж. Через велику потребу в пам’яті у великих мережах із великою кількістю датчиків, попередній розподіл ключів може бути вигідним методом.

У роботі [11] запропоновано схему попереднього розподілу для сенсорних мереж під назвою *A Pairwise Key Pre-distribution Scheme for Wireless Sensor Networks*. Так, 64-бітного ключа достатньо для безпечного зв’язку між вузлами в цій схемі, а підпис у 2621 разів ефективніший з енергетичної точки зору, ніж у схемі *RSA*. Для порівняння, обчислювальна продуктивність цієї схеми подібна до 3200-бітного шифрування за допомогою *AES*. Загальновідомо, що симетрична криптосистема *AES* дуже енергоефективна.

### 2.1.3 Цифрові підписи з гіпереліптичними кривими

У комп'ютерній криптографії дуже ефективними є системи з гіпереліптичними кривими, створені шляхом розширення еліптичних кривих. Навпаки, вони використовують точкові групи, і їх надійність залежить від проблеми дискретного логарифмування в гіпереліптичних кривих.

Можливим є перенесення криптографічної системи з гіпереліптичними кривими на алгоритм *DSA* і створення команди цифрового підпису на основі системи *HEC-DSA* [12]. Цей цифровий підпис може вирішити проблему визначення повноти файлу та ідентифікатора підпису, що особливо підходить для операцій в Інтернеті, де потрібно перевірити особу.

### 2.1.4 Цифровий підпис на основі *RSA*

Ефективною може бути схема підпису, яка базується на *RSA* [13]. Схема стійка до атак з адаптивним вибраним повідомленням. Цей тип безпеки є найвищим із можливих для схеми цифрового підпису, що означає, що схему з таким рівнем безпеки можна безпечно використовувати в усіх програмах. Іншою важливою особливістю є те, що ця схема не має стану, на відміну від інших перевірено безпечних схем, і її ефективність також вища, ніж ці схеми.

### 2.1.5 Порівняння поточної та ефективної схем підпису

Порівняння поточних і ефективних схем підпису розділено на три групи відповідно до використання схем цифрового підпису: системи, орієнтовані на продуктивність, системи з обмеженою продуктивністю та пам'яттю, схеми цифрового підпису для веб-додатків.

*Системи, орієнтовані на продуктивність*

Системи, у яких продуктивність використовуваної схеми цифрового підпису має першочергове значення, а системна пам’ять не обмежена, можна розділити відповідно до того, чи вимагають вони швидкого підпису чи швидкої перевірки підпису.

На сьогоднішній день перевірка підпису відбувається набагато частіше, ніж генерація підпису. Таким чином, вигідно мати схеми, в яких верифікація не вимагає обчислень. В даний час найбільш поширеною такою схемою є *RSA*. Вже було можливо факторизувати (з великими витратами) *RSA* за допомогою 768-бітного ключа, тому рекомендується використовувати щонайменше 2048-бітні ключі, щоб забезпечити безпеку підписаних адміністраторів у майбутньому.

В даний час найбільш поширеною схемою для швидкого підписання є схема *DSA*, але тільки її варіант *ECDSA*, який використовує набагато коротші ключі, зберігаючи при цьому таку ж безпеку, як і інші схеми цього типу, має перспективу в майбутньому. У результаті прискорюється весь процес генерації підпису, а також перевірка, яка в цих схемах потребує більше обчислень.

Схеми, в яких підписування швидке, але перевірка складна, є перевагою в тих системах, де перевірка підписів покладена на зовнішнє джерело. Прикладом можуть бути датчики, де підписи перевіряються на обчислювально потужному сервері, який збирає підписані дані з датчиків.

Схема цифрового підпису *Feige-Fiat-Shamir* також є перспективною з точки зору того, що ні створення підпису, ні його перевірка не є надто складними. Переважною схемою в цих системах є та, у якій потрібне швидке підписання, а також швидка перевірка підпису. Недоліком цієї схеми є те, що вона використовує дуже довгі ключі порівняно з іншими варіантами.

Що стосується майбутнього *RSA* і *DSA*, то велика проблема цих схем полягає в тому, що зі збільшенням довжини ключів потреба в обчисленнях зростає набагато швидше, ніж їх безпека. Продуктивність цих великих ключових схем дуже низька. За допомогою *RSA* час, необхідний для підписання, збільшується разом із довжиною ключа в третьому ступені. Обчислення підпису за допомогою *RSA* з 2048- бітним ключем займає приблизно у 8 разів більше часу, ніж з 1024-бітним ключем. *DSA* поводиться подібно до *RSA* з довгими ключами.

Нові теоретичні пропозиції показують, що за допомогою еліптичних кривих і білінійного зіставлення можна зменшити ключ шифрування та дешифрування, тим самим підвищивши продуктивність усієї схеми підпису. Таким чином, у майбутньому ми можемо очікувати дедалі більшого розширення схем підпису з використанням цих методів.

*Системи з обмеженою продуктивністю та пам'яттю*

У цих системах дуже важливо, щоб схема підпису, яка використовується, була більш ефективною в продуктивності та створювала короткі підписи. На практиці такими системами можуть бути смарт-карти, мобільні пристрої, датчики тощо. В даний час найчастіше використовуються *RSA* і *DSA*. *RSA* має перевагу швидкої перевірки підпису, але створює довгі підписи, що є великим недоліком у системах з обмеженою пам’яттю з високою швидкістю необхідності збільшення ключів у майбутньому. *DSA* створює набагато коротші підписи, ніж *RSA*, але, з іншого боку, перевірка підпису складніша за цією схемою, яка часто є бажаною в

цих системах. Проблеми з безпекою та продуктивністю надалі такі ж, як описано раніше для систем, орієнтованих на продуктивність. Перевага застосування в цих системах одноразової схеми підпису *Merkle* порівняно з типами *DSA* і *RSA* полягає в тому, що схема стійка до атак з боку квантових комп'ютерів. Недоліком є використання довших ключів і створення довших підписів.

У мережах датчиків для розповсюдження ключів поточні схеми підпису *RSA* та *DSA* не застосовуються через обмежені ресурси датчиків. В даний час запропоновано методи вирішення цієї проблеми, одним з них є схема, яка використовує дуже короткі ключі, а її продуктивність порівнянна з симетричною криптосистемою *AES*.

Очікується, що в майбутньому ці системи використовуватимуть схеми короткого підпису з еліптичними кривими та білінійним узгодженням без сертифікатів. Ці схеми дуже ефективні з точки зору обчислень і генерують короткі підписи.

*Схеми цифрового підпису для веб-додатків*

Найбільш широко використовуваною схемою підпису в цих областях є *RSA*. Технологія використовується у всіх відомих протоколах, що забезпечують безпечне спілкування в Інтернеті, таких як *SSL*, *S/MIME* та *S/WAN*. Вона досі використовується в операційних системах усіх відомих виробників. У цих системах підпис набагато частіше перевіряється, ніж генерується, тому вимагає схеми підпису з ефективною перевіркою. Наразі *RSA* вважається безпечним лише з постійно зростаючим ключем, і, отже, сам підпис також збільшується в розмірі. У веб-спілкуванні, щоб мережа була менш завантажена під час надсилання цифрових підписів, вигідніше мати коротші цифрові підписи. Навіть у цих системах виникне вже описана проблема щодо безпеки та продуктивності *RSA* з довшими ключами.

У майбутньому схема цифрового підпису з використанням *HEC-DSA* може вирішити проблему виявлення повноти ідентифікатора файлу та підпису в Інтернет-операціях, де потрібно перевірити ідентичність. Можна очікувати заміни

*RSA* більш ефективними схемами, які використовують еліптичні криві або білінійне узгодження.

## 2.2 Складові реалізації цифрового підпису

Для реалізації практичної частини було обрано мову програмування *C#*, зважаючи на її перспективність. Реалізація має дві програми, сервер і клієнт, які призначені для демонстрації властивостей вибраних схем підпису та порівняння їх властивостей і ефективності.

### 2.2.1 *.NET Framework* і мова *C#*

Мова програмування *C#* (англ. *si-sharp*) є частиною технології *.NET Framework*, розробленої *Microsoft* у 2002 році [14]. Ця технологія заснована на принципі *CLR* (*Common Language Runtime*), тому підтримує взаємну сумісність між окремими мовами програмування. *CLR* - це віртуальна машина, яка виконує весь код, скомпільований в універсальну мову - байт-код (проміжна мова *Microsoft-MSIL*), який схожий на *assembler*. Байт-код - це машинний код віртуальної машини, тому неважливо, якою мовою буде програмуватися даний додаток, усі мови повністю еквівалентні. *NET Framework* дозволяє записати універсально необхідні функції у зовнішній файл на обраній мові програмування та скомпілювати ці файли в бібліотеки *DLL*, які потім можна використовувати для різних додатків.

*C#* - об'єктно-орієнтована мова програмування, за основу якої *Microsoft* взяла *C++* і *Java*. *C#* було розроблено з наміром збалансувати потужність *C++* і поєднати його з можливостями швидкої розробки додатків, які пропонують такі мови, як *Visual Basic, Delphi*. Вона була створена як проста, сучасна об’єктно-орієнтована мова для загального використання, та її реалізації повинні забезпечувати підтримку наступних принципів розробки програмного забезпечення, таких як сильна перевірка типів, перевірка меж полів, виявлення спроб використання неініціалізованих змінних і автоматичне керування пам’яттю. Надійність, довговічність і продуктивність також є важливими характеристиками.

В якості середовища розробки використовувалася версія *Visual C# Express*, яка є у вільному доступі. Продукт необмежений у використанні, і його можна безкоштовно завантажити з веб-сайту *Microsoft*.

Щоб реалізувати програму, що працює з криптографічними методами, використано *.NET Framework*, що працює з класами в бібліотеці *SystemSecurity Cryptography* для таких областей: симетричне шифрування/дешифрування; асиметричне шифрування/дешифрування; хешування; цифровий підпис; підпис *XML*.

### 2.2.2 Функції хешування в *.NET*

Подібно до алгоритмів шифрування, алгоритми хешування реалізовані в *.NET Framework*. Основний клас *HashAlgorith*m є абстрактним і містить методи для перевантаження та обчислення хешу для вбудованого керування у форматі масиву байтів або потоку.

Обчислення хеш-значення для вказаного масиву байтів:

```
byte [] ComputeHash (byte [] value);
```

Обчислення хеш-значення для вказаного потоку:

```
byte [] ComputeHash (Stream str);
```

*.NET* надає наступні алгоритми хешування:

*MD5 - System.Security.Cryptography.MD5*

*SHA-1 - System.Security.Cryptography.SHA1*

*SHA-256 - System.Security.Cryptography.SHA256*

*SHA-384 - System.Security.Cryptography.SHA384*

*SHA-512 - System.Security.Cryptography.SHA512*

### 2.2.3 Підпис даних у *.NET*

*Microsoft .NET* підтримує такі три класи для підпису даних:

- *RSA - System.Security.Cryptography.RSACryptoServiceProvider*

- *DSA - System.Security.Cryptography.DSACryptoServiceProvider*

- *ECDSA - System.Security.Cryptography.ECDsaCng*

*Використання цифрового підпису RSACryptoServiceProvider*

*RSACryptoServiceProvider* надає два набори методів для цифрового підпису, один для підпису адміністративного хеш, а другий - для підпису відкритого тексту.

Наступна пара методів використовується для підпису керуючого хешу:

```
public byte [] SignHash(byte [] rgbHash, string str);
public bool VerifyHash (byte [] rgbHash, string str, byte [] rgbSignature);
```

Наступні кілька методів використовуються для підпису та перевірки підпису чистих даних:

```
public byte [] SignData (byte [] buffer, object halg);
public bool VerifyData (byte [] buffer, object halg, byte [] signature);
```

*RSACryptoServiceProvider* може працювати з алгоритмами хешування *SHA-1* і *MD5*, і підтримує ключі довжиною від 384 до 16384 біт.

*Цифровий підпис за допомогою DSACryptoServiceProvider*

*DSACryptoServiceProvider* містить той самий набір попередніх методів, що й *RSACryptoServiceProvider*, але використовує лише *SHA-1* і підтримує ключі від 512 до 1024 біт.

*Цифровийпідпис за допомогою ECDsaCng*

Цей клас також містить той самий набір методів, що й два вже описані. На відміну від попередніх двох класів, *ECDsaCng* (*Elliptic Curve Digital Signature Algorithm Cryptography Next Generation*) не має обмежень на використання алгоритмів хешування та може підписувати дані 256- або 384-бітним ключем. З наведених вище фактів у всіх трьох класах підтримується лише 160-бітний *SHA-1*.

### 2.2.4 Робота з великими числами в *.NET*

Сама *.NET Framework* дозволяє працювати з великими числами за допомогою класу *System.Numerics.BigInteger*, але цей клас містить лише базові операції над цими числами. Для роботи з великими числами в Інтернеті можна знайти кілька класів, які можна безкоштовно завантажити, які дозволяють визначати такі великі числа. Деякі з цих класів мають перевагу перед вбудованим класом *BigInteger* у тому, що, окрім методів роботи з цими числами, вони також містять деякі функції, які часто використовуються в криптографії, які можуть бути корисними при реалізації схем цифрового підпису.

Клас *BigInt* дозволяє створювати велике число, наприклад, з рядкового типу даних або масиву байтів:

```
public BigInt (string s);
public BigInt (byte [] value);
```

Крім звичайних математичних функцій, клас містить метод обчислення модуля і піднесення числа типу *BigInt* до степеня з його подальшою модуляцією:

```
public BigInt Mod(BigInt mod);
public BigInt PowerMod(BigInt exponent, BigInt mod);
```

Серед часто використовуваних функцій криптографії цей клас дозволяє, серед іншого, знаходити найбільший спільний дільник і перевіряти прості числа за допомогою алгоритму Рабіна-Міллера, який також використовувався в реалізації:

```
        public static BigInt Gcd(BigInt x, BigInt y); //greatest common
divisor
        public bool IsPerfectSquare(); //square number test
        public    bool    IsProbablyPrimeRabinMiller(int    repetitions);
//prime number test
```

## 2.3 Особливості роботи з додатками

### 2.3.1 Реалізація серверного додатку

Рішення для цифрового підпису даних показано в програмі мережі клієнт-сервер. Сервер забезпечує клієнтам завантаження різних типів документів, підписаних сервером. Клієнт може перевірити, чи вміст завантаженого документа не було змінено під час передачі по мережі.

Після натискання кнопки «*Start*» сервер починає прослуховувати та очікувати з'єднання клієнта на порту, заданому в полі «*Port number*» (див. рис. 2.1). Сервер запускає новий потік для кожного нового підключеного клієнта, таким чином дозволяючи кільком клієнтам підключатися до сервера одночасно час. Ця кнопка також автоматично генерує ключі для всіх підтримуваних схем цифрового підпису. Кнопка «*Stop*» використовується для зупинки сервера, який можна перезапустити кнопкою «*Start*».

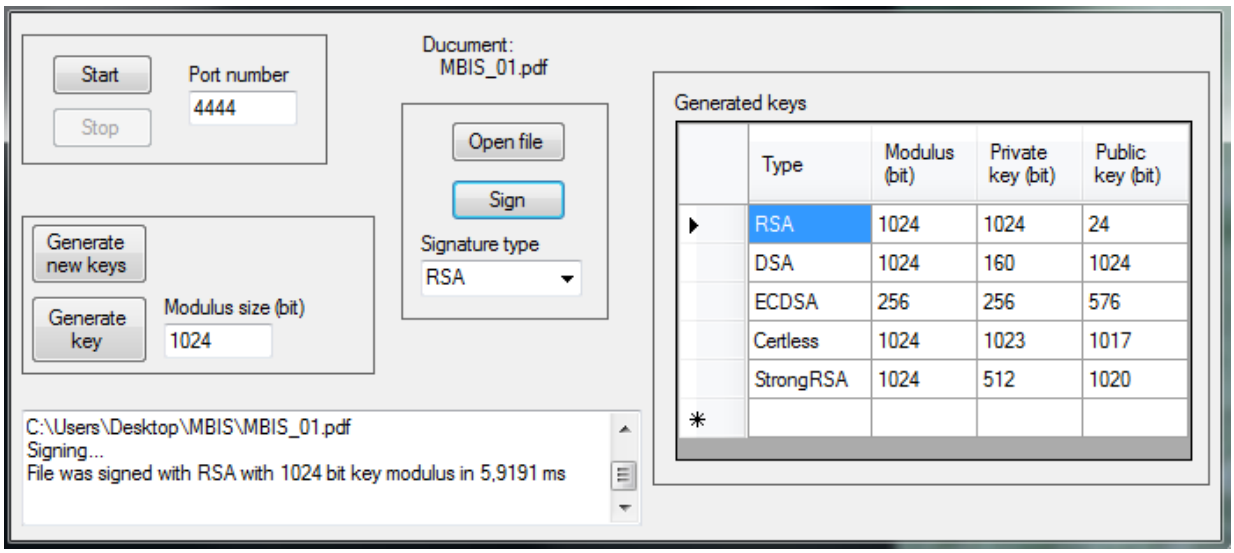

Рис. 2.1. Графічний інтерфейс користувача серверної програми

Додаток підтримує схеми підпису *RSA*, *DSA* та *ECDSA*, які включають *.NET Framework*, а також дві інші, які підтримуються поточною реалізацією. У схемі *Certless* реалізовано функції безсертифікатного підпису на основі ідентифікації користувача з використанням алгоритму *ElGamal*. Друга схема - *Strong RSA*.

Для автоматично згенерованих ключів 1024-бітний модуль використовується у випадку всіх схем, крім *ECDSA*, де використовується 256-бітовий модуль. Через обчислювальну складність генерації ключів для всіх п’яти схем цифрового підпису більші модулі не використовуються. Також можна створити нові пари ключів за допомогою кнопки «*Generate new keys*». І в цьому випадку використовуються ті самі модулі, що і при генерації ключів за допомогою кнопки «*Start*».

*Certless* - це схема на основі ідентифікації, тому за замовчуванням ідентифікатором використовується адреса електронної пошти.

Після вибору схеми в пункті «*Signature type*» та введення розміру модуля в бітах у полі «*Modulus size*» можна згенерувати нову пару ключів лише для вибраної схеми цифрового підпису за допомогою кнопки «*Generate key*». У випадку *Certless* з’являється пункт «*User ID*», який визначає ідентифікатор, що використовується у якості відкритого ключа. Розміри поточних згенерованих ключів відображаються в таблиці *Generated keys*, в якій в окремих схемах вказано розмір модуля, закритого та відкритого ключів у бітах.

Розглянуті схеми підтримують такі розміри модулів:

- *RSA*: від 384 до 16384 біт;
- *DSA*: від 512 до 1024 біт;
- *ECDSA*: 256 і 384 біт;
- *Certless* (без сертифіката): необмежено, але ділиться на 8;
- *StrongRSA*: 384 і від 512 до 2048 біт.

Щоб підписати документ, необхідно спочатку вибрати файл за допомогою кнопки «*Open file*». Програма дозволяє вибрати та підписати будь-який тип документа, назва якого після вибору відображається в пункті «*Document*» над

кнопкою «*Open file*», потім схема підпису вибирається зі списку «*Signature type*». Після натискання кнопки «*Sign*» відбувається підписання таким чином, що документ перетворюється на байтове поле, яке хешується функцією *SHA-1*, а отриманий 160- бітний хеш підписується за допомогою вибраної схеми цифрового підпису за допомогою поточного закритого ключа. Розмір ключа підпису вказано в таблиці «*Generated keys*» в правій частині графічного інтерфейсу користувача.

Для всіх схем для хешування документа використовується функція *SHA-1*, оскільки тільки вона підтримується для всіх реалізованих схем підпису. Під час визначення часу підписання та перевірки схем для порівняння, важливо підписати або перевірити хеш такої ж довжини, яка становить 160 біт у випадку *SHA-1*.

### 2.3.2 Реалізація клієнтського додатку

Клієнтська програма дозволяє завантажувати документи з цифровим підписом із сервера та перевіряти їх автентичність. Таким чином можна перевірити, чи були вони змінені під час передачі мережею.

Після введення *IP*- адреси сервера в пункті «*Server IP*» та порту в полі «*Server port*» клієнт може підключитися до сервера за допомогою кнопки «*Connect*» (див. рис. 2.2). Про успішне підключення користувач отримує інформацію у вікні зв’язку в нижній частині графічного інтерфейсу користувача програми. Кнопку «*Disconnect*» можна використати для остаточного відключення від сервера.

Після успішного підключення користувач може завантажити доступні відкриті ключі з сервера за допомогою кнопки «*Get public keys*», яка одразу з’явиться в таблиці відкритих ключів після завантаження. Щоб завантажити документ, підписаний сервером за допомогою кнопки «*Get signed document*», користувач повинен спочатку вибрати тип підпису, який користувач хоче завантажити зі списку «*Signature type*». Клієнт підтримує ті самі схеми підпису, які використовує сервер. Після успішного завантаження підписаного документа

його назва з суфіксом з'явиться в пункті «*Signed document*». Тепер кнопка перевірки підпису «*Verify*» є активною, і клієнт може виконати перевірку. Результат і час, необхідний для перевірки, вказані у відповідному вікні.

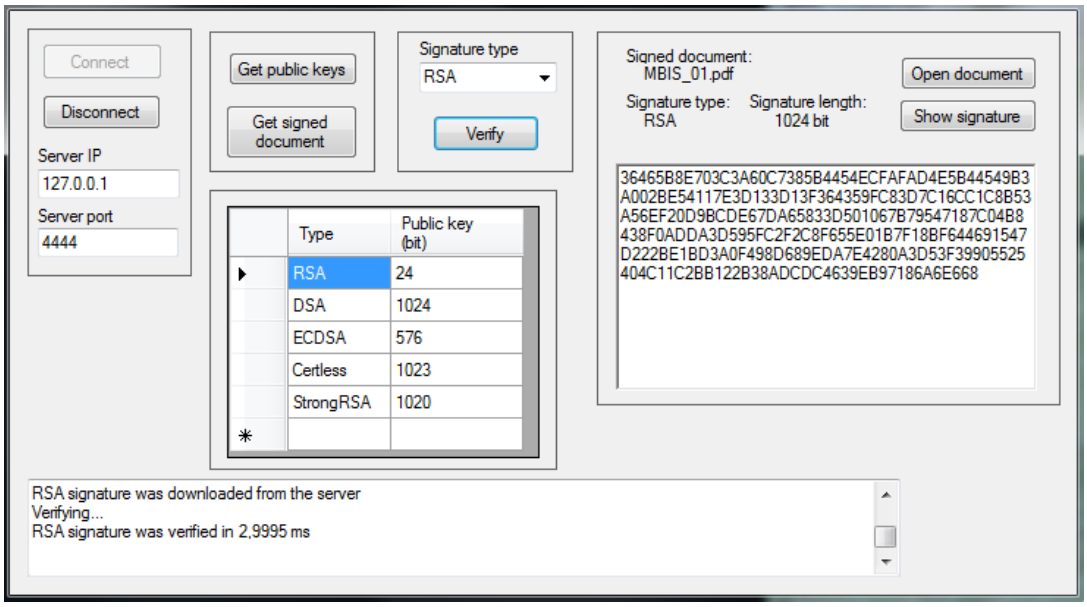


Рис. 2.2. Графічний інтерфейс користувача для клієнтської програми

Додаток за рис. 2.2, також дозволяє показувати вміст завантаженого підпису за допомогою кнопки «*Show signature*». Вміст відображається у вікні з правого боку графічного інтерфейсу користувача, а тип підпису також вказано в елементі «*Signature type*», його довжина - у «*Signature length*». Вміст записується в шістнадцятковому вигляді для *RSA*, *DSA* і *ECDSA*, а для *Certless* і *StrongRSA* записуються розміри параметрів, що формують цифровий підпис. Відкрити завантажений документ можна безпосередньо з програми, натиснувши кнопку Відкрити документ, яка відкриється в додатку з присвоєним розширенням в операційній системі.

# 3. РОЗРОБКА РЕКОМЕНДАЦІЙ ЩОДО ПІДВИЩЕННЯ ЗАХИЩЕНОСТІ СИСТЕМИ ЕЛЕКТРОННОГО ДОКУМЕНТООБІГУ

## 3.1 Визначення ефективності розроблюваних заходів

Розглянуті вище додатки дозволяють виявити деякі властивості досліджуваних схем цифрового підпису (*RSA, StrongRSA, DSA, ECDSA* та *Certless*), за якими можна порівняти їх ефективність. У той час як серверна програма може визначити час, необхідний для підписання 160- бітного хешу за допомогою відповідних схем, клієнтська програма дозволяє визначити час, необхідний для перевірки підпису, завантаженого з сервера.

Бібліотека *Microsoft.Office* містить клас для експорту значень в додаток *Microsoft Excel*, який використовувався в даній роботі для зберігання часу, виміряного функціями класу *System.Diagnostics.Stopwatch*. Підпис і перевірку виконано 100 разів поспіль, і середні значення розраховано на основі визначеного часу. Нижче наведено опис дослідження підпису і перевірки та показано їхні графічні залежності.

Серверна програма містить таблицю, яка показує розміри закритого та відкритого ключів окремих схем підпису. Розміри відкритих ключів, завантажених із сервера, вказані в таблиці клієнта. З точки зору ефективності ще однією важливою особливістю є довжина підписів, яку можна перевірити в клієнтському додатку після його завантаження.

### 3.1.1 Визначення часу підпису

Визначений час, необхідний для підписання 160- бітного хешу, створеного функцією хешування *SHA-1*, наведено на рисунках 3.1 - 3.5. *RSA, ECDSA* та *Certless* використовують для підпису закритий ключ такого ж розміру, як і довжина використовуваного модуля, а в *Strong RSA* розмір становить половину.

Оскільки розмір модуля збільшується для *DSA*, час, необхідний для підписання, залишається постійним, оскільки ця схема завжди використовує 160- бітний закритий ключ.

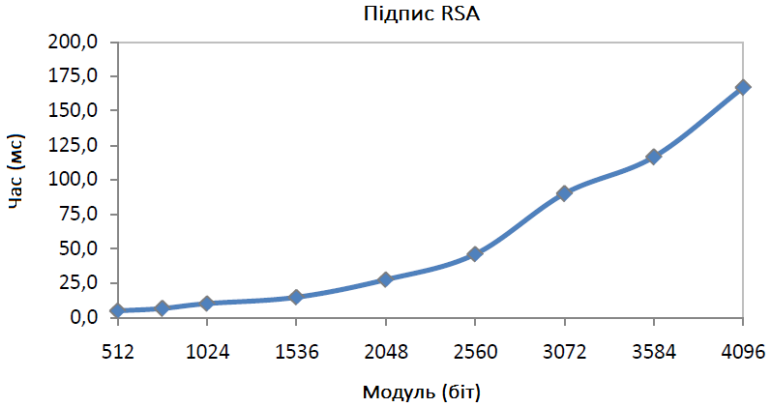


Рис. 3.1. Залежність часу, необхідного для підписання 160- бітного хешу, від розміру модуля, який використовується схемою *RSA*

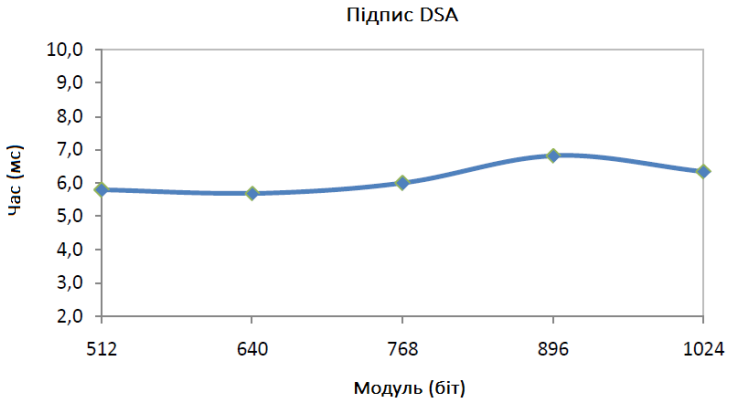


Рис. 3.2. Залежність часу, необхідного для підписання 160- бітного хешу, від розміру модуля, який використовується схемою *DSA*

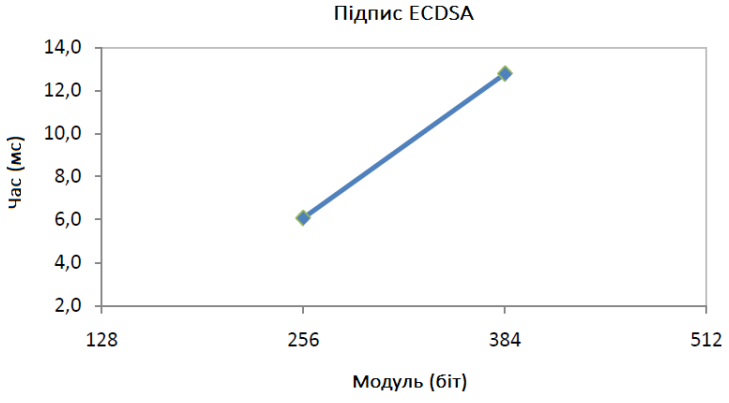

Рис. 3.3. Залежність часу, необхідного для підпису 160- бітного хешу, від розміру модуля, який використовується схемою *ECDSA*

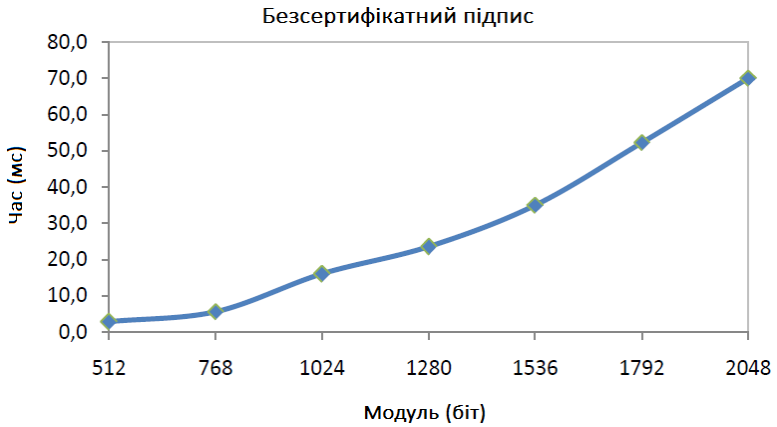


Рис. 3.4. Залежність часу, необхідного для підписання 160- бітного хешу, від розміру модуля, який використовується безсертифікатною схемою (*Cerless*)

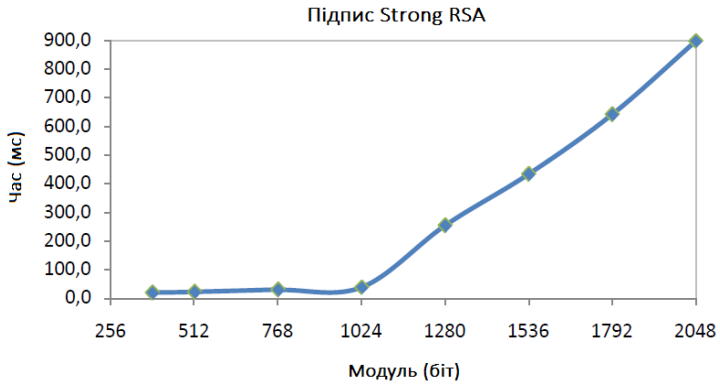


Рис. 3.5. Залежність часу, необхідного для підписання 160- бітного хешу, від розміру модуля, який використовується схемою *Strong RSA*

Підписання є найшвидшим для схем *DSA* та *ECDSA*, але у випадку *ECDSA* використовувані ключі забезпечують більший захист. Третя найшвидша за порядком схема *RSA*, а у випадку *Certless* підписання вимагає трохи більше обчислень. Найповільнішою з реалізованих схем є *Strong RSA*, яка у випадку 2048-бітного модуля підписує значно повільніше, ніж інші схеми.

### 3.1.2 Визначення часу перевірки

Виміряний час, необхідний для перевірки підпису для окремих схем, показано на малюнках 5.6-5.10. *DSA*, *Certless* і *Strong RSA* використовують відкритий ключ такої довжини, як розмір їхнього модуля, тоді як *ECDSA* використовує трохи більший, ніж подвоєний модуль. Найкоротший ключ у випадку схеми *RSA*, де відкритий ключ має необов'язковий розмір, і в цій реалізації використовувалися 24 біти.

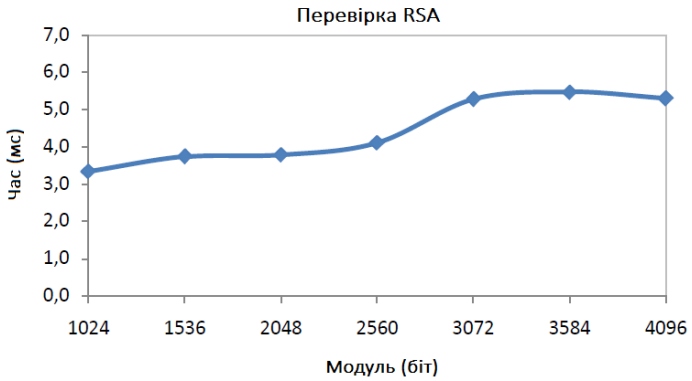


Рис. 3.6. Залежність часу, необхідного для перевірки підпису *RSA*, від розміру використовуваного модуля

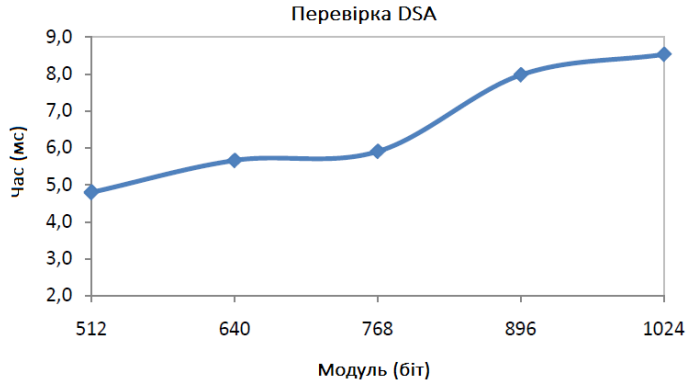


Рис. 3.7. Залежність часу, необхідного для перевірки підпису *DSA*, від розміру використовуваного модуля

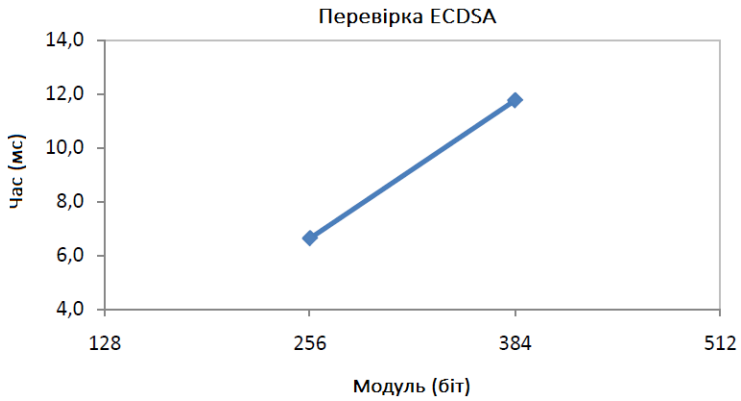


Рис. 3.8. Залежність часу, необхідного для перевірки підпису *ECDSA*, від розміру використовуваного модуля

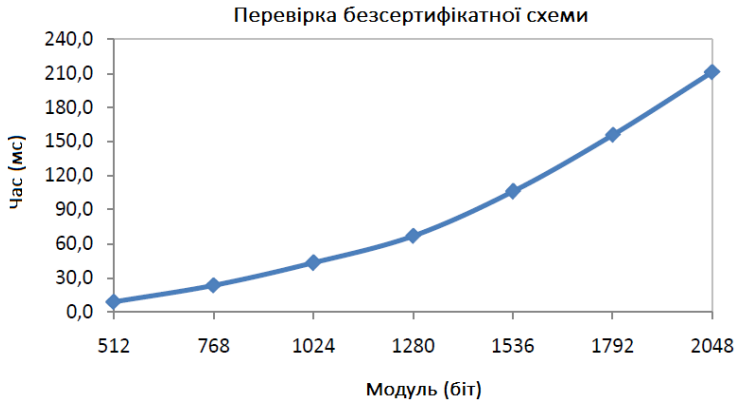


Рис. 3.9. Залежність часу, необхідного для перевірки підпису *Certless*, від розміру використовуваного модуля

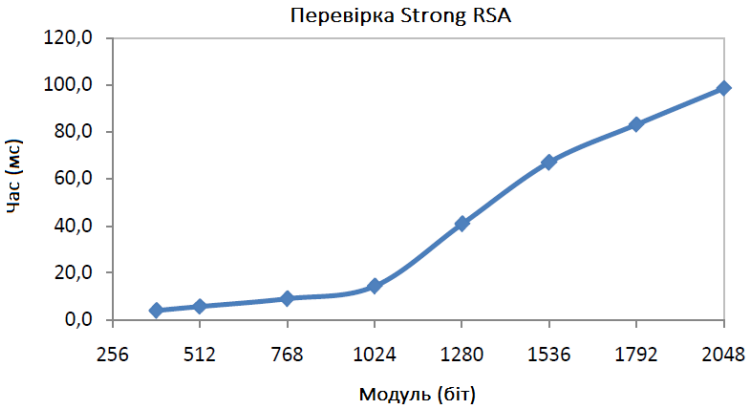


Рис. 3.10. Залежність часу, необхідного для перевірки підпису *Strong RSA*, від розміру використовуваного модуля

Найшвидша перевірка відбувається у випадку *RSA*, де час, необхідний для перевірки підпису, не залежить від довжини використовуваного модуля

(опціонально обраного розміру відкритого ключа). Подібні значення визначено для *DSA* і *ECDSA*, але з модулями *ECDSA* забезпечується вищий рівень безпеки, ніж в інших схемах. Найбільш трудомісткою перевіркою реалізованих схем є для *Strong RSA* та *Certless*.

### 3.1.3 Довжина підпису

Довжину підписів можна записати на стороні клієнта після завантаження. *DSA* має найкоротший підпис серед реалізованих схем, де він становить 320 біт. Підписи також є короткими для *ECDSA*, вони дорівнюють подвоєній довжині модуля, але в цій схемі використовуються значно менші модулі порівняно з іншими. У випадку *RSA* та *Certless* довжина підпису дорівнює розміру використовуваного модуля. *Strong RSA* створює підпис довжиною до 1024 біт.

### 3.1.4 Підсумок результатів вимірювань

Порівняння властивостей окремих досліджуваних схем цифрового підпису наведено в таблицях 3.1 та 3.2. Модулі, які використовуються в першій таблиці, відповідають рівням безпеки, які вважаються безпечними, а час, необхідний для їх прориву, становить $10^{12}$ років. Однак, за деяким припущеннями [15], ці значення не вважаються однозначно безпечними для довгострокових підписів. Обрано 1024 біт, оскільки більшість схем підтримують це значення модуля. У випадку *ECDSA* не вдалося вибрати 160- бітний модуль, який би відповідав цим рівням безпеки, оскільки досліджувана схема підтримує лише 256 і 384 біт. У *ECDSA* 256- бітний модуль забезпечує безпеку на рівні приблизно 2560 біт для проблем *IFP* і *DLP*.

Таблиця 3.1

Порівняння властивостей досліджуваних схем (необхідний час для прориву - $10^{12}$ років)

| Проблема | Схема | Модуль (біт) | Підпис (мс) | Перевірка (мс) | Довжина підпису (біт) |
|---|---|---|---|---|---|
| *IFP* | *RSA* | 1024 | 10,6 | 3,3 | 1024 |
| | *Strong RSA* | 1024 | 39,2 | 14,4 | 1022 |
| *DLP* | *DSA* | 1024 | 6,3 | 8,5 | 320 |
| | *Cerless* | 1024 | 16,1 | 43,1 | 1024 |
| *ECDLP* | *ECDSA* | 256 | 6,1 | 6,6 | 512 |

Наступна таблиця (табл. 3.2) містить схеми з модулями, які забезпечують вищий рівень безпеки, ніж у попередньому випадку, і час, необхідний для їх злому, становить $10^{20}$ років. Досліджувана схема *DSA* не підтримує більше за модуль 1024 біт, тому це значення також внесено до наступної таблиці.

Таблиця 3.2

Порівняння властивостей досліджуваних схем (необхідний час для прориву - $10^{20}$ років)

| Проблема | Схема | Модуль (біт) | Підпис (мс) | Перевірка (мс) | Довжина підпису (біт) |
|---|---|---|---|---|---|
| *IFP* | *RSA* | 2048 | 27,7 | 3,8 | 2048 |
| | *Strong RSA* | 2048 | 899 | 98,8 | 4095 |
| *DLP* | *DSA* | 1024 | 6,3 | 8,5 | 320 |
| | *Cerless* | 2048 | 70,2 | 211,5 | 2048 |
| *ECDLP* | *ECDSA* | 256 | 6,1 | 6,6 | 512 |

## 3.2 Придатність технологій цифрового підпису для застосування на пристроях зі слабкою обчислювальною потужністю

У слабких обчислювальних пристроях, таких як смарт-карти та сенсорні мережі, дуже важливо, щоб розгорнута схема мала дуже ефективний підпис, перевірку або в деяких випадках і те, і інше. Через обмежені можливості пам'яті, іншою важливою ознакою є довжина створюваного підпису. Чим коротші підписи, створені схемою, тим більше їх можна зберегти в пам'яті пристрою.

Схема *DSA* має швидке підписання, тоді як *RSA* має швидку перевірку. *ECDSA* ефективніша за дві інші схеми, оскільки вона має швидкість підпису на рівні *DSA* та швидкість перевірки на рівні *RSA*. Найкоротші підписи з досліджуваних схем створені *DSA*, підпис якої має значення 320 біт. Короткі підписи також створюються *ECDSA*, де довжина підпису дорівнює подвоєному значенню використаного модуля.

Результати виміряних значень показують, що схеми *Certless* і *StrongRSA* дещо відстають від трьох інших у швидкості підписання та перевірки, але стійкі до найнебезпечніших атак. Довжина створених підписів на рівні *RSA*.

*Certless* - це схема без сертифіката. Ця функція може бути перевагою в системах із проблемним попереднім розподілом автентифікованих ключів (наприклад, бездротові мережі). Ця схема підпису реалізована таким чином, щоб будь-яку схему типу *ElGamal* можна було створити як схему без сертифіката. У цьому випадку використовується *ElGamal*, але за допомогою більш ефективної схеми, такої як *DSA*, можна було б збільшити швидкість підпису та перевірки і скоротити підпис до 320 біт.

З табл. 3.2 видно, що *ECDSA* є найбільш придатним варіантом для обчислювально слабких пристроїв із порівнюваних схем.

## 3.3 Рекомендації щодо застосування технологій цифрового підпису з метою підвищення захисту системи електронного документообігу

### 3.3.1 Безпечні сховища

Використовуючи цифровий підпис, звичайно, також потрібно враховувати питання безпеки. Так, зберігання секретного ключа, наприклад, на жорсткому диску персонального комп’ютера є не найбільш безпечним. І тому були розроблені більш безпечні та переважно портативні пристрої, до яких можна віднести чіп-карти та *usb*- жетони.

Індивідуальні чіпові картки можна розділити на дві основні групи:

- пасивні - чіп-карти, що містять важливу інформацію, служать лише елементом пам’яті;·

- активні - у випадку карт з активним чіпом (оснащених процесором на додаток до пам’яті), де використовується інший метод захисту збережених даних автентифікації на карті пам'яті (дані зберігаються тут у спеціально захищеній області пам'яті, яка не дозволяє їх легко прочитати).

Різниця між активними та пасивними чіп-картками полягає в тому, як використовується збережений ключ. У випадку активних смарт-карт вибрана конфіденційна криптографічна інформація ніколи не залишає середовище смарт-карти, тому, на відміну від пасивних карт, які надають всю криптографічну інформацію, вони не можуть бути використані неналежним чином пристроєм (зчитувачем), з яким спілкується карта. Зокрема, з активними картками процесор на картці, який має доступ до даних автентифікації в захищеній області пам’яті карти, через запрограмовану послідовність кроків доводить пристрою автентифікації, що він «знає» правильну автентифікацію.

*USB*- токени використовують той самий принцип, що й карти з чіпом, головна перевага полягає в тому, що не потрібен зчитувач, а достатньо *USB*-роз’єму. Зазвичай чіп-картки, такі як *USB*- токени, захищені *PIN*-кодом. Обидва ці пристрої можна придбати в центрі сертифікації для їх кваліфікованих сертифікатів.

### 3.3.2 Можливості використання кваліфікованого цифрового підпису

*Підписання та шифрування електронної пошти*

Підписання електронної пошти, як «звичайним» цифровим підписом, так і кваліфікованим підписом, є найпоширенішою функцією цифрового підпису. Оскільки вона забезпечує вже згадані властивості цілісності даних, автентифікації, невідмовність і обов'язковість до документа. Таким чином, одержувач повідомлення може бути впевнений, що він дійсно отримав повідомлення від даного відправника.

Фактичному підпису передує налаштування поштового клієнта, щоб повідомлення могли підписуватися цифровим способом. У цьому випадку можна використати *Outlook Express*, і оскільки він є частиною операційної системи *Microsoft Windows*.

Після попереднього отримання сертифіката потрібно призначити сертифікат обліковому запису, який має ту саму адресу електронної пошти, яку користувач вказав у заявці. Сама установка є складною. У меню «Інструменти» > «Акаунти» > «Пошта» > «Наш обліковий запис» > «Властивості» > «Безпека» потрібно вибирати сертифікат для підпису та шифрування, а система автоматично пропонує необхідні сертифікати. І тепер можливим є підписання чи шифрування відповідних повідомлень. Для того щоб підписати електронну пошту, потрібно натиснути кнопку «Підписати», і процедура буде завершеною. Одержане повідомлення буде автоматично перевірено поштовим клієнтом електронної пошти з підписом, але слід зауважити, що для того, щоб повідомлення вважалося надійним, воно повинно мати сертифікати відповідного центра сертифікації, позначені як надійні в сховищі сертифікатів. Також можливим є встановлення автоматичної перевірки *CRL* у меню «Інструменти» > «Параметри» > «Безпека» > «Додатково» > «Перевірка відкликання» > лише під час роботи в режимі онлайн.

Що стосується шифрування повідомлення, то тут відправник спочатку повинен отримати відкритий ключ одержувача, який він отримує або з підписаного повідомлення, надісланого одержувачем, або зі сторінок центру сертифікації одержувача. Якщо він хоче отримати відкритий ключ із отриманого повідомлення, він відкриває отримане повідомлення та натискає «Файл» > «Властивості» > «Безпека» > «Переглянути сертифікати» > «Додати до каталогу». Далі ще раз потрібно натиснути «Зашифрувати» під час шифрування повідомлення, і поштовий клієнт зашифрує повідомлення загальнодоступною адресою одержувача. Важливо зазначити, що якщо одержувач повідомлення не використовує поштовий клієнт, а отримує доступ до повідомлень через інтерфейс *www*, то більшість безкоштовних поштових сервісів відображатимуть підпис як вкладення до повідомлення, оскільки вони не використовують протокол *S/MIME*,

за допомогою якого можна цифрово підписувати та шифрувати повідомлення, але винятком є, наприклад, *Gmail*.

*Електронний банкінг*

Термін електронний банкінг включає домашній банкінг, інтернет-банкінг, *GSM*-банкінг і телефонний банкінг. У даному випадку мова йде лише про домашнній - та інтернет-банкінг, в яких можливим є використання цифрового підпису.

Різниця між домашнім та інтернет-банкінгом полягає у використанні спеціального додатку у випадку домашнього банкінгу, який встановлюється безпосередньо на комп'ютері і тому не підлягає передачі. Тоді як для доступу до інтернет-банкінгу достатньо інтернет-браузера, і можливим є вхід до свого облікового запису з будь-якого місця.

При налаштуванні додатку домашнього банкінгу клієнт повинен спочатку створити свій цифровий підпис. Клієнт отримує персональний сертифікат, а його відкритий ключ реєструється в банку. Як правило, цей сертифікат зберігається на смарт-картці. Банки здебільшого видають та приймають виключно власні сертифікати.

За допомогою Інтернет-банкінгу клієнт має можливість вибрати, як він хоче захистити та реалізувати спосіб входу в свій обліковий запис і згодом авторизувати платіжні доручення. Банки пропонують форму *SMS*-ключа, ідентифікаційного номера та *PIN*-коду, а також персонального сертифіката. Проблема полягає в тому, що ці сертифікати не можна використовувати так само, як кваліфіковані сертифікати.

Електронний банкінг є досить популярним додатком серед клієнтів, оскільки він значно полегшує роботу з фінансами, забезпечує гнучкість і нижчі комісії. Крім того, це також економить кошти для самих банків.

*Електронна комерція*

Електронна комерція останнім часом переживає великий бум. Цифровий підпис тут використовується між постачальниками, торговцями або клієнтами під час надсилання рахунків-фактур і замовлень. Це прискорює весь бізнес-процес, і

перевага полягає в тому, не є потрібними особисті контакти. Водночас ділові партнери мають можливість вибирати, чи використовуватимуть вони у своїй діяльності кваліфікований цифровий підпис і відповідно кваліфіковані сертифікати, чи достатньо буде комерційних сертифікатів.

*Інші можливості*

1) Спілкування з регіональною та муніципальною владою або муніципальні установи, які використовують електронні кабінети, але, як показує практика, комунікація за допомогою подібних електронних засобів ще не є достатньо розвиненою.

2) Сертифікати сервера - призначені для безпечного зв'язку між користувачами та сервером, де важливі дані відвідувачів сервера захищені за допомогою сертифіката та *SSL*, і зрозуміло, що доступ мають лише власники певних сертифікатів.

3) Продаж цінних паперів - використовує кваліфіковані сертифікати для підписання електронних листів, пов'язаних із звітністю про операції з цінними паперами на ринках, інформаційні зобов'язання емітентів цінних паперів.

4) Підписання документів (*pdf, doc, xls*), фотографії та практично всі дані, які можна перевести в цифрову форму.

# ВИСНОВКИ

Основною метою поточної роботи є дослідження технологій цифрового підпису та визначення підходів до розробки нових ефективних схем підпису.

У практичній частині досліджено кілька ефективних схем на обраній мові програмування та продемонстровано застосування відповідних додатків для порівняння та аналізу ефективності досліджуваних схем.

Було виявлено, що найбільш використовувані сьогодні схеми підпису RSA та DSA. Вони також мають характеристики, які можуть бути менш перспективними в майбутньому. Спочатку для схеми RSA використовувалася функція хешування MD5, яка пізніше була змінена на більш безпечну SHA-1. Навіть DSA використовує SHA-1, який наразі більше не є рекомендованою функцією хешування для забезпечення довгострокової безпеки. З часом його слід замінити більш безпечним SHA-2.

Визначено, що в даний час досить поширеними схемами цифрового підпису є RSA і DSA, з високою швидкістю вдосконалення в комп'ютерних технологіях, та потребують використання все більших ключів.

Слід зазначити, що для RSA зі збільшенням довжини ключа безпека зростає повільно порівняно з обчислювальними вимогами, які зростають набагато швидше. Такі фактори призводять до значного зниження ефективності цих схем і, у випадку RSA, до створення довгих підписів, що є головним недоліком у системах з обмеженою пам'яттю.

Визначено, що ефективність схем цифрового підпису можна підвищити за допомогою еліптичних кривих, як у випадку схеми цифрового підпису ECDSA.

Таким чином, можна очікувати розширення бездротових сенсорних мереж, в яких проблемним питанням є безпечний розподіл ключів шифрування між окремими вузлами, а також проблемою є забезпечення автентичності та електронного підпису даних. Поточні схеми цифрового підпису не підходять для ефективного застосування у цих мережах через великі вимоги до пам’яті.

Поточна тенденція підвищення ефективності рухається до теоретичних проектів схем підпису з використанням білінійного узгодження. За допомогою криптосистем, побудованих на основі ідентифікації користувача, можна створювати цифрові підписи без необхідності сертифікатів, що вирішує проблему з автентифікацією. Поєднуючи методи білінійного зіставлення або еліптичної кривої з цифровим підписом без сертифіката, можна створювати ефективні схеми цифрового підпису, які генерують досить короткі підписи.

За результатами дослідження, рекомендації щодо застосування технологій цифрового підпису зосереджено за наступними напрямками:

1) Вибір окремих схем цифрового підпису, залежно від вимог до функціонування комунікаційної системи.

2) Застосування відповідних технологій у системах з високими вимогами до ефективності ресурсоємності криптографічних продуктів.

3) Використання безпечних сховищ.

4) Рекомендації щодо ефективного застосування кваліфікованого цифрового підпису.

Сформовані рекомендації можуть бути використані співробітниками відділів інформаційної безпеки організацій та фізичними особами, що мають на меті реалізацію заходів з підвищення захищеності функціонування систем електронного документообігу.

# ПЕРЕЛІК ПОСИЛАНЬ

1. Ясінська, Алла. (2022). ПРОБЛЕМИ ТА ПЕРСПЕКТИВИ ЕЛЕКТРОННОГО ДОКУМЕНТООБІГУ В УМОВАХ ЦИФРОВОЇ ТРАНСФОРМАЦІЇ. Молодий вчений. 128-134. 10.32839/2304-5809/2022-11-111-27.

2. Malanchuk, L. & Zhakun, Yu. (2021). АНАЛІЗ СИСТЕМИ ЕЛЕКТРОННОГО ДОКУМЕНТООБІГУ В ДЕРЖАВНИХ УСТАНОВАХ ТА ШЛЯХИ ЇЇ ВДОСКОНАЛЕННЯ. Bulletin National University of Water and Environmental Engineering. 3. 73. 10.31713/ve320217.

3. Фещенко, Євгенія & Ганбарова, Діана. (2023). ОСОБЛИВОСТІ ВЕДЕННЯ ЕЛЕКТРОННОГО ДОКУМЕНТООБІГУ СУБ'ЄКТАМИ ГОСПОДАРЮВАННЯ В СУЧАСНИХ УМОВАХ. Молодий вчений. 127-133. 10.32839/2304-5809/2023-2-114-24.

4. Головацька, С. (2021). СИСТЕМИ ЕЛЕКТРОННОГО ОБЛІКУ І ЗВІТНОСТІ ПІДПРИЄМСТВ: АКТУАЛЬНІ АСПЕКТИ. Підприємництво і торгівля. 11-16. 10.36477/2522-1256-2021-32-02..

5. Плотніков, В & Борцова, Ю. (2020). Захист даних засобом цифрового підпису. Automation of technological and business processes. 11. 49-55. 10.15673/atbp.v11i4.1599.

6. Boschini, Cecilia & Fiore, Dario & Pagnin, Elena. (2022). Progressive and Efficient Verification for Digital Signatures. 10.1007/978-3-031-09234-3_22.

7. Lawrence, Linju & Shreelekshmi, R. (2021). Chained Digital Signature for the Improved Video Integrity Verification. 10.3233/FAIA210284.

8. Fukumitsu, Masayuki & Hasegawa, Shingo & Isobe, Shuji & Shizuya, Hiroki. (2014). On the Impossibility of Proving Security of Strong-RSA Signatures via the RSA Assumption. 10.1007/978-3-319-08344-5_19.

9. HARN, L., REN, J., LIN. Ch. Design of DL-based certificateless digital signatures. The Journal of Systems and Software, 82:789-793, 2009.

10. Wang, Neng-Chung & Chen, Young & Chen, Hong-Li. (2013). An Efficient Grid-Based Pairwise Key Predistribution Scheme for Wireless Sensor Networks. Wireless Personal Communications. 78. 801-816. 10.1007/s11277-013-1493-1.

11. Huang, Hui-Feng. (2008). A Pairwise Key Pre-distribution Scheme for Wireless Sensor Network. 77-82. 10.1007/978-3-540-69304-8_9.

12. Jian-zhi, Deng & Xiao-hui, Cheng & Qiong, Gui. (2009). Design of Hyper Elliptic Curve Digital Signature. Proceedings - 2009 International Conference on Information Technology and Computer Science, ITCS 2009. 2. 45 - 47. 10.1109/ITCS.2009.146.

13. Li, Y.-J & Li, W. & Zhu, X.-Y & Ge, J.-H. (2007). New signature scheme based on the strong RSA assumption. 34. 634-637.

14. Rousselt, Rick. (2021). Pro Microsoft Teams Development: A Hands-on Guide to Building Custom Solutions for the Teams Platform. 10.1007/978-1-4842-6364-8.

15. Rosenberg, Burton. (2010). Handbook of Financial Cryptography and Security. Handbook of Financial Cryptography and Security. 10.1201/9781420059823.

# ДЕМОНСТРАЦІЙНІ МАТЕРІАЛИ

**(Презентація)**