МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ „ЛЬВІВСЬКА ПОЛІТЕХНІКА”

МЕТОДИЧНІ ВКАЗІВКИ
ДЛЯ ВИКОНАННЯ ЛАБОРАТОРНИХ РОБІТ З ДИСЦИПЛІНИ:

ТЕХНОЛОГІЇ ЗАХИСТУ ІНФОРМАЦІЇ

Львів – 2022

Пропоновані методичні вказівки призначені для виконання лабораторних робіт з дисципліни Технології захисту інформації. Вказані завдання повинні сформувати в студентів навички з основних принципів захисту інформації, які знадобляться їм для вирішення поставлених завдань.

Укладач: Басюк Т. М., к.т.н., доцент кафедри ІСМ

ВИМОГИ ДО ОФОРМЛЕННЯ ЗВІТІВ ПРО ВИКОНАННЯ ЛАБОРАТОРНИХ РОБІТ

Звіти про виконання лабораторних робіт оформляються на зшитих аркушах формату А4. Після захисту лабораторних робіт звіти здаються для зберігання на кафедру.

Кожен звіт повинен містити такі розділи:

- Номер та ***назва роботи;***
- ***Мета виконання лабораторної роботи;***
- ***Індивідуальне завдання*** з детальним формулюванням розв'язуваної задачі, використовувані ***(власні)*** теоретичні відомості;
- ***Відповіді на контрольні запитання***
- Текст програми реалізації.
- Результати кроків виконання алгоритму.
- Результати роботи програми.
- ***Висновки.*** Вказується призначення роботи, можливі варіанти вдосконалення та які знання отримано в ході виконання роботи.

Звіт повинен бути написаний українською мовою, акуратно та грамотно, з дотриманням правил оформлення ділової документації. Назви розділів звіту візуально виділити розміром, підкресленням.

ВСТУП

Лабораторний практикум містить лабораторні роботи з курсу "Технології захисту інформації". В теоретичних відомостях до виконання лабораторних робіт коротко подано опис методів та способів на основі яких ґрунтується виконання роботи. Детальність опису порядку виконання робіт зменшується, зважаючи на досвід, отриманий студентами при виконанні попередніх завдань.

При виконанні лабораторних робіт необхідно реалізувати програму на мовах об'єктно-орієнтованого програмування відповідно до заданого варіанту, яка забезпечує реалізацію відповідного способу обробки, аналізу чи криптографічного перетворення інформації.

До лабораторних занять студент повинен попередньо підготуватися, використовуючи рекомендовану літературу.

В результаті проведення робіт студент повинен закріпити отримані теоретичні знання, а також відповідним чином оформити звіт з матеріалами, отриманими при дослідженні складеної програми, який повинен містити:

- титульний лист;
- мета виконання лабораторної роботи;
- індивідуальне завдання з детальним формулюванням розв'язуваної задачі, використовувані (власні) теоретичні відомості;;
- відповіді на контрольні питання;
- блок-схему та опис функціонування програми;
- тексти програм з поясненнями (коментарями);
- результати виконання (графіки, таблиці, перетворені тексти);
- висновки - вказується призначення роботи, можливі варіанти вдосконалення та які знання отримано в ході виконання роботи.
- звіт повинен бути написаний українською мовою, акуратно та грамотно, з дотриманням правил оформлення ділової документації.

Лабораторна робота №1

Тема роботи: Симетричні методи шифрування інформації.

Мета роботи: Навчитися опрацьовувати (шифрувати та дешифрувати) файли на основі методів симетричного шифрування.

Теоретичні відомості

Симетричне шифрування буває двох видів:

– *Блокове шифрування* - інформація розбивається на блоки фіксованої довжини (наприклад, 64 або 128 біт), після чого ці блоки по черзі шифруються. Причому, у різних алгоритмах шифрування або навіть у різних режимах роботи того самого алгоритму блоки можуть шифруватися незалежно один від іншого або "зі зчепленням" - коли результат шифрування поточного блоку даних залежить від значення попереднього блоку або від результату його шифрування.

– *Потокове шифрування* – застосовується у випадку коли інформацію неможливо розбити на блоки - скажемо, якийсь потік даних, кожний символ якого повинен бути зашифрований і відправлений, не чекаючи інших даних, достатніх для формування блоку. Тому алгоритми потокового шифрування шифрують дані побітно або посимвольно.

Блокові шифри бувають двох основних видів:

1. *шифри перестановки* – переставляють елементи відкритих даних (біти, букви, символи) у деякому новому порядку;
2. *шифри заміни* – заміняють елементи відкритих даних на інші елементи за певним правилом.

Шифри заміни діляться на дві групи: *моноалфавітні* (код Цезаря) та *поліалфавітні* (шифр Відженера). У *моноалфавітних* шифрах заміни буква вихідного тексту заміняється на іншу, заздалегідь певну букву.

Одним з найпростіших шифрів заміни є шифр Цезаря (100-44 р. до н. є.). Літери абетки тут ототожнені з цифрами. В системі Цезаря використано 26 символів (26 літер латинської абетки), які перенумеровані числами від 0 до 25 (див. табл.1.).

Таблиця 1. Таблиця шифрування в системі Цезаря

A	00	H	07	O	14	V	21
B	01	I	08	P	15	W	22
C	02	J	09	Q	16	X	23
D	03	K	10	R	17	Y	24
E	04	L	11	S	18	Z	25
F	05	M	12	T	19		
G	06	N	13	U	20		

Шифрування здійснюється відповідною підстановкою: *00 (a) – 03 (d); 01 (b) – 04 (e); 02 (c) – 05 (f);...; 25 (z) – 02 (c).* Це означає, що в шифрограмі кожну літеру явного тексту замінюють на літеру, розташовану в абетці на три позиції далі. Висловлюючись сучасною мовою, римляни застосовували операцію додавання до номера літери числа 3 за модулем 26 (1):

$$C = P + 3(mod\ 26) \qquad (1)$$

Де, *C* - номер літери в криптограмі;

P - номер відповідної літери в явному тексті.

Наприклад, латинському слову *imperium* (імперія) відповідає криптограма *Ipshulxp,* а латинському тексту *Veni, vidi, vici* (прийшов, побачив, переміг), коли з нього викинути коми й пропуски між словами, відповідає криптограма *yhqlylglylfl.*

Безключові шифри переставляють символи, використовуючи запис вихідного тексту одним способом (наприклад, рядок за рядком) і передачу цього тексту в іншому порядку (наприклад, стовпець за стовпцем). Перестановка робиться у всьому вихідному тексті. Інший метод полягає в тому, щоб розділити вихідний текст на групи заздалегідь певного розміру (блоки), а потім використовувати ключ, щоб переставити символи в кожному блоці окремо.

Іншим порівняно простим методом шифрування є *квадрат Полібія* (або шифр в'язниці). Шифр дає змогу спілкуватися шляхом застосування одиничного базису. Для шифрування й дешифрування використовують ключ у вигляді квадрата у який вписано літери абетки, наприклад, української (33 літери та знаки " . " (крапка), " , " (кома), " ' " (апостроф) - усього 36 клітинок) (табл.2).

Таблиця.2. Квадрат Полібія

	1	**2**	**3**	**4**	**5**	**6**
1	а	б	в	г	ґ	д
2	е	є	ж	з	и	і
3	ї	й	к	л	м	н
4	о	п	р	с	т	у
5	ф	х	ц	ч	ш	щ
6	ь	ю	я	.	,	’

Кожну літеру або знак відкритого тексту замінюють парою цифр – номером рядка й номером стовпця в таблиці, на перетині яких міститься цей символ. Наприклад, для тексту *хто тут* криптограма виглядає так: 52 45 41 45 46 45. У випадку в'язниці вистукують кожну цифру криптограми, роблячи паузи між окремими цифрами.

У *поліалфавітних* підстановках для заміни деякого символу вихідного повідомлення в кожному випадку його появи послідовно використовуються різні символи з деякого набору, що є обмежений.

Опис методу шифрування перестановочним шифром

У перестановочних шифрах позиції символів повідомлення змінюються, але значення повідомлення залишається незмінним. *Простий шифр* – це спеціальна таблиця (сітка), куди повідомлення вписується одним способом, а потім зчитується – іншим. Повідомлення вписується в рядки сітки, а зчитується по стовпцях. Давній варіант цього способу полягав у записі повідомлення на смугу, обгорнену навколо циліндра, що потім розкручувалася й відправлялася з посильним. Обидві ці форми є простим чергуванням (розшаруванням), тільки використовуваним для різних цілей. Для розшифрування необхідно лише визначити глибину сітки (або діаметр циліндра).

Наприклад, повідомлення

«ЦЕ ПОВІДОМЛЕННЯ ДЛЯ ВІДПРАВКИ»

записується в таблицю по черзі по рядках. Результат заповнення таблиці з 3 рядків і 9 стовпців показаний в таблиці:

Ц	Е	П	О	В	І	Д	О	М
Л	Е	Н	Н	Я	Д	Л	Я	В
І	Д	П	Р	А	В	К	И	

Отримаємо шифротекст який утворюється шляхом зчитування символів порядкове: *ЦЛІЕЕДПНПОНРВЯАІДВДЛКОЯИМВ*. Описана система кодування носить назву простого перестановочного шифру.

Вдосконалення цієї методики полягає в тому, щоб читати стовпці сітки в більше складному порядку, чим просто зліва на право. Для вказання порядку зчитування стовпців можна використати ключове слово, алфавітне впорядкування букв якого й визначає порядок читання стовпців.

Спробуємо застосувати в якості ключа слово *КОМБАЙН*, як текст повідомлення візьмемо текст *«ЦЕ ПОВІДОМЛЕННЯ НЕОБХІДНО ВІДПРАВИТИ»*. На рис.1. показані дві таблиці, заповнені текстом повідомлення й ключовим словом, при цьому ліва таблиця відповідає заповненню до перестановки, а права таблиця – заповненню після перестановки.

К	*О*	*М*	*Б*	*А*	*Й*	*Н*
4	*7*	*5*	*2*	*1*	*3*	*6*
Ц	Е	П	О	В	І	Д
О	М	Л	Е	Н	Н	Я
Н	Е	О	Б	Х	І	Д
Н	О	В	І	Д	П	Р
А	В	И	Т	И		

А	*Б*	*Й*	*К*	*М*	*Н*	*О*
1	*2*	*3*	*4*	*5*	*6*	*7*
В	О	І	Ц	П	Д	Е
Н	Е	Н	О	Л	Я	М
Х	Б	І	Н	О	Д	Е
Д	І	П	Н	В	Р	О
И	Т		А	И		В

Рис.1. Горизонтальний перестановочний шифр

У верхньому рядку лівої таблиці записаний ключ, а номера під буквами ключа визначені відповідно до природного порядку відповідних букв ключа в алфавіті. Якби в ключі зустрілися однакові букви, вони б були пронумеровані зліва направо. У правій таблиці стовпці переставлені відповідно до впорядкованих номерів букв ключа.

При зчитуванні вмісту правої таблиці по стовпцях і запису шифртекста групами по чотири букви одержимо шифроване повідомлення:

ВНХДИ ОЕБІТ ІНІП ЦОННА ПЛОВИ ДЯДР ЕМЕОВ

КОНТРОЛЬНІ ПИТАННЯ

1. Особливості алгоритмів симетричного та асиметричного шифрувань.
2. Поняття блокового шифрування.
3. Шифрування в системі Цезаря.
4. Шифрування з допомогою перестановочного шифру.

ПОРЯДОК ВИКОНАННЯ РОБОТИ

1. Написати програму на мові С++ (чи іншій за згодою викладача) яка виконує криптографічні перетворення (шифрування та дешифрування) над файлами за одним з методів **симетричного шифрування** відповідно до заданого варіанту приведеного в таблиці.

№	Завдання
1	Модифікація шифру Цезаря при якій текст розбивається на частини по 8 символів і у межах кожної частини здійснюється «різний» зсув, де ключ – масив зсувів кожного блоку. Результат представити у рядковому представленні
2	Модифікація шифру Цезаря при якій текст розбивається на частини по 16 символів і у межах кожної частини здійснюється «різний» зсув, де ключ – масив зсувів кожного блоку. Результат представити в бітовому представленні (в шістнадцятковій системі числення)
3	Модифікація шифру Цезаря при якій текст розбивається на частини по

	32 символи і у межах кожної частини здійснюється «різний» зсув, де ключ – масив зсувів кожного блоку. Результат представити у вигляді кодів символів.
4	Шифр Полібія для укр. абетки. Результат представити в бітовому представленні (в шістнадцятковій системі числення)
5	Шифр Полібія для англ. абетки. Результат представити у вигляді кодів символів.
6	Шифр Цезаря, що передбачає зсув на 8 символів даних англійської абетки. Результат представити у рядковому представленні.
7	Перестановочний шифр із ключем для укр. мови. Результат представити в бітовому представленні (в шістнадцятковій системі числення)
8	Шифр Цезаря що передбачає зсув на 5 символів даних української абетки. Результат представити в бітовому представленні (в шістнадцятковій системі числення)
9	Перестановочний шифр із ключем для англ. мови. Результат представити в бітовому представленні (в шістнадцятковій системі числення)
10	Перестановочний шифр із ключем для англ. мови. Результат представити у вигляді кодів символів.
11	Шифр Цезаря що передбачає зсув на 3 символи даних української абетки. Результат представити у вигляді кодів символів.
12	Перестановочний шифр із ключем для укр. мови. Результат представити у вигляді кодів символів.
13	Шифр Цезаря, що передбачає зсув на 8 символів даних англійської абетки. Результат представити у рядковому представленні.
14	Шифр Полібія для англ. абетки. Результат представити у вигляді кодів символів.
15	Шифр Цезаря що передбачає зсув на 5 символів даних української абетки. Результат представити в бітовому представленні (в шістнадцятковій системі числення)
16	Модифікація шифру Цезаря при якій текст розбивається на частини по 8 символів і у межах кожної частини здійснюється «різний» зсув, де ключ – масив зсувів кожного блоку. Результат представити у рядковому представленні
17	Модифікація шифру Цезаря при якій текст розбивається на частини по 16 символів і у межах кожної частини здійснюється «різний» зсув, де ключ – масив зсувів кожного блоку. Результат представити в бітовому представленні (в вісімковій системі числення)

18	Модифікація шифру Цезаря при якій текст розбивається на частини по 32 символи і у межах кожної частини здійснюється «різний» зсув, де ключ – масив зсувів кожного блоку. Результат представити у вигляді кодів символів.
19	Шифр Полібія для укр. абетки. Результат представити в бітовому представленні (в двійковій системі числення)
20	Шифр Полібія для англ. абетки - результат в цифровому представленні
21	Шифр Цезаря що передбачає зсув на 3 символи даних української абетки. Результат представити у вигляді кодів символів.
22	Перестановочний шифр із ключем для укр. мови. Результат представити в бітовому представленні (в шістнадцятковій системі числення)
23	Шифр Цезаря, що передбачає зсув на 8 символів даних англійської абетки. Результат представити у рядковому представленні.
24	Перестановочний шифр із ключем для англ. мови. Результат представити в бітовому представленні (в вісімковій системі числення)
25	Перестановочний шифр із ключем для англ. мови. Результат представити у вигляді кодів символів.
26	Шифр Цезаря що передбачає зсув на 5 символів даних української абетки. Результат представити в бітовому представленні (в двійковій системі числення)
27	Перестановочний шифр із ключем для укр. мови. Результат представити у вигляді кодів символів.
28	Шифр Цезаря що передбачає зсув на 3 символи даних української абетки. Результат представити у вигляді кодів символів.
29	Шифр Полібія для англ. абетки. Результат представити у вигляді кодів символів.
30	Шифр Цезаря, що передбачає зсув на 8 символів даних англійської абетки. Результат представити у рядковому представленні.

2. Здійснити компіляцію програми.

3. Результати подати у вигляді текстів програми, відповідних пояснень та скріншотів.

4. *Додаткове завдання.* Провести порівняння розподілу ймовірностей наявних символів файлу до та після шифрування. Результати ймовірностей представити у табличній та графічній формах для кожного символу

5. Оформити звіт згідно вимог.

Лабораторна робота №2

Тема роботи: Блокове шифрування інформації та шифри моноалфавітної заміни.

Мета роботи: Навчитися опрацьовувати (шифрувати та дешифрувати) файли з допомогою блокового шифрування інформації, методів моноалфавітної заміни та методу гамування.

Теоретичні відомості

В алгоритмі блокової перестановки в кожному блоці змінюється послідовність деяких подблоків всередині блоку, наприклад байт або біт в слові, причому порядок перестановок визначається ключем. Нехай є деяке вихідне повідомлення «MESSAGE», яке необхідно закодувати (Рис 1). Це повідомлення має довжину в 7 байт у випадку використання ASCII коду. Розіб'ємо даний блок тексту на три підблоки: «MES», «SA» і «GE».

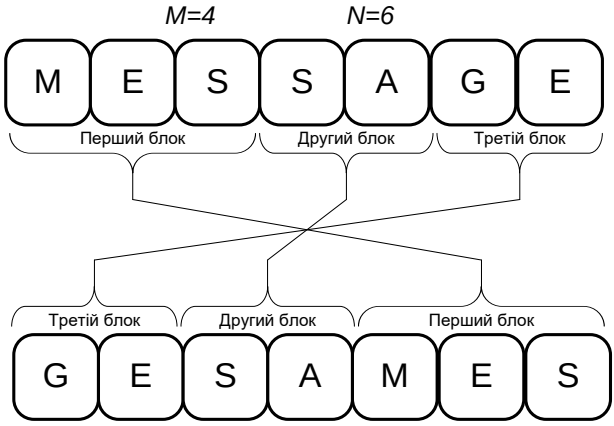

Рис.1. Приклад блокової перестановки

Після перестановки ми отримаємо зашифроване повідомлення: «GESAMES». Дешифрування відбувається за тією ж схемою, але в зворотному порядку. Механізм шифрування / дешифрування цього блокового шифру реалізується за допомогою наступного алгоритму:

```
/ / Функція яка здійснює перестановку в блоці
/ / Str - вихідний текст до перестановки
/ / N і m – границі підблоків в блоці
/ / Return - текст після перестановки
/ / Правило формування:
/ / 1. блок розбивається на три підблоки, межами яких служать m і n
/ / 2. Третій підблок записується на місце першого
```

```
/ / 3. Другий - на місце другого
/ / 4. Перший - на місце третього
function TForm1.cryptblock (str:block;n,m:integer):block;
var
i,k:integer;
retstr :block;
begin
k := 1;
for i:=n to sblock do begin
retstr[k] := str[i];
k:=k+1;
end;
for i:=m to n-1 do begin
retstr[k] := str[i];
k:=k+1;
end;
for i:=1 to m-1 do begin
retstr[k] := str[i];
k:=k+1;
end;
cryptblock := retstr;
end;
```

Аналогічно проводиться операція перестановки над групами біт. Примітка: для блоків великої довжини метод перестановок стає неефективним, так як і довжина підблоків в цьому випадку також велика. Тому шифрування повинно здійснюватися в декілька раундів, тобто перемішування проводиться декілька разів над одним і тим же блоком, але з різними значеннями коефіцієнтів M і N. При дешифруванні необхідно відтворити цю послідовність псевдовипадкових чисел у зворотному порядку.

В алгоритмі шифрування методом зсуву (скремблера) вихідний текст розбивається на підблоки і всередині кожного такого підблока реалізується операція циклічного зсуву на кілька біт в зазначеному напрямку. Наприклад, для блоку довжиною в 7 байт, де шифрування здійснюється циклічним зсувом вліво на 3 біти, схема шифрування буде наступною (Рис.2).

Відповідно для розкодування необхідно здійснити циклічний зсув підблоку в протилежну сторону на таку ж кількість біт. Зазвичай величина зсуву та його напрямок визначається паролем, що забезпечує прив'язку до ключа. У тому випадку, якщо блок шифрується по байтах, слід виключати

ситуації, при яких розмір зсуву кратний 8.

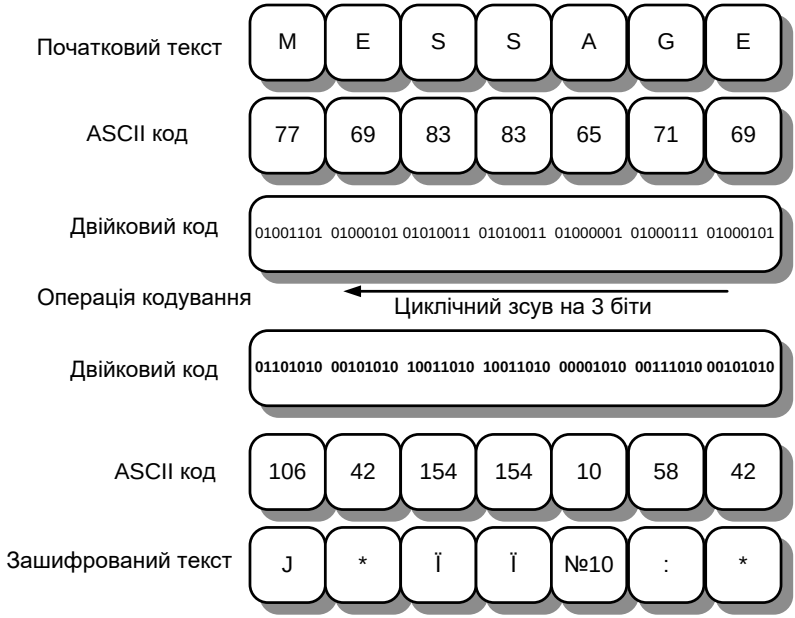

Рис.2. Шифрування методом циклічного зсуву

Програмна реалізація такого шифру містить три функціональних блоки, де кожен може бути оформлений у вигляді окремої процедури: «розпакування» (procedure в бітове представлення, процедура, що здійснює циклічний зсув), і процедура «упаковки» бітового представлення в текстовий блок.

Шифрування методом гамування

В адитивних шифрах символи початкового повідомлення замінюються числами, які складаються по модулю з числами гами. Ключем шифру є гамма, символи якої послідовно повторюються. Шифр з нескінченною випадковою гамою, наприклад, «одноразовий блокнот» є криптографічно абсолютно стійкими. *Примітка*. Результатом використовуваної операції додавання цілих чисел по модулю є залишок від ділення націло, наприклад:

$$(7+11) \bmod 5 = 18 \bmod 5 = 3$$

Гамування є потоковою процедурою, чутливою до синхронізації гами і шифротексту. У разі пропуску одного символу, весь наступний текст буде дешифрований з помилками. У сучасних стандартах шифрування використовується побітове додавання повідомлення і гами по модулю 2, так

як це відповідає операції (XOR - виключне АБО), що апаратно реалізована в арифметико-логічному пристрої процесора.

Метод складання по модулю N. Перед шифруванням символи повідомлення замінюються їх номерами в алфавіті. Основа модуля N визначає кількість символів у використовуваному алфавіті. Шифрування виконується за формулою:

$$C_i = (T_i + G_i) \bmod N$$

при цьому отриманий N -й символ залишається N -му, а не нульовим. Потім виконується заміна отриманих чисел на літери шифрограми. Дешифрування виконується за формулою:

$$T_i = (C_i - G_i + N) \bmod N$$

де T_i – це символи вихідного повідомлення,

C_i - символи зашифрованого повідомлення,

G_i - символи гами.

Наприклад, використовуючи український алфавіт, знак пробілу і десять цифр (табл. 1) і гаму «ТИГР» зашифруємо повідомлення «ЛЕГІОН_27».

Таблиця 1. Алфавіт «Українські букви та цифри»

Буква	А	Б	В	Г	Ґ	Д	Е	Є	Ж	З	И	І
Код	01	02	03	04	05	06	07	08	09	10	11	12
Буква	Ї	Й	К	Л	М	Н	О	П	Р	С	Т	У
Код	13	14	15	16	17	18	19	20	21	22	23	24
Буква	Ф	Х	Ц	Ч	Ш	Щ	Ь	Ю	Я	Пробіл		
Код	25	26	27	28	29	30	31	32	33	34		
Цифра	0	1	2	3	4	5	6	7	8	9		
Код	35	36	37	38	39	40	41	42	43	44		

Примітка. Цифри 0, 3, 4 є подібними на букви О, З, Ч тому в таблиці вони зображені підкресленими. Отримаємо шифрограму:

T	Л	Е	Г	І	О	Н	_	2	7
G	Т	И	Г	Р	Т	И	Г	Р	Т
T	16	7	4	12	19	18	34	37	42
G	23	11	4	21	23	11	4	21	23
T+G	39	18	8	33	42	29	38	58	65
mod N	39	18	8	33	42	29	38	14	21
C	***4***	***Н***	***Є***	***Я***	***7***	***Ш***	***З***	***Й***	***Р***

Рис. 3. Схема шифрування гамуванням за модулем ***N***

Дешифруємо з тією ж гаммою повідомлення «**ХНДИТБ4ИЗЇ**» (рис. 4). Отримаємо вихідний текст «ВЕБ_900_КБ»

C	Х	Н	Д	И	Т	Б	4	И	3	Ї
G	Т	И	Г	Р	Т	И	Г	Р	Т	И
C	26	18	6	11	23	2	39	11	38	13
G	23	11	4	21	23	11	4	21	23	11
C-G	3	7	2	-10	0	-9	35	-10	15	2
+44	47	51	46	34	44	35	79	34	59	46
mod N	3	7	2	34	0	35	35	34	15	2
0 → 44	3	7	2	34	44	35	35	34	15	2
T	В	Е	Б	_	9	0	0	_	К	Б

Рис. 4. Схема дешифрування гамуванням за модулем ***N***

Примітка. Для обчислення залишку від ділення націло можна використовувати стандартну програму операційної системи Microsoft Windows «Калькулятор», Вид - Інженерний - кнопка «Mod».

Шифри моноалфавітної заміни

У XVI столітті французький дипломат Блез де Віженер запропонував модифікацію шифру заміни, яка згодом отримала його ім'я. Шифрування здійснюється по таблиці, що представляє собою квадратну матрицю розмірністю *Xn*, де *n* – кількість символів використовуваного алфавіту. У даному шифрі ключ задається фразою з ***d*** літер. Ключова фраза підписується з повторенням під повідомленням. Буква шифротексту знаходиться на перетині стовпця, який визначається буквою відкритого тексту, і рядка, що визначається буквою ключа. Якщо ключ є коротший за повідомлення його використовують декілька разів:

$$Vig_d(m_i)=(m_i+k_{i \bmod d})(\mathrm{mod}\ n),$$

де mi, ki, Vigd (mi) – порядкові номери в алфавіті наступних символів відкритого тексту, ключ та шифротекст відповідно. Зворотне перетворення має наступний вид:

$$Vig_d^{-1}(m_i)=(m_i-k_{i \bmod d}+n)(\mathrm{mod}\ n)$$

На рис.5 показана таблиця Віженера для української мови. Перший рядок містить всі символи алфавіту, наступні рядки – утворюються з попереднього методом циклічного зсуву символів на один вліво.

	а	б	в	г	ґ	д	е	є	ж	з	и	і	ї	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ь	ю	я
а	а	б	в	г	ґ	д	е	є	ж	з	и	і	ї	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ь	ю	я
б	б	в	г	ґ	д	е	є	ж	з	и	і	ї	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ь	ю	я	а
в	в	г	ґ	д	е	є	ж	з	и	і	ї	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ь	ю	я	а	б
г	г	ґ	д	е	є	ж	з	и	і	ї	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ь	ю	я	а	б	в
ґ	ґ	д	е	є	ж	з	и	і	ї	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ь	ю	я	а	б	в	г
д	д	е	є	ж	з	и	і	ї	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ь	ю	я	а	б	в	г	ґ
е	е	є	ж	з	и	і	ї	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ь	ю	я	а	б	в	г	ґ	д
є	є	ж	з	и	і	ї	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ь	ю	я	а	б	в	г	ґ	д	е
ж	ж	з	и	і	ї	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ь	ю	я	а	б	в	г	ґ	д	е	є
з	з	и	і	ї	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ь	ю	я	а	б	в	г	ґ	д	е	є	ж
и	и	і	ї	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ь	ю	я	а	б	в	г	ґ	д	е	є	ж	з
і	і	ї	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ь	ю	я	а	б	в	г	ґ	д	е	є	ж	з	и
ї	ї	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ь	ю	я	а	б	в	г	ґ	д	е	є	ж	з	и	і
й	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ь	ю	я	а	б	в	г	ґ	д	е	є	ж	з	и	і	ї
к	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ь	ю	я	а	б	в	г	ґ	д	е	є	ж	з	и	і	ї	й
л	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ь	ю	я	а	б	в	г	ґ	д	е	є	ж	з	и	і	ї	й	к
м	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ь	ю	я	а	б	в	г	ґ	д	е	є	ж	з	и	і	ї	й	к	л
н	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ь	ю	я	а	б	в	г	ґ	д	е	є	ж	з	и	і	ї	й	к	л	м
о	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ь	ю	я	а	б	в	г	ґ	д	е	є	ж	з	и	і	ї	й	к	л	м	н
п	п	р	с	т	у	ф	х	ц	ч	ш	щ	ь	ю	я	а	б	в	г	ґ	д	е	є	ж	з	и	і	ї	й	к	л	м	н	о
р	р	с	т	у	ф	х	ц	ч	ш	щ	ь	ю	я	а	б	в	г	ґ	д	е	є	ж	з	и	і	ї	й	к	л	м	н	о	п
с	с	т	у	ф	х	ц	ч	ш	щ	ь	ю	я	а	б	в	г	ґ	д	е	є	ж	з	и	і	ї	й	к	л	м	н	о	п	р
т	т	у	ф	х	ц	ч	ш	щ	ь	ю	я	а	б	в	г	ґ	д	е	є	ж	з	и	і	ї	й	к	л	м	н	о	п	р	с
у	у	ф	х	ц	ч	ш	щ	ь	ю	я	а	б	в	г	ґ	д	е	є	ж	з	и	і	ї	й	к	л	м	н	о	п	р	с	т
ф	ф	х	ц	ч	ш	щ	ь	ю	я	а	б	в	г	ґ	д	е	є	ж	з	и	і	ї	й	к	л	м	н	о	п	р	с	т	у
х	х	ц	ч	ш	щ	ь	ю	я	а	б	в	г	ґ	д	е	є	ж	з	и	і	ї	й	к	л	м	н	о	п	р	с	т	у	ф
ц	ц	ч	ш	щ	ь	ю	я	а	б	в	г	ґ	д	е	є	ж	з	и	і	ї	й	к	л	м	н	о	п	р	с	т	у	ф	х
ч	ч	ш	щ	ь	ю	я	а	б	в	г	ґ	д	е	є	ж	з	и	і	ї	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц
ш	ш	щ	ь	ю	я	а	б	в	г	ґ	д	е	є	ж	з	и	і	ї	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч
щ	щ	ь	ю	я	а	б	в	г	ґ	д	е	є	ж	з	и	і	ї	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш
ь	ь	ю	я	а	б	в	г	ґ	д	е	є	ж	з	и	і	ї	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ
ю	ю	я	а	б	в	г	ґ	д	е	є	ж	з	и	і	ї	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ь
я	я	а	б	в	г	ґ	д	е	є	ж	з	и	і	ї	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ь	ю

Рис.5. Таблиця Віженера для українського алфавіту

Наприклад потрібно зашифрувати слово: *ДНІПРО*

Використавши для цього ключ *САМ*. Виходячи з попередньої інформації запишемо рядок початкового тексту разом з розташованим під ним рядком з циклічно повторюваним ключем:

ДНІПРО
САМСАМ

В результаті шифрування ,початковий елемент якого відображений на рис.6, отримаємо шифротекст: *ЦНЧЄРБ*

Для криптоаналізу шифру Віженера можна використовувати метод Казіскі. У середині XIX століття німецький математик Казіскі запропонував визначати довжину парольної фрази за відстанню між однаковими фрагментами шифротексту. Припустимо, знайдено два однакових фрагмента шифротексту, відстань між якими становить 20 символів. Це може означати, що два однакових фрагмента відкритого тексту були зашифровані з однією і тією ж позицією ключа, що дозволяє припустити, що парольна фраза має довжину 4, 5, 10 або 20 символів. Дізнавшись (або вгадавши) довжину парольного фрази можна здійснити частотний криптоаналіз шифротексту для вибірки кожного *i-го* символу шифротексту.

Далі розшифрування здійснюється наступним чином. Під літерами шифротексту послідовно записуються літери ключа; в рядку таблиці, відповідної чергової букві ключа та відбувається пошук відповідної букви шифротексту. Буква, яка знаходиться над нею в першому рядку таблиці є відповідною буквою вихідного тексту. Для збільшення надійності шифру можна рекомендувати його використання після попередньої псевдовипадковою перестановки літер в кожному рядку таблиці.

Шифрування біграмами

На початку XVI століття абат з Німеччини Йоганн Трісемус запропонував шифрування двох букв одночасно. Шифри, які почали застосовувати даний підхід отримали назву ***біграмних.*** Зазвичай такі шифри використовують таблиці заповнені символами використовуваного алфавіту. Найбільш відомий шифр біграм отримав назву Playfair. Він застосовувався Великобританією в Першій світовій війні. Для використання шифру Трісемуса зазвичай використовувалися таблиці для записування літер абетки й ключове слово. У таблицю спочатку вписувалося по рядках ключове слово, причому повторювані літери відкидалися. Потім ця таблиця доповнювалася літерами абетки, які не увійшли до неї одна за одною.

Для української абетки таблиця Трісемуса може мати розмір 4x8. Оберемо ключ слово БАНДЕРОЛЬ. Розглянемо шифрувальну таблицю з ключем, поданим на рис.6.

Б	А	Н	Д	Е	Р	О	Л
Ь	В	Г	Є	Ж	З	И	І
Ї	Й	К	М	П	С	Т	У
Ф	Х	Ц	Ч	Ш	Щ	Ю	Я

Рис.6. Шифрувальна таблиця з ключовим словом

Відкритий текст розбивався на пари літер (біграми) і текст зашифрованого повідомлення будувався за правилами:

1. Відкритий текст вихідного повідомлення розбивається на пари літер (біграми). Текст повинен мати парну кількість літер і в ньому не повинно бути біграм, котрі містили б дві однакові літери. Якщо цих вимог недодержано, то текст модифікується навіть через незначні орфографічні помилки.

2. Якщо дві літери відкритого тексту не потрапляють в один рядок чи стовпець (як, наприклад, літери А та И на рис.6.), то відшукують літери в кутах прямокутника, що визначається даною парою літер. (У нашому прикладі це літери АИ та ОВ. Пара літер АИ перетворюється в пару ОВ.) Послідовність літер у біграмі шифртексту має бути дзеркально розташована стосовно послідовності літер у біграмі відкритого тексту;

3. Якщо обидві літери біграми відкритого тексту належать до одного стовпця таблиці, то за літери шифртексту вважаються літери, котрі розміщено під ними (наприклад, біграма НК дає біграму шифртексту ГЦ). Якщо при цьому літера відкритого тексту перебуває в нижньому рядку, то для шифртексту береться відповідна літера з верхнього рядка того самого стовпця (наприклад, біграма ВХ дає біграму шифртексту ЙА);

4. Якщо обидві літери біграми відкритого тексту належать до одного рядка таблиці, то за літери шифртексту вважаються літери, котрі розміщено праворуч від них (наприклад, біграма НО дає біграму шифртексту ДЛ). Якщо при цьому літера відкритого тексту перебуває в крайньому правому стовпці, то для шифру беруть відповідну літеру з лівого стовпця в тому самому рядку (наприклад, біграма ХЯ дає біграму шифртексту ЦФ).

Зашифруємо текст:

ВСЯКИЙ СВОГО ЩАСТЯ КОВАЛЬ

Розбиття цього тексту на біграми дає:

ВС ЯК ИЙ СВ ОГ ОЩ АС ТЯ КО ВА Л Ь

Дана послідовність біграм відкритого тексту перетворюється за допомогою шифрувальної таблиці (рис.6.) на таку послідовність біграм шифртексту:

ЗЙ ЦУ ВТ ЙЗ НИ РЮ РЙ УЮ ТН ЙВ БІ

При розшифровуванні застосовується зворотний порядок дій.

Шифрування біг рамами з подвійним квадратом

В 1854 році англієць Чарльз Уїтстон розробив новий метод шифрування біграмами, котрий називають подвійним квадратом. Свою назву цей шифр дістав за аналогією з полібіанським квадратом. Шифр Уїтстона відкрив новий

етап в історії розвинення криптографії. На відміну від полібіанського, шифр "подвійний квадрат" використовує одразу дві таблиці(рис.7), що розміщуються на одній горизонталі, а шифрування здійснюється біграмами, як у шифрі Плейфейра. Ці модифікації призвели до появи якісно нової криптографічної системи ручного шифрування. Шифр "подвійний квадрат" виявився значно надійнішим.

Пояснімо процедуру шифрування цим шифром на прикладі. Нехай є дві таблиці з довільно розташованими в них українськими абетками. Перед зашифровуванням повідомлення розбивають на біграми. Кожна біграма шифрується окремо. Першу літеру біграми відшукують у лівій таблиці, а другу літеру – у правій таблиці. Потім подумки будують прямокутник у такий спосіб, аби літери біграми містилися в його протилежних вершинах. Інші дві вершини цього прямокутника надають літери біграми шифртексту.

Ж	Щ	Н	Ю	Р			И	Ч	Г	Я	Т
И	Т	Ь	Ц	Б			Ф	Ж	Ь	М	О
Я	М	Е	Л	С			З	Ю	Р	В	Щ
В	І	П	Ч	А			Ц	У	П	Е	Л
Х	Д	У	О	К			Є	А	Н	Б	Х
З	Є	Ф	Г	Ш			І	К	С	Ш	Д

Рис.7. Довільно розташовані символи шифру "подвійний квадрат"

Припустімо, що шифрується біграма вихідного тексту ИЛ. Літера И міститься в стовпці 1 і рядку 2 лівої таблиці. Літера Л міститься в стовпці 5 і рядку 4 правої таблиці. Це означає, що прямокутник утворено рядками 2 та 4, а також стовпцями 1 лівої таблиці та 5 правої таблиці. Отже, до біграми шифртексту входять літера О, розміщена в стовпці 5 і рядку 2 правої таблиці, і літера В, розташована в стовпці 1 і рядку 4 лівої таблиці, тобто отримуємо біграму шифртексту ОВ.

Якщо обидві літери біграми повідомлення містяться в одному рядку, т і літери шифртексту беруть з того самого рядка. Першу літеру біграми шифртексту беруть з лівої таблиці в стовпці, який відповідає другій літері біграми повідомлення. Друга ж літера біграми шифртексту береться з правої таблиці в стовпці, який відповідає першій літері біграми повідомлення. Тому біграма повідомлення То перетворюється в біграму шифртексту БЖ. В аналогічний спосіб шифруються всі біграми повідомлення:

Повідомлення		ПО		ВТ		ОР		НА		ПЕ		РЕ		ДА		ЧА
Шифртекст		ЛЬ		ЛЖ		НЛ		ЧУ		ЧП		ЯА		ДА		УО

Рис.8. Процес шифрування

Шифрування методом подвійного квадрата надає достатньо стійкий до розкриття і простий у застосуванні шифр. Зламування шифр тексту "подвійний квадрат" потребує певних аналітичних зусиль, при цьому довжина повідомлення має бути не менш ніж тридцять рядків.

КОНТРОЛЬНІ ПИТАННЯ

1. Поняття блокового шифрування інформації?
2. Мета та особливості використання операції зсуву?
3. Режими шифрування методом Скремблера?
4. Поняття гамування та його особливості?
5. Способи підвищення якості шифрування шифру Віженера?
6. Особливості реалізації шифрування методом біграм та біграм з подвійним квадратом?

Порядок виконання роботи

1. Написати програму на мові С++ (чи іншій за згодою викладача) яка виконує шифрування повідомлення методом накладення гамми використовуючи таблицю 1. Гамма шифру і повідомлення вибираються згідно з варіантом з табл. 3.

Таблиця 3.

Варіант	Гамма	Повідомлення
1	БІЛОВИД	АСВААН_25000_БОЛ
2	БІЛОГОСТ	НЕОПРОМА_3000_Т
3	БІЛОМИР	ЧЕГДОМИН_5000_Т
4	БІЛОСЛАВ	АМУРЗЕТ_35000_ШТ
5	БІЛОТУР	ХІРОСІМА_06081945
6	БЛАГОВИД	ПІРЕНИЇ_2500_ТОНН
7	БЛАГОВІСТ	ДИКОПОЛЕВА_65
8	БЛАГОДАР	ДНІАППРО_1500_БАР
9	БОГУСЛАВ	ГРОВКМАДА_150_КГ
10	БОЖЕДАР	ШМАКІВКА_5000_ШТ

11	БОЛЕСЛАВ	ХАЛКІНГОЛ_1939_Г
12	БОРИМИР	ЧОРАПНЕМОРЕ_201
13	БОРИМИСЛ	ХЕЙЛУНЦЗЯН_200_КГ
14	БОРИСЛАВ	МАРАПУНЬКА_19
15	БРАТИМИР	ТЯНЬАНЬМЕНЬ_1989
16	БРАТИСЛАВ	НЕРЮНГРІ_230_КМ
17	БРАТОМИЛ	ПРІАМУРСК_50А
18	БУДИВИД	ЕГЕРШЕЛЬД_100_КГ
19	БУДИМИР	ЧУМІКАН_1500_ШТУК
20	БУДИСЛАВ	БІРОБІДЖАН_100_Т
21	БУЙМИР	НАПЕНКІЙ_500_КГ
22	БУРЕВІЙ	СУЙФЕНЬХЕ_20_ЧЕЛ
23	ВЕЛЕМУДР	АПРИМАЕП_680063
24	ВЕЛИЧАР	АЯНОМАЙСКІЙ_200_Т
25	ВЕРБАН	ОРРААВПМА_300_Т
26	ВЕРНИДУБ	ААМУРЗЕТ_35000_ШТ
27	ВЕРНИСЛАВ	ХАТПСАТ_06081945
28	ВЕСЕЛАН	ПІРЕОИЇ_2500_ТОНН
29	ВИТОМИР	ДИКОЬОПОЛЕВА_65
30	ВИШАТА	ДНІАППРО_1500_БАР

3. Використовуючи алфавіт (табл. 1), дешифрувати повідомлення, яке зашифроване методом гамування. Гамма шифру та повідомлення вибираються згідно з варіантом з таблиці 4.

Таблиця 4

Варіант	Гамма	Шифрограма
1	ДАЛЕМИЛ	ЙЇЇЛЩД18ЗБДЕЯШС
2	ДАЛЕМИР	ТЬЩХИЩЇИЄЦЗЙ1Д9
3	ДАЛІБОР	0Ф26ОЯ7ЇВ_8ЖС6ММЬ
4	ДАНИЛО	7ОЯМ_ДННРЗАНФАЕ5
5	ДАНКО	Т_05ЮЦСИЗКЇВ5Я2Н
6	ДАНТУР	СВИЬЇ6ЇПЙР03122М
7	ДАРИБОГ	УЇТЙЩ5Л_ДВНЇБЗЕАН

8	ДАРОМИР	ЩПФЄ2Ь2ЕЗДИВМЕНН2
9	ДЕРЖИКРАЙ	ЩЗ20715ФИТЛПИТШЩИК
10	ДЕРЖИСЛАВ	Я6ШЮТУЕНЙЙКДЇМО
11	ДИБАЧ	0Л45ЧЬХДПЙИБЗЬ10
12	ДИВОЗІР	ДЗЧЦ_РУ6ЇЗЇЕУНРЮ
13	ДМИТРО	ЖРБУУЄ_СГЖОЧ2ВФА1
14	ДОБРИБІЙ	РС7Р_СУЖХАП11И
15	ДОБРИВОД	ГБ2ФШ0Ц0И2БЧФХЮ
16	ДОБРИК	4Й4ИОТ1_ДНАБУ2НХ9
17	ДОБРИЛО	ИГЯЇЩТ6АВ_ЮГЙМБ
18	ДОБРИНЯ	Е0ЮУФЗХЙЇЕБШЦЗЇ
19	ДОБРИСВІТ	ЮЬЯЯ1ОТ0ГСИА0ЬЖМ
20	ДОБРОГОСТ	Є7ТИХ_ФИЖГГТВКЖЬ
21	ДОБРОДУМ	СУЛЇ1ГБ1Ф0ЙА1Е403
22	ДОБРОЛИК	ЄФЄДЦХФЬМДГ2КЇГУ
23	ДОБРОМИСЛ	ИШО0СЄХ9ГЙЇЇ7АГИ
24	ДОБРОСЛАВ	ПФХ0ЧГПШ7Й73568ДЧ
25	ДОМОРАД	ЬТЄФЩ9НИИПЗЇЙ016
26	ДОМОСЛАВ	ЙЇЇЛЩД183БДЕЯШС
27	ДОРОГОБУГ	ТЬЩХИЩЇИЄЦЗЙ1Д9
28	ДОРОГОМИР	0Ф26ОЯ7ЇВ_8ЖС6ММЬ
29	ДОРОГОМИСЛ	7ОЯМ_ДННР3АНФАЕ5
30	ДОРОГОСИЛ	Т_05ЮЦСИЗКЇВ5Я2Н

4. Оформити алгоритм функціонування програми та здійснити його опис

5. Оформити звіт в згідно вимог, результати подати у вигляді текстів програми, відповідних пояснень та скріншотів.

6. Додаткове завдання. Здійснити написання програми яка виконує шифрування даних над файлами за одним з методів відповідно до заданого варіанту приведеного в таблиці. Інформацію (текстовий блок для шифрування) взяти у відповідності до вибраної предметної галузі (узгоджується з викладачем) але не менше ніж 200 слів.

Таблиця 3

№	Завдання
1	Шифрування методом Віженера повідомлення на англійській мові (мін 60 слів) з довжиною ключа в 10 символів
2	Шифрування методом біграм фрази на українській мові. Ключове слово довільне та вводиться користувачем.
3	Шифрування методом біграм з використанням квадрата 5 на 5 фрази на укр. мові. Ключове слово вводиться користувачем.
4	Шифрування методом біграм з подвійним квадратом фрази на українській мові. Ключове слово вводиться користувачем.
5	Шифрування методом Віженера повідомлення на українській мові (мін 80 слів) з довжиною ключа в 6 символів
6	Шифрування методом біграм фрази на англійській мові. Ключове слово довільне та вводиться користувачем.
7	Шифрування методом біграм з подвійним квадратом з використанням квадрата 5 на 5 фрази на англійській мові. Ключове слово довільне та вводиться користувачем.
8	Шифрування методом Віженера повідомлення на англійській мові (мін 40 слів) із складеним ключем із двох слів. Обмеження на ведення по 6 символів.
9	Шифрування методом біграм без використання ключового слова.
10	Шифрування методом біграм з подвійним квадратом з використанням квадрата 5 на 5 фрази на українській мові. Ключове слово довільне та вводиться користувачем.
11	Шифрування методом Віженера повідомлення на українській мові (мін 60 слів) із складеним ключем м із двох слів. Обмеження на ведення по 5 символів.
12	Шифрування методом біграм з подвійним квадратом фрази на англійській мові. Ключове слово вводиться користувачем.
13	Шифрування методом біграм з подвійним квадратом з використанням квадрата 6 на 6фрази на англійській мові. Ключове слово вводиться користувачем.
14	Шифрування методом біграм без використання ключового слова фрази на українській мові.
15	Шифрування методом Віженера повідомлення на українській мові (мін 40 слів) із складеним ключем із двох слів. Обмеження на ведення по 5 символів.
16	Шифрування методом Віженера повідомлення на англійській мові (мін 60 слів) з довжиною ключа в 10 символів
17	Шифрування методом біграм фрази на українській мові. Ключове слово довільне та вводиться користувачем.
18	Шифрування методом біграм з використанням квадрата 5 на 5

	фрази на укр. мові. Ключове слово вводиться користувачем.
19	Шифрування методом біграм з подвійним квадратом фрази на українській мові. Ключове слово вводиться користувачем.
20	Шифрування методом Віженера повідомлення на українській мові (мін 80 слів) з довжиною ключа в 6 символів
21	Шифрування методом біграм фрази на англійській мові. Ключове слово довільне та вводиться користувачем.
22	Шифрування методом біграм з подвійним квадратом з використанням квадрата 5 на 5фрази на англійській мові. Ключове слово довільне та вводиться користувачем.
23	Шифрування методом Віженера повідомлення на англійській мові (мін 40 слів) із складеним ключем із двох слів. Обмеження на ведення по 6 символів.
24	Шифрування методом біграм без використання ключового слова.
25	Шифрування методом біграм з подвійним квадратом з використанням квадрата 5 на 5 фрази на українській мові. Ключове слово довільне та вводиться користувачем.
26	Шифрування методом Віженера повідомлення на українській мові (мін 60 слів) із складеним ключем м із двох слів. Обмеження на ведення по 5 символів.
27	Шифрування методом біграм з подвійним квадратом фрази на англійській мові. Ключове слово вводиться користувачем.
28	Шифрування методом біграм з подвійним квадратом з використанням квадрата 6 на 6фрази на англійській мові. Ключове слово вводиться користувачем.
29	Шифрування методом біграм без використання ключового слова фрази на українській мові.
30	Шифрування методом Віженера повідомлення на українській мові (мін 40 слів) із складеним ключем із двох слів. Обмеження на ведення по 5 символів.

Лабораторна робота №3

Тема роботи: Захист даних з допомогою комбінованих алгоритмів шифрування та застосування електронного цифрового підпису

Мета роботи: Вивчити принцип роботи асиметричного алгоритму шифрування на прикладі алгоритму RSA. Освоїти методику створення комбінованих алгоритмів шифрування, які поєднують переваги методів симетричної та асиметричної криптографії та навчитись застосовувати електронний цифровий підпис.

Теоретичні відомості

Як відомо з попередніх лабораторних робіт, алгоритми симетричного шифрування використовують ключі відносно невеликої довжини і тому можуть швидко здійснювати шифрування великих обсягів даних. При використанні алгоритму симетричного шифрування відправник і одержувач застосовують для шифрування й розшифрування даних один і той же секретний ключ. Таким чином, алгоритми симетричного шифрування ґрунтуються на припущенні про те, що зашифроване повідомлення не зможе прочитати ніхто, крім того хто володіє ключем для його розшифрування. При цьому, якщо ключ не скомпрометований, то при розшифруванні повідомлення автоматично виконується аутентифікація відправника, з огляду на те, що тільки він має ключ за допомогою якого можна розшифрувати повідомлення. Таким чином, для симетричних криптосистем актуальною є проблема безпечного розподілу секретних ключів. У зв'язку з цим без ефективної організації захищеного розподілу ключів використання звичайної системи симетричного шифрування в обчислювальних мережах є практично неможливою.

Вирішенням даної проблеми здійснюється є використання асиметричних алгоритмів шифрування – криптосистем з відкритим ключем. У них для зашифрування даних використовується – «відкритий ключ», а для розшифрування – інший, «закритий, секретний ключ». При цьому розуміється, що ключ розшифрування не може бути визначений з ключа зашифрування. В асиметричних криптосистемах відритий ключ і криптограма можуть бути відправлені по незахищених каналах. Концепція таких систем заснована на застосуванні одно направлених функцій.

В якості прикладу одно направленої функції може служити операція цілочисельного множення. Наприклад, задача - обчислення добутку двох великих цілих чисел p і q, $n = p * q$. Це відносно нескладне завдання для ЕОМ. Зворотне йому завдання – факторизація або розкладання на множники великого цілого числа практично є нерозв’язаною при достатньо великих значеннях n.

Іншим прикладом однонаправленої функції є модульна експонента з фіксованою основою і модулем. Наприклад, якщо $y = a^x$, то можна записати, що $x = loga\ (y)$. Завдання дискретного логарифмування формулюється наступним чином. Для відомих цілих a, n, y необхідно знайти таке число x, при якому $a^x (mod\ n) = y$. Наприклад, якщо $a = 2^{664}$ і $n = 2^{664}$ знаходження значення степеня x для відомого значення y вимагає здійснення близько 10^{26} операцій, що вимагає значного часу для сучасних персональних машин. У зв'язку з тим, що в даний час не вдалося довести, що не існує ефективного

алгоритму обчислення дискретного логарифму за прийнятний час, то модульна експонента також умовно віднесена до однонапрвлених функцій. Іншим важливим класом функцій, які використовуються при побудові криптосистем з відкритим ключем є, так звані, однонапрвлені функції з секретом. Функція відноситься до даного класу за умови, що вона є одно направленою і , крім того передбачає ефективне обчислення оберненої функції, якщо відомий секрет. У даній лабораторній роботі досліджується криптосистема RSA, яка використовує модульну експоненту з фіксованим модулем і показником степеня (тобто одно направлену функцію з секретом).

Шифр RSA

Шифр RSA є на сьогодні є найбільш популярною системою шифрування з відкритим ключем. RSA використовує розкладання великих чисел (кілька сот розрядів) на прості множники, що вимагає великих обчислень і визначає його стійкість. Першим етапом асиметричного шифрування є створення одержувачем шифрограми пари ключів. Процедура створення ключів RSA полягає в наступному:

1. Вибираються два простих числа *P* та *Q*. Простим числом прийнято вважати натуральне число, яке має тільки два натуральних дільника: одиницю та самого себе. Всі решта натуральні числа, крім одиниці називаються складовими. Зокрема послідовність простих чисел починається як: *2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, 97, 101, 103, 107, 109, 113, 127, 131, 137, 139, 149, 151, 157……..*

Виберемо наприклад P=7 та Q=13.

2. Здійснюється обчислення добутку N=P*Q, в нашому прикладі:

$\boldsymbol{n} = 7 \cdot 13 = 91.$

3. Далі обчислюється функція Ейлера $\boldsymbol{\varphi(n)}$ яка визначає кількість натуральних чисел, що є , меншими за n и та є взаємно простими з ним:

$$\boldsymbol{\varphi(n)} = (\boldsymbol{p} - 1) \cdot (\boldsymbol{q} - 1).$$

В нашому прикладі: $\boldsymbol{\varphi(n)} = (7 - 1) \cdot (13 - 1) = 72$.

4. Здійснюється вибір довільного цілого числа $\boldsymbol{e}$ з врахуванням виконання умови: $0 < \boldsymbol{e} < \boldsymbol{n}$ та взаємно простого із значенням функції Ейлера. В нашому прикладі візьмемо $\boldsymbol{e} = 5$. При цьому, пара чисел $(\boldsymbol{e}, \boldsymbol{n})$ оголошується відкритим ключем шифру. В нашому прикладі $(\boldsymbol{e}, \boldsymbol{n}) = (5, 91)$.

5. Обчислюємо ціле число $\boldsymbol{d}$ яке буде секретним ключем, виходячи з відношення:

$$(\boldsymbol{d} \cdot \boldsymbol{e}) \bmod \boldsymbol{\varphi(n)} = 1$$

Це відношення означає, що результатом ділення добутку чисел $\boldsymbol{e}$ та $\boldsymbol{d}$ на

значення функції Ейлера повинно бути число 1. З огляду на те, d можна розрахувати за формулою:

$$d = \frac{k \cdot \varphi(n) + 1}{e},$$

le змінна k набуває послідовно значення 1, 2, 3,.. до тих пір, поки не буде отримано ціле число d. Знайдемо d в нашому прикладі:

$$d = \frac{k \cdot 72 + 1}{5},$$

при $k = 1$, d – не ціле, при $k = 2$, $d = 29$. З огляду на те, пара чисел (d, n) буде нашим закритим ключем шифру : $(d, n) = (29, 91)$.

RSA-шифрування повідомлення T здійснюється з допомогою відкритого ключа (e, n) за формулою:

$$C_i = T_i^e \bmod n,$$

де T_i та C_i числові еквіваленти символів початкового та зашифрованого повідомлення. RSA-дешифрування зашифрованого повідомлення C виконується з допомогою закритого ключа (d, n) за формулою:

$$T_i = C_i^d \bmod n.$$

В якості прикладу початкового повідомлення використаємо фразу «ДВГУПС» та зашифруємо її з допомогою відкритого ключа (5, 91) (рис. 1). Коди символів візьмемо з табл.1.

Таблиця 1. Алфавіт «Українські букви та цифри»

Буква	А	Б	В	Г	Д	Е	Ї	Ж	З	И	Й	К
Код	01	02	03	04	05	06	07	08	09	10	11	12
Буква	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц
Код	13	14	15	16	17	18	19	20	21	22	23	24
Буква	Ч	Ш	Щ	Є	Ґ	Ь	І	Ю	Я	Пробіл		
Код	25	26	27	28	29	30	31	32	33	34		
Цифра	0	1	2	3	4	5	6	7	8	9		
Код	35	36	37	38	39	40	41	42	43	44		

Символи початкового повідомлення , T_i	Коди символів T_i, (табл. 1)	Зашифровані коди символів, C_i
Д	5	$5^5 \bmod 91 = 31$
В	3	$3^5 \bmod 91 = 61$
Г	4	$4^5 \bmod 91 = 23$
У	21	$21^5 \bmod 91 = 21$
П	17	$17^5 \bmod 91 = 75$
С	19	$19^5 \bmod 91 = 80$

Рис. 1. Шифрування RSA

В результаті отримаємо шифрограму «31, 61, 23, 21, 75, 80».

Дешифрування шифрограми «31, 61, 23, 21, 75, 80» здійснюється закритим ключем (29, 91), рис. 2.

Коди шифрограми, c_i	Дешифровані коди символів T_i	Початковий текст T_i
31	$31^{29} \bmod 91 = 5$	Д
61	$61^{29} \bmod 91 = 3$	В
23	$23^{29} \bmod 91 = 4$	Г
21	$21^{29} \bmod 91 = 21$	У
75	$75^{29} \bmod 91 = 17$	П
80	$80^{29} \bmod 91 = 19$	С

Рис. 2. Дешифрування RSA

Модульна арифметика (mod – арифметика)

Модульна арифметика – це система арифметики цілих чисел, пов'язана з залишками від ділення цілих чисел на певне задане натуральне число. Фактично в ній розглядаються класи еквівалентності певного натурального числа. При цьому mod визначає цілий залишок від ділення двох цілих додатних чисел.

Пояснення:

R1=12 (mod 10) = 12-10*mod(12/10)=12-10*1=2

Поняття a mod (-N) не існує а обрахувати від'ємне ціле число за модулем N можна так: – 9 mod 4 =- (9 mod 4)= –1 + 4 = 3

Можна скористатися також функцією Int(x) – ціле число

Для a > 0 r = a mod N = a –N * Int(a/N)

Для a < 0 r = a mod N = a + N * (Int(-a/N)+1)

Наприклад:

r = 9 mod 4 = 9 –4 * Int(9/4) = 9 –4*2 = 9 – 8 = 1

r = –9 mod 4 = –9 + 4 * (Int(+9/4) = — 9 + 4*(2+1) = 3

В теорії чисел визначено відношення ($\equiv$) зрівняності цілих чисел: a $\equiv$ b (mod N), яке визначає, що 'a' зрівняне з 'b' по модулю N, 'a' та 'b' – цілі, N≠0, якщо виконується рівність: a = b + k*N. Або іншими словами: N ділить (a –b): N| (a-b) та 'b' називають вирахуванням числа 'a' по модулю N.

Вираз a $\equiv$ b (mod N)є рівносильний твердженню про те, що залишки від ділення a' і 'b'на N рівні. Приклад, 17 $\equiv$ 5 (mod 12) означає, що 17 mod 12 = 5 та 5 mod 12 = 5.

Однією з властивостей модульної арифметики якою будемо

користуватись є:

$$x^m \bmod N = x^{a+b+c} \bmod N = (x^a x^b x^c) \bmod N = [(x^a \bmod N)*(x^b \bmod N)*(x^c \bmod N)] \bmod N$$

Її зручно використовувати у випадку коли необхідно обраховувати великі числа, які значно більші за розрядність комп'ютера.

Наприклад : $x^{53} \bmod N$

Розкладемо 53 на двійкові співмножними 1, 2, 4, 8, 32, 64, … .

$x^{53} = x^{(32+16+4+1)}$

Необхідно спочатку знайти x^2, $x^4 = (x^2)^2$, $x^8 = (x^4)^2$, $x^{16} = (x^8)^2$, $x^{32} = (x^{16})^2$.

Тепер потрібно послідовно помножити $x^{32} * x^{16} * x^{4*} x$. Всі операції необхідно виконувати за модулем N. В результаті отримаємо результат за 8 операцій.

Приклад:

$3^{19} \bmod 15 = 3^{19} - 15 \cdot \mathrm{Int}(3^{19}/15) = 1162261467 - 15 \cdot 77484097 = 12$

$19 = 16 + 2 + 1$

1	3	3	
2	$3^2 = 9$	9	$3 \cdot 9 = 27 \bmod 15 = 12$
4	$9^2 = 81 \bmod 15 = 6$		
8	$6^2 = 36 \bmod 15 = 6$		
16	$6^2 = 36 \bmod 15 = 6$	6	$12 \cdot 6 = 72 \bmod 15 = 12$

Електронний цифровий підпис

Технологія застосування системи електронного цифрового підпису (ЕЦП) припускає наявність мережі абонентів, які обмінюються підписаними електронними документами. При обміні електронними документами по мережі значно знижуються витрати, пов'язані з їх обробкою, зберіганням і пошуком. Одночасно при цьому виникає проблема, як аутентифікації автора електронного документа, так і самого документа, тобто встановлення справжності автора й відсутності змін в отриманому повідомленні.

В алгоритмах ЕЦП як і в асиметричних системах шифрування використовуються однонаправлені функції. Концепція формування цифрового підпису основана на оборотності асиметричних шифрів, а також на взаємопов'язаності вмісту повідомлення, підпису і пари ключів. Зміна хоча б одного з названих елементів зробить неможливим підтвердження справжності підпису, у випадку її реалізації за допомогою асиметричних алгоритмів шифрування та хеш-функцій.

Система ЕЦП включає дві процедури: формування цифрового підпису та його перевірки. У процедурі формування підпису використовується секретний ключ відправника повідомлення, в процедурі перевірки підпису - відкритий ключ відправника.

При цьому електронний цифровий підпис надійніше вирішує не тільки традиційні завдання авторства і достовірності документа, які раніше забезпечувалися рукописним підписом під паперовим документом, але й наступні важливі завдання електронного документообігу: цілісність документа; неможливість підробки підпису; запобігання відмови від підпису; юридичну значимість документа.

Обчислення ЕЦП

Алгоритм цифрового підпису розпочинає свою роботу з попереднього хешування повідомлення – обчислюється значення деякої контрольної функції від всього повідомлення. Для обчислення хеш-образу H повідомлення T в даній лабораторній роботі пропонується використовувати спрощену хеш-функцію квадратичної згортки:

$$H_i = \left(H_{i-1} + M_i\right)^2 mod\ n,$$

де n визначається з відкритого ключа автора повідомлення, M_i – коди символів повідомлення, відкритого чи попередньо зашифрованого. Після опрацювання останнього символу отримуємо хеш-образ всього повідомлення H.

В алгоритмах ЕЦП призначення відкритого та закритого ключів змінюється, а саме – повідомлення підписується закритим ключем відправника, після чого будь-хто може перевірити достовірність з допомогою відкритого ключа. Обчислення ЕЦП S здійснюється за хеш образом H повідомлення, що пересилається T з допомогою закритого ключа (d, n) автора повідомлення за формулою:

$$S = H^d \ mod\ n$$

Формування повідомлення підписаного електронним цифровим підписом для передачі здійснюється приєднанням останнього (ЕЦП) S до повідомлення M: SM. Наприклад, поставимо електронний цифровий підпис на повідомлення «**МАША**».

У відповідності з алгоритмом формування ЕЦП RSA вибираємо два взаємно простих числа p і q **та** обчислюємо значення $n=p*q=91$, вибираємо значення секретного ключа $d=29$ та обчислюємо значення відкритого ключа $e= 5$.Вектор ініціалізації H_0 приймем рівним нулю (вибирається довільним чином). Закритий ключ відправника $(d, n) = (29, 91)$. Обрахуємо хеш-образ

повідомлення (рис. 3) застосовуючи таблицю 1.

i	Символи тексту M_i	Коди символів M_i	Обчислення хеш-образу H
$H_0 = 0$			
1	М	14	$H_1 = (H_0 + M_1)^2 \ mod \ n = (0+14)^2 \ mod \ 91 = 14$
2	А	1	$H_2 = (H_1 + M_2)^2 \ mod \ n = (14+1)^2 \ mod \ 91 = 43$
3	Ш	26	$H_3 = (H_2 + M_3)^2 \ mod \ n = (43+26)^2 \ mod \ 91 = 29$
4	А	1	$H_4 = (H_3 + M_4)^2 \ mod \ n = (29+1)^2 \ mod \ 91 = 81$
Хеш-образ			$H = 81$
Цифровий підпис			$S = H^d \ mod \ n = 81^{29} \ mod \ 91 = 9$

Рис. 3. Обчислення ЕЦП

Хеш-образом H повідомлення «**МАШІА**» є число 81. Обчислення ЕЦП S за хеш-образом з допомогою закритого ключа здійснюється за другою формулою. Електронним цифровим підписом повідомлення є число **9**. Сформуємо повідомлення для передачі додавши до нього ЕЦП, отримаємо повідомлення «**9МАШІА**».

Перевірка справжності ЕЦП

Здійснюється за такими етапами:

1. Відділення ЕЦП від основного повідомлення

2. Виділення з ЕЦП хеш образу отриманого повідомлення відкритим ключем за формулою:

$$H' = S^e \ mod \ n.$$

3. Обчислення хеш-образу H'' отриманого повідомлення на стороні отримувача за першою формулою.

4. Порівняння H' та H''. ЕЦП визнається достовірний якщо значення хеш образів співпадають:

$$H' = H''.$$

Наприклад, допустимо, що при передачі повідомлення його вид «**9МАШІА**» змінився на «**9МИШІА**». Перевіримо достовірність отриманого повідомлення (рис. 4).

Хеш-образ з ЕЦП			$H' = S^e \bmod n = 9^5 \bmod 91 = 81$
i	Символи прийнятого тексту M_i	Коди символів M_i	Обчислення хеш-образу на стороні отримувача H''
			$H_0 = 0$
1	М	14	$H_1 = (H_0 + M_1)^2 \bmod n = (0+14)^2 \bmod 91 = 14$
2	И	10	$H_2 = (H_1 + M_2)^2 \bmod n = (14+10)^2 \bmod 91 = 30$
3	Ш	26	$H_3 = (H_2 + M_3)^2 \bmod n = (30+26)^2 \bmod 91 = 42$
4	А	1	$H_4 = (H_3 + M_4)^2 \bmod n = (42+1)^2 \bmod 91 = 29$
Обчислений хеш-образ			$H'' = 29$

Рис. 4. Перевірка легітимності ЕЦП

На рис. 4 показано, що значення хеш образу функції повідомлення на стороні відправника рівне $H' = 81$ а отриманого повідомлення $H'' = 29$. З нерівності $H' \neq H''$ можна зробити висновок, що під час передачі повідомлення пройшло його випадкове чи спеціальне спотворення. З рис. 3 та 4 видно, що навіть при зміні однієї букви в початковому повідомленні його хеш образ різко змінюється. Зазначені зміни легко виявляються в процесі перевірки достовірності ЕЦП.

КОНТРОЛЬНІ ПИТАННЯ

1. Поняття асиметричного шифрування даних.
2. Особливості створення відкритого за закритого ключа.
3. Шифрування та дешифрування даних згідно з методом RSA.
4. Поняття електронного цифрового підпису та способи його завдання?
5. Обчислення хеш образу повідомлення?
6. Перевірка легітимності електронного цифрового підпису?

Порядок виконання роботи

1. Написати програму на мові С++ (чи іншій за згодою викладача) яка виконує операції:

1.1. Створення відкритого та закритого ключа при заданих для конкретного варіанту значеннях p та q (табл. 2). В якості числа e, яке

входить в склад відкритого ключа, необхідно взяти найбільше просте число, менше $\boldsymbol{p}$.

1.2. Використовуючи алфавіт (табл.1.), зашифруйте повідомлення з допомогою створеного відкритого ключа. Повідомлення вибирається згідно з варіантом з табл..2.

Таблиця 2

Варіант	До завдання 1.1.		До завдання 1.2.
	$\boldsymbol{p}$	$\boldsymbol{q}$	
1	19	73	ВТБ17П
2	29	73	ДКБ18П
3	17	29	СБР23П
4	23	61	РКЦ24П
5	13	31	ПАЙ_50
6	23	31	СЕЙФ02
7	53	73	МІН_10
8	31	37	РАНГ01
9	17	37	РЕЙС56
10	23	79	КУПЕ14
11	13	41	СБОР21
12	23	41	ПЕНЯ15
13	17	41	ГОСТ24
14	71	79	ВОХР05
15	19	43	МАКС90
16	13	61	ПЛЮС57
17	41	79	КУРС04
18	13	53	ЩКОДБ8
19	59	61	РАУТ03
20	13	83	ТІНДА2
21	13	19	ДОП_20
22	19	29	СКЛАД7
23	17	67	АГЕНТ8

24	13	17	БАГАЛ4
25	31	73	МІНУС5
26	19	73	ВТБ17П
27	29	73	ДКБ18П
28	17	29	СБР23П
29	23	61	РКЦ24П
30	13	31	ПАЙ_50

1.3. Дешифрування зашифрованого тексту з допомогою створеного в 1.1. закритого ключа. Шифрограма вибирається згідно з варіантом з табл. 3.

Таблиця 3

Варіант	Криптограма
1	377, 754, 1310, 1, 1233, 35, 1045
2	352, 254, 1621, 1868, 533, 1772, 1608
3	1, 172, 225, 32, 335, 443, 469, 379
4	623, 18, 1215, 168, 623, 44, 1071
5	40, 336, 307, 207, 291, 307, 329, 396
6	468, 409, 568, 1, 286, 82, 297, 40
7	2965, 725, 2021, 565, 3300, 1939, 2957, 3810
8	547, 894, 725, 247, 1081, 803, 219, 205
9	532, 305, 574, 232, 15, 309, 265, 427
10	1348, 1029, 591, 1047, 1491, 24, 1677, 705
11	191, 139, 117, 270, 458, 346, 1, 79
12	258, 186, 43, 408, 633, 686, 734, 682
13	310, 256, 385, 580, 218, 1, 256, 326
14	5482, 2017, 5226, 1, 2949, 1584, 3032
15	537, 681, 721, 167, 1, 470, 439, 729
16	736, 349, 259, 1, 466, 691, 437, 628
17	558, 2355, 1046, 2538, 1, 2561, 648
18	100, 670, 501, 661, 532, 171, 680, 74
19	1640, 1705, 2497, 718, 114, 3256, 1640, 114
20	271, 266, 462, 170, 490, 462
21	29, 9, 151, 76, 222, 1, 31, 9, 151

22	337, 1, 48, 286, 363, 22, 48, 304
23	408, 843, 1123, 555, 151, 99, 322, 408
24	27, 152, 208, 141, 142, 200, 208, 1
25	1862, 827, 1483, 2158, 180, 2158, 1, 489, 1347
26	377, 754, 1310, 1, 1233, 35, 1045
27	352, 254, 1621, 1868, 533, 1772, 1608
28	1, 172, 225, 32, 335, 443, 469, 379
29	623, 18, 1215, 168, 623, 44, 1071
30	40, 336, 307, 207, 291, 307, 329, 396

2. Оформити алгоритм функціонування програми та здійснити його опис
3. Результати подати у вигляді текстів програми та скрін шотів.
4. Додаткове завдання. Написати програму яка виконує операції:

1.1. Створення хеш-образу повідомлення, варіанти в табл. 1, та обчисліть його ЕЦП.

1.2. Перевірте автентичність відправленого повідомлення та ЕЦП цього повідомлення на стороні отримувача, варіанти повідомлень в табл.4.

1.3. Сформулюйте висновки щодо використання ЕЦП

Таблиця 4.

Варіант	p	q	Повідомлення до завдання 1.1.	До завдання 1.2.	
				Повідомлення	ЕЦП в тому числі
1	19	73	ОФ5	444ЗАО4	444
2	29	73	Р27	1222ДВ30	1222
3	17	29	ІП6	432РК20	432
4	23	61	Т34	52ІЛ62	52
5	13	31	К35	205РЯД9	205
6	23	31	Д65	407МИР4	407
7	53	73	КВ7	1826ЄЄС5	1826
8	31	37	Н18	926ДОМ2	926
9	17	37	КР4	576СУ24	576
10	23	79	КГ8	936ДПИ4	936
11	13	41	Б52	113ЗСК4	113

12	23	41	КП6	920ВИП3	920
13	17	41	Н2О	228МІН7	228
14	71	79	СО2	2205ЧОЛ9	2205
15	19	43	ЧП3	594ЦДО6	594
16	13	61	Ч21	781Е2Е4	781
17	41	79	М43	1472СЄВ9	1472
18	13	53	СМ5	179ДТВ3	179
19	59	61	Н32	1650ПК31	1650
20	13	83	Г05	691СКА2	691
21	13	19	Т50	131КВ49	131
22	19	29	МІ6	361ОХА2	361
23	17	67	КМ7	174ОФ17	174
24	13	17	ПО6	179КМ65	179
25	31	73	ПК6	690КГ75	690
26	19	73	ОФ5	4443АО4	444
27	29	73	Р27	1222ДВ30	1222
28	17	29	ІП6	432РК20	432
29	23	61	Т34	52ІЛ62	52
30	13	31	К35	205РЯД9	205

Лабораторна робота №4

Тема роботи: Шифрування даних методом стеганографії.

Мета роботи: Навчитися опрацьовувати (шифрувати та дешифрувати) файли з прихованими інформаційними повідомленнями.

Теоретичні відомості

Як інформацію, що підлягає шифруванню і дешифруванню, розглядають тексти, побудовані на деякій абетці (алфавіті).

- **абетка** - скінчена множина використовуваних для кодування інформації знаків;

- **текст** - упорядкований набір з елементів абетки;
- **шифрування** - процес перетворення вихідного тексту, що має також назву відкритого тексту, на шифрований текст на основі ключа;
- **дешифрування** - зворотний шифруванню процес, що на основі ключа шифрований текст перетворює у вихідний;
- **ключ** - інформація, яка необхідна для безперешкодного шифрування і дешифрування.

Для приховання точного змісту інформаційного повідомлення можуть застосовуватися різні прийоми, суть яких зводиться до того, що у відповідність одній послідовності знаків чи слів однієї мови ставляться знаки чи слова іншої. Як приклади можна навести так званий шифр "Аве Марія", у кодовому варіанті якого кожному слову, а часом і фразі ставляться у відповідність кілька слів явної релігійної тематики, у результаті чого повідомлення набуває вигляду специфічного тексту духовного змісту.

У загальному випадку способи приховування або самого факту наявності повідомлення, або його точного змісту називаються стеганографією. Слово "стеганографія" у перекладі з грецької буквально означає "тайнопис". До неї належить величезна кількість секретних засобів зв'язку, таких як невидиме чорнило, мікрофотознімки, умовне розташування знаків (застосовуване в сигнальному агентурному зв'язку), цифрові підписи, таємні канали і засоби зв'язку на плаваючих частотах тощо.

Стеганографія займає свою нішу в забезпеченні безпеки інформації: вона не заміняє, а доповнює криптографію, хоча криптографія, як окремий напрямок з'явилася пізніше. За наявності шифрованого повідомлення, тобто при застосуванні криптографічних методів захисту, супротивнику хоча і невідомий зміст повідомлення, але відомий факт наявності такого повідомлення. При використанні ж стеганографічних методів супротивнику невідомо, чи є отриманий зміст повідомлення остаточним, чи за ним приховано додатковий зміст.

Стосовно комп'ютерних технологій доцільно зазначити, що стеганографія використовує методи розміщення файлу - "повідомлення" у файлі - "контейнері", змінюючи файл - "контейнер" таким чином, щоб зроблені зміни були практично непомітні.

Основні принципи комп'ютерної стеганографії

К. Шеннон запропонував загальну теорію тайнопису, що є базисом стеганографії як науки. У сучасній комп'ютерній стеганографії існує два основних типи файлів: повідомлення - файл, що призначений для приховання, і контейнер-файл, що може бути використаний для приховання в ньому

повідомлення. При цьому контейнери бувають двох типів. Контейнер-оригінал (або “Порожній” контейнер) - це контейнер, що не містить схованої інформації. Контейнер-результат (або “Заповнений” контейнер) - це контейнер, що містить сховану інформацію. Під ключем розуміється секретний елемент, що визначає порядок занесення повідомлення в контейнер.

Основними положеннями сучасної комп'ютерної стеганографії є:

1. Методи приховання повинні забезпечувати автентичність і цілісність файлу.

2. Безпека методів ґрунтується на збереженні стеганографічним перетворенням основних властивостей відкрито переданого файлу при внесенні в нього секретного повідомлення й деякої невідомої супротивникові інформації — ключа.

3. Навіть якщо факт прихування повідомлення став відомий видобування повідомлення представляє собою складне обчислювальне завдання.

У зв'язку зі зростанням ролі глобальних комп'ютерних мереж стає все більше важливим значення стеганографії. Аналіз інформаційних джерел комп'ютерної мережі Internet дозволяє зробити висновок, що в наш час стеганографічні системи можуть активно використовуватись для рішення наступних основних завдань:

1. Захист конфіденційної інформації від несанкціонованого доступу. Це область використання є найбільш ефективною при рішенні проблеми захисту конфіденційної інформації. Так, наприклад, тільки одна секунда оцифрованого звуку із частотою дискретизації 44100 Гц і рівнем відліку 8 біт у стерео режимі дозволяє приховати за рахунок заміни найменш значимих молодших розрядів близько 100 Кбайт інформації. При цьому, зміна значень становить менш 1 %. Така зміна не виявляється при прослуховуванні файлу більшістю людей.

2. Подолання систем моніторингу й керування мережними ресурсами. Стеганографічні методи, спрямовані на протидію системам моніторингу й керування мережними ресурсами промислового шпигунства, дозволяють протистояти спробам контролю над інформаційним простором при проходженні інформації через сервери керування локальних і глобальних обчислювальних мереж.

3. Захист авторського права на деякі види інтелектуальної власності. Ще однією областю використання стеганографії є захист авторського права від піратства. На комп'ютерні графічні зображення наноситься спеціальна мітка, що залишається невидимою для людини, але розпізнається

спеціальним програмним забезпеченням. Таке програмне забезпечення вже використовується зокрема в технології HP Mark у BLu-ray дисках.

Способи застосування комп'ютерної стеганографії

На сьогоднішній день наявне широка множина методів та засобів в яких можна використовувати стеганографію. Наведемо деякі приклади:

- Використання зарезервованих полів комп'ютерних форматів файлів – суть методу полягає в тому, що частина поля розширень, що не заповнюється інформацією про формат файлу заповнюється нулями. Відповідно ми можемо використати цю «нульову» частину для запису своїх даних.
- Метод приховання інформації в невикористовуваних місцях оптичних дисків - при використанні цього методу інформація записується в невикористовувані частини диска, приміром, на нульову доріжку..
- Метод використання особливих властивостей полів форматів, які не відображаються на екрані - цей метод заснований на спеціальних «невидимих» полях, що використовуються для зберігання виносок, покажчиків. Приміром, написання чорним шрифтом на чорному тлі.
- Використання особливостей файлових систем - при зберіганні на жорсткому диску файл завжди займає ціле число кластерів. Як приклад у файловій системі FAT32 стандартний розмір кластера - 4Кб. Відповідно для зберігання 1Кб інформації на диску виділяється 4Кб інформації, з яких 1Кб потрібний для зберігання файлу, а інші 3 Кб можна використати для зберігання інформації.

По суті комп'ютерна стеганографія базується на двох принципах. *Перший* полягає в тому, що файли, які в першу чергу утримують оцифроване зображення або звук, можуть бути певною мірою видозмінені без втрати функціональності, на відміну від інших типів даних, що вимагають абсолютної точності. *Другий принцип* полягає в нездатності органів відчуття людини розрізнити незначні зміни в кольорі зображення або якості звуку, що особливо легко використати стосовно об'єкта, що несе надлишкову інформацію, будь то 16-бітний звук або 24-бітне зображення.

Для цілей стеганографії за звичай використовується 24 - бітний BMP формат (на піксель виділяється три байти). Корисна (передана) інформація записується як молодший біт кожного з кольорів (RGB). Створені зміни не вловимі для людського ока. Нехай маємо число 180, у двійковому коді воно виглядає так: 10110100. Давайте приховаємо його в послідовності з восьми байт, що наведена у першій колонці таблиці. Для цього замінимо у двійкове представлення чисел послідовності (друга колонка) молодші біти

(підкреслені) бітами нашого числа. Одержимо третю колонку таблиці, десяткове подання чисел якої запишемо в четвертій колонці.

Таблиця 1. Результати стеганографії

Вихідні значення (десяткові)	Двійкове представлення	Послідовність після заміни	Десяткові значення після заміни
135	10000111	10000111	135
121	01111001	01111000	120
120	01111000	01111001	121
107	01101011	01101011	107
143	10001111	10001110	142
98	01100010	01100011	99
103	01100111	01100110	102
102	01100110	01100110	102

Щільність упакування 1:8, тобто для приховання даного файлу необхідний контейнер, що має обсяг в 8 разів більше. В якості контейнера доцільно використати звукові файли або зображення яскравих кольорів без чітких геометричних фігур

Алгоритми стеганографії

Всі алгоритми приховання інформації можна розділити на кілька підгруп:

- Працюють із самим цифровим сигналом. Наприклад, метод LSB.
- «Вбудовування» прихованої інформації. У цьому випадку відбувається накладення приховуваного зображення (звуку чи тексту) поверх оригіналу.
- Використання особливостей форматів файлів. Сюди можна віднести запис інформації в метадані або зарезервовані поля файлу.

По способі вбудовування інформації стегоалгоритми можна розділити на лінійні (адитивні) та нелінійні. Алгоритми адитивного приховання інформації полягають у лінійній модифікації вихідного зображення, а її добування в декодері виробляється кореляційними методами. При цьому приховувана інформація за звичай «вплавляється» в зображення-контейнер. У нелінійних методах вбудовування інформації здійснюється шляхом скалярного або векторного квантування.

Метод LSB (Least Significant Bit, найменший значущий біт) – суть цього методу полягає в заміні останніх значущих бітів у контейнері (зображення,

аудіо або відеозапису) на біти приховуваного повідомлення. Різниця між порожнім і заповненим контейнерами повинна бути не відчутна для органів сприйняття людини. Суть методу полягає в наступному: Допустимо, є 8-бітне зображення в градаціях сірого. 00h (00000000b) позначає чорні кольори, FFh (11111111b) — білий. Всього є 256 градацій (2^8).

Також припустимо, що повідомлення складається з 1 байта — наприклад, 01101011b. При використанні 2 молодших біт в описах пікселів, нам буде потрібно 4 пікселя. Допустимо, вони чорних кольорів. Тоді пікселі, що містять сховане повідомлення, будуть виглядати в такий спосіб: 00000001 00000010 00000010 00000011. Тоді кольори пікселів зміниться: першого - на 1/255, другого й третього - на 2/255 і четвертого - на 3/255. Такі градації, мало того що непомітно для людини, можуть взагалі не відобразитися при використанні низькоякісних пристроїв виведення.

Інші методи приховання інформації в графічних файлах орієнтовані на формати фалів із втратою, приміром, JPEG. На відміну від LSB вони більше стійкі до геометричних перетворень. Це виходить за рахунок варіювання в широкому діапазоні якості зображення, що приводить до неможливості визначення джерела зображення.

Ехо-методи застосовуються в цифровий аудіостеганографії й використовують нерівномірні проміжки між ехо-сигналами для кодування послідовності значень. При накладенні ряду обмежень задовольняється умова непомітності для людського сприйняття. Ехо характеризується трьома параметрами: початковою амплітудою, ступенем загасання, затримкою. При досягненні якогось порогу між сигналом та ехом вони змішуються. У цій точці людське вухо не може вже відрізнити ці два сигнали. Найчастіше для цих цілей використовується затримка близько 1/1000, що є цілком прийнятною для більшості записів та слухачів. Для позначення логічного нуля й одиниці використовуються дві різні затримки. Вони обидві повинні бути меншими, ніж поріг чутливості вуха слухача до одержуваної луни.

Фазове кодування (phase coding, фазове кодування) – застосовується в цифровій аудіостеганографії. Відбувається заміна вихідного звукового елемента на відносну фазу, яка і є секретним повідомленням. Фаза елементів, що йдуть підряд, повинна бути додана таким чином, щоб зберегти відносну фазу між вихідними елементами. Фазове кодування є одним з найефективніших методів приховання інформації.

КОНТРОЛЬНІ ПИТАННЯ

1. Що таке стеганографія і які її базові принципи?
2. Поняття контейнера, їх види?
3. Опишіть відомі алгоритми стеганографії?
4. Перспективи розвитку стеганографії?

Порядок виконання роботи

1. Написати програму на мові С++ (чи іншій за згодою викладача) яка виконує криптографічні перетворення (шифрування та дешифрування) над файлами за **стеганограмним** способом (файл повідомлення, файл - "контейнер" і файл – результат) відповідно до заданого варіанту, таблиця 1. Провести порівняння розподілу ймовірностей наявних символів файлу до та після шифрування (приховування). На вибір вибрати розподіл по 10-15 символах.

№	Завдання
1	Розміщення однобітних елементів повідомлення в файл текстового формату
2	Реалізація фазового кодування в процесі приховання інформації в звуковій доріжці
3	Розміщення однобітних елементів повідомлення в файл графічного формату 16 кольорів від початку файлу
4	Розміщення двобітних елементів повідомлення в файл графічного формату 256 кольорів від кінця файлу
5	Розміщення чотирибітних елементів повідомлення в файл графічного формату 256 кольорів від заданої позиції в файлі
6	Розміщення чотирибітних елементів повідомлення в файл графічного формату 24-біти на колір
7	Розміщення однобайтних елементів повідомлення в файл графічного формату 24-біти на колір
8	Розміщення чотирибітних елементів повідомлення в файл графічного формату 32-біти на колір
9	Розміщення однобайтних елементів повідомлення в файл графічного формату 32-біти на колір
10	Розміщення чотирибітних елементів повідомлення в файл графічного формату 256 кольорів від заданої позиції в файлі
11	Розміщення однобітних елементів повідомлення в текстовому файлі починаючи із кінця
12	Розміщення однобітних елементів повідомлення в rtf файлі

13	Розміщення двобітних елементів повідомлення в файл графічного формату 256 кольорів від кінця файлу
14	Реалізація фазового кодування в процесі приховання інформації в звуковій доріжці
15	Розміщення однобайтних елементів повідомлення в файл графічного формату 24-біти на колір
16	Розміщення однобітних елементів повідомлення в файл текстового формату
17	Реалізація фазового кодування в процесі приховання інформації в звуковій доріжці
18	Розміщення однобітних елементів повідомлення в файл графічного формату 16 кольорів від початку файлу
19	Розміщення двобітних елементів повідомлення в файл графічного формату 256 кольорів від кінця файлу
20	Розміщення чотирибітних елементів повідомлення в файл графічного формату 256 кольорів від заданої позиції в файлі
21	Розміщення чотирибітних елементів повідомлення в файл графічного формату 24-біти на колір
22	Розміщення однобайтних елементів повідомлення в файл графічного формату 24-біти на колір
23	Розміщення чотирибітних елементів повідомлення в файл графічного формату 32-біти на колір
24	Розміщення однобайтних елементів повідомлення в файл графічного формату 32-біти на колір
25	Розміщення чотирибітних елементів повідомлення в файл графічного формату 256 кольорів від заданої позиції в файлі
26	Розміщення однобітних елементів повідомлення в текстовому файлі починаючи із кінця
27	Розміщення однобітних елементів повідомлення в rtf файлі
28	Розміщення двобітних елементів повідомлення в файл графічного формату 256 кольорів від кінця файлу
29	Реалізація фазового кодування в процесі приховання інформації в звуковій доріжці
30	Розміщення однобайтних елементів повідомлення в файл графічного формату 24-біти на колір

2. Здійснити компіляцію програми.

3. Оформити звіт в згідно вимог, результати подати у вигляді таблиць – парні варіанти, графіків – непарні.

Лабораторна робота №5

Тема роботи: Захист даних з допомогою мереж Фейстеля..

Мета роботи: Вивчити принципи роботи мережі Фейстеля, навчитися шифрувати інформацію за допомогою використання блокового криптоалгоритму.

Теоретичні відомості

В 1973 році Хорст Фейстель в журналі Scientific American опублікував статтю «Криптографія і комп'ютерна безпека»(«Cryptography and Computer Privacy»), в якій розкрив деякі важливі аспекти шифрування, а також ввів конструкцію, названу пізніше мережею Фейстеля. Ця схема була використана в проекті Lucifer фірми IBM, над яким працював Фейстель і Дон Коперсміт (Don Coppersmith). Цей проект був скоріше експериментальним, але став базисом для стандарту шифрування DES. Ітеративна структура алгоритму дозволила спростити його реалізацію в апаратному середовищі.

Мережа Фейстеля отримала широке поширення, внаслідок забезпечення вимоги щодо багаторазового використання ключа і матеріалу початкового блоку інформації. Класична мережа Фейстеля має структуру приведену на рис.1.

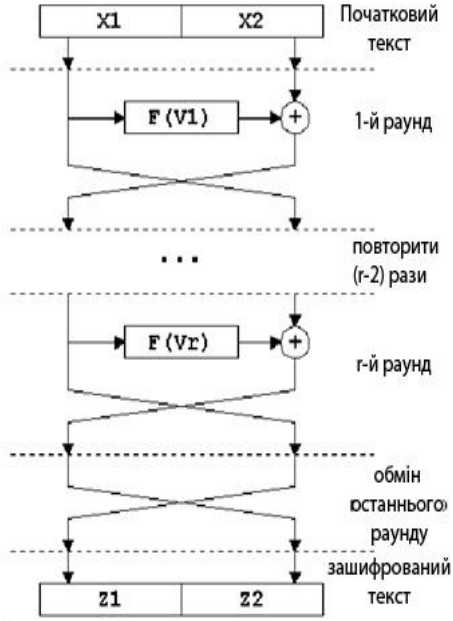

Рис.1. Класична структура мережі Фейстеля

Незалежні потоки інформації, які породжені з початкового блоку, носять назву гілки мережі. У класичній схемі їх дві. Величини Vi носять назву параметри мережі. Функція F називається утворюючою. При цьому, під раундом (циклом) мережі Фейстеля розуміють дію, яка полягає в одноразовому обчисленні утворюючої функції з подальшим накладенням її значення на іншу вітку з обміном їх місцями. Оптимальна кількість раундів K - від 8 до 32. Часто, кількість раундів не фіксується розробниками алгоритму, а лише вказуються допустимі межі даного параметру.

Дана схема є оборотною. Мережа Фейстеля володіє такою властивістю, що навіть якщо в якості утворюючої функції F буде використано необоротне перетворення, то і в цьому випадку можна відновити всі кроки перетворень. Це відбувається внаслідок того, що для зворотного перетворення мережі Фейстеля не потрібно обчислювати функцію F^{-1}.

Мережа Фейстеля є симетричною за рахунок використання операції XOR і для її оборотності не має значення парність чи непарність кількості раундів. Використання модифікації мережі Фейстеля для більшої кількості гілок пов'язано з тим, що при великих розмірах кодованих блоків (128 і більше біт) стає незручно працювати з математичними функціями по модулю 64 і вище. Основні одиниці інформації, що опрацьовуються більшістю процесорів - це байт і подвійне машинне слово 32 біта. З огляду на те, логічно розбивати вихідні блоки не на дві, а на 4 частини. У цьому випадку мережа Фейстеля може приймати такий вид:

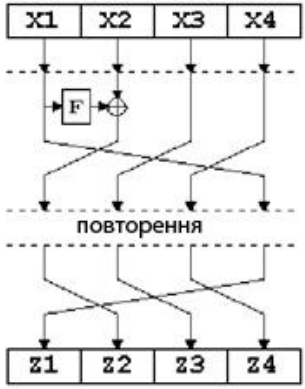

Рис.2. Структура модифікованої мережі Фейстеля

Алгоритм призначений для шифрування і дешифрування інформації, представленої у вигляді слів, розрядністю 128 біт на основі 64-бітового ключа. Операції шифрування і дешифрування є інверсними і використовують один і той же ключ. Розглянемо шифрування одного блоку.

Позначимо X1X2X3X4 конкатенацію (об‘єднання) послідовностей X1. X2, X3 і X4, в якій біти послідовностей X1, X2, X3, X4 йдуть один за одним. Розмірність послідовності дорівнює сумі розмірностей всіх складових. Символом + позначимо операцію побітового додавання за модулем 2.

Ітеративний процес шифрування описується наступними формулами:

X1(i) = X2(i-1)+F(Vi), i = 1, 2, ... ,n;

X2(i) = X3(i-1), i = 1, 2, ... ,n;

X3(i) = X4(i-1), i = 1, 2, ... ,n;

X4(i) = X1(i-1), i = 1, 2, ... ,n;

де F(Vi) – перетворююча функція

n – кількість раундів, може змінюватися в залежності від вимог щодо швидкодії та криптостійкості (n= 8 ÷128);

Vi =X1(i-1)+h(K) - параметр мережі;

h(K)= K1 ROL i +K2 ROR i,

K1 та K2 – ліва та права частини ключа K,

ROL та ROR - операції циклічного зсуву вліво та вправо відповідно.

Такий алгоритм має ряд переваг. В першу чергу - простота реалізації та висока швидкодія, що досягається за рахунок використання операцій, що мають високу швидкість виконання.

Дешифрування блоку інформації проводиться тією ж мережею Фейстеля, але з інверсним порядком параметрів мережі. У явному вигляді ключ в алгоритмі не використовується, що підвищує його крипостійкість. При знанні ключа, але відсутності інформації щодо кількості раундів криптоаналітику буде досить складно дешифрувати зашифровану інформацію.

КОНТРОЛЬНІ ЗАПИТАННЯ

1. Представте класичну структуру мережі Фейстеля.
2. Опишіть поняття раунда в мережі Фейстеля?
3. Якими властивостями володіє мережа Фейштеля?
4. Яким чином здійснюється використання ключа при шифруванні?

Порядок виконання роботи

1. Написати програму на мові C++ (чи іншій за згодою викладача) яка виконує криптографічні перетворення відповідно до заданого варіанту, таблиця 1

2. Розробити програму шифрування і дешифрування тексту. Провести шифрування вихідного тексту, отримати шифрограму, здійснити її дешифрування і порівняння з вихідним текстом.

3. Оформити звіт згідно вимог.

Таблиця 1.Параметри для мережі Фейстеля

Номер варіанту	Кількість раундів	Утворююча функція
1	8	Додавання
2	10	Виключне АБО
3	12	Множення за модулем 2^N+1
4	14	Множення за модулем 2^N
5	10	Арифметичний зсув вправо
6	18	Арифметичний зсув вліво
7	20	Множення
8	8	Виключне АБО
9	24	Множення за модулем 2^N+1
10	20	Множення за модулем 2^N
11	18	Арифметичний зсув вправо
12	28	Арифметичний зсув вліво
13	12	Додавання
14	14	Виключне АБО
15	24	Множення за модулем 2^N+1
16	22	Множення за модулем 2^N
17	8	Арифметичний зсув вправо
18	10	Арифметичний зсув вліво
19	22	Додавання
20	14	Виключне АБО
21	20	Множення за модулем 2^N+1
22	18	Множення за модулем 2^N
23	28	Арифметичний зсув вправо
24	30	Арифметичний зсув вліво
25	34	Додавання
26	12	Виключне АБО
27	16	Множення за модулем 2^N+1
28	20	Множення за модулем 2^N
29	24	Арифметичний зсув вправо
30	18	Арифметичний зсув вліво

Лабораторна робота №6

Тема роботи: Ключовий обмін Діффі-Хеллмана.

Мета роботи: освоїти методи генерації великих простих чисел і методи перевірки великих чисел на простоту. Ознайомитися з теоремою Ейлера, навчитися будувати початкові корені по модулю n. Вивчити схему обміну ключами Діффі-Хеллмана.

Теоретичні відомості

Генерація простих великих чисел

Будь-яка криптосистема побудована на використанні ключів. Якщо для забезпечення конфіденційного обміну інформацією між двома користувачами процес обміну ключами тривіальний, то в системі, де кількість користувачів складає десятки і сотні, управління ключами - серйозна проблема. Якщо не забезпечено надійне управління ключовою інформацією, то, заволодівши нею, зловмисник одержує необмежений доступ до всієї інформації. У цьому випадку необхідно в процес шифрування ввести деякі випадкові величини. Зокрема, для реалізації алгоритму RSA потрібні великі прості числа. При цьому, для чисел, близьких до n, ймовірність того, що вибране число виявиться простим, дорівнює 1/ lnn. Тому повна кількість простих чисел, менших n рівне n/lnn.

Вважається, що ймовірність вибору двома людьми одного і того ж великого простого числа надзвичайно мала. Існують різні ймовірнісні перевірки чисел на простоту, що визначають, чи є число простим із заданим ступенем вірогідності. Знайдені таким чином числа називають «промисловими простими», тобто вони прості з контрольованою можливістю помилки. В якості одного з таких методів використовується алгоритм розроблений Майклом Біном згідно з ідеєю Гаррі Міллера.

Тест Рабіна-Міллера

Вибрати для перевірки випадкове число p. Обчислити як найбільше число ділення $p-1$ на 2, тобто b - найбільший показник ступеня числа 2, на яку ділиться $p-1$ без остачі. Потім обчислити m як таке, що $p=1+2^b*m$

1. Вибрати випадкове число a, менше за p.
2. Встановити $j=0$ та $z=a^m \bmod p$.
3. Якщо $z=1$ чи $z=p-1$, то p проходить перевірку і може бути простим числом.
4. Якщо $j>0$ і $z=1$, то p не є простим числом.
5. Встановити $j=j+1$. Якщо $j<b$ і $z<p-1$, встановити $z=z^2 \bmod p$ і повернутися до пункту 4. Якщо $z=p-1$, то p проходить перевірку і може бути

простим числом.

6. Якщо j=b і $z \neq p-1$, то p не є простим числом.

Повторити цю перевірку потрібно t раз. Доведено, що в цьому тесті ймовірність проходження перевірки складеним числом зменшується скоріше, ніж в інших. Гарантується, що 75% можливих значень a виявляться показниками того, що вибране число p – складене. Це означає, що ймовірність прийняти складене число p за просте не перевищує величини $(1/4)^t$.

Алгоритм генерації простого числа

1. Згенерувати випадкове n-бітове число p.

2. Встановити його старший і молодший біти рівними 1. Старший біт гарантуватиме необхідну довжину шуканого числа, а молодший біт – забезпечувати його непарність.

3. Переконатися, що p не ділиться на невеликі прості числа: 3, 5, 7, 11 і т.д. Найбільш ефективною є перевірка на можливість ділення на всі прості числа, менші 2000.

4. Виконати тест Рабіна-Міллера мінімум 5 разів.

Якщо p не пройшло хоча б одну перевірку в пунктах 3 або 4, то воно не є простим. Перевірка того, що випадкове непарне число p не ділиться на 3, 5 і 7, відсікає 54% непарних чисел. Перевірка можливості ділення на всі прості числа, які менші 256, відсікає 80% складених непарних чисел.

Навіть якщо складене число «пройшло» через цей алгоритм, це буде відразу ж помічено, тому що шифрування і дешифрування не будуть працювати.

Побудова початкового кореня по модулю n

Згідно з теоремою Ейлера для будь-яких взаємно простих a та n виконується відношення:

$$a^{\varphi(n)} \equiv 1 \pmod{n}, \tag{1}$$

де $\mu(n)$ визначає функцію Ейлера, значення якої рівне кількості додатних цілих значень, менших n та взаємно простих з n. Для простого числа p виконується:

$$\varphi(p) = p - 1. \tag{2}$$

Якщо припустити, що два числа p і q – прості, тоді для $n=p^{\alpha}q^{\beta}$ функція Ейлера матиме вид:

$$\begin{aligned} \varphi(n) &= \varphi\left(p^{\alpha} \cdot q^{\beta}\right) = \varphi(p^{\alpha}) \cdot \varphi\left(q^{\beta}\right) = \\ &= \left[(p-1) \cdot p^{\alpha-1}\right] \cdot \left[(q-1) \cdot q^{\beta-1}\right]. \end{aligned} \tag{3}$$

Розглянемо більш загальне співвідношення, ніж (1). Визначають, що число *a*, є взаємно простим з модулем *n*, належить показнику m, якщо *m* - таке найменше натуральне число, що виконується порівняння:

$$a^m \equiv 1 (mod\ n) \qquad (4)$$

Якщо *a* і *n* є взаємно простими, то існує, принаймні, одне число $m=\mu(n)$, що задовольняє (4). Найменше з додатних чисел *m*, для яких виконується (4), називається довжиною періоду послідовності, що генерується ступенями *a*. З огляду на те, справедливі наступні властивості.

Властивість 1. Числа $a^0, a^1, \ldots a^{m-1}$, - попарно непорівнянні по модулю *n*.

Властивість 2. $a^\gamma \equiv a^{\gamma'} (\mathrm{mod}\ n) \Leftrightarrow \gamma \equiv \gamma' (\mathrm{mod}\ m)$.

Доведення: розділимо γ та γ' на *m* із залишками: $\gamma = m \cdot q + r,\ \gamma' = m \cdot q' + r'$.

Тоді: $a^\gamma \equiv a^{\gamma'} \Leftrightarrow a^{m \cdot q + r} \equiv a^{m \cdot q' + r'} \Leftrightarrow a^r \equiv a^{r'} \Leftrightarrow r = r'$

Звідси випливає наступна властивість.

Властивість 3. Число *a*, що належить показнику $\mu(n)$ називається початковим коренем за модулем *n*

Властивість 4. За будь-яким простим модулем *p* існує початковий корінь. Початкові корені існують за модулями *2, 4,* p^α, $2p^\alpha$, де *p* - непарне просте число, а $\alpha \in N$.

Властивість 5. Нехай $c = \mu(n)$ і $q_1, q_2, \ldots q_k$, - різні прості дільники числа *c*. Число *a*, взаємно просте з модулем *n*, буде початковим коренем тоді і лише тоді, коли не виконується жодне з наступних порівнянь:

$$a^{c/q_1} \equiv 1 (\mathrm{mod}\ n),\ a^{c/q_2} \equiv 1 (\mathrm{mod}\ n),\ \ldots,\ a^{c/q_k} \equiv 1 (\mathrm{mod}\ n).$$

Доведення. Необхідність випливає з того, що $\alpha^{\mu(n)} \equiv 1 (\mathrm{mod}\ n)$ порівняння не має місця при менших показниках ступеня.

Достатність: припустимо, що α не задовольняє жодному з порівнянь і нехай α належить показнику $m<c$. Тоді $m|c \rightarrow c=mu$. Позначимо через *q* простий дільник *u*. Тоді легко отримати протиріччя:

$$a^{c/q} = a^{mu/q} = (a^m)^{u/q} \equiv 1 (\mathrm{mod}\ n).$$

Якщо деяка послідовність має довжину $\mu(n)$, тоді ціле число *a* генерує своїми ступенями множину всіх ненульових залишків по модулю *n*. Таке ціле число називають початковим коренем за модулем *n*. Для числа *n* їх кількість дорівнює $\mu(\mu(n-1))$, де $\mu()$ - функція Ейлера.

Приклад. Нехай n=41. Отримаємо $c= \mu(41)=40=2^3*5$. Отже, початковий корінь не повинен задовольняти двом порівнянням:

$$a^8 \equiv 1 (\mathrm{mod}\ 41),\ \ a^{20} \equiv 1 (\mathrm{mod}\ 41).$$

Досліджуємо числа 2, 3, 4, …: $2^8 \equiv 10$, $2^{20} \equiv 1$, $3^8 \equiv 1$, $4^8 \equiv 18$, $4^{20} \equiv 1$, $5^8 \equiv 18$, $5^{20} \equiv 1$, $6^8 \equiv 10$, $6^{20} \equiv 40$. Звідси видно, що 6 є найменшим початковим коренем за модулем 41.

Алгоритм обміну ключами за схемою Діффі-Хеллмана

У 1976 році була опублікована робота американських математиків У. Діффі та М.Е. Хеллмана «Нові напрямки в криптографії». У ній вони запропонували конструкцію так званого «відкритого розподілу ключів».

Мета алгоритму полягає в тому, щоб два учасники могли безпечно обмінюватися ключем, який надалі може використовуватися в будь-якому алгоритмі симетричного шифрування. Сам алгоритм Діффі-Хеллмана може застосовуватися лише для обміну ключами. Алгоритм побудований на тому, що складно рахувати дискретні алгоритми.

Безпека обміну ключами в алгоритмі Діффі-Хеллмана витікає з того факту, що, хоча відносно легко обчислити експоненти по модулю простого числа, проте важко обчислити дискретні логарифми. Для великих простих чисел задача вважається нерозв'язною.

Припустимо, що двом абонентам необхідно провести конфіденційну переписку, а в їх розпорядженні немає початково обумовленого секретного ключа. Однак між ними існує канал, що захищений від модифікації, тобто дані, які передаються по ньому, можуть бути «прослухані», але не змінені. У цьому випадку дві сторони можуть створити однаковий секретний ключ, при цьому жодного разу не передавши його по мережі, згідно з наступним алгоритмом (рис.1).

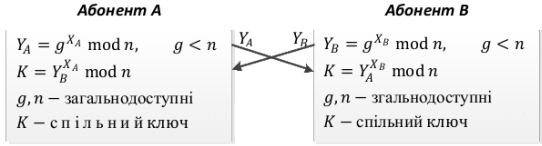

Рис.1. Обмін ключами за схемою Діффі-Хеллмана

Алгоритм полягає в наступному:

1. Задаються глобальні відкриті елементи:
   1) n – випадкове велике просте число;
   2) g – початковий корінь n.
2. Обчислюється ключ абонентом A:
   1) вибирається велике секретне число $X_A (X_A < n)$;
   2) обчислення відкритого значення: $Y_A: Y_A = g^{X_A} \bmod n$.
3. Обчислюється ключ абонентом B:
   1) вибирається велике секретне число $X_B (X_B < n)$;;

2) обчислення відкрите значення: Y_B: $Y_B = g^{X_B} \bmod n$.

4. Обчислюється секретний ключ абонентом *A*: $K = Y_B^{X_A} \bmod n$

5. Обчислюється секретний ключ абонентом *B*: $K = Y_A^{X_B} \bmod n$

Необхідно відзначити, що алгоритм Діффі-Хеллмана працює тільки на лініях зв'язку, які надійно захищені від модифікації.

Приклад. Нехай $n=97$ і $g=5$. Абонент *A* згенерував випадкове число $X_A=36$. Абонент *B* згенерував випадкове число $X_B=58$. Ці елементи вони тримають в таємниці. Далі кожен з них обчислює новий елемент:

$$Y_A = 5^{36} \bmod 97 = 50 \text{ та } Y_B = 5^{58} \bmod 97 = 44$$

Далі вони обмінюються цими елементами по каналу зв'язку. Тепер абонент *A*, отримав Y_B та знаючи свій секретний елемент X_A, обчислює спільний ключ: $K_A = 44^{36} \bmod 97 = 75$. Аналогічно робить абонент *B*: $K_B = 50^{58} \bmod 97 = 75$.

КОНТРОЛЬНІ ЗАПИТАННЯ

1. Поняття великих простих чисел та процес їх генерації.
2. Алгоритм ефективної реалізації піднесення цілого числа в степінь за модулем
3. На чому базується безпека ключового обміну по схемі Діффі-Хеллмана
4. Опишіть процедуру обміну ключами по схемі Діффі-Хеллмана

Порядок виконання роботи

1. Написати програму на мові C++ (чи іншій за згодою викладача), яка надає засоби:

А) генерації великих простих чисел, які перевищують 2^{64} .Програма за заданими t (кількість перевірок за тестом Рабіна-Міллера) та n (кількість біт) повинна генерувати просте n-бітне число, відображаючи при цьому кількість ітерацій алгоритму генерації простого числа, які необхідно виконати для його генерації та час який для цього був необхідний;

Б) програма за заданими межами діапазону повинна виводити всі прості числа з цього діапазону, відображаючи час, витрачений на генерацію всіх чисел;

В) визначити для заданого числа перші 100 початкових коренів, відображаючи при цьому сумарний час витрачений програмою на їх пошук;

Додаткове завдання. Змоделювати ключовий обмін між абонентами за схемою Діффі-Хеллмана. Програма повинна отримувати великі прості числа Xa, Xb та n випадковим чином з допомогою алгоритму генерації простого числа, а також надавати засоби з їх завдання користувачем.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Бернет С. Официальное руководство RSA Security [Текст] : пер. с англ. / С. Бернет, С. ПЄйн. – 2-е изд., стереотип. – М. : Бином, 2009. – 384 с.

2. Долгов В. Криптографические методы защиты информации : учеб. пособие [Текст] / В. Долгов, В. Анисимов. – Хабаровск : Изд-во ДВГУПС, 2008. – 155 с.

3. Олифер В. Компьютерные сети. Принципы, технологии, протоколы : учеб. для вузов [Текст] / В. Олифер, Н. Олифер. – 4-е изд., перераб. и доп. – СПб. : Питер, 2010. – 944 с.

4. Панасенко С. Алгоритмы шифрования. Специальный справочник [Текст] / С. Панасенко. – СПб. : БХВ-Петербург, 2009. – 576 с.

5. Партыка Т. Информационная безопасность : учеб. пособие [Текст] / Т. Партыка, И. Попов. – 3-е изд., перераб. и доп. – М : Форум, 2008. – 431 с. – (Профессиональное образование).

6. Фергюсон Н. Практическая криптография / Н. Фергюсон, Б. Шнайер. – М. : Издательский дом «Вильямс», 2005. – 424 с.

7. Маслов, Ю. Как подписывать с помощью ЕЦП Електронные документы различных форматов [Електронный ресурс]. / Ю. Маслов. // Информационная безопасность. – 2009. – № 1. Режим доступа : http://www.itsec.ru/articles2/bypub/insec-1-2009.

8. Шнайер, Б. Подводные камни безопасности в криптографии [Електронный ресурс] : пер. с англ. / Б. Шнайер // ЦИТ МГТУ им. Н. Є. Баумана. – Електрон. журнал. – М : ЦИТ МГТУ им. Н. Є. Баумана, 2008 – Режим доступа : http://www.citforum.ru/security/cryptography/ pitfalls.shtml.