МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ОДЕСЬКИЙ ДЕРЖАВНИЙ ЕКОЛОГІЧНИЙ УНІВЕРСИТЕТ

Факультет Комп'ютерних наук,
управління та адміністрування
Кафедра Інформаційних технологій

МАГІСТЕРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему: Розробка інформаційної системи для сертифікаційного центру

Виконав студент 2 курсу групи МІС-20
спеціальності 122 Комп'ютерні науки
_____
Головатюк Володимир Андрійович

Керівник к.т.н., доцент
Гнатовська Ганна Арнольдівна

Рецензент к.геогр.н., доцент
Кузніченко Світлана Дмитрівна

Одеса 2021

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ОДЕСЬКИЙ ДЕРЖАВНИЙ ЕКОЛОГІЧНИЙ УНІВЕРСИТЕТ

Факультет Комп'ютерних наук, управління та адміністрування
Кафедра Інформаційних технологій
Рівень вищої освіти магістр
Спеціальність 122 Комп'ютерні науки
(шифр і назва)

ЗАТВЕРДЖУЮ

Завідувач кафедри_____

_____

" 28 " жовтня 2021 р.

З А В Д А Н Н Я
НА МАГІСТЕРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ

Головатюку Володимиру Андрійовичу
(прізвище, ім'я, по батькові)

1. Тема роботи Розробка інформаційної системи для сертифікаційного центру

керівник роботи Гнатовська Ганна Арнольдівна, к.т.н., доцент,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом закладу вищої освіти від " 18 " жовтня 2021р. №216 «С»

2. Строк подання студентом роботи 9 грудня 2021р.

3. Вихідні дані до роботи: мова програмування: PHP; СУБД: Mysql; бібліотека: Openssl; алгоритми асиметричного шифрування: RSA, DSA; хеш-алгоритми: md2/4/5, sha1, sha224/256/384/512

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити): Вступ; Дослідження та аналіз криптографічних алгоритмів; Аналіз інфраструктури відкритих ключів; Проектування ІС сертифікаційного центру висновки.

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

6. Консультанти розділів роботи

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

7. Дата видачі завдання « 28 » жовтня 2021 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломної роботи	Термін виконання етапів роботи	Оцінка виконання етапу	
			у %	за 4-х бальною шкалою
1.	Дослідження та аналіз криптографічних алгоритмів	28.10.21 – 02.11.21	96%	відмінно
2.	Аналіз інфраструктури відкритих ключів Особливості організації засвідчувального центру	02.11.21 –10.11.21	94%	відмінно
3.	Огляд та аналіз аналогічних систем. Постановка завдання	11.11.21 –12.11.21	92%	відмінно
4.	Проектування ІС для сертифікаційного центру	12.11.21 – 21.11.21	92%	відмінно
5.	Рубіжна атестація	22.11.20	96%	відмінно
6.	Програмна реалізація ІС сертифікаційного центру	21.11.21-04.12.21	94%	відмінно
7.	Оформлення пояснювальної записки та презентації	04.12.21-08.12.21	96%	відмінно
8.	Подання роботи на кафедру	09.12.21		
9.	Перевірка на плагіат	10.12.21		
10.	Рецензування	16.12.21		
	Інтегральна оцінка виконання етапів календарного плану (як середня по етапам)		94%	відмінно

Студент _____ Головатюк В.А.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Гнатовська Г.А.
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Тема магістерської роботи «Розробка інформаційної системи для сертифікаційного центру».

Актуальність магістерської роботи полягає в розробки інформаційної системи для сертифікаційного центру, що забезпечить можливість створення, видачі, керування сертифікатами криптографічних ключів електронного цифрового підпису для користувачів будь-якої корпоративної організації , що вирішить задачу електронного документаобігу для власних потреб огранизації та надасть можливість розвитку електронної комерції.

Об'єкт дослідження – процеси виконання криптографічних асиметричних перетворень для створення електронного цифрового підпису.

Предмет дослідження – механізми інфраструктури відкритих ключів для розв'язання завдань побудови і супровіду інформаційної системи власного сертифікаційного центру.

Мета роботи – розробка інформаційної системи для сертифікаційного центру, що надає можливість створення, видачі, керування сертифікатами криптографічних ключів електронного цифрового підпису користувачів, яка може бути застосована у будь-який корпоративній організації.

В роботі було проведено дослідження та аналіз криптографічних алгоритмів, визначена інфраструктура відкритих ключів, проведено проектування та програмна реалізація інформаційної системи для сертифікаційного центру. Практична цінність роботи полягає в тому, що розроблена система може буди використана для організації власного центру сертифікації будь-якою корпоративною організацією.

Ключові слова: КРИПТОГРАФІЧІ АЛГОРИТМИ, ЕЛЕКТРОННИЙ ЦИФРОВИЙ ПІДПИС, СЕРТИФІКАТИ КРИПТОГРАФІЧНИХ КЛЮЧІВ.

Магістерська робота містить 71 сторінку, 4 таблиці, 29 рисунків, 13 посилань.

ANNOTATION

The theme of master's work is "Development of an Information System for a Certification Centre".

The urgency of the master's thesis is to develop an information system for the certification center, which will provide the ability to create, issue, manage certificates of cryptographic keys of electronic digital signature for users of any corporate organization, which will solve the problem of electronic document management.

The object of research – the processes of cryptographic asymmetric transformations to create an electronic digital signature.

The subject of the research is the mechanisms of public key infrastructure for solving the tasks of building and maintaining the information system of one's own certification center. The purpose of the work is to develop an information system for the certification center, which provides the ability to create, issue, manage certificates of cryptographic keys of electronic digital signature of users, which can be used in any corporate organization.

The research and analysis of cryptographic algorithms were carried out, the public key infrastructure was determined, the design and software implementation of the information system for the certification center was carried out. The practical value of the work is that the developed system can be used to organize its own certification center by any corporate organization.

Keywords: CRYPTOGRAPHIC ALGORITHMS, ELECTRONIC DIGITAL SIGNATURE, CRYPTOGRAPHIC KEY CERTIFICATES.

The master's thesis contains 71 pages, 4 tables, 29 figures, 13 references.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧОК І ТЕРМІНІВ

Скорочення

CRL – список відкликаних сертифікатів

CSR – запит на підпис сертифіката

PKI – інфраструктура відкритих ключів

SHA – Алгоритм безпечного хешування

АЦСК – Акредитований Центр сертифікації ключів

АЦСК ІДД – Акредитований центр сертифікації ключів Інформаційно-довідкового департаменту ДФС

ЕЦП – электронно-цифровий підпис

ЗЦ – засвідчувальний центр

ЦЗО – Центральний засвідчувальний орган

ЦС – Центр сертифікації

Терміни

Відкритий ключ – ключ, створений у парі із закритим ключем, що має такий же електронний відбиток, як і закритий ключ, якому він відповідає, використовується для шифрування даних і перевірки підпису.

Довірені центри сертифікації – список центрів сертифікації, яким довіряють власники сертифікатів. Щоб зробити будь-який центр довіреним, достатньо одержати від нього сертифікат і внести його в список довірених центрів.

Закритий ключ – ключ, що зберігається в безпечному сховищі, створений з використанням алгоритмів шифрування, що має свій унікальний електронний відбиток, використовується для одержання зашифрованих даних і підписи даних.

Кореневі центри сертифікації – центри сертифікації, яким довіряють всі, або керуючись політикою підприємства, або завдяки

передвстановленим настроюванням сховища сертифікатів, і які можуть перебувати на початку шляху довіри.

Особисті сертифікати – сертифікати, які зберігаються в користувача в особистому сховищі сертифікатів.

Сертифікат – електронний документ, який містить електронний ключ користувача, інформацію про користувача підпис, що засвідчує, центру видачі сертифікатів і інформацію про термін дії сертифіката.

Шлях довіри – ланцюжок документів, який дозволяє впевнитися, що пред'явлений сертифікат був виданий довіреним центром; останньою ланкою в цьому ланцюжку є пред'явлений сертифікат.

ВСТУП

В останні роки все більш актуальним стає ведення документообігу онлайн, ця тенденція з кожним роком зберігається і посилюється. В умовах розвитку сучасної цивілізації все частіше паперова технологія зберігання та обробки інформації замінюється на електронну. У таких умовах особливу актуальність набуває створення електронного документообігу. Важливою є розробка електронних механізмів захисту документа, оскільки паперові носії мають такі атрибути, як друку, водяні знаки та особлива фактура поверхні. Електронний цифровий підпис (ЕЦП) допомагає вести діяльність засобами мережі Інтернеті, замінити паперові документи на електронні з можливістю надання таких документів через Інтернет в режимі онлайн.

Все більше зростає інтерес до систем захищеного електронного документообігу, які використовують електронний цифровий підпис. Адже з їхнім використанням суттєво прискорюється проведення численних комерційних операцій, скорочуються обсяги паперової бухгалтерської документації, заощаджується час співробітників і витрати підприємства, пов'язані з підписанням договорів, оформленням платіжних документів, наданням звітності в контролюючі органи, одержанням довідок від різних держустанов і іншим [1].

Згідно із законодавством України, електронним документом є інформація, яка зафіксована у вигляді електронних даних, включаючи обов'язкові реквізити документа, зокрема, електронний цифровий підпис. Електронний підпис є обов'язковим реквізитом електронного документа, без якого він не може мати справжню юридичну чинність і служити основою для здійснення операцій [2].

Технологія електронного цифрового підпису базується на криптографічних асиметричних перетвореннях, тобто, підпис створюється за допомогою одного персонального секретного ключа, а перевіряється іншим – відкритим. Технологія передбачає наявність сертифіката ключа – спеціального

електронного документа, необхідного для верифікації відкритого ключа (як доказ правомірності володіння даним персональним ключем з боку конкретної особи). Саме це й дозволяє за допомогою ЕЦП поступово перевести в електронний режим багато послуг держорганів країни, а також прискорити і захистити виробництво реєстраційних процедур, оформлення документів, участь у тендерах, підписання контрактів (у тому числі міжнародних), виконання електронних платежів і розвиток електронної комерції.

Широкому колу користувачів ЕЦП стала доступна з початку 2006 року, коли в Україні був відкритий Центральний засвідчувальний орган. Основними його функціями стали реєстрація засвідчувальних центрів і центрів сертифікації ключів, формування і обслуговування кореневих сертифікатів ключів, акредитація засвідчувальних центрів і центрів сертифікації ключів.

Відповідно до законодавства України під послугами ЕЦП мається на увазі: надання в користування засобів ЕЦП; надання допомоги при генерації відкритих і особистих ключів; формування і обслуговування (блокування, скасування, відновлення) сертифікатів відкритих ключів, надання інформації про сертифікати; послуги фіксування часу. Установлення довіри – це основний механізм інфраструктури відкритих ключів. Довіреною особою в централізованій системі сертифікатів виступає центр сертифікації. Відкриті ключі, і дані про власника, що зберігаються разом з ними, підписані центром сертифікації, вважаються достовірними. У цьому полягає важливе значення центру сертифікації [3].

Актуальність магістерської роботи полягає в розробки інформаційної системи для сертифікаційного центру, що забезпечить можливість створення, видачі, керування сертифікатами криптографічних ключів електронного цифрового підпису для користувачів будь-якої корпоративної організації , що вирішить задачу електронного документаобігу для власних потреб огранизації та розвиток електронної комерції.

Об'єкт дослідження – процеси виконання криптографічних асиметричних перетворень для створення електронного цифрового підпису.

Предмет дослідження – механізми інфраструктури відкритих ключів для розв'язання завдань побудови і супровіду інформаційної системи власного сертифікаційного центру.

Мета роботи – розробка інформаційної системи для сертифікаційного центру, що надає можливість створення, видачі, керування сертифікатами криптографічних ключів електронного цифрового підпису користувачів, яка може бути застосована у будь-який корпоративній організації.

Інформаційна система для сертифікаційного центру повинна забезпечувати формування власного секретного ключа, сертифікату, що містить відкритий ключ даного центру і підписаний їм самим. ІС для сертифікаційного центру повинна надійно зберігати закритий ключ свого кореневого сертифіката, тому що їм підписуються сертифікати клієнтів. Відкритий же ключ, як і сам сертифікат, повинен бути привселюдно доступний, тому що з його допомогою користувачі можуть перевіряти сертифікати один одного.

1 ДОСЛІДЖЕННЯ ТА АНАЛІЗ КРИПТОГРАФІЧНИХ АЛГОРИТМІВ

1.1 Дослідження криптографічних систем з відкритим ключем

Криптографічна система з відкритим ключем – являє собою систему шифрування або електронного цифрового підпису (ЕЦП), при якому відкритий ключ передається по відкритому (тобто незахищеному, доступному для спостереження) каналу, і використовується для перевірки ЕЦП і для шифрування повідомлення. Для здійснення генерації ЕЦП і для виконання операції розшифрування повідомлення використовується секретний ключ. Ідея криптографії з відкритим ключем дуже тісно пов'язана з ідеєю однобічних функцій, тобто таких функцій, що по відомому x досить просто знайти значення $f(x)$, тоді як визначення x з $f(x)$ складно в сенсі теорії. В основі таких функцій лежать відомі обчислювальні задачі [4]:

- задача факторизації (розкладання на множники) великих цілих чисел (RSA);
- задача дискретного логарифмування (EGSA).

Розглянемо схему шифрування з відкритим ключем. Припустимо, що K – множина ключів, а e і d – ключі шифрування і розшифрування відповідно. Ee – функція шифрування для довільного ключа $e \epsilon K$, така що:

$$E_{\mathrm{e}}\,(m) = \; c,$$

де $c \in C, C$ – множина шифртекстів,

$m \in M, M$ – множина повідомлень.

D_d – функція розшифрування, яка дозволяє знайти вихідне повідомлення m, коли відомо шифртекст c :

$$D_d(c) = m.$$

{Ee: $e \epsilon K$} – набір для шифрування, а $\{D_d: d \epsilon K\}$ – відповідний набір за допомогою якого можливо виконати розшифрування. Кожна пара (E, D) має наступну властивість: коли відомо E_e, неможливо виконати розв'язати рівняння $E_e(m) = c$, тобто для даного довільного шифртекста $c \epsilon C$, неможливо знайти повідомлення $m \epsilon M$. Тому можливо стверджувати, що за e неможливо визначити відповідний ключ розшифрування d. E_e є однобічною функцією, а d – лазівкою [5].

На рисунку 1.1 наведена схема передачі інформації, яка здійснюється між особою A для особи B. В реальному житті ці особи можуть бути представленими як фізичними особами, так і деякими компаніями або організаціями.

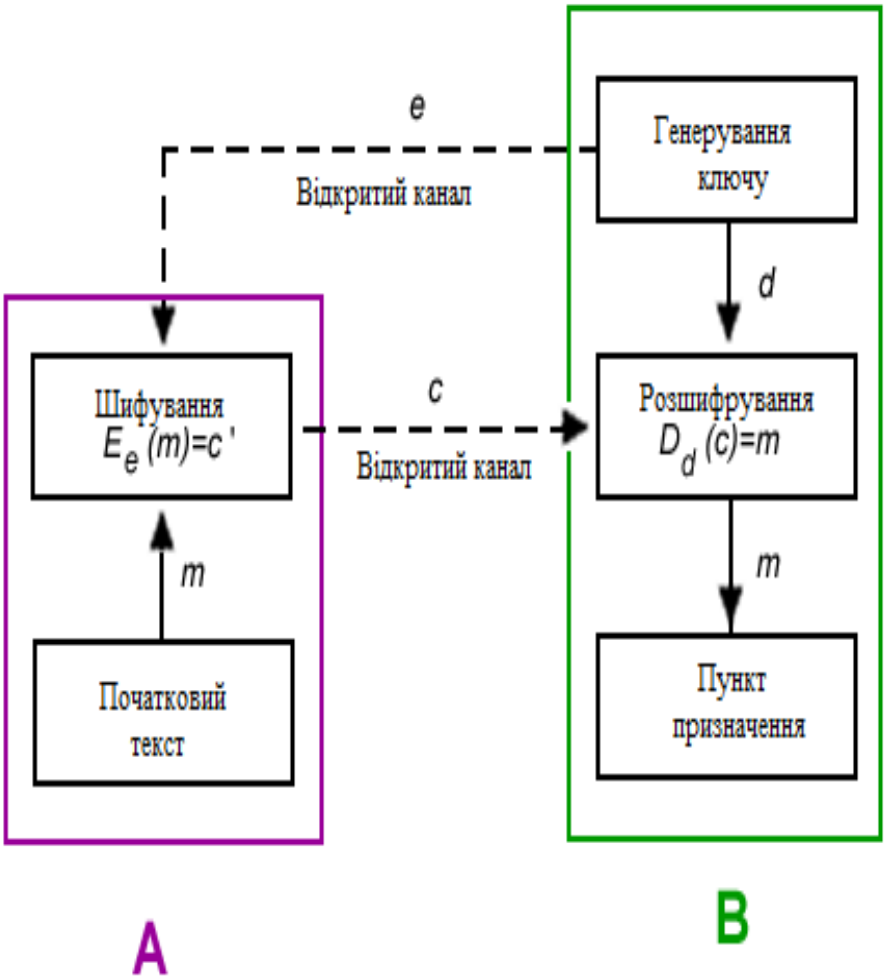

Рисунок 1.1 – Схема алгоритму передачі інформації між особами A і B

В реальному житті ці особи між якими здійснюється передача інформації можуть бути представленими як фізичними особами, так і деякими компаніями або організаціями.

1.2 Криптоаналіз алгоритмів з відкритим ключем

При здійсненні аналізу криптосистема з відкритим ключем являє собою ідеальну систему, що не потребує безпечного каналу для здійснення передачі ключа шифрування. Вважається, що два легальні користувачі А та В могли б здійснювати спілкування використовуючи відкритий канал, при цьому цім користувачам не потрібно зустрічатись, щоб передати один одному ключі [4]. Але існує третя сторона, яка виконує роль активного перехоплювача, що дозволяє їй здійснювати захоплення системи, щоб виконати розшифрування повідомлення, яке було призначене особі В, при цьому не виконуючи злам системи шифрування яка зображена на рисунку 1.2.

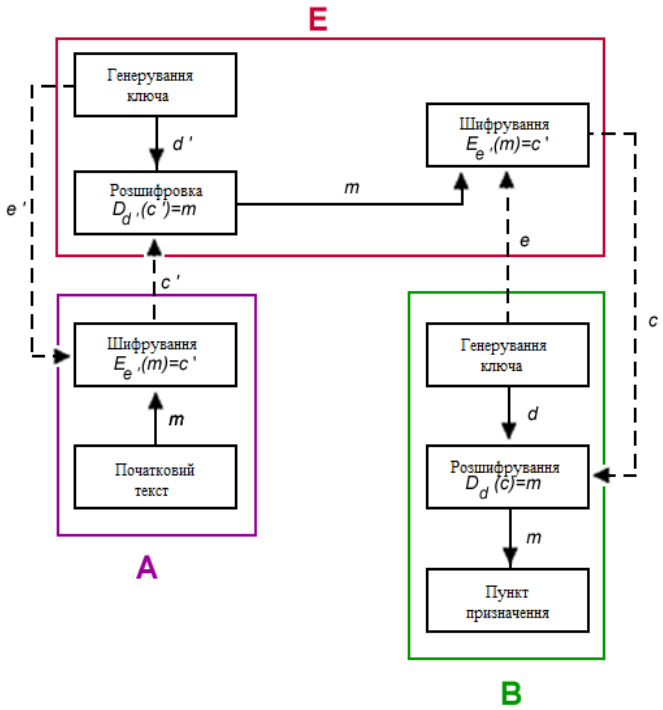

Рисунок 1.2 – Схема захвату третьою стороною системи передачі даних

Наведений вид атаки має назву «людина посередині». За наведеною схемою можливо стверджувати що ніхто з учасників не здогадується, що існує третя особа, для якої є можливим здійснення перехоплення повідомлення m, а також можлива заміна повідомлення на неправильне повідомлення m'. Очевидно, виникає необхідність аутентифікації відкритих ключів, для чого використовують сертифікати. Розподілене керування ключами в PGP можливо здійснити за допомогою поручителів [4].

Ще одна форма атаки – обчислення закритого ключа за відомим відкритим, її можно побачити на рисунку 1.3. Криптоаналітик знає алгоритм шифрування E_e, аналізуючи його, намагається знайти D_d. Цей процес спрощується, якщо криптоаналітик перехопив будь-які криптотексти, що послала особа А особі В.

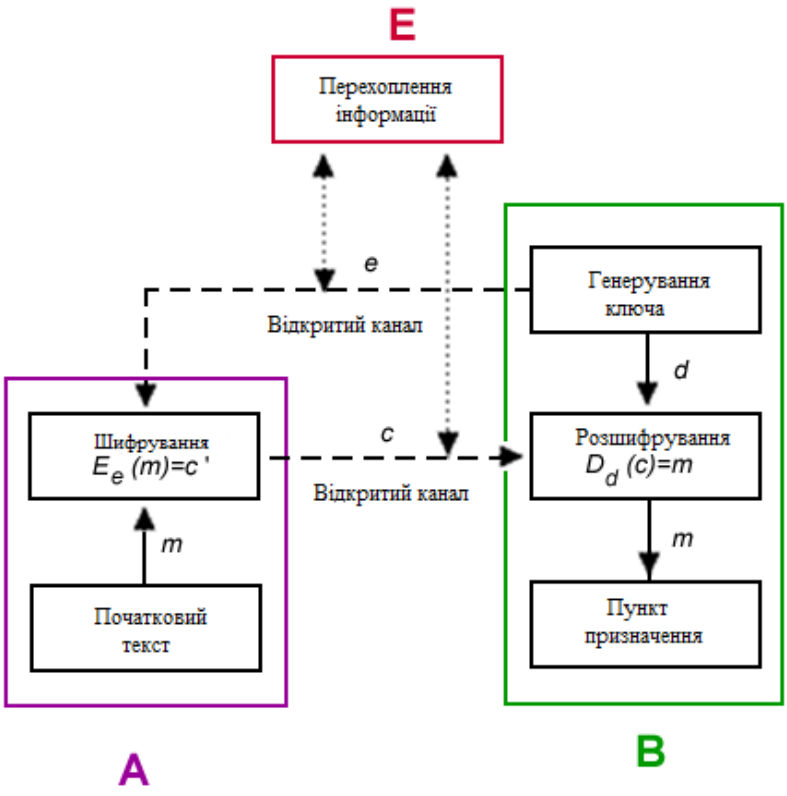

Рисунок 1.3 – Схема атаки добором закритого ключа при відомому відкритому ключі

Більшість криптосистем з відкритим ключем засновані на проблемі факторизації великих чисел. Наприклад, RSA використовує в якості відкритого ключа n добуток двох великих чисел. Складність злому такого алгоритму полягає в трудомісткості розкладання числа n на множники. Але цю задачу розв'язати реально. І з кожним роком процес розкладання стає усе швидшим.

Також задача розкладання потенційно можна розв'язати за допомогою алгоритму Шора при використанні досить потужного квантового комп'ютера.

Для багатьох методів несиметричного шифрування криптостійкість, отримана в результаті криптоаналіза, суттєво відрізняється від величин, що заявляються розроблювачами алгоритмів на підставі теоретичних оцінок. Тому в багатьох країнах питання застосування алгоритмів шифрування даних перебуває в полі законодавчого регулювання [4].

Розглянемо переваги асиметричного шифрування.

Перевага асиметричних шифрів перед симетричними шифрами полягає у відсутності необхідності попередньої передачі секретного ключа по надійному каналу.

У симетричній криптографії ключ тримається в секреті для обох сторін, а в асиметричній криптосистемі тільки один секретний ключ.

При симетричному шифруванні необхідно оновлювати ключ після кожного факту передачі, тоді як в асиметричних криптосистемах пари (E, D) можна не змінювати значний час.

У великих мережах число ключів в асиметричній криптосистемі значно менше, ніж у симетричній.

Але у асиметричного шифрування можливо визначити і недоліки.

Перевага алгоритму симетричного шифрування над несиметричним полягає в тому, що в перший відносно легко внести зміни.

Хоча повідомлення надійно шифруються, але «засвічуються» одержувач і відправник самим фактом пересилання шифрованого повідомлення.

Несиметричні алгоритми використовують більш довгі ключі, ніж симетричні.

Процес шифрування-розшифрування з використанням пари ключів проходить на два-три порядки повільніше, ніж шифрування-розшифрування того ж тексту симетричним алгоритмом.

У чистому вигляді асиметричні криптосистеми вимагають суттєво більших обчислювальних ресурсів, тому на практиці використовуються в комбінації з іншими алгоритмами.

Для ЕЦП повідомлення спочатку підлягає хешуванню, а за допомогою асиметричного ключа підписується лише невеликий результат хеш-функції.

Для шифрування алгоритми використовуються у формі гібридних криптосистем, де великі обсяги даних шифруються симетричним шифром на сеансовому ключі, а за допомогою асиметричного шифру передається тільки сам сеансовий ключ.

1.3 Електронний цифровий підпис

Електронний цифровий підпис – реквізит електронного документа, що дозволяє встановити відсутність викривлення інформації в електронному документі з моменту формування ЕЦП і перевірити приналежність підпису власникові сертифіката ключа ЕЦП. Значення реквізиту отримують у результаті криптографічного перетворення інформації з використанням закритого ключа ЕЦП.

Цифровий підпис призначений для аутентіфикації особи, що підписала електронний документ. Крім цього, використання цифрового підпису дозволяє здійснити наступне.

Контроль цілісності переданого документа: при будь-якій випадковій або навмисній зміні документа підпис стане недійсним, тому що він обчислений на підставі вихідного стану документа й відповідає лише йому.

Захист від змін (підробки) документа: гарантія виявлення підробки при контролі цілісності робить підроблення недоцільним у більшості випадків.

Неможливість відмови від авторства. Так як створити коректний підпис можна, лише знаючи закритий ключ, а він повинен бути відомий тільки власникові, то власник не може відмовитися від свого підпису під документом [1].

Доказове підтвердження авторства документа: Так як створити коректний підпис можна, лише знаючи закритий ключ, а він повинен бути відомий тільки власникові, то власник пари ключів може довести своє авторство підпису під документом. Залежно від деталей призначення документа можуть бути підписані такі поля, як «автор», «внесені зміни», «мітка часу» і т.д.

Електронний цифровий підпис – це по суті шифрування повідомлення алгоритмом з відкритим ключем. Текст, зашифрований секретним ключем, поєднується з вихідним повідомленням. Тоді перевірка підпису – розшифрування відкритим ключем, якщо текст, що буде отриманий, аналогічний вихідному тексту, – підпис вірний.

Використання хеш-функції дозволяє оптимізувати даний алгоритм. Відбувається шифрування не самого повідомлення, а значення хеш-функції, узятої від повідомлення. Даний метод забезпечує наступні переваги.

Зниження обчислювальної складності. Як правило, документ значно більше його хеша.

Сумісність. Більшість алгоритмів оперує з рядками біт даних, але деякі використовують інші уяви. Хеш-функцію можна використовувати для перетворення довільного вхідного тексту в підходящий формат.

Підвищення криптостійкості. Криптоаналітик не може, використовуючи відкритий ключ, підібрати підпис під повідомлення, а тільки під його хеш.

Цілісність. Без використання хеш-функції великий електронний документ у деяких схемах потрібно ділити на досить малі блоки для

застосування ЕЦП. При верифікації неможливо визначити, чи всі блоки отримані і чи у правильному порядку вони розташовані.

Асиметрична схема цифрового підпису охоплює три процеси. Генерація ключової пари. За допомогою алгоритму генерації ключа рівноімовірним образом з набору можливих закритих ключів вибирається закритий ключ, обчислюється відповідний йому відкритий ключ. Формування підпису. Для заданого електронного документа за допомогою закритого ключа обчислюється підпис. Перевірка (верифікація) підпису. Для даних документа і підпису за допомогою відкритого ключа визначається дійсність підпису. Для того, щоб використання цифрового підпису мало сенс, необхідне виконання двох умов. Верифікація підпису повинна проводитися відкритим ключем, який відповідає саме тому закритому ключу, який використовувався під час підписання. Без володіння закритим ключем повинне бути обчислювально складно створити легітимний цифровий підпис [6].

На рисунку 1.4 наведено схему, що відображає підпис даних та його перевірку на підписаних даних.

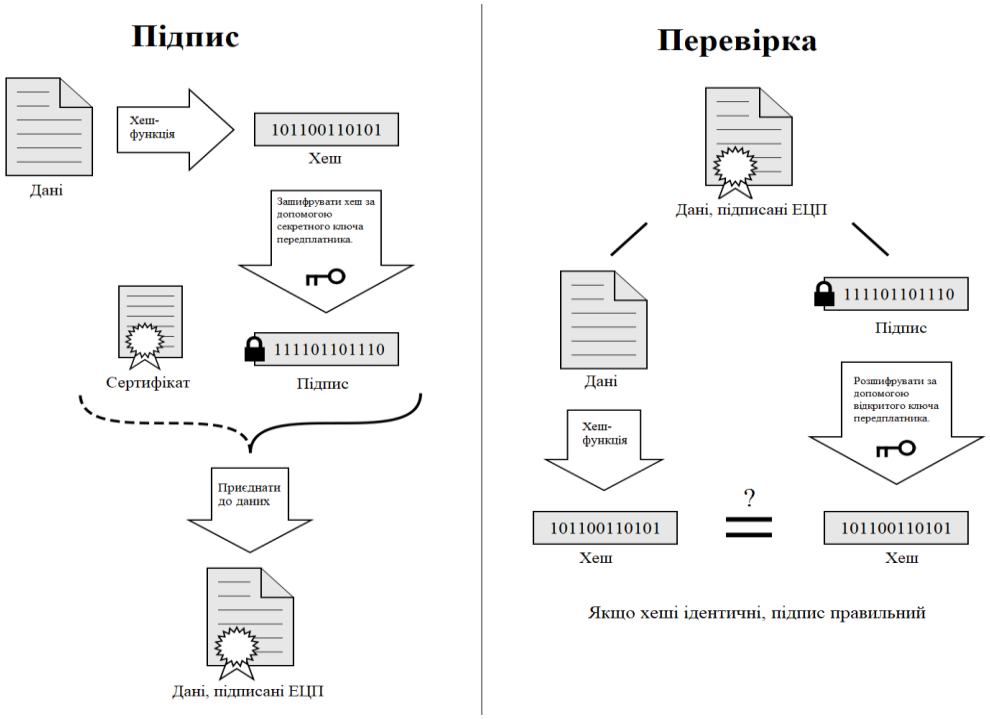

Рисунок 1.4 – Схема підпису даних і її перевірки на підписаних даних

Наведемо найбільш відомі алгоритми ЕЦП:

– американські стандарти електронного цифрового підпису: DSA, ECDSA;
– український стандарт електронного цифрового підпису: ДСТУ 4145-2002;
– стандарт PKCS#1 описує, зокрема, схему електронного цифрового підпису на основі алгоритму RSA;
– схема Шнора;
– elgamal.

Алгоритм цифрового підпису RSA.

Першою і найбільш відомою в усьому світі конкретною системою ЕЦП стала система RSA, математична схема якої була розроблена в 1977 р. у Массачусетському технологічному інституті США [4].

Спочатку необхідно обчислити пару ключів (секретний ключ і відкритий ключ). Для цього відправник (автор) електронних документів обчислює два векликі прості числа *P* и *Q*, потім знаходить їхній добуток

$$N = P * Q$$

і значення функції

$$\varphi(N) = (P - 1)(Q - 1)$$

Далі відправник обчислює число *E* з умов:

$$\mathrm{E} \leq \varphi(N), \mathrm{НОД}(\mathrm{E}, \varphi(N)) = 1$$

і число *D* з умов:

$$D < N, \mathrm{E} * D \equiv 1 \ (\mathrm{mod}\ \varphi(N)).$$

Пари чисел (E, N) є відкритим ключем. Цю пару чисел автор передає партнерам по листуванню для перевірки його цифрових підписів. Число D зберігається автором як секретний ключ для підписування [4].

Узагальнена схема формування і перевірки цифрового підпису RSA показана на рисунку 1.5.

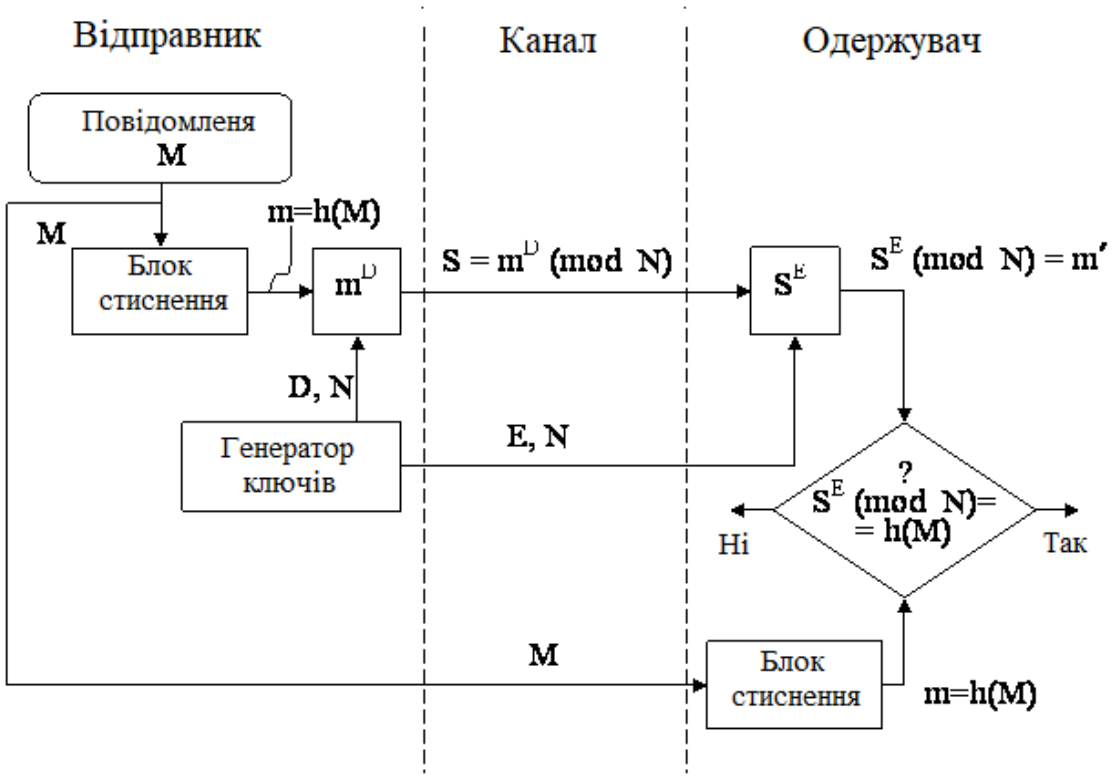

Рисунок 1.5 – Схема алгоритму цифрового підпису RSA

Припустимо, що відправник прагне підписати повідомлення M перед його відправленням. Спочатку повідомлення M (блок інформації, файл, таблиця) стискають за допомогою хеш-функції $h(\cdot)$ в ціле число m [5]:

$$m = h(M).$$

Потім обчислюють цифровий підпис S під електронним документом M, використовуючи хеш-значення m і секретний ключ D:

$$S = m^d \ (mod\ N).$$

Пара (M, S) передається партнерові-одержувачеві як електронний документ M, підписаний цифровий підписом S, причому підпис S сформований власником секретного ключа D.

Після приймання пари (M, S) одержувач обчислює хеш-значення повідомлення M двома різними способами. Насамперед він відновлює хеш-значення m', застосовуючи криптографічне перетворення підпису S з використанням відкритого ключа E [5]:

$$m' = S^E (mod\ N).$$

Крім того, він знаходить результат хешування прийнятого повідомлення M за допомогою такої ж хеш-функції $h(\cdot)$:

$$m = h(M).$$

Якщо виконується рівність обчислених значень, тобто

$$S^E \ (mod\ N) = h\ (M),$$

то одержувач визнає пару (M, S) справжньою. Доведене, що тільки власник секретного ключа D може сформувати цифровий підпис S документу M, а визначити секретне число D по відкритому числу E не легше, ніж розкласти модуль N на множники.

Крім того, можна строго математично довести, що результат перевірки цифрового підпису S буде позитивним тільки в тому випадку, якщо при обчисленні S був використаний секретний ключ D, відповідний до відкритого ключа E. Тому відкритий ключ E іноді називають «ідентифікатором», що підписує.

Розглянемо недоліки алгоритму цифрового підпису RSA.

При обчисленні модуля *N*, ключів *E* і *D* для системи цифрового підпису RSA необхідно перевіряти велику кількість додаткових умов, що зробити практично важко. Невиконання кожної із цих умов уможливлює фальсифікацію цифрового підпису з боку того, хто виявить таке невиконання. Під час підписання важливих документів не можна допускати таку можливість навіть теоретично.

Для забезпечення криптостійкості цифрового підпису RSA стосовно спроб фальсифікації на рівні, наприклад, національного стандарту США шифрування інформації (алгоритм DES), тобто 1018, необхідно використовувати при обчисленнях *N*, *D* і *E* цілі числа не менш 2^{512} (або близько 10154) кожне, що вимагає великих обчислювальних витрат, що перевищують на 20...30% обчислювальні витрати інших алгоритмів цифрового підпису при збереженні того ж рівня криптостійкості [5].

Цифровий підпис RSA вразливий до так званої мультиплікативної атаки. Інакше кажучи, алгоритм цифрового підпису RSA дозволяє зловмисникові без знання секретного ключа D сформувати підписи під тими документами, у яких результат хешування можна обчислити як добуток результатів хешування вже підписаних документів.

1.4 Основні засади побудови хеш-функцій

Хешування (hashing) передбачає виконання перетворення вхідного масиву даних, що має довільну довжину у вихідний бітовий рядок, що передбачає наявність фіксованої довжини. Такі перетворення також називаються хеш-функціями або функціями згортки, а їх результати називають хешем, хеш-кодом або дайджестом повідомлення (message digest).

Хешування застосовується для порівняння даних: якщо у двох масивів хеш-коди різні, масиви гарантовано різняться; якщо однакові – масиви, швидше за все, однакові. У загальному випадку однозначної відповідності між

вихідними даними і хеш-кодом немає в силу того, що кількість значень хеш-функцій менше, ніж варіантів вхідного масиву; існує безліч масивів, що дають однакові хеш-коди – так звані колізії. Імовірність виникнення колізій відіграє немаловажну роль в оцінці якості хеш-функцій [5].

Серед безлічі існуючих хеш-функцій прийнято виділяти криптографічно стійкі, застосовувані в криптографії. Для того, щоб хеш-функція H вважалася криптографічно стійкою, вона повинна задовольняти трьом основним вимогам, на яких заснована більшість застосувань хеш-функцій у криптографії:

- необоротність: для заданого значення хеш-функції m повинне бути обчислювально нездійсненно знайти блок даних X, для якого $H(X) = m$;
- стійкість до колізій першого роду: для заданого повідомлення M повинне бути обчислювально нездійсненно підібрати інше повідомлення N, для якого $H(N) = H(M)$.

Стійкість до колізій другого роду: повинне бути обчислювально нездійсненне підібрати пари повідомлень, що мають однаковий хеш. Дані вимоги не є незалежними. Оборотна функція нестійка до колізій першого і другого роду. Функція, нестійка до колізій першого роду, нестійка до колізій другого роду; зворотне невірно. Слід зазначити, що не доведене існування необоротних хеш-функцій, для яких обчислення будь-якого прообразу заданого значення хеш-функції теоретично неможливо. Звичайне знаходження зворотного значення є лише обчислювально складною задачею.

Атака що називається «дні народження» дозволяє знаходити колізії для хеш-функції з довжиною значень n битів у середньому за приблизно $2^{n/2}$ обчислень хеш-функції. Тому n-бітна хеш-функція вважається криптостійкою, якщо обчислювальна складність знаходження колізій для неї близька до $2^{n/2}$.

Для криптографічних хеш-функцій також важливо, щоб при наймен-шій зміні аргументу значення функції сильно змінювалося (лавинний ефект). Зокрема, значення хеша не повинне давати витоку інформації навіть про

окремі біти аргументу. Ця вимога є запорукою криптостійкості алгоритмів хешування, хешуючих користувацький пароль для одержання ключа.

Хеш-функції: Adler-32, CRC, Hashcart, HAVAL, Keccak, LM-хеш, MD2, MD4, MD5, MD6, N-Hash, PJW-32, RIPEMD-128, RIPEMD-160, RIPEMD-256, RIPEMD-320,SHA-1, SHA-2 (SHA-224, SHA-256, SHA-384 і SHA-512), Skein, Snefru, Tiger, TTH, Whirlpool, ГОСТ Р 34.11-94.

Основні засади побудови хеш-функцій. Загальноприйнятим принципом побудови хеш-функцій є ітеративна послідовна схема. За цією методикою ядром алгоритму є перетворення k біт в n біт. Величина n – розрядність результату хеш-функції, а k – довільне число, більше n. Базове перетворення повинне мати всі властивості хеш-функції, тобто необоротність і неможливість інваріантної зміни вхідних даних [4].

Хешування проводиться за допомогою проміжної допоміжної змінної розрядністю в n біт. У якості її початкового значення вибирається довільне відоме всім сторонам значення, наприклад, 0. Вхідні дані розбиваються на блоки по $(k-n)$ біт. На кожній ітерації хешування із значенням проміжної величини, отриманої на попередній ітерації, поєднується чергова $(k-n)$ – бітна порція вхідних даних, і над k-бітним блоком, що вийшов, проводиться базове перетворення. У результаті весь вхідний текст виявляється «перемішаним» з початковим значенням допоміжної величини.

Завдяки характеру перетворення базову функцію часто називають стискаючою. Значення допоміжної величини після фінальної ітерації надходить на вихід хеш-функції який зображен на рисунку 1.6. Іноді з значенням, що вийшло, роблять додаткові перетворення. Але в тому випадку, якщо стискаюча функція спроектована з достатнім степенем стійкості, ці перетворення зайві [7].

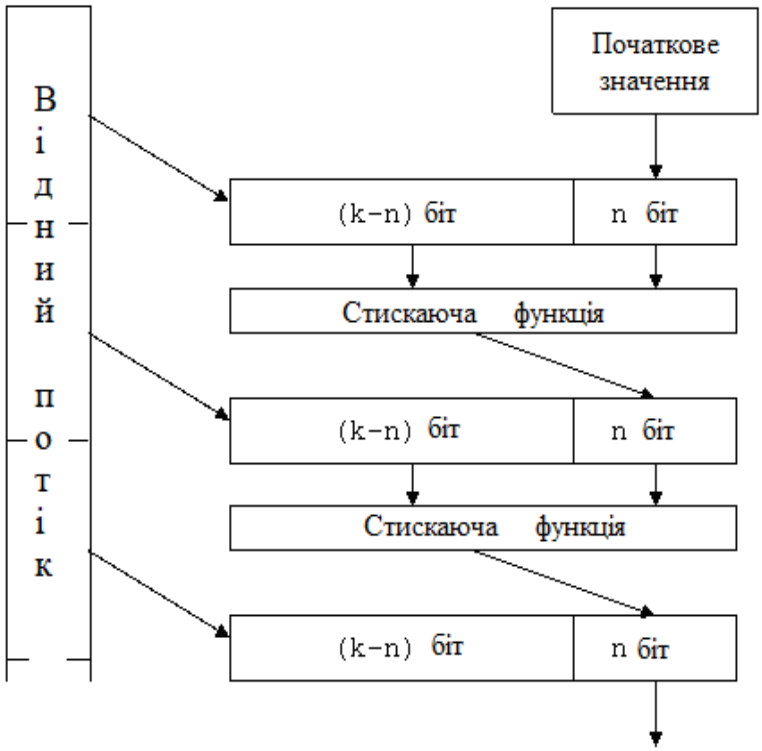

Рисунок 1.6 – Схема функціонування ітеративної хеш-функції

При проектуванні хеш-функції за ітеративною схемою виникають два взаємозалежні питання: що робити з даними, не кратними числу $(k-n)$, і як додавати в хеш-суму довжину документа, якщо це потрібно. Є два варіанти розв'язку цих питань. У першому варіанті в початок документа перед хешуванням додається поле фіксованої довжини (32 біта), у якому у двійковому вигляді записується вихідна довжина тексту. Потім об'єднаний блок даних доповнюється нулями до найближчого кратного $(k-n)$ біт розміру. У другому варіанті документ доповнюється праворуч одним битом «1», а потім до кратного $(k-n)$ біт розміру бітами «0» [7]. У цьому варіанті необхідність у полі довжини відпадає – ніякі два різні документи після вирівнювання по границі не стануть однаковими. Крім більш популярних однопрохідних алгоритмів хешування існують і багатопрохідні алгоритми. У цьому випадку вхідний блок даних на етапі розширення неодноразово повторюється, а вже потім доповнюється до найближчої границі.

1.5 Застосування односпрямованих хеш-функцій на основі симетричних блокових алгоритмів

Односпрямовану хеш-функцію можна побудувати, використовуючи симетричний блоковий алгоритм. Більш очевидний підхід є в тому, коли повідомлення *M* шифрується за допомогою блокового алгоритму в режимі CBC або CFB з використанням фіксованого ключа і деякого вектора ініціалізації IV. Можна розглядати останній блок шифртексту як повідомлення *M* хеш-значення. З таким підходом не завжди можливо побудувати безпечну хеш-функцію, але завжди можна одержати код аутентифікації повідомлення Message Authentication Code (MAC).

Куди більш безпечний варіант самої хеш-функції можна отримати в якості ключа, використовуючи блок повідомлення, попереднє хеш-значення – як вхід, а поточне хеш-значення – як вихід. Реальні хеш-функції створюють куди більш складними. Довжина хеш-значення куди більше з довжиною блоку, а довжина блоку визначається довжиною ключа. Оскільки більшість блокових алгоритмів є 64-бітовими, деякі схеми хешування проектують так, щоб хеш-значення мало довжину, яка дорівнює подвійний довжині блоку [7].

Одержувана хеш-функція коректна тоді, коли безпека схеми хешування базується на безпеці блокового алгоритму, яка є в її основі. Схема хешування, довжина якої хеш-значення дорівнює довжині блоку, показана на рисунку 1.7.

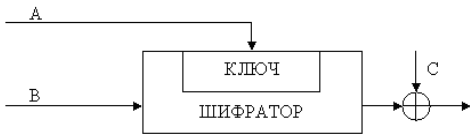

Рисунок 1.7 – Схема алгоритму формування хеш-функції

Робота хеш-функції описується виразами [4]:

$$H_0 = I_H,$$
$$H_i = E_A(B) \oplus \mathrm{C},$$

де $\oplus$ – додавання за модулем 2 (АБО);

$I_н$ – деяке випадкове початкове значення;

A, *B*, *C* можуть приймати значення $\mathrm{M}_i, \mathrm{H}_i - 1, (\mathrm{M}_i \oplus \mathrm{H}_i - 1)$ або бути константами.

Повідомлення M розбивається на блоки M_i прийнятої довжини, які обробляються по черзі. За набагато меншу кількість операцій над вхідними даними, можна забезпечити стійкість щодо двох основних вимог до хеш-функції. Інші 12 схем безпечного хешування, у яких довжина хеш-значення дорівнює довжині блоку, перераховані в таблиці 1.1.

Таблиця 1.1 – Схеми безпечного хешування

Номер схеми	Функція хешування
1	$\mathrm{H}_i = \mathrm{E}_{Hi-1}(\mathrm{M}_i) \oplus \mathrm{M}_i$
2	$\mathrm{H}_i = \mathrm{E}_{Hi-1}(\mathrm{M}_i \oplus \mathrm{H}_{i-1}) \oplus \mathrm{M}_i \oplus \mathrm{H}_{i-1}$
3	$\mathrm{H}_i = E_{Hi-1}(\mathrm{M}_i) \oplus \mathrm{M}_i \oplus \mathrm{H}_{i-1}$
4	$\mathrm{H}_i = \mathrm{E}_{Hi-1}(\mathrm{M}_i \oplus \mathrm{H}_{i-1}) \oplus \mathrm{M}_i$
5	$\mathrm{H}_i = \mathrm{E}_{Mi}(\mathrm{H}_{i-1}) \oplus \mathrm{H}_{i-1}$
6	$\mathrm{H}_i = \mathrm{E}_{Mi}(\mathrm{M}_i \oplus \mathrm{H}_{i-1}) \oplus \mathrm{M}_i \oplus \mathrm{H}_{i-1}$
7	$\mathrm{H}_i = \mathrm{E}_{Mi}(\mathrm{H}_{i-1}) \oplus \mathrm{M}_i \oplus \mathrm{H}_{i-1}$
8	$\mathrm{H}_i = E_{Mi}(\mathrm{M}_i \oplus \mathrm{H}_{i-1}) \oplus \mathrm{H}_{i-1}$
9	$\mathrm{H}_i = \mathrm{E}_{Mi} \oplus H_{i-1}(\mathrm{M}_i) \oplus \mathrm{M}_i$
10	$\mathrm{H}_i = \mathrm{E}_{Mi} \oplus H_{i-1}(\mathrm{H}_{i-1}) \oplus \mathrm{H}_{i-1}$
11	$\mathrm{H}_i = \mathrm{E}_{Mi} \oplus H_{i-1}(M_i) \oplus \mathrm{H}_{i-1}$
12	$\mathrm{H}_i = \mathrm{E}_{Mi} \oplus H_{i-1}(\mathrm{H}_{i-1}) \oplus \mathrm{M}_i$

Алгоритм необхідно проектувати спеціально, виходячи з двох вимог: стійкість та швидкість [7]. Перші чотири схеми хешування, що є безпечними при всіх атаках, наведені на рисинку 1.8.

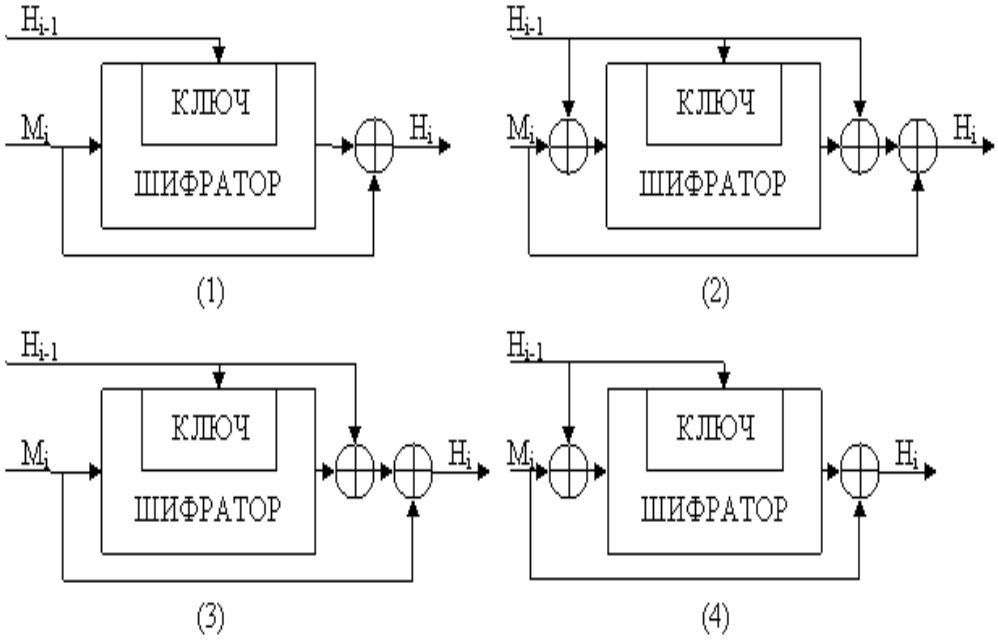

Рисунок 1.8 – Схеми алгоритму безпечного хешування

1.6 Дослідження алгоритму безпечного хешування SHA

Алгоритм безпечного хешування SHA розроблений НІСТ і АНБ США в рамках стандарту безпечного хешування SHS (Secure Hash Standard) в 1992 р. Алгоритм хешування SHA призначений для використання разом з алгоритмом цифрового підпису DSA.

При введенні повідомлення *M* довільної довжини менше 264 біт алгоритм SHA виробляє 160-бітове вихідне повідомлення, яке називається дайджестом повідомлення MD (Message Digest). Потім цей дайджест повідомлення використовується як вхід алгоритму DSA, який обчислює цифровий підпис повідомлення *M*. Формування цифрового підпису для дайджесту повідомлення, а не для самого повідомлення, підвищує

ефективність процесу підписання, оскільки дайджест повідомлення звичайно набагато коротше самого повідомлення.

Такий же дайджест повідомлення повинен обчислюватися користувачем, що перевіряє отриманий підпис, при цьому в якості входу в алгоритм SHA використовується отримане повідомлення *M*.

Алгоритм хешування SHA є безпечним, бо він спроектований так, щоб було обчислювально неможливо відновити повідомлення, відповідне до даного дайджесту, а також знайти два різні повідомлення, які дадуть однаковий дайджест. Будь-яка зміна повідомлення при передачі з дуже великою ймовірністю викличе зміну дайджесту, і прийнятий цифровий підпис не пройде перевірку [5].

Розглянемо роботу алгоритму хешування SHA. Вихідне повідомлення *M* доповнюють щоб воно стало кратним 512 бітам. Додаткове додавання повідомлення виконується таким чином: спочатку додається одиниця, потім випливають нулі, які необхідні для одержання повідомлення, яке на 64 біта коротше, та кратне 512, а потім додають 64-бітову довжинц вихідного повідомлення.

Ініціалізується п'ять 32-бітових змінних у вигляді:

$$\mathrm{A} = 0\mathrm{x}67452301$$

$$\mathrm{B} = 0\mathrm{xEFCDAB89}$$

$$\mathrm{C} = 0\mathrm{x98BADCFE}$$

$$\mathrm{D} = 0\mathrm{x10325476}$$

$$\mathrm{E} = 0\mathrm{xC3D2E1F0}$$

Далі наведено головний цикл алгоритму. У ньому обробляється по 512 біт повідомлення по черзі для всіх 512-бітових блоків, що є у повідомленні. Перші п'ять змінних *A*, *B*, *C*, *D*, *E* копіюються в інші змінні *a*, *b*, *c*, *d*, *e* [5]:

$$\mathrm{a} = \mathrm{A}, b = \mathrm{B}, \mathrm{c} = \mathrm{C}, d = D, \mathrm{e} = \mathrm{E}.$$

Головний цикл містить чотири цикли по 20 операцій кожний. Кожна операція реалізує нелінійну функцію від трьох з п'яти змінних *a*, *b*, *c*, *d*, *e*, а потім робить зсув і додавання.

Алгоритм SHA має наступний набір нелінійних функцій [5]:

$f_t\,(\mathrm{X}, Y, Z) = (X \wedge Y) \vee ((X) \wedge Z)$ для $t = 0\ldots19$,

$f_t\,(\mathrm{X}, Y, Z) = \mathrm{X} \oplus Y \oplus Z$ для $t = 20\ldots39$,

$f_t\,(\mathrm{X}, Y, Z) = (X \wedge Y) \vee (X \wedge Z) \vee (Y \wedge Z)$ для $t = 40\ldots59$,

$f_t\,(\mathrm{X}, Y, Z) = \mathrm{X} \oplus Y \oplus Z$ для $t = 60\ldots79$,

де t – номер операції.

В алгоритмі використовуються також чотири константи:

$\mathrm{K_t}$ = 0x5A827999 для $t = 0\ldots19$,

$\mathrm{K_t}$ = 0x6ED9EBA1 для $t = 20\ldots39$,

$\mathrm{K_t}$ = 0x8F1BBCDC для $t = 40\ldots59$,

$\mathrm{K_t}$ = 0xCA62C1D6 для $t = 60\ldots79$.

Блок повідомлення перетвориться із шістнадцяти 32-бітових слів $(\mathrm{M_0}\ldots\mathrm{M_{15}})$ у вісімдесят 32-бітових слів $(W_0\ldots W_{79})$ за допомогою наступного алгоритму [5]:

$W_t = \mathrm{M}_t$ для $t = 0\ldots15$,

$W_t = (W_{t-3} \oplus W_{t-8} \oplus W_{t-14} \oplus W_{t-16}) <<< 1$ для $t = 16\ldots79$,

де t – номер операції,

W_t – t-й субблок розширеного повідомлення,

$<<< S$ – циклічний зсув вліво на S біт.

З урахуванням введених позначень головний цикл із вісімдесяти операцій можна описати так:

цикл по t от 0 до 79

$\mathrm{TEMP} = (\mathrm{a} <<< 5) + f_t\,(b, c, d) + \mathrm{e} + W_t + \mathrm{K}_t$

$\mathrm{e} = d$

$d = \mathrm{c}$

$\mathrm{c} = (b <<< 30)$

$b = \mathrm{a}$

$\mathrm{a} = \mathrm{TEMP}$

конец_цикла

Схема виконання однієї операції показана на рис. 1.9.

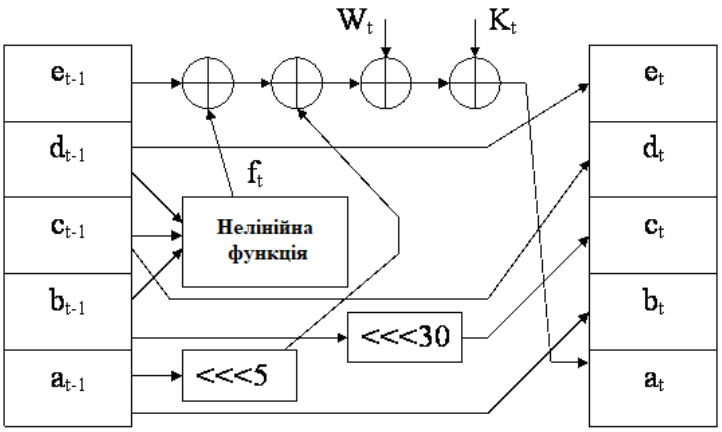

Рисунок 1.9 – Схема алгоритму SHA для виконання однієї операції

Після закінчення головного циклу значення *a*, *b*, *c*, *d*, *e* складаються з *A*, *B*, *C*, *D*, *E* відповідно, і алгоритм приступає до обробки наступного 512-бітового блоку даних. Остаточний вихід формується у вигляді конкатенації значень *A*, *B*, *C*, *D*, *E*.

Відмінності SHA від MD5 полягають у наступному:

SHA видає 160-бітове хеш-значення, тому він більш стійкий до атак повного перебору і атакам «дня народження», ніж MD5, що формує 128-бітові хеш-значения. Функція стиску SHA складається з 80 кроків, а не з 64, як в MD5. Розширення вхідних даних проводиться не простим їхнім повторення в іншому порядку, а рекуррентною формулою. Ускладнений процес перемішування.

1.7 Керування відкритими ключами

Важливою проблемою всієї криптографії з відкритим ключем, у тому числі і систем ЕЦП, полягає в керуванні відкритими ключами. Відкритий ключ доступний будь-якому користувачеві, то необхідний механізм перевірки того, що цей ключ належить саме своєму власникові. Необхідно забезпечити доступ будь-якого користувача до відкритого ключа будь-якого іншого користувача,

захистити ці ключі від підміни зловмисником, а також організувати відкликання ключа у випадку його компрометації [4].

За допомогою сертифікатів вирішується задача захисту ключів від підміни. Сертифікат дозволяє засвідчити пов'язані в ньому дані про власника і його відкритий ключ підписом будь-якої довіреної особи. Існують системи сертифікатів двох типів: централізовані і децентралізовані. У децентралізованих системах шляхом перехресного підписування сертифікатів знайомих і довірених людей кожним користувачем будується мережа довіри. У централізованих системах сертифікатів використовуються центри сертифікації, підтримувані довіреними організаціями.

Центр сертифікації формує закритий ключ і власний сертифікат, формує сертифікати кінцевих користувачів і засвідчує їхню автентичність своїм цифровим підписом. Також центр веде бази виданих і відкликаних сертифікатів і проводить відкликання колишніх і компрометованих сертифікатів. Звернувшись у сертифікаційний центр, можна отримати власний сертифікат відкритого ключа, а також сертифікат іншого користувача і довідатися, які ключі відкликані.

Зберігання закритого ключа. Закритий ключ є уразливим компонентом усієї криптосистеми цифрового підпису. Зловмисник, який вкрав закритий ключ користувача, може створити дійсний цифровий підпис будь-якого електронного документа від імені цього користувача. Особливу увагу потрібно приділяти зберіганню закритого ключа. Користувач може зберігати закритий ключ на своєму комп'ютері, захистивши його паролем. Однак такий спосіб зберігання має недоліки, захищеність ключа повністю залежить від захисту комп'ютера, і користувач може підписати документи тільки на цьому комп'ютері [4].

В наш час існують пристрої зберігання закритого ключа такі як: дискети, смарт-карти, USB, таблетки Touch-Memory.

Один з самих захищених способів зберігання закритого ключа – на смарт-карті. Для того, щоб використовувати смарт-карту, користувачеві

необхідно її мати, і також вводити Pin-код, виходить двухфакторна аутентифікація. Після цього документ, що підписується, або його хеш передається в карту, її процесор здійснює підписування хеша і передає підпис назад. У процесі формування підпису не відбувається копіювання закритого ключа, тому увесь час існує тільки єдина копія ключа.

2 АНАЛІЗ ІНФРАСТРУКТУРИ ВІДКРИТИХ КЛЮЧІВ

Інфраструктура відкритих ключів (PKI) – технологія аутентифікації за допомогою відкритих ключів.

Основні механізми інфраструктура відкритих ключів [5]:

- установлення довіри (у рамках заданої моделі довіри);
- система іменування суб'єктів, що забезпечує унікальність імені в рамках системи;
- зв'язок імені суб'єкта і пари ключів (відкритий і закритий) з підтвердженням цього зв'язку засобами центру, що засвідчує, якому довіряє суб'єкт, що перевіряє правильність зв'язку.

Інфраструктура відкритих ключів не реалізує авторизацію, довіру, іменування суб'єктів криптографії, захист інформації або ліній зв'язку, але може використовуватися як одна зі складових при їхній реалізації.

Інфраструктура відкритих ключів також може використовуватися для реалізації конфіденційності, цілісності переданих даних, неспростовності (або апелірується – non-repudiation).

2.1 Об'єкти інфраструктури відкритих ключів

Інфраструктура відкритих ключів реалізується в моделі клієнт-сервер, тобто перевірка будь-якої інформації, надаваною інфраструктурою може відбуватися тільки з ініціативи клієнта.

Основні компоненти інфраструктури відкритих ключів.

Центр, що забезпечує зв'язок ідентичності суб'єкта криптографії (звичайне ім'я і будь-яка додаткова інформація) і його відкритого ключа. У дуже малих інфраструктур відкритих ключів центр, що засвідчує, може бути відсутнім, у середніх і великих він обов'язково присутній. Сертифікат відкритого ключа – це підписана засвідчувальним центром структура даних ідентичності суб'єкта криптографії і його відкритого ключа. Для підпису

сертифіката засвідчувальний центр випускає сертифікат, який підписує ключем, зазначеним у сертифікаті. Такий сертифікат називається самопідписаним (самовиданим) [7].

Реєстраційний центр – компонент системи, що перевіряє правильність наданих суб'єктом криптографії даних для формування запису про ідентичність. Засвідчувальний центр цієї інформації, довіряє реєстраційному центру перевірку. Реєстраційний центр, виписує сертифікат тільки тоді, коли перевіривши правильність інформації, підписує її своїм ключем і передає засвідчувальному центру, який, перевіривши ключ реєстраційного центру. Один реєстраційний центр може працювати з декількома засвідчувальними центрами, тобто входити до декількох інфраструктур відкритих ключів. Виходить що один засвідчувальний центр може працювати з декількома реєстраційними центрами.

Репозиторій – сховище випущених ЦС сертифікатів. У Законі України «Про електронний цифровий підпис». Він називається реєстр сертифікатів ключів підписів.

Архів сертифікатів – сховище всіх виданих будь-коли сертифікатів, які включають сертифікати зі строком дії, який завершен. Архів використовується для перевірки дійсності електронного підпису, яким засвідчувалися документи.

Центр реєстрації – опціональна компонента інфраструктури, призначена для реєстрації кінцевих користувачів і забезпечення їх взаємодії із центром сертифікації.

Мережний довідник – опціональна компонента інфраструктури, що містить сертифікати і списки відкликаних сертифікатів, що служить для цілей поширення цих об'єктів серед користувачів.

Основні задачі системи інформаційної безпеки, які вирішує інфраструктура керування відкритими ключами, які наведені нижче [8]:

- забезпечення конфіденційності інформації;
- забезпечення цілісності інформації;

- забезпечення аутентифікації користувачів і ресурсів, до яких звертаються користувачі;
- забезпечення можливості підтвердження зроблених користувачами дій з інформацією (незреченність).

2.2 Аналіз структури сертифікату відкритого ключа

Сертифікат відкритого ключа: сертифікат ЕЦП, сертифікат ключа підпису, являє собою цифровий або паперовий документ, що підтверджує відповідність між відкритим ключем і інформацією, що ідентифікує власника ключа. Воно містить інформацію про власника ключа, відомості про відкритий ключ, його призначення і області застосування, назва центра сертифікації і так далі.

Відкритий ключ може бути використаний для організації захищеного каналу зв'язку із власником двома способами:

- для перевірки подпису власника (аутентифікація);
- для шифрування даних, що йому посилають (конфіденціальність).

Існує дві моделі організації інфраструктури сертифікатів: централізована (PKI) і децентралізована.

Розглянемо принцип роботи. Сертифікати, використовуються для обміну зашифрованими даними в великих мережах. Криптосистема с відкритим ключем вирішує проблему обміну секретними ключами між учасниками безпечного обміну, однак не вирішує проблему довіри до відкритих ключів. Ідея сертифіката – наявність третьої сторони, якій довіряють інші сторони інформаційного обміну. Передбачається, що таких третіх сторін не багато, і їхні відкриті ключі всім відомі яким-небудь способом. Таким чином, підробка відкритого ключа третьою стороною легко виявляється [8].

Перелік обов'язкових і необов'язкових полів, які можуть бути присутніми у сертифікаті, визначається стандартом його формату (наприклад, X.509). Як правило, сертифікат містить у собі наступні поля:

- ім'я власника сертифіката (ім'я користувача, якому належить сертифікат);
- один або кілька відкритих ключів власника сертифіката;
- ім'я засвідчувального центру;
- серійний номер сертифіката, привласнений засвідчувальним центром;
- термін дії сертифіката (дата початку дії і дата закінчення дії);
- інформація про використані криптографічні алгоритми;
- електронний цифровий підпис, сгенерований з використанням секретного ключа засвідчувального центру (підписується результат хешування усієї інформації, що зберігається в сертифікаті).

2.3 Організація засвідчувального центру

Центр сертифікації або засвідчувальний центр – організація або підрозділ організації, яка випускає сертифікати ключів електронного цифрового підпису. Це компонент глобальної служби каталогів, відповідальний за керування криптографічними ключами користувачів. Відкриті ключі, а також будь-яка інша інформація про користувачів зберігається засвідчувальними центрами у вигляді цифрових сертифікатів.

Центр сертифікації сам формує власний секретний ключ. Сертифікат, який містить відкритий ключ даного центру і підписаний їм самим. В майбутньому, після створення і підпису власного сертифіката, центр веде базу даних виданих сертифікатів і списків відкликаних сертифікатів. Для нормальної роботи із центром сертифікації його власний сертифікат потрібно мати доданий у список довірених на тому комп'ютері, де його потрібно використовувати. Після включення в такий список, будь-який сертифікат,

виданий довіреним центром, який є достовірним, а його власник – гідним довіри. Засвідчувальний центр забезпечує наступні дії [8]:

- виготовляє сертифікати ключів підписів;
- створює ключі електронних цифрових підписів по обліку учасників інформаційної системи з гарантією збереження в таємниці закритого ключа електронного цифрового підпису;
- припиняє і відновляє дію сертифікатів ключів підписів, а також анулює їх;
- веде реєстр сертифікатів ключів підписів, забезпечує його актуальність і можливість вільного доступу до нього учасників інформаційних систем;
- перевіряє унікальність відкритих ключів електронних цифрових підписів у реєстрі сертифікатів ключів підписів і архіві засвідчу вального центру;
- видає сертифікати ключів підписів у формі документів на паперових носіях або у формі електронних документів з інформацією про їхню дію;
- здійснює по обліку користувачів сертифікатів ключів підписів підтвердження справжності електронного цифрового підпису в електронному документі у відношенні виданих їм сертифікатів ключів підписів;
- може надавати учасникам інформаційних систем інші пов'язані з використанням електронних цифрових підписів послуги.

2.4 Архітектура інфраструктур відкритих ключів

В основному виділяють п’ять видів архітектурних інфраструктур відкритих ключів [7]:

- проста інфраструктура відкритих ключів (одиночний ЦС);
- ієрархічна інфраструктура відкритих ключів;

– мережна інфраструктура відкритих ключів;

– крос-сертифіковані корпоративні інфраструктура відкритих ключів;

– архітектура мостового ЦС.

В основному інфраструктура відкритих ключів діляться на різні архітектури по наступних ознаках:

– кількість ЦС (а також кількість ЦС, які довіряють один одному);

– складність перевірки шляху сертифікації;

– наслідок видачі зловмисником себе за ЦС.

Найпростіша з архітектур – архітектура одиночного ЦС. У цьому випадку всі користувачі довіряють одному ЦС і листуються між собою. У даній архітектурі, якщо зловмисник видасть себе за ЦС, необхідно просто перевипустити всі виписані сертифікати і продовжити нормальну роботу.

Ієрархічна інфраструктура відкритих ключів.

Ієрархічна структура – архітектура інфраструктури відкритих ключів, що найчастіше зустрічається. У цьому випадку головним всієї структури стоїть один Головний ЦС, якому все довіряють і йому підкоряються нижчестоящі ЦС. Крім цього головного ЦС у структурі присутній ще один ЦС, який підкоряється вищому, якому у свою чергу приписані будь-які нижчестоящі або користувачі ЦС. Приватний приклад ієрархічної інфраструктури відкритих ключів являє собою корпоративна інфраструктура відкритих ключів. Якщо в нас є одна велика фірма, у якої в підпорядкуванні безліч філіалів по всій країні. У головному будинку фірми є головний ЦС і в кожному філіалі є ЦС, який підкоряється головному. В ієрархічній інфраструктурі відкритих ключів, навіть якщо зловмисник видав себе за будь-який ЦС, мережа продовжує працювати без нього, а коли він відновлює нормальну працездатність – він просто знову включається в структуру [7].

Будь-який сертифікат може бути перевірений шляхом побудови ланцюжка сертифікатів від кореневого засвідчувального центру і його верифікації, тобто перевірки зв'язаності суб'єкта сертифіката і його відкритого ключа. Процедура верифікації ланцюжка сертифікатів має на увазі, що всі

«правильні» ланцюжки починаються із сертифікатів, виданих одним кореневим засвідчу вальним центром. Щоб покладатися на сертифікат, сторона, що довіряє, потрібно впевнитися, що:

- кожний сертифікат у ланцюжку підписаний за допомогою відкритого ключа наступного сертифіката в ланцюжку;
- термін дії сертифіката не минув і сертифікат не анульований і кожний сертифікат задовольняє ряду критеріїв, що задаються сертифікатами, розташованими вище в ланцюжку.

Існують два способи одержання стороною, яка довіряє, сертифікатам для перевірки ланцюжка: модель із проштовхуванням і модель із добуванням. Модель із проштовхуванням – це моделья яка припускає, що відправник передає одержувачеві разом зі своїм сертифікатом усі сертифікати ланцюжка і одержувач може негайно їх перевірити. При використанні моделі з добуванням посилає тільки сертифікат відправника, а одержувач сам повинен витягти сертифікат центру, що засвідчує. Оскільки кожний сертифікат містить ім'я видавця, одержувачеві відомо, де перевірити сертифікат.

Перевагою ієрархічної архітектури полягає в тому, що не всі сторони повинні автоматично довіряти всім засвідчувальним центрам. Фактично єдиним засвідчувальним центром, якому необхідно довіряти, це кореневий засвідчувальний центр.

Мережна інфраструктура відкритих ключів.

Мережна архітектура інфраструктура відкритих ключів також є і окремим випадком ієрархічної архітектури. Але в цьому випадку немає одного головного ЦС, якому все довіряють. У цій архітектурі всі ЦС довіряють ЦС, які знаходяться поруч, а кожний користувач довіряє тільки тому ЦС, в якому виписав сертифікат. У дану архітектуру інфраструктури відкритих ключів легко додається новий ЦС. У даній архітектурі найбільш складна побудова ланцюжков сертифікації [10].

Незалежні засвідчувальні центри взаємно сертифікуються, тобто випускають сертифікати, і поєднуються в пари взаємної сертифікації. Взаємна

сертифікація допомагає взаємодіяти засвідчувальним центрам і кінцевим суб'єктам з різних доменів інфраструктури відкритих ключів. У результаті між ними формується мережа довіри.

Перевага мережної архітектури полягає в компрометації одного центру в мережі засвідчувальних центрів які не обов'язково ведуть до втрати довіри всієї інфраструктури відкритих ключів.

Цей вид архітектури який можна розглядати як змішаний вид ієрархічної і мережної види архітектур. Є кілька фірм, у кожної з яких організована деяка своя інфраструктура відкритих ключів, але вони прагнуть спілкуватися між собою, у результаті чого виникає загальна межфірмена інфраструктура відкритих ключів. В архітектурі крос-сертифікованої корпоративної інфраструктури відкритих ключів досить складна система ланцюжка сертифікації [5].

Архітектура мостового ЦС розроблялась щоб убрати недоліки складного процесу сертифікації в крос-сертифікованій корпоративній інфраструктурі відкритих ключів. У цьому випадку всі компанії довіряють не одній або двом фірмам, а одному певному мостовому ЦС, являє собою практично їх головним ЦС, але він не є основним пунктом довіри, а виступає в ролі посередника між іншими ЦС.

2.5 Огляд та аналіз аналогічних систем

Ухвалення Законів України «Про електронні документи і електронний документообіг» і «Про електронний цифровий підпис» створило всі умови для впровадження в державних, недержавних і комерційних організаціях, які між юридичними та фізичними особами електронного документообігу, визнаного державою легітимним.

Національна система електронних цифрових підписів України, від імені держави гарантує якість послуг ЕЦП, а Центральний засвідчувальний орган,

регулює їх якість, будучи органом акредитації і державного нагляду за діяльністю центрів сертифікації ключів, акредитованих і зареєстрованих [2].

На сьогоднішній день у системі функціонують центральний засвідчувальний орган, засвідчувальний центр, НБУ, 32 акредитованих ЦСК, близько 200 тисяч підписантів і користувачів ЕЦП, що застосовують посилені сертифікати відкритих ключів [1].

Постановою Кабінету Міністрів України від 06 серпня 2014 року №311 «Про утворення територіальних органів Державної фіскальної служби та визнання такими, що втратили чинність, деяких актів Кабінету Міністрів України», утворено юридичну особу публічного права - Інформаційно-довідковий департамент ДФС, який став правонаступником Інформаційно-довідкового департаменту Міндоходів.

У складі Інформаційно-довідкового департаменту ДФС функціонує Акредитований центр сертифікації ключів, метою якою є безкоштовне надання послуг електронного цифрового підпису органам державної влади, органам місцевого самоврядування, підприємствам, установам та організаціям всіх форм власності, іншим суб’єктами господарської діяльності та фізичним особам.

Робота АЦСК ІДД забезпечується сучасним програмно-технічним комплексом у складі якого використовуються унікальні програмно-апаратні рішення, які забезпечують високий рівень надійності захисту інформації.

До послуг ЕЦП, які надаються АЦСК ІДД належать [1]:

- реєстрація заявників;
- надання у користування надійних засобів ЕЦП;
- допомога при генерації відкритих та особистих ключів;
- обслуговування посилених сертифікатів ключів заявників, що включає сертифікацію відкритих ключів заявників, розповсюдження та зберігання посилених сертифікатів ключів, управління статусом посилених сертифікатів ключів та розповсюдження інформації про статус сертифікатів ключів;

– надання послуги фіксування часу.

Консультативні послуги за зверненням підписувачів наведені на рисунку 2.1.

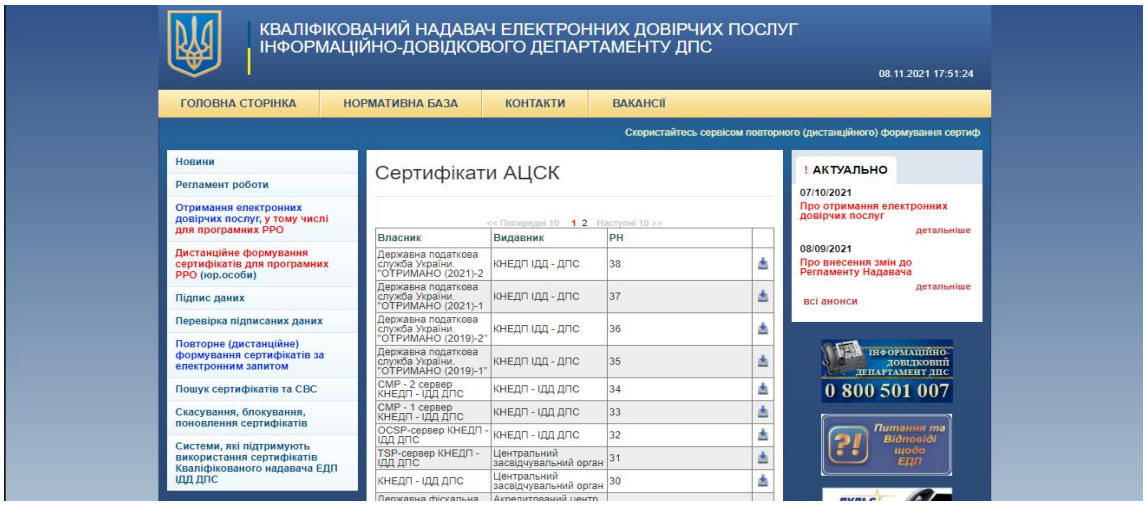

Рисунок 2.1 – Головна сторінка центру сертифікації ключів ІДД

У червні 2008 року АЦСК «MASTERKEY» став першим в Україні центром, діяльність якого була сертифікована по міжнародному стандарту системи менеджменту інформаційної безпеки ISO/IEC 27001:2005 наведено рисунку 2.2.

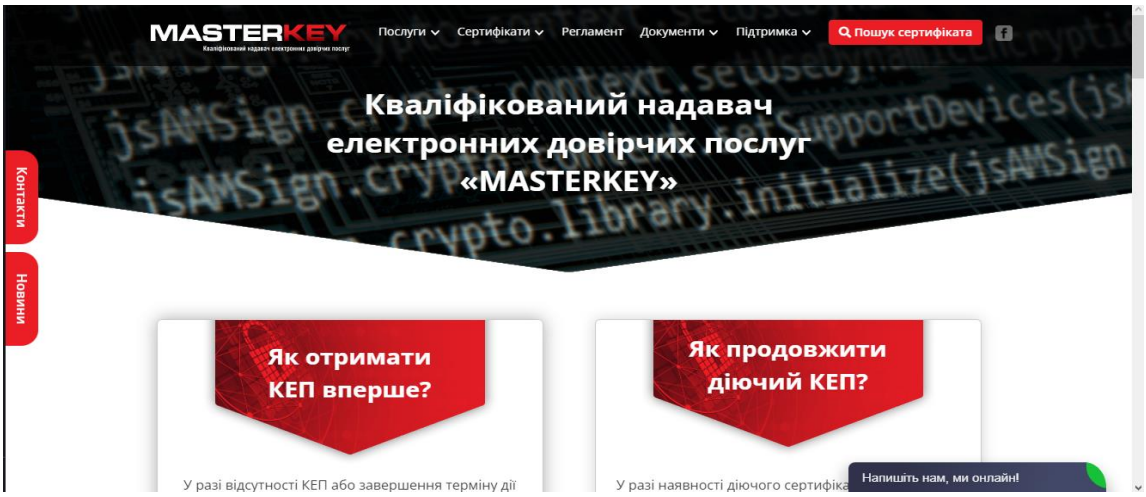

Рисунок 2.2 –Центр сертифікації ключів «MASTERKEY»

Акредитований Центр сертифікації ключів «MASTERKEY» являє собою сервіс-провайдером електронного цифрового підпису і займається обслуговуванням клієнтів, надаючи повний комплекс послуг, пов'язаних з його використанням. Акредитованим центром сертифікації ключів «MASTERKEY» і Державною Податковою Адміністрацією України був підписаний договір, згідно з яким клієнти АЦСК мають можливість надавати звітність не тільки в пенсійний фонд України, а і у Державну Податкову Адміністрацію України.

АЦСК «Український сертифікаційний центр» був створений в 2007 році з метою технічного забезпечення ефективного переходу на електронний документообіг з накладенням електронного цифрового підпису і надає багато послуг для організації захищеного юридично значимого електронного документообігу. АЦСК «Український сертифікаційний центр» створює, видає і засвідчує цифрові сертифікати юридичних і фізичних осіб для здійснення електронного документообігу в рамках комерційних і державних інформаційних систем наведених на рисунку 2.3.

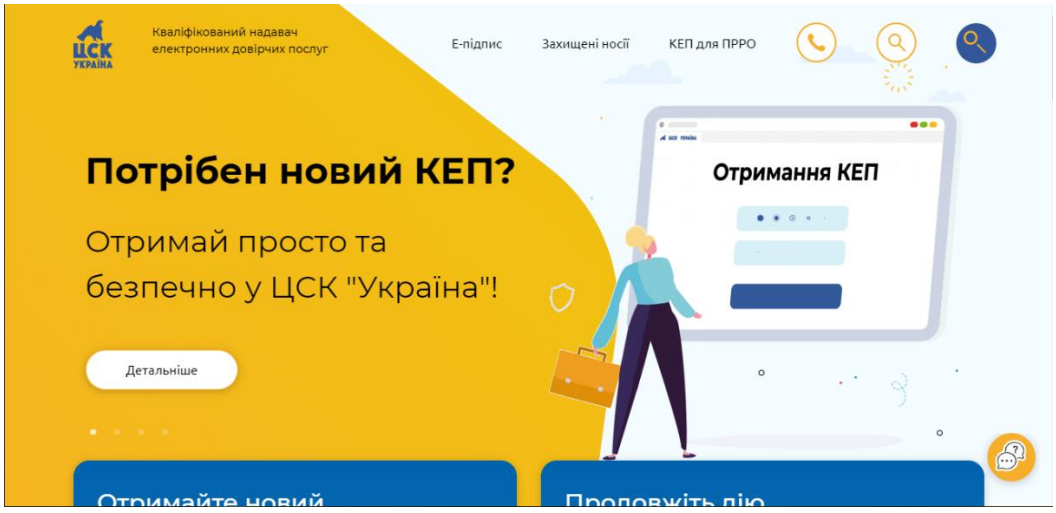

Рисунок 2.3 – АЦСК «Український сертифікаційний центр»

Автоматизована система УСЦ створена для обробки даних, отриманих клієнтами, для формування сертифікатів для клієнтів, ведення бази даних

УСЦ, збереження, поширення, блокування, відновлення і анулювання сертифікатів, формування списків відкликаних сертифікатів, генерації міток часу. Однак пошук і перегляд сертифікатів на даному сайті реалізований не досить зручно для звичайних користувачів. Акредитований центр сертифікації ключів публічного акціонерного товариства комерційний банк «ПРИВАТБАНК» відкрит 01.07.2014р. який можно побачити на рисунку 2.4.

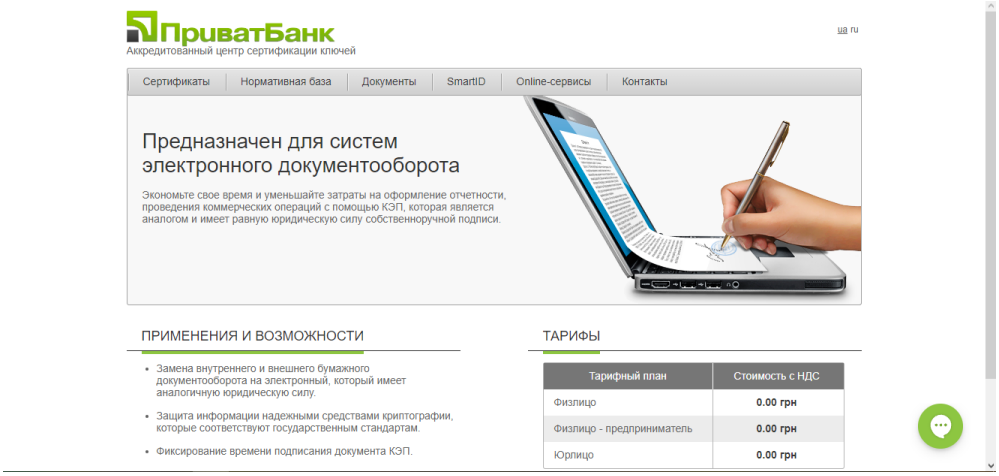

Рисунок 2.4 – Сертифікація ключів «ПРИВАТБАНК»

Центр «ПРИВАТБАНК» реєструє підписантів, який формує сертифікати, зберігає, а також розповсюджує їх, управляє статусом цих сертифікатів, розповсюджує інформацію про статус сертифікатів різним користувачам. За бажанням клієнта Центр надає додаткові послуги: формування позначки часу; повідомлень про статус сертифікатів у реальному часі за підписом Центру; консультації щодо створення; перевірки та використання ЕЦП; засобів генерації особистого та відкритого ключів; а також допомагає при генерації ключів; створенні заявки на формування сертифіката. Крім того факту, на сайті з'явилися онлайн сервіси: для клієнтів – щоб можна було перевірити підписи та перевірити усі сертифікат, а для співробітників – отримати сертифікати, підписати документ, а після всього цього і перевірити підписи [9].

Ключі ЕЦП, отримані в АЦСК Приватбанк, використовуються для роботи з сервісом «Електронні документи і звіти». Повний список центрів сертифікації ключів, які надають послуги ЕЦП є на офіційному сайті Центрального засвідчувального органу [3], які наведено в таблиці 2.1.

Таблиця 2.1 – Електронний реєстр суб'єктів, які надають послуги, пов'язані з ЕЦП

Найменування юридичної особи	Найменування Центру	Статус Центру
Генеральний штаб Збройних Сил України	Акредитований центр сертифікації ключів Збройних Сил	акредитований
Державна казначейська служби України	Акредитований центр сертифікації ключів Державної казначейської служби України	акредитований
Державне підприємство "Головний інформаційно-обчислювальний центр Державної адміністрації залізничного транспорту України"	Акредитований центр сертифікації ключів державного підприємства "Головний інформаційно-обчислювальний центр Державної адміністрації залізничного транспорту України"	акредитований
Державне підприємство "Національні інформаційні системи"	Акредитований центр сертифікації ключів органів юстиції України	акредитований
Державне підприємство "Український інститут інтелектуальної власності"	Акредитований центр сертифікації ключів Державного підприємства "Український інститут інтелектуальної власності"	акредитований
Державне підприємство «Українські спеціальні системи»	Акредитований центр сертифікації ключів державного підприємства "Українські спеціальні системи"	акредитований
Інформаційно-довідковий департамент ДФС	Акредитований центр сертифікації ключів Інформаційно-довідкового департаменту ДФС	акредитований
Національний банк України	Засвідчувальний центр Національного банку України	акредитований
Публічне акціонерне товариство "Комерційний банк "Приватбанк"	Акредитований центр сертифікації ключів Публічного акціонерного товариства "Комерційний банк "Приватбанк"	акредитований

На даний момент, цей сегмент національної інфраструктури ЕЦП перебуває на початковому етапі свого становлення.

2.6 Постановка задачі

Задача даної роботи – побудова і супровід власного центру сертифікації. Центр сертифікації вирішує наступні задачі:

- створення і видачу сертифікатів криптографічних ключів електронного цифрового підпису користувачів;
- гарантія збереження в таємниці закритого ключа електронного цифрового підпису;
- керування сертифікатами (містить у собі можливість припинення і поновлення дії сертифікатів, їх анулювання у випадку компрометації ключа або інших причин);
- ведення реєстру сертифікатів ключів, забезпечення його актуальності і можливості вільного доступу до нього користувачів;
- ведення списку відкликаних сертифікатів – списку скомпрометованих або недійсних, по будь-якій причині сертифікатів, – підтримка його в актуальному стані, випуск відновлень;
- повідомлення власників сертифікатів про закінчення терміну дії сертифіката.

Центр сертифікації формує власний секретний ключ, сертифікат, який містить відкритий ключ центру і підписаний їм самим. Центр сертифікації надійно зберігає закритий ключ свого кореневого сертифіката, тому підписуються сертифікати клієнтів. Відкритий ключ, як і сам сертифікат, повинен бути привселюдно доступний, тому що з його допомогою користувачі можуть перевіряти сертифікати один одного.

3 ПРОЕКТУВАННЯ ІС ДЛЯ СЕРТИФІКАЦІЙНОГО ЦЕНТРУ

3.1 Вибір програмних засобів реалізації системи

Для програмної реалізації ІС для сертифікаційного центру була використана мова програмування PHP, бібліотека Openssl, що входить до складу PHP у вигляді модуля, і база даних Mysql. Вибір мови програмування заснований на тому, що PHP інтенсивно застосовується для розробки різноманітних веб-додатків, підтримується переважною більшістю хостинг-провайдерів і є одним з лідерів серед мов програмування, що застосовуються для створення динамічних веб-сайтів. Мова відрізняється простотою, має високу швидкість виконання, багату функціональність (завдяки наявності великого набору вбудованих засобів для розробки веб-додатків і додаткових модулів), а також має відкритий вихідний код і є кросплатформеним [11].

Криптографічна бібліотека Openssl входить до складу PHP у якості модуля, має відкритий вихідний код, мультиплатформена. Дозволяє створювати ключі RSA, DH, DSA і сертифікати X.509, підписувати їх, формувати CSR і CRT. Підтримувані асиметричні алгоритми: RSA, DSA, Diffie-Hellman key exchange, ГОСТ Р 34.10-2001, ГОСТ Р 34.10-94. Підтримувані хеш-функції: MD5, MD2, SHA, MDC-2, ГОСТ Р 34.11-94. (Підтримка алгоритмів ГОСТ з'явилася у версії 1.0.0, у цей момент перебуває в стадії бета-тестування). У модулі PHP відсутня функція для створення CRL, тому вона була реалізована. Серверна СУБД потрібна для надійного зберігання всіх сертифікатів, даних про них і CRL, а також швидкого доступу і пошуку даних. СУБД Mysql має відмінну швидкодію для читання і пошуку даних, тому відмінно підходить для створення ЦС [12].

3.2 Вибір та застосування функцій бібліотеки Openssl

Для здійснення розробки ІС для сертифікаційного центру була використана бібліотека Openssl – resource openssl_pkey_new ([array

$configargs]). Функція генерує ключову пару: відкритий і секретний ключі. Повертає ресурс ключової пари, або FALSE у випадку невдачі.

Для настроювання генерації ключової пари використовується масив configargs з настроюваннями (табл.3.1), або встановлений у системі файл openssl.cnf у випадку відсутності цього параметра.

Таблиця 3.1 – Масив configargs

Ключ configargs	Тип	Еквівалент openssl.conf	Опис
config	string		файл конфігурації Openssl (якщо вилучити, буде використовуватися openssl.conf, установлений у системі)
digest_alg	string	default_md	алгоритм відбитка, наприклад sha1 або md5
x509_extensions	string	x509_extensions	секція розширень для сертифіката X509
req_extensions	string	req_extensions	секція розширень для CSR
private_key_bits	integer	default_bits	кіл-ть біт закритого ключа (рекомендується не менш 1280, оптимально 2048)
private_key_type	integer		тип закритого ключа: OPENSSL_KEYTYPE_DSA, OPENSSL_KEYTYPE_DH або OPENSSL_KEYTYPE_RSA
encrypt_key	boolean	encrypt_key	шифрувати закритий ключ?

mixed openssl_csr_new (array $dn, resource &$privkey [, array $configargs [, array $extraattribs]])

Функція генерує запит на підпис сертифіката CSR (Certificate Signing Request).

privkey – ресурс ключової пари, попередньо сгенерованої функцією openssl_pkey_new.

resource openssl_csr_sign (mixed $csr, mixed $cacert, mixed $priv_key, int $days [, array $configargs [, int $serial = 0]])

Функція підписує CSR (іншим сертифікатом або собою) і створює сертифікат.

Повертає ресурс сертифіката X509, або FALSE у випадку помилки.

Масив dn містить необхідну для створення сертифіката інформацію про унікальне ім’я (Distinguished Name) (табл.3.2).

Таблиця 3.2 –Масив dn

Ключ dn	Опис
countryname	назва країни, 2 великі букви
stateorprovincename	назва штату або області
localityname	місцезнаходження (наприклад, місто)
organizationname	назва організації
organizationalunitname	назва підрозділу організації
commonname	загальне ім'я (наприклад, ПІБ або host–ім'я сервера)
emailaddress	адреса e–mail

csr – ресурс попередньо сгенерованого CSR, або шлях до файлу CSR в PEM кодуванні.

cacert – ресурс сертифіката, яким буде підписаний цільовий, або NULL для створення самопідписаного сертифіката.

priv_key – ресурс секретного ключа, за допомогою якого буде підписаний цільовий сертифікат.

days – строк придатності генеруємого сертифіката в днях.

serial – серійний номер генеруємого сертифіката.

Одержання ресурсу сертифіката з PEM-закодованих даних сертифіката (x509certdata).

resource openssl_pkey_get_private (mixed $key [, string $passphrase = ""])

Одержання ресурсу секретного ключа. key – PEM-закодовані дані секретного ключа, або шлях до файлу, що містить ці дані.

passphrase – парольна фраза, необхідно вказати, якщо секретний ключ зашифрований.

array openssl_pkey_get_details (resource $key)

Функція повертає масив даних про ключ: число біт, строкове представлення відкритого ключа і алгоритм.

bool openssl_x509_export (mixed $x509, string &$output [,bool $notext])

Функція для експорту сертифіката X509 у форматі PEM.

bool openssl_pkey_export (mixed $key, string &$out [, string $passphrase [, array $configargs]])

Функція для експорту секретного ключа у форматі PEM.

passphrase – опціональний параметр; пароль для шифрування секретного ключа.

bool openssl_pkcs12_export (mixed $x509, string &$out, mixed $priv_key, string $pass [, array $args])

Функція для ескпорта сертифіката і секретного ключа у форматі PKCS#12.

Дані в цьому форматі потрібні користувачам для установки їх сертифікатів разом із секретним ключем у систему.

pass – пароль для шифрування файлу PKCS#12; якщо шифрування не потрібно, указати порожній рядок.

void openssl_x509_free (resource $x509cert)

Функція для звільнення з пам'яті ресурсу сертифіката.

void openssl_pkey_free (resource $key)

Функція для звільнення з пам'яті ресурсу ключової пари.

bool openssl_sign (string $data, string &$signature, mixed $priv_key_id [, int $signature_alg = OPENSSL_ALGO_SHA1])

Функція для підпису даних. data – дані для підпису.

3.3 Виконання конфігурації бібліотеки Openssl

Настроювання функцій бібліотеки Openssl задаються в спеціальному файлі openssl.cnf. Шлях до нього можна змінити, указавши в ключі config

масиву configargs, параметра відповідних функцій. Він має формат INI і містить безліч тонких настроювань для використання даної бібліотеки, розділених на секції [13].

signature – якщо функція завершилася успішно, підпис вертається в цю змінну.

priv_key_id – ресурс закритого ключа для підпису.

signature_alg – хеш-функція:

OPENSSL_ALGO_SHA1 (використовується за замовчуванням);

OPENSSL_ALGO_MD5;

OPENSSL_ALGO_MD4;

OPENSSL_ALGO_MD2.

Далі наведемо найбільш важливі настроювання.

Значення параметра використання ключів keyusage (у настроюваннях використовуються комбінації, перераховані нижче):

- digitalsignature – цифровий підпис;
- nonrepudiation – незречення від авторства;
- keyencipherment – шифрування криптографічних ключів;
- dataencipherment – шифрування користувацьких даних;
- keyagreement – узгодження ключів (при використанні шифру Діффі-Хеллмана);
- keycertsign – перевірка підпису сертифікатів (тільки для сертифікатів ЦС);
- crlsign – перевірка підпису CRL;

encipheronly – шифрування даних (має значення тільки при встановленому keyagreement);

decipheronly – розшифрування даних (має значення тільки при встановленому keyagreement).

Значення параметра розширеного використання ключів extkeyusage:

- serverauth – аутентифікація SSL/TLS web-сервера;
- clientauth – аутентифікація SSL/TLS web-клієнта;

- codesigning – підпис програмного коду;
- emailprotection – захист e-mail (S/MIME);
- timestamping – сервер штампів часу;
- mscodeind – Microsoft Individual Code Signing (authenticode);
- mscodecom – Microsoft Commercial Code Signing (authenticode);
- msctlsign – Microsoft Trust List Signing;
- mssgc – Microsoft Server Gated Crypto;
- msefs – Microsoft Encrypted File System (шифрування файлів Windows);
- nssgc – Netscape Server Gated Crypto.

Для створення сертифіката ЦС у параметрі basicconstraints потрібно вказати CA:true.

Для декодування й кодування даних у формат ASN1 минулого написані класи ASN1 (абстрактний) і похідні від нього: ASN1_SEQUENCE, ASN1_SET, ASN1_NULL, ASN1_UTF8STRING, ASN1_TELETEXSTRING, ASN1_ASCIISTRING, ASN1_BITSTRING, ASN1_OCTETSTRING, ASN1_INT, ASN1_GENERALTIME, ASN1_ENUM, ASN1_UTCTIME, ASN1_BOOL, ASN1_OID (для відповідних типів даних).

Для створення CRL були написані класи X509_CERT і X509_CRL (рис.4.1).

ASN1_SEQUENCE X509_CERT::DECODE ($crt_data_der)

Функція декодує дані сертифіката формату DER (crt_data_der), повертає об'єкт класу ASN1_SEQUENCE.

string X509 _CRL:: CREATE ($ci, $ca_pkey, $ca_decoded)

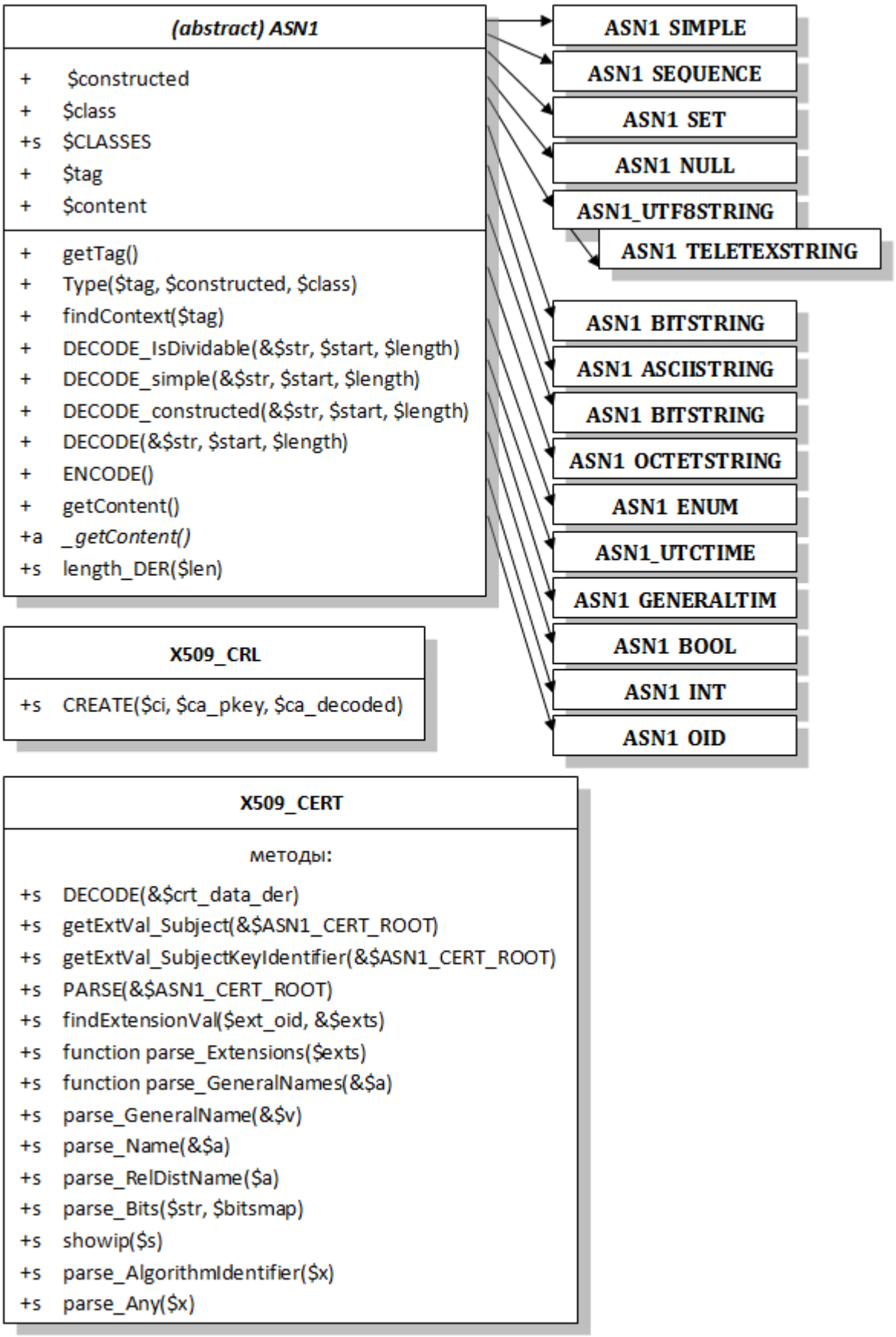

Рисунок 4.1 – Діаграма класів для створення CRL

Функція генерує і підписує CRL, повертає його в Der-Форматі.

Функція створення CRL (Certificate Revokation List).

c_i – масив необхідних даних для створення CRL.

Ключі:

- (int) no – опціонально; номер CRL;
- (int) version – версія CRL; якщо не зазначена, то v2;
- (int) days – число днів до наступного відновлення CRL;
- (int) alg – хеш-функція (див. константи вище); якщо не зазначена, то OPENSSL_ALGO_SHA1.

(array) revoked – список відкликаних сертифікатів, кожний елемент списку – масив з наступними ключами:

- (int) serial – серійний номер сертифіката;
- (string) rev_date – дата відкликання (типу Utctime);
- (int) reason – причина відкликання:
- unspecified,
- keycompromise,
- Cacompromise,
- affiliationchanged,
- superseded,
- cessationofoperation,
- certificatehold,
- removefromcrl,
- privilegewithdrawn.

(string) compr_date – (якщо причина відкликання = keycompromise) дата компрометації ключа (типу Generaltime);

(int) hold_instr – (якщо причина відкликання = certificatehold) інструкції затримки:

0 – None,

1 – Callissuer,

2 – Reject.

ca_pkey – ресурс закритого ключа, пов'язаного із сертифікатом ЦС.

ca_decoded – декодовані дані сертифіката ЦС, об'єкт класу ASN1_SEQUENCE, повернутий функцією X509::DECODE.

3.4 Проектування бази даних ІС сертифікаційного центру

Схема використовуваної бази даних зображена на рисунку 3.2.

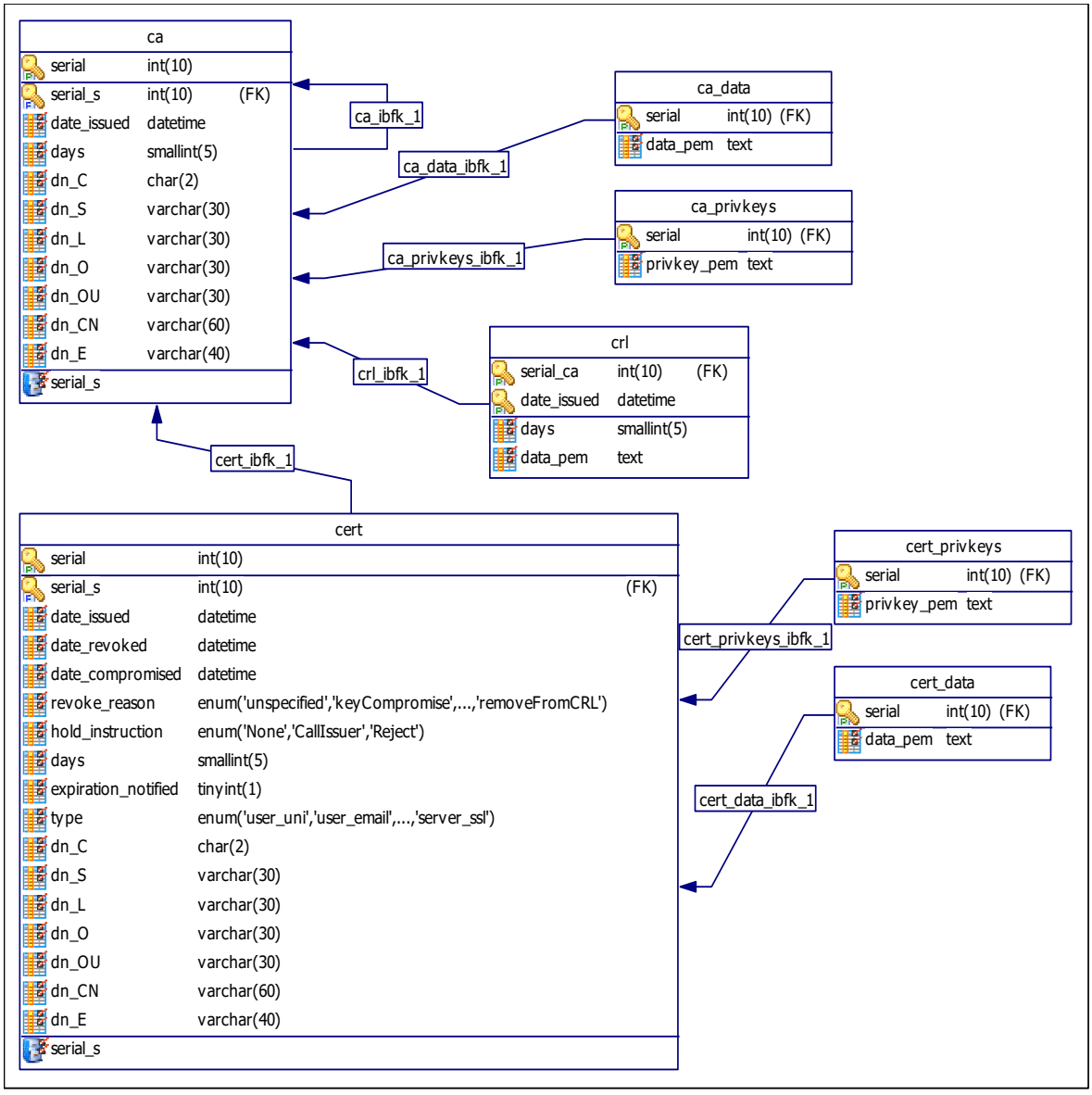

Рисунок 3.2 – Схема бази даних ІС сертифікаційного центру

4 ПРОГРАМНА РЕАЛІЗАЦІЯ ІС СЕРТИФІКАЦІЙНОГО ЦЕНТРУ

4.1 Створення кореневого сертифіката ЦС (режим адміністратора)

Для створення кореневого сертифіката необхідно зайти на сайт із правами адміністратора, увівши пароль у розділі «Адмін.панель», і вибрати «Створити ЦС» у розділі «Адміністративна панель».

Далі заповнити форму, зображену на рисунку 4.1, указавши дані про сертифікат ЦС і термін дії (рекомендується вказувати тривалий строк близько 10 років), і нажати кнопку «Створити». Результат запиту буде виведений на екран, у випадку успіху буде виведений серійний номер сертифіката ЦС.

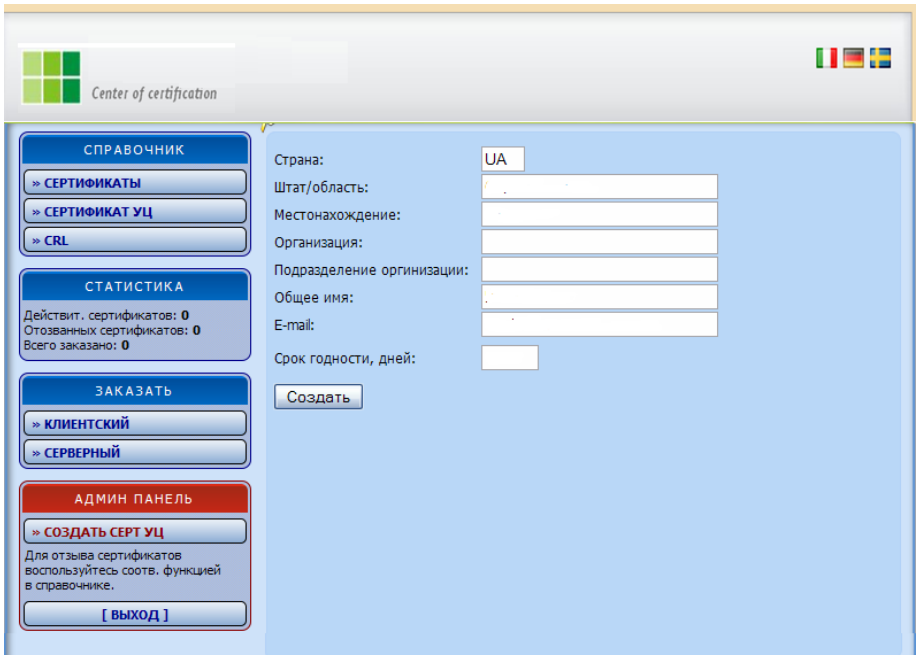

Рисунок 4.1 – Створення кореневого сертифіката ЦС (режим адміністратора)

4.2 Створення CRL (режим адміністратора)

Для створення списку відкликаних сертифікатів (CRL) необхідно зайти на сайт із правами адміністратора, увівши пароль, і в розділі «Адмін.панель» вибрати «Згенерувати CRL».

Цей пункт буде доступний, якщо з моменту останнього відновлення CRL були зроблені відкликання сертифікатів (тоді до назви пункту додасться кількість відкликаних сертифікатів після останнього відновлення, наприклад, «Згенерувати CRL (6)») або якщо настав час для наступного відновлення CRL за розкладом (звичайно 30 днів) (тоді до назви пункту додасться «(!)»).

Повідомлення про закінчення дії (режим адміністратора).

Також є можливість оповіщення клієнтів засобом email про закінчення терміну дії їх сертифікатів і необхідності замовлення нового.

Якщо є прострочені сертифікати, власники яких ще не сповіщені, у розділі «Адмін.панель» буде присутній пункт типу «Повідомити (4)», де 4 – кількість прострочених сертифікатів, власники яких ще не сповіщені. Їх можно побачити на рисунку 4.2.

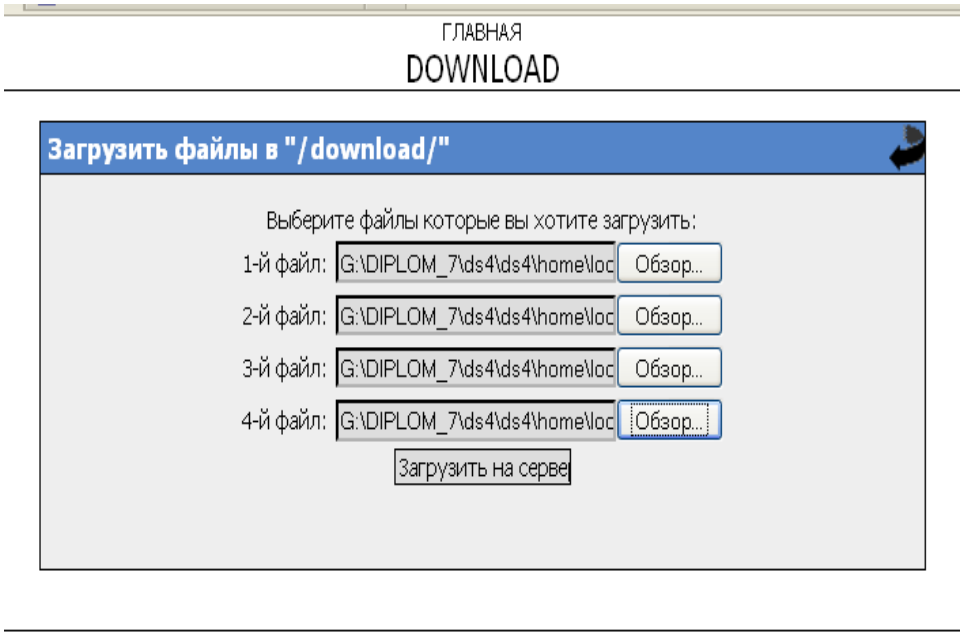

Рисунок 4.2 – Завантаження прострочених сертифікатів

Результат виконання буде показаний на екрані який зображен на рисунку 4.3.

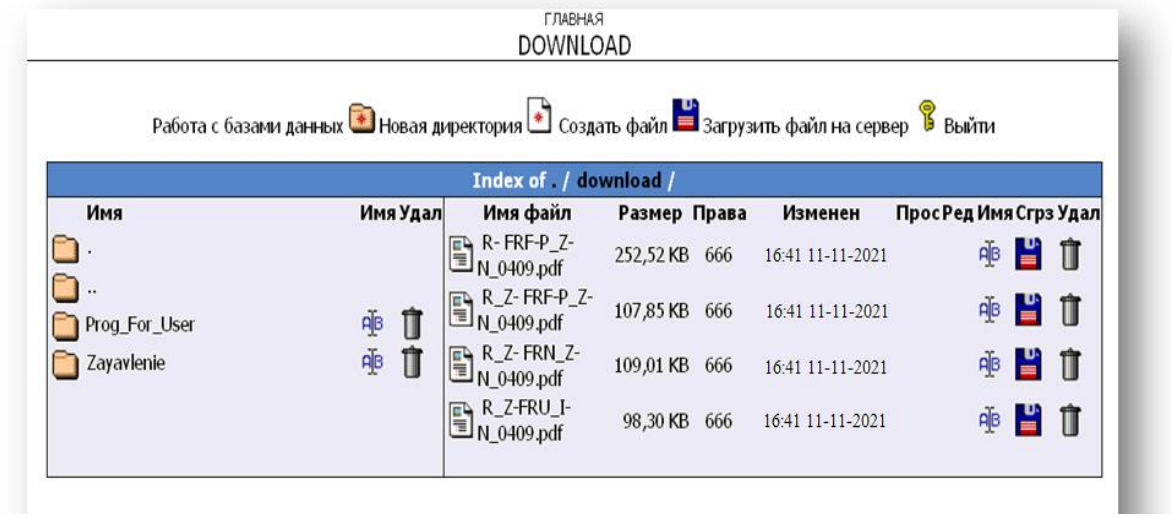

Рисунок 4.3 – Повідомлення про закінчення дії (режим адміністратора)

4.3 Відкликання сертифікатів (режим адміністратора)

Для відкликання сертифікатів необхідно зайти в довідник сертифікатів. У режимі адміністратора в таблиці сертифікатів буде доступний стовпець «Адмін», у якому є пункт «Відкликати» який є на рисунку 4.4.

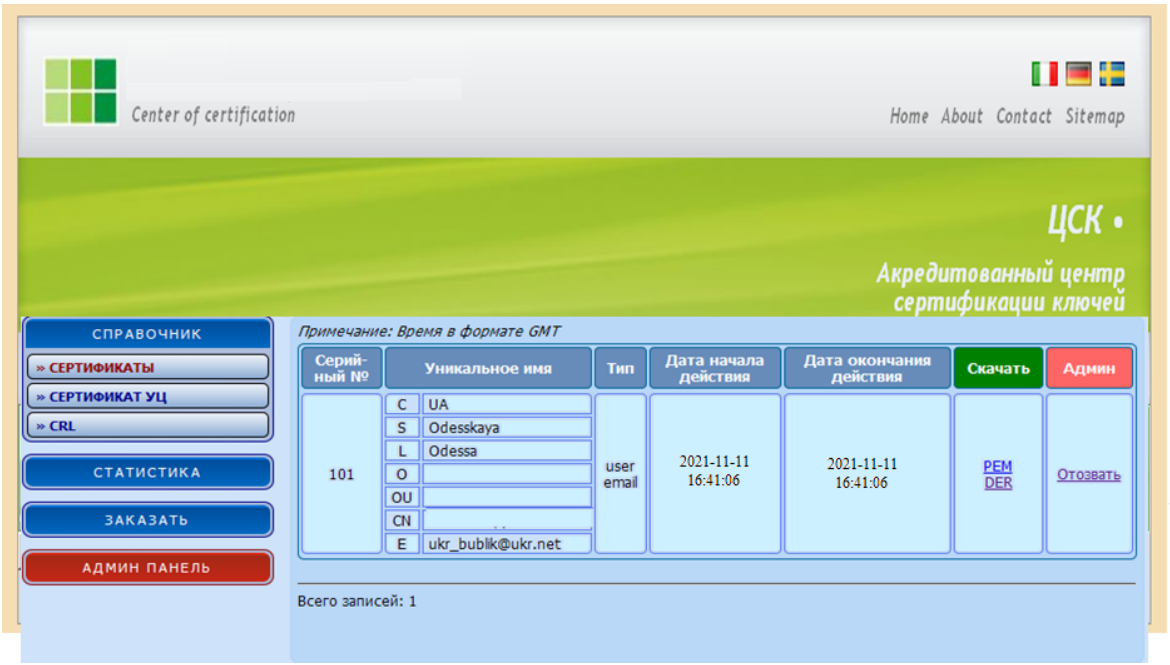

Рисунок 4.4 – Відкликання сертифікатів

Після переходу по посиланню «Відкликати» потрібно вказати причину відкликання:

– компрометація ключа,
– зміна приналежності,
– заміна,
– припинення дії,
– Нажати «Відкликати» (рисунок 4.5).

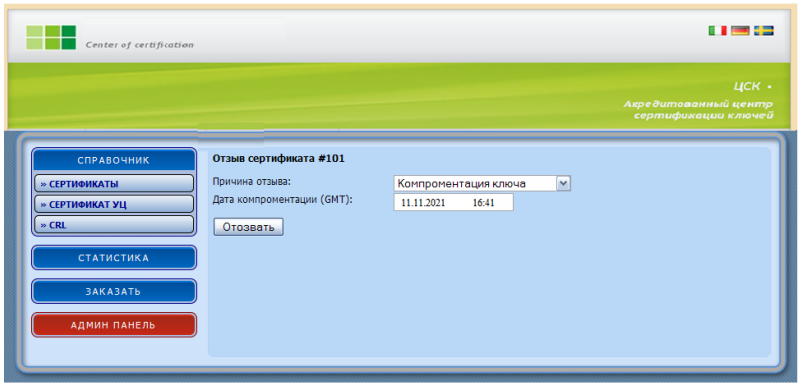

Рисунок 4.5 – Відкликання сертифікатів (крок 2) (режим адміністратора)

4.4 Робота з ІС в режимі «Статистика»

У розділі «Статистика» відображається кількість дійсних на даний момент сертифікатів, кількість відкликаних сертифікатів, а також загальна кількість замовлених сертифікатів у цьому ЦС.

Замовлення сертифікатів. Користувач може зайти на головну сторінку сайту ЦС який можно побачити на рисунку 4.6.

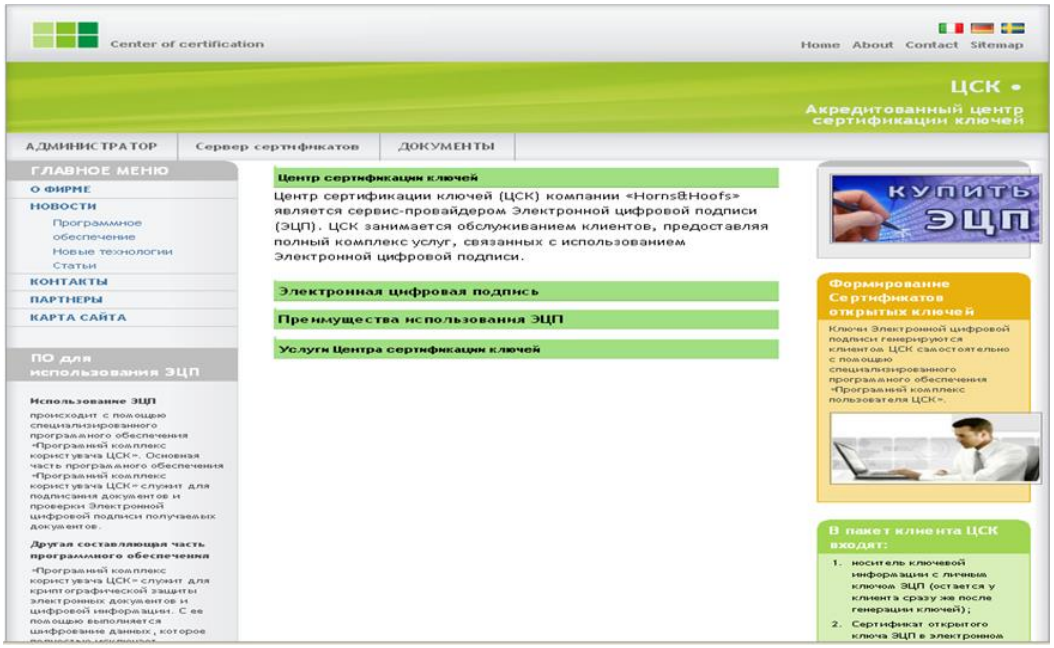

Рисунок 4.6 – Головна сторінка ІС сертифікаційного центру

Користувач у додатку центру може ознайомитися і заповнити договір із ЦС для замовлення сертифіката який ми маємо на рисунках 4.7 – 4.8.

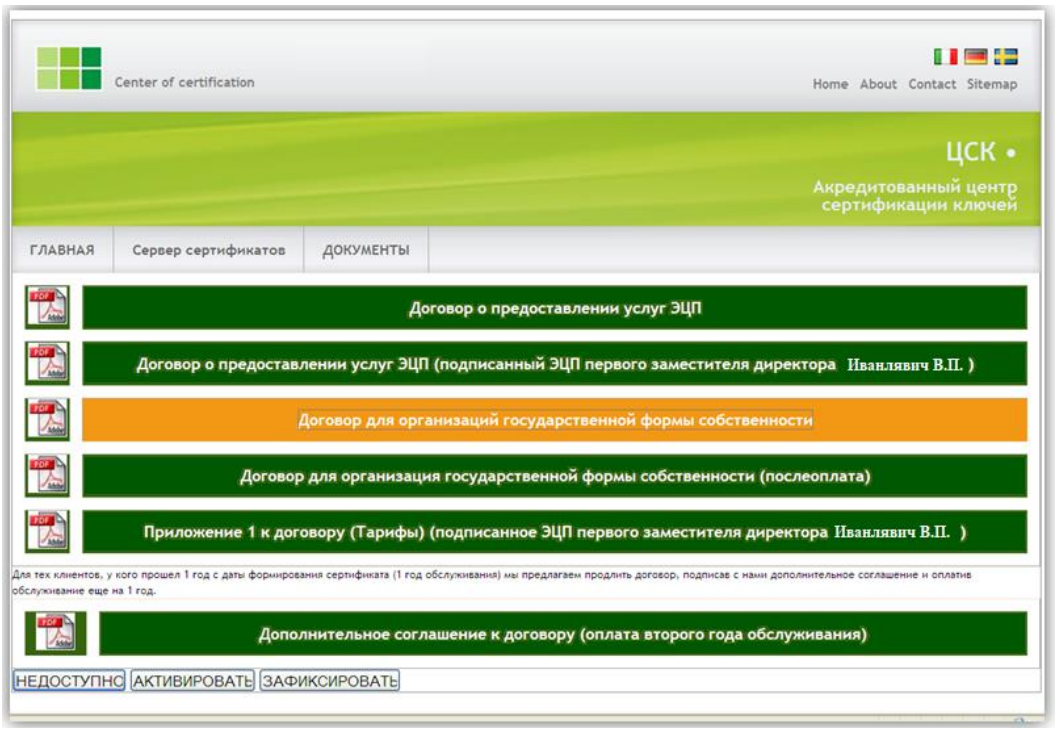

Рисунок 4.7 – Ознайомлення з договорами

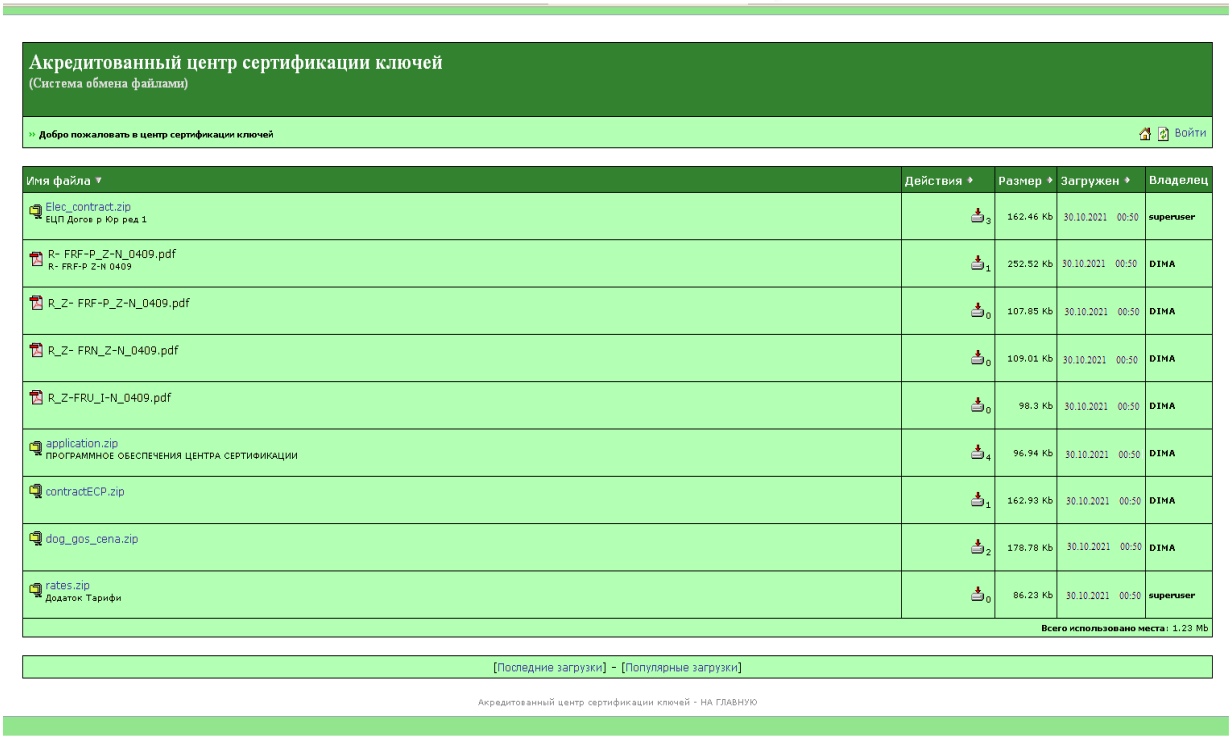

Рисунок 4.8 – Заповнення договору

Для замовлення сертифікатів треба зайти в розділ «Замовити» у підрозділ «Клієнтський» або «Серверний». Відкриється форма для заповнення даних. Для замовлення клієнтського сертифіката треба вибрати тип із запропонованих:

– e-mail (підпис і шифрування);
– e-mail (тільки підпис);
– e-mail (тільки шифрування);
– SSL клієнт;
– універсальний (SSL + e-mail).

Треба заповнити дані унікального імені (країна, область, місцезнаходження, організація, підрозділ організації, загальне ім'я і e-mail), тривалість дії в днях, і парольну фразу з підтвердженням (опціонально) він зображен на рисунку 4.9. Потім нажати кнопку «Створити». У випадку успіху буде виведений серійний номер тільки що створеного сертифіката. На

зазначену електронну адресу буде висланий лист, що містить: сертифікат, секретний ключ і файл формату PKCS#12 для установки їх у систему.

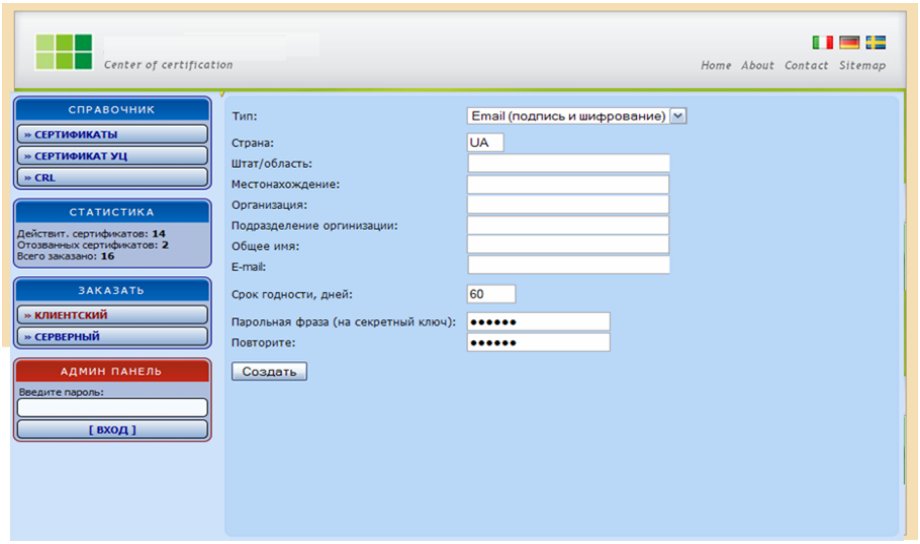

Рисунок 4.9 – Форма замовлення сертифіката

4.5 Довідник сертифікатів

Для доступу до довідника потрібно зайти в розділ «Довідник» у підрозділ «Сертифікати» його можно побачити на рисунку 4.10.

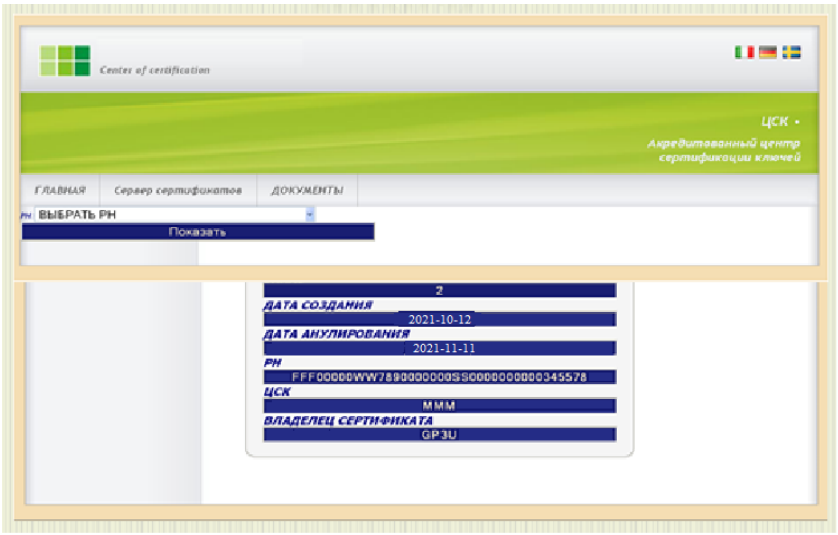

Рисунок 4.10 – Довідник сертифікатів

У сторінці, що відкриється є можливість знайти:

– список усіх дійсних сертифікатів;
– сертифікат по серійному номеру;
– список сертифікатів, що задовольняють певним вимогам: поля унікального імені, тип, дати початку/кінця дії, статус (комбінація статусів: дійсні, відкликані, прострочені).

Приклад списку знайдених сертифікатів показано на рисунку 4.11 (був виконаний пошук дійсних сертифікатів з початком терміну дії ≤2010-05-24). Якщо кількість знайдених сертифікатів більше 10, список розділяється на сторінки по 10 (за замовчуванням) сертифікатів на 1 сторінці.

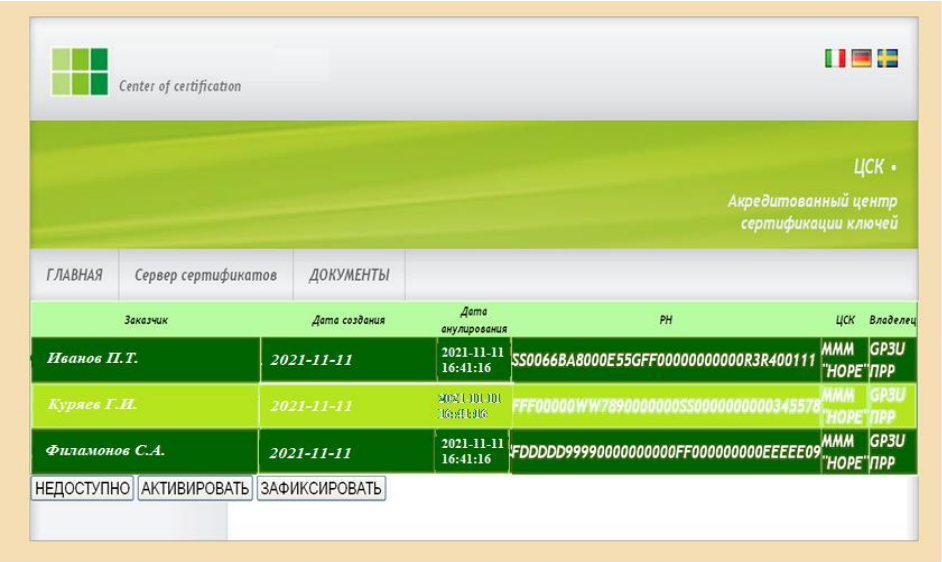

Рисунок 4.11 – Список знайдених сертифікатів

У таблиці відображена інформація про сертифікати і посилання на завантаження у форматах PEM і DER.

Довідник

У розділі «Довідник» у підрозділі «Сертифікат ЦС» можна одержати інформацію про кореневий сертифікат і скачати його, щоб установити в систему в якості довіреного для використання даного ЦС зображені на рисунках 4.12 – 4.13.

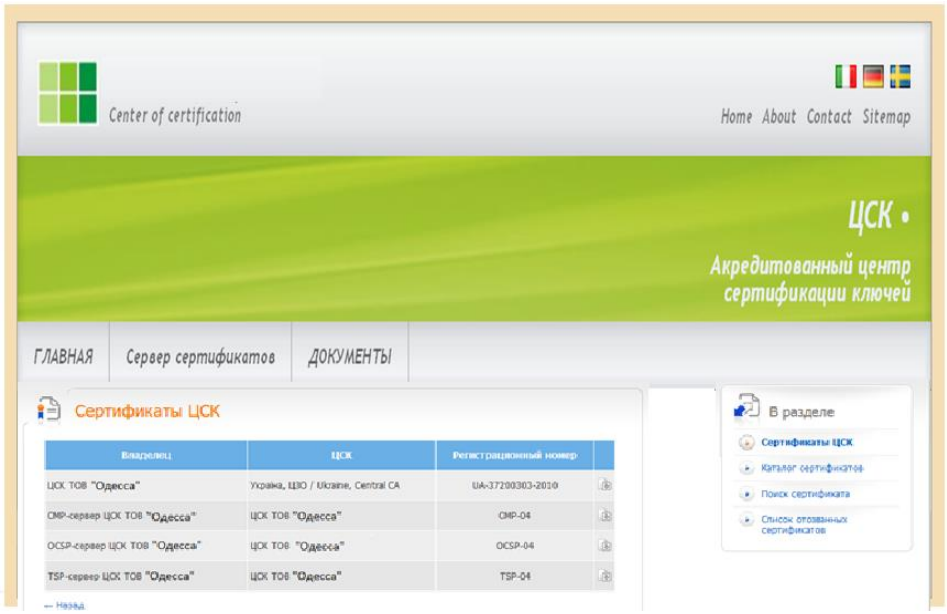

Рисунок 4.12 – Список сертифікатів

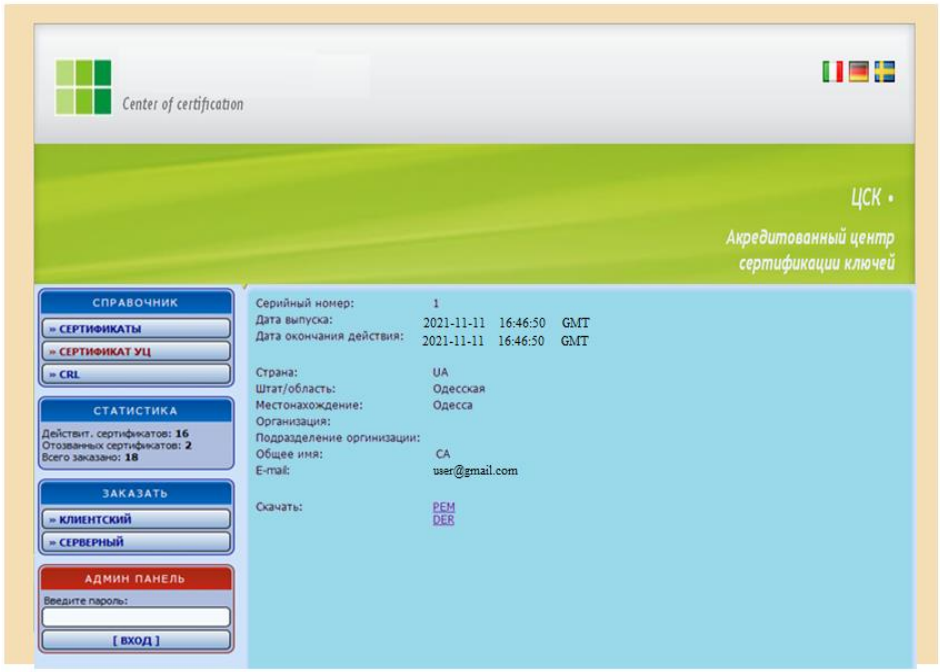

Рисунок 4.13 – Сертифікат ЦС

Довідник – CRL. У розділі «Довідник» у підрозділі «CRL» представлені випущені списки відкликаних сертифікатів який можно побачити на рисунку 4.14.

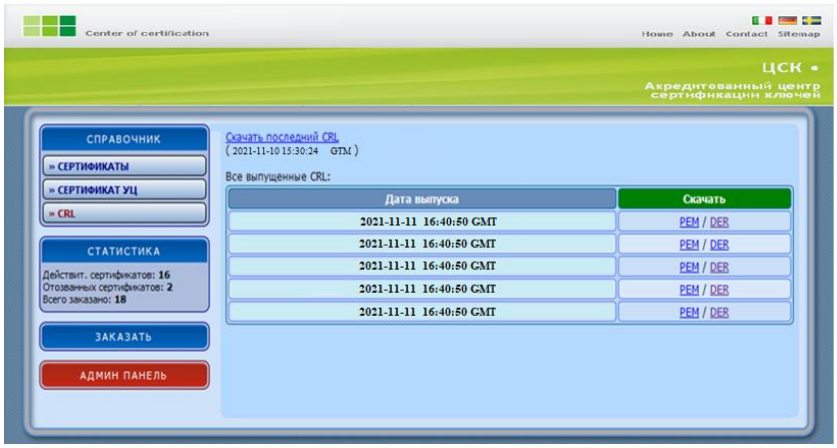

Рисунок 4.14 – Списки відкликаних сертифікатів (CRL)

4.6 Виконання налаштування ЦС (режим адміністратора)

У розділі «Адмін.панель» зайти пункт «Налаштування». У сторінці, що відкрилася, можна змінити пароль адміністратора, зробити різні настроювання ЦС, у тому числі вибрати алгоритм асиметричного шифрування (RSA, DSA), хеш-алгоритм (md2/4/5, sha1, sha224/256/384/512 і інші), довжину ключа, період відновлення CRL який можно побачити на рисунку 4.15.

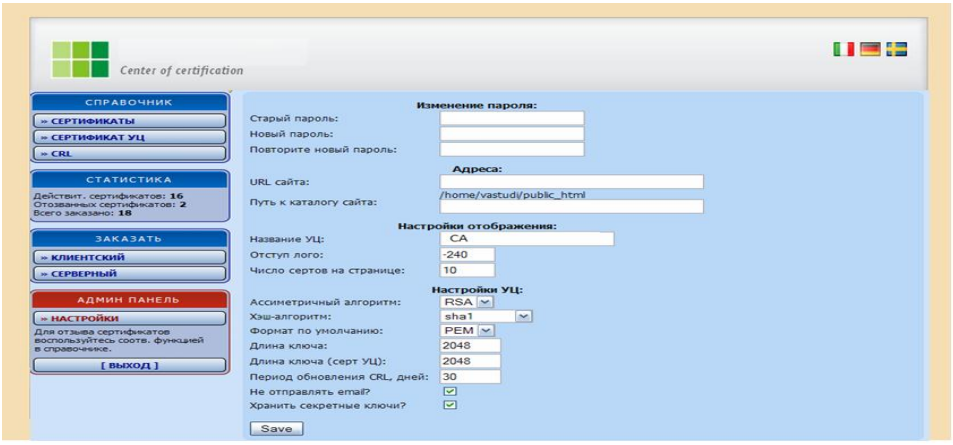

Рисунок 4.15 – Налаштування ЦС

ВИСНОВКИ

В результаті виконання магістерської роботи було проведено проектування та програмну інформаційної системи для сертифікаційного центру. Розробка такої інформаційної системи надає можливість створення, видачі, обліку та керування сертифікатами криптографічних ключів електронного цифрового підпису для користувачів будь-якої корпоративної організації , що вирішить задачу електронного документообігу для власних потреб організації та сприяє розвитоку електронної комерції.

ІС для центру сертифікації розв’язує наступні задачі: створення і видача сертифікатів криптографічних ключів електронного цифрового підпису для різних схем використання; виконує керування сертифікатами (можливість припинення і поновлення дії сертифікатів, їх анулювання у випадку компрометації ключа або ряду інших причин); забезпечує ведення реєстру сертифікатів ключів підписів, його актуальності і можливості вільного доступу до нього користувачів, а також зручного пошуку сертифікатів по різних параметрах; підтримує ведення списку відкликаних сертифікатів (CRL) – списку скомпрометованих або недійсних по будь-якій причині сертифікатів, підтримка його в актуальному стані, випуск відновлень.

Програмна реалізація системи виконана з використанням сучасних технологій та програмних засобів розробки: мова PHP з використанням можливостей бібліотеки Openssl, СУБД Mysql.

Використання такої інформаційної системи для сертифікаційного центру, задовольнить потреби будь-якої корпоративної організації, та є обов'язковим реквізитом електронного документа, без якого він не може мати справжню юридичну чинність і служити основою для здійснення комерційних операцій.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Портал знань. Система електронного документообігу. Електронний цифровий підпис. URL: http://www.znannya.org/?view=e-government-documents (дата звернення: 02.10.2021).
2. Радник. Закон України «Про електронні довірчі послуги» URL: https://radnuk.com.ua/praktyka_zakupivel/nadporogi/zakon-ukrainy-pro-elektronni-dovirchi-posluhy-novi-pravyla-zastosuvannia-elektronnykh-pidpysiv-ta-pechatok/ (дата звернення: 06.09.2021).
3. Електронний реєстр чинних, блокованих та скасованих сертифікатів відкритих ключів URL: https://czo.gov.ua/ca-registry (дата звернення: 15.09.2021).
4. Остапов С.Е., Валь Л.О. Основи криптографії. Навчальний посібник. Ч: Книги-XXI, 2008. – 188 с.
5. Франчук В.М. Захист інформаційних ресурсів: криптографічні та стеганографічні методи захисту даних. – К.: НПУ ім.. М.П. Драгоманова, 2012. – 120 с.
6. В. Ємець, А. Мельник, Р. Попович. Сучасна криптографія. Основні поняття. К.: БаК, 2003. – 144 с.
7. Остапов С.Е., Євсеєв С.П., Король О.Г. Технології захисту інформації. Навчальний посібник. Х.: ХНЕУ, 2013. – 249 с.
8. Гончарова Л.Л., Возненко А Д., Стасюк О. І., Коваль Ю. О. Основи захисту інформації в телекомунікаційних та комп'ютерних мережах. К.: 2013. – 435 с.
9. Акредитований центр сертифікації ключів – ПриватБанк. URL: https://acsk.privatbank.ua/main (дата звернення: 05.10.2021).
10. О.Г. Додонов, Д.В. Ланде, В.Г. Путятін. Інформаційні потоки в глобальних комп’ютерних мережах. К.: Наук. думка, 2009. – 295 с.
11. Люк Веллинг, Лора Томсон. Разработка веб-приложений с помощью PHP и MySQL. М.: Вильямс, 2010. – 848 с.

12. Ярцев В.П. Організація баз даних та знань: навчальний посібник. К., ДУТ, 2018, – 214 с.
13. Гутманс Э., Баккен С., Ретанс Д. PHP 5. Профессиональное программирование. пер. с англ. СПб.: Символ-Плюс, 2006. – 704 с.