МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ
КАФЕДРА КОМП'ЮТЕРИЗОВАНИХ СИСТЕМ ЗАХИСТУ ІНФОРМАЦІЇ

ДОПУСТИТИ ДО ЗАХИСТУ
Завідувач кафедри
_____ С.В. Казмірчук
«_____»_____ 20__ р.

На правах рукопису
УДК 004.056.55(079.2)

ДИПЛОМНА РОБОТА

ЗДОБУВАЧА ВИЩОЇ ОСВІТИ
ОСВІТНЬОГО СТУПЕНЯ «БАКАЛАВР»

Тема: Криптографічний модуль захисту системи електронних платежів

Виконавець: Д.А.Лактіонова
Керівник: к.т.н., доцент С.С. Ільєнко
Нормоконтролер: к.т.н., доцент С.С. Ільєнко

Київ 2021

НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

Факультет: Кібербезпеки, комп'ютерної та програмної інженерії

Кафедра: Комп'ютеризованих систем захисту інформації

Освітній ступінь: Бакалавр

Спеціальність: 125 «Кібербезпека»

Освітньо-професійна програма: «Безпека інформаційних і комунікаційних систем»

ЗАТВЕРДЖУЮ

Завідувач кафедри

_____ С.В. Казмірчук

«___» _____ 2021 р.

ЗАВДАННЯ

на виконання дипломної роботи

здобувача вищої освіти Лактіонової Діани Андріївни

1. Тема: *Криптографічний модуль захисту системи електронних платежів* затверджена наказом ректора від «26» квітня 2021 р. № 652/ст.

2. Термін виконання: з 10.05.2021 р. по 20.06.2021 р.

3. Вихідні дані: проаналізувати існуючі системи електронних платежів, дослідити можливі проблеми електронних платежів, оплата безготівковими коштами, та безпечну транзакцію в інтернеті, продемонструвати алгоритм безпечної транзакції в інтернеті.

4. Зміст пояснювальної записки: аналіз існуючих систем електронних платежів, їх проблеми, протоколи та безпечну оплату в інтернеті.

КАЛЕНДАРНИЙ ПЛАН

виконання дипломної роботи

№ п/п	Етапи виконання дипломної роботи	Термін виконання етапів	Примітка
1.	Уточнення постановки задачі	19.04.2021	*Виконано*
2.	Аналіз літературних джерел	22.04.2021	*Виконано*
3.	Обґрунтування рішення	27.04.2021	*Виконано*
4.	Дослідження сучасних принципів організації систем електронних платежів та методів захисту інформації	29.04.2021	*Виконано*
5.	Розробка авторського криптографічного модуля захисту системи електронних платежів	15.05.2021	*Виконано*
6.	Дослідження запропонованого програмного модуля захисту системи електронних платежів базі криптографічних перетворень	20.05.2021	*Виконано*
7.	Оформлення і друк пояснювальної записки	03.06.2021	*Виконано*
8.	Оформлення презентації	05.05.2021	*Виконано*
9.	Отримання рецензій від рецензентів	08.06.2021	*Виконано*
10.	Підготовка до захисту в ЕК	14.06.2021	*Виконано*

Здобувач вищої освіти Д.А. Лактіонова

(підпис, дата)

Керівник дипломної роботи С.С. Ільєнко

(підпис, дата)

РЕФЕРАТ

Дипломна робота складається зі вступу, двох розділів, висновків до розділів та загального висновку, списку використаних джерел, додатків, загальний обсяг роботи складає 51 сторінку, має рисунків та сторінок додатків. Список використаних джерел містить 27 пунктів, та займає 3 сторінки.

Метою дипломної роботи є розробка авторського криптографічного модуля захисту системи електронних платежів.

Завданням даної дипломної роботи є проведення дослідження сучасних методів та підходів щодо захисту систем електронних платежів; розробка авторського криптографічного модулю захисту системи електронних платежів з використанням середовища розробки на C ++; проведення дослідження власного програмного модуля захисту системи електронних платежів на основі забезпечення конфіденційності та цілісності інформації.

В дипломній роботі розглянуте питання безпечних транзакцій в інтернету, та сама система електронних платежів, аналіз загроз оплати, та модулі безпечних транзакцій. Реалізована програма виконує шифрування двома шифрами, хешованого повідомлення, та передає зашифрований текст, дешифрує і порівнює початковий хеш з дешифрованим, тим самим перевіряючи канал транзакції.

Запропонований програмний модуль, можна використовувати в транзакціях в інтернеті, для забезпечення безпеки інтернет транзакції.

Ключові слова: SET, транзакція в інтернеті, оплата, шифрування, дешифрування, безпека даних, безпечна транзакція.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

БСШ – блокові симетричні шифри

ДСТУ – Державний Стандарт України

СЕП – система електронних платежів

НБУ – Національний банк України

КБ – комерційний банк

АТ – акційне товариство

СУБД – система управління базами даних

СУІБ – система управління інформаційною безпекою

ЕЦП – Електронно Цифровий Підпис

ЕП – електронна пошта

НД ТЗІ – нормативні документи в галузі технічного захисту інформації

ЦП – цифровий підпис

ВПС – віртуально виділений сервер

ЕПС – електронно платіжні системи

ВСТУП

Актуальність: Задача захисту систем електронних платежів на сьогоднішній день є однією з актуальних тем, тому що популярність платіжних карток кожен день зростає, тож стає питання захисту самої системи грошових транзакцій, щоб система була зручною, а головне безпечною на сьогоднішній день придумано багато протоколів захисту безготівкової оплати, та різних секретних даних, які нікому не повідомляються, так дату закінчення терміну дії та CVV – секретного коду картки нікому не можна повідомляти, і саме це є проблемою, зробити так, щоб ці дані не могли зчитати, перехопити, та використати, для зняття готівки з вашої картки.

Метою дипломної роботи є розробка авторського криптографічного модуля захисту системи електронних платежів.

Об'єктом дослідження є процедура криптографічних перетворень.

Предмет дослідження: методи реалізації системи електронних платежів за умови забезпечення конфіденційності та цілісності інформації.

Методи дослідження: Проведені дослідження базуються на сучасних методах побудови захищених інформаційних мереж, методах симетричного та асиметричного шифрування даних.

Завданням дипломної роботи є:

проведення дослідження сучасних методів та підходів щодо захисту систем електронних платежів; розробка авторського криптографічного модулю захисту системи електронних платежів з використанням середовища розробки на C ++; проведення дослідження власного програмного модуля захисту системи електронних платежів на основі забезпечення конфіденційності та цілісності інформації.

Оцінка сучасного стану проблеми на основі вітчизняної та зарубіжної літератури:

Сьогодні триває робота з розвитку та вдосконалення системи електронних платежів Національного банку України, що, безсумнівно, є необхідним перейти на нові міжнародні стандарти фінансової звітності, запровадити нові інноваційні інструменти переказу коштів. Це дуже важливо покращити надійність системи, а також рівень довіри учасники під час розрахунків за допомогою платіжних засобів.

Необхідно вирішити питання конкуренції та зв'язку між банківською та торгівельною мережами. Система електронних платежів Національного банку України на сучасному етапі має багато проблем, факторів, що уповільнюють її розвиток. По-перше, це недосконаліть чинного законодавства України, яке призводить до значних розбіжностей у поглядах і діях між учасники системи. Крім того, рівень інформаційного забезпечення учасників системи електронних платежів Національного банку України є досить низьким, що призводить до непрозорості проведених операцій в системі; залишається невисоким рівень нормативного забезпечення учасників системи електронних платежів Національного банку України, який треба зробити доступнішим та зрозумілішим для клієнтів.

Галузь застосування: Даний програмний модуль може бути застосований в сфері захисту інформації, для безпечної передачі даних транзакції.

Практична цінність дипломної роботи полягає у створенні власного програмного модуля з використанням інтегрованого середовища розробки Dev C++ за рахунок використання методу симетричного шифрування AES та асиметричного шифрування RSA, що дає змогу поетапно продемонструвати реалізацію системи електронних платежів для проведення безпечних транзакцій через мережу Інтернет на основі забезпечення конфіденційності та цілісності інформації.

РОЗДІЛ 1. ЗАГАЛЬНІ ПОНЯТТЯ ВИКОРИСТАННЯ СИСТЕМ ЕЛЕКТРОННИХ ПЛАТЕЖІВ (СЕП)

1.1. Класифікація та структура систем електронних платежів в Україні та світі.

Цифрові гроші - є платіжним засобом, що представлений у електронному вигляді, оборот яких, гарантує анонімність. Такі кошти є прямою аналогією готівковим купюрам, вони це електронні документи які мають номінальну вартість вказівку на емітента – це організація що видали ці електронні кошти, індивідуальні ознаки (серія та номер) і елементи захисту від підробки, вони завірені електронним цифровим підписом емітента. Електронні гроші – це грошові зобов'язання емітента в електронному вигляді, які знаходяться на електронному носії у розпорядженні користувача. Такі грошові зобов'язання:

— фіксується і зберігається на електронному носію

— випускається емітентом при отриманні від інших осіб грошових коштів в обсязі не менш як емітована грошова вартість

— приймаються як засіб платежу іншими.

Безготівковий рахунок і електронні кошти це не одне й те саме. Електронні кошти на відміну від безготівкового розрахунку є неперсоніфікованим платіжним засобом та можуть мати окремий обіг відмінний від банківського обігу грошей проте може бути обіг і в державних або банківських установах та системах. Вірогідно в майбутньому центральні банки будуть виробляти емісію (випуск в обіг грошей або цінних паперів) електронних коштів так само як зараз друкують банкноти. Обіг електронних грошей неможливий без допомоги комп'ютерних мереж, інтернету, платіжних карт, електронних гаманців і пристроїв які взаємодіють з платіжними картами такі як:

— Банкомати;

— Pos-термінали;

— Платіжні кіоски.

— Критерії, яким повинні відповідати електронні засоби:

— Анонімність покупця (при оплаті електронними засобами щоб не можна було відстежувати і визначати, хто саме купив продукт або послугу)

— Незалежність від місця зберігання (безпека електронних грошей не повинна залежати від місця їх зберігання)

— Захист від шахрайства (не можна вдруге використовувати точну копію електронної монети)

— Автономність (система не повинна викликати третю особу в чеках)

— Подільність (якщо сума платежу менше суми в електронних засобах на ресурсі, електронні засоби необхідно розділити на дрібні частини, щоб власник міг отримати залишок після покупки)

Ці критерії повинні відповідати, щоб система електронних платежів вважалася ідеальною. Застосовувані криптографічні протоколи, що забезпечують функціонування електронних грошей, їх переказ між абонентами та гарантію еквівалентності операцій із звичайними грошима, називаються протоколами електронних платежів, здійснених за допомогою електронних засобів.

Проектування та функціонування будь-яких складних систем вимагає певної науково-методологічної бази, яка включає електронну платіжну систему. Як і будь-яку таку систему, СЕП можна розділити на три рівні: операційний, системний та технічний.

Оперативний рівень – це СЕП в аспекті ефективної підтримки ділової частини діяльності людини в тій галузі, для якої вона використовується.

Системний рівень – СЕП як розгляд його системної організації, що визначається вимогами оперативного рівня. Елементи, виділені на цьому рівні: обчислювальні системи, канали зв'язку та маршрутизатори, говорить про здатність організовувати обчислення, розподіл функціональних можливостей між структурними елементами системи, вибір та розробку системного та прикладного програмного забезпечення.

Технічний рівень – це огляд інформаційної системи як технічної конструкції, практичної реалізації та можливостей організації її роботи. Основне завдання рівня – визначити переносимість і можливість взаємодії окремих елементів конструкції.

Структура системи електронних платежів

В даний час ні в кого немає сумнівів в тому, що електронна комерція відкриває нові можливості для продавців, покупців і фінансових установ. Їх швидкий розвиток в Інтернеті призвів до появи великої кількості різних інтернет-сервісів і магазинів, які будуть надавати платіжні послуги онлайн пластиковими картами або електронними грошима. Таким чином, дуже легко створити зручну і надійну платіжну систему, яка забезпечує систему розрахунків між фінансовими установами, комерційними організаціями і користувачами Інтернету при купівлі і продажу товарів і послуг через Інтернет. Спочатку для розрахунків використовувалися пластикові карти, потім з'явилися електронні гроші різних платіжних систем.

На початку XXI століття виник новий швидко розвивається ринок банківських пластикових карт. Пластикові карти - це персоналізований платіжний інструмент, який дозволяє людині робити безготівкову оплату товарів або послуг, а також отримувати готівку у відділеннях банкоматів або банкоматів. Приймальна карта торгового підприємства і відділення банку створює мережу пунктів обслуговування карт.

Історично склалося так, що велика кількість платіжних інструментів, в тому числі разом і орієнтованих на віддалене використання, пов'язане з певними банківськими рахунками. Різні стандартні пластикові карти, система SWIFT і багато систем, які використовуються в Інтернеті, засновані на наявності незалежних банківських рахунків. Платіж - це, по суті, команда на переказ визначеної суми на конкретний рахунок.

Тому банківські структури, які беруть участь у створенні та обслуговування платіжної системи, є її важливою частиною [3].

Банк-емітент випускає карти і гарантує виконання фінансових відносин, пов'язаних з використанням випущеної їм пластикової карти як засіб платежу. Банк-еквайєр підтримує необхідні операції при забезпеченні взаємодії учасників сервісу за допомогою платіжної системи:

— обробка авторизаційних записів;

— переказ грошових коштів за товари та послуги з рахунків продавців;

— отримання, сортування та відправка документів (паперових і електронних), що фіксують використання договору з використанням карт;

— розсилка стоп-листів - списків карт, операції за якими з яких-небудь причин припинені.

Крім того, банк-еквайєр може видавати ідентифікаційні дані на карти в своїх відділеннях або через банкомати, які йому потрібні. Перші технічні функції банк-еквайєр може передати спеціалізованим сервісним організаціям - процесним центрам.

Функції еквайра та емітента може поєднувати один банк.

Всі електронні платіжні системи доступні в двох основних класах: кредитні системи і дебетові системи.

Кредитні платіжні системи, засновані на використанні кредитних карт для електронних розрахунків між сторонами договору, із залученням додаткових заходів безпеки: шифрування повідомлень, електронний підпис. Всі кредитні системи вимагають підтвердження платоспроможності клієнтів або додаткових характеристик наданих коштів платежу банком-емітентом або іншої уповноваженої організацією.

Схеми дебетових платежів фіксуються на використовуваних електронних еквівалентних чеках і чеках. Електронні гроші повністю імітують реальні гроші. Організація, що управляє платіжною системою - емітент - виробляє високоякісні електронні копії грошей, які користувачі купують для оплати покупок, а потім продавець повертає їх емітенту.

Емітувати (випускати) електронні гроші можуть як банки, так і небанківські організації. Вони не забезпечуються гарантіями держави, випущену

електронну готівку може викупити лише сам емітент, тому переказ електронних грошей з однієї платіжної системи в іншу є досить незручною і дорогою операцією. Однак низька ціна транзакції робить електронні гроші привабливим інструментом оплати в Інтернеті.

Електронні чеки схожі на звичайні паперові чеки і служать для платника його банку вказівкою: перерахувати гроші зі свого рахунку на рахунок одержувача або передати їх власнику чека. Відмінність від паперових чеків полягає в тому, що вони підписуються в електронному вигляді, і самі чеки видаються в електронному вигляді.

Відмінною рисою продажу та зняття готівки з пластикових карток є те, що ці операції здійснюються магазинами та банками «в кредит» [3].

Система електронних платежів в Україні та світі

Електронна платіжна система є власністю Національного банку України, вона забезпечує розрахунки між банківськими установами у національній валюті України. СЕП базується на повністю без паперової технології та передачі електронних повідомлень через власну телекомунікаційну систему Національного банку.

СЕП здійснює платежі від клієнтів банку або за власними зобов'язаннями. Ця система не забезпечує пряме обслуговування клієнтів від комерційних банків.

Національний банк України є власником та оператором СЕП, забезпечує розробку, вдосконалення та функціонування програмно-апаратних комплексів та систем захисту інформації, розробляє відповідну нормативно-правову базу. НБУ гарантує надійність та безпеку СЕП. НБУ відповідає за нагляд за платіжною системою.

Порядок функціонування системи електронних платежів Національного банку України, прийняття та виключення з неї, перекази з використанням цієї системи та інші питання, пов'язані з системою електронних платежів Національного банку України, визначаються Національним банком України .

Обов’язковою умовою переказу через електронну платіжну систему Національного банку України є встановлення кореспондентських відносин між

банком та Національним банком України шляхом відкриття кореспондентського рахунку в Національному банку України.

СЕП була створена в 1993 р. Перші виплати за СЕП були здійснені 5 січня 1993 р. З 1 січня 1994 р. Паперові та телеграфні довідкові аркуші для міжбанківських розрахунків в Україні були повністю скасовані [1. 504ст].

Електронні платіжні системи, призначені для проведення онлайн-платежних операцій. За допомогою платіжної системи ви можете оплачувати товари та послуги проектів та сервісів. Наприклад, оплата за мобільний зв'язок, комунальні послуги, кабельне чи супутникове телебачення, послуги провайдера, покупки в інтернет-магазинах.

Електронні (цифрові) гроші використовуються як засіб платежу в електронних платіжних системах; це аналог готівки, який при необхідності, як правило, миттєво передається з одного електронного гаманця на інший. Усі способи електронних платежів можна розділити на дві групи за способом доступу до електронного рахунку:

— системи, що мають веб-інтерфейс для управління електронним гаманцем;

— системи, що вимагають встановлення додаткового програмного забезпечення для управління електронним гаманцем.

Як повідомляє Укрінформ та прес-служба НБУ.

Відтепер система доступна для міжбанківських платежів 23 години на добу, 7 днів на тиждень. Учасники СЕП можуть здійснити міжбанківський переказ з 01:00 до 24:00. При переході на новий банківський день - з 00:00 до 01:00 - технологічна перерва на годину.

Міжбанківські перекази коштів банками через СЕП 23/7 є добровільними опціями. Банки самостійно визначають час початку та закінчення роботи в системі, виходячи із власних потреб та потреб клієнтів. В першу ніч на 3 серпня цього року платежі проводились у системі АТ "Альфа-Банк", АТ "Перший інвестиційний банк", АТ "ІдеяБанк". На світанку до них приєдналися ще два банки - АТ КБ "ПриватБанк" та АТ "Ощадбанк".

Загалом за перші дні роботи за новими правилами всі 75 банків України та Державна казначейська служба України здійснили платежі в СЕП, у тому числі чотири банки (АТ «Альфа-Банк», АТ «ОПТ Банк», АТ). КБ "ПриватБанк"). АКБ "Укргазбанк") направив початкові платежі в СЕП до кінця дня.

Протягом банківського дня 3 серпня система обробила 1 600 000 платежів на загальну суму понад 119 млрд. Грн., Зокрема: у період з 01:00 до 8:30 - 28 640 платежів на суму 6 800 000 000 грн. У період з 08:30 до 19:00 (робочий час визначається попередніми положеннями СЕП) – понад 1 303 000 платежів на суму 112 млрд. Грн. У період з 19:00 до 00:00 – 13870 виплати в сумі 69 000 000 грн

«Тепер клієнти банку матимуть майже цілодобовий доступ до фінансових послуг і зможуть здійснювати платежі зручним способом. Це позитивно сприятиме розширенню фінансової доступності, впровадженню нових інноваційних інструментів та безготівкових переказів шляхом збільшення та збільшення частки безготівкових переказів, - зазначив Володимир Нагорнюк, директор Департаменту інформаційних технологій НБУ [2].

Робота СЕП у режимі 23/7 стане першим етапом переходу на цілодобовий режим роботи, в той час як зараз працює Національний банк. Йдеться про те, що переходить від поточного до наступного банківського дня, здійснюючи миттєво, без призупинення прийому платежів у СЕП під час переходу на новий день. Такий режим роботи СЕП планується реалізувати в межах роботи зі створення наступного покоління СЕП (СЕП-4) разом із перехідними системами на міжнародному рівні ISO 20022.

Як повідомлялося, до цього СЕП було запропоновано приймати міжбанківські виплати з 8:30 до 19:00 у робочі дні та не функціонувати в світлих та вихідних днях. Система електронних платежів Національного банку - державна банківська платіжна система, що забезпечує проведення міжбанківського переказу через рахунки, відкриті в Національному банку України [2].

Збільшення використання платіжних систем неминуче, оскільки вони мають дуже важливі і незаперечні переваги, такі як:

— доступність – будь-який користувач має можливість безкоштовно відкрити власний електронний рахунок;

— простота використання – для відкриття та використання електронного рахунку не потрібно яких-небудь спеціальних знань, всі наступні дії інтуїтивно зрозумілі;

— мобільність – незалежно від місця свого знаходження користувач може здійснювати зі своїм рахунком будь-які фінансові операції;

— оперативність – переказ коштів з рахунку на рахунок відбувається протягом декількох секунд;

— безпека – передача інформації ведеться з використанням SSL протоколу з кодовим ключем 128-bit або іншими криптографічними алгоритмами.

— На сьогодні для здійснення платіжних операцій у мережі Інтернет можна скористатися такими платіжними системами:

— українські: ГлобалМані, EasyPay, LiqPay, iPay.ua, Простір;

— міжнародні: PayPal, Skrill, Neteller, Payoneer, Perfect Money, ChronoPay, WebMoney, Яндекс.Деньги, Qiwi, RBK Money, PayCash.

При видачі пластикової картки клієнту здійснюється її персоналізація. На карту заносяться дані, що дозволяють зробити ідентифікацію карти і її власника, а також можливість здійснити перевірку платоспроможності картки клієнта, коли проводиться видача готівки або при прийомі її до оплати. Процес затвердження продажу або видачі готівки по кредитній карті називається авторизацією. Для її проведення точка обслуговування робить запит до платіжної системи про підтвердження повноважень пред'явника картки і його фінансових можливостей. Технологія авторизації кредитних карт залежить від технічної оснащеності точки обслуговування, схеми платіжної системи і типу картки.

Більш простий механізм організації платежу передбачає попередню реєстрацію власника пластикової карти на авторизационном сервері. Тримач банківської кредитної картки (Visa, MasterCard, Diners Club, JCB і т.д.) повинен зареєструватися в платіжній системі Cyber Plat, вказуючи персональні дані (прізвище, ім'я, по батькові, паспортні дані, адреса електронної пошти, телефон, поштова адреса) і параметри своєї карти (назва платіжної системи, дату закінчення дії, ім'я власника). Інформація про карту передається в захищеному вигляді лише в банк і не надається іншим учасникам процесу.

Процедура покупки товарів в магазинах для користувачів, зареєстрованих на сервері авторизації, здійснюється за такою технологією (рис. 1.1).

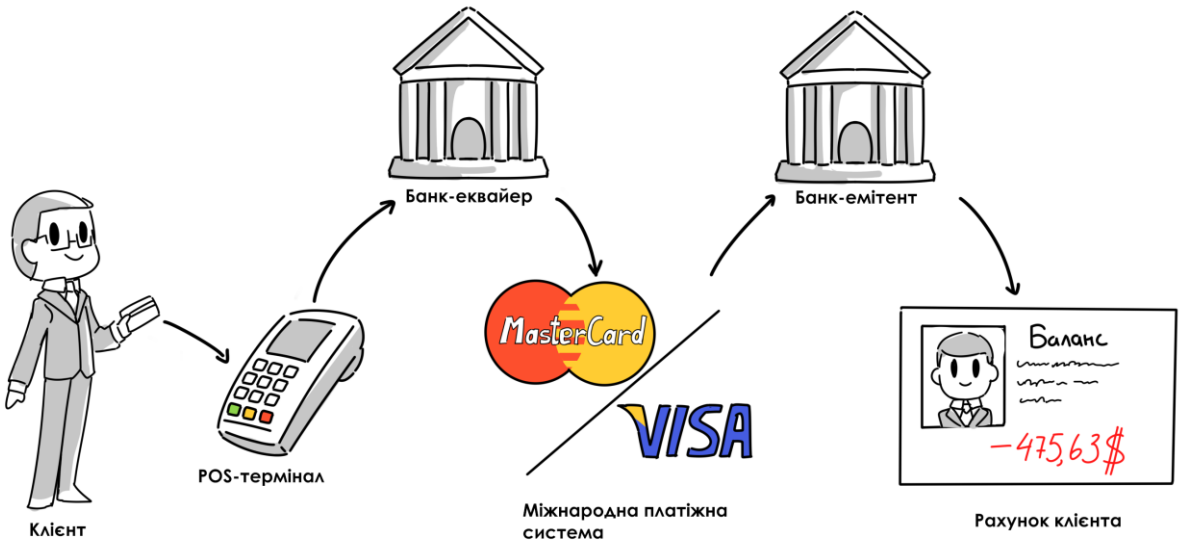

Рис. 1.1 Процедура покупки товарів в магазині.

1.2. Характеристика сучасних ризиків та загроз СЕП

Інформація є ресурсом, який подібно іншим важливим бізнес-ресурсам, є суттєвим для бізнесу організації і тому потребує відповідного захисту. Це суттєво важливо у все більш взаємопов’язаному діловому середовищі. Внаслідок цієї зростаючої взаємопов’язаності інформація тепер наражається на зростаючу

кількість і більшу різноманітність загроз та вразливостей [18, 19]. Розвиток інформаційних технологій, глобальної мережі Інтернет, а також стрімке зростання обчислювальних можливостей комп'ютерних систем, швидке зростання обсягів оброблюваних даних у сучасних системах електронних платежів комерційного банку, поява нових форм електронних послуг, запропонованих систем електронних платежів, висувають нові вимоги до надійності і забезпеченню безпеки у внутрішньо платіжних банківських системах.

На сьогоднішній день не існує науково обґрунтованої концепції й механізмів забезпечення фінансової безпеки банківської діяльності національної платіжної системи в цілому [5]. Система електронних платежів являє собою сукупність правил, організаційних заходів, програмно-технічних засобів, які використовуються банком для забезпечення здійснення розрахунків у межах України між банками як за дорученнями клієнтів банків, так і за зобов’язаннями банків. СЕП виконує міжбанківський переказ у файловому режимі та в режимі реального часу [5]. Дана система належить до багаторівневих критичних систем, тому що її відмова, відступ від обмежень, що задаються, або зміни в роботі підсистеми можуть викликати серйозні наслідки або привести до краха всієї системи в цілому.

У відповідності до СОУ Н НБУ 65.1 СУІБ 1.0:2010 який відповідає стандарту ISO/IEC 27001:2005 Information technology – Security techniques – Information security management systems – Requirements Національний банк України має власний досвід створення платіжних та інформаційних систем. Зокрема Національним банком України було створено Систему електронних платежів Національного банку України (СЕП НБУ), яка визначена законодавством України як державна система міжбанківських розрахунків, та систему роздрібних платежів із використанням платіжних карток – Національну систему масових електронних платежів [18, 19]. Для захисту платіжних повідомлень використовується система захищеної електронної пошти (СЗЕП), призначена для обміну електронними повідомленнями у форматі SMF-70 через

мережу передачі даних довільного типу відповідно до критеріїв НД ТЗІ 2.5-004-99 [8]. Загальна структура підсистеми захисту інформації під ВПБС і можливих загроз на окремі її складові наведені на (рис. 1.2). Національна система масових електронних платежів складається з: регіональних процесінгових центрів (РПЦ) в обласних управліннях НБУ або комерційних підприємствах – до 25 РПЦ на всю Україну; банків-емітентів і банків-еквайєрів НСМЕП зі своїми банківськими підсистемами, торгівельною інфраструктурою та інфраструктурою сфери послуг. Загальна структура НСМЕП включає в себе такі основні елементи (рис. 1.2, 1.3).

1. Центр системної ініціалізації та системної персоналізації (установа НБУ).

2. Розрахунковий банк (РБ) системи на базі Головного управління НБУ. Схема розрахунків – клірингова.

3. Головний та регіональні процесінгові центри (ГПЦ та РПЦ) в обласних управліннях НБУ або комерційних установах (до 25 РПЦ на всю Україну). Вони виконують обробку міжбанківських транзакцій, розрахунок клірингу, керування системою.

4. Банки-емітенти і банки-еквайєри НСМЕП із своїми банківськими системами, торгівельною інфраструктурою та інфраструктурою сфери послуг.

5. Користувачі карток – фізичні та юридичні особи.

6. Картки на інтегрованих схемах (або смарткартки). Для забезпечення захисту банківської інформації в ВПБС на різних рівнях використовуються криптографічні механізми, однак, бурхливе зростання обчислювальної техніки, створення систем і технологій кібертерроризму призводить до появи нових загроз (активних і пасивних атак) і злому підсистеми захисту ВПБС. Під загрозою розуміється сукупність умов чинників, що створюють небезпеку несанкціонованого, в тому числі випадкового, доступу до інформації, результатом якого може стати знищення, зміна, блокування, копіювання, поширення інформації, загальна класифікація загроз наведена на рис. 1.4 [23].

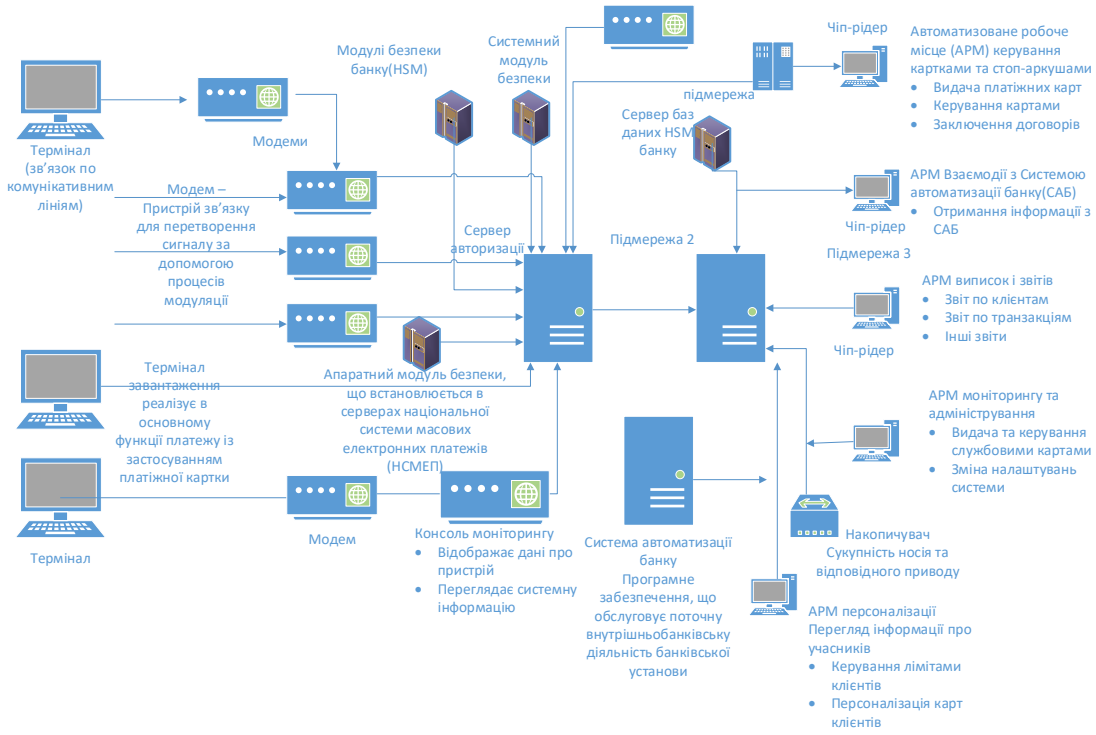

Рис. 1.2 Структура системи електронних платежів банку

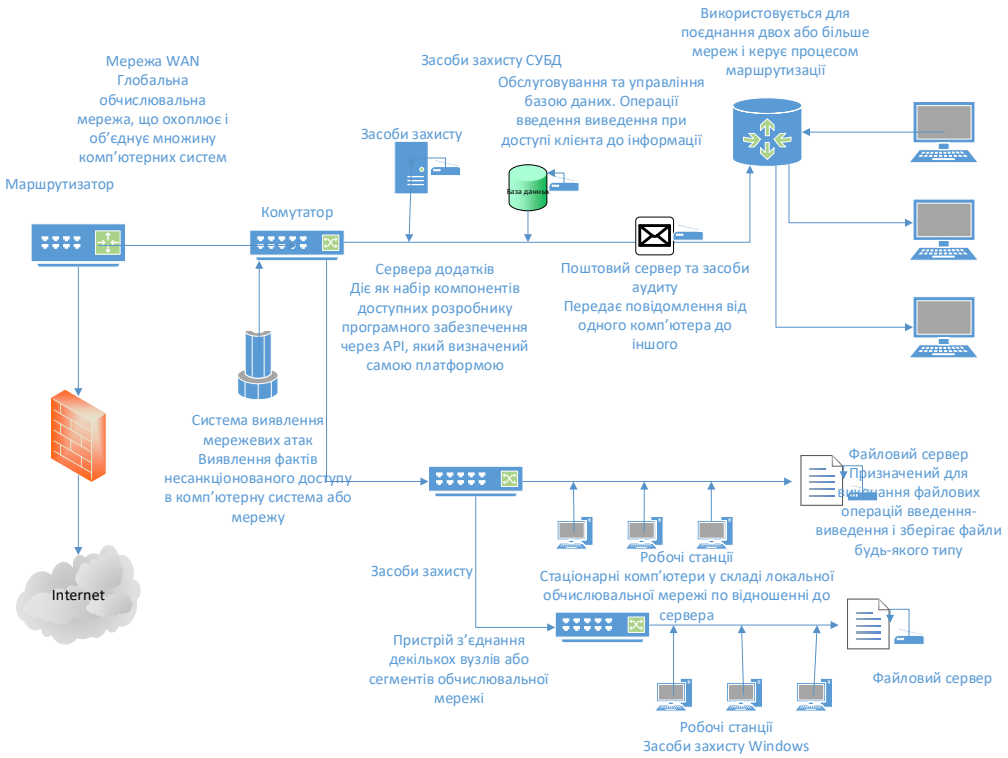

Рис. 1.3 Структурна схема підсистеми захисту інформації в СЕП

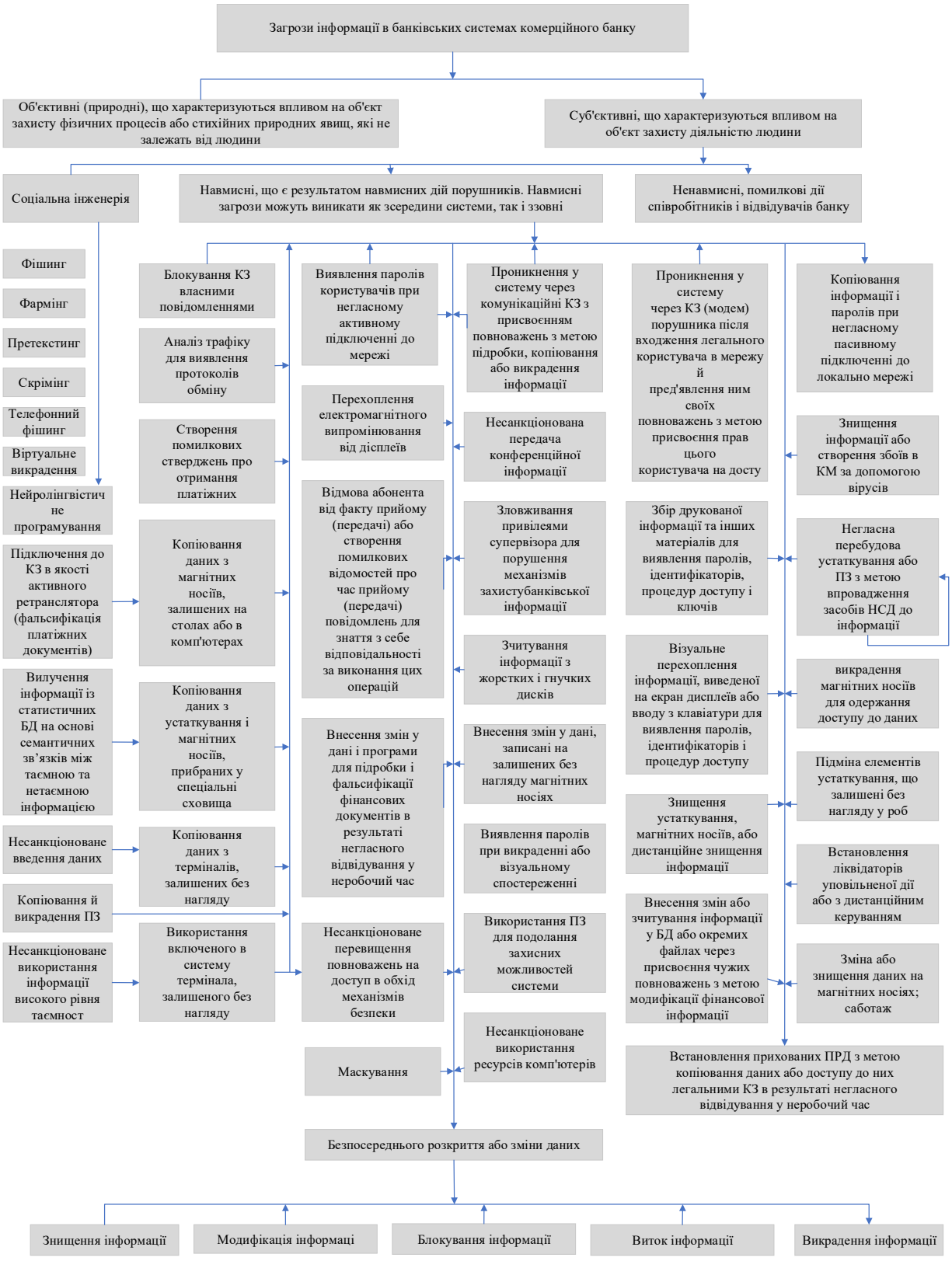

Рис. 1.4 Загальна класифікація загроз комерційного банку

1.3. Дослідження методів та підходів щодо захисту СЕП

Одним з найбільш вразливих місць в системі електронних платежів є передача платежів та інших повідомлень між банками, між банком і банкоматом, між банком і клієнтом. Результати дослідження атак на сучасні КМ з боку компанії «Arbor Networks» показані на рис. 1.5, 1.6, 1.7.

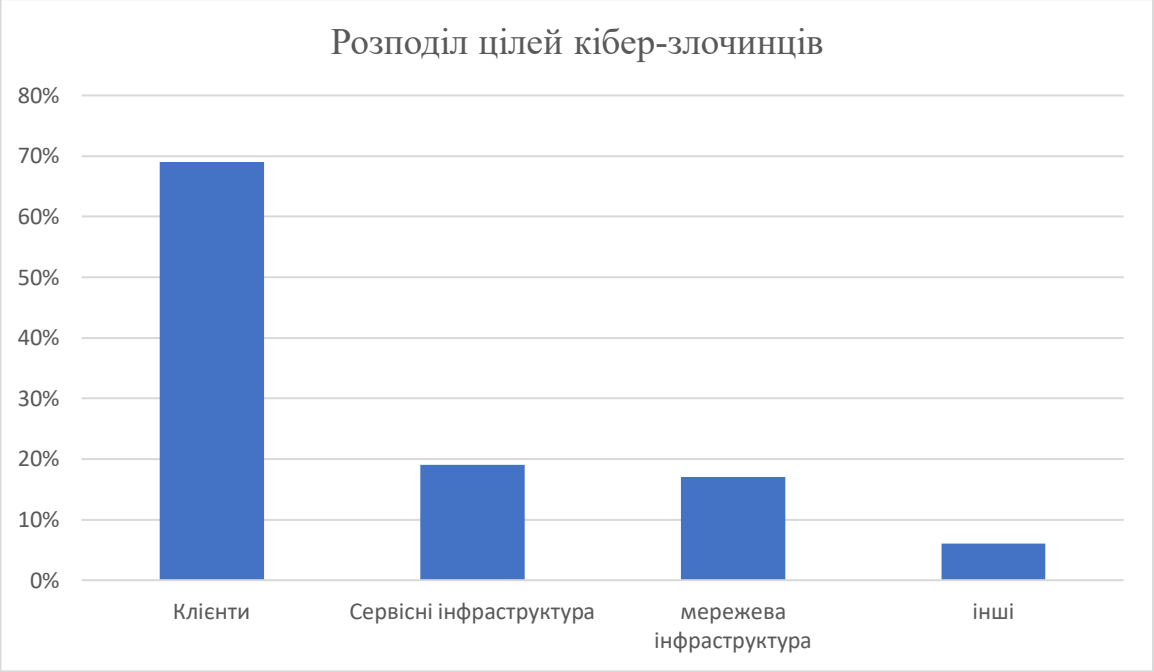

Рис. 1.5 Розподіл цілей кібер-злочинців

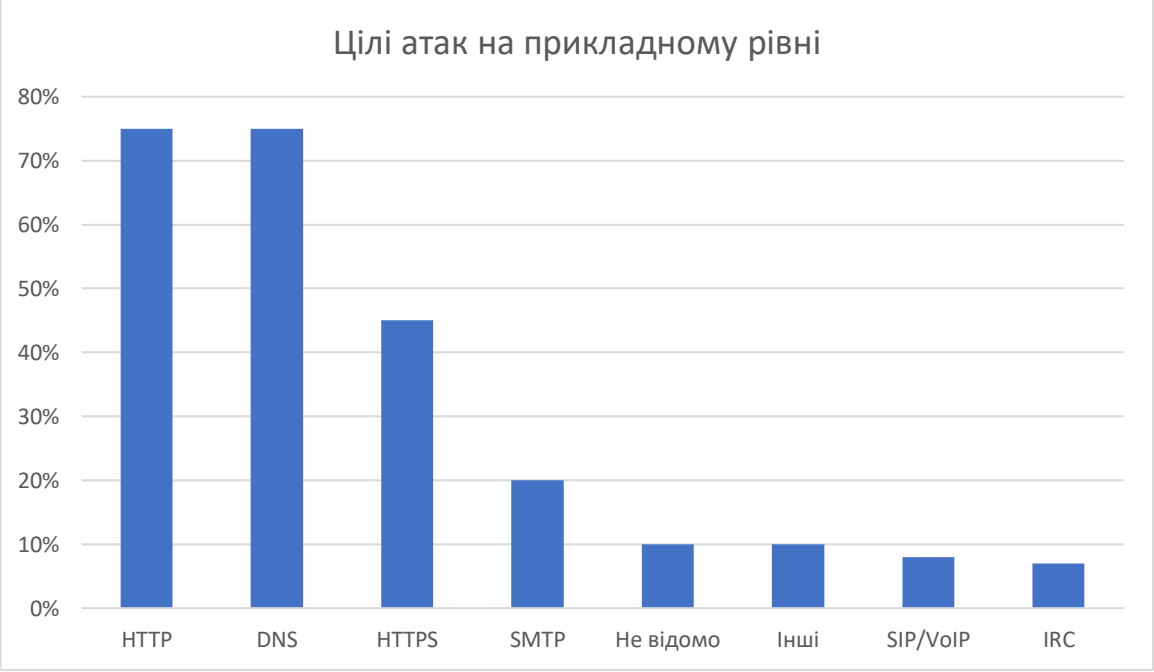

Рис. 1.6 Цілі атак на прикладному рівні

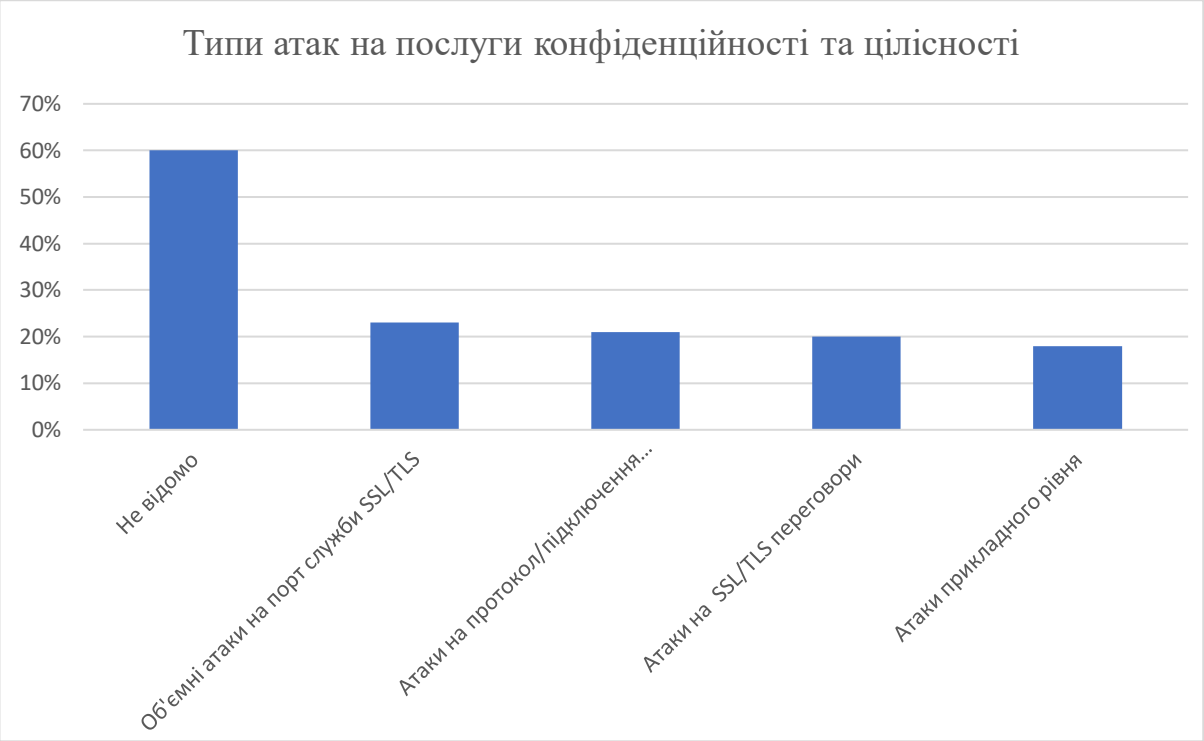

Рис. 1.7 Типи атак на послуги конфіденційності та цілісності

Відповідно до СОУ Н НБУ 65.1 СУІБ 1.0:2010 року керівництво комерційних банків запропонувало процесний підхід до управління інформаційною безпекою, закликає користувачів підкреслюючи важливість:

а) розуміння вимог до інформаційної безпеки організації і необхідності розробки політик і цілей інформаційної безпеки;

б) впровадження заходів безпеки і забезпечення їх функціонування для управління ризиками інформаційної безпеки організації в контексті загальних бізнес-ризиків організації;

с) моніторинг і аналіз продуктивності та ефективності СУІБ (системи управління інформаційної безпеки);

г) постійне поліпшення на основі об'єктивних вимірювань. Цей стандарт приймає Закон про перевірку поведінки “Плануй-Виконуй-Перевіряй-Дій”(«Plan-Do-Check-Act»), далі іменований «PDCA», який використовується для структурування всіх процесів СУІБ. СУІБ забезпечує вибір адекватних і взаємопов'язаних заходів безпеки, які забезпечують захист інформаційних ресурсів СУІБ і конфіденційність зацікавлених сторін. Основні етапи побудови

банку SMSB показані на рис. 1.8. Аналіз основних вимог стандарту визначає основні функції системи управління ІБ, які показані на рис. 1.9 [18, 19, 20].

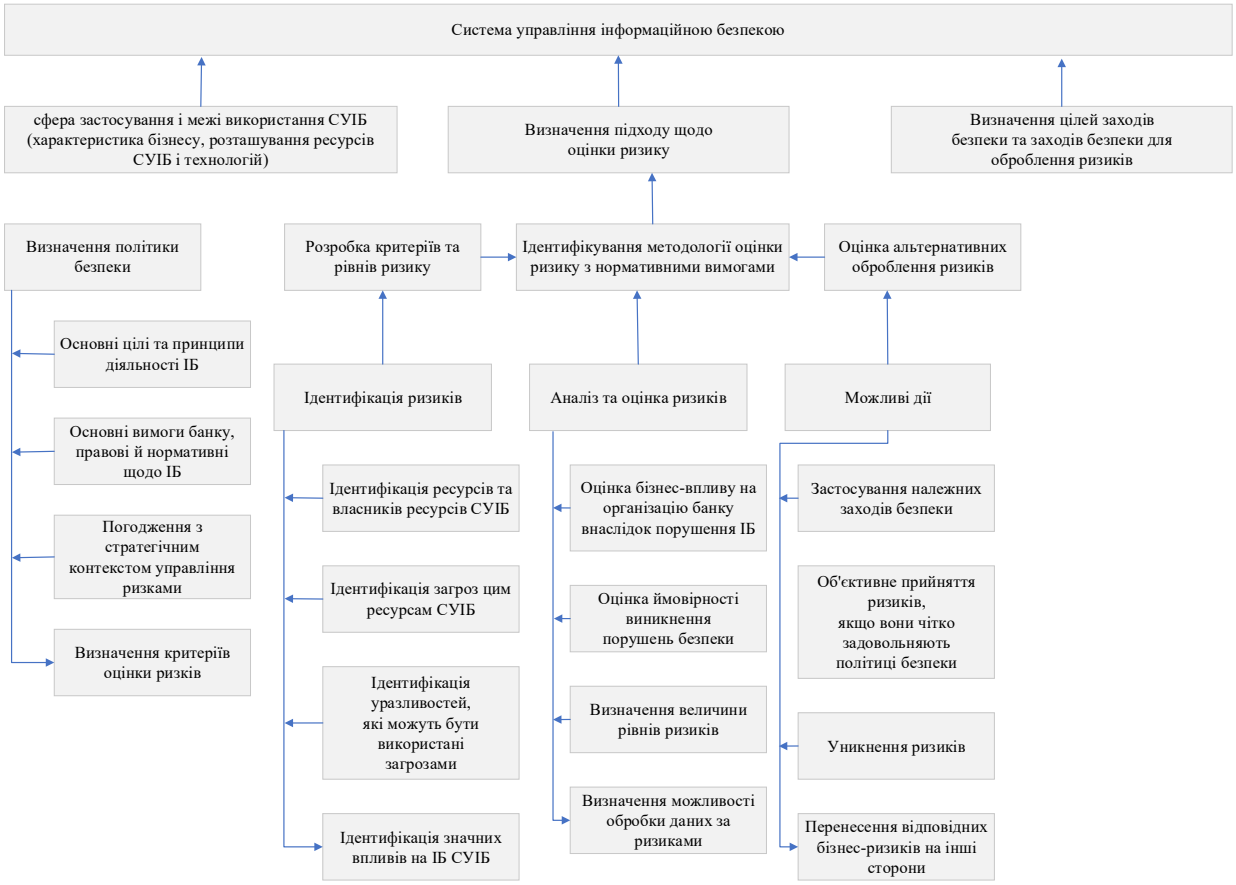

Рис. 1.8 Етапи побудови СУІБ комерційного банку

Відповідно до міжнародних стандартів ISO 7498, ISO/IEC 10181 для Для забезпечення необхідних показників безпеки існує п’ять основних загальних послуг, основними з яких є лише дві: автентичність та цілісність, а також для їх забезпечення використовуються механізми безпеки, більшість з яких реалізовані на основі криптографічних методів перетворення інформації.

Основні механізми забезпечення цілісності та надійності інформації у ВПС на різних рівнях базуються на використанні блочно-симетричних стандартів шифру (3DES, ГОСТ 28147-89). Прикладом програмної реалізації цих механізмів є програмне забезпечення для криптографічного захисту інформації "Грифон-Б" і "Грифон-Л", призначеного для криптографічного захисту конфіденційної інформації в автоматизованих банківських системах і використовується для

обміну інформацією в корпоративній мережі банку з клієнтами, які працюють із системою. Клієнт-Банк ", в системах обслуговування пластикових карток. [6 – 9, 20, 21].

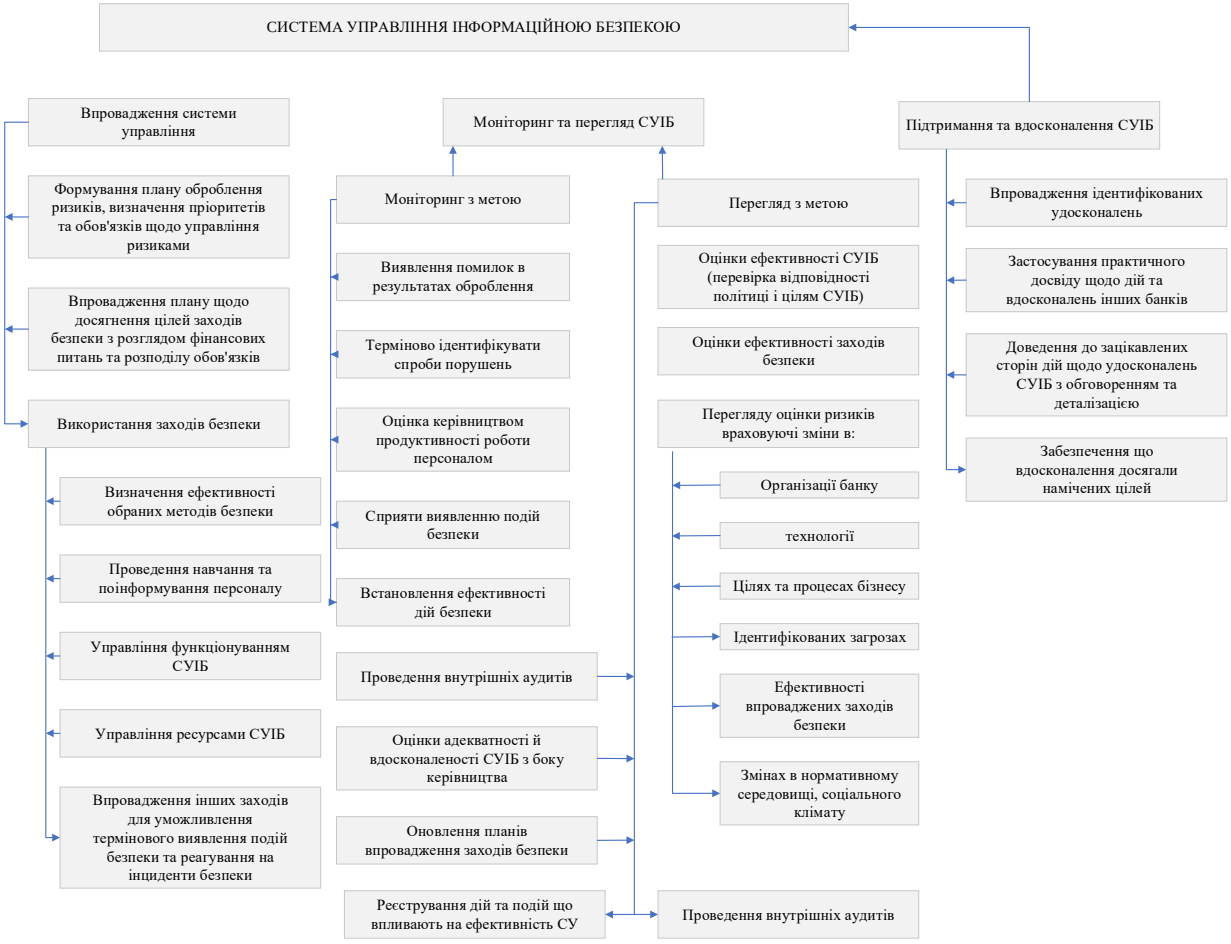

Рис. 1.9 Основні функції СУІБ комерційного банку

Програмний засіб криптографічного захисту інформації “Грифон-Л” [20] призначене для використання у сфері банківської діяльності, зокрема, для обміну конфіденційною (у т.ч. фінансової) інформацією всередині корпоративної мережі банку, з клієнтами, які працюють за системою “Клієнт-Банк”, в системах обслуговування пластикових карт та ін. Бібліотека процедур криптографічного захисту інформації “Тайфун-PKCS#11” містить процедури, призначені для забезпечення цілісності та конфіденційності інформації, здійснення автентифікації відправника за допомогою криптографічних механізмів захисту (електронний цифровий підпис, шифрування, імітація та хеш-функції) шляхом

вбудовування в конкретні прикладні системи [14]. Процедури, які входять до складу бібліотеки реалізують: шифрування / дешифрування даних за алгоритмом ГОСТ 28147-89; вироблення/перевірка імітації за алгоритмом ГОСТ 28147-89; вироблення/перевірка ЕЦП за алгоритмами ДСТУ 4145-2002, ГОСТ 34.310-95, 34.311-95; вироблення ключів шифрування за схемою Діффі-Хеллмана (використовується відкритий розподіл ключів відповідно до вимог ISO 11166-94) [27]. Швидкісні характеристики програмних засобів, що реалізують алгоритми криптографічних перетворень (для ПЕВМ на базі Intel Celeron 2,4 ГГц) [27]:

— швидкість шифрування/ розшифрування даних у режимі простої заміни БСШ ГОСТ 28147-89 не менш 8 Мбайт/с;

— швидкість обчислення хеш-функції даних відповідно до ГОСТ 34.311-95 не менш 3 Мбайт/с;

— час вироблення ЕЦП відповідно до ГОСТ 34.310-95 при довжині ключа 512 біт не більш 0,003 с;

— час перевірки ЕЦП відповідно до ГОСТ 34.310- 95 при довжині ключа 512 біт не більш 0,006 с;

— час вироблення ЕЦП відповідно до ГОСТ 34.310-95 при довжині ключа 1024 біт не більш 0,01 с;

— час перевірки ЕЦП відповідно до ГОСТ 34.310- 95 при довжині ключа 1024 біт не більш 0,02 с;

— час вироблення ЕЦП (з обчисленням перед підпису) згідно ДСТУ 4145-2002 для основного поля степеня 163 не більш 0,0068 с;

— час перевірки ЕЦП згідно ДСТУ 4145-2002 для основного поля степеня 163 не більш 0,013 с. Криптографічні перетворення в бібліотеці “Тайфун-PKI PKCS#11” реалізуються з використанням об’єктної бібліотеки програмних процедур криптографічного захисту інформації “Тайфун-W32” версії 2.01 [27].

Система захищеної електронної пошти “Бриз” призначена для здійснення обміну електронними повідомленнями у форматі SMF-70, захищеними з

використанням механізмів криптографічного захисту (електронний цифровий підпис, шифрування/розшифрування, вироблення іміто-вставок), між клієнтами електронної пошти (ЕП), зареєстрованими на вузлах ЕП, через мережу передачі даних довільного типу й відповідає критеріям НД ТЗІ 2.5- 004-99 [13]. Проведений аналіз стандартів показав, що для забезпечення конфіденційності, автентичності та цілісності використовується БСШ Російський ГОСТ 28147-89 – застарілий алгоритм симетричного шифрування, розроблений в 1989 році, крім того крипостійкість БСШ, ґрунтується на крипостійкості S-боксів, які для даного шифру "надходять" з Російської Федерації, що істотно впливає на безпеку ВПС в цілому. На сьогоднішній день в Україні немає національних стандартів на алгоритми БСШ та формування хеш-функцій, які використовуються в електронних цифрових підписах. Так національний стандарт ЕЦП ДСТУ 4145 використовується з Російським стандартом формування хеш-коду ГОСТ 34.311-95. Розроблені національні стандарти ДСТУ-7624-2014 "Інформаційні технології. Криптографічний захист інформації. Алгоритм симетричного блокового перетворення" – встановлює криптографічний алгоритм симетричного блокового перетворення для забезпечення конфіденційності та цілісності (як додаткової послуги) інформації під час її обробки. Стандарт пропонується використовувати під час розробки засобів криптографічного захисту інформації в інформаційних, телекомунікаційних та інформаційно-телекомунікаційних системах, а також при модернізації діючих систем для заміни ДСТУ ГОСТ 28147:2009 дозволять суттєво змінити рівень інформаційної безпеки в СЕП; національний стандарт ДСТУ 7564-2014 "Інформаційні технології. Криптографічний захист інформації. Функція хешування" установлює алгоритм обчислення хеш-значення для послідовностей двійкових символів, що застосовують в криптографічних методах захисту, для забезпечення цілісності та автентичності інформації під час її передавання, оброблення і зберігання, зокрема під час використання електронного цифрового підпису, що визначений ДСТУ 4145. Стандарт використовують під час розробки засобів криптографічного захисту інформації в інформаційно-

телекомунікаційних системах, а також при модернізації діючих систем для заміни функції хешування згідно з ГОСТ 34.311 [16, 17]. На рис. 1.10 наведений взаємозв’язок між механізмами і вживаними стандартами у підсистемі безпеки ВПС.

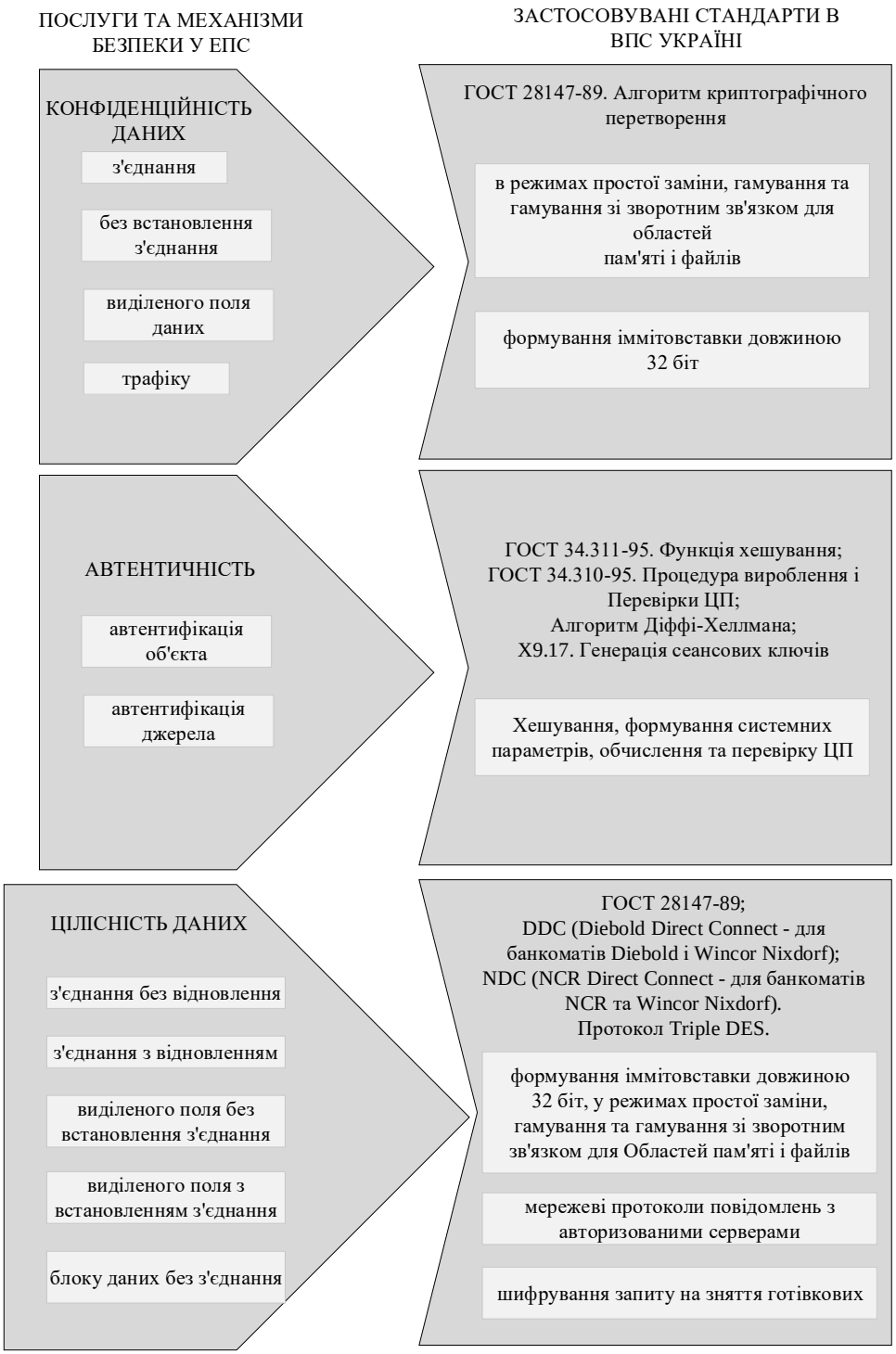

Рис. 1.10 Взаємозв’язок між механізмами і стандартами безпеки ВПС

Одним з нині існуючих методів забезпечення безпеки це протоколи безпечної транзакції, один з таких це SET.

SET (Security Electronics Transaction). SET заснований на використанні цифрових сертифікатів відповідно до стандарту X.509. Secure Transaction Protocol SET - це стандарт, розроблений MasterCard і VISA при значній участі IBM, GlobeSet та інших партнерів. Це дозволяє користувачам купувати товари в Інтернеті, використовуючи найбільш безпечний в даний час механізм оплати. SET - це відкритий стандартний багатосторонній протокол для безпечних платежів з використанням пластикових карт в Інтернеті. SET забезпечує перехресну аутентифікацію облікового запису власника картки, продавця і банку продавця для перевірки готовності до оплати товару, цілісності і конфіденційності повідомлення, шифрування цінних та вразливих даних. Тому SET можна назвати стандартною технологією або системою протоколів для безпечних платежів з використанням пластикових карт через Інтернет.

SET дозволяє споживачам і продавцям перевіряти справжність усіх учасників транзакції, що відбувається в мережі, з використанням криптографії, в тому числі з використанням цифрових сертифікатів.

Обсяг потенційних продажів в сфері електронної комерції обмежується досягненням необхідного рівня інформаційної безпеки, що забезпечується покупцями, продавцями і фінансовими установами, стурбованими безпекою платежів через Інтернет. Як згадувалося раніше, основними завданнями захисту інформації є забезпечення її доступності, конфіденційності, цілісності та юридичної значимості. SET, на відміну від інших протоколів, дозволяє вирішувати ці проблеми інформаційної безпеки.

В результаті того, що багато компаній розробляють власне програмне забезпечення для електронної комерції, виникає інша проблема. При використанні цього програмного забезпечення у всіх учасників операції повинні бути однакові додатки, що практично неможливо. Отже, необхідний спосіб забезпечити механізм взаємодії додатків різних розробників.

У зв'язку з вищезгаданими проблемами VISA і MasterCard разом з іншими компаніями, що займаються технічними проблемами (наприклад, IBM, яка є ключовим учасником в розробці протоколу SET), визначили специфікацію і набір протоколів SET стандарту. Ця відкрита специфікація швидко стала стандартом де-факто для електронної комерції. У цій характеристиці шифрування інформації забезпечує її конфіденційність. Цифрові підписи і сертифікати забезпечують ідентифікацію та аутентифікацію (аутентифікацію) учасників транзакції. Цифровий підпис також використовується для забезпечення цілісності даних. Відкритий набір протоколів використовується для забезпечення взаємодії між реалізаціями різних постачальників.

SET надає наступні особливі вимоги для захисту транзакцій електронної торгівлі:

— Конфіденційність платежів і конфіденційність інформації про замовлення, переданої разом з платіжними даними;

— Підтримка цілісності цих платежів; цілісність забезпечується цифровим підписом;

— Спеціальна криптографія з відкритим ключем для аутентифікації;

— Аутентифікація кредитної картки, яка забезпечується електронним підписом і сертифікатами власника картки;

— Аутентифікація продавця і можливість прийому платежів пластиковими картами з використанням електронного підпису і сертифікатів продавця;

— Підтвердження того, що банк продавця є активною організацією, яка може приймати платежі з пластикових карт за допомогою зв'язку з процесінговою системою; це підтвердження забезпечується ЕЦП і сертифікатами банку продавця;

— Готовність оплачувати транзакції в результаті аутентифікації сертифіката відкритого ключа для всіх сторін;

— Безпека передачі даних за рахунок переважного використання криптографії.

Основною перевагою SET перед багатьма існуючими системами захисту інформації є використання цифрових сертифікатів (стандарт X.509, версія 3), які пов'язують власника картки, продавця і банк продавця з іншими учасниками платіжних систем банківських установ VISA і MasterCard.

SET дозволяє підтримувати існуючі відносини між банком, власниками карт і продавцями та інтегрується з існуючими системами на основі наступних якостей:

— Відкритий, повністю задокументований стандарт для фінансової індустрії;

— На основі міжнародних стандартів платіжних систем;

— Використовує існуючі технології і правові механізми в фінансовому секторі.

1.4. Висновки до першого розділу

В першому розділі були розглянуті Системи електронних платежів, загальні поняття, та базові принципи цієї теми. Було розглянуто безпеку та загальні загрози електронним коштам. Виявили, що електронні гроші не рівні готівці, а це така валюта, різна для кожного емітента – власника (банку) який випускає електронні кошти. Переглянули різні системи та явища які впливають на безпеку СЕП, які умови для загрози цілісності, які вразливості даної системи, що потрібно покращити, як діяти, та що потрібно змінити. Розглянули класифікацію СЕП. Розглянули типи атак на СЕП. Було ознайомлено з різними стандартами та механізмами захисту Електронно платіжних систем та стандартами, які прийняти стосовно кожного механізму.

СЕП розвивається кожен день, системи захисту повинні вдосконалюватися, як можливо частіше, тому що джерел загрози СЕП збільшується з кожним днем.

РОЗДІЛ 2. Система електронних платежів на базі криптографічних перетворень.

2.1. Опис середовища розробки

Dev cpp або Dev з ++ IDE - середовище розробки на с та с ++, графічне доповнення до компілятора MinGW GCC.

IDE є безкоштовним і з відкритим кодом. Поставляється як інсталятор або портативна версія, що полегшує транспортування середовища програмування з проектами.

Dev-C ++ програма створена для платформ Windows. Мільйони розробників, студентів та дослідників використовують Dev-C ++ з моменту його першого випуску в 1998 році. Він залишається улюбленим інструментом викладання серед університетів та шкіл у всьому світі.

Особливості DevCpp:

- Зручний редактор із підсвічуванням синтаксису, нумерацією рядків, автовідступом тощо.
- Можливість автоматичного заповнення коду для зручності використання та покращення продуктивності.
- Використання закладок в редакторі для швидкої навігації кодом.
- Експорт вихідних файлів або всього проект у HTML або RTF, щоб опублікувати вихідний код на вашому веб-сайті.
- Заготовки кодів та шаблони для вставлення.
- Вбудований менеджер проектів.
- Імпортуват проектів з MS Visual C ++.
- Можливість налаштування асоціації файлів за розширенням - c, cpp, h тощо.
- У навігаторі класів є два варіанти перегляду - перегляд функцій, класів та їх членів для всього проекту та поточного редагованого файлу.

- Гнучкі налаштування робочого столу, редактора та компілятора, безліч різних опцій.
- Використовуваний компілятор Mingw GCC, може працювати з будь-яким компілятором GCC.
- Можливість налагодження проекту - вбудований налагоджувач GDB.
- Є можливість працювати з CVS (завантажується окремо).
- Існує портативна версія програми, яка не потребує встановлення.
- Багатомовний користувальницький інтерфейс з підтримкою російської та української мов.

Тестування та розробка була здійснена на системі з даними характеристиками:

- Windows Home 10
- 64-х бітна система
- Процесор Intel core i5-10300H 2.50 GHz

Для програмної реалізації була обрана мова програмування C++, яка нині є дуже популярною, і до сьогоднішнього дня вважається надійною і стабільною мовою програмування.

У мові C++ є багато бібліотек, які включає в себе різноманітні контейнери та алгоритми, введення – виведення, певні вираження та підтримує багатопочність.

C++ поєднує в собі високорівневі та низькорівневі можливості, що спрощує реалізацію багатьох програм, використовуючи рівні по потребі, залежно від того що потрібно реалізувати.

Ще одна перевага мови C++ це дуже розвинена підтримка об’єктно орієнтованого програмування.

2.2. Опис програмного забезпечення

Для проведення безпечної транзакції через інтернет, щоб дані не були вкрадені, потрібно передавати дані в зашифрованому вигляді, щоб оплату не перенаправили, застосовуються RSA та AES шифрування, де у відправника інформація шифрується його ключами, а у отримувача, дешифрується, порівнюється з початковим значенням, і дозволяє транзакцію у разі співпадіння даних. SET протокол повинен дозволяти перевірити отримувача платежу та відправника платежу, щоб забезпечити безпечну транзакцію грошового переказу між платником та отримувачем.

Цей протокол повинен забезпечити безпеку власників банківських карт на усіх рівнях переказу коштів, за умовами цього протоколу, при утвердженому замовленні, отримувач платежу не може змінювати умови заказу, тобто змінити ціну та картку, це створене для того, щоб грошовий переказ не можна було перенаправити (рис. 2.3).

1.Програма запитує дані відправника

2. Йде шифрування AES

3. Шифрування RSA

4. Програма відправляє зашифроване повідомлення Отримувачу

5. Йде дешифрування AES

6. Дешифрування RSA

7. Порівняння початкового значення з дешифрованим

8.Дозвіл/заборона транзакції

У програмному додатку були застосовані 2 методи шифрування симетричній AES – тобто ключ шифрування і дешифрування один і той самий, та RSA – асиметричний, тобто ключі які дешифрують і шифрують різні.

SET для безпечної транзакції використовує шифрування даних для безпечної транзакції, спочатку дані хешуються, потім шифруються двома шифрами.

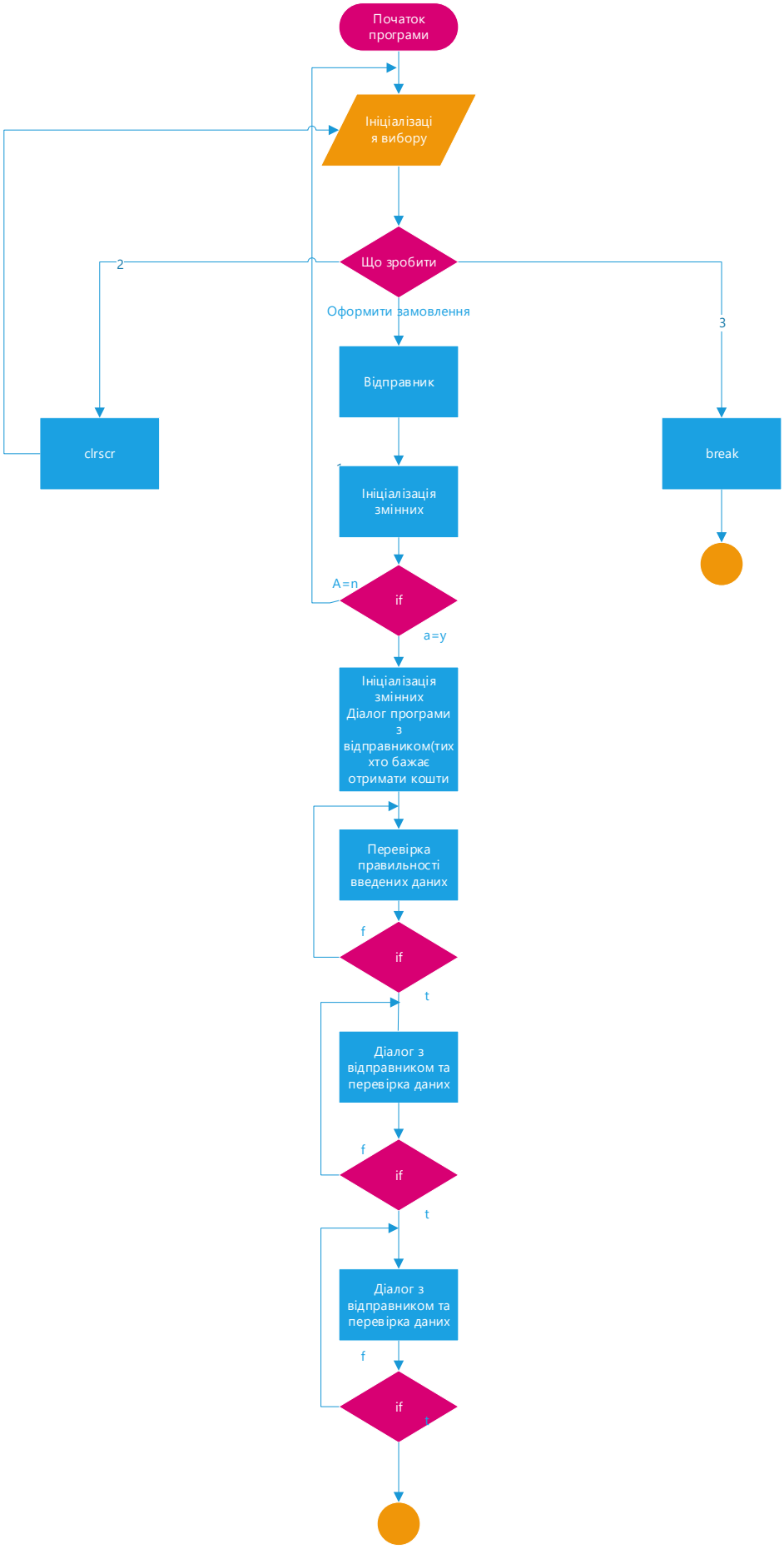

Рис 2.1 Блок-схема програмної реалізації ч.1

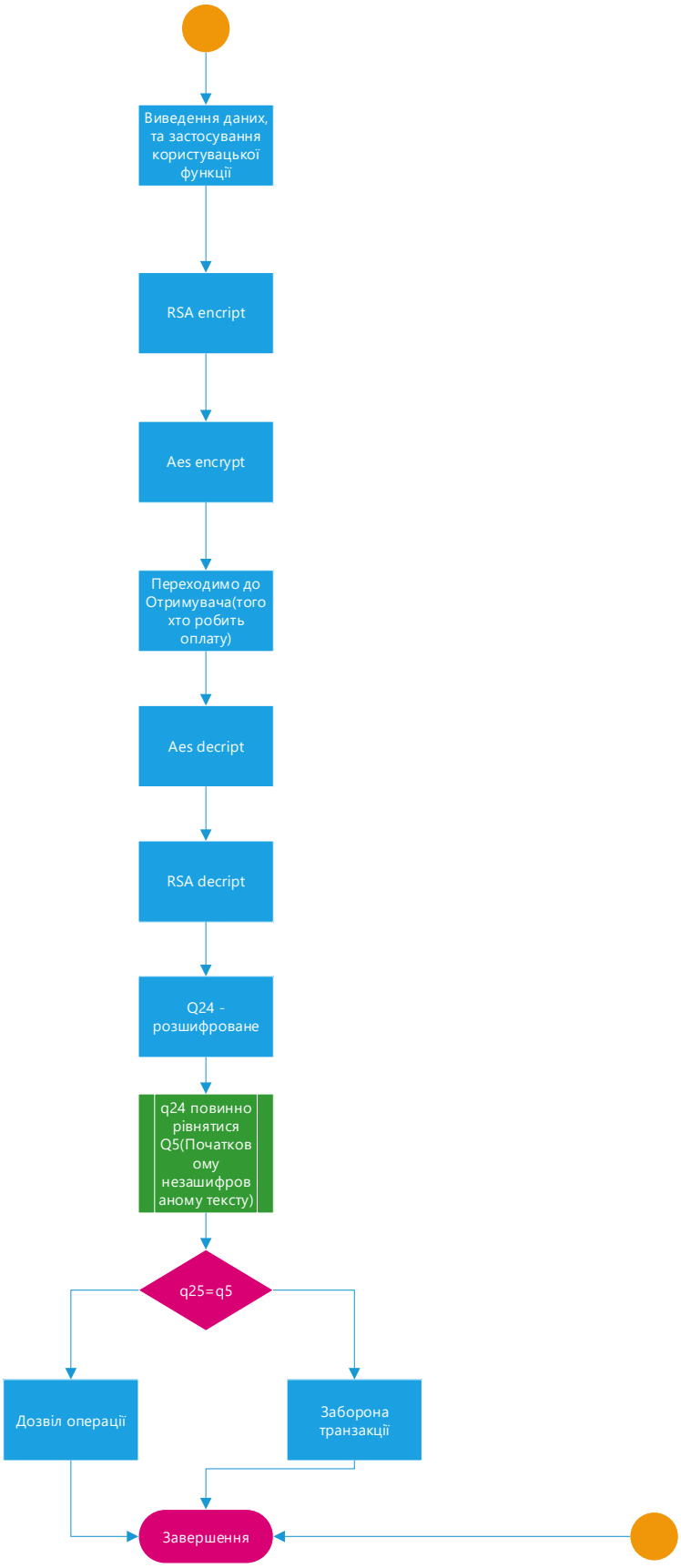

Рис 2.2 Блок-схема програмної реалізації ч.2

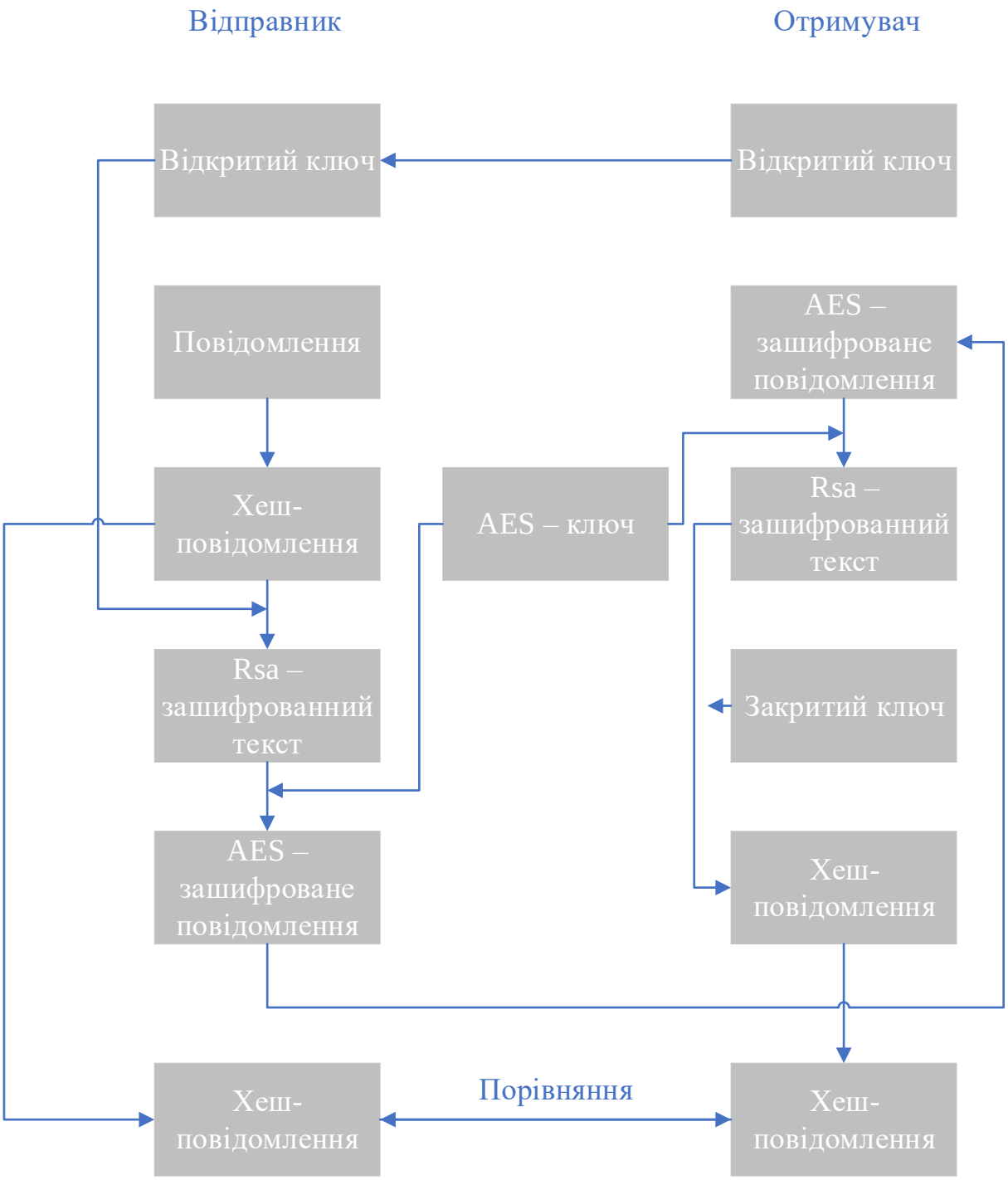

Рис 2.3 Схема роботи програми

Підключення Бібліотек та файлів – заголовків до програми.

В програмну реалізацію були добавлені як стандартні бібліотеки, та файли заголовки, такі як: iostream, math.h, time.h, iomanip, cstring, fstream, sstream, string.h, cstdio, conio.h, cstdlib, windows.h, string.

Так і користувацький файл, такий як #include "str.h", він був застосований для того, щоб реалізувати шифрування AES.

Застосовувалося в програмному модулі симетричне шифрування AES та асиметричне RSA. І хочу AES є симетричним він залишається дуже надійним шифром, який в США на державному рівні, між державними установами використовують ключ 128 – бітний, а на інформації з грифом «дуже таємно» використовують 196 та 256 ключі, що збільшує надійність збереження інформації таємною.

Функція Хешування виконується за допомогою стандартного XOR.

Функція хешування виглядає ось так: h=H(M), де M – повідомлення будь якої довжини.

Функція хешування повинна мати наступні властивості щоб вважатися готовою до використання:

- ❖ Можливість використання функції хешування для повідомлень будь якої довжини.
- ❖ На виході функція повинна давати значення певної довжини, зазвичай 2 символи.
- ❖ Реалізація алгоритму не має бути важкою для розуміння, як апаратна так і програмна частина.
- ❖ Так як алгоритм відомий, то можна його вдосконалити, додавши ефект рандомізації – циклічного одно-бітового зсуву кожного блоку повідомлення.

Даний Хеш не використовує ключ, він поєднується між собою і використовує зсув (рис. 2.4).

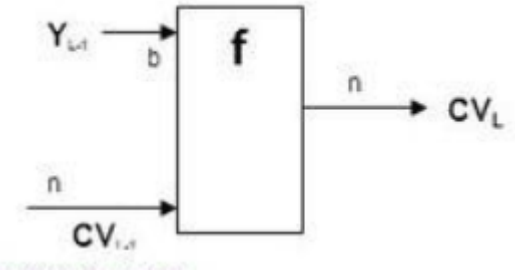

Рис. 2.4 схема реалізації Хеш-функції.

Y – початкові дані(повідомлення)

CV – Зсув (однобітовий)

f – функція стискання

CV_L – кінцевий результат

b – довжина блока який вводимо

n – довжина хеш - коду

В програмній реалізації результат, тобто довжина хеш – коду дорівнює два символи. Було обрано 128-бітну розрядність, щоб на виході отримати 8 символів. Вичислюється скільки ітерацій хешування буде проходити для розрахунку хеш-символів. Символи додаються за допомогою XOR, та відбувається зсув. Тож хоча алгоритм хешування XOR відомий, проте дешифрувати з використанням зсуву займе достатньо багато часу, і платіж перестане бути дійсним, тому хешування методом XOR і було обрано у дану програмну реалізацію.

Шифрування RSA

Безпека RSA алгоритму побудована на складності факторизації цілих чисел – тобто розкладання числа на прості множники, на числа які можуть ділитися на себе і на 1, наприклад 1,3,5,7,11 и т.д.

Алгоритм RSA (рис. 2.5) використовує для шифрування і дешифрування різні ключі.

Алгоритм RSA використовується для того, щоб ідентифікувати користувачів транзакції, для забезпечення збереження інформації та проведення безпечної оплати. Іноді ті хто оплачує бажають залишитися анонімними, так як транзакції і покупки бувають різними, і саме для таких випадків протокол безпечної транзакції в інтернеті є незамінною річчю яку користувачі можуть використати для власної безпеки себе і своїх даних, щоб перевірити того хто оплачує, щоб дані не були викрадені, так і збоку платника, щоб він був впевнений, що його дані залишаться лише при ньому, він може перевірити безпечність шляху транзакції, та переконатися, в тому, чи безпечно передавати свої дані.

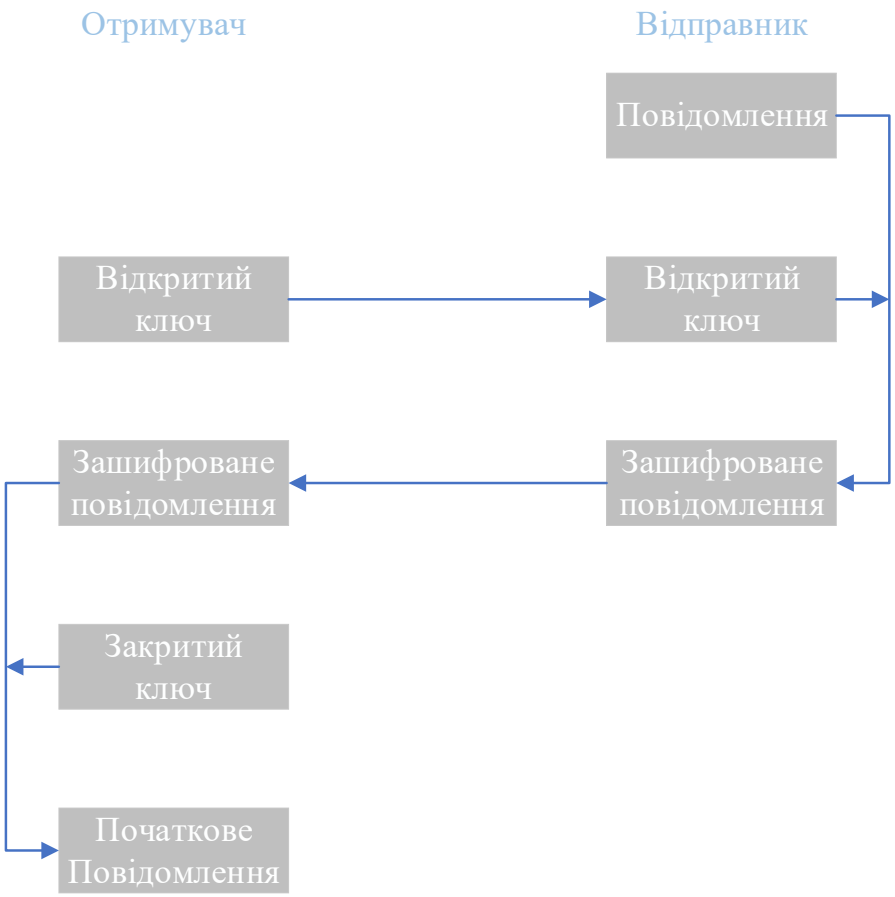

Рис. 2.5 Алгоритм RSA

Шифрується текст відкритим ключем, який передається по надійному каналу, але він не обов'язково повинен бути секретним так як, розшифрувати текст зашифрованих даним відкритим ключем використавши той самий ключ – неможливо. Для дешифрування повідомлення використовується закритий ключ, який нікому не повідомляється, і зберігається у власника. Для програмного додатку було використано все готові ключі – відкритий та закритий, які до цього було знайдено за алгоритмом:

- ❖ Обирається 2 простих числа p та q
- ❖ Обчислюється добуток цих чисел n=p*q
- ❖ Обчислюється функція Ейлера f(n)=(p-1)(q-1)
- ❖ Обирається число e, таке, що воно більше одного та менше f(n), також воно повинно бути взаємно простим з f(n)
- ❖ Обчислюється d, яке розраховується з рівняння (d*e)%f(n), також воно повинно бути взаємно простим з e.

Два ключі формується з чисел n, e і d:

Відкритий ключ формується з чисел {n,e} – його можна відправляти, щоб отримати зашифроване повідомлення

Закритий ключ формується з чисел: {n,d}- ключове значення d, так як n використовується й у відкритому ключі, а значення d нікому не повідомлялося, і може бути будь яким якщо воно виконує зазначені умови.

Шифрування повідомлення(M): $(M^e)\%n$

Дешифрування повідомлення: $(M^d)\%n$

В програмній реалізації кожен символ повідомлення при шифруванні підноситься до степеню e, та береться по модулю n.

При дешифруванні виконується аналогічна формула, тільки замість відкритого значення e, використовується d.

В програмній реалізації використовувався шифр RSA бо він є надійним, тому що закритий ключ є лише у власника, а передається тільки відкритий, яким можна зашифрувати, але не дешифрувати повідомлення. Тож обирався даний метод шифрування, через його практичну легкість розуміння користувачами, та досить високу крипто-стійкість, для його реалізації, потрібно лише придумати 2 прості числа, тож отримати відкритий і закритий ключ двом особам доволі легко.

Алгоритм шифрування та дешифрування AES

Це симетричний алгоритм блокового шифрування(розмір блока 128 біт). Цей алгоритм прийшов на зміну його попередника алгоритму DES, так як в новому алгоритмі можна було використовувати ключ довжиною 128, 192, 256 біт, на відміну від попередника де можна було використати лише 56-бітний ключ.

Для 128 біт – а саме такий було використано у програмному додатку, алгоритм має 10 раундів:

- SUBBytes
- SHIFTRows
- MIXColomns(не використовується у 10 раунді)
- ADDRoundKey

SUBBytes

Йде обробка кожного байту, проводячи нелінійну заміну байтів за допомогою таблиці замін (S-box).

S-box складається з двох кроків створення.

Спочатку отримується зворотне число з поля Галуа $GF(2^8)$ – скінченне поле множин елементів, найменше поле GF(2) - F_2 містить тільки 2 елементи 0 та 1. Операції які проводяться у полі Галуа – звичайні за винятком 1+1 = 0.

Потім до кожного байту з яких складається S-box (рис. 2.8) застосовується XOR операція яка має вигляд:

$B`_i = b_i \text{ XOR } b_{(i+4) \bmod 8} \text{ XOR } b_{(i+6) \bmod 8} \text{ XOR } b_{(i+7) \bmod 8} \text{ XOR } c_i$,

Де $0 \leq i < 8$, і де b_i є і-тий біт b, а c_i – і-тий біт константи.

$c = 63_{16} = 99_{10} = 01100011_2$. Таким чином забезпечується захист від атак, заснованих на простих алгебраїчних властивостях.

SHIFTRows

На цьому етапі рядки таблиці циклічно зсуваються на n байтів по горизонталі, де n – номер рядка починаючи з нульового .

Фактично – SHIFTRows це проста перестановка байтів в таблиці відповідних розмірів.

MIXColomns(не використовується у 10 раунді)

На цьому етапі чотири байти кожної колонки змішуються використовуючи поле Галуа $GF(2^8)$ по модулю x^4+1 на фіксоване $c(x)=3x^3+x^2+x+2$

Під час цього етапу кожен стовпчик множиться на матрицю, яка для 128 бітного ключа виглядає так:

$$\begin{pmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 2 \\ 3 & 1 & 1 & 2 \end{pmatrix}$$

XORRoundKey

Побайтовий XOR ключа з байтами таблиці повідомлення

В програмній реалізації алгоритм AES, виконує роль другого шифру для безпечної передачі повідомлення, цей алгоритм прийнятий як міжнародний

стандарт для шифрування даних. Цей алгоритм має дуже високу стійкість до зламу лобовою атакою, дуже близьку до ста відсотків, тому що на злам 128-бітного ключа, потрібно $1.02 * 10^{18}$ степені років, тобто 2^{128} операцій. І це тільки для 128-бітного ключа, для 192х і 256-ти бітних геометрично більше: $1.872 * 10^{37}$ і $3.31*10^{56}$ відповідно. Програма реалізація виконує заміну для кожного з 16 байтів, використовуючи S-box, виконується зсув вліво та змішуються колонки, для шифрування і дешифрування використовується ключ з файлу “keyfile”, тобто умова така, що ключ AES не передається. Так як, 128-бітного ключа досить, щоб шифр був дуже стійким, його у програмній реалізації і використали, цей шифр є дуже надійний і стійкий до зламу. І хоча, цей шифр є симетричним – він є досить надійним для використання на державному рівні.

2.3. Дослідження впровадження запропонованої СЕП на базі криптографічних перетворень.

Програмна розробка представлена у даній роботі, готова до роботи з першого запуску. Для зручності підключена українська мова, інтерфейс програми дружелюбний, на кожному етапі є підказка що потрібно ввести, на першому етапі потрібно обрати пункт меню (рис. 2.6), що саме бажаєте зробити, перше це основна функція програми де реалізоване шифрування і дешифрування AES, RSA та хешування.

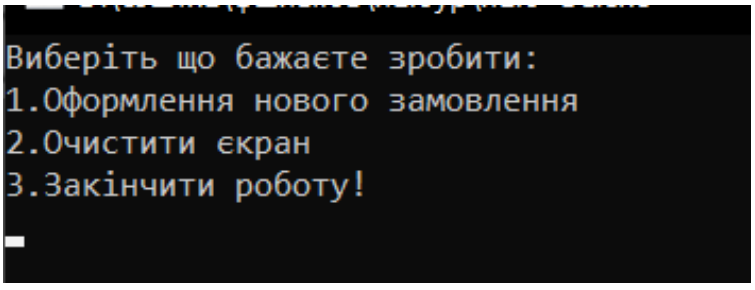

Рис. 2.6 Пункти меню

Перша дія – основна дія програми де буде відбуватися шифрування та дешифрування (рис. 2.7).

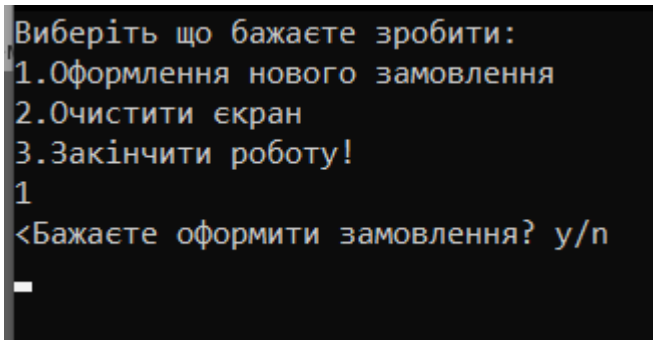

Рис. 2.7 Обираємо перший пункт меню

На даному етапі потрібно підтвердити вибір, чи бажаєте ви оформити замовлення.

В разі підтвердження, програма оформлення замовлення продовжить діалог з користувачем, в разі відмови (рис. 2.8), програмний додаток повернеться у початкове меню і запропонує обрати інший пункт.

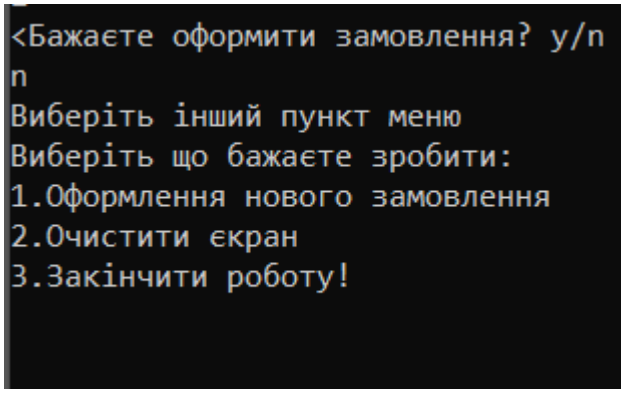

Рис. 2.8 Демонстрація відмови від оформлення замовлення

В разі підтвердження вибору, програма запросить введення картки, потім перевірить, щоб введені данні були правильними, тобто номер картки повинен бути 16 цифр(рис. 2.9).

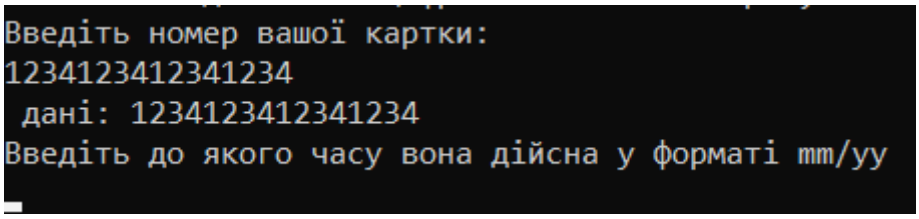

Рис. 2.9 Введення даних картки

В разі неправильно введеного номеру картки (рис. 2.10) , висвітиться діалог, де програма запросить знову ввести ваші дані, щоб перейти до наступного кроку (рис. 2.11).

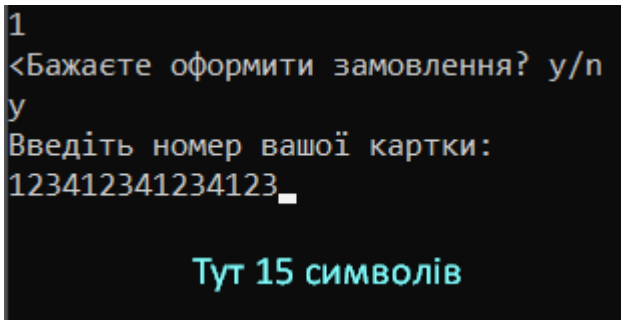

Рис. 2.10 Демонстрація неправильно заповненої картки

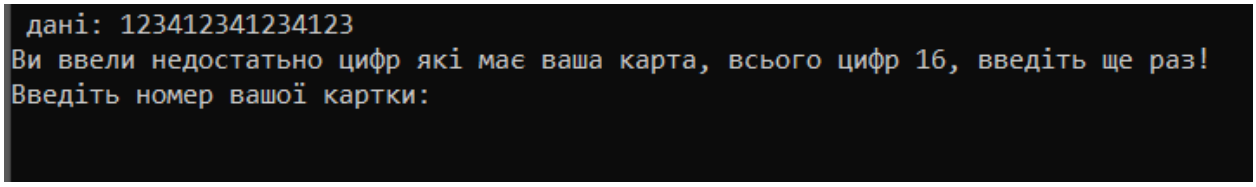

Рис. 2.11 Повернення до заповнення номеру картки

На наступному кроці потрібно буде ввести дату дійсності карти, у форматі місяц/рік (рис. 2.12).

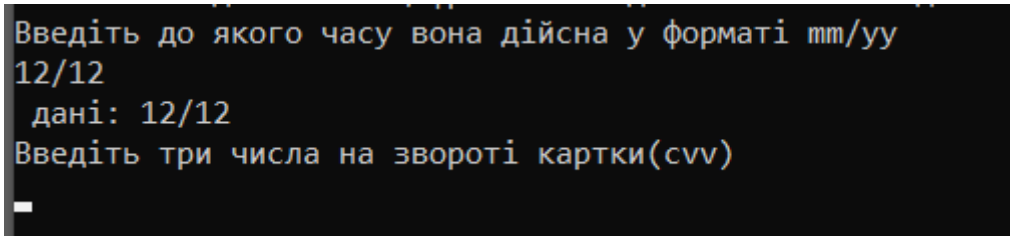

Рис. 2.12 Заповнення дати закінчення терміну дії картки

При неправильно введеній кількості символів, програма звернеться з діалогом і попросить ввести ще раз, а також підскаже знову у якому форматі потрібно вводити дані (рис. 2.13).

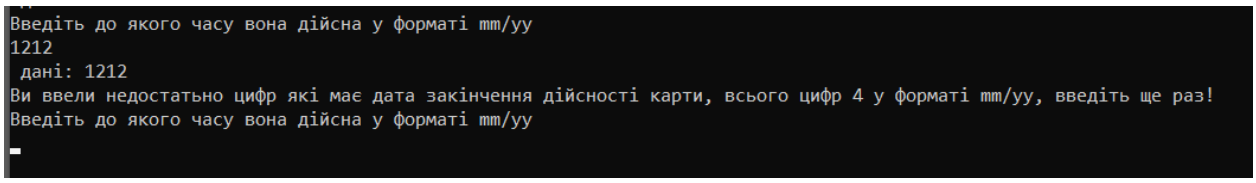

Рис. 2.13 Демонстрація неправильно заповненої дати

Наступний крок – це введення трьох секретних цифр, які вказані на звороті картки (CVV) (рис. 2.14)

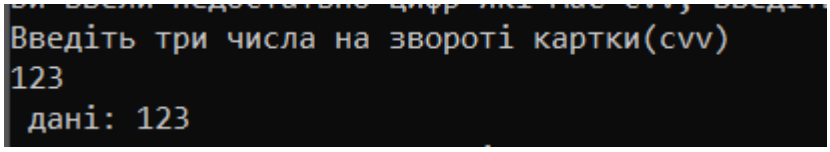

Рис. 2.14 Заповнення CVV

В разі неправильно заповнених даних, програма звернеться з проханням ввести дані ще раз. (рис. 2.15)

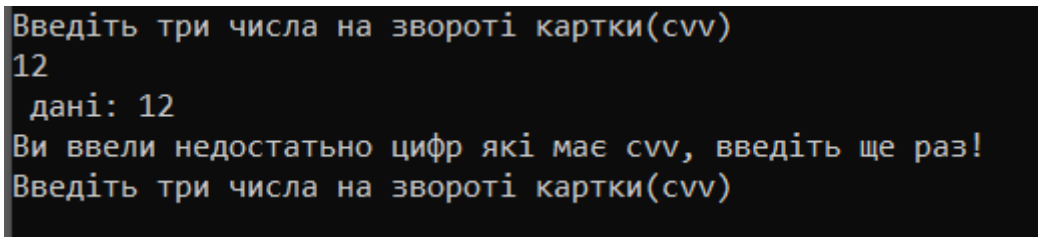

Рис. 2.15 демонстрація неправильно заповненої CVV

На наступному кроці програма виводить ваші введені з клавіатури дані, щоб той хто вводив мав можливість їх переглянути. (рис. 2.16)

Так той хто вводив дані може переглянути їх, та переконатися, що дані введені вірно.

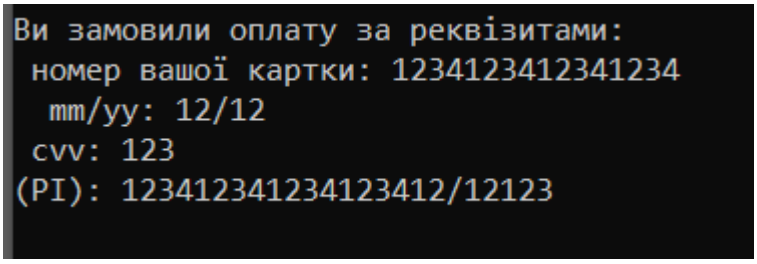

Рис. 2.16 виведення заповнених даних на екран

Наступним кроком йде хешування даних, які складаються з об’єднання номера картки, терміну її дії та секретних трьох цифр які записані на звороті картки(CVV). Результат хешування буде виведено на екран (рис. 2.17).

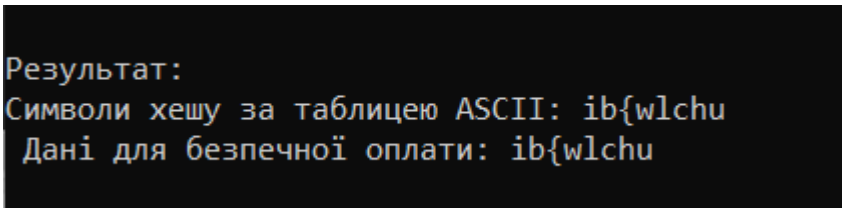

Рис. 2.17 Хешування даних

На наступному кроці йде шифрування RSA з використання відкритого ключа іншого користувача. Необхідно на даному етапі ввести відкритий ключ, який можна отримати у іншого користувача (рис. 2.18).

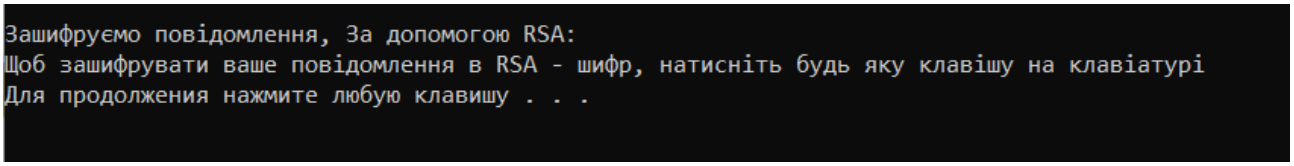

Рис. 2.18 Шифрування RSA користувачем

На наступному кроці відбувається шифрування тексту за допомогою алгоритму RSA (рис. 2.19)

Текст буде зашифровано за допомогою ключа алгоритму RSA

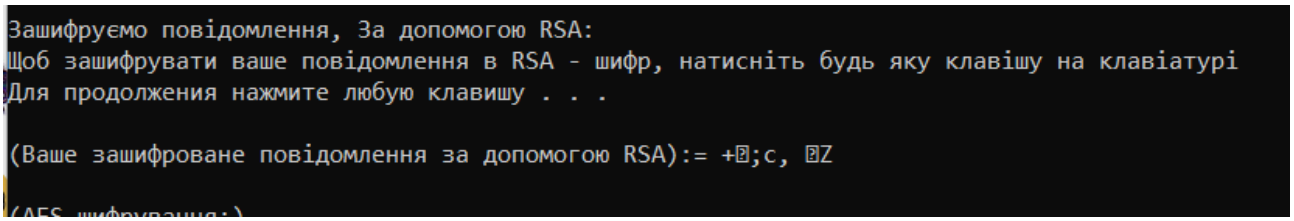

Рис. 2.19 Шифруємо повідомлення за допомогою RSA

На наступному кроці відбувається шифрування тексту алгоритмом AES, де ключ зчитується з файлу «keyfile» (рис. 2.20).

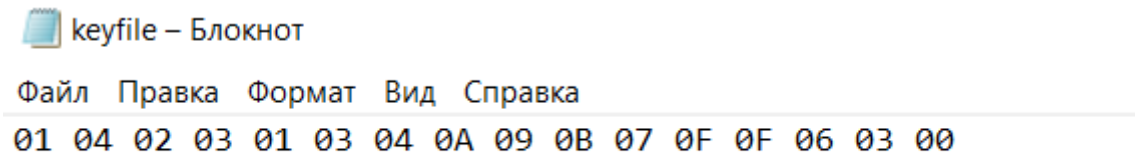

Рис. 2.20 демонстрація Keyfile – файлу

Зашифроване повідомлення демонструється відправнику на екрані, так повідомляє відправника, про те, що цей зашифрований файл відправляється отримувачу.

Потім програма пропонує перейти до отримувача, для цього потрібно натиснути будь яку клавішу на клавіатурі, наприклад можна натиснути «Enter» (рис. 2.21).

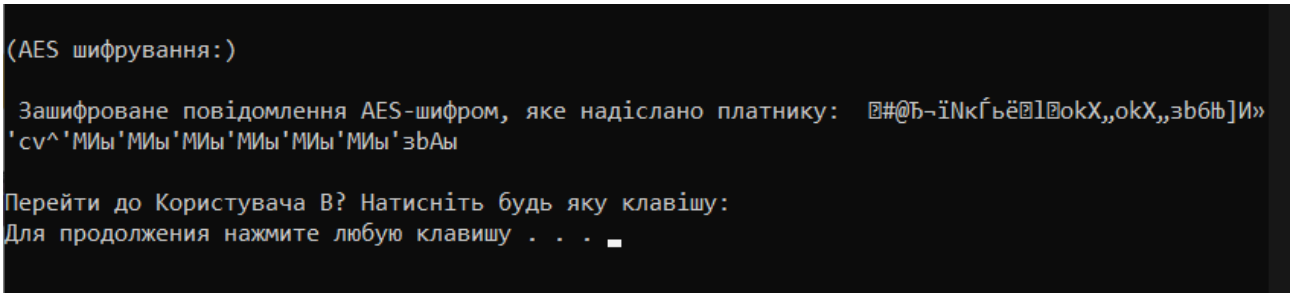

Рис 2.21 Зашифроване повідомлення AES-шифром, та пропонування програми перехід до діалогу з отримувачем платежу

Програма переходить до екрану отримувача, де одразу демонструє зашифроване за допомогою шифру AES повідомлення, та пропонує його дешифрувати (рис. 2.22).

Для того, щоб повідомлення дешифрувати, потрібно з того самого файлу, який передбачено протоколом SET мають і отримувач платежу і його відправник. Keyfile (рис. 2.23).

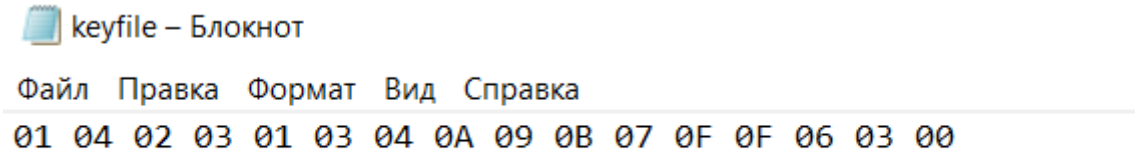

Рис. 2.22 демонстрація Keyfile – файлу

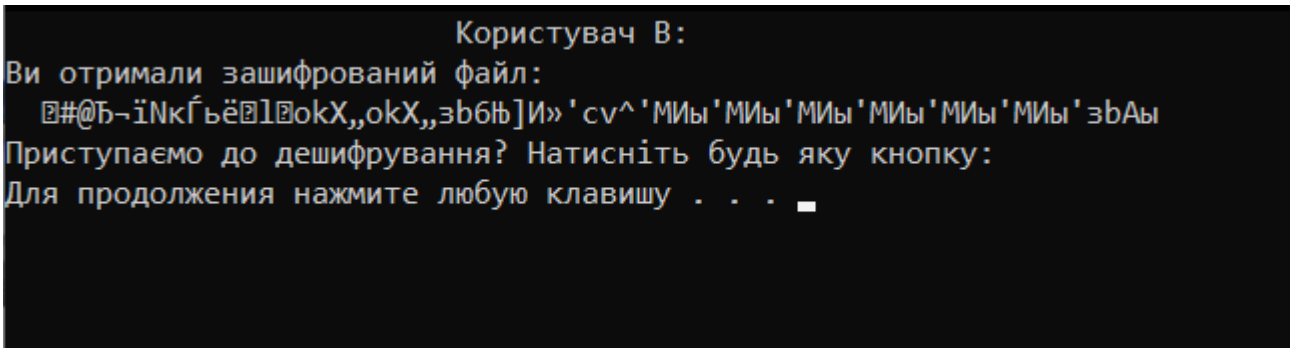

Рис. 2.23 Екран користувача B, та зашифроване AES – шифром повідомлення

Для того щоб дешифрувати повідомлення отримувач повинен натиснути будь яку клавішу на клавіатурі.

Програма дешифрує повідомлення;

Потім програма продемонструє дешифроване повідомлення (рис. 2.24)

Ключ для дешифрування береться з файлу, який за умовою використання SET протоколу не передається. А зберігається в базі даних банку, тож ключ ніяк не можна перехопити, тому саме це симетричне шифрування є одним, з найбезпечніших шифрувань на сьогоднішній день, для того, щоб передавати

потрібну інформацію, навіть у сфері банку а саме інформацію про грошові транзакції.

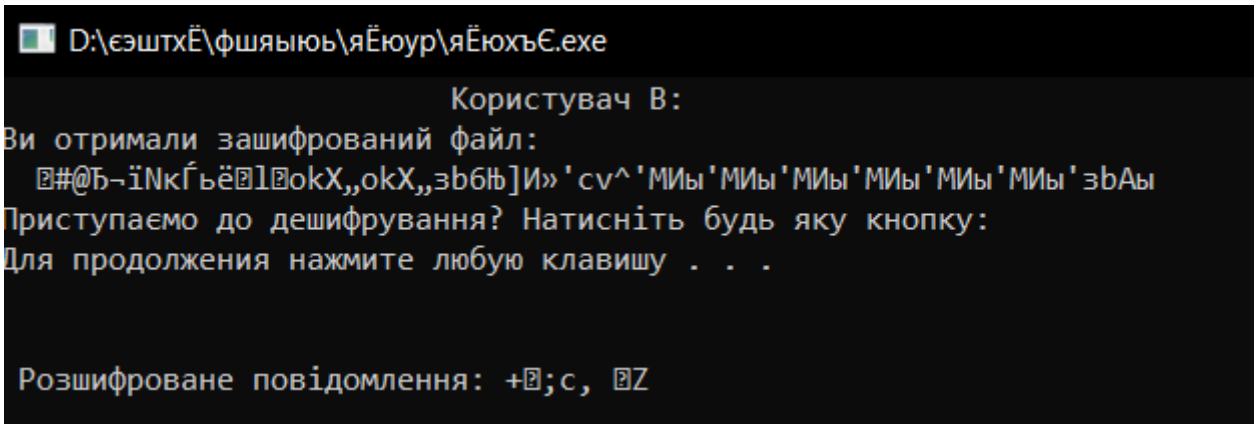

Рис. 2.24 Дешифроване повідомлення AES

Наступним кроком, програма переходить до дешифрування повідомлення за допомогою ключа RSA.

Програма просить ввести секретний ключ отримувача, який знаходиться за алгоритмами формування ключа RSA. Закритий ключ: {n,d}, який отримувач нікому не повинен повідомляти (рис. 2.25).

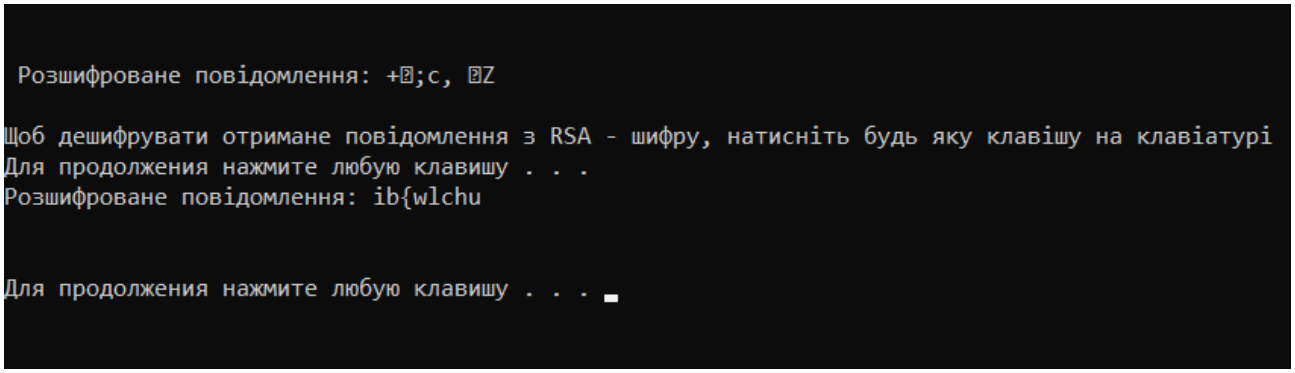

Рис. 2.25 Введення ключа отримувача, дешифрування RSA

Наступним етапом буде екран який не видно ні отримувачу ні відправнику, де порівнюється початковий хешований текст з отриманим, на основі збігу, вирішується дозволяти транзакцію чи ні, в разі збігу транзакція вважається безпечною і видає дозвіл на проведення цієї оплати (рис. 2.26).

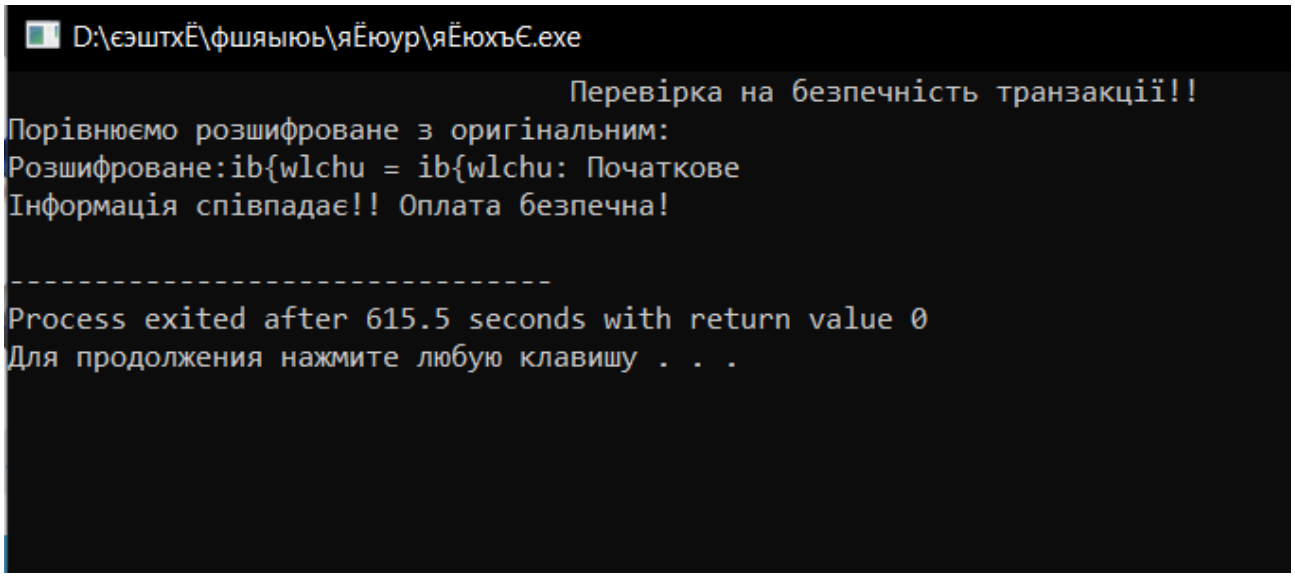

Рис. 2.26 Порівняння початкового тексту(хешу) з розшифрованим, підтвердження оплати

На даному рисунку продемонстровано не збіг отриманого результату і початкового, тож на якомусь етапі, були введені неправильні дані, або помилка при передачі даних (рис. 2.27). Такі дані бувають, якщо було порушено передачу даних, тобто так можна виявити, чи безпечний канал передачі транзакції, а також чи є безпечним видавання даних по цій транзакції.

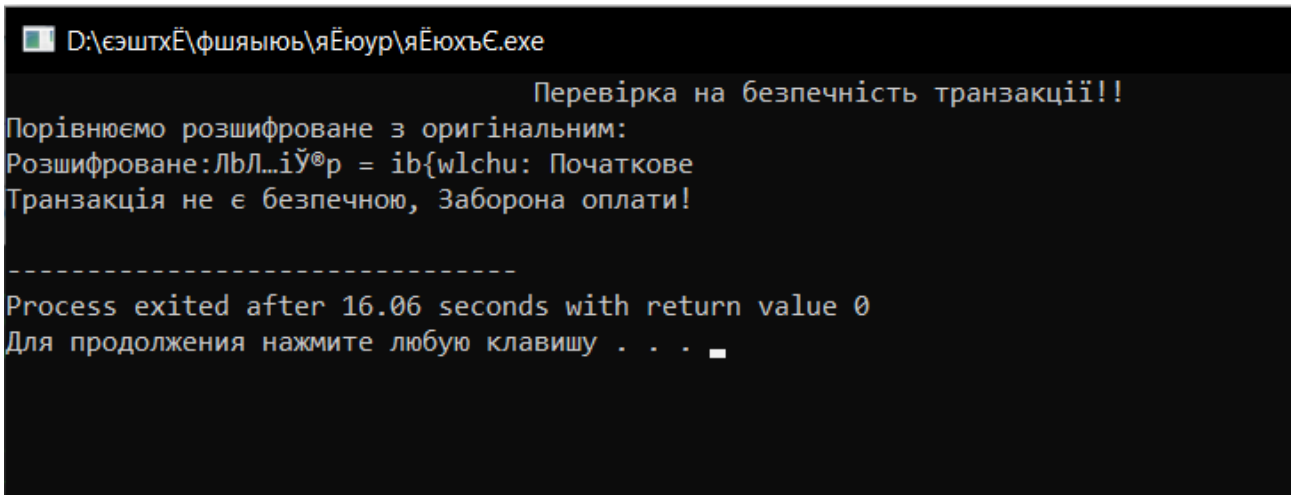

Рис. 2.27 Порівняння початкового тексту(хешу) з розшифрованим, Заборона транзакції

При неправильній відповіді на питання програма просить ще раз відповісти на питання (рис. 2.28).

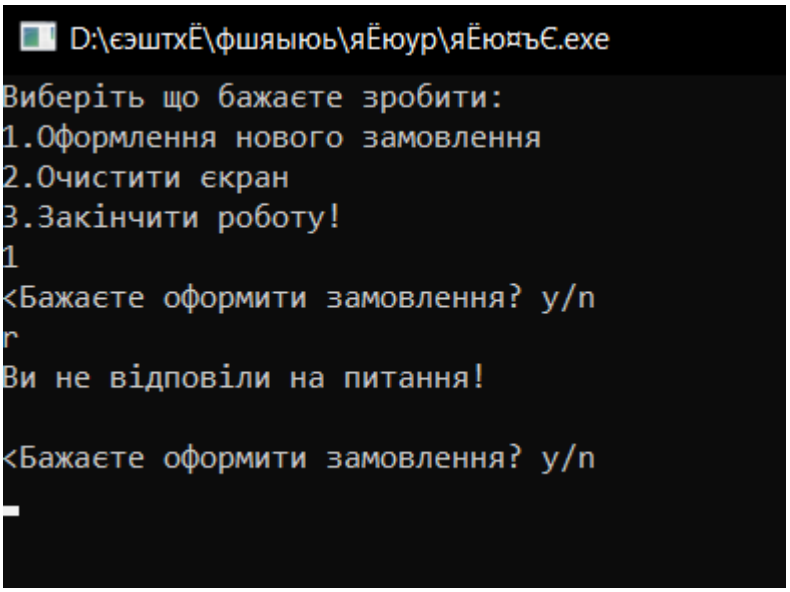

Рис. 2.28 Демонстрація неправильного вводу на питання про оформлення

На скриншоті продемонстрована функція очищення екрану. Спочатку додаємо інформацію на екран (рис. 2.29), потім очищуємо його (рис. 2.30).

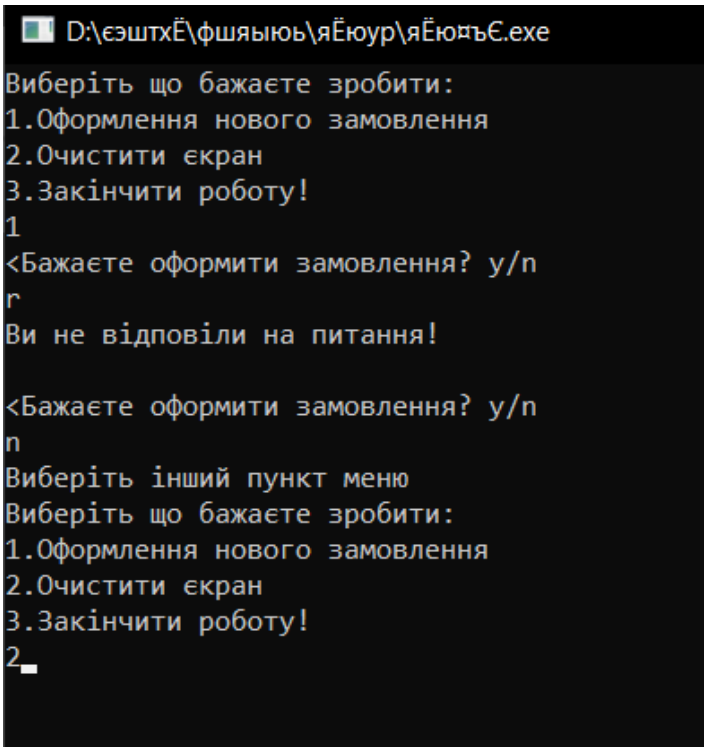

Рис. 2.29 Заповнюємо екран інформацією

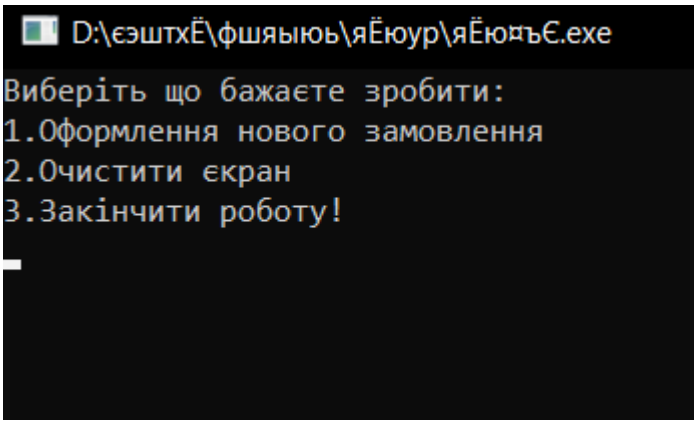

Рис. 2.30 Демонстрація очищення екрану

В даному пункті меню(3) можна завершити роботу з програмою після її початку, по бажанню користувача (рис. 2.31).

Наприклад, якщо на цей момент користування програмою користувачу не цікаво, меню дозволить йому здійснити вихід з вікна програми, щоб продовжити роботу.

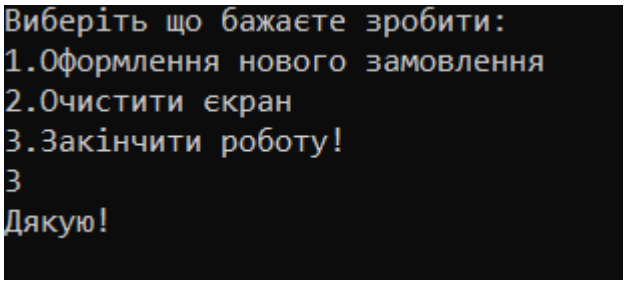

Рис. 2.31 Демонстрація закінчення роботи з програмою.

На даному зображенні показано, що при виборі пункту меню якого не існує, програма просить ще раз ввести пункт меню і запропонує вибрати його з вже існуючих, тобто посилання на меню при виборі неправильного пункту меню, для того, щоб користувач, мав змогу обрати правильно, якщо помилився при попередньому виборі (рис. 2.32).

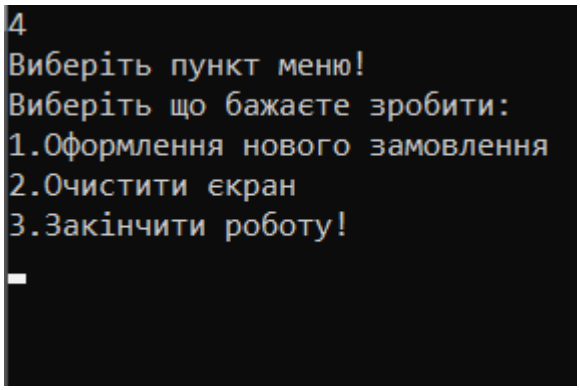

Рис. 2.32 Демонстрація неправильно обраного пункту меню

2.4. Висновки до другого розділу

В даному розділі була показана практична частина дипломної роботи. Було описано програмний додаток, його реалізацію, апаратну частину, його алгоритм роботи, Шифри які використовувалися, це: AES(128 бітний) і RSA, та хешування,

Була розроблена проста програмна реалізація SET протоколу, який дозволяє безпечно оплачувати в інтернеті, для програмного додатку використовувалася, одна з найпопулярніших на сьогоднішній день мова програмування C++.

Головна перевага даного протоколу – це передача даних в зашифрованому вигляді, що підвищує безпеку транзакції в інтернеті.

ВИСНОВКИ

Ідеальна платіжна система повинна бути безпечною, тобто не повинно бути так, що кошти з рахунку списуються без відома власника, реалізація використання системи не повинно бути складним. Також СЕП повинен забезпечувати високий рівень конфіденційності для клієнтів та система повинна бути вільною, тобто без обмежень для клієнтів

Цього не так просто досягти, для цього на сьогоднішній день використовують різні протоколи безпеки, щоб забезпечити конфіденційність і анонімність тих клієнтів, які цього бажають, але в противагу цього, забезпечити іншого клієнта, який виконує одну операцію з попередньо вказаним, безпекою, щоб у клієнта не було проблем саме через анонімність транзакції, саме протокол SET дозволяє проводити такі перевірки, і запобігати викраданням коштів з рахунків власників карток. Проте з кожним днем кібер – злочинці вигадують нові способи, які в майбутньому могли привести до зламу навіть найнадійніші протоколи безпеки.

Тож потрібно кожен день вдосконалювати ті системи які ми маємо, або писати нові протоколи безпечних транзакцій, які будуть більш надійними за ті які зараз існують, тому що по всім даним система електронних платежів буде існувати і надалі, тому потрібно забезпечити безпечне і комфортне користування даними можливостями. Використання різних проколів безпеки покращує систему транзакцій і робить її більш безпечною для використання звичайними користувачами. В ході аналізу і розробки програмного додатку, було встановлено, що найбільш безпечні алгоритми шифрування саме асиметричні, тому що наявність різних ключів шифрування і дешифрування, а також те що ключ шифрування можна передавати по безпечному не секретному каналу – дуже зручно, так як цим ключем не можна буде дешифрувати повідомлення.

В даній роботі був розроблений програмний додаток на основі SET протоколу, з використання AES симетричного шифру та RSA – асиметричного шифру. Аналіз допоміг зрозуміти, що передавати інформацію про заказ та платіж

у зашифрованому вигляді забезпечує безпеку як отримувача платежу так і відправника. Безпека полягає в тому що секретний ключ нікому не повідомляється, а зберігається у власника, тож злам цього шифру буде проходити досить довго, настільки, що цей платіж все перестане бути актуальним.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Арбузов С. Г., Колобов Ю. В., Міщенко В. І., Науменкова С. В. Міжбанківський переказ коштів у СЕП // Банківська енциклопедія. — Київ : Центр наукових досліджень Національного банку України : Знання, 2011. — 504 с. — (Інституційні засади розвитку банківської системи України).

2. Стаття новин УКРІНФОРМ: Система електронних платежів НБУ запрацювала в режимі 23/7[Електронний ресурс] - Режим доступу: https://www.ukrinform.ua/rubric-economy/3075433-sistema-elektronnih-plateziv-nbu-zapracuvala-v-rezimi-237.html

3. Смирнов С.Н. Электронный бизнес. - М.: ДМК Пресс Компания АйТи. 2003. - 240 с.

4. Стаття «Структура платіжних систем в Інтернеті» [Електронний ресурс] - Режим доступу : https://studme.org/296124/informatika/struktura_platezhnyh_sistem_internete

5. Правила Національної системи масових електронних платежів, затверджені постановою Правління Національного банку України від 10.12.2004 № 620.

6. ГОСТ 34.310-95. Межгосударственный стандарт. Информационная технология. Криптографическая защита информации. Процедура выработки и проверки электронной цифровой подписи на базе асимметричного криптографического алгоритма. Киев. Госстандарт Украины. 1998. Захист інформації

7. ГОСТ 34.311-95. Межгосударственный стандарт. Информационная технология. Криптографическая защита информации. Функция хеширования. Киев. Госстандарт Украины. 1998

8. ГОСТ Р 34.10-94. Информационная технология. Криптографическая защита информации. Процедуры выработки и проверки электронной цифровой подписи на базе асимметричного криптографического алгоритма.

9. ГОСТ Р34.11-94. Информационная технология. Криптографическая защита информации. Функция хэширования.

10. Криптография в банковском деле. / М.И. Анохин, Н.П. Варновский, В.М. Сидельников, В.В. Ященко. – М.: МИФИ. 1997. – 274 с.

11. Аволио Ф.М. Защита информации на предприятии / Ф.М. Аволио, Г. Шипли // Сети и системы связи. – 2000. − № 8. – С. 91−99.

12. Диффи У. Защищенность и имитостойкость / У. Диффи, М. Хеллман // Введение в криптографию. – 1979. – № 3. − С. 79–109.

13. Євсеєв С.П. Механизмы обеспечения аутентичности банковских данных во внутриплатежных системах комерческого банка / С.П. Євсеєв, В.Е. Чевардин, С.А. Радковский // Збірник наукових статей ХНЕУ. – Х.: ХНЕУ, 2008. – Вип. 6. – С. 40 – 44.

14. Задірака В.К. Методи захисту банківської інформації. / В.К. Задірака, О.С. Олесюк, Н.О. Недашковський. – К.: Вища школа, 1999. – 264 с.

15. Корченко А.О. Банківська безпека / А.О. Корченко, Л.М. Скачек, В.О. Хорошко; за загальним ред. д.т.н. проф. В.О. Хорошка. – К: ПВП “Задруга”, 2014. – 185 с.

16. Національний Стандарт України ДСТУ 7624- 2014. Інформаційні технології. Криптографічний захист інформації. Алгоритм симетричного блокового перетворення. К:. 2014. – 235 с.

17. Національний Стандарт України ДСТУ 7564- 2014. Інформаційні технології. Криптографічний захист інформації. Функція хешування. К:. 2014. – 41 с.

18. Стандарт України СОУ Н НБУ 65.1 СУІБ 1.0:2010. Методи захисту в банківській діяльності система управління інформаційною безпекою. Вимоги. (ISO/IEC 27001:2005, MOD).К: НБУ., 2010. – 67 с.

19. Стандарт України СОУ Н НБУ 65.1 СУІБ 1.0:2010. Звід правил для управління інформаційною безпекою (ISO/IEC 27002:2005, MOD)).К: НБУ., 2010. – 209 с.

20. Программное средство криптографической защиты информации "Грифон-Б" [Електронний ресурс]. – Режим доступу: http://www.banksoft.com.ua/index.php?id=28.

21. Программное средство «Библиотека функций криптографической защиты информации "Грифон-Л" [Електронний ресурс]. – Режим доступу: http://www.banksoft.com.ua/index.php?id=27.

22. Логинов А.А. Общие принципы функционирования электронных платежных систем и осуществление мер безопасности при защите от злоупотребления и мошенничества / А.А. Логинов, Н.С. Елхимов // Конфидент. – 1995. – № 4. – С.48 – 54

23. Worldwide Infrastructure Security Report. 2014. Arbor Networks, Inc Електронний ресурс]. – Режим доступу: https://www.google.com.ua/url?sa=t&rct=j&q=&esrc=s&source=web&cd=1&cad=rja&uact=8&ved=0CB8QFjAA&url=http%3A%2F%2Fpages.arbornetworks.com%2Frs%2Farbor%2Fimages%2FWISR2014_EN2014.pdf&ei=DyR2VfznJOPgyQOghoN4&usg=AFQjCNGP0_ZTliItqCtofJcXfZT9QHRiQ&sig2=4hgA_Iye1idQyQgsTIZXg&bvm=bv.95039771,d.bGQ.

24. https://www.geeksforgeeks.org/secure-electronic-transaction-set-protocol/

25. https://medium.com/@jinkyulim96/algorithms-explained-rsa-encryption-9a37083aaa62

26. Вікіпедія - [Електронний ресурс] . Режим доступу: https://uk.wikipedia.org/wiki/Advanced_Encryption_Standard

27. Бібліотека процедур криптографічного захисту інформації «Тайфун -PKI PKCS#11 – [Електронний ресурс]. Режим доступу - http://www.ict.com.ua/?lng=1&sec=7

28. Ситник І.П., Микитенко Я.В. Сучасний стан та перспективи розвитку системи електронних платежів Національного банку України. Вчені записки ТНУ ім. В.І. Вернадського. Серія «Економіка і управління». Вип.30(69). № 2. 2019. С. 223–227.

Додаток А

```
#include <iostream>
#include <math.h>
#include <time.h>
#include <iomanip>
#include <cstring>
#include <fstream>
#include <sstream>
#include "str.h"
#include <string.h>
#include <cstdio>
#include <conio.h>
#include <cstdlib>
#include <windows.h>
#include <string>
using namespace std;
long int encrypt( long int i, long int e, long int n )
{ long int current, result;
   current = i - 97;
   result = 1;
   for ( long int j = 0; j < e; j++ )
   { result = result * current;
      result = result % n;}
   return result;}
long int decrypt(long int i, long int d, long int n)
{ long int current, result;
   current = i;
   result = 1;
   for ( long int j = 0; j < d; j++ )
```

Продовження додатку А

```
  {    result = result * current;
       result = result % n;    }
   return result + 97;}
string check(string text, int len, int hash, int qua){
        if(len % hash != 0){
                text += "0";
                qua++;
                cout << "\nКількість бітів тексту непарна. Додаємо літеру-заповнювач:\nТекст після перетворення: " << text << endl;    }
        return text; }
string hashi(string tex) {
        int bits=16;
        int hash = bits / 8;
        char H[hash] = {0};
        int len = tex.length();
        int qua = len / hash;
        check(tex, len, hash, qua);
        string H1;
        int count = 1;
        for(int i = 0; i < hash; i++){
                for(int j = 0; j < qua; j++){
                        H[i] = H[i] << 1; //зсув вліво
                        H[i] = H[i] ^ tex[i + j * hash];
                        if(H[i]<0)H[i]=H[i]*-1;
                        count++; }   }
        cout <<"\nРезультат:\nСимволи хешу за таблицею ASCII: ";
        for(int i = 0; i < hash; i++) cout << H[i];
        H1=string(H,8);
```

Продовження додатку А

```
        return H1;}
void AddRoundKey(unsigned char* state, unsigned char* roundKey) {
        for (int i = 0; i < 16; i++) {
                state[i] ^= roundKey[i];   }}
void SubRoundKey(unsigned char* state, unsigned char* roundKey) {
        for (int i = 0; i < 16; i++) {
                state[i] ^= roundKey[i];}}
void SubBytes1(unsigned char* state) {
        for (int i = 0; i < 16; i++) {
                state[i] = s[state[i]];         }}
void SubBytes(unsigned char* state) {
        for (int i = 0; i < 16; i++) {
                state[i] = inv_s[state[i]];}}
void ShiftRows1(unsigned char* state) {
        unsigned char tmp[16];

        /* колонка 1 */
        tmp[0] = state[0];
        tmp[1] = state[5];
        tmp[2] = state[10];
        tmp[3] = state[15];

        /* колонка 2 */
        tmp[4] = state[4];
        tmp[5] = state[9];
        tmp[6] = state[14];
        tmp[7] = state[3];
```

Продовження додатку А

```
        /* колонка 3 */
        tmp[8] = state[8];
        tmp[9] = state[13];
        tmp[10] = state[2];
        tmp[11] = state[7];

        /* колонка 4 */
        tmp[12] = state[12];
        tmp[13] = state[1];
        tmp[14] = state[6];
        tmp[15] = state[11];
        for (int i = 0; i < 16; i++) {
                state[i] = tmp[i];}}
void InverseMixColumns(unsigned char* state) {
        unsigned char tmp[16];
        tmp[0] = (unsigned char)mul14[state[0]] ^ mul11[state[1]] ^ mul13[state[2]] ^
mul9[state[3]];
        tmp[1] = (unsigned char)mul9[state[0]] ^ mul14[state[1]] ^ mul11[state[2]] ^
mul13[state[3]];
        tmp[2] = (unsigned char)mul13[state[0]] ^ mul9[state[1]] ^ mul14[state[2]] ^
mul11[state[3]];
        tmp[3] = (unsigned char)mul11[state[0]] ^ mul13[state[1]] ^ mul9[state[2]] ^
mul14[state[3]];

        tmp[4] = (unsigned char)mul14[state[4]] ^ mul11[state[5]] ^ mul13[state[6]] ^
mul9[state[7]];
        tmp[5] = (unsigned char)mul9[state[4]] ^ mul14[state[5]] ^ mul11[state[6]] ^
mul13[state[7]];
        tmp[6] = (unsigned char)mul13[state[4]] ^ mul9[state[5]] ^ mul14[state[6]] ^
mul11[state[7]];
```

Продовження додатку А

```
        tmp[7] = (unsigned char)mul11[state[4]] ^ mul13[state[5]] ^ mul9[state[6]] ^
mul14[state[7]];


        tmp[8] = (unsigned char)mul14[state[8]] ^ mul11[state[9]] ^ mul13[state[10]]
^ mul9[state[11]];
        tmp[9] = (unsigned char)mul9[state[8]] ^ mul14[state[9]] ^ mul11[state[10]] ^
mul13[state[11]];
        tmp[10] = (unsigned char)mul13[state[8]] ^ mul9[state[9]] ^ mul14[state[10]]
^ mul11[state[11]];
        tmp[11] = (unsigned char)mul11[state[8]] ^ mul13[state[9]] ^ mul9[state[10]]
^ mul14[state[11]];


        tmp[12] = (unsigned char)mul14[state[12]] ^ mul11[state[13]] ^
mul13[state[14]] ^ mul9[state[15]];
        tmp[13] = (unsigned char)mul9[state[12]] ^ mul14[state[13]] ^
mul11[state[14]] ^ mul13[state[15]];
        tmp[14] = (unsigned char)mul13[state[12]] ^ mul9[state[13]] ^
mul14[state[14]] ^ mul11[state[15]];
        tmp[15] = (unsigned char)mul11[state[12]] ^ mul13[state[13]] ^
mul9[state[14]] ^ mul14[state[15]];


        for (int i = 0; i < 16; i++) {
                state[i] = tmp[i];}}
void ShiftRows(unsigned char* state) {
        unsigned char tmp[16];
        /* колонка 1 */
        tmp[0] = state[0];
        tmp[1] = state[13];
        tmp[2] = state[10];
        tmp[3] = state[7];
```

Продовження додатку А

```
        /* колонка2 */
        tmp[4] = state[4];
        tmp[5] = state[1];
        tmp[6] = state[14];
        tmp[7] = state[11];
        /* колонка 3 */
        tmp[8] = state[8];
        tmp[9] = state[5];
        tmp[10] = state[2];
        tmp[11] = state[15];
        /* колонка 4 */
        tmp[12] = state[12];
        tmp[13] = state[9];
        tmp[14] = state[6];
        tmp[15] = state[3];
        for (int i = 0; i < 16; i++) {
                state[i] = tmp[i];    }}
void InitialRound(unsigned char* state, unsigned char* key) {
        SubRoundKey(state, key);
        ShiftRows(state);
        SubBytes(state);}
void MixColumns(unsigned char* state) {
        unsigned char tmp[16];

        tmp[0] = (unsigned char)mul2[state[0]] ^ mul3[state[1]] ^ state[2] ^ state[3];
        tmp[1] = (unsigned char)state[0] ^ mul2[state[1]] ^ mul3[state[2]] ^ state[3];
        tmp[2] = (unsigned char)state[0] ^ state[1] ^ mul2[state[2]] ^ mul3[state[3]];
```

Продовження додатку А

```
        tmp[3] = (unsigned char)mul3[state[0]] ^ state[1] ^ state[2] ^ mul2[state[3]];


        tmp[4] = (unsigned char)mul2[state[4]] ^ mul3[state[5]] ^ state[6] ^ state[7];
        tmp[5] = (unsigned char)state[4] ^ mul2[state[5]] ^ mul3[state[6]] ^ state[7];
        tmp[6] = (unsigned char)state[4] ^ state[5] ^ mul2[state[6]] ^ mul3[state[7]];
        tmp[7] = (unsigned char)mul3[state[4]] ^ state[5] ^ state[6] ^ mul2[state[7]];


        tmp[8] = (unsigned char)mul2[state[8]] ^ mul3[state[9]] ^ state[10] ^ state[11];
        tmp[9] = (unsigned char)state[8] ^ mul2[state[9]] ^ mul3[state[10]] ^ state[11];
        tmp[10] = (unsigned char)state[8] ^ state[9] ^ mul2[state[10]] ^
mul3[state[11]];
        tmp[11] = (unsigned char)mul3[state[8]] ^ state[9] ^ state[10] ^
mul2[state[11]];


        tmp[12] = (unsigned char)mul2[state[12]] ^ mul3[state[13]] ^ state[14] ^
state[15];
        tmp[13] = (unsigned char)state[12] ^ mul2[state[13]] ^ mul3[state[14]] ^
state[15];
        tmp[14] = (unsigned char)state[12] ^ state[13] ^ mul2[state[14]] ^
mul3[state[15]];
        tmp[15] = (unsigned char)mul3[state[12]] ^ state[13] ^ state[14] ^
mul2[state[15]];
        for (int i = 0; i < 16; i++) {
                state[i] = tmp[i];}}
void Round1(unsigned char* state, unsigned char* key) {
        SubBytes1(state);
        ShiftRows1(state);
        MixColumns(state);
        AddRoundKey(state, key); }
```

Продовження додатку А

```
void Round(unsigned char* state, unsigned char* key) {
        SubRoundKey(state, key);
        InverseMixColumns(state);
        ShiftRows(state);
        SubBytes(state);}
void FinalRound(unsigned char* state, unsigned char* key) {
        SubBytes1(state);
        ShiftRows1(state);
        AddRoundKey(state, key);
}
void AESEncrypt(unsigned char* message, unsigned char* expandedKey, unsigned
char* encryptedMessage) {
        unsigned char state[16]; // Зберігає перші 16 байтів вихідного повідомлення

        for (int i = 0; i < 16; i++) {
                state[i] = message[i];}
        int numberOfRounds = 9;
        AddRoundKey(state, expandedKey); // початковий раунд
        for (int i = 0; i < numberOfRounds; i++) {
                Round1(state, expandedKey + (16 * (i + 1)));}
        FinalRound(state, expandedKey + 160);
        for (int i = 0; i < 16; i++) {
                encryptedMessage[i] = state[i];}}
void AESDecrypt(unsigned char* encryptedMessage, unsigned char* expandedKey,
unsigned char* decryptedMessage)
{
        unsigned char state[16];
        for (int i = 0; i < 16; i++) {
```

Продовження додатку А

```
        state[i] = encryptedMessage[i];}
        InitialRound(state, expandedKey + 160);

        int numberOfRounds = 9;
        for (int i = 8; i >= 0; i--) {
                Round(state, expandedKey + (16 * (i + 1)));}
        SubRoundKey(state, expandedKey);
        for (int i = 0; i < 16; i++) {
                decryptedMessage[i] = state[i];}}
 int clear_screen ()  {
       system("cls");    }

int main() {
        SetConsoleOutputCP(1251);//українська мова
        int choice;
        a1:
        cout<<"Виберіть що бажаєте зробити: "<<endl<<"1.Оформлення нового замовлення"<<endl<<"2.Очистити єкран"<<endl<<"3.Закінчити роботу!"<<endl;
        cin >> choice;
    switch(choice){
        case 1:
      a2:
        cout<<"<Бажаєте оформити замовлення? y/n"<<endl;
        char a;
        cin>>a;
        if(a=='n'){
                cout<<"Виберіть інший пункт меню" <<endl;
                        goto a1;}
```

Продовження додатку А

```
if(a=='y'){
    a4:
cout<<"Введіть номер вашої картки: "<<endl;
string q;
int maxchar =16 ;
cin >> setw(maxchar) >> q;
cout << " дані: " << q<<endl;
cin.ignore();
if(q.length()<maxchar){
    cout<<"Ви ввели недостатньо цифр які має ваша карта, всього цифр 16, введіть ще раз! "<<endl;
goto a4;}
a5:
cout<<"Введіть до якого часу вона дійсна у форматі mm/yy "<<endl;
string q1;

maxchar =5 ;
cin >> setw(maxchar) >> q1;
cout << " дані: " << q1<<endl;
cin.ignore();
    if(q1.length()<maxchar){
    cout<<"Ви ввели недостатньо цифр які має дата закінчення дійсності карти, всього цифр 4 у форматі mm/yy, введіть ще раз! "<<endl;
goto a5;}
a6:
cout<<"Введіть три числа на звороті картки(cvv) "<<endl;
string q2;
```

Продовження додатку А

```
                maxchar =3 ;
                cin >> setw(maxchar) >> q2;
                cout << " дані: " << q2<<endl;
                cin.ignore();


                if(q2.length()<maxchar){
                cout<<"Ви ввели недостатньо цифр які має cvv, введіть ще раз! "<<endl;
                goto a6;}
                cout<<"Ви замовили оплату за реквізитами:"<<endl;
                cout<<" номер вашої картки: "<<q<<"\n  mm/yy: "<<q1<<"\n cvv: "<<q2<<endl;
                string q4=q+q1+q2;
                cout<<"(PI): "<<q4<<endl;
                string q5;
                q5=hashi(q4);
                cout<<"\n Дані для безпечної оплати: "<<q5<<endl;
                string q7;
                cout << "\nЗашифруємо повідомлення, За допомогою RSA: " <<  endl;


        long int encryptedText[100];
                memset(encryptedText, 0, sizeof(encryptedText));
                long int decryptedText[100];
                memset(decryptedText, 0, sizeof(decryptedText));
                cout<<"Щоб зашифрувати ваше повідомлення в RSA - шифр, натисніть будь яку клавішу на клавіатурі "<<endl;
                system("pause");
                long int n, e, d;
```

Продовження додатку А

```
            n=119;
            e=5;
            d=77;
            cout<<"e = "<<endl;
            cin>>e;
    for (long int i = 0; i < q5.length(); i++)
    {
        encryptedText[i] = encrypt((int)q5[i], e, n);
    }
    char q10[100];
    for ( long int i = 0; i < q5.length(); i++ ){
        q10[i]=(char)encryptedText[i];    }
   string q6=string(q10,8);
    cout<<"\n(Ваше зашифроване повідомлення за допомогою RSA):=
"<<q6<<endl;
        cout<<"\n(AES шифрування:)"<<endl;
          string message = q6;
          int originalLen = sizeof(message);
          int paddedMessageLen = originalLen;
          if ((paddedMessageLen % 16) != 0) {
                paddedMessageLen = (paddedMessageLen / 16 + 1) * 16;}
          unsigned char* paddedMessage = new unsigned char[paddedMessageLen];
          for (int i = 0; i < paddedMessageLen; i++) {
                if (i >= originalLen) {
                      paddedMessage[i] = 0;}
                else {
                      paddedMessage[i] = message[i];}}
          unsigned char* encryptedMessage = new unsigned char[paddedMessageLen];
```

Продовження додатку А

```
        string str;
        ifstream infile;
        infile.open("keyfile", ios::in | ios::binary);
        if (infile.is_open()){
                getline(infile, str); // Перший рядок файлу повинна бути ключем
                infile.close();}
        else cout << "Unable to open file";
        istringstream hex_chars_stream(str);
        unsigned char key[16];
        int i = 0;
        unsigned int c;
        while (hex_chars_stream >> hex >> c){
                key[i] = c;
                i++;}
        unsigned char expandedKey[176];
        KeyExpansion(key, expandedKey);
        for (int i = 0; i < paddedMessageLen; i += 16) {
                AESEncrypt(paddedMessage + i, expandedKey, encryptedMessage + i);
}
        for (int i = 0; i < paddedMessageLen; i++) {
                cout << " ";
        }
cout<<"\n Зашифроване повідомлення AES-шифром, яке надіслано платнику:
"<<encryptedMessage<<endl;
        cout << endl;
        cout<<"Перейти до Отримувача? Натисніть будь яку клавішу:"<<endl;
        system("pause");
        clear_screen ();
```

Продовження додатку А

```
   cout<<"\t \t \t Отримувач: "<<endl;
    cout<<"Ви отримали зашифрований файл: "<<endl;
    cout<<"  "<<encryptedMessage<<endl;
        cout<<"Приступаємо до дешифрування? Натисніть будь яку кнопку:
"<<endl;
    system("PAUSE");
    string keystr1;
        ifstream keyfile;
        keyfile.open("keyfile", ios::in | ios::binary);
        if (keyfile.is_open())
        {
                getline(keyfile, keystr1); // Перший рядок файлу повинна бути
ключем
                keyfile.close();}
        else cout << "Unable to open file";
        key[16]=0;
        i = 0;
        c=0;
        while (hex_chars_stream >> hex >> c)
        {key[i] = c;
                i++;}
 expandedKey[176]=0;
        KeyExpansion(key, expandedKey);
        int messageLen = strlen((const char*)encryptedMessage)
        unsigned char* decryptedMessage = new unsigned char[messageLen];
        for (int i = 0; i < messageLen; i += 16) {
                AESDecrypt(encryptedMessage + i, expandedKey, decryptedMessage +
i);}
```

Продовження додатку А

```
for (int i = 0; i < messageLen; i++) {
                cout << " ";
        }
        cout << endl;
        char q21[100];
        for (int i = 0; i < messageLen; i++) {
                q21[i]=(char)decryptedMessage[i];
        }
        string q22=string(q21,8);
        cout<<"\n Розшифроване повідомлення: "<< q22<<endl;
        cout << endl;
        cout<<"Щоб дешифрувати отримане повідомлення з RSA - шифру, натисніть будь яку клавішу на клавіатурі "<<endl;
                system("pause");
   for (long int i = 0; i < q22.length(); i++)
   {decryptedText[i] = decrypt(q22[i], d, n); }
   char q23[100];
   for (long int i = 0; i < q22.length(); i++)    {
      q23[i]=(char)decryptedText[i];  }
   string q24=string(q23,8);
        cout<<"Розшифроване повідомлення: "<< q24<<endl;
    cout <<  endl <<  endl;
system("pause");
clear_screen();
cout<<"\t \t \t Перевірка на безпечність транзакції!!"<<endl;
cout<<"Порівнюємо розшифроване з оригінальним:"<<endl;
cout<<"Розшифроване:"<<q24<<" = "<<q5<<": Початкове"<<endl;
if(q24==q5){
```

Продовження додатку А

```
        cout<<"Інформація співпадає!! Оплата безпечна!"<<endl;}
else cout<<"Транзакція не є безпечною, Заборона оплати!"<<endl;
        // Звільняємо пам'ять
        delete[] paddedMessage;
        delete[] encryptedMessage;
                        }
                        if(a!='y' and a!='n') {
                        cout<<"Ви не відповіли на питання! \n "<<endl;
                        goto a2;}
                        break;
      case 2:  clear_screen (); goto a1;  break;
      case 3: cout<<"Дякую!"<<endl; break;
      default:
        cout << "Виберіть пункт меню!" << endl;
        goto a1;
        break;  }
      return 0;}
```