Міністерство освіти і науки України
Центральноукраїнський національний технічний університет
Механіко-технологічний факультет
Кафедра кібербезпеки та програмного забезпечення

Захист інформації в комп'ютерних системах

Методичні вказівки до виконання лабораторних робіт для студентів денної форми навчання за спеціальністю 123 "Комп'ютерна інженерія"

ЗАТВЕРДЖЕНО
на засіданні кафедри кібербезпеки та програмного забезпечення, протокол № 1 від 05.07.2017

Кропивницький
2017

Захист інформації в комп’ютерних системах : методичні вказівки до виконання лабораторних робіт для студентів денної форми навчання за спеціальністю «Комп’ютерна інженерія» / уклад. О.А. Смірнов, О.К. Конопліцька-Слободенюк, В.Д. Хох, С.А. Смірнов – Кропивницький: ЦНТУ – 2017. – 42 с.

Укладачі: Смірнов О. А., докт. техн. наук, професор;
Конопліцька-Слободенюк О. К., викладач;
Хох В.Д., аспірант;
Смірнов С.А., канд. техн. наук, ст. викладач.

Рецензенти: Сидоренко В. В., докт. техн. наук, професор;
Доренський О. П., канд. техн. наук.

ЗМІСТ

ВСТУП

Мета: Основною метою є теоретична та практична підготовка студентів щодо вивчення теоретичних підвалин теорії захисту інформації у комп'ютерних системах та їх практичних застосувань.

Методи навчання та інформаційно-методичне забезпечення

Заняття проводяться у формі лекцій та лабораторних занять. На лекціях викладається основний теоретичний матеріал. На практичних заняттях студенти навчаються застосовувати теорію до розв'язання задач. Навички, одержані студентами на заняттях, використовуються ними при розв'язанні завдань модульних контрольних робот. Засвоєння навчального матеріалу контролюється за допомогою контрольних робіт та семестрових іспитів. Щотижня надаються консультації.

Завдання:

– вивчення основних понять теорії захисту інформації у комп'ютерних системах;

– набуття практичних навичок з криптографії шляхом програмної реалізації основних алгоритмів;

– освоєння методики кодування із виправленнями;

– освоєння основних методів економного кодування;

– освоєння сучасних методів та алгоритмів захисту інформації в комп'ютерних системах.

У результаті вивчення навчальної дисципліни студент повинен:

– знати: Методи та засоби забезпечення передачі інформації в АСУ. Загрози, яким підлягає інформація. Основні міри протидії загрозам безпеці, принципи побудови систем захисту, основні механізми захисту; Криптографічні методи захисту. Види засобів криптозахисту даних. Переваги і недоліки. Місце і роль засобів криптозахисту; Економне кодування.

Префіксний код та його дерево. Кодування методом Фано. Економічність його. Приклади. Декодування. Однозначність декодування.; Алгоритм Хаффмана. Основні методи економного кодування без втрат послідовної дискретної інформації; Боротьба з помилками, які виникають у каналах передачі даних. Стисла характеристика методів боротьби з помилками; Принципи завадостійкого кодування. Основні характеристики завадостійких кодів. Класифікація завадостійких кодів. Математичний опис процесу кодування і декодування.; Блочні лінійні коди. Коректуючи властивості блочних кодів. Коди з перевіркою на парність. Коди Хеммінга.; Циклічні коди. Засоби опису циклічних кодів. Властивості циклічних кодів по виявленню помилок. Укорочені циклічні коди.; Коди Боуза-Чоудхурі-Хоквінгема. Коди Ріда-Соломона. Код Файра. Згорточні коди. Принципи кодування і декодування.; Ланцюговий код. Каскадні коди.; Сучасні симетричні алгоритми шифрування (AES).; Сучасні симетричні алгоритми шифрування (Калина).; Основні поняття захисту Web-ресурсів.

– вміти: Програмно реалізовувати наступні проекти: Визначення ймовірності події; Шифрування та дешифрування методами Цезаря й Відженера; Шифрування та дешифрування методами ДСТУ 28147:2009 та DES; Шифрування та дешифрування методами методом RSA; Оптимальний код Хафмана та Шеннона-Фано; Код Хеммінга.; Циклічні коди та їх застосування.

Структурно логічна схема підготовки бакалавра

Враховуючи послідовність накопичення знань та інформації, дисципліна вивчається після викладання наступних дисциплін:

– Вища математика. Теорія ймовірності та математична статистика.

– Алгоритми та методи обчислень.

– Програмування.

– Організація баз даних.

– Інженерія програмного забезпечення.

Для опанування матеріалу дисципліни «Захист інформації у комп'ютерних системах» окрім лекційних та лабораторних занять, тобто аудиторного навантаження, значна увага приділяється самостійній роботі.

До основних видів самостійної роботи студента відносимо:

1. Вивчення лекційного матеріалу.

2. Робота з літературними джерелами.

3. Розв'язання практичних задач за індивідуальними варіантами.

4. Підготовка до модульних, підсумкового контролю, екзамену (денна та заочна).

5. Виконання курсової роботи для денної форми навчання.

6. Виконання контрольної роботи для заочної форми навчання.

Студенти заочної форми навчання (ЗФН) здебільшого вивчають матеріал самостійно впродовж семестру, тобто самостійно відпрацьовують теми лекцій, а також лабораторних робіт. Для них на початку семестру проводиться установча сесія, під час якої начитують лекції та проводять лабораторні роботи.

Для підвищення рейтингу впродовж семестру студент може виконати згідно запропонованої викладачем теми самостійну роботу, обсяг якої складає не менше 10 сторінок.

Форма підсумкового контролю: залік.

Максимальну кількість балів студент може одержати у випадку відвідування всіх лекцій, лабораторних занять, виконання і захисту виконаних самостійних завдань у встановлений термін, проходження контролю.

В якості самостійного завдання необхідно виконати теоретичну роботу, згідно обраної студентом теми.

Шкала оцінювання: національна та ECTS

Сума балів за всі види навчальної діяльності	Оцінка ECTS	Оцінка за національною шкалою
		для екзамену
90 – 100	**A**	відмінно
82-89	**B**	добре
74-81	**C**	
64-73	**D**	задовільно
60-63	**E**	
35-59	**FX**	незадовільно з можливістю повторного складання
0-34	**F**	незадовільно з обов’язковим повторним вивченням дисципліни

Вибравши предметну область, над якою ви будете працювати, ви повинні виконати завдання до лабораторних робіт, а також відповісти на питання в кінці кожної лабораторної роботи. Звіт повинен містити хід виконання завдань а також графічні матеріали, що підтверджують виконання цих завдань.

Лабораторна робота N1

ТЕМА. Визначення ймовірності події

МЕТА. Оволодіти методикою визначення ймовірності події.

Теоретичні відомості. Для визначення ймовірності події можливо використати наступний "частотний" метод, що ґрунтується на стійкості послідовності значень появи події. Згідно цього методу слід провести серію дослідів однієї розмірності, в кожному з яких підраховують Ni – кількість тих випадків, коли настає подія, де i-номер досліду. Накопичена частота V обчислюється за такою формулою:

$$V_n = \frac{\sum_{i=1}^{n} N_i}{n \cdot d},$$

де d – розмірність кожного досліду

Отримані числа записуються в таблицю:

№ досліду	1	2		……	N
Ni				…….	
Vi				…….	

Отриману послідовність { Vi } дослідимо на предмет наявності властивості стабільності. Ця властивість полягає в тому, що починаючи з номера N для всіх i>N має місце.

$$| Vi - P| < \varepsilon \text{ де } \varepsilon = 0{,}0001$$

Якщо ε – нескінченно мала величина, то маємо рівність.

$$P = \lim_{i \to \infty} V_i$$

де i – номер серії досліду, V_i – накопичена частота появи букви після i-тої серії.

Завдання

Побудувати таблицю ймовірностей всіх літер алфавіту та записати цю таблицю до файлу. Файл з якого беруться частота появи усіх букв, повинен бути не менш ніж 30 кБ.

Таблиця – Згідно з експериментальними даними безумовні ймовірності букв російського алфавіту

Бука	Імовірність	Буква	Імовірність	Буква	Імовірність
Пробіл	0,175	м	0,026	ч	0,012
о	0.090	д	0.025	й	0.010
е	0.072	п	0.023	х	0.009
а	0.062	у	0.021	ж	0.007
и	0.062	я	0.018	ю	0.006
т	0.053	ы	0.016	ш	0.006
н	0.053	з	0.016	ц	0.004
с	0.045	ь,ъ	0.014	щ	0.003
р	0.040	б	0.014	э	0.003
в	0.038	г	0.013	ф	0.002
л	0.035				
к	0.028				

Лабораторна робота N2

ТЕМА. Шифрування та дешифрування методами Цезаря й Відженера

МЕТА. Оволодіти методами кодування тексту засобами криптографії

Теоретичні відомості

Метод Цезаря.

Розглянемо задачу, що полягає в захисті інформації від несанкціонованого доступу. Розглянемо алфавіт укр. мови та занумеруємо літери так, щоб А мала номер 0, Б – номер 1 і т.д. та "розташуємо" їх в рядок згідно номерів. Внизу випишемо ще такий самий рядок, зсунувши його вправо на одну літеру та переставимо літеру Я на початок другого рядка:

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

Цим самим виконаємо зсув на одну літеру. Аналогічним чином визначаємо зсув на N, або що те саме-циклічну перестановку літер алфавіту. Використаємо цей другий рядок для кодування тексту з метою конфіденційності (захисту від несанкціонованого використання).

Розглянемо обернену задачу – декодування тексту, закодованого при певному зсуві алфавіту. Для того щоб взламати текст закодований методом цезаря застосовується наступний підхід:

Використаємо статистичні методи для визначення величини зсуву наступним чином:

1.Побудуємо таблицю 1 ймовірності всіх літер алфавіту на базі величезного текстового файлу.

2.Побудуємо таблицю 2 частот появи літер в закодованому текстовому файлі.

3.Визначаємо величину зсуву алфавіту:

а)Підрахувати суму різниць частоти літери із кожного рядка та ймовірності появи відповідної літери. Наприклад для уявного зсуву на 1 маємо таку суму:

$$S1 = |V_2(А)-V_1(Я) + V_2(Б)-V_1(А) + V_2(В)-V_1(Б) + ... + V_2(Я)-V_1(Ю)|$$

де

$V_2(А)$ – частота появи літер А в декодованому тексті, $V_1(Я)$ – ймовірність появи літери Я в укр. тексті.

б)Якщо позначити через S_i – суму, побудовану в а) для уявного зсуву на і всіх літер алфавіту, то можливо побудувати множину $\{S_i\}, i = 0,...30$ в якій найменший елемент буде відповідати величині зсуву літер алфавіту.

4.Таким чином статистичним методом знайдемо можливість для декодування тексту, про який відомий спосіб кодування.

Завдання 1

Використати номер свого прізвища в списку групи, як величину зсуву рядка літер алфавіту для кодування інформації. Побудувати програму на основі описаного вище алгоритму для кодування та декодування тексту з лаб. роботи N1.

Метод Відженера.

Кодування тексту більш складним засобом криптографії.

Теоретичні відомості. Розглянемо задачу "шифрування" тексту (інформації) з метою захисту її від зловмисників, розглянуту в попередній роботі, та відмітимо слабкість методу Цезаря щодо "зламу" шифру-зсуву алфавіту. Вдосконалимо цей метод наступним чином:

1) випишемо в рядок №1 весь текст для кодування;

2) виберемо слово (зашифровуючий ключ) та заповнимо його копіями рядок, №2, розташований під рядком №1.

3) використаємо цифрові коди літер алфавіту якими виписано текст та слово-ключ.

Наприклад:

код(А) = 0,код(Б) = 1,...,код(Я) = 30

4) виконаємо операцію додавання за модулем кодів літер вказаних рядків та запишемо результат такого "політерного" додавання до третього рядка.

Наприклад, закодуємо український алфавіт ключем КУ.

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20

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

К У К У К У К У К У К У К У К У К У К У К У К У К У К У К У К

К Ф М……

Якщо код(А) = 0,

код(К) = 12,

код(У) = 21,

то (код(А) + код(К)) mod 30 = 12

код(Б) = 1,

то (код(Б) + код(У)) mod 30 = 22

код (В) = 2,

то код (В) + код (К)) mod 30 = 14

код(М) = 14

В результаті дії 4) отримаємо зашифрований текст в третьому рядку, який в подальшому передається відправником по каналу зв'язку для отримувача інформації, який знає, як було закодовано текст т.ч. розшифрує інформацію. Можлива також передача ключа разом з інформацією. Якщо ключ забуто, але відома його довжина(кількість літер), то можливе декодування при наявності програмного засобу.

Використовуючи датчик випадкових чисел від 0 до 30 для побудови „довгого” зашифровуючого ключа, можливо створити засіб кодування тексту, який неможливо декодувати без отримання ключа по каналах зв'язку.

Завдання 2

Використати номер свого прізвища в списку групи для визначення в алфавіті першої літери слова-ключа, а друга літера цього слова буде наступною за першою. Побудувати програмний засіб для кодування тексту описаним вище алгоритмом та декодування .

Лабораторна робота №3

ТЕМА. Шифрування та дешифрування методами ДСТУ 28147:2009 та DES

МЕТА. Оволодіти методами шифрування та дешифрування

Теоретичні відомості. Побудовані на принципах розсіювання та перемішування з метою максимально ускладнити підбір ключа шифрування.

Розсіювання являє собою розповсюдження впливу одного знаку тексту на багатьох знаків зашифрованого тексту з метою заховати статистичні властивості тексту.

Перемішування означає використання таких шифруючих перетворень, які ускладнюють відтворення зв’язку між текстом та зашифрованим його видом.

Найрозповсюдженішим способом досягнення ефектів розсіювання та перемішування є використання послідовності простих ключів, де кожен простий шифр найчастіше використовує прості перестановки та підстановки. Перестановка означає просте переставлення символів текстів, виконане за допомогою секретного ключа. Під підстановкою розуміють заміну кожного символу тексту іншим символом із того ж самого алфавіту, яка також визначається секретним ключем.

DES (Data Encryption Standard) – стандарт шифрування США використовує комбінацію підстановок та перестановок шляхом шифрування 64-бітових блоків даних за допомогою 64-бітового ключа, в якому значущими є 56, а решта служить для контролю на парність. Дешифрування здійснюється повторенням операцій шифрування в оберненому порядку. Узагальнена схема шифрування в алгоритмі DES має вигляд:

Спрощений алгоритм має вигляд:

Нехай прочитано 8-байтів в буфер T, який перетворюється за допомогою матриці початкової перестановки IP виду:

$$\begin{bmatrix} 58 & 50 & 42 & 34 & 26 & 18 & 10 & 2 \\ 60 & 52 & 44 & 36 & 28 & 20 & 12 & 4 \\ 62 & 54 & 46 & 38 & 30 & 22 & 14 & 6 \\ 64 & 56 & 48 & 40 & 32 & 24 & 16 & 8 \\ 57 & 49 & 41 & 33 & 25 & 17 & 9 & 1 \\ 59 & 51 & 43 & 35 & 27 & 19 & 11 & 3 \\ 61 & 53 & 45 & 37 & 29 & 21 & 13 & 5 \\ 63 & 55 & 47 & 39 & 31 & 23 & 15 & 7 \end{bmatrix}$$

Згідно наведеної матриці IP 58 біт з T стає 1-м, 50-й стає 2-м і т.д., тобто з T маємо T0 і запишемо $T_0 = IP(T)$. Потім T_{10} розділимо на L_0 – ліву та R_0 – праву половини. Потім виконується 16 ітерацій, де в результаті першої ітерації маємо $T_1, T_1 = L_1 R_1$ – тобто конкатенація рядків, а друга така $L_2 = R_1$, $R_2 = L_1 + f(R_1, K_1) (\text{mod } 2)$, де f – функція шифрування, аргументами якої будуть

R_1 – права (4-байти) частина з попереднього кроку та K_1 – 48-бітовий ключ, отримані в результаті перетворень з 64-бітового ключа К. З метою спрощення вважатимемо, що $K_i = K$ а функція $f\ xor$ – операція додавання R_1 до K за модулем 2. На останньому кроці матимемо $T_{16} = R_{16}L_{16}$ з якої відновлюємо позиції бітів за допомогою матриці оберненої перестановки IP^{-1}:

40	6	48	16	56	24	64	32
39	7	47	15	55	23	63	31
38	6	46	14	54	22	62	30
37	5	45	13	53	21	61	29
36	4	44	12	52	20	60	28
35	3	43	11	51	19	59	27
34	2	42	10	50	18	58	26
33	1	41	9	49	17	57	25

Отримаємо зашифрований блок T' із вхідного T.

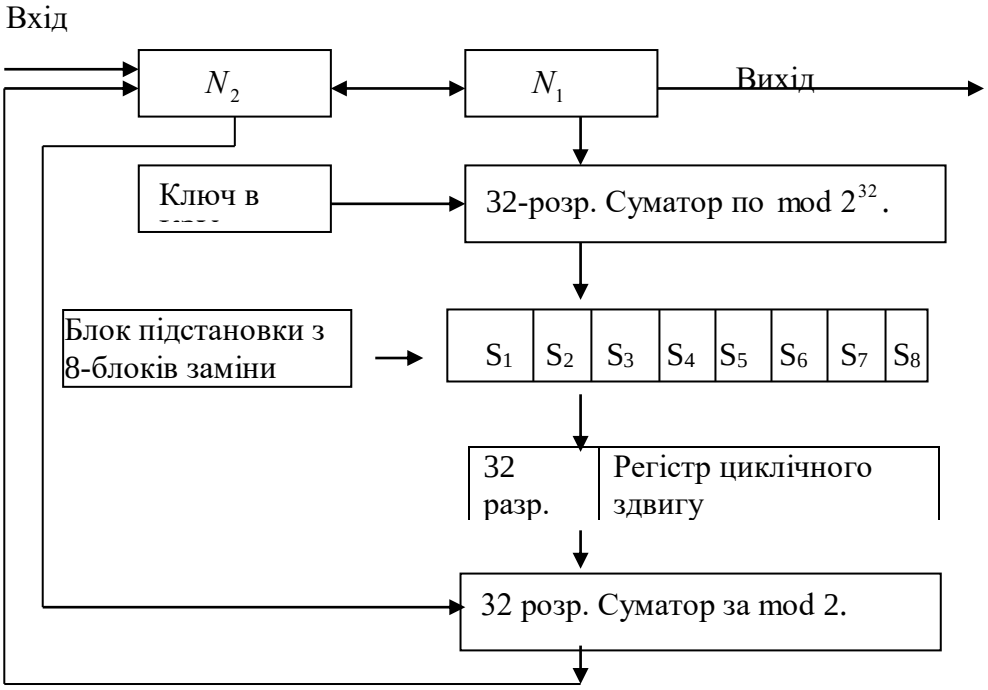
Дешифрування є інверсним (оберненим) по відношенню до процесу шифрування.

Стандарт шифрування даних – ДСТУ 28147:2009 являє собою алгоритм для апаратної та програмної реалізації криптографічного перетворення блоків даних (по 64 біти) із 256-бітовим ключем. Серед чотирьох режимів роботи алгоритму схематично розглянемо:

– шифрування в режимі простої заміни;

– шифрування в режимі гамування.

Для реалізації алгоритму шифрування даних наведемо схему:



Дані, що підлягають шифруванню, розбивають на 64-розрядні блоки та поблочно проходять тут процедуру 32-х ітерацій шифрування ключем К, що зберігається в КЗУ у вигляді восьми 32-розрядних К і підключів (чисел), $К = К_7К_6К_5К_4К_3К_2К_1К_0$.

Робота цієї процедури розпочинається з поділу вхідного блоку на дві половини, по 32 біти, де в першій половині містяться молодші біти зліва направо, а в правій – старіші, які вводять до накопичувачів N_1 та N_2 в оберненому порядку, тобто зліва направо розміщено 32-й, 31-й…1-й біти.

Перша ітерація процедури полягає в послідовному виконанні двох операцій додавання:

– змісту N_1 із ключем K_0 за модулем 32;

– потім додавання за модулем 2 до результату попередньої операції змісту накопичувача N_2, результат пишемо в N_1 а в N_2 заносимо початковий зміст регістру N_1.

Першу операцію додавання звуть підстановкою (заміною) бо виконується блоком підстановки , що має 8 вузлів по 64-біта кожний, в першому суматорі СМ1. Кожен вузол можливо уявити у вигляді таблиці – перестановки

шістнадцяти чотирьохрозрядних двійкових числа (в діапазоні від 0000….1111) та є загальним (рідко змінюються). Вхідні дані визначають рядок 8 таблиці, а число 8 рядку буде вихідним, які послідовно об'єднуються у 32-розрядний блок. Наступна операція – зсув циклічно вліво на 11 розрядів вихідного блоку, отриманого після .

Друга операція – порозрядне додавання за модулем 2 виконується в другому суматорі $СМ_2$ над вихідним блоком після зсуву, та змістом накопичувача N_2. Результат цієї операції записують в N_1, а збережене старе значення N_1 записують в N_2. Перша ітерація процедури шифрування на цьому завершена. Аналогічно, на другому циклі з КЗУ зчитується $К_1$, потім $К_2$, а починаючи із 25 – 32-ї ітерацій порядок зчитування підключів $К_i$ змінюється на протилежний.

На заключній 32-й ітерації результат із суматора вводиться в накопичувач N_2, а в N_1 вводимо попереднє значення накопичувача N_2. Об'єднавши послідовно значення із N_1 та N_2 отримаємо зашифрований блок даних.

Розшифрування здійснюється в оберненому порядку, тільки порядок зчитування підключів із КЗУ в наступному порядку: $К_0$, $К_2$, $К_3$, $К_4$, $К_5$, $К_6$, $К_7$, $К_7$, $К_6$, $К_5$, $К_4$, $К_3$, $К_2$, $К_1$, $К_0$, $К_7$, $К_6$, $К_5$, $К_4$, $К_3$, $К_2$, $К_1$, $К_0$, $К_7$, $К_6$, $К_5$, $К_4$, $К_3$, $К_2$, $К_1$, $К_0$.

Опис режиму простої заміни закінчено.

Шифрування в режимі гамування полягає у застосуванні гами шифрів, що побудовані "випадковим" чином із використанням датчиків псевдовипадкових чисел. Ці шифри використовують не тільки для процедури кодування, але й передаються для виконання дешифрування. Робота алгоритму ДСТУ 28147:2009 відрізняється від роботи в попередньому режимі саме тим, що замість блоку підстановки використано гаму шифрів, що буде змінена після закінчення шифрування та передачі даних.

Завдання.

Побудувати програмну реалізацію алгоритмів.

– DES;

– ДСТУ 28147:2009.

Порівняти ефективність шифрування між цими методами.

Лабораторна робота N 4

ТЕМА. Шифрування та дешифрування методами методом RSA

МЕТА. Освоєння альтернативних методів криптографії.

Теоретичні відомості.

У RSA відкритим параметром є багаторазрядний модуль перетворення (не менш 512 біт)

$$N_j = P_j \cdot Q_j,$$

де p_j, q_j – сильні прості числа відповідної розрядності.

Ключі E_k. і D_k зв'язані співвідношенням

$$E_k \cdot D_k \equiv 1 (\mathrm{mod}\, \varphi(N_j)),$$

де $\varphi(N_j) = (P_j - 1)(Q_j - 1)$ – функція Ейлера.

Ключ формування підпису E_k є закритим, а ключ зняття підпису D_k звичайно відкритий.

Формування зашифрованої інформації здійснюється за правилом:

$$ЗІ = ВІ^{E_k} (\mathrm{mod}\, N_j),$$

де:

ВІ – відкрита інформація;

ЗІ – зашифрована інформація.

Для розшифрування повідомлення використовується співвідношення

$$ВІ' = (ЗІ')^{D_k} (\mathrm{mod}\, N_j),$$

де:

ВІ’ – відкритий текст, отриманий прийомною стороною із прийнятої закритої інформації (ЗІ’).

Вважаємо, що :

1) Символ тексту розглядаємо, як числа (ASCII кодом), та позначаємо через m .

2) Відомі обом сторонам взаємнопрості числа e і d, де e-зашифровуючий ключ, d-розшифровуючий ключ. Піднесення до ступеню e числа m означає шифрування, а піднесення отриманого числа до ступеня d буде дешифруванням символу, що має код m .

Завдання 1:

1. Побудувати програмний засіб для реалізації наведеного методу .

2. Яким чином можливо "зламати" текст, зашифрований цим методом?

Лабораторна робота №5

ТЕМА. Оптимальний код Хафмана та Шеннона-Фано.

МЕТА Вивчення кодів, що стискають інформацію.

Теоретичні відомості. Цей метод ґрунтується на ідеї кодування найкоротшим кодовим словом те повідомлення, що має найбільшу ймовірність. Ця ідея реалізується шляхом послідовного стискання початкової множини повідомлень та ілюструється наступним прикладом: для п'яти різних повідомлень із ймовірностями:

$P_1 = 0.4$; $P_2 = 0.2$; $P_3 = P_4 = 0.15$; $P_5 = 0.1$

Номер	Ймовірність та кодові позначення				
	Вхідна множина ймовірностей	1-стискання	2-стискання	3-стискання	коди
1	0.4	0,4	0,4	0,4 0	0
2	0.2	0,2 0	0,35 0		100
3	0.15	0,15 1		0,6 1	101
4	0,15 0	0,25	0,25 1		110
5	0,1 1				111

В цій таблиці наведено три послідовні стискання вихідної множини:

Перше стискання отримане шляхом заміни двох (4 + 5) повідомлень одним та переписування трьох інших повідомлень. Друге стискання отримане із результату першого шляхом заміни (2 та 3) із найменшими двома ймовірностями. Так само виконується третє стискання в результаті якого

отримаємо два числа, сума яких = 1. Кодуємо кожну пару "стиснутих" чисел 0 та 1 та виписуємо кодові слова, рухаючись у зворотному напрямку:

(3стиск-->2стиск-->1стиск).

Отримаємо колонку кодових слів. Обчислимо середню довжину кодового слова:

$$L = \sum_{i=5}^{5} L_i P_i = 1 \cdot 0.4 + 2(0.2 + 0.15 + 0.15 + 0.1) = 0.4 + 1.2 = 1.6 \text{ біт}$$

Середня довжина L вимірюється в бітах, бо слово "Біт" означає, як кодовий символ, так і одиницю виміру інформації. З іншого боку L може визначитися, як M (ξ) ,де ξ – випадкова величина, визначена на множині з п'яти повідомлень A (i-те повідомлення має ймовірність P_i, i = 1,2,3..5) та має значення $\xi_i = \xi \{i\} = i$. Тобто L = M(ξ)."Середнє" слово містить кількість інформації, що дорівнює H(ξ), яка обчислюється за формулою

$$H(\xi) = \sum_{i=1}^{5} P_i \cdot \log_2\left(\frac{1}{P_i}\right) \quad \text{(бітів)}$$

Дробове число H(ξ) вказує на те , що деякі кодові символи несуть такі частини інформації, що мають спільну частину, тобто перекриваються за рахунок спільної частинки інформації. Виконується нерівність:

$$L \geq H(\xi),$$

яка вказує на теоретичну границю стискання інформації шляхом кодування.

Код Фано має подібну (до наведеної вище) схему стискання. Тільки використовується розбиття на дві "рівноймовірні" групи та подальше кодування цих частин 0 та 1 до того моменту коли залишиться одне число в групі.

1	0.4	0	0		00
2	0.1		1		01
3	0.2	1	0		10
4	0.15		1	0	110
5	0.15			1	111

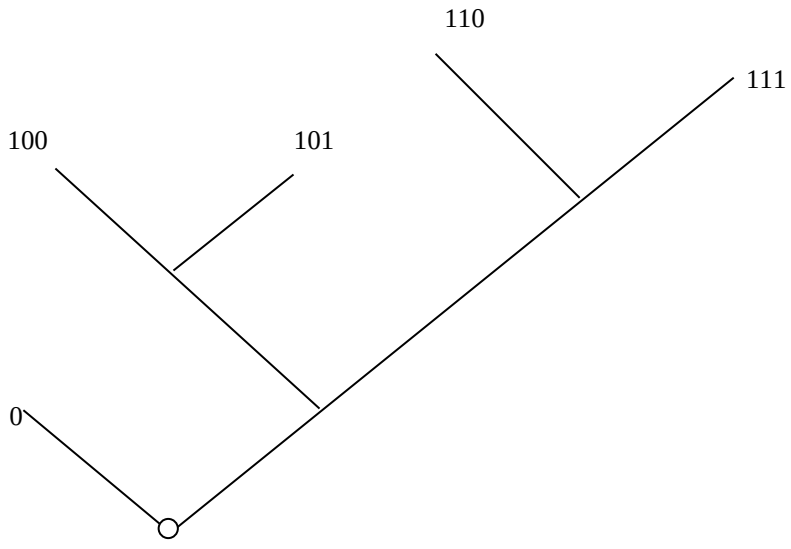
Кодові слова для кожного повідомлення отримаємо шляхом "зворотнього" виписування рядка 0 та 1.

Середня довжина L1 слова буде такою:

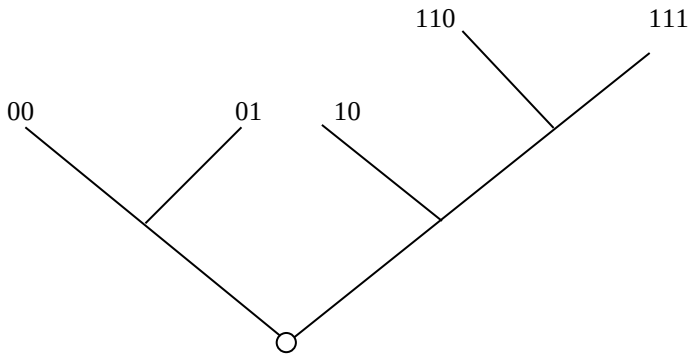
$$L_1 = 2 \cdot 0.4 + 2 \cdot 0.1 + 2 \cdot 0.2 + 3(0.15 + 0.15) = 1.4 + 0.9 = 2.3 \text{ біта}$$

Порівнюючи наведені методи маємо , що $L_1 > L$.

Дерево для оптимального коду Хафмана



Дерево для коду Фано



Практична реалізація коду Шеннона-Фано

Спочатку кожному символу інформаційного алфавіту ставиться у відповідність код нульової довжини («порожній» код) і вага, рівна ймовірності появи символу на виході інформаційного джерела в рамках обраної інформаційної моделі. Всі символи алфавіту сортуються по убуванню або зростанню їхніх ваг, після чого впорядкований ряд символів у деякому місці ділиться на дві частини так, щоб у кожній з них сума ваг символів була приблизно однакова. До коду символів, що належать однієї із частин, додається «0», а до коду символів, що належать іншої частини, додається «1» (додається значення, що, формує черговий крайній правий розряд коду). Як неважко зрозуміти, кожна із зазначених частин сама по собі є впорядкованим рядом символів. Кожний із цих рядів, якщо він містить більше одного символу, у свою чергу ділиться на дві частини відповідно до описаного вище принципом, і до коду символів знову додаються відповідні двійкові значення й т.д. Процес завершується тоді, коли у всіх отриманих у такий спосіб рядах залишається рівно по одному символу.

Як видно, алгоритм Шеннона-Фано в дійсності не будує кодове дерево, однак його роботу можна проілюструвати такою побудовою. При поділі групи символів відбувається як би розгалуження деякого вузла бінарного дерева (листовий вузол стає вузлом батьком для двох новостворених дочірніх вузлів),

а присвоєння нулів і одиниць рівносильно встановленню відповідності між кодами й маршрутами руху по дереву від кореневого вузла до листового вузла. Робота алгоритму Шеннона-Фано проілюстрована на наступному прикладі (рисунок нижче).

Як же ми ділили на групи ? Досить просто:

Розміщаємо символи по ймовірності появи.

ймовірність першої групи (p1) і другої (p2) дорівнює нулю;

p1 < = p2 ?

– так: додати в першу групу символ з початку таблиці;

– ні: додати в другу групу символ з кінця таблиці;

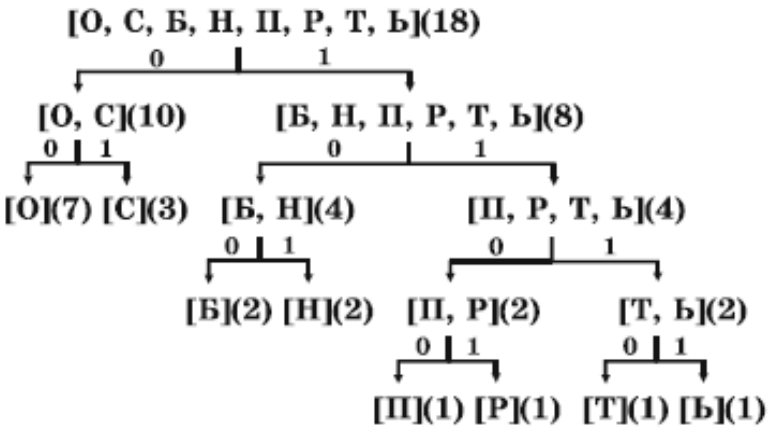
3. якщо всі символи розділені на групи, то завершити алгоритм, інакше перейти до кроку 2

Кодируемое сообщение:
«ОБОРОНОСПОСОБНОСТЬ»

Статистика появления букв в сообщении:

Б(2), Н(2), О(7), П(1), Р(1), С(3), Т(1), Ь(1)

Построение системы префиксных кодов:



Система префиксных кодов:

Б	Н	О	П	Р	С	Т	Ь
«100»	«101»	«00»	«1100»	«1101»	«01»	«1110»	«1111»

Ілюстрація роботи алгоритму Шеннона-Фано.

Практична реалізація кода Хафмана

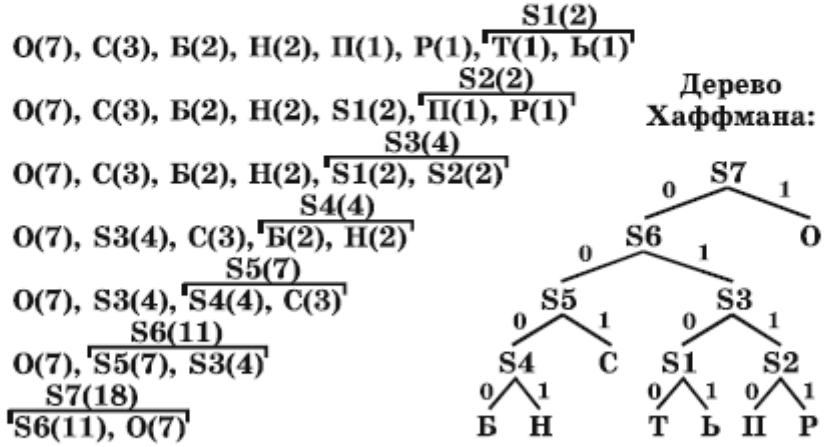
На початковому етапі роботи алгоритму кожному символу інформаційного алфавіту ставиться у відповідність вага, рівна ймовірності (частоті) появи даного символу в інформації. Символи містяться в список, що сортується по убуванню ваг. На кожному кроці (ітерації) два останніх елементи списку поєднуються в новий елемент, що потім міститься в список замість двох поєднуваних елементів. Новому елементу списку ставиться у відповідність вага, дорівнює сумі ваг елементів, що заміщаються. Кожна ітерація закінчується впорядкуванням отриманого нового списку, що завжди містить на один елемент менше, ніж старий список. Паралельно з роботою зазначеної процедури здійснюється послідовна побудова кодового дерева. На кожному кроці алгоритму будь-якому елементу списку відповідає кореневий вузол бінарного дерева, що складається з вершин, які відповідають елементам, об'єднанням яких був отриманий даний елемент. При об'єднанні двох елементів списку відбувається об'єднання відповідних дерев в одне нове бінарне дерево, у якому кореневий вузол відповідає новому елементу, що поміщається в список, а елементам списку, що заміщуються, відповідають дочірні вузли цього кореневого вузла. Алгоритм завершує роботу, коли в списку залишається один елемент, що відповідає кореневому вузлу побудованого бінарного дерева. Це дерево називається деревом Хаффмана. Система префіксних кодів може бути отримана шляхом присвоювання конкретних двійкових значень ребрам цього дерева. Приклад побудови дерева Хаффмана наведений нижче.

Кодируемое сообщение:
«ОБОРОНОСПОСОБНОСТЬ»

Статистика появленя букв в сообщении:

Б(2), Н(2), О(7), П(1), Р(1), С(3), Т(1), Ь(1)

Построение системы префиксных кодов:



Система префиксных кодов:

Б	Н	О	П	Р	С	Т	Ь
«0000»	«0001»	«1»	«0110»	«0111»	«001»	«0100»	«0101»

Ілюстрація роботи алгоритму Хаффмана

Завдання

1. Обчислити ймовірність літер вашого прізвища, доповнити отриману множину чисел ще одним числом , так щоб сума усіх дорівнювала 1. Закодувати прізвище оптимальним кодом Хафмана та кодом Фано, побудувати їх дерева, обчислити середні довжини слів та кількість інформації в них.

2. Побудувати програми, що стискають текстову інформацію в файлі за оптимальним методом Шеннона-Фано та методом Хафмана, а потім розтискають її. Розмір файлу не менший 30 кБ.

Лабораторна робота N 6

Тема. Код Хеммінга

Мета. Вивчення кодів, що виправляють помилки, породжені при передачі інформації по каналу зв'язку. Побудувати код Хемінга, що виправляє одиничні помилки та виявляє подвійні помилки.

Теоретичні відомості. Розглянемо задачу побудови коду, що складається із 16-ти кодових слів виду a1a2a3a4, де a1 = 0 або 1 в якому можливе виявлення та виправлення помилок.

Для розв'язку використаємо три допоміжні (перевірочні) символи, тобто кожне з 16-ти кодових слів кодуємо двійковим словом довжини 7: $a_1a_2a_3a_4a_5a_6a_7$. Нашу задачу можливо переформулювати, як визначення одного із чисел 0,1...7, які б вказували на місце похибки (0 – немає похибки). Накладемо умови (у вигляді рівностей за mod 2) на перевірочні символи:

$a_5 = a_2 + a_3 + a_4$

$a_6 = a_1 + a_3 + a_4$

$a_7 = a_1 + a_2 + a_4$

Для виявлення похибки достатньо обчислити суму:

$$S_1 = a_4 + a_5 + a_6 + a_7.$$

Якщо $S_1 = 1$ то похибка є, інакше "немає".

При наявності похибки перевіримо, чи не міститься вона серед a_1 чи a_7.

Для цього обчислимо

$$S_2 = a_2 + a_3 + a_6 + a_7.$$

Якщо $S_1 = S_2 = 1$ то похибка в a_6 або a_7.

Якщо $S_1 = 1$, $S_2 = 0$ то похибка в a_4 або a_5.

Якщо $S_1 = 0$, $S_2 = 1$ то похибка в a_2 або a_3.

Якщо $S_1 = S_2 = 0$ то похибка в a_1 або немає.

Для визначення місця похибки слід обчислити суму

$$S_3 = a_1 + a_3 + a_5 + a_7.$$

Тобто матимемо три перевірочні відношення

$$S_1 = a_4 + a_5 + a_6 + a_7 \neq 0$$

$$S_2 = a_3 + a_2 + a_6 + a_7 \neq 0$$

$$S_3 = a_1 + a_3 + a_5 + a_7 \neq 0$$

які шляхом порівняння з 0(mod2) "обчислюють" або відсутність похибки, або вказують її місце. Положення одиничної похибки визначається числом $S_1S_2S_3$ в двійковій системі лічби. Таким чином маємо код (4,7) Хемінга довжина слова 7 та 4 інформаційні символи.

Якщо виявляти подвійну похибку, то слід виконати перевірку:

$$S_0 = a_0 + a_1 + a_2 + a_3 + a_4 + a_5 + a_6 + a_7 \neq 0$$

де a_0 – додатковий (четвертий) перевірочний символ, значення обчислюємо за формулою

$$a_0 = a_1 + a_2 + a_3 + a_4 + a_5 + a_6 + a_7 \pmod 2$$

Попередні a5, a6, a7 допомагали виявити та виправити одиночну похибку.

Якщо виникла подвійна похибка, то $S_0 = 0(\text{mod } 2)$ та хоча б одне із чисел S1,S2,S3 буде не нульовим. Виправлення такої похибки неможливе. Побудована множина кодових слів зветься розширеним кодом (4,8) Хемінга (довжина слова 8 та з них інформаційними символами).

Завдання .

Скласти програму побудови коду Хемінга, його систему перевірочних відношень та його розширений варіант для кодових слів довжини 2m-1 (символів), серед яких присутні m перевірочних символів.

Якщо N ваш номер в журнальному списку групи, то $m = N(\text{mod } 5) + 3$.

Лабораторна робота № 7

ТЕМА. Циклічні коди та їх застосування

МЕТА. Опанувати циклічні коди

Теоретичні відомості

Циклічні коди – найцінніша частина теорії кодування завдяки ідеальній пристосованості до реалізації в сучасних технічних засобах. Але теоретичне обґрунтування алгоритмів кодування-декодування та їх простих реалізацій вимагає використання складного алгебраїчного апарату, тому познайомимося лише із результатами без їхнього обґрунтування.

Циклічним зсувом вектора $\vec{a}$, $\vec{a} = (a_0, a_1, ..., a_{n-1})$ із координатами із множини F, F = {0,1}, будемо називати вектор $\vec{a}'$, $\vec{a}' = (a_{n-1}, a_0, a_1, ..., a_{n-1})$. Наприклад для вектора (01101) послідовні зсуви мають вигляд: (10110),(01011),(10101),(11010).

Циклічним кодом зветься лінійний код, який разом із кожним своїм вектором містить також його циклічний зсув. Наприклад циклічним є код із породжуючою матрицею G, де

$$G = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{pmatrix}.$$

При розгляді циклічних кодів використовують операцію циклічного зсуву для опису якої використовують поліноми . Для цього із кожним вектором $\vec{a} = (a_0, a_1, ..., a_{n-1})$ зв´яжемо поліном a(x), де $a(x) = a_0 + a_1 x + ... + a_{n-1} x^{n-1}$. Циклічному зсуву $\vec{a}$ вправо на 1 відповідатиме поліном $a'(x) = a_{n-1} + a_0 x + a_1 x^2 + ... + a_{n-2} x^{n-1}$. Порівнявши $a(x)$ та $a'(x)$ отримаємо, що $a'(x) = xa(x) - a_{n-1}(x^n - 1)$. Будемо вважати x породжуючим для групи степенів x від 0 до n-1, причому $x^n = 1$ та $x^k x^m = x^r$, де $r \equiv (k+m) (\mathrm{mod}\, n)$. Із урахуванням цього маємо наступне порівняльне співвідношення для $a(x)$ та $a'(x)$: $a'(x) = xa(x)$, тобто циклічний зсув довільного вектора маємо після множення цього вектора на x. Враховуючи породжуючі властивості x, маємо таке правило множення двох поліномів степені $\le n-1$: для

$$a(x) = a_0 + a_1 x + ... + a_{n-1} x^{n-1},$$

$$b(x) = b_0 + b_1 x + ... + b_{x-1} x^{n-1},$$

знаходимо добуток спочатку звичайним способом: розкриваючи дужки, множимо x^k та x^m; згідно правила маємо $x^n, r = (k+m)(\text{mod}\, n)$, а коефіцієнти a_k та b_m визначаємо по правилу множення на множині $F, F = \{0,1\}$, беручи добуток за модулем 2. Наприклад: n = 5, $a(x) = 1 + x + x^2 + x^4, b(x) = 1 + x^3 + x^4$ двійкові поліноми. Тоді $a(x)b(x) = \langle x + x = 2x = 0 \rangle = 1 + x^4$.

Таким же чином визначимо операцію додавання поліномів. Наприклад, $a(x) + b(x) = x + x^2 + x^3$. Відносно цих двох операцій на множині F_n – всіх поліномів степеня $\le n$ має місце наступне твердження для деякої V, $V, V \subset F_n$ - довільний поліном $a(x) \in V$ можливо уявити, як добуток фіксованого породжуючого полінома $q(x)$ та деякого підходящого $s(x)$, то

$$a(x) = q(x)s(x).$$

Тобто, якщо відомий породжуючий поліном, то тим самим вичерпуються можливі добутки на поліном $s(x)$ - довільний поліном степеня менше n.

Нагадаємо, що для визначення довільного коду треба навести повний список кодових слів, а для лінійного коду потрібен список кодових векторів. Для циклічного коду достатньо привести лише один кодовий поліном $q(x)$ – породжуючий. Визначимо породжуючу матрицю циклічного коду по заданому $q(x)$, де $q(x) = q_0 + q_1 x + ... + q_m x^m$ - поліном степеня m, де $n = m + k$. Розглянемо наступні поліноми $\{q(x), xq(x), x^2 q(x), ..., x^{k-1} q(x)\}$, які є кодовими поліномами, бо степінь їх не перевищує $n-1$. З іншого, векторного, боку всі вони утворюють лінійно незалежну систему і всякий кодовий вектор є лінійною комбінацією цих базисних векторів. Тоді випишемо в рядок матриці ці поліноми із наведеної вище множини та отримаємо породжуючу матрицю порядку $k \times n$:

Матриця матиме вигляд:

$$k_{\text{рядків}}\left(\begin{Bmatrix} g_0 & g_1 & \dots & g_m & 0 & \dots & 0 \\ 0 & g_0 & g_1 & g_{m-1} & g_m & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & 0 & g_0 & g_1 & g_m \end{Bmatrix}\right) = \begin{pmatrix} g(x) \\ xg(x) \\ \dots \\ x^{k-1}g(x) \end{pmatrix}$$

Наприклад, розглянемо двійковий циклічний код довжини 7 із породжуючим поліномом $g(x) = 1 + x^2 + x^3 = (1011000)$. Тоді

$x\,g(x) = x + x^3 + x^4 = (0101100) = 0*1 + 1*x + 0*x^2 + 1*x^3 + 1*x^4 + 0*x^5 + 0*x^6$;

$x^2 g(x) = (0010110)$,

$x^3 g(x) = (0001011)$

Отримаємо матрицю

$$G = \begin{pmatrix} 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 \end{pmatrix}$$

Серед циклічних кодів існують такі, що виправляють довільну задану кількість помилок при передачі, бо справедлива наступна теорема(без доведення).

Для довільних значень m та t $(t<(2^m-1)/2)$ існує двійковий циклічний код довжини 2^m-1, який виправляє всі комбінації із t або меншого числа помилок і має не більше ніж mt перевірочних символів.

Про перевірочний поліном h(x) треба сказати, що він задовольняє рівності $h(x) = \dfrac{x^n - 1}{g(x)}$, де g(x)- породжуючий поліном. Використовуючи таке визначення, можливо побудувати перевірочну матрицю циклічного коду. А саме, нехай $h(x) = h_0 + h_1x + h_2x^2 + \dots + h_kx^к$ – породжуючий поліном. Для множини F_n- всіх поліномів степеня n маємо, що $x^n = 1$,тобто наведена умова для g(x) та h(x) матиме вигляд $g(x)*h(x) = 0$.

Розглянемо матрицю H порядку m n , що має наступний вид:

$$H = \begin{pmatrix} 0 & \dots & 0 & h_k & \dots & h_1 & h_0 \\ 0 & 0 & h_k & h_{k-1} & \dots & h_0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ h_k & \dots & \dots & h1 & h_0 & 0 & 0 \end{pmatrix}$$

де перший рядок складають коефіцієнти полінома h(x) записані в оберненому порядку, а наступні рядки складені для поліномів xh(x),… $x^{m+1}h(x)$.

Наприклад, для наведеного вище циклічного коду(7,4) n = 7 із породжуючим поліномом $g(x) = 1 + x^2 + x^3$, якій ділить X^7-1.

Дійсно:

$$X^7-1 = (X-1)(X^3 + X + 1)(X^3 + X^2 + 1),$$

тобто перевірочний поліном:

$$h(x) = (X-1)(X^3 + X + 1) = 1 + X^2 + X^3 + X^4 = (1011100),$$

тоді перевірочна матриця має вигляд:

$$H = \begin{pmatrix} 0 & 0 & 1 & 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 & 0 \end{pmatrix}.$$

Перевірка побудованих матиць здійснюється наступним матричним рівнянням

$G * H^T = \varnothing$

де $\varnothing$ – нульова матриця.

Завдання:

1) Знайти G-породжуючу та H-перевірочну матриці циклічного коду заданої довжини n з породжуючим поліномом g(x) , та зробити перевірку свого варіанту.

2) Побудувати всі кодові слова для коду із завдання 1.

3) Скласти програму для реалізації алгоритму.

Варіанти:

1. n = 7, а) $g(x) = 1 + x^3 + x$;
2. n = 7, б) $g(x) = (x-1)(1 + x^2 + x^3)$;
3. n = 9, а) $g(x) = 1 + x^3 + x^6$;
4. n = 9, б) $g(x) = 1 + x + x^2$;
5. n = 9, в) $g(x) = (x-1)(1 + x + x^2)$;
6. n = 9, г)$g(x) = (x-1)(1 + x^3 + x^6)$;
7. n = 15, а) $g(x) = x-1$;
8. n = 15, б)$g(x) = 1 + x + x^2$;
9. n = 15, в)$g(x) = 1 + x^3 + x^4$;
10. n = 15, г)$g(x) = 1 + x + x^4$;
11. n = 15, д.)$g(x) = 1 + x + x^2 + x^3 + x^4$;
12. n = 15, е.)$g(x) = x^2 + x + 1$;
13. n = 15, ж)$g(x) = (x-1)(1 + x + x^2)$;
14. n = 15, з)$g(x) = (x-1)(1 + x^3 + x^4)$;
15. n = 15, и)$g(x) = (x-1)(1 + x + x^4)$;

Лабораторна робота№8

ТЕМА. LZW-архіватор

МЕТА. Опанувати LZW-архіватор

Теоретичні відомості

Алгоритм LZW побудовано навколо таблиці фраз (словника), яка відображає рядки символів стискуваного повідомлення в коді фіксованої довжини 12 бітів. Таблиця має властивість передування, яка означає, що для кожної фрази словника, яка має вигляд wK маємо фразу w яка міститься в словнику. Алгоритм кодера такий:

Ініціалізація словника односимвольними фразами, тобто символами вхідного алфавіту:

А	1
Б	2
В	3

Прочитати перший символ вхідного повідомлення в фразу w;

Крок алгоритму:

Прочитати черговий символ повідомлення k;

Якщо k = кінець_повідомлення то

    Видати код для w

    Вихід;

Кінець якщо;

Якщо фраза wk вже присутня в словнику то

    Замінити w на код фрази wk

    Повторити крок алгоритму

Інакше

    Видати код w;

    Видати фразу wk в словник

    W: = k

    Повторити крок алгоритму

Кінець якщо;

Символи повідомлення	WK = код + символ	Вихід	Вихід словник (код)
А	1		-
Б	1Б	1	1Б(4)
А	2А	2	2А(5)
Б	1Б	-	-
В	4В	4	4В(6)
Б	3Б	3	3Б(7)
А	2А	-	-
Б	5Б	5	5Б(8)
А	2А	-	-
Б	5Б	-	
А	8А	8	8А(9)
А	1А	1	1А(10)
А	1А	-	
А	10А	10	10А(11)
А	1А	-	
А	10А		
		11	

Декодер LZW (Lempel-Ziv-Welch) повинен використати той же словник, що й кодер, будуючи його за аналогічними правилами при відновленні стиснутих даних. Для цього кожен зчитаний рядок символів розкладається на попередню фразу w та символ k. Потім рекурсивно для попередньої фрази w до тих пір, поки вона не стає кодом одного символу. При цьому завершується декодування рядка вхідних символів.

Оновлення словника проводиться для кожного нового рядка крім першого. Після завершення декодування рядка його останній символ разом із попередньою фразою додається в словник. Нова фраза отримає те ж значення (номер) що й в декодері. Таким чином декодер відтворює словник кодера.

Алгоритм декодування має вид:

Код = першому коду повідомлення();

Попередній Код = Код;

Видати символ К у якого Код(К) = = код;

Останній символ:

Код = черговий код повідомлення();

Вхідний код = Код;

Якщо кінець повідомлення то вихід;

Якщо Код_невідомий то

Видати (останній символ)

Код = Попередній символ;

Вхідний код = Код(Попередній код, Останній код);

Кінець якщо;

Наступний символ:

Якщо Код = = Код(w_k) то

В_Стек(К);

Код = Код(w);

Повторити наступний символ;

Інакше якщо Код = = Код(К);

Видати К

Останній символ = к

Поки Стек_не_пустий видати (з_стеку)

Додати в словник (Попередній код, К);

Попередній код = вхідний код;

Повторити наступний символ

Кінець якщо.

Завдання

Запрограмувати наведений вище алгоритм кодування та декодування.

Список використаної літератури

1. Хорошко В.А. Методы и средства защиты информации / Хорошко В.А., Чекатков А.А. / Под ред. Ю.С. Ковтанюка. – К.: Юниор, 2003. – 504 с.

2. Термінологічний довідник з питань технічного захисту інформації / Коженевський С.Р., Кузнецов Г.В., Хорошко В.О., Чирков Д.В. / За ред. проф. В.О. Хорошка. – К.: ДУІКТ, 2007. – 365 с.

3. Макс Ронге. Разведка и контрразведка / М. Ронге /. – К.: СИНТО, 1993. – 239 с.

4. Мухачев В.А. Методы практической криптографии / Мухачев В.А., Хорошко В.А./ . – К.: ПолиграфКонсалтинг, 2005. – 214 с.

5. Мицан И.Б. Стеганографические методы сокрытия информации / Мицан И.Б. // Специальная техника и вооружение. Научно-методическое издание. – К., № 1 – 5, 2001. – С. 28 – 32.

6. Хорошко В.О. Основи комп'ютерної стеганографії. Навчальний посібник / В.О. Хорошко, О.Д. Азаров, М.Є. Шелест, Ю.Є. Яремчик /. – Вінниця: ВДТУ, 2003. – 143 с.

7. Конахович Г.Ф. Компьютерная стеганография. Теория и практика / Конахович Г.Ф., Пузыренко А.Ю. /. – К.: «МК-Пресс», 2006. – 288 с.

8. Безопасность информационных технологий. Методология создания систем защиты/ В.В. Домарев. – К.: ООО "ТИД "ДС", 2001. – 688 с.

9. Энциклопедия промышленного шпионажа/ Под общ. ред. Е.В. Куренкова – С.Петербург: ООО "Изд-во Полигон", 1999. – 512 с.

10. Хорев А;А. Способы и средства защиты информации. – М.: МО РФ, 2000. -316 с.

11. А.Ю.Щербаков. Введение в теорию и практику компьютерной безопасности. -М.: издатель Молгачева С.В., 2001.

12. Бармен, Скотт. Разработка правил информационной безопасности.: Пер. с англ. – М.: Издательский дом "Вильямс", 2002.

13. С.Л.Емельянов Основы информационной безопасности.– Одесса: Юридична література, 2003.-198с.

14. Про державну таємницю. Закон України №3855-XII від 21.01.1994 р. (в редакції Закону № 1079-14 від 21.09.99).

15. Про затвердження зводу відомостей, що становлять державну таємницю. Наказ Голови Служби безпеки України від 12.08.2005 р. № 440.

16. НД ТЗІ 1.1 – 002 – 99. Общие положения по защите информации в компьютерных системах от несанкционированного доступа. Нормативний документ ДСТЗИ СБ Украйни. Киев, 1999.

17. Про інформацію. Закон України № 2657-ХП від 02.10.92 р.

18. Концепція технічного захисту інформації в Україні. Постанова КМУ №1126 8.10.97.

19. Положення про технічний захист інформації в Україні. Указ Президента України №1229/99 від 27.09.99.

20. Тимчасові рекомендації з технічного захисту інформації вид витоку каналами побічних електромагнітних випромінювань і наводок. (ТР ТЗІ-ПЕМВН-95). Затверджені наказом ДСТЗІ від 09.06.95р. № 25.

21. НД ТЗІ 1.4-001-2000. Типове положення про службу захисту інформації в автоматизованій системі.

22. НД ТЗІ 2.5-005-99. Класифікація автоматизованих систем і стандартні функціональні профілі захищеності оброблюваної інформації від несанкціонованого доступу.

23. НД ТЗІ 2.1-001-2001. Створення комплексів технічного захисту інформації. Атестація комплексів. Основні положення.

24. НД ТЗІ 3.7-001-99. Методичні вказівки щодо розробки технічного завдання на створення комплексної системи захисту інформації в автоматизованій системі (Зі зміною № 1, затвердженою наказом ДСТСЗІСБУ 18.06.02 № 37).

25. НД ТЗІ 2.5-010-2003. Вимоги до захисту інформації \№ЕВ-сторінки від несанкціонованого доступу.

26. 13.ГОСТ Р 51275-99. Защита информации. Объект информатизации. Факторы, воздействующие на информацию. Общие положения.

27. http://www.intuit.ru/department/security/secbasics/

28. Галатенко В.А. Основы информационной безопасности Интернет-университет информационных технологий – ИНТУИТ.ру, 2008

29. Лапонина О.Р. Основы сетевой безопасности: криптографические алгоритмы и протоколы взаимодействия Интернет-университет информационных технологий – ИНТУИТ.ру, 2005

30. Галатенко В.А. Стандарты информационной безопасности Интернет-университет информационных технологий – ИНТУИТ.ру, 2005

31. Э. Мэйволд Безопасность сетей Эком, 2006

32. Хаулет Т. Защитные средства с открытыми исходными текстами БИНОМ. Лаборатория знаний, Интернет-университет информационных технологий – ИНТУИТ.ру, 2007

33. Department of Defense Trusted Computer System Evaluation Criteria DoD 5200.28-STD, 1993.

34. Information Technology Security Evaluation Criteria (ITSEC). Harmonized Criteria of France – Germany – the Netherlands – the United Kingdom Department of Trade and Industry, London, 1991.

35. Security Architecture for Open Systems Interconnection for CCITT Applications. Recommendation X.800 CCITT, Geneva, 1991.

36. Site Security Handbook. Holbrook P., Reynolds J., Request for Comments: 1244, 1991.

37. James Nechvatal, Elaine Barker, Lawrence Bassham, William Burr, Morris Dworkin, James Foti, Edward Roback Report on the Development of the Advanced Encryption Standard (AES) Computer Security Division Information Technology Laboratory National Institute of Standards and Technology Technology Administration U.S. Department of Commerce. 2000г. 116с.

38. Государственный Стандарт Российской Федерации Информационная технология. Криптографическая защита информации. Процедуры выработки и

проверки электронной цифровой подписи на базе асимметричного криптографического алгоритма 1994г.

39. Государственный Стандарт Российской Федерации Информационная технология. Криптографическая защита информации. Функция хэширования 1994г.

40. RFC 3280 Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile 2002г. 129с.

41. RFC 3281 An Internet Attribute Certificate Profile for Authorization 2002г. 40с.

42. RFC 2510 Internet X.509 Public Key Infrastructure Certificate Management Protocols 1999г. 72с.

43. RFC 2511 Internet X.509 Certificate Request Message Format 1999г. 25с.

44. RFC 3369 Cryptographic Message Syntax 2002г. 60с.

45. RFC 2560 X.509 Internet Public Key Infrastructure Online Certificate Status Protocol – OCSP 1999г. 23с.

46. RFC 2797 Certificate Management Messages over CMS 2000г. 47с.

47. RFC 3379 Delegated Path Validation and Delegated Path Discovery Protocol Requirements 2002г. 15с.

48. RFC 2633 S/MIME Version 3 Message Specification 1999г. 32с.

49. RFC 2632 S/MIME Version 3 Certificate Handling 1999г. 13с.

50. Security Architecture for the Internet Protocol RFC 2401 1998г. 66с.

51. Internet Security Association and Key Management Protocol (ISAKMP) RFC 2408 1998г. 86с.

52. The Internet Key Exchange (IKE) RFC 2409 1998г. 41с.

53. RFC 2412 The OAKLEY Key Determination Protocol 1998г. 55с.

Інформаційні ресурси

Бібліотеки, Інтернет, електронні книги.