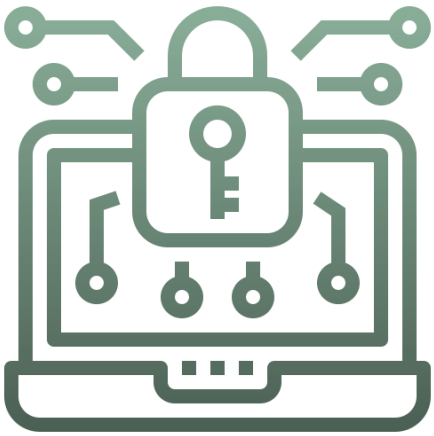
МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»

Н.О. ЩУР, О.А. ПОКОТИЛО

ОСНОВИ КРИПТОЛОГІЇ

Навчальний посібник



Житомир
2021

УДК 004.056.55
Щ98

Рекомендовано до друку вченою радою
державного університету «Житомирська політехніка»
(протокол № 9 від 29 грудня 2021 року)

Рецензенти:

І.А. Пількевич – доктор технічних наук, професор, професор кафедри комп'ютерних інформаційних технологій Житомирського військового інституту імені С.П. Корольова

К.В. Молодецька – доктор технічних наук, професор, професор кафедри комп'ютерних технологій і моделювання систем Поліського національного університету

А.А. Єфіменко – кандидат технічних наук, доцент, завідувач кафедри комп'ютерної інженерії та кібербезпеки Державного університету «Житомирська політехніка»

Н.О. Щур

Щ98 Щур Н.О., Покотило О.А. Основи криптології: навч. посібник. – Житомир: Державний університет «Житомирська політехніка», 2021. 120 с.

ISBN 978-966-683-597-3

У посібнику висвітлено основні поняття криптології, наведено класифікацію шифрів, описано класичні криптосистеми та методи їх криптоаналізу, розглядаються основні алгоритми симетричної та асиметричної криптографії, алгоритми хешування та цифрового підпису, представлено елементи криптографії на еліптичних кривих.

Посібник призначений для здобувачів вищої освіти в галузі 12 «Інформаційні технології» за спеціальністю 125 «Кібербезпека».

ISBN 978-966-683-597-3 **УДК 004.056.55**

ЗМІСТ

ПЕРЕДМОВА

Однією з найважливіших умов успішного функціонування будь-якої інформаційної системи є захист її даних від несанкціонованого доступу. На сьогоднішній постає стратегічно важливе питання якості підготовки закладами вищої освіти майбутніх фахівців з кібербезпеки, які б у своїй діяльності ефективно використовували різноманітні методи захисту інформації, зокрема криптографічні. Криптографія займається розробкою алгоритмів перетворення повідомлень, в тому числі шляхом шифрування з використанням спеціальних (ключових) даних. Дослідженням вразливих місць таких алгоритмів та розробкою методів зламу зашифрованих повідомлень займається криптоаналіз. Ці два наукових напрями тісно пов’язаних між собою і разом складають науку криптологію.

Навчальний посібник «Основи криптології» спрямований на ознайомлення студентів із загальними принципами побудови систем криптографічного захисту даних шляхом використанням сучасних симетричних та асиметричних алгоритмів. Структуру та зміст навчального посібника визначено, виходячи зі змісту робочої програми навчальної дисципліни «Прикладна криптологія», її змістовних модулів, тем навчальних занять, теоретичного та практичного досвіду авторів.

Посібник містить три розділи, кожен з яких доповнений контрольними питаннями, тестовими завданнями та практичними задачами для самостійного опрацювання. У першому розділі висвітлено основні поняття та визначення криптології, розглянуто класифікацію алгоритмів шифрування, описано класичні криптосистеми та методи їх криптоаналізу. Другий розділ присвячено симетричним потоковим та блоковим алгоритмам шифрування, представлені режими застосування блокових шифрів. У третьому розділі розглядаються асиметричні алгоритми шифрування, алгоритми обміну ключами, функції хешування, схеми створення цифрового підпису, наводяться основні положення криптографії на еліптичних кривих.

Посібник призначений для здобувачів вищої освіти в галузі 12 «Інформаційні технології» за спеціальністю 125 «Кібербезпека».

РОЗДІЛ 1. ОСНОВНІ ПОНЯТТЯ ТА ВИЗНАЧЕННЯ. КЛАСИЧНІ ШИФРИ ТА ЇХ КРИПТОАНАЛІЗ

1.1. ОСНОВНІ ПОНЯТТЯ КРИПТОЛОГІЇ

Сучасні інформаційно-комунікаційні технології інтенсивно впроваджуються в усі сфери людського життя. Інформаційні ресурси стають головною цінністю наукового, економічного та технічного розвитку будь-якої галузі як в Україні, так і у світі. При цьому великого значення набуває проблема захисту даних, що полягає у забезпеченні їх конфіденційності, цілісності та достовірності при зберіганні, обробці та передачі.

Забезпечення *конфіденційності* полягає у вирішення проблеми захисту інформаційних ресурсів від несанкціонованого ознайомлення з їх змістом. Залежно від контексту замість терміна «конфіденційні» дані можуть виступати терміни «секретні», «приватні», «обмеженого доступу».

Цілісність інформаційних ресурсів полягає у гарантуванні неможливості їх несанкціонованої зміни (модифікації).

Доступність ресурсу полягає у можливості його використання за вимогою користувача, який має відповідні повноваження.

Криптологія (грецьк. «таємний» та «слово, вчення») – наука, яка вивчає методи побудови та аналізу систем захисту інформаційних ресурсів, основаних на математичних перетвореннях даних з використанням секретних параметрів. Криптологія поєднує у собі два взаємозалежні напрями: криптографію та криптоаналіз. Фундамент криптології як науки у 1949 р. заклала робота американського вченого Клода Шеннона «Теорія зв'язку в секретних системах», у якій фактично вперше було представлено математичну модель шифрів.

Криптографія (грецьк. «таємний» та «писання», «тайнопис») –

- наука про принципи, засоби та методи перетворення даних з метою приховування їх змісту, запобігання несанкціонованого використання або підробки;
- наука, що вивчає математичні методи, пов'язані з такими функціями захисту даних, як конфіденційність, цілісність та автентичність;

🔒 напрям у криптології, що вивчає основні закономірності, протиріччя, методи, системи та засоби забезпечення конфіденційності, цілісності, дійсності, доступності та спостережливості інформації та ресурсів тощо, ґрунтуючись на криптографічних перетвореннях.

Криптоаналіз (грецьк. «таємний» та «аналіз») –

🔓 наука про методи та способи розкриття зашифрованих повідомлень, а також про тактику та стратегію їх застосування;

🔓 наука про математичні методи порушення безпеки криптографічних систем;

🔓 напрям у криптології, що вивчає основні закономірності, протиріччя, методи та засоби аналізу криптографічних систем, ґрунтуючись на їх вхідних та вихідних даних, у тому числі можливо на частині ключових даних, що здійснюється з метою визначення спеціальних (ключових) даних і значущої інформації, які можуть бути використані для порушення конфіденційності, цілісності, справжності, доступності, неспростовності (спостережливості) інформації та ресурсів тощо.

Кожне *криптографічне перетворення* однозначно визначається ключем (секретним параметром) та описується криптографічним алгоритмом. **Криптографічний алгоритм** являє собою набір математичних правил та процедур, який описує такі види перетворень, як шифрування, формування та перевірка цифрового підпису, обчислення хеш-значень, спеціальних криптографічних контрольних сум тощо. Сукупність криптографічних алгоритмів, що використовуються для шифрування називають **шифром**.

Криптографічний ключ (ключ) –

🗝 секретний змінний елемент шифру, що застосовується для шифрування конкретного повідомлення;

🗝 таємний стан деяких параметрів алгоритму криптографічного перетворення, що забезпечує вибір одного перетворення із сукупності можливих для даного алгоритму.

Шифрування даних – це процес, що складається із:

1) *Зашифрування* – процес перетворення відкритого тексту до виду, незрозумілого несанкціонованому користувачеві. *Відкритий текст* являє собою вихідне повідомлення, що підлягає зашифруванню. Результатом зашифрування відкритого тексту є *шифротекст*, що також називають *криптотекстом* або *криптограмою*.

2) *Розшифрування (*син. *дешифрування)* – процес обернений до зашифрування, тобто перетворення шифрованого повідомлення до початкової інформації (відкритого тексту) за допомогою певних правил шифру та відомого ключа.

Криптосистема – це система криптографічного перетворення даних, що містить у собі п’ять компонентів: множину відкритих текстів, множину шифротекстів, множину ключів, сімейство зашифровуючих та розшифровуючих перетворень. Фахівець, який займається розробкою криптосистем називається *криптографом*.

Криптостійкість – це властивість криптосистеми протидіяти атакам супротивника, спрямованим на отримання секретного ключа або відкритого повідомлення. Стійкість криптосистеми визначається її здатністю протидіяти усім можливим атакам. Під *атакою на криптосистему* розуміється спроба порушення безпеки конкретної реалізації криптосистеми. Вдалу криптоатаку називають *зламом*. В криптографії існує загальноприйняте правило, яке сформулював голландський вчений Огюст Керкхоф (Auguste Kerckhoffs): *стійкість зашифрованого повідомлення забезпечується в першу чергу ключем*. Тобто передбачається ймовірність того, що сам алгоритм шифрування, шифротекст або якась його частина є відомими зловмиснику та доступні для вивчення.

Криптостійкість часто вимірюється кількістю операцій, необхідних для перебору всіх можливих ключів, або інтервалом часу, необхідного для зламу. Вона оцінюється у процесі проведення криптографічного аналізу. Фахівець який займається розробкою методів криптоаналізу називається *криптоаналітиком*. Синонімами є терміни зловмисник, порушник, супротивник.

Відношення між описаними вище термінами можна представити у вигляді схеми обміну секретними повідомленнями (рис. 1.1).

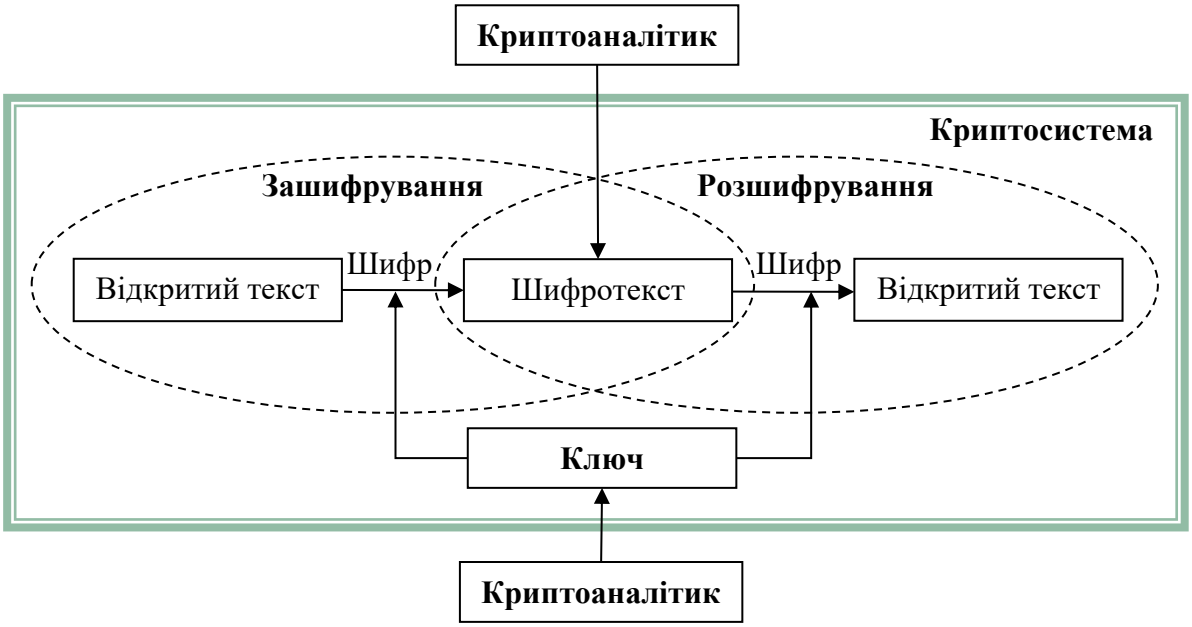


Рис. 1.1. Схема обміну секретними повідомленнями

Розглянемо основні принципи криптології.

Принцип рівної міцності захисту. На шляху від одного законного власника до іншого інформація може захищатись різними способами в залежності від загроз, що виникають. Так утворюється ланцюг захисту інформації з ланками різного типу. Противник прагне знайти найслабкішу ланку, щоб з найменшими витратами дістатися до даних. Законні власники повинні враховувати це у своїй стратегії захисту інформації криптографічними методами, не варто робити якусь ланку дуже міцною, якщо є слабкіші ланки.

Принцип доцільності захисту. На сучасному рівні технічного розвитку засоби зв'язку, засоби перехвату повідомлень, а також засоби захисту інформації вимагають занадто великих витрат. Тому існує проблема співвідношення вартості інформації, витрат на її захист та витрат на її здобування. Перш ніж захищати інформацію криптографічними методами, треба вирішити два питання: «чи отримає противник внаслідок атаки інформацію, що буде більш цінною, ніж вартість самої атаки?» та «чи є інформація, яку захищає її власник, більш цінною, ніж вартість захисту?». Відповідь на ці два питання визначає вибір відповідних засобів криптографічного захисту.

Принцип використання ключа. Розробка хорошого шифру – справа надзвичайно трудомістка. Тому бажано збільшити термін життя цього шифру і використовувати його для шифрування якнайбільшої кількості повідомлень. Але при цьому виникає небезпека, що противник вже зламав шифр і вільно читає шифровані повідомлення. Саме тому в сучасних шифрах використовують ключі. Знання ключа дозволяє швидко та просто відновити початковий текст. Без знання ключа дешифрування тексту має бути практично недосяжним.

Принцип стійкості шифру. Здатність шифру протидіяти різноманітним атакам на нього називається стійкістю шифру. З математичної точки зору проблема отримання строго доведених оцінок стійкості для будь-якого шифру ще не вирішена. Ця проблема відноситься до проблем нижніх оцінок обчислювальної складності задачі, ще нерозв'язаних математично. Тому стійкість конкретного шифру оцінюється шляхом різноманітних спроб його зламування, а отримані результати оцінюють в залежності від кваліфікації криптоаналітиків, які атакують цей шифр. Таку процедуру називають перевіркою стійкості.

Принцип Керкхофа. Стійкість сучасного шифру має визначатись, в першу чергу, ключем. Зміст цього принципу полягає в тому, що захищеність інформації не повинна залежати від таких факторів, які важко змінити при появі загрози. При використанні ключів законним власникам інформації легше перешкоджати противнику, оскільки міняти їх можна досить часто. Щоправда, тепер перед законними власниками виникає інша задача – як таємно обмінятись ключами перед тим, як обмінюватись шифрованими повідомленнями.

Принцип використання різноманітних шифрів. Не існує єдиного шифру, що підходить для всіх випадків. Вибір шифру залежить від особливостей інформації (може мати різний характер, тобто бути документальною, телефонною, телевізійною тощо), від цінності інформації, від обсягів інформації, від потрібної швидкості її передачі, від тривалості захисту (державні та військові таємниці зберігаються десятками років, біржеві – декілька годин), від можливостей противника (можна протидіяти окремій особі, можна протидіяти

потужній державній структурі), а також від можливостей власників із захисту своєї інформації.

З часом задачі криптології значно розширилися та вийшли за межі шифрування повідомлень. На сьогоднішній день вони також включають розробку систем цифрового підпису, протоколів автентифікації та ідентифікації користувачів тощо.

Криптографічні системи та шифри класифікуються за різними ознаками.

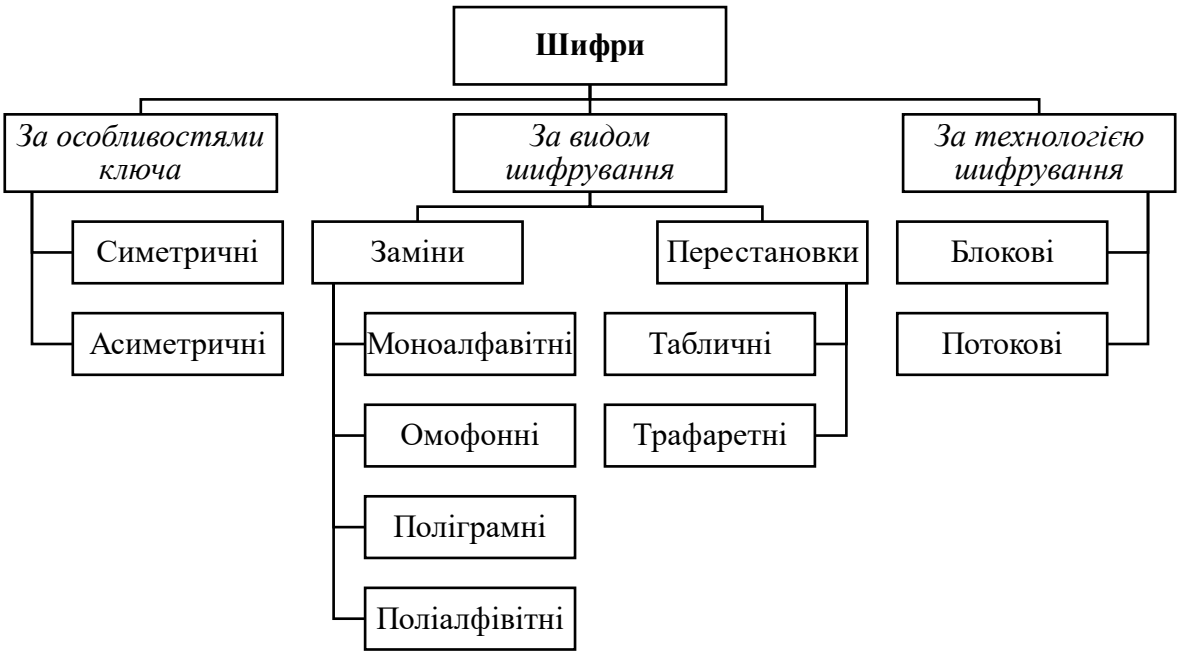


Рис. 1.2. Класифікація криптосистем

За особливостями ключа розрізняють:

✓ **Симетричні криптосистеми (із закритим ключем, одноключові)** – криптосистема, у якій один і той самий алгоритм, а також один і той самий ключ використовується для шифрування та дешифрування повідомлень (рис. 1.3);

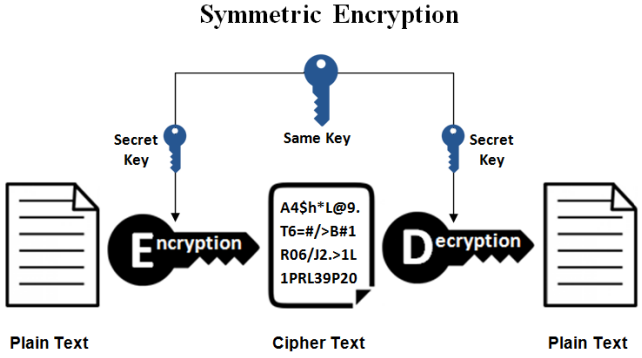


Рис. 1.3. Схема роботи симетричної криптосистеми

✓ **Асиметричні криптосистеми (із відкритим ключем, двоключові)** – криптосистема, у якій використовуються два ключі – відкритий (публічний) і закритий (секретний), які математично пов'язані один з одним. Повідомлення зашифровується за допомогою відкритого ключа, що доступний усім бажаючим, а розшифровується за допомогою закритого ключа, відомого тільки одержувачу (рис. 1.4).

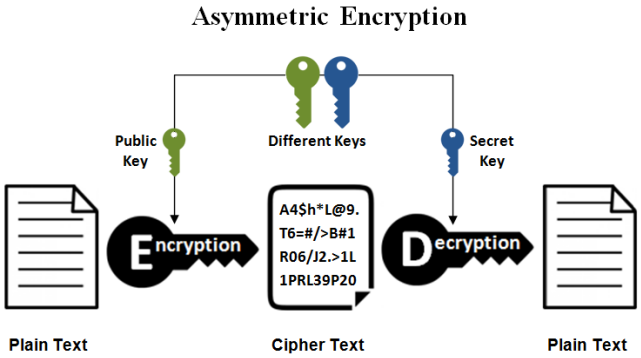


Рис. 1.4. Схема роботи асиметричної криптосистеми

Залежно від виду криптографічного перетворення криптосистеми можуть містити:

✓ **Шифр підстановки (заміни)** – це шифр, у якому кожен символ відкритого тексту у шифротексті замінюється іншим символом. Найчастіше виділяють такі типи шифрів підстановки:

- проста підстановка, або моноалфавітна заміна – це шифр, який кожен символ відкритого тексту замінюється відповідним символом шифротексту, при чому, конкретній літері відкритого повідомлення відповідає єдина, завжди одна і та сама, літера шифротексту;
- однозвучний шифр підстановки схожий на простий шифр підстановки за винятком того, що один символ відкритого тексту символ відкритого тексту замінюється на один з декількох можливих символів шифротексту;
- поліграмний шифр підстановки – це шифр, який блоки символів шифрує по групах, наприклад, біграма – це група з двох символів, триграма – з трьох символів і т.д.;
- поліалфавітна підстановка складається з декількох простих шифрів підстановки, тобто одна і та сама літера відкритого тексту може бути замінена

кожен раз по різному (відбувається циклічне застосування декількох моноалфавітних шифрів);

✓ **Шифр перестановки** – це шифр, у якому символи повідомлення переставляються місцями безпосередньо у відкритому тексті за певним правилом, що залежить від ключа. Найчастіше перестановка виконується за допомогою *таблиці*, комірки якої спочатку заповнюють відкритим текстом в деякому порядку, а потім шифротекст зчитують відповідно до заздалегідь визначеного алгоритму. В історичному плані також цікавим є *трафаретний шифр*, у якому для шифрування використовувався трафарет з прорізаними комірками. Приклавши трафарет до аркушу паперу, в прорізаних комірках записували повідомлення, а решту аркуша заповнювали довільними символами.

За технологією шифрування розрізняють:

✓ **Блокові шифри** здійснюють шифрування блоків фіксованої довжини, що складаються з послідовності символів відкритого тексту;

✓ **Потокові шифри** здійснюють шифрування окремих символів відкритого тексту.

1.2. КЛАСИЧНІ АЛГОРИТМИ ШИФРУВАННЯ

1.2.1. ШИФР ЦЕЗАРЯ

Розглянемо один з найдавніших та найбільш поширених шифрів простої (моноалфавітної) заміни – шифр Цезаря, названий на честь римського імператора *Гая Юлія Цезаря*. У цьому шифрі кожна літера повідомлення зсувається в алфавіті на *K* позицій вперед від символу, що замінюється. При досягненні кінця алфавіту виконується циклічний перехід до його початку. При необхідності розділові знаки та пробіли ігноруються. Таким чином, наприклад, літерам алфавіту відповідатимуть числові позиції (табл. 1.1, табл. 1.2):

Інформаційна безпека – це стан захищеності систем обробки і зберігання даних, при якому забезпечено конфіденційність, доступність і цілісність інформації.

Таблиця 1.1. Нумерація позицій літер англійського алфавіту

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25

Таблиця 1.2. Нумерація позицій літер українського алфавіту

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
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32

Ключем шифрування є деяке фіксоване секретне число *K* – від 1 до 25 для англійського (латинського) алфавіту та *K* – від 1 до 32 для українського. При дешифруванні літера зашифрованого тексту замінюється на літеру розташовану в алфавіті на *K* позицій назад.

Приклад 1.1:

Відомо, що Цезар для шифрування використовував ключ *K*=3, тобто відбувався зсув символів повідомлення на три позиції вперед у латинському алфавіті (рис. 1.5). Отже, повідомлення римського імператора *ALEA JACTA EST* (Жереб кинутий) після зашифрування буде мати вигляд *DOHDMDFWDHVW*.

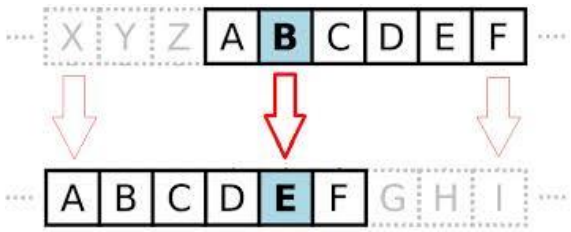


Рис. 1.5. Заміна символів повідомлення у шифрі Цезаря з ключем K=3

Зазначимо, що цей алгоритм шифрування, на сьогоднішній день, являється нестійким до зламу і не використовується на практиці, проте є важливим для вивчення. Оскільки відомо, що навіть дуже складні сучасні криптосистеми в якості типових складових використовують прості шифри заміни.

1.2.2. ШИФР ЧАСТОКОЛУ

Шифр частоколу є представником шифрів перестановки. Ключем є ціле число *K* – висота частоколу. При шифруванні літери повідомлення записуються як степені (їх кількість висота частоколу), а потім отримані літери в степенях переписуються по рядках зверху донизу.

Приклад 1.2:

Зашифруємо повідомлення Я ОТРИМАЮ ІСПИТ АВТОМАТОМ із ключем *K*=3. Записавши по три літери зверху догори, а потім зверху до низу по рядкам, отримаємо шифротекст ТМІИВМООИЮПАОТЯРАСТТАМ (рис. 1.6).

Для розшифрування, потрібно підрахувати кількість літер шифротексту, поділити їх на ключ та записати літери по *K* штук (*K* – ключ) в порядку зверху донизу.

Т	М	І	И	В	М	О
О	И	Ю	П	А	О	Т
Я	Р	А	С	Т	Т	А	М

Рис. 1.6. Шифрування за алгоритмом частоколу

1.2.3. ШИФР ПЛЕЙФЕРА

Шифр Плейфера є біграмним, тобто текст повідомлення розбивається на біграми (групи з двох символів). Таким чином, шифр Плейфера є більш стійкий до зламу у порівнянні із шифром Цезаря, так як ускладнюється його частотний аналіз. Він може бути проведений, але не для 26 можливих символів (англійський алфавіт), а для 26x26 = 676 можливих біграм.

Для шифрування шифр Плейфера використовує матрицю 5x5 (для англійського алфавіту), яка містить ключове слово або фразу. Щоб скласти ключову матрицю, в першу чергу потрібно заповнити порожні клітинки матриці літерами ключового слова (виключаючи літери, що повторюються), потім заповнити клітинки, що лишилися символами алфавіту, що не зустрічаються в ключовому слові, по порядку (рис. 1.7). В англійських текстах зазвичай пропускається символ «Q», щоб зменшити алфавіт, в інших версіях «I» і «J» об’єднуються в одну клітинку.

P	L	A	Y	F
I	R	E	X	M
B	C	D	G	H
J	K	N	O	S
T	U	V	W	Z

Рис. 1.7. Матриця шифру Плейфера

Для того щоб зашифрувати повідомлення, необхідно розбити його на біграми (групи з двох символів) та відшукати ці біграми в матриці. Якщо кількість літер повідомлення непарна, то для формування останньої біграми додається «X».

Два символи біграми відповідають кутам прямокутника в ключовий матриці. Визначаємо положення кутів цього прямокутника відносно один одного. Потім, керуючись наступними 4 правилами, зашифровувати пари символів вихідного тексту.

Правила шифрування біграм

1. Якщо дві літери біграми однакові – додаємо після першого символу «X».
2. Якщо літери біграми знаходяться в різних стовпцях і різних рядках – замінюємо їх на літери, що знаходяться в тих самих рядках (стовпцях), але відповідно в інших кутах прямокутника.
3. Якщо літери біграми зустрічаються в одному рядку – замінюємо їх на літери, розташовані в найближчих стовпцях праворуч від відповідних літер. Якщо літера остання у рядку, то вона замінюється на перший символ цього ж рядка.
4. Якщо літери біграми зустрічаються в одному стовпці – перетворюємо їх в літери того ж стовпця, що знаходяться безпосередньо під ними. Якщо літера є нижньою в стовпці – вона замінюється на першу літеру цього ж стовпчика.

Приклад 1.3:

Зашифруємо повідомлення HIDE THE GOLD IN THE TREE STUMP із використанням ключової фрази PLAYFAIR EXAMPLE. Матрицею шифрування буде матриця описана вище (рис. 1.7).

Для шифрування розіб’ємо текст на біграми HI DE TH EG OL DI NT HE TR EX ES TU MP. Знайдемо літери першої біграми у матриці та замінимо їх на літери, що знаходяться у протилежних кутах прямокутника (рис. 1.8).

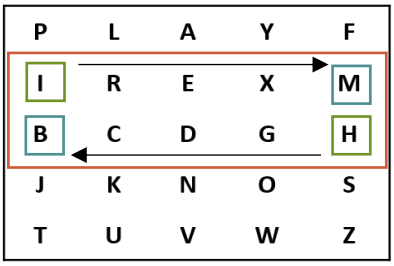


Рис. 1.8. Шифрування біграм

Далі, користуючись правилами шифрування біграм, отримаємо шифротекст: BM ND ZB XD KY BE JV DM UI XM MN UV IF.

1.2.4. КРИПТОСИСТЕМА ХІЛЛА

У 1929 році американський математик *Лестер Хілл* придумав новий поліграмний шифр заміни, в якому використовувалися як модульна арифметика, так і лінійна алгебра.

Ключем шифру є квадратна матриця $K(n \times n)$, елементи якої числа від 0 до 25, $\det K \neq 0$, $n \geq 2$. Літери алфавіту нумеруються в порядку їхнього зростання від 0 до 25. При шифруванні відкритий текст розбивається на блоки з n літер, числові значення яких розглядаються як вектор розмірності n. Кожен вектор множиться на матрицю шифрування $K(n \times n)$ по модулю 26 (для англійського алфавіту).

Приклад 1.4:

Повідомлення *HELP* зашифруємо за допомогою ключової матриці:

$$K = \begin{pmatrix} 3 & 3 \\ 2 & 5 \end{pmatrix},\ det\, K = 15 - 6 = 9 \neq 0.$$

Розіб'ємо відкритий тест на вектори розмірністю 2, літерам поставимо у відповідність їх числові значення:

$$P_1 = \begin{pmatrix} H \\ E \end{pmatrix} = \begin{pmatrix} 7 \\ 4 \end{pmatrix} \qquad P_2 = \begin{pmatrix} L \\ P \end{pmatrix} = \begin{pmatrix} 11 \\ 15 \end{pmatrix}.$$

Помножимо ключову матрицю на кожен вектор відкритого тексту та отримаємо шифротекст *HIAT*:

$$K \cdot P_1 = \begin{pmatrix} 3 & 3 \\ 2 & 5 \end{pmatrix} \cdot \begin{pmatrix} 7 \\ 4 \end{pmatrix} = \begin{pmatrix} 33 \\ 34 \end{pmatrix} mod\ 26 = \begin{pmatrix} 7 \\ 8 \end{pmatrix} = HI;$$

$$K \cdot P_2 = \begin{pmatrix} 3 & 3 \\ 2 & 5 \end{pmatrix} \cdot \begin{pmatrix} 11 \\ 15 \end{pmatrix} = \begin{pmatrix} 78 \\ 97 \end{pmatrix} mod\ 26 = \begin{pmatrix} 0 \\ 19 \end{pmatrix} = AT.$$

Для того щоб дешифрувати повідомлення, кожен блок шифротексту з n літер множиться на обернену (за модулем 26) матрицю до матриці шифрування.

Шифротекст *HIAT* дешифруємо за допомогою матриці оберненої до ключової: $K^{-1} = \begin{pmatrix} 15 & 17 \\ 20 & 9 \end{pmatrix}$ та отримаємо повідомлення *HELP*.

$$K^{-1} \cdot P_1 = \begin{pmatrix} 15 & 17 \\ 20 & 9 \end{pmatrix} \cdot \begin{pmatrix} 7 \\ 8 \end{pmatrix} = \begin{pmatrix} 241 \\ 212 \end{pmatrix} mod\ 26 = \begin{pmatrix} 7 \\ 4 \end{pmatrix} = HE;$$

$$K^{-1} \cdot P_2 = \begin{pmatrix} 15 & 17 \\ 20 & 9 \end{pmatrix} \cdot \begin{pmatrix} 0 \\ 19 \end{pmatrix} = \begin{pmatrix} 323 \\ 171 \end{pmatrix} mod\ 26 = \begin{pmatrix} 11 \\ 15 \end{pmatrix} = LP.$$

1.2.5. ШИФР ВІЖЕНЕРА

На протязі століть використання простого моноалфавітного шифру заміни було достатнім, щоб забезпечити таємність. Подальший розвиток частотного криптоаналізу, спочатку арабами, а потім і в Європі, зруйнував його стійкість. Таким чином криптографи мали придумати новий, більш стійкий шифр. Вчений епохи Відродження *Леона Батіста Альберті* вперше запропонував замість одного секретного алфавіту, використовувати два або більше, послідовно або циклічно змінюючи їх за певним правилом. Ґрунтуючись на ідеях попередника, свій шифр створив французький посол в Римі *Блез де Віженер*.

Шифр Віженера складається з послідовності декількох шифрів Цезаря з різними значеннями зсуву, що визначаються літерами ключового слова. Кожна літера відкритого тексту зсувається вперед на позицію відповідної літери ключа. Якщо ключове слово менше за повідомлення, то воно циклічно повторюється.

Приклад 1.5:

Повідомлення *ATTACK AT DAWN* зашифруємо ключем *LEMON*. В результаті чого отримаємо шифротекст *LXFOPVEFRNHR*.

	A	T	T	A	C	K	A	T	D	A	W	N
	L	E	M	O	N	L	E	M	O	N	L	E
	0	19	19	0	2	10	0	19	3	0	22	13
+	11	4	12	14	13	11	4	12	14	13	11	4
	11	23	5	14	15	21	4	5	17	13	7	17
	L	X	F	O	P	V	E	F	R	N	H	R

Для зашифрування може використовуватися й таблиця Віженера (таб.1.3).

Таблиця 1.3. Таблиця Віженера

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
B	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
C	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
D	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
E	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
F	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
G	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
H	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
I	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
J	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
K	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
L	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
M	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
N	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
O	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
P	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
R	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
S	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
T	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
U	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
V	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
W	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
X	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
Y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
Z	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y

У загальному випадку таблиця Віженера складається з алфавіту, циклічно зміщеного на один символ ліворуч. Під час зашифрування кожна літера повідомлення замінюється на літеру, що знаходиться на перетині літер першого рядка (алфавіт повідомлення) і першого стовпчика (алфавіт ключа) в таблиці Віженера.

При дешифруванні потрібно відшукати у першому стовпчику літеру ключа і за літерами шифротексту визначити, в якому стовпчику зверху знаходиться літера відкритого тексту.

Приклад 1.6:

Повідомлення *PURPLE*, зашифроване ключем *SMART* за допомогою таблиці Віженера (табл. 1.4), перетвориться у шифротекст *HGRGEW*.

При дешифруванні потрібно відшукати у першому стовпчику літеру ключа і за літерами шифротексту визначити, в якому стовпчику зверху знаходиться літера відкритого тексту.

Таблиця 1.4. Шифрування повідомлення за таблицею Віженера

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
B	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
C	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
D	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
E	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
F	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
G	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
H	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
I	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
J	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
K	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
L	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
M	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
N	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
O	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
P	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
R	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
S	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
T	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
U	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
V	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
W	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
X	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
Y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
Z	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y

1.3. ОСНОВИ КРИПТОАНАЛІЗУ КЛАСИЧНИХ ШИФРІВ

1.3.1. ЧАСТОТНИЙ КРИПТОАНАЛІЗ

Криптоаналіз шифру Цезаря ґрунтується на *частотному аналізі* появи окремих символів природньої мови у тексті. Частота символу у повідомленні дорівнює кількості його появи у тексті, поділеній на загальну кількість літер тексту. Для кожної мови справедливо наступне: у досить довгих текстах кожна

літера зустрічається із приблизно однаковою частотою, залежно від самої літери і незалежно від конкретного тексту. Тобто імовірність появи окремих літер, а також їх порядок у словах і фразах природньої мови підпорядковуються статистичним закономірностям. Так, наприклад, відомо, що в українській та англійській мовах частоти появи літер розподілені наступним чином (табл. 1.5).

Отже, літера з найбільшою частотою в шифротексті буде замінюватися на літеру з найбільшою частотою у мові. А кількість позицій між ними буде визначати довжину ключа. Однак, якщо текст не дуже великий, то закономірності будь-якої природньої мови можуть проявлятися в ньому не обов'язково в строгій відповідності з таблицею частот. В такому випадку розглядається відношення наступної літери за частотою появи у зашифрованому тексті та найчастішою літерою мови.

Таблиця 1.5. Частоти появи літер в українській та англійській мовах

Українська мова						**Англійська мова**					
А	0,072	**Ї**	0,006	**У**	0,04	**A**	0,082	**J**	0,002	**S**	0,063
Б	0,017	**Й**	0,008	**Ф**	0,001	**B**	0,015	**K**	0,008	**T**	0,091
В	0,052	**К**	0,035	**Х**	0,012	**C**	0,028	**L**	0,040	**U**	0,028
Г, Ґ	0,016	**Л**	0,036	**Ц**	0,006	**D**	0,043	**M**	0,024	**V**	0,010
Д	0,035	**М**	0,031	**Ч**	0,018	**E**	**0,127**	**N**	0,067	**W**	0,023
Е	0,017	**Н**	0,065	**Ш**	0,012	**F**	0,022	**O**	0,075	**X**	0,001
Є	0,008	**О**	**0,094**	**Щ**	0,001	**G**	0,020	**P**	0,019	**Y**	0,020
Ж	0,009	**П**	0,029	**Ь**	0,029	**H**	0,061	**Q**	0,001	**Z**	0,001
З	0,023	**Р**	0,047	**Ю**	0,004	**I**	0,070	**R**	0,0060		
И	0,061	**С**	0,041	**Я**	0,029						
І	0,057	**Т**	0,055								

Приклад 1.7:

Дано текст, зашифрований за допомогою шифру моноалфавітної заміни:
ДАФИНЦШЕИЮЯЗЩШФЬИТЧИВЮЯШХСЯЯЗВИЇШЧШЮФЬСПЄСПИЇОЛ
РПЧИЦРЗФЬРІИЇЇШЛСЯИФСЦРЇЧЄШЦАСЇШЧШІСХЗЧИЮДАФИНЧИЮЮ
ИЦРВЧМЦИУШЙШЧСЛМЇСЛЯШЙШЕШЇШЧШЮФЬСПЄ

При зашифруванні відкритого тексту використовувався алфавіт АБВГҐДЕЄЖЗИІЇЙКЛМНОПРСТУФХЦЧШЩЬЮЯабвгґдеєжзиіїйклмнопрсту фхцчшщьюя. Припускаючи, що текст зашифрований за допомогою шифру Цезаря, складемо таблицю появи літер в даному шифротексті (табл. 1.6).

Таблиця 1.6. Зустрічальності літер у шифротексті

А	**Б**	**В**	**Г, Ґ**	**Д**	**Е**	**Є**	**Ж**	**З**	**И**	**І**	**Ї**	**Й**	**К**	**Л**	**М**
3	0	3	0	2	2	3	0	4	14	2	8	2	0	4	2
Н	**О**	**П**	**Р**	**С**	**Т**	**У**	**Ф**	**Х**	**Ц**	**Ч**	**Ш**	**Щ**	**Ь**	**Ю**	**Я**
2	1	4	5	10	1	1	7	2	6	10	16	1	4	7	6

З табл. 1.6 видно, що найчастіше у тексті з'являється літера «Ш» – 16 разів. А з табл. 1.5 відомо, що найчастіше в текстах українською мовою зустрічається літера «О». Тому можемо припустити, що літері «Ш» в шифротексті, ймовірно, відповідає літера «О» у відкритому тексті. Якщо послідовності літер А, Б,…, О,…, Ш,…, Я ототожнити із послідовністю їх позицій в алфавіті 0, 1,…, 18,…, 28,…, 33, то можна обчислити ключ *K*: 28–18=10.

Тепер ми можемо відновити початкове повідомлення, записавши його із розділовими знаками: *Шукаємо щастя по країнах, століттях, а воно скрізь і завжди з нами; як риба в воді, так і ми в ньому, і воно біля нас шукає нас самих. Нема його ніде від того, що воно скрізь.*

1.3.2. МЕТОД КАЗІСКІ ТА МЕТОД ФРІДМАНА

У 1863 році офіцер прусської армії, майор *Фрідріх Казіскі* запропонував метод зламу поліалфавітного шифру на прикладі шифру Віженера. Метод Казіскі заснований на наступній ідеї: повторення літер в ключі разом з повторенням літер у відкритому тексті дає повторення літер в зашифрованому тексті. Автор прийшов до висновку, що відстань між повтореннями в шифротексту будуть рівні або кратні довжині (періоду) ключа. Щоб знайти довжину ключа виконаємо наступні дії:

1) знайдемо у шифротексті однакові відрізки довжиною не менше трьох символів (зауважмо, що такі однакові відрізки можуть з'явитися в тексті з досить малою ймовірністю);
2) визначимо відстань між стартовими позиціями відрізків у шифротексті;
3) візьмемо один із спільних дільників цих відстаней в якості довжини ключа.

Для уточнення довжини ключа будемо використовувати метод Фрідмана, що був винайдений американським криптологом *Вільямом Фрідманом* у 1920 році. Цей метод базується на обчислені індексу збігу (ІЗ), який дозволяє визначити для деякої послідовності $x = (x_1 x_2 \dots x_n)$ з літер алфавіту $A =$

$\{a_1, a_2, \dots, a_m\}$ ймовірність того, що два випадкових елемента цієї послідовності збігаються. Значення ІЗ обчислюються за формулою:

$$I_c(x) = \frac{\sum_{i=0}^{m-1} n_i(n_i-1)}{n(n-1)}, \quad (1.1)$$

де n_i – кількість появи літери *i* в послідовності *x, n* – загальна кількість літер в *x*.

Довжину ключа можна визначити за формулою:

$$l \approx \frac{k_p-k_r}{I_c(x)-k_r+\frac{k_p-I_c(x)}{n}}, \quad (1.2)$$

де $k_r = \frac{1}{m}, k_p = \sum_{i=0}^{m-1} p_i^2$, де p_i – частота появи літери *i* в природній мові.

Відомо, що ІЗ рядків осмисленого тексту для різних природніх мов такий:

$I_c(x) = 0{,}058$ – українська мова;

$I_c(x) = 0{,}065$ – англійська мова

Нехай криптограма $c = (c_1 c_2 \dots c_n)$, отримана за допомогою шифру Віженера з ключем рівним *l*. Запишемо її літери в *l* стовпців.

Таблиця 1.7. Запис шифротексту за довжиною ключа

C_1	C_2	…	C_l
c_1	c_2	…	c_l
c_{l+1}	c_{l+2}	…	c_{2l}
c_{2l+1}	c_{2l+2}	…	c_{3l}
…	…	…	…

Якщо довжину ключа визначено правильно, то кожний стовпець C_i – це відрізок відкритого тексту, зашифрованого простою заміною. Тоді ІЗ кожного стовпця буде близьким до ІЗ осмислених текстів цією мовою. Наприклад, для осмислених текстів англійською мовою ІЗ лежатиме в межах $0{,}038 < I_c(x) < 0{,}065$. Якщо довжину ключа визначено неправильно, то стовпці C_i будуть випадковими, а ІЗ таких стовпців буде близьким до 0,038.

Для текстів англійською мовою довжину ключа можна визначити за таблицею 1.8.

Таблиця 1.8. Визначення довжини ключа за значенням ІЗ

l	***1***	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***	***10***	∞
$I_c(x)$	0,0667	0,0525	0,0478	0,0445	0,0441	0,0431	0,0424	0,0414	0,0410	0,0407	0,0384

Припустимо, що на першому етапі ми знайшли довжину ключа l. Тепер для кожного стовпчика C_i визначимо літери, що найчастіше повторюються та за допомогою частотного аналізу знайдемо літери ключа.

Приклад 1.8:

Дано текст, зашифрований шифром Віженера:

MRGFNIATXZQVFFNUXFFYBTCETYXIIXGZKACJLRGKQYEIXOYYAUAPX
YIJLHPRGVTSFPAYNNYURZOPHXWYXLFRNUTZBRFKAHFWFZESYUWZ
MOLLBSBZBJHFPLXKHVIVMZTZHUIWAETIUEDFGLXDIEXIYJIUXPNNEI
XABVCINTVCIEZYYDAZGZIWTYXJIKTRZLMFFKALGZNVKZXIIMXUUNA
PGVXFUSMISKHVYVOCRVXRIWTYXZOIRFNUXZNXLDUDPZGVHVOWM
OYJERLAUGLVTUXTHRBUQZTYTXORNKBASFFXGHQVDSHUYJSYHDYU
WYXYYKHVTUCDACAHXSEVGJIEFZGLXRSBXSYKOEPPNYAKTUACEFYI
LFWEAHCIAUALLZNXMVCKLRRHGFNXMOYUESKPM

Потрібно визначити ключ та прочитати текст.

Використаємо спочатку метод Казіскі для знаходження довжини ключа. У шифротексті триграма TYX зустрічається 3 рази. Відстань між першою і другою появою становить 156 символів, між першою і третьою – 210. НСД (156, 210) = 6, тому можна припустити, що довжина ключового слова рівна 6.

Для підтвердження гіпотези скористаємося методом Фрідмана. Обчислимо ІЗ за формулою (2.1) для всього шифротексту $I_c(\text{с}) = 0{,}043$. Обчислимо довжину ключа за формулою (2.2): $l \approx 6{,}64$. За отриманими даними та за таблицею 2.4 можна зробити висновок, що довжина ключового слова обрана правильно і дорівнює 6.

Запишемо шифротекст у таблицю із 6 стовпчиків (табл. 1.9).

Таблиця 1.9. Запис шифротексту за довжиною ключа 6

C_1	C_2	C_3	C_4	C_5	C_6
M	R	G	F	N	I
A	T	X	Z	Q	V
F	F	N	U	X	F
F	Y	B	T	C	E
T	Y	X	I	I	X
G	Z	K	A	C	J
L	R	G	K	Q	Y
E	I	X	O	Y	Y
A	U	A	P	X	Y
I	J	L	H	P	R
G	V	T	S	F	P
A	Y	N	N	Y	U
R	Z	O	P	H	X
W	Y	X	L	F	R
N	U	T	Z	B	R
F	K	A	H	F	W

C_1	C_2	C_3	C_4	C_5	C_6
F	Z	E	S	Y	U
W	Z	M	O	L	L
B	S	B	Z	B	J
H	F	P	L	X	K
H	V	I	V	M	Z
T	Z	H	U	I	W
A	E	T	I	U	E
D	F	G	L	X	D
I	E	X	I	Y	J
I	U	X	P	N	N
E	I	X	A	B	V
C	I	N	T	V	C
I	E	Z	Y	Y	D
A	Z	G	Z	I	W
T	Y	X	J	I	K
T	R	Z	L	M	F
F	K	A	L	G	Z
N	V	K	Z	X	I
I	M	X	U	U	N
A	P	G	V	X	F
U	S	M	I	S	K
H	V	Y	V	O	C
R	V	X	R	I	W
T	Y	X	Z	O	I
R	F	N	U	X	Z
N	X	L	D	U	D
P	Z	G	V	H	V
O	W	M	O	Y	J
E	R	L	A	U	G
L	V	T	U	X	T
H	R	B	U	Q	Z
T	Y	T	X	O	R
N	K	B	A	S	F
F	X	G	H	Q	V
D	S	H	U	Y	J
S	Y	H	D	Y	U
W	Y	X	Y	Y	K
H	V	T	U	C	D
A	C	A	H	X	S
E	V	G	J	I	E
F	Z	G	L	X	R
S	B	X	S	Y	K
O	E	P	P	N	Y
A	K	T	U	A	C
E	F	Y	I	L	F
W	E	A	H	C	I
A	U	A	L	L	Z
N	X	M	V	C	K
L	R	R	H	G	F
N	X	M	O	Y	U
E	S	K	P	M	

Підрахуємо кількість появи кожної літери алфавіту по стовпцях. Занесемо дані в таблицю 1.10 (комірки, що позначені кольором відповідають літерам, що зустрічаються найчастіше).

Таблиця 1.10. Кількості появи літер по стовпцям шифротексту

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**
C_1	9	1	1	2	6	7	2	5	5	0	0	3	1	6	2	1	0	3	2	6	1	0	4	0	0	0
C_2	0	1	1	0	5	5	0	0	3	1	4	0	1	0	0	1	0	6	4	1	4	8	1	4	9	8
C_3	6	4	0	0	1	0	9	3	1	0	3	3	5	4	1	2	0	1	0	7	0	0	0	13	2	2
C_4	4	0	0	2	0	1	0	6	5	2	1	7	0	1	4	5	0	1	3	2	9	5	0	1	2	6
C_5	1	3	5	0	0	3	2	2	6	0	0	3	3	3	3	1	4	0	2	0	4	1	0	10	11	0
C_6	0	0	3	4	3	6	1	0	4	5	6	1	0	2	0	1	0	5	1	1	4	4	4	2	4	5

Знайдемо тепер саме ключове слово. Так як кожен з стовпців таблиці є результатом зашифрування фрагменту відкритого тексту простою заміною, то спробуємо застосувати частотний аналіз, тобто виконаємо зсув відносно літери, що найчастіше зустрічається у кожному стовпці (табл. 1.11).

Таблиця 1.11. Визначення літер ключового слова із застосуванням частотного аналізу

Стовпець шифротексту	Літера, що найчастіше зустрічається	Зсув відносно E	Можлива літера ключового слова
C_1	A	0 – 4 mod 26 = 22	W
	E	4 – 4 mod 26 = 0	A
	F	5 – 4 mod 26 = 1	B
	N	13 – 4 mod 26 = 9	J
	T	19 – 4 mod 26 = 15	P
C_2	Y	24 – 4 mod 26 = 20	U
	V	21 – 4 mod 26 = 17	R
	Z	25 – 4 mod 26 = 21	V
	R	17 – 4 mod 26 = 13	N
C_3	X	23 – 4 mod 26 = 19	T
	G	6 – 4 mod 26 = 2	C
	T	19 – 4 mod 26 = 15	P
	A	0 – 4 mod 26 = 22	W
C_4	U	20 – 4 mod 26 = 16	Q
	L	11 – 4 mod 26 = 7	H
	H	7 – 4 mod 26 = 3	D
	Z	25 – 4 mod 26 = 21	V
C_5	C	2 – 4 mod 26 = 24	Y
	I	8 – 4 mod 26 = 4	E
	X	23 – 4 mod 26 = 19	T
	Y	24 – 4 mod 26 = 20	U
	U	20 – 4 mod 26 = 16	Q

Стовпець шифротексту	Літера, що найчастіше зустрічається	Зсув відносно E	Можлива літера ключового слова
C_6	F	5 – 4 mod 26 = 1	B
	K	10 – 4 mod 26 = 6	G
	J	9 – 4 mod 26 = 5	F
	R	17 – 4 mod 26 = 13	N
	Z	25 – 4 mod 26 = 21	V
	V	21 – 4 mod 26 = 17	R

Отже, ключове слово: ARTHUR. Тепер можемо дешифрувати текст, розділяючи слова пропусками: Many traces we found of him in the bog girt island where he had hid his savage ally a huge driving wheel and a shaft half filled with rubbish showed the position of an abandoned mine beside it were the crumbling remains of the cottages of the miners driven away no doubt by the foul reek of the surrounding swamp in one of these a staple and chain with a quantity of gnawed bones showed where the animal had been confined a skeleton with a tangle of brown hair adhering to it lay among the debris.

Контрольні запитання до розділу 1

1. У чому полягає забезпечення конфіденційності, цілісності, доступності, інформаційних ресурсів?
2. Дайте визначення поняттям: криптологія, криптографія та криптоаналіз.
3. Що таке криптографічний алгоритм та шифр?
4. Що таке криптографічний ключ?
5. Розкрийте поняття зашифрування та дешифрування даних.
6. Дайте визначення відкритого та закритого тексту.
7. Назвіть складові криптографічної системи.
8. У чому полягає криптостійкість криптографічної системи?
9. Що таке атака на криптографічну систему?
10. Дайте коротку класифікацію шифрів.
11. Опишіть алгоритм шифрування Цезаря.
12. До якого виду шифрів заміни (підстановки) відносять шифр Цезаря?
13. Опишіть алгоритм шифру частоколу.
14. Опишіть алгоритм шифру Плейфера.
15. Опишіть алгоритм шифрування криптосистемою Хілла.
16. Що являє собою ключ в криптосистемі Хілла?
17. Що є ключем у шифрі Віженера?
18. Опишіть алгоритм шифрування Віженера.
19. У чому суть методу частотного криптоаналізу?
20. Поясніть відмінність між шифрами моноалфавітної та поліалфавітної підстановки (заміни).
21. У чому полягає основна слабкість шифрів простої моноалфавітної заміни

22. Яка літера найчастіше зустрічається у текстах українською (англійською) мовою?
23. Які кроки потрібно виконати для визначення довжини ключа у шифрі Віженера методом Казіскі?
24. Як уточнити довжину ключа методом Фрідмана?
25. Що таке індекс збігу?

Тести до розділу 1

1. *Криптологія – це…*
   а) наука, що займається розробкою засобів та методів приховування факту передачі інформації
   б) наука, яка вивчає методи побудови та аналізу систем захисту інформаційних ресурсів, основаних на математичних перетвореннях даних з використанням секретних параметрів
   в) наука про методи та способи розкриття зашифрованих повідомлень, а також про тактику та стратегію їх застосування
   г) наука про математичні методи порушення безпеки криптографічних систем
   д) наука про методи та способи встановлення справжності інформаційних об'єктів
2. *Фундамент криптології як науки у 1949 р. заклала праця «Теорія зв'язку в секретних системах». Який вчений є автором цієї праці?*
   а) Брюс Шнайєр
   б) Вільям Фрідман
   в) Алан Тюрінг
   г) Фрідріх Казіскі
   д) Клод Шеннон
3. *Набір математичних правил та процедур, який описує такі види перетворень, як шифрування, формування та перевірка ЦП, обчислення хеш-значень, криптографічних контрольних сум тощо називається криптографічний...*
   а) шифр
   б) ключ
   в) алгоритм
   г) протокол
   д) аналіз
4. *Ключ шифрування – це…*
   а) блок інформації фіксованої довжини, що одержується із відкритих даних, однозначно відповідний відкритим даним

б) блок інформації фіксованої довжини, що одержується із зашифрованих даних, однозначно відповідний відкритим даним

в) дані, одержані в результаті криптографічного перетворення блоку даних і (або) його параметрів

г) конкретний секретний стан деяких параметрів алгоритму криптографічного перетворення даних, що забезпечує вибір одного перетворення із сукупності можливих для даного алгоритму

д) секретна інформація для автентифікації користувачів або даних

5. *Початкове повідомлення, що підлягає зашифруванню прийнято називати…*
   а) відкритим текстом
   б) оригінальним текстом
   в) простим текстом
   г) чистим текстом
   д) криптотекстом
6. *Процес перетворення відкритого тексту до виду, незрозумілого несанкціонованому користувачеві називається...*
   а) дешифруванням
   б) кодуванням
   в) зашифруванням
   г) декодуванням
   д) гамуванням
7. *За принципом Керкхофа криптографічна стійкість шифру повинна визначатися тільки...*
   а) довжиною ключа шифрування
   б) часом шифрування
   в) секретністю алгоритму шифрування
   г) складністю шифру
   д) секретністю ключа
8. *Криптосистема називається асиметричною, якщо …*
   а) відкритий текст завжди розбивається на блоки однакової довжини
   б) один і той самий ключ використовується для шифрування та дешифрування повідомлень
   в) використовуються два ключа – один для шифрування, а другий – для дешифрування повідомлень
   г) використовуються циклічно повторювальні операції (раунди)
   д) перший та останній символи криптограми завжди різні

9. *Властивість криптосистеми протидіяти атакам супротивника, спрямованим на отримання секретного ключа або відкритого повідомлення називається...*
   а) конфіденційністю
   б) цілісністю
   в) доступністю
   г) достовірністю
   д) криптостійкістю
10. *Шифр, в якому літера тексту циклічно замінюється на літеру в абетці на k позицій вперед (праворуч), де k – ключ, називається шифром…*
   а) Полібія
   б) Цезаря
   в) Хілла
   г) Плейфера
   д) Віженера
11. *Який вигляд буде мати повідомлення CAESAR після зашифрування за допомогою шифру Цезаря з ключем 3?*
   а) BZDRZQ
   б) FDHVDU
   в) HFJXFW
   г) JHLZHY
   д) ECGUCT
12. *Який вигляд буде мати повідомлення CRYPTOGRAPHY після зашифрування за допомогою шифру частоколу з ключем 3?*
   а) YOAYRTRHCPGP
   б) RTRHYOAYCPGP
   в) FUSBWJRUSDKB
   г) CRPYTGORPAHY
   д) YRCARGOTPYHP
13. *Що являє собою ключ в криптосистемі Хілла?*
   а) секретне слово
   б) таблиця з секретним словом
   в) квадратна матриця та її розмірність
   г) ціле число від 1 до 25
   д) матриця 5x5 англійського алфавіту
14. *У шифрі Плейфера, для того, щоб зашифрувати повідомлення необхідно спочатку…*

а) розбити його на біграми

б) визначити координати кожної літери в квадраті

в) знайти порядковий номер кожної літери за таблицею

г) розбити його на триграми

д) літери повідомлення записати як степені

15. *Який шифротекст буде у результаті шифрування слова CAR за допомогою шифру Плейфера із заданою ключовою матрицею?*

D	A	R	K	H
O	S	E	B	C
F	G	I	L	M
N	P	Q	T	U
V	W	X	Y	Z

а) HSER

б) SHER

в) XMSE

г) ORKD

д) BDAW

16. *Криптоаналітик виявив, що у шифротексті LODDOB VKDO DRKX XOFOB найчастіше зустрічається літера O. Також відомо, що відкритий текст був англійською мовою та шифрувався шифром Цезаря. Яке значення найімовірніше буде мати ключ шифрування?*

а) 16

б) 4

в) 18

г) 10

д) 12

17. *Відкритий текст APRICOT зашифровано за допомогою шифру Цезаря як FUWNHTY. Яку довжину ключа використано?*

а) 10

б) 7

в) 3

г) 23

д) 5

18. *Шифр Віженера є шифром…*

а) моноалфавітної заміни

б) поліалфавітної заміни

в) моноалфавітної перестановки

г) поліалфавітної перестановки

д) мультиалфавітної підстановки

19. *Якщо слово WISDOM зашифрувати за допомогою шифру Віженера з ключем GREAT, то отримаємо шифротекст…*
    а) QRMAFU
    б) KJMLFS
    в) CZWDHS
    г) DZWDIS
    д) BYVCHR
20. *Метод криптоаналізу, що ґрунтується на припущенні про існування залежності між частотою появи символів відкритого тексту та символів шифротексту – це...*
    а) лінійний аналіз
    б) частотний аналіз
    в) метод повного перебору
    г) метод Фрідмана
    д) метод Казіскі

Задачі до розділу 1

1. Відкритий текст RAINBOW зашифруйте за допомогою шифру Цезаря, використовуючи ключ $K = 5$.
2. Зашифруйте відкритий текст ОЧІКУВАННЯ РАДОСТІ ТЕЖ РАДІСТЬ, використовуючи шифр частоколу з ключем $K = 3$.
3. Дешифруйте за допомогою шифру частоколу повідомлення TEOSTSWHISARTEEEOSMB, використовуючи ключ $K = 4$.
4. Відкритий текст BETTER LATE THAN NEVER зашифруйте за допомогою шифру Плейфера, використовуючи ключ FRIEND.
5. Шифротекст QOOFQMDFTFELTQDKBFFP дешифруйте за допомогою шифру Плейфера, використовуючи ключ WORK.
6. У криптосистемі Хілла з матрицею $\begin{pmatrix} 3 & 3 \\ 2 & 5 \end{pmatrix}$ зашифруйте текст FLY A KITE.
7. У криптосистемі Хілла з матрицею $\begin{pmatrix} 6 & 24 & 1 \\ 13 & 16 & 10 \\ 20 & 17 & 15 \end{pmatrix}$ зашифруйте текст LIVE AND LEARN.
8. У криптосистемі Хілла з матрицею $\begin{pmatrix} 8 & 5 & 10 \\ 21 & 8 & 21 \\ 21 & 12 & 8 \end{pmatrix}^{-1}$ дешифруйте текст WRHSHXLASLQV.
9. Використовуючи шифр Віженера з ключовим словом TIME, зашифруйте відкритий текст LIKE CURES LIKE.

10. Використовуючи шифр Віженера з ключовим словом СЕКРЕТ, зашифруйте відкритий текст З ВИДИМОГО ПІЗНАВАЙ НЕВИДИМЕ.

11. Використовуючи шифр Віженера з ключовим словом WISDOM, дешифруйте зашифрований текст ADWUMFDQFJMAQKSQWYWOAQSUOZWDZ.

12. За допомогою частотного криптоаналізу відновіть текст OIITCSYVJEGIXSXLIWYRWLMRIERHCSYGERRSXWIIEWLEHS, зашифрований алгоритмом Цезаря.

13. За допомогою частотного криптоаналізу відновіть текст ВЬСЄРУХЗТДЇЕЯЯХЧХВЙУХЗШСЄЧЖНГХЇЄЗФХМЄЖЬЩО, зашифрований алгоритмом Цезаря.

14. За допомогою методу Казіскі доведіть, що наступний текст TCKQZANRAEGDNJAIWOVCJNBZVBOCWNRCPUSNFHKUSHCHDRAPH FJIWOVCJVBPBFANZEGMCESWGZANRAEGYESWGSIBFAYSWQSNFBK GTKYZKJSNFUNROPYSWQSNFVWISRVGEBBOUONRJEFWKAOJQWJFD EESKGVAEGPBQNROPRHDRWNBKJ, зашифрований алгоритмом Віженера з довжиною ключа 3. Знайдіть літери ключового слова та відновіть повідомлення.

15. За допомогою методу Казіскі та Фрідмана здійсніть криптоаналіз тексту ERXEHJVYSOSPKMUVCOGSIXFUFLTHTVYCBTWPTMCLHTRGCMGQE AGRDVFEGTDJPFPWPGVLIASCSGABHAFDIASEFBTVZGIIIHDGIDDKA VYCCXQGJQPKMVIYCLTQIKPMWQEQDYHGEMCTPCKRAXTKVJSPWV YJXMHNVCFNWRDCCMVQNCKXFVYCSTBIVPDYOEFBTVZGIIQXWPX APIHWICSUMVYCTGSOPFPLACUCXMSUJCCMWCCRDUSCSJTMCEYY CZSVYCRKMRKMVKOJZAB, зашифрованого алгоритмом Віженера. Знайдіть літери ключового слова та відновіть повідомлення.

РОЗДІЛ 2. СИМЕТРИЧНІ КРИПТОГРАФІЧНІ СИСТЕМИ

2.1. ПОТОКОВІ СИМЕТРИЧНІ ШИФРИ

Блоковий алгоритм призначений для шифрування блоків певної довжини. Однак може виникнути необхідність шифрування даних не блоками, а по символах. Потоковий алгоритм шифрування усуває необхідність розбивати повідомлення на ціле число блоків досить великої довжини і, тому, може працювати в реальному часі.

Потоковий шифр – шифр, що перетворює кожен символ (літеру, біт або байт) відкритого тексту у символ шифротексту, залежно від ключа та розташування символів у тексті.

Ключовий потік (гама) – це бітова послідовність, що визначається за допомогою ключа шифру.

Стійкість потокових шифрів цілком залежить від якості й криптографічної стійкості генератора псевдовипадкових чисел, за допомогою якого отримується потік ключа.

Генератор псевдовипадкових чисел (ГПВЧ) – це пристрій або алгоритм, який за заданими параметрами генерує послідовність *псевдовипадкових чисел* (ПВЧ). Метою використання ГПВЧ у потокових шифрах є отримання нескінченної ключової послідовності, за використання відносно малої довжини самого початкового ключа. Найважливішою характеристикою ГПВЧ є довжина періоду повторення, після якого випадкові числа, на виході ГПВЧ почнуть повторюватися.

В потокових шифрах для шифрування бінарних даних (потоку бітів) виконується їх додавання з гамою за модулем 2 (операція XOR, eXclusive OR – виключне або), що позначається $\oplus$ (табл. 2.1). В результаті чого виходять біти зашифрованих даних (рис. 2.1).

Таблиця 2.1. Операція XOR над бітами

$\oplus$	**0**	**1**
0	0	1
1	1	0

При дешифруванні одержувач, використовуючи точно такий самий ключ, виконує додавання за модулем 2 кожного символу ключа та шифротексту.

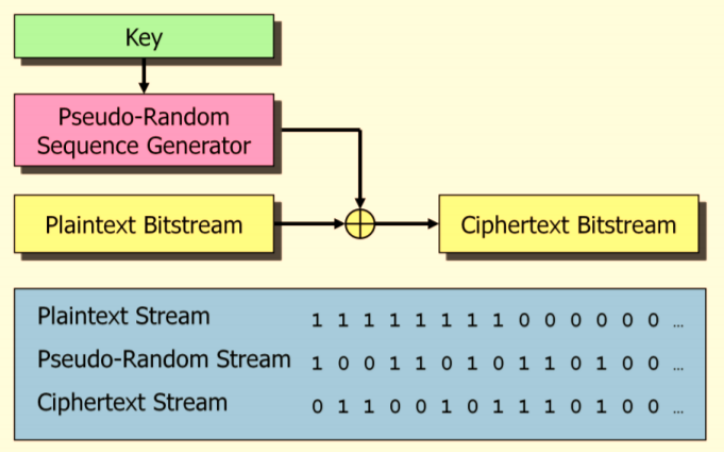


Рис. 2.1. Принцип роботи потокового шифру

Отже, оскільки зашифрувальне (й розшифрувальне) перетворення для всіх потокових шифрів таке саме, то вони можуть відрізнятися тільки способом побудови генераторів ключів. Виходить, що безпека системи повністю залежить від властивостей генератора потоку ключів.

Синхронні потокові шифри – це шифри, у яких потік ключів генерується незалежно від відкритого й зашифрованого повідомлення.

Під час зашифрування генератор потоку ключів видає біти потоку ключів, які ідентичні бітам потоку ключів під час розшифрування. Втрата знаку зашифрованого повідомлення призведе до порушення синхронізації між цими двома генераторами й неможливості розшифрування частини повідомлення. Очевидно, що в цій ситуації відправник та одержувач повинні повторно синхронізуватися для продовження роботи.

Потокові шифри, що самосинхронізуються (асинхронні потокові шифри) – це шифри, в яких потік ключів створюється функцією ключа й фіксованою кількістю знаків зашифрованого повідомлення (використовується n попередніх знаків шифротексту для обчислення потоку ключа).

Отже, внутрішній стан генератора потоку ключів є функцією попередніх n бітів зашифрованого повідомлення. Тому генератор потоку ключів, який розшифровує, прийнявши n бітів, автоматично синхронізується із шифрувальним генератором.

Приклад 2.1:

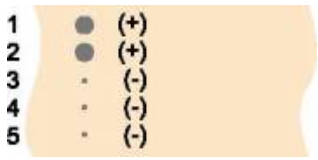
Шифрування за допомогою операції XOR повідомлення *SUN* із використанням псевдовипадкової ключової послідовності 00001011 00010010 00001111:

Відкритий текст	01010011 01010101 01001110
Ключова гама	00001011 00010010 00001111
Результат додавання за модулем 2	01011000 01000111 01000001
Шифротекст	*XGA*

2.1.1. ШИФР ОДНОРАЗОВОГО БЛОКНОТУ (ШИФР ВЕРНАМА)

У 1949 році Клод Шеннон опублікував роботу, в якій довів абсолютну стійкість шифру Вернама. Інших шифрів з цією властивістю не існує. При цьому умови, яким повинен задовольняти ключ, настільки сильні, що практичне використання шифру Вернама є важко здійсненним. Тому він використовується тільки для передачі повідомлень найвищої секретності.

Шифр одноразового блокноту, або шифр Вернама, було запропоновано у 1917 році співробітниками телеграфної компанії AT&T *Мейджором Джозефом Моборном* та *Гільбертом Вернамом*. Відкритий текст представлявся у вигляді п'ятизначних імпульсних комбінацій – коді Бодо. Наприклад, літера «А» на паперовій стрічці мала вигляд (рис. 2.2):



«+» – отвір;
«-» – його відсутність.

Рис. 2.2. Літера «А» на паперовій стрічці

У класичному розумінні одноразовий блокнот є унікальною послідовністю символів ключа, що згенерована випадковим чином. Заздалегідь готувалася «гама» – перфострічка з випадковими знаками. Потім електромеханічно складалися її імпульси з імпульсами знаків відкритого тексту. Отримана сума представляла собою шифротекст. На приймальному кінці імпульси, отримані по каналу зв'язку, складалися з імпульсами тієї ж самої гами, в результаті чого відновлювалися вихідні імпульси повідомлення.

Ідея шифру Вернама легко поширюється на двійкові дані. Ключем виступає послідовність випадкових символів. При цьому ключ повинен володіти трьома критично важливими властивостями:

1) бути дійсно випадковим;

2) за розміром збігатися з заданим відкритим текстом (ключ ні в якому разі не зациклюється);

3) застосовуватися тільки один раз.

Навіть, якщо криптографічний аналітик спробує використовувати всі можливі набори ключів, заданих для зашифрованих даних, і відновити варіанти вхідних даних, то вони виявляться рівноймовірними. Тобто немає можливості вибрати вхідні дані, які були насправді надіслані.

Недоліком шифру одноразового блокнота є складність процедури обміну ключем. Ключ довгий, таємний, і його необхідно застосовувати лише одного разу. Відповідно має існувати надійний канал передавання таємного ключа. Сама назва шифру пішла від того, що дві сторони каналу зв'язку мали однаковий таємний блокнот із ключовою послідовністю. В одному сеансі зв'язку вони використовували необхідну кількість бітів з однієї й тієї самої сторінки блокнота в якості ключової послідовності. Після завершення сеансу відповідний листок блокнота знищували й у наступному сеансі для формування ключа використовували наступний листок блокнота.

Іншим недоліком є необхідність синхронізації приймання та передавання. Якщо приймач втратить хоча б один біт шифротексту, він не зможе розшифрувати текст. Ще одним недоліком шифру є те, що він не забезпечує захист даних від спотворення. Незважаючи на ці недоліки, абсолютна стійкість є унікальною властивістю цього шифру.

2.1.2. ПОТОКОВИЙ ШИФР RC4

RC4 (Rivest Cipher 4, Ron's Code) – потоковий шифр, який розробив Рональд Рівест у 1987 р. Шифр RC4 застосовувався в деяких широко поширених стандартах і протоколах таких, як WEP, WPA і TLS (до 2015 року).

Головними факторами, що сприяли широкому використанню RC4, були простота його апаратної й програмної реалізації, а також висока швидкість роботи алгоритму в обох випадках.

Алгоритм RC4 включає в себе два етапи. На першому, підготовчому, етапі проводиться ініціалізація матриці стану і масиву ключів, а також початкова перестановка матриці стану, на основі значень ключа. На другому, основному, етапі обчислюються безпосередньо псевдовипадкові числа *k*.

Ініціалізація матриці стану і масиву ключів

Матриця стану (256 байт) початково заповнюється значеннями:

$$S[0] = 0, S[1] = 1, S[2] = 2, \ldots, S[254] = 254, S[255] = 255.$$

Ключем є послідовність байтів, що записується у тимчасовий масив *T*:

$$T[0], T[1], T[2], \ldots, T[254], T[255].$$

Якщо ключ шифрування має точно *256* байтів, то байти копіюються в масив *T*, інакше байти повторюються, поки не заповниться масив *T*:

```
for
    i = 0 to 255 do S[i] = i;
T[i] = K[i mod k - len];
```

.

Перестановка матриці стану на основі значень ключа

Кожен черговий елемент S_i обмінюється місцями з елементом S_j, номер *j* якого визначається сумою номера елементу *j*, самого елементу S_i та елементу ключа T_i (значення лічильників *i* та *j* спочатку дорівнюють *0*):

```
j = 0;
for
    i = 0 to 255 do
    {
        j = (j + S[i] + T[i])mod 256;
        Swap(S[i], S[j]);
    }
```

.

Перестановка матриці стану і генерація ключового потоку

Два елементи матриці стану міняються місцями на основі двох індивідуальних змінних *i* та *j*, що обчислюються за відповідними формулами. Потім сума значень елементів S_i та S_j визначає індекс *t* елементу S_t, який використовуватиметься як ключ *k*:

```
i, j = 0;
while (true)
    i = (i + 1)mod 256;
j = (j + S[i])mod 256;
Swap(S[i], S[j]);
t = (S[i] + S[j])mod 256;
k = S[t];
```

Шифрування (розшифрування)

Після того як ключовий потік *k* був створений, байт відкритих даних зашифровується за допомогою *k* (виконується додавання за модулем 2), щоб створити байт зашифрованих даних. Розшифрування являє собою обернений процес.

2.1.3. ГЕНЕРАТОРИ ПСЕВДОВИПАДКОВИХ ПОСЛІДОВНОСТЕЙ

ГПВЧ можуть використовуватися як генератори ключів у потокових шифрах. ГПВЧ генерує послідовність бітів, схожу на випадкову. Насправді, такі послідовності обчислюються за певними правилами та не є випадковими, тому вони можуть бути абсолютно точно відтворені як на передаючій, так і на приймаючій сторонах.

Для використання з криптографічною метою ГПВЧ повинен мати такі властивості:

1) Непередбачуваність (неможливо визначити наступне число з імовірністю, більшою за $\frac{1}{m}$, де m– потужність алфавіту генератора);
2) Відтворюваність дозволяє повторити (за відомих початкових значень) з абсолютною точністю послідовність на виході ГПВЧ в довільний момент часу та довільну кількість разів;
3) Великий період повторення (настільки великий, що його неможливо відтворити сучасними технічними засобами);
4) Числа, що генеруються мають бути статистично рівномірно розподілені.

Отже, у випадку застосування псевдовипадкової ключової послідовності користувачам не потрібно обмінюватися довгим ключем. Вони обмінюються таємними параметрами однакових генераторів. Після цього вони можуть згенерувати однакові псевдовипадкові ключові послідовності. Таємні значення

параметрів власне і є ключем. Це, як правило, справжня випадкова послідовність бітів, однак значно коротша за можливі довжини повідомлень. Таку випадкову послідовність називають паростком (*seed*), який породжує гаму.

Лінійний конгруентний генератор (LCG)

Для обчислення чергового числа x_{i+1} використовується формула:

$$x_{i+1} = (a \cdot x_i + b) mod\, m,$$

де a, b, m – деякі константи, а x_i – попереднє псевдовипадкове число, а x_0 – початкове значення.

Якщо параметри LCG обрані правильно, то генератор буде породжувати випадкові числа з максимальним періодом, що дорівнює m.

Приклад 2.2:

Нехай $a = 5, b = 3, m = 11, x_0 = 1$.

$x_1 = (5 \cdot 1 + 3) mod\, 11 = 8;$ $x_5 = (5 \cdot 4 + 3) mod\, 11 = 1;$

$x_2 = (5 \cdot 8 + 3) mod\, 11 = 10;$ $x_6 = (5 \cdot 1 + 3) mod\, 11 = 8;$

$x_3 = (5 \cdot 10 + 3) mod\, 11 = 9;$ $x_7 = (5 \cdot 8 + 3) mod\, 11 = 10;$

$x_4 = (5 \cdot 9 + 3) mod\, 11 = 4;$ $x_8 = (5 \cdot 10 + 3) mod\, 11 = 9.$

Лінійні конгруентні генератори не рекомендують використовувати, оскільки криптоаналітики навчилися відновлювати всю послідовність ПВЧ із кількох значень.

ГПВЧ на основі регістрів зсуву з лінійним зворотним зв'язком (LFSR)

Регістри зсуву зі зворотним зв'язком можуть застосовуватися для отримання потоку псевдовипадкових бітів і складаються з двох частин: власне *n*-бітного регістра зсуву та пристрою зворотного зв'язку. Зворотний зв'язок є функцією XOR певних бітів регістра, які називають ***відводами*** (taps). Розташування відводів для зворотного зв'язку може бути представлене многочленом з коефіцієнтами 0 або 1.

Алгоритм LFSR

1. У регістр записується початкове значення.
2. Обчислюється XOR відповідних бітів (відводів).
3. Обчислений біт записується до першої позиції регістру ліворуч.
4. Усі решта біт зсуваються на одну позицію праворуч.

5. Останній біт з правого боку усувається з регістру і стає черговим бітом псевдовипадкової послідовності.

6. Алгоритм повторюється з 2 кроку.

Приклад 2.3:

Розглянемо 4-бітовий LFSR із многочленом зворотного зв'язку $x^4 + x^3 + 1$, тобто відведенням першого й четвертого розрядів (рис. 2.3).

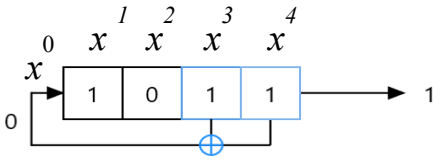


Рис. 2.3. LFSR

Запишемо у регістр початкове значення 1011. На кожному кроці весь вміст регістра зсувається праворуч на один розряд. При цьому можна отримати як результат один біт. На вільне місце ліворуч надходить біт, що дорівнює результату обчислення функції зворотного зв'язку – XOR відповідних відводів. Вихідну послідовність генератора псевдовипадкових біт утворює останній біт внутрішнього стану (біт, який витягується). У результаті чого отримаємо біти псевдовипадкової послідовності:

Номер стану	**Внутрішній стан Q R S T**	**Результат обчислення S⊕T**	**Біт, псевдовипадкової послідовності**
1	1 0 1 1	0	1
2	0 1 0 1	1	1
3	1 0 1 0	1	0
4	1 1 0 1	1	1
5	1 1 1 0	1	0
6	1 1 1 1	0	1
7	0 1 1 1	0	1
8	0 0 1 1	0	1
9	0 0 0 1	1	1
10	1 0 0 0	0	0

ГПВЧ на основі алгоритму BBS

Широке поширення в криптографії набув алгоритм Блюма – Блюма – Шуба (від прізвищ авторів – *L. Blum, M. Blum, M. Shub*), який називають ще генератором квадратичних лишків.

Послідовність, сформована за допомогою цього методу, має статистичні властивості, близькі до генераторів випадкових чисел, а метод є досить крипостійким.

Алгоритм BBS

1. Обирають два великих простих числа p та q, які конгруентні. При діленні цих чисел на *4* повинен виходити однаковий залишок *3*.

2. Обчислюється $M = p \cdot q$ *– ціле число Блюма*.

3. Вибирається інше випадкове ціле число x, взаємно просте з M.

4. Обчислюється $x_0 = x^2 mod\ M$ *– стартове число генератора*.

5. На кожному *n*-му кроці обчислюється $x_{n+1} = x_n^2 mod\ M$.

6. Результатом *n*-го кроку є один (зазвичай молодший) біт числа x_{n+1}.

Приклад 2.4:

Нехай $p = 11$, $q = 19$, (переконуємося, що $11\ mod\ 4 = 3$, $19\ mod\ 4 = 3$).

Тоді $M = p \cdot q = 11 \cdot 19 = 209$.

Виберемо x, взаємно просте з M: нехай $x = 3$.

Обчислимо стартове число генератора x_0:

$$x_0 = x^2 mod\ M = 3^2 mod\ 209 = 9\ mod\ 209 = 9$$

Далі обчислимо перші десять значень x_i за алгоритмом *BBS*. Як випадковий біт будемо брати молодший біт у двійковому записі числа x_i.

$x_1 = 9^2 mod\ 209 = 81\ mod\ 209\ = 81$;	1
$x_2 = 81^2 mod\ 209 = 6561\ mod\ 209\ = 82$;	0
$x_3 = 82^2 mod\ 209 = 6724\ mod\ 209\ = 36$;	0
$x_4 = 36^2 mod\ 209 = 1296\ mod\ 209\ = 42$;	0
$x_5 = 42^2 mod\ 209 = 1764\ mod\ 209\ = 92$;	0
$x_6 = 92^2 mod\ 209 = 8464\ mod\ 209\ = 104$;	0
$x_7 = 104^2 mod\ 209 = 10816\ mod\ 209\ = 157$;	1
$x_8 = 157^2 mod\ 209 = 24649\ mod\ 209\ = 196$;	0
$x_9 = 196^2 mod\ 209 = 38416\ mod\ 209\ = 169$;	1
$x_{10} = 169^2 mod\ 209 = 28561\ mod\ 209\ = 137$;	1

2.2. БЛОКОВІ СИМЕТРИЧНІ ШИФРИ

Характерною особливістю блокових шифрів є те, що дані, представлені в комп'ютерній пам'яті, розбиваються на блоки фіксованої довжини (наприклад, 64 біта, 128 біт). Якщо останній фрагмент коротше довжини блоку – його доповнюють будь-яким способом (наприклад, незначущими нулями). Алгоритми зашифрування або розшифрування блокових шифрів використовують один і той самий секретний ключ. Ключем є послідовність бітів фіксованої довжини, з якої генеруються раундові ключі за яким-небудь математичним правилом.

Існують різні методи побудови блокових симетричних шифрів.

SP-мережа. Підхід до побудови блокового шифру, що містить послідовні перестановки й підстановки, називають мережею перестановок – підстановок (substitution – permutation network – SP) або SP-мережею. Прикладом здійснення такого підходу до побудови шифру є стандарт блокового симетричного шифрування AES.

В основі роботи SP-мережі лежать методи маскування надлишковості відкритого тексту, якими, за Шенноном, є перемішування й розсіювання.

Перемішування маскує зв'язок між відкритим текстом і шифротекстом. Воно ускладнює спроби знайти у шифротексті надлишковість і статистичні закономірності. Простим способом перемішування є підстановка.

Розсіювання (дифузія) розсіює надлишковість відкритого тексту, поширюючи її по всьому шифротексту. Простим способом здійснити розсіювання є перестановка (транспозиція). Сучасні шифри використовують форми розсіювання, які дають можливість розкидати частини повідомлення по всьому повідомленню, що сприяє виникненню так званого лавинного ефекту.

Лавинний ефект означає, що невеликі зміни в початкових даних (чи в ключі) можуть викликати значні зміни в зашифрованих даних.

Мережі Фейстеля. Одним з найпоширеніших способів побудови блокових шифрів є використання мережі Фейстеля, названа на честь дослідника, який працював свого часу в корпорації IBM і був одним з авторів стандарту DES. Мережею Фейстеля є загальний метод перетворення за допомогою довільної

функції (яку позначають *f*-функцією) у перестановку на множині блоків. Цю конструкцію, яку винайшов Хорст Фейстель, було використано в багатьох шифрах, зокрема в DES і ГОСТ 28147-89.

Розглянемо принцип побудови мережі Фейстеля (рис. 2.4). Кожен блок даних розбивається на дві рівні частини – ліву (L) і праву (R). Права частина видозмінюється деякою функцією $f(R,K)$ залежно від раундового ключа K. Здійснюється додавання за модулем 2 лівої частини L та $f(R,K)$. Результат додавання присвоюється новому правому підблоку, а правий підблок присвоюється без змін новому лівому підблоку (вхідні дані для наступного раунду). Описані операції повторюються N раундів. При переході від одного раунду до іншого міняються раундові ключі (K_0 на K_1 і т.д.)

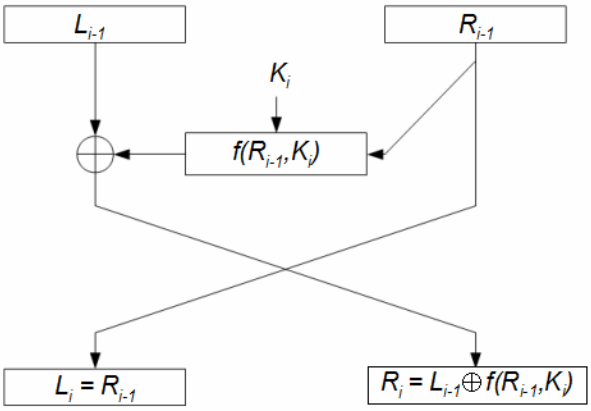


Рис. 2.4. Структура мережі Фейстеля

Функція шифрування (*f*-функція), як правило, є послідовністю залежних від ключа нелінійних підстановок, розсіювальних перестановок і зсувів.

Дешифрування відбувається так само, як і зашифрування, за винятком лише того, що ключі йдуть у зворотному порядку.

2.2.1. АЛГОРИТМ DES

Американський стандарт шифрування даних (Data Encryption Standard), оснований на мережі Фейстеля та прийнятий у 1977 році, є типовим представником сімейства блокових шифрів.

Ключ шифрування складається з 56 випадкових бітів; додається ще 8 біт в позиціях 8, 16,…, 64, таким чином, щоб кожен байт містив непарну кількість одиниць. (використовується при знаходженні помилок при обміні та зберіганні ключів).

Процес шифрування полягає в початковій перестановці 64 бітів вхідного блоку, шістнадцяти циклах шифрування та кінцевій перестановці бітів (рис. 2.5).

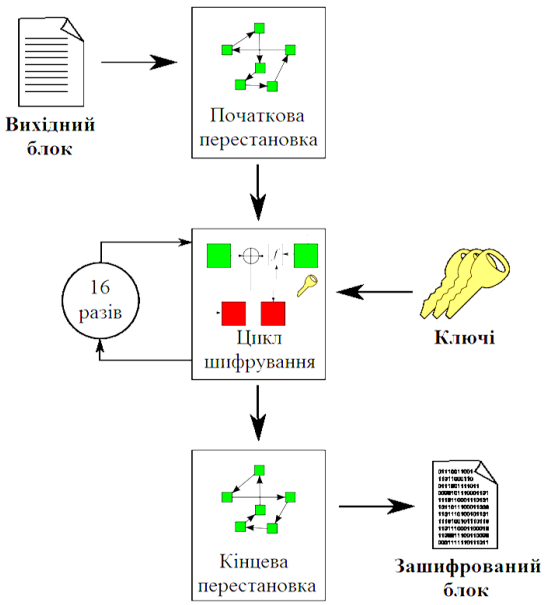


Рис. 2.5. Загальна схема алгоритму DES

Розглянемо алгоритм докладніше.

Початкова перестановка

Початковий текст, що являє собою 64-бітний блок $X = (x_1,\ x_2,\ x_3, \ldots, x_{64})$ перетворюється в 64-бітний блок $X_0 = IP(X)$ за допомогою початкової перестановки *IP* (Initial Permutation), що визначається таблицею 2.2.

Таблиця 2.2. Матриця початкової перестановки IP

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

Раунди шифрування

Після *IP*-перестановки 16 разів повторюється процедура шифрування блоку X_0 за допомогою функції f та раундових ключів K_i, де $i = 1, 2, \ldots, 16$ (рис. 2.6).

Кожен раунд шифрування містить такі етапи:

1. $X_0 = IP(X)$ розбивається на дві половини L_0, R_0, *де* L_0 – перші (старші) 32 біти блоку X_0, а R_0 – останні (молодші) 32 біти блоку X_0.

2. Права половина R_i– це бітове додавання L_{i-1}та $f(R_{i-1},\ K_i)$ по модулю 2:

   $R_i = L_{i-1} \oplus f(R_{i-1},\ K_i)$.

3. Ліва половина L_i дорівнює правій половині попереднього блоку R_{i-1} без змін:

   $L_i = R_{i-1}$.

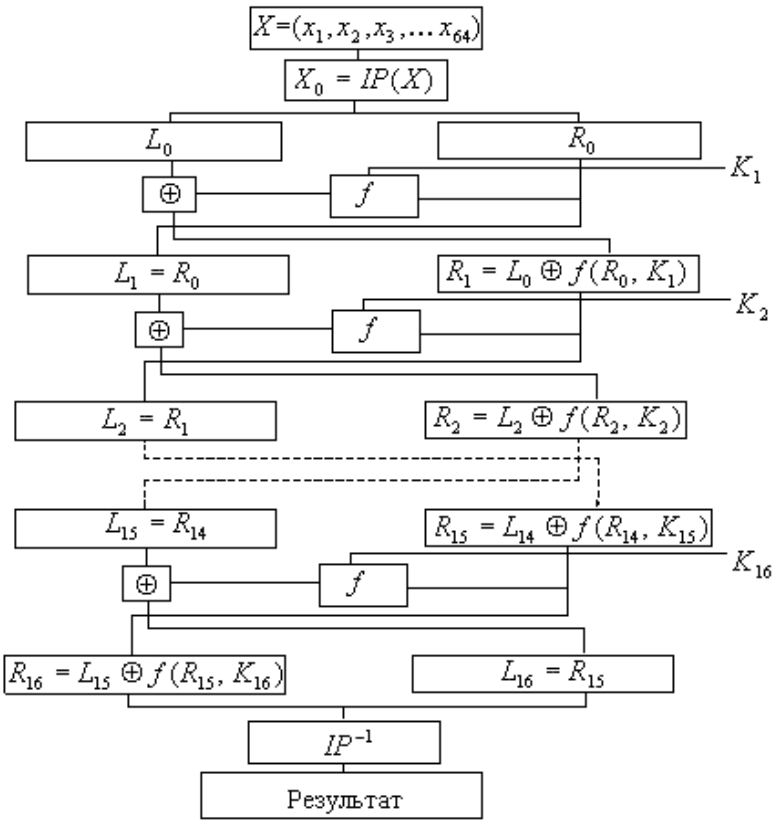


Рис. 2.6. Схема шифрування алгоритму DES

Після 16-ї ітерації ліва і права половини блока не міняються місцями.

Основна функція шифрування (функція Фейстеля)

Аргументи функції f – 32-бітовий вектор R_{i-1}та 48-бітовий підключ K_i. Для обчислення функції f використовуються:

1) функція розширення E;
2) перетворення S, яке складається з 8 перетворень S-блоків;
3) перестановка P.

Функція E розширює 32-бітовий вектор R_{i-1} до 48-бітового вектора $E(R_{i-1})$ шляхом дублювання деяких бітів R_{i-1}. За її допомогою вирішують два завдання: приводять розмір правої половини у відповідність із розміром ключа для виконання операції XOR, а також викликають лавинний ефект.

Порядок бітів вектора $E(R_{i-1})$ зазначений у таблиці 2.3.

Таблиця 2.3. Перестановка з розширенням

32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	1

Отриманий після розширення блок $E(R_{i-1})$ додається по модулю 2 із раундовими ключами K_i. Потім представляється у вигляді восьми послідовних блоків $B_1,\ B_2,\ \ldots, B_8$, тобто $E(R_{i-1}) \oplus K_i = B_1 B_2 \ldots B_8$.

Кожен B_j являється 6-бітовим блоком. Далі кожен з блоків B_j перетворюється у 4-бітовий блок $B_j^{'}$ за допомогою перетворень S_j. Перетворення S_j визначаються таблицею 2.4. Індекс j вказує, який з масивів S-боксу використовувати. Застосувавши операцію вибору до кожного із блоків B_j, одержимо 32-бітний блок $B_1^{'},\ B_2^{'},\ \ldots, B_8^{'}$.

Таблиця 2.4. S-бокси алгоритму DES

		Номер стовпця															
		0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	
Номер рядка	**0**	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7	**S1**
	1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8	
	2	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0	
	3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13	
	0	15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10	**S2**
	1	3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5	
	2	0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15	
	3	13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9	
	0	10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8	**S3**
	1	13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1	
	2	13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7	
	3	1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12	
	0	7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15	**S4**
	1	13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9	
	2	10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4	
	3	3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14	
	0	2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9	**S5**
	1	14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6	
	2	4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14	
	3	11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3	
	0	12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11	**S6**
	1	10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8	
	2	9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6	
	3	4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13	
	0	4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1	**S7**
	1	13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6	

		Номер стовпця															
		0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
	2	1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
	3	6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12
	0	13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
	1	1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
	2	7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
	3	2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11

S8

Приклад 2.5:

Припустимо, що B_3 = 101111. Знайдемо B_3' – ?

Перший і останній розряди B_3 – двійковий запис числа *a,* $0 \le a \le 3$.

Середні чотири розряди B_3 – двійковий запис числа *b*, $0 \le b \le 15$.

Пара чисел (*a*,*b*) визначає число, що знаходиться в перетині рядка *a* та стовпця *b*. Двійкове представлення цього числа дає B_3'.

У нашому випадку $a = 11_2 = 3$, $b = 0111_2 = 7$, а число обумовлене парою (3,7), дорівнює 7. Його двійкове представлення B_3' = 0111.

Отриманий блок $B_1', B_2', \ldots, B_8'$ перетворюється за допомогою матриці перестановки *P* (табл. 2.5).

Підстановка за допомогою S-боксів є ключовим кроком алгоритму DES. Інші операції алгоритму лінійні й легко піддаються аналізу. S-бокси нелінійні, і саме вони визначають безпеку DES. До того усі вісім S-бокси різні.

Таблиця 2.5 Матриця перестановки P

16	7	20	21	29	12	28	17
1	15	23	26	5	18	31	10
2	8	24	14	32	27	3	9
19	13	30	6	22	11	4	25

Таким чином, $f(R_{i-1}, K_i) = P(B_1', B_2', \ldots, B_8')$.

Генерація ключів

Ключ *K*– 64-бітний блок з вісьмома бітами контролю парності, що розміщені в позиціях 8, 16, 24, 32, 40, 48, 56, 64. Ще раз відзначимо, що на кожній ітерації використовується нове значення ключа $K_1, K_2, \ldots, K_{16}$, яке обчислюється із початкового значення ключа K.

Для видалення контрольних бітів і підготовки ключа до роботи використовується перестановка ключа (табл. 2.6).

Таблиця 2.6. Матриця перестановки ключа

57	49	41	33	25	17	9	C_0
1	58	50	42	34	26	18	
10	2	59	51	43	35	27	
19	11	3	60	52	44	36	
63	55	47	39	31	23	15	D_0
7	62	54	46	38	30	22	
14	6	61	53	45	37	29	
21	13	5	28	20	12	4	

Ця перестановка визначається двома блоками C_0 та D_0 по 28 біт кожний. Тобто 56-бітовий ключ ділиться на 2 половини, які потім циклічно зсуваються на один чи два біти ліворуч в залежності від етапу.

Тобто C_i, D_i, де i = 1,2,3,...,16 визначаються з C_{i-1}, D_{i-1}, одним або двома лівими циклічними зсувами згідно таблиці 2.7.

Таблиця 2.7. Матриця зсуву для обчислення ключів

Раунд	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Число зсуву	1	1	2	2	2	2	2	2	1	2	2	2	2	2	2	1

Після зсуву C_iD_i знову вибирається 48 бітів з 56 бітів та міняється їх порядок за наступною таблицею (табл. 2.8):

Таблиця 2.8. Матриця перестановки зі стисненням

14	17	11	24	1	5
3	28	15	6	21	10
23	19	12	4	26	8
16	7	27	20	13	2
41	52	31	37	47	55
30	40	51	45	33	48
44	49	39	56	34	53
46	42	50	36	29	32

Наприкінці шифрування виконується відновлення позицій бітів за допомогою матриці перестановок IP^{-1} (табл. 2.9).

Таблиця 2.9. Матриця кінцевої перестановки IP^{-1}

40	8	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

Дешифрування

При дешифруванні даних всі дії відбуваються в зворотному порядку. Ключі застосовуються в зворотному порядку. Функція *f*, перестановки *IP* і IP^{-1} такі самі як і в процесі шифрування.

Упродовж останніх десятиліть криптоаналітики інтенсивно досліджували DES. Зараз його вже не вважають стійким, в основному через недостатню довжину ключа. Для того щоб поліпшити безпеку DES, було запропоновано потрійний DES (Triple DES). Він використовує три каскади DES для зашифрування й розшифрування. Сьогодні використовуються дві версії потрійних DES: потрійний DES із двома ключами та потрійний DES із трьома ключами.

У потрійному DES із двома ключами є тільки два ключі K_1 і K_2 (рис. 2.7)

DES-EEE2 – шифрування виконується три рази, перший і третій каскади використовують K_1; другий каскад використовує K_2.

DES-EDE2 – на першому і третьому каскаді виконується зашифрування з ключем K_1; другий каскад застосовує розшифрування з ключем K_2.

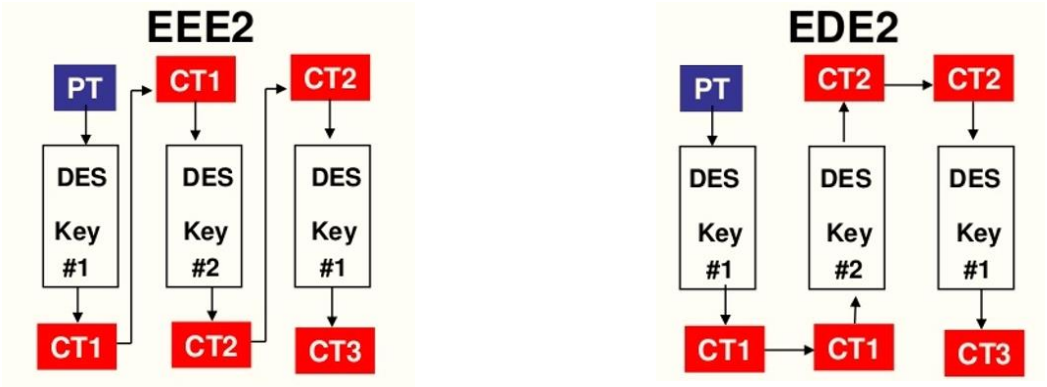


Рис. 2.7. Потрійний DES із двома ключами

Потрійний DES із трьома ключами використовує три різні ключі K_1, K_2 та K_3 (рис. 2.8).

DES-EEE3 – шифрування виконується три рази з трьома різними ключами

DES-EDE3 – виконуються операції шифрування, дешифрування та знову шифрування з трьома різними ключами.

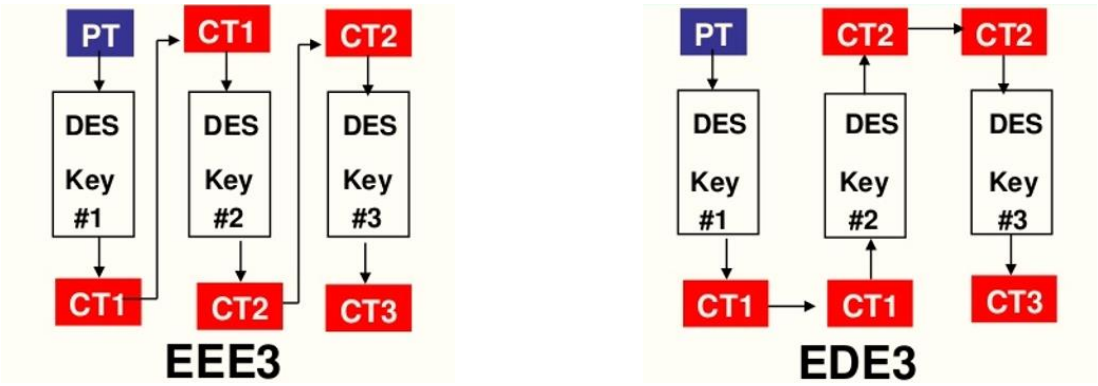


Рис. 2.8. Потрійний DES із трьома ключами

2.2.2. АЛГОРИТМ IDEA

IDEA (International Data Encryption Algorithm) – це алгоритм блокового симетричного шифрування, що був запропонований на заміну стандарту DES. Початкова версія алгоритму IDEA з'явилася у 1990 р. та мала назву PES (Proposed Encryption Standard). Через рік алгоритм був модифікований, щоб посилити криптостійкість до диференціального криптоаналізу. Нова версія отримала назву IPES (Improved PES – покращений PES), а ще через рік алгоритм змінив назву на IDEA.

Алгоритм IDEA оперує 64-бітовими блоками відкритого тексту, довжина ключ дорівнює 128 біт.

У IDEA використовуються такі математичні операції:

- ✓ $\boxplus$ – додавання – за модулем 2^{16};
- ✓ $\odot$ – множення за модулем $2^{16} + 1$;
- ✓ $\oplus$ – додавання за модулем 2^1 (XOR).

Комбінування цих трьох операцій забезпечує комплексне перетворення вхідних даних, суттєво ускладнюючи криптоаналіз IDEA у порівнянні з DES.

Процес шифрування складається з 8 раундів, з яких випливає завершальне перетворення.

Шифрування за алгоритмом IDEA

1. Початковий 64-бітний блок ділиться на чотири 16-бітних підблоки: D_1, D_2, D_3, D_4.

2. На кожному раунді чотири підблоки піддаються операціям XOR, додаванню і множенню один з одним та із шістьома 16-бітовими раундовими підключами.

3. Між раундами обмінюються місцями другий і третій підблоки.

4. В завершальному перетворенні чотири підблоки поєднуються із чотирма 16-бітовими підключами (другий і третій блоки не міняються місцями).

5. Після виконання усіх перетворень конкатенація підблоків D_1', D_2', D_3' та D_4' являє собою зашифрований блок.

На кожному раунді події відбуваються в наступній послідовності (рис. 2.9):

1. Перемножуються D_1 і перший підключ K_1.
2. Додаються D_2 і другий підключ K_2.
3. Додаються D_3 і третій підключ K_3.
4. Перемножуються D_4 і четвертий підключ K_4.
5. Виконується XOR над результатами етапів (1) і (3).
6. Виконується XOR над результатами етапів (2) і (4).
7. Перемножуються результати етапу (5) і п’ятий підключ K_5.
8. Додаються результати етапів (6) і (7).
9. Перемножуються результати етапу (8) і шостий підключ K_6.
10. Додаються результати етапів (7) і (9).
11. Виконується XOR над результатами етапів (1) і (9).
12. Виконується XOR над результатами етапів (3) і (9).
13. Виконується XOR над результатами етапів (1) і (10).
14. Виконується XOR над результатами етапів (4) і (10).

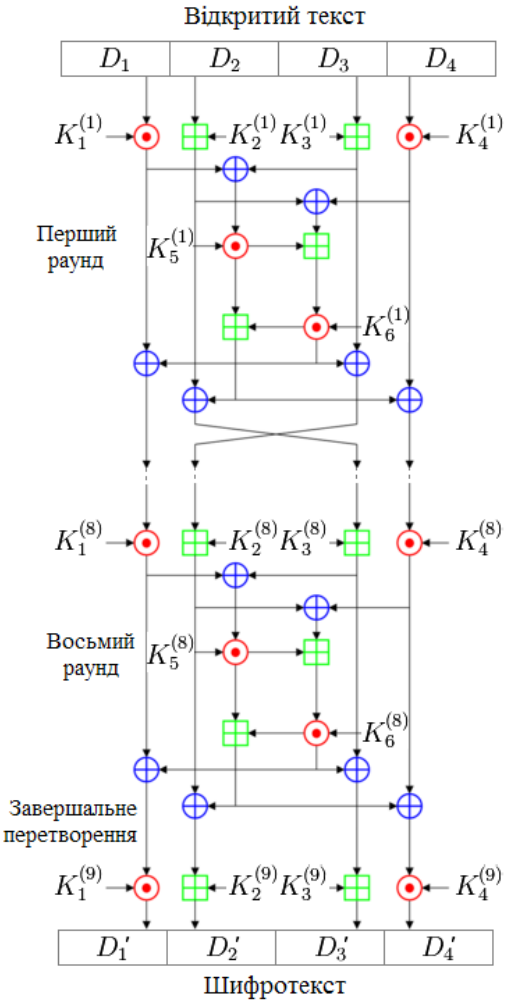


Рис. 2.9. Схема шифрування за алгоритмом IDEA

Генерація ключів

З 128-бітного ключа для кожного з восьми раундів шифрування генерується по шість 16-бітних підключів, а для завершального перетворення генерується чотири 16-бітних підключі. Усього буде потрібно 52 = 8 x 6 + 4 різних підключів по 16 біт кожен.

Процес генерації п’ятдесяти ключів полягає в наступному:

Початковий 128-бітний ключ розбивається на вісім підключів по 16 біт кожен: $K_1^{(1)}K_2^{(1)}K_3^{(1)}K_4^{(1)}K_5^{(1)}K_6^{(1)}K_1^{(2)}K_2^{(2)}$.

Далі початковий 128-бітний ключ циклічно зсувається на 25 позицій ліворуч, після чого знову розбивається на вісім 16-бітних підключів: $K_3^{(2)}K_4^{(2)}K_5^{(2)}K_6^{(2)}K_1^{(3)}K_2^{(3)}K_3^{(3)}K_4^{(3)}$.

Процедура циклічного зсуву і розбиття продовжується доти, поки не будуть згенеровані всі 52 16-бітних підключі (табл. 2.10).

Таблиця 2.10. Таблиця підключів для кожного раунду зашифрування

Номер раунду	**Підключі**
1	$K_1^{(1)}K_2^{(1)}K_3^{(1)}K_4^{(1)}K_5^{(1)}K_6^{(1)}$
2	$K_1^{(2)}K_2^{(2)}K_3^{(2)}K_4^{(2)}K_5^{(2)}K_6^{(2)}$
3	$K_1^{(3)}K_2^{(3)}K_3^{(3)}K_4^{(3)}K_5^{(3)}K_6^{(3)}$
4	$K_1^{(4)}K_2^{(4)}K_3^{(4)}K_4^{(4)}K_5^{(4)}K_6^{(4)}$
5	$K_1^{(5)}K_2^{(5)}K_3^{(5)}K_4^{(5)}K_5^{(5)}K_6^{(5)}$
6	$K_1^{(6)}K_2^{(6)}K_3^{(6)}K_4^{(6)}K_5^{(6)}K_6^{(6)}$
7	$K_1^{(7)}K_2^{(7)}K_3^{(7)}K_4^{(7)}K_5^{(7)}K_6^{(7)}$
8	$K_1^{(8)}K_2^{(8)}K_3^{(8)}K_4^{(8)}K_5^{(8)}K_6^{(8)}$
Завершальне перетворення	$K_1^{(9)}K_2^{(9)}K_3^{(9)}K_4^{(9)}$

Дешифрування

Процес дешифрування аналогічний процесу шифрування. Єдина відмінність полягає в тому, що для дешифрування використовуються інші підключі (табл. 2.11).

Мультиплікативна інверсія підключа K позначається $1/K$ та визначається рівнянням $(1/K) * K \equiv 1\, mod\, (2^{16}+1)$ Адитивна інверсія підключа K позначається $-K$ та $-K + K \equiv 0\, mod\, (2^{16})$.

Алгоритм IDEA не став міжнародним стандартом шифрування, однак його можна вважати одним із найпоширеніших у світі алгоритмів шифрування. IDEA використовується у багатьох додатках, у тому числі в досить відомій програмі захисту даних PGP.

Таблиця 2.11. Таблиця підключів для кожного раунду дешифрування

Номер раунду	**Підключі**
1	$1/K_1^{(9)}\ -K_2^{(9)}\ -K_3^{(9)}\ 1/K_4^{(9)}\ K_5^{(8)}\ K_6^{(8)}$
2	$1/K_1^{(8)}\ -K_3^{(8)}\ -K_2^{(8)}\ 1/K_4^{(8)}\ K_5^{(7)}\ K_6^{(7)}$
3	$1/K_1^{(7)}\ -K_3^{(7)}\ -K_2^{(7)}\ 1/K_4^{(7)}\ K_5^{(6)}\ K_6^{(6)}$
4	$1/K_1^{(6)}\ -K_3^{(6)}\ -K_2^{(6)}\ 1/K_4^{(6)}\ K_5^{(5)}\ K_6^{(5)}$
5	$1/K_1^{(5)}\ -K_3^{(5)}\ -K_2^{(5)}\ 1/K_4^{(5)}\ K_5^{(4)}\ K_6^{(4)}$
6	$1/K_1^{(4)}\ -K_3^{(4)}\ -K_2^{(4)}\ 1/K_4^{(4)}\ K_5^{(3)}\ K_6^{(3)}$
7	$1/K_1^{(3)}\ -K_3^{(3)}\ -K_2^{(3)}\ 1/K_4^{(3)}\ K_5^{(2)}\ K_6^{(2)}$
8	$1/K_1^{(2)}\ -K_3^{(2)}\ -K_2^{(2)}\ 1/K_4^{(2)}\ K_5^{(1)}\ K_6^{(1)}$
Завершальне перетворення	$1/K_1^{(1)}\ -K_2^{(1)}\ -K_3^{(1)}\ 1/K_4^{(1)}$

2.2.3. УДОСКОНАЛЕНИЙ СТАНДАРТ ШИФРУВАННЯ AES

У 1997 році Американський інститут стандартизації NIST (National Institute of Standards & Technology) оголосив конкурс на новий стандарт симетричного криптоалгоритму.

Згідно з вимогами конкурсу, алгоритм мав обов'язково:

- ✓ бути симетричним;
- ✓ бути блокових шифром;
- ✓ мати довжину блока 128 біт і підтримувати три довжини ключа: 128, 192 і 256 біт.

2 жовтня 2000 року NIST оголосив переможця. Ним став бельгійський алгоритм RIJNDAEL. У 2001 році алгоритм був затверджений як стандарт шифрування та отримав назву AES – Advanced Encryption Standard (удосконалений стандарт шифрування).

Математична база

Скінченне поле GF(2^8) складається з многочленів вигляду:

$$a(x) = a_7x^7 + a_6x^6 + a_5x^5 + a_4x^4 + a_3x^3 + a_2x^2 + a_1x + a_0, \text{ де } a_i \in \{0,1\}.$$

У вигляді многочлена $a(x)$ скінченого поля GF(2^8) можна подати будь-який байт, що складається з бітів $a_7a_6a_5a_4a_3a_2a_1a_0$.

Приклад 2.6:

Байт: 01011010.

Многочлен:

$$0 \cdot x^7 + 1 \cdot x^6 + 0 \cdot x^5 + 1 \cdot x^4 + 1 \cdot x^3 + 0 \cdot x^2 + 1 \cdot x^1 + 0 = x^6 + x^4 + x^3 + x.$$

Операції над елементами скінченого поля GF(2^8) вводяться наступним чином.

Додавання

$\forall a(x), b(x) \in GF(2^8)$

$$a(x) + b(x) = c(x) = c_7x^7 + c_6x^6 + c_5x^5 + c_4x^4 + c_3x^3 + c_2x^2 + c_1x + c_0,$$

де $c_i = a_i \oplus b_i, i = 0,1, \dots, 7$.

Приклад 2.7:

Додавання у двійковій формі:

$$\begin{array}{r} 10110001 \\ 10001111 \\ \hline 00111110. \end{array}$$

Те саме у вигляді многочленів:

$$(x^7 + x^5 + x^4 + 1) + (x^7 + x^3 + x^2 + x + 1) = x^5 + x^4 + x^3 + x^2 + x.$$

Множення

Щоб задати множення у полі GF(2^8), потрібно спочатку зафіксувати нерозкладний многочлен степеня 8 з коефіцієнтами із множини {0,1} (нерозкладність означає, що він ділиться лише на себе і на одиницю). Таких многочленів є декілька, автори AES вибрали такий:

$$m(x) = x^8 + x^4 + x^3 + x + 1 = 11B_{16}$$

Два елементи поля GF(2^8) множать за модулем $m(x)$ так:

1) Множать як звичайні многочлени.
2) Проміжний результат ділять на $m(x)$ і за остаточний результат приймають остачу від ділення.

Приклад 2.8:

1) $(x^6+x^5+x^4+x^2)\cdot(x^7+x^5+x^4+x)=x^{13}+x^{11}+x^{10}+x^7+x^{12}+x^{10}+x^9+$
$+x^6+x^{11}+x^9+x^8+x^5+x^9+x^7+x^6+x^3$
$=x^{13}+x^{12}+x^9+x^8+x^5+x^3.$

2)

$x^{13}+x^{12}+x^9+x^8+x^5+x^3$	$x^8+x^4+x^3+x+1$
$x^{13}+x^9+x^8+x^6+x^5$	x^5+x^4+1
$x^{12}+x^6+x^3$	
$x^{12}+x^8+x^7+x^5+x^4$	
$x^8+x^7+x^6+x^5+x^4+x^3$	
$x^8+x^4+x^3+x+1$	
$x^7+x^6+x^5+x+1$	.

Звідси

$$(x^{13}+x^{12}+x^9+x^8+x^5+x^3)\ mod\ (x^8+x^4+x^3+x+1)$$
$$=x^7+x^6+x^5+x+1.$$

Алгоритм AES

AES є симетричним ітеративним блоковим алгоритмом шифрування зі 128 довжиною блока та зі змінною довжиною ключа. Довжина ключа може дорівнювати 128, 192 або 256 бітів. На відміну від DES, алгоритм AES не використовує збалансовану мережу Фейстеля. AES базується на архітектурі SQUARE (КВАДРАТ), для якої характерно:

1) представлення блоку у вигляді масиву байтів;
2) шифрування за один раунд всього блоку даних;
3) виконання криптографічних перетворень, як над окремими байтами масиву, так і над його рядками і стовпцями.

Блок проміжного результату називають **станом**. Матриця стану має 4 рядки та 4 стовпці (Nb).

Матриця стану при Nb=4:

$$\begin{pmatrix} s_{0,0} & s_{0,1} & s_{0,2} & s_{0,3} \\ s_{1,0} & s_{1,1} & s_{1,2} & s_{1,3} \\ s_{2,0} & s_{2,1} & s_{2,2} & s_{2,3} \\ s_{3,0} & s_{3,1} & s_{3,2} & s_{3,3} \end{pmatrix}.$$

Основним елементом, яким оперує алгоритм AES, є байт – послідовність 8 біт, що обробляються як єдине ціле.

Задавати значення байта зручно в шістнадцятковій системі числення. Для цього байт ділиться на дві групи з 4-х біт: група старших біт в байті представляється першим шістнадцятковим символом, а група молодших біт – другим. Наприклад, для байта 10101100 отримаємо: 10101100 = 1010 1100 = AC.

Приклад 2.9:

Розглянемо перетворення тексту у матрицю:

Відкритий текст: A SECRET MESSAGE

У шістнадцятковому вигляді: 41 20 53 45 43 52 45 54 20 4D 45 53 53 41 47 45.

Отримаємо: $\begin{pmatrix} 41 & 43 & 20 & 53 \\ 20 & 52 & 4D & 41 \\ 53 & 45 & 45 & 47 \\ 45 & 54 & 53 & 45 \end{pmatrix}$.

Ключ шифру розглядають як матрицю байтів, яка має 4 рядки і кількість стовпців (*Nk*), що дорівнює довжині ключа, поділеній на 32.

Матриця ключа шифру при *Nk*=4:

$$\begin{pmatrix} k_{0,0} & k_{0,1} & k_{0,2} & k_{0,3} \\ k_{1,0} & k_{1,1} & k_{1,2} & k_{1,3} \\ k_{2,0} & k_{2,1} & k_{2,2} & k_{2,3} \\ k_{3,0} & k_{3,1} & k_{3,2} & k_{3,3} \end{pmatrix}.$$

Вхідні та вихідні дані розглядають як одновимірні масиви з індексами 0,...,*Nb*-1. Елементами масиву є байти. Ці блоки мають довжину 16, 24 або 32 байти.

Кількість циклів шифрування *Nr* залежить від значень *Nk*:

	Nk (Довжина ключа)	*Nb (Довжина блоку)*	*Nr (Кількість раундів)*
AES-128	4 (128)	4 (128)	10
AES-192	6 (192)		12
AES-256	8 (256)		14

Шифрування за алгоритмом AES

I. Початкового додавання раундового ключа.

II. *Nr-1* раундів, кожен з яких складається з чотирьох етапів:

1. Підстановка байтів;
2. Зсув рядків;

3. Перемішування стовпців;
4. Додавання раундового ключа.

III. Завершального раунду *Nr*, в якому пропускається перемішування стовпців. Розглянемо кожен з чотирьох етапів детальніше.

Підстановка байтів

Виконується окремо для кожного байта і складається з двох послідовних перетворень.

1. Байт розглядають як елемент поля GF(2^8). Якщо він ненульовий, до нього шукають обернений відносно множення в полі GF(2^8). Якщо ж байт нульовий, оберненого не існує. Тому нульовому байту 00000000 відповідає він сам.

2. Над утвореним байтом виконують таке перетворення:

$$\begin{pmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \\ y_7 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{pmatrix} \cdot \begin{pmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \end{pmatrix} \oplus \begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{pmatrix}.$$

На основі цих двох перетворень створено спеціальну таблицю замін байтів в шістнадцятковій системі, що називається *S*-боксом (табл. 2.12):

Таблиця 2.12. S-бокс алгоритму AES

		y															
		0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
x	0	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
	1	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
	2	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
	3	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
	4	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
	5	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
	6	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
	7	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
	8	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
	9	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
	a	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
	b	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
	c	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
	d	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
	e	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
	f	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

Приклад 2.10:

Отриману матрицю із прикладу 2.9 перетворимо за допомогою *S*-боксу:

$$\begin{pmatrix} 41 & 43 & 20 & 53 \\ 20 & 52 & 4D & 41 \\ 53 & 45 & 45 & 47 \\ 45 & 54 & 53 & 45 \end{pmatrix} \Rightarrow \begin{pmatrix} 83 & 1A & B7 & ED \\ B7 & 00 & E3 & 83 \\ ED & 6E & 6E & A0 \\ 6E & 20 & ED & 6E \end{pmatrix}.$$

Зсув рядків

Рядки стану циклічно зсувають на різні кількості байтів:

Nb	Кількість зсувів...			
	0-го рядка (-)	1-го рядка ($C1$)	2-го рядка ($C2$)	3-го рядка ($C3$)
4	0	1	2	3
6	0	1	2	3
8	0	1	3	4

Обернення етапу зсуву рядків полягає у циклічному зсуві праворуч трьох нижніх рядків на *Nb-C1*, *Nb-C2*, *Nb-C3* байтів відповідно.

Перемішування стовпців

Стовпці стану розглядають як многочлен над полем GF(2^8) та множать за модулем $x^4 + 1$ на фіксований многочлен $c(x)$:

$$c(x) = 03_{16} \cdot x^3 + 01_{16} \cdot x^2 + 01_{16} \cdot x + 02_{16}.$$

Якщо $a(x)$ – стовпець до застосування до нього перемішування, а $b(x)$ – після, то перетворення можна записати так:

$$b(x) = c(x) \otimes a(x),$$

або у матричному вигляді:

$$\begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{pmatrix} = \begin{pmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{pmatrix} \cdot \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix}.$$

Додавання раундового ключа

Побітове додавання за модулем 2 раундового ключа до відповідних бітів, отриманих у попередньому циклі. Раундовий ключ отримують з розширеного ключа шифру. Довжина раундового ключа дорівнює довжині блока *Nb*.

Генерація ключів

Розширений ключ – одновимірний масив 4-байтових слів – позначають $W[Nb \cdot (Nr + 1)]$.

Алгоритм розширення ключа при Nk ≤ 6

1. Перші *Nk* 4-байтових слів $W[i]$ послідовно вибираються з ключа шифру: 0-е слово – перші чотири байти, 1-е слово – другі чотири байти і т.д.

2. У слові $W[i-1]$ виконують циклічний зсув байтів за схемою: $(a,b,c,d) \Rightarrow (b,c,d,a)$, де a,b,c,d – байти.

3. Потім до кожного з 4-х байтів одержаного слова застосовують *S*-блок. До результату додають раундову сталу за модулем 2 (табл. 2.13).

Таблиця 2.13. Масив раундових констант Rcon

01	02	04	08	10	20	40	80	1B	36
00	00	00	00	00	00	00	00	00	00
00	00	00	00	00	00	00	00	00	00
00	00	00	00	00	00	00	00	00	00

4. Решту слів $W[i]$ визначають за формулою: $W[i] = W[i-Nk] \oplus W[i-1]$

При $Nk > 6$ виконується те саме, за винятком одного: якщо $i-4$ кратне Nk, то перед кроком 4 до кожного байта слова ще раз застосовують *S*-блок.

Дешифрування

I. Перед першим раундом дешифрування виконується операція додавання з ключем.

II. Потім виконується 9 раундів дешифрування, кожен з яких здійснює такі операції:

1. Зсув рядків в зворотному порядку. Байти в останніх трьох рядках матриці зсуваються циклічно вліво на різне число байт.

2. Обернена операція до операції підстановки байтів. Байти матриці замінюються новими значеннями за таблицею зворотної заміни, що є інвертованим *S*-боксом (табл. 2.14).

Таблиця 2.14. Інвертований S-бокс алгоритму AES

		y															
		0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
x	0	52	09	6a	d5	30	36	a5	38	bf	40	a3	9e	81	f3	d7	fb
	1	7c	e3	39	82	9b	2f	ff	87	34	8e	43	44	c4	de	e9	cb
	2	54	7b	94	32	a6	c2	23	3d	ee	4c	95	0b	42	fa	c3	4e
	3	08	2e	a1	66	28	d9	24	b2	76	5b	a2	49	6d	8b	d1	25
	4	72	f8	f6	64	86	68	98	16	d4	a4	5c	cc	5d	65	b6	92
	5	6c	70	48	50	fd	ed	b9	da	5e	15	46	57	a7	8d	9d	84
	6	90	d8	ab	00	8c	bc	d3	0a	f7	e4	58	05	b8	b3	45	06
	7	d0	2c	1e	8f	ca	3f	0f	02	c1	af	bd	03	01	13	8a	6b
	8	3a	91	11	41	4f	67	dc	ea	97	f2	cf	ce	f0	b4	e6	73
	9	96	ac	74	22	e7	ad	35	85	e2	f9	37	e8	1c	75	df	6e
	a	47	f1	1a	71	1d	29	c5	89	6f	b7	62	0e	aa	18	be	1b
	b	fc	56	3e	4b	c6	d2	79	20	9a	db	c0	fe	78	cd	5a	f4
	c	1f	dd	a8	33	88	07	c7	31	b1	12	10	59	27	80	ec	5f
	d	60	51	7f	a9	19	b5	4a	0d	2d	e5	7a	9f	93	c9	9c	ef
	e	a0	e0	3b	4d	ae	2a	f5	b0	c8	eb	bb	3c	83	53	99	61
	f	17	2b	04	7e	ba	77	d6	26	e1	69	14	63	55	21	0c	7d

3. Процедура, зворотна процедурі перемішування стовпців. Кожен стовпець матриці розглядається як 4-бітовий многочлен над полем GF(2^8) і множиться на фіксований многочлен:

$c^{-1}(x) = 0b_{16} \cdot x^3 + 0d_{16} \cdot x^2 + 09_{16} \cdot x + 0e_{16}$ по модулю многочлена x^4+1.

Таку операцію можна записати в матричному вигляді:

$$\begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix} = \begin{pmatrix} 0e & 0b & 0d & 09 \\ 01 & 0e & 0b & 0d \\ 0d & 09 & 0e & 0b \\ 0b & 0d & 01 & 0e \end{pmatrix} \cdot \begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{pmatrix}$$

4. Операція додавання з ключем по модулю 2.

III. Завершальний раунд не містить операцію перемішування стовпців.

2.2.4. НАЦІОНАЛЬНИЙ СТАНДАРТ ШИФРУВАННЯ ДСТУ 7624:2014

«Калина» – блоковий симетричний шифр, описаний у національному стандарті України ***ДСТУ 7624:2014*** *«Інформаційні технології. Криптографічний захист інформації* (введений в дію з 1 липня 2015 р.).

Основні характеристики

1) Спроектований на основі SP-мережі (AES);

2) Забезпечує захист від відомих методів криптоаналізу;

3) Має високу швидкодію на сучасних і перспективних програмних та програмно-апаратних платформах;

4) Визначає 10 режимів роботи.

Розміри блока даних можуть бути такими: 128, 256 або 512 бітів. **Матриця стану** має 8 рядків та Nb стовпців, що являють собою елементи поля $GF(2^8)$.

Матриця стану при Nb=2:

$$\begin{pmatrix} s_{0,0} & s_{0,1} \\ s_{1,0} & s_{1,1} \\ s_{2,0} & s_{2,1} \\ s_{3,0} & s_{3,1} \\ s_{4,0} & s_{4,1} \\ s_{5,0} & s_{5,1} \\ s_{6,0} & s_{6,1} \\ s_{7,0} & s_{7,1} \end{pmatrix}.$$

Довжина ключа може також бути 128, 256 або 512 бітів. **Ключ** шифру розглядають як матрицю байтів, яка має матриця байтів, яка має 8 рядків та *Nk* стовпців.

Матриця ключа шифру при *Nk*=4:

$$\begin{pmatrix} k_{0,0} & k_{0,1} & k_{0,2} & k_{0,3} \\ k_{1,0} & k_{1,1} & k_{1,2} & k_{1,3} \\ k_{2,0} & k_{2,1} & k_{2,2} & k_{2,3} \\ k_{3,0} & k_{3,1} & k_{3,2} & k_{3,3} \\ k_{4,0} & k_{4,1} & k_{4,2} & k_{4,3} \\ k_{5,0} & k_{5,1} & k_{5,2} & k_{5,3} \\ k_{6,0} & k_{6,1} & k_{6,2} & k_{6,3} \\ k_{7,0} & k_{7,1} & k_{7,2} & k_{7,3} \end{pmatrix}.$$

Кількість раундів шифрування алгоритму «Калина» (*Nr*) залежить від значень *Nb* і *Nk*:

Розмір блоку	***Кількість раундів шифрування для різних довжин ключа***		
	*Довжина ключа **128** бітів (Nk = 2)*	*Довжина ключа **256** бітів (Nk = 4)*	*Довжина ключа **512** бітів (Nk = 8)*
128 *(Nb = 2)*	10	14	–
256 *(Nb = 4)*	–	14	18
512 *(Nb = 8)*	–	–	18

Шифрування за алгоритмом «Калина»

I. Додавання з нульовим ключем по модулю 2^{64}.

II. *Nr-1* раундів, кожен з яких складається з чотирьох етапів:

1. Підстановка байтів;
2. Зсув рядків;
3. Перемішування стовпців;
4. Додавання раундового ключа по модулю 2

III. Завершальний раунд *Nr*, в якому замість $\oplus$ виконується додавання по модулю 2^{64}.

Розглянемо кожен з чотирьох етапів детальніше.

Додавання з нульовим ключем по модулю 2^{64}

Операція $\boxplus$ забезпечує побітове додавання раундового ключа до матриці стану за модулем 2^{64}. Фактично тут додають відповідні стовпці поточного стану

шифру та раундового ключа. При цьому, якщо результат додавання перевищує 8 біт, то беруться молодші 8 бітів, а старша часина відкидається.

Підстановка байтів

Кожен байт матриці стану замінюється відповідно до заданої таблиці підстановки (табл. 2.15). Задано (рекомендовано) чотири таблиці підстановок «байт-у-байт». Причому для байтів одного рядка поточного стану шифру застосовано одну й ту саму підстановку.

Заміна одного байту полягає у виборі з таблиці підстановки нового значення за адресою, яку задає поточне значення байту.

Таблиця 2.15. Підстановки алгоритму «Калина»

Підстановка π_0:

A8	43	5F	06	6B	75	6C	59	71	DF	87	95	17	F0	D8	09
6D	F3	1D	CB	C9	4D	2C	AF	79	E0	97	FD	6F	4B	45	39
3E	DD	A3	4F	B4	B6	9A	0E	1F	BF	15	E1	49	D2	93	C6
92	72	9E	61	D1	63	FA	EE	F4	19	D5	AD	58	A4	BB	A1
DC	F2	83	37	42	E4	7A	32	9C	CC	AB	4A	8F	6E	04	27
2E	E7	E2	5A	96	16	23	2B	C2	65	66	0F	BC	A9	47	41
34	48	FC	B7	6A	88	A5	53	86	F9	5B	DB	38	7B	C3	1E
22	33	24	28	36	C7	B2	3B	8E	77	BA	F5	14	9F	08	55
9B	4C	FE	60	5C	DA	18	46	CD	7D	21	B0	3F	1B	89	FF
EB	84	69	3A	9D	D7	D3	70	67	40	B5	DE	5D	30	91	B1
78	11	01	E5	00	68	98	A0	C5	02	A6	74	2D	0B	A2	76
B3	BE	CE	BD	AE	E9	8A	31	1C	EC	F1	99	94	AA	F6	26
2F	EF	E8	8C	35	03	D4	7F	FB	05	C1	5E	90	20	3D	82
F7	EA	0A	0D	7E	F8	50	1A	C4	07	57	B8	3C	62	E3	C8
AC	52	64	10	D0	D9	13	0C	12	29	51	B9	CF	D6	73	8D
81	54	C0	ED	4E	44	A7	2A	85	25	E6	CA	7C	8B	56	80

Підстановка π_1:

CE	BB	EB	92	EA	CB	13	C1	E9	3A	D6	B2	D2	90	17	F8
42	15	56	B4	65	1C	88	43	C5	5C	36	BA	F5	57	67	8D
31	F6	64	58	9E	F4	22	AA	75	0F	02	B1	DF	6D	73	4D
7C	26	2E	F7	08	5D	44	3E	9F	14	C8	AE	54	10	D8	BC
1A	6B	69	F3	BD	33	AB	FA	D1	9B	68	4E	16	95	91	EE
4C	63	8E	5B	CC	3C	19	A1	81	49	7B	D9	6F	37	60	CA
E7	2B	48	FD	96	45	FC	41	12	0D	79	E5	89	8C	E3	20
30	DC	B7	6C	4A	B5	3F	97	D4	62	2D	06	A4	A5	83	5F
2A	DA	C9	00	7E	A2	55	BF	11	D5	9C	CF	0E	0A	3D	51
7D	93	1B	FE	C4	47	09	86	0B	8F	9D	6A	07	B9	B0	98
18	32	71	4B	EF	3B	70	A0	E4	40	FF	C3	A9	E6	78	F9
8B	46	80	1E	38	E1	B8	A8	E0	0C	23	76	1D	25	24	05
F1	6E	94	28	9A	84	E8	A3	4F	77	D3	85	E2	52	F2	82
50	7A	2F	74	53	B3	61	AF	39	35	DE	CD	1F	99	AC	AD
72	2C	DD	D0	87	BE	5E	A6	EC	04	C6	03	34	FB	DB	59
B6	C2	01	F0	5A	ED	A7	66	21	7F	8A	27	C7	C0	29	D7

Підстановка π_2:

93	D9	9A	B5	98	22	45	FC	BA	6A	DF	02	9F	DC	51	59
4A	17	2B	C2	94	F4	BB	A3	62	E4	71	D4	CD	70	16	E1
49	3C	C0	D8	5C	9B	AD	85	53	A1	7A	C8	2D	E0	D1	72
A6	2C	C4	E3	76	78	B7	B4	09	3B	0E	41	4C	DE	B2	90
25	A5	D7	03	11	00	C3	2E	92	EF	4E	12	9D	7D	CB	35
10	D5	4F	9E	4D	A9	55	C6	D0	7B	18	97	D3	36	E6	48
56	81	8F	77	CC	9C	B9	E2	AC	B8	2F	15	A4	7C	DA	38
1E	0B	05	D6	14	6E	6C	7E	66	FD	B1	E5	60	AF	5E	33
87	C9	F0	5D	6D	3F	88	8D	C7	F7	1D	E9	EC	ED	80	29
27	CF	99	A8	50	0F	37	24	28	30	95	D2	3E	5B	40	83
B3	69	57	1F	07	1C	8A	BC	20	EB	CE	8E	AB	EE	31	A2
73	F9	CA	3A	1A	FB	0D	C1	FE	FA	F2	6F	BD	96	DD	43
52	B6	08	F3	AE	BE	19	89	32	26	B0	EA	4B	64	84	82
6B	F5	79	BF	01	5F	75	63	1B	23	3D	68	2A	65	E8	91
F6	FF	13	58	F1	47	0A	7F	C5	A7	E7	61	5A	06	46	44
42	04	A0	DB	39	86	54	AA	8C	34	21	8B	F8	0C	74	67

Підстановка π_3:

68	8D	CA	4D	73	4B	4E	2A	D4	52	26	B3	54	1E	19	1F
22	03	46	3D	2D	4A	53	83	13	8A	B7	D5	25	79	F5	BD
58	2F	0D	02	ED	51	9E	11	F2	3E	55	5E	D1	16	3C	66
70	5D	F3	45	40	CC	E8	94	56	08	CE	1A	3A	D2	E1	DF
B5	38	6E	0E	E5	F4	F9	86	E9	4F	D6	85	23	CF	32	99
31	14	AE	EE	C8	48	D3	30	A1	92	41	B1	18	C4	2C	71
72	44	15	FD	37	BE	5F	AA	9B	88	D8	AB	89	9C	FA	60
EA	BC	62	0C	24	A6	A8	EC	67	20	DB	7C	28	DD	AC	5B
34	7E	10	F1	7B	8F	63	A0	05	9A	43	77	21	BF	27	09
C3	9F	B6	D7	29	C2	EB	C0	A4	8B	8C	1D	FB	FF	C1	B2
97	2E	F8	65	F6	75	07	04	49	33	E4	D9	B9	D0	42	C7
6C	90	00	8E	6F	50	01	C5	DA	47	3F	CD	69	A2	E2	7A
A7	C6	93	0F	0A	06	E6	2B	96	A3	1C	AF	6A	12	84	39
E7	B0	82	F7	FE	9D	87	5C	81	35	DE	B4	A5	FC	80	EF
CB	BB	6B	76	BA	5A	7D	78	0B	95	E3	AD	74	98	3B	36
64	6D	DC	F0	59	A9	4C	17	7F	91	B8	C9	57	1B	E0	61

Зсув рядків

Рядки стану циклічно зсувають праворуч на різну кількість байтів, залежно від розміру блока (рис. 2.10):

Номер рядка	Значення зсуву, байтів		
	Довжина блоку 128 бітів	*Довжина блоку 256 бітів*	*Довжина блоку 512 бітів*
0	0	0	0
1	0	0	1
2	0	1	2

3	0	1	3
4	1	2	4
5	1	2	5
6	1	3	6
7	1	3	7

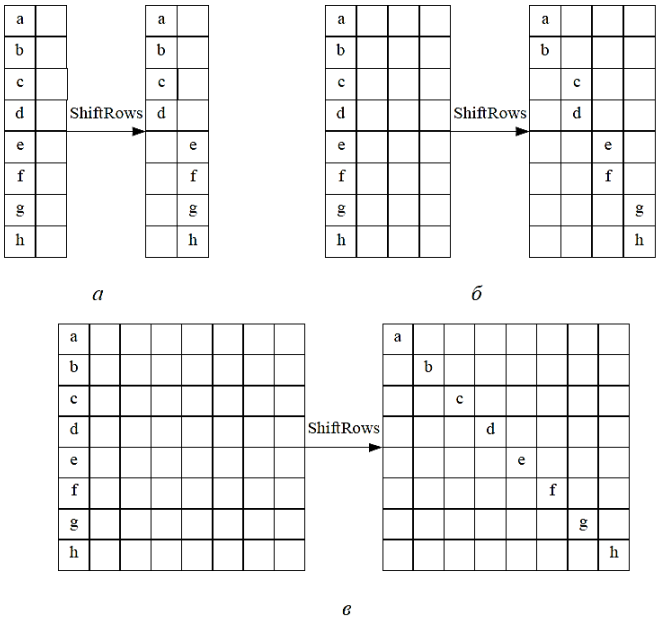


Рис. 2.10. Зсув рядків: а – 128-бітовий блок; б – 256-бітовий блок; в – 512-бітовий блок

Перемішування стовпців

Стовпці стану розглядають як многочлен над полем $GF(2^8)$ та множать за модулем $x^8 + 1$ на фіксований многочлен $c(x)$:

$$c(x) = 01_{16} \cdot x^7 + 05_{16} \cdot x^6 + 01_{16} \cdot x^5 + 08_{16} \cdot x^4 + 06_{16} \cdot x^3 + 07_{16} \cdot x^2 + 04_{16} \cdot x + 01_{16}.$$

Або у матричному вигляді:

$$\begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \end{pmatrix} = \begin{pmatrix} 01 & 01 & 05 & 01 & 08 & 06 & 07 & 04 \\ 04 & 01 & 01 & 05 & 01 & 08 & 06 & 07 \\ 07 & 04 & 01 & 01 & 05 & 01 & 08 & 06 \\ 06 & 07 & 04 & 01 & 01 & 05 & 01 & 08 \\ 08 & 06 & 07 & 04 & 01 & 01 & 05 & 01 \\ 01 & 08 & 06 & 07 & 04 & 01 & 01 & 05 \\ 05 & 01 & 08 & 06 & 07 & 04 & 01 & 01 \\ 01 & 05 & 01 & 08 & 06 & 07 & 04 & 01 \end{pmatrix} \cdot \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \\ a_7 \end{pmatrix}.$$

Для множення у полі $GF(2^8)$ алгоритм «Калина» використовує нерозкладний многочлен $m(x) = x^8 + x^4 + x^3 + x^2 + 1$.

Додавання раундового ключа

Побітове додавання за модулем 2 раундового ключа до відповідних бітів, отриманих у попередньому раунді.

Розгортання ключів

1. З ключа шифрування K формується допоміжний ключ K_t з довжиною, що дорівнює розміру блока ($64 \times Nb$ біт) з використанням трьох раундів зашифрування. Вхіднимим даними для перетворення є число $Nb + Nk + 1$ (у двійковому вигляді), інші байти заповнюються нулями. У якості раундових ключів використовується ключ шифрування K (якщо ключ довше блоку, використовується його молодша і старша половини).

2. На основі ключа K та допоміжного ключа K_t формуються раундові ключі K_{2i} (з парними індексами) довжиною, що дорівнює розміру блока ($64 \times Nb$ біт), з використанням двох раундів зашифрування для кожного раундового ключа. У якості раундових ключів використовується результат додавання по модулю 2^{64} допоміжного ключа K_t та змінної tmv_i – двійкове значення, яке залежить від індексу раундового ключа, який формується.

3. З раундових ключів K_{2i} з парними індексами формуються раундові ключі K_{2i+1} (з непарними індексами) шляхом циклічного зсуву попереднього ключа з парним індексом вліво на $2 \times Nb + 3$ байт.

Загалом використовується $Nr+1$ раундових ключів K_i ($i = 0,1 .., Nr$), кожен довжиною $64 \times Nb$ біт.

Дешифрування:

I. Виконуються операції з п. II, але на початку замість $\oplus$ виконується віднімання по модулю 2^{64} з ключем останнього раунду.

II. $Nr-1$ раундів, кожен з яких складається з чотирьох етапів:

1. Додавання раундового ключа за модулем 2;
2. Зворотна операція до перемішування стовпців;
3. Зсув рядків в зворотному порядку;
4. Обернена операція до підстановки байтів.

III. Віднімання з ключем нульового раунду по модулю 2^{64}.

Розглянемо кожен з чотирьох етапів детальніше.

Операція, зворотна операції перемішування стовпців

Стовпці стану множать на фіксований многочлен $c^{-1}(x)$ обернений до $c(x)$:

$$c^{-1}(x) = 95_{16} \cdot x^7 + 76_{16} \cdot x^6 + A8_{16} \cdot x^5 + 2F_{16} \cdot x^4 + 49_{16} \cdot x^3 + D7_{16} \cdot x^2 + CA_{16} \cdot x + AD_{16}.$$

Або у матричному вигляді:

$$\begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \end{pmatrix} = \begin{pmatrix} AD & 95 & 76 & A8 & 2F & 49 & D7 & CA \\ CA & AD & 95 & 76 & A8 & 2F & 49 & D7 \\ D7 & CA & AD & 95 & 76 & A8 & 2F & 49 \\ 49 & D7 & CA & AD & 95 & 76 & A8 & 2F \\ 2F & 49 & D7 & CA & AD & 95 & 76 & A8 \\ A8 & 2F & 49 & D7 & CA & AD & 95 & 76 \\ 76 & A8 & 2F & 49 & D7 & CA & AD & 95 \\ 95 & 76 & A8 & 2F & 49 & D7 & CA & AD \end{pmatrix} \cdot \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \\ a_7 \end{pmatrix}.$$

Зсув рядків в зворотному порядку

Рядки стану циклічно зсувають ліворуч на різну кількість байтів, залежно від розміру блока (рис. 2.11):

Номер рядка	Значення зсуву, байтів		
	Довжина блоку 128 бітів	*Довжина блоку 256 бітів*	*Довжина блоку 512 бітів*
0	0	0	0
1	0	0	1
2	0	1	2
3	0	1	3
4	1	2	4
5	1	2	5
6	1	3	6
7	1	3	7

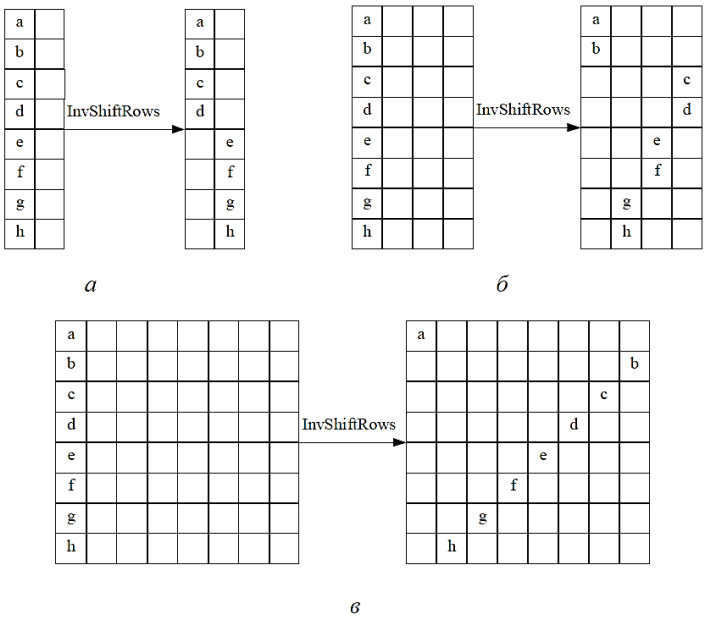


Рис 2.11. Зсув рядків в обереному порядку: а – 128-бітовий блок; б – 256-бітовий блок; в – 512-бітовий блок

Обернена операція до операції підстановки байтів

Кожен байт матриці стану замінюється відповідно до заданої таблиці зворотної заміни (табл. 2.16).

Таблиця 2.16. Обернені підстановки алгоритму «Калина»

Підстановка $_{-1}\pi_0$:

A4	A2	A9	C5	4E	C9	03	D9	7E	0F	D2	AD	E7	D3	27	
E3	A1	E8	E6	7C	2A	55	0C	86	39	D7	8D	B8	12	6F	
CD	8A	70	56	72	F9	BF	4F	73	E9	F7	57	16	AC	50	
9D	B7	47	71	60	C4	74	43	6C	1F	93	77	DC	CE	20	
99	5F	44	01	F5	1E	87	5E	61	2C	4B	1D	81	15	F4	
D6	EA	E1	67	F1	7F	FE	DA	3C	07	53	6A	84	9C	CB	
83	33	DD	35	E2	59	5A	98	A5	92	64	04	06	10	4D	
97	08	31	EE	AB	05	AF	79	A0	18	46	6D	FC	89	D4	
FF	F0	CF	42	91	F8	68	0A	65	8E	B6	FD	C3	EF	78	
CC	9E	30	2E	BC	0B	54	1A	A6	BB	26	80	48	94	32	
A7	3F	AE	22	3D	66	AA	F6	00	5D	BD	4A	E0	3B	B4	
8B	9F	76	B0	24	9A	25	63	DB	EB	7A	3E	5C	B3	B1	
F2	CA	58	6E	D8	A8	2F	75	DF	14	FB	13	49	88	B2	
E4	34	2D	96	C6	3A	ED	95	0E	E5	85	6B	40	21	9B	
19	2B	52	DE	45	A3	FA	51	C2	B5	D1	90	B9	F3	37	
0D	BA	41	11	38	7B	BE	D0	D5	69	36	C8	62	1B	82	

Підстановка $_{-1}\pi_1$:

83	F2	2A	EB	E9	BF	7B	9C	34	96	8D	98	B9	69	8C	
3D	88	68	06	39	11	4C	0E	A0	56	40	92	15	BC	B3	
6F	F8	26	BA	BE	BD	31	FB	C3	FE	80	61	E1	7A	32	
70	20	A1	45	EC	D9	1A	5D	B4	D8	09	A5	55	8E	37	
A9	67	10	17	36	65	B1	95	62	59	74	A3	50	2F	4B	
D0	8F	CD	D4	3C	86	12	1D	23	EF	F4	53	19	35	E6	
5E	D6	79	51	22	14	F7	1E	4A	42	9B	41	73	2D	C1	
A6	A2	E0	2E	D3	28	BB	C9	AE	6A	D1	5A	30	90	84	
B2	58	CF	7E	C5	CB	97	E4	16	6C	FA	B0	6D	1F	52	
0D	4E	03	91	C2	4D	64	77	9F	DD	C4	49	8A	9A	24	
A7	57	85	C7	7C	7D	E7	F6	B7	AC	27	46	DE	DF	3B	
9E	2B	0B	D5	13	75	F0	72	B6	9D	1B	01	3F	44	E5	
FD	07	F1	AB	94	18	EA	FC	3A	82	5F	05	54	DB	00	
E3	48	0C	CA	78	89	0A	FF	3E	5B	81	EE	71	E2	DA	
B8	B5	CC	6E	A8	6B	AD	60	C6	08	04	02	E8	F5	4F	
F3	C0	CE	43	25	1C	21	33	0F	AF	47	ED	66	63	93	

Підстановка $_{-1}\pi_2$:

45	D4	0B	43	F1	72	ED	A4	C2	38	E6	71	FD	B6	3A	
50	44	4B	E2	74	6B	1E	11	5A	C6	B4	D8	A5	8A	70	
A8	FA	05	D9	97	40	C9	90	98	8F	DC	12	31	2C	47	
99	AE	C8	7F	F9	4F	5D	96	6F	F4	B3	39	21	DA	9C	
9E	3B	F0	BF	EF	06	EE	E5	5F	20	10	CC	3C	54	4A	
94	0E	C0	28	F6	56	60	A2	E3	0F	EC	9D	24	83	7E	
7C	EB	18	D7	CD	DD	78	FF	DB	A1	09	D0	76	84	75	
1D	1A	2F	B0	FE	D6	34	63	35	D2	2A	59	6D	4D	77	
8E	61	CF	9F	CE	27	F5	80	86	C7	A6	FB	F8	87	AB	
3F	DF	48	00	14	9A	BD	5B	04	92	02	25	65	4C	53	
F2	29	AF	17	6C	41	30	E9	93	55	F7	AC	68	26	C4	
CA	7A	3E	A0	37	03	C1	36	69	66	08	16	A7	BC	C5	
22	B7	13	46	32	E8	57	88	2B	81	B2	4E	64	1C	AA	
58	2E	9B	5C	1B	51	73	42	23	01	6E	F3	0D	BE	3D	
2D	1F	67	33	19	7B	5E	EA	DE	8B	CB	A9	8C	8D	AD	
82	E4	BA	C3	15	D1	E0	89	FC	B1	B9	B5	07	79	B8	

Підстановка $_{-1}\pi_3$:

B2	B6	23	11	A7	88	C5	A6	39	8F	C4	E8	73	22	43	C3
82	27	CD	18	51	62	2D	F7	5C	0E	3B	FD	CA	9B	0D	0F
79	8C	10	4C	74	1C	0A	8E	7C	94	07	C7	5E	14	A1	21
57	50	4E	A9	80	D9	EF	64	41	CF	3C	EE	2E	13	29	BA
34	5A	AE	8A	61	33	12	B9	55	A8	15	05	F6	03	06	49
B5	25	09	16	0C	2A	38	FC	20	F4	E5	7F	D7	31	2B	66
6F	FF	72	86	F0	A3	2F	78	00	BC	CC	E2	B0	F1	42	B4
30	5F	60	04	EC	A5	E3	8B	E7	1D	BF	84	7B	E6	81	F8
DE	D8	D2	17	CE	4B	47	D6	69	6C	19	99	9A	01	B3	85
B1	F9	59	C2	37	E9	C8	A0	ED	4F	89	68	6D	D5	26	91
87	58	BD	C9	98	DC	75	C0	76	F5	67	6B	7E	EB	52	CB
D1	5B	9F	0B	DB	40	92	1A	FA	AC	E4	E1	71	1F	65	8D
97	9E	95	90	5D	B7	C1	AF	54	FB	02	E0	35	BB	3A	4D
AD	2C	3D	56	08	1B	4A	93	6A	AB	B8	7A	F2	7D	DA	3F
FE	3E	BE	EA	AA	44	C6	D0	36	48	70	96	77	24	53	DF
F3	83	28	32	45	1E	A4	D3	A2	46	6E	9C	DD	63	D4	9D

2.2.5. РЕЖИМИ РОБОТИ БЛОКОВИХ СИМЕТРИЧНИХ ШИФРІВ

Режим шифрування – метод застосування блокового шифру, що дозволяє перетворити послідовність блоків відкритих даних у послідовність блоків зашифрованих даних. При цьому для шифрування одного блоку можуть використовуватися дані іншого блоку. Зазвичай режими шифрування використовуються для модифікації процесу шифрування так, щоб результат шифрування кожного блоку був унікальним незалежно від даних, що шифруються і не дозволяв зробити будь-які висновки про їх структуру. Розглянемо основні режими роботи блокових шифрів, що найчастіше зустрічаються в системах криптографічного захисту даних.

Режим простої заміни (ECB, Electronic Coding Book). Це найпростіший режим роботи блокових шифрів – усі блоки відкритого тексту шифруються

окремо та незалежно один від одного із застосуванням того самого ключа шифрування (рис. 2.12). Крім того, обробка може бути розпаралелена, якщо використовуються кілька шифрувальних процесорів. Суттєвим недоліком є те, що при використанні одного ключа ідентичні блоки відкритого тексту перетворюються в ідентичні блоки зашифрованого тексту.

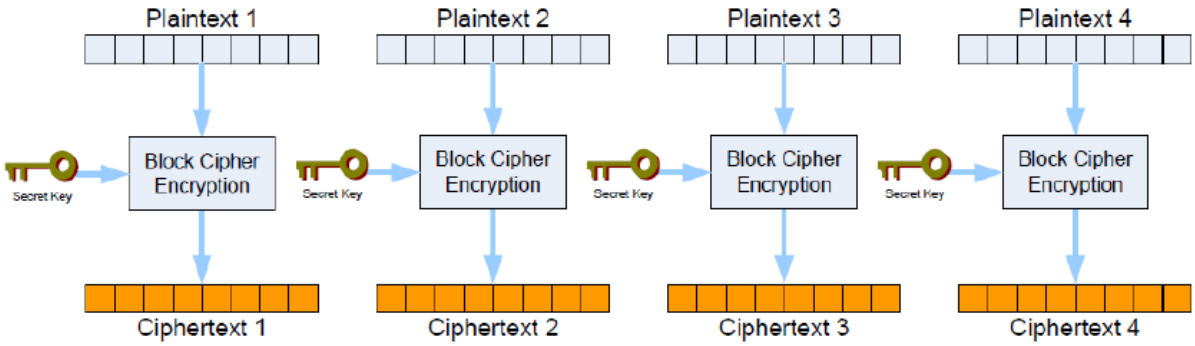


Рис. 2.12. Режим простої заміни (ECB, Electronic Coding Book)

Режим зв'язування блоків (CBC, Cipher Block Chaining). Для зв'язування використовується механізм зворотного зв'язку, оскільки результат шифрування попередніх блоків використовується для шифрування поточного блоку. Кожен блок додається за модулем 2 із попередньо зашифрованим блоком, а потім результат передається на вхід функції шифрування (рис. 2.13). Для шифрування першого блоку відкритого тексту використовують вектор ініціалізації IV (Initialization Vector) – послідовність випадкових символів, що дорівнює розміру блоку.

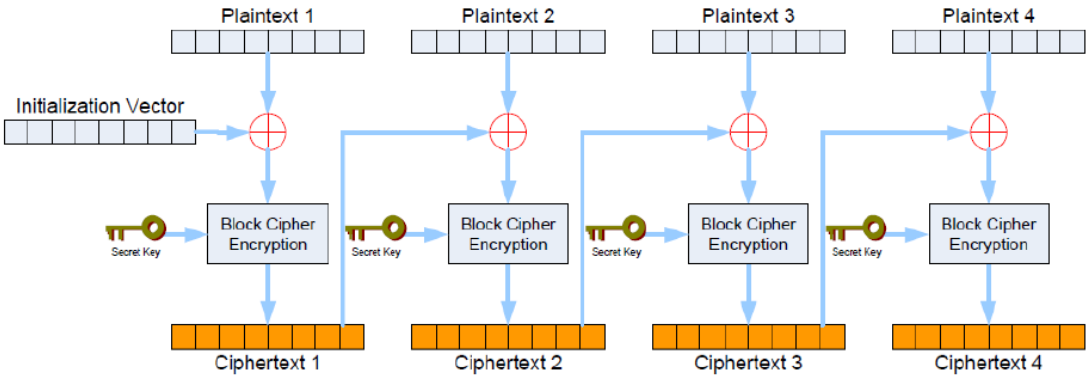


Рис. 2.13. Режим зв'язування блоків (CBC, Cipher Block Chaining)

Таким чином будь-який блок шифру залежить не тільки від початкового тексту, але і від усіх попередніх блоків тексту. Блоки з ідентичними початковими даними перетворюються у блоки із різними зашифрованими даними, проте

помилка в одному блоці може поширюватися на інші блоки. Недоліком режиму є те, що шифрування не піддається розпаралеленню.

Останній блок шифротексту в CBC залежить від IV, ключів і всіх бітів відкритого тексту, тому він може використовуватися для автентифікації повідомлення та відправника і називається кодом автентифікації повідомлення або MAC (Message Authentication Code).

Режим зі зворотнім зв'язком по шифротексту (CFB, Cipher Feedback). CFB перетворює блоковий шифр у потоковий, що самосинхронізується. При шифруванні до вектору ініціалізації IV застосовується функція шифрування для отримання першого вихідного блоку (першого блоку гами). Перший блок гами через операцію додавання за модулем 2 додається до першого блоку відкритого тексту, результатом є перший блок зашифрованого тексту (рис. 2.14). Функція шифрування застосовується знову до першого блоку шифротексту для отримання другого блоку гами. Отриманий другий блок гами додається за модулем 2 до другого блоку відкритого тексту, як результат – другий блок зашифрованого тексту. Функція шифрування викликається знову і операції повторюються.

З використанням CFB існує можливість шифрувати блоки довжиною менше *n* біт (не потрібне доповнення блоків). Помилка у відкритих даних впливає на всі подальші зашифровані дані, але самоусувається в ході розшифрування.

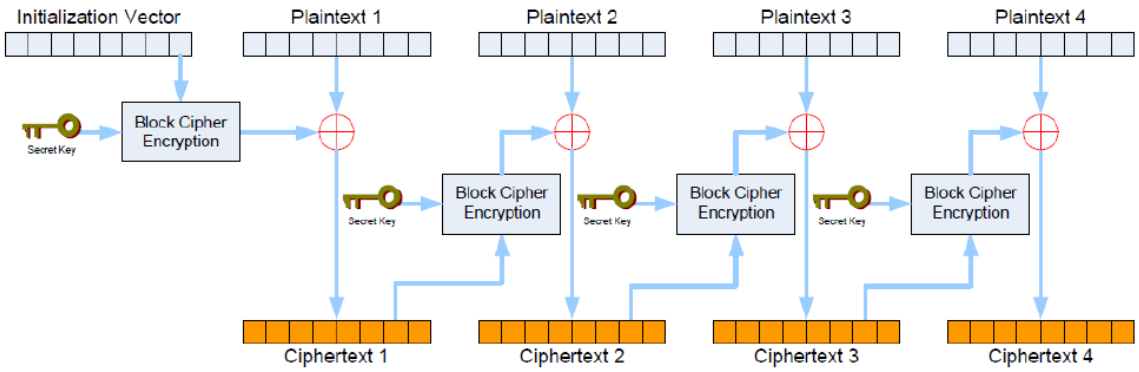


Рис. 2.14. Режим зі зворотнім зв'язком по шифротексту (CFB, Cipher Feedback)

Режим зі зворотнім зв'язком по виходу (OFB, Output Feedback). OFB перетворює блоковий шифр у синхронний потоковий шифр. Спочатку

відбувається шифрування вектору ініціалізації IV для генерації послідовності вихідних блоків (шифрограми), які додаються за модулем до відкритого тексту, щоб сформувати зашифрований текст (рис. 2.15). Операції виконуються із підмножиною бітів, що являє собою частину попередньо зашифрованого блоку. Результат одразу передається на наступний крок для отримання наступного блоку гами. Помилка у відкритих даних не впливає на всі подальші зашифровані дані. Режим OFB подібний до режиму CFB за винятком того, що блоки, які додають за модулем 2 із блоками відкритого тексту, генерують незалежно від відкритого або шифрованого тексту.

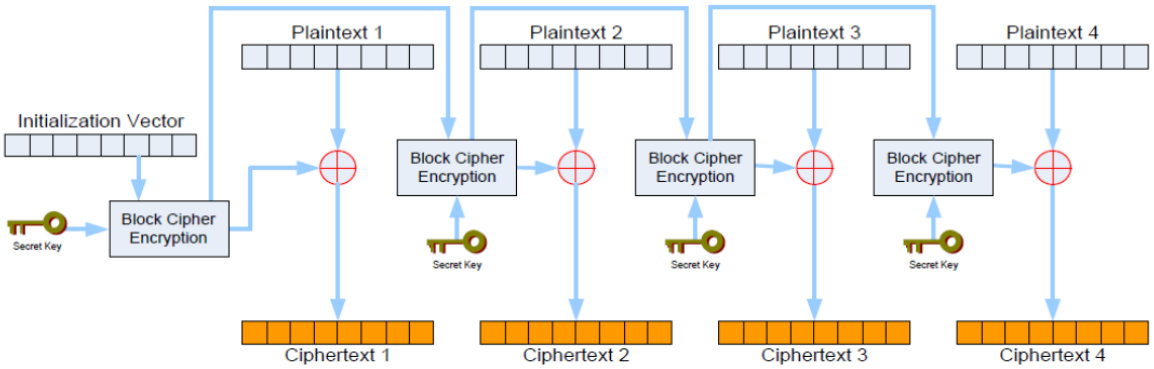


Рис. 2.15. Режим зі зворотнім зв'язком по виходу (OFB, Output Feedback)

Режим лічильника (CTR, Counter Mode). Режим CTR призначений для швидкого надійного шифрування блоків відкритого тексту із можливістю застосування паралельних обчислень. Цей режим використовує лічильник, значення якого додається за модулем до кожного блока відкритого тексту, який необхідно зашифрувати (рис. 2.16). Кожен блок на виході лічильника повинен бути відмінний від іншого блоку. Унікальне значення лічильника гарантує, що кожний блок зашифровується унікальним значенням ключового потоку.

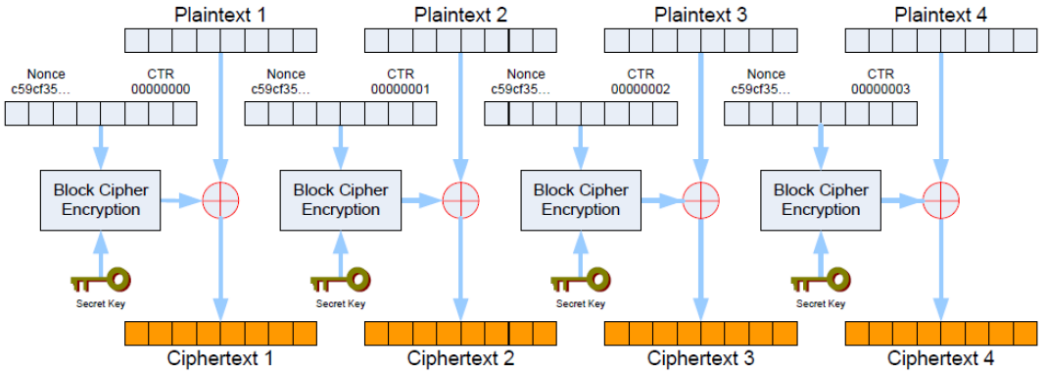


Рис. 2.16. Режим лічильника (CTR, Counter Mode)

Контрольні запитання до розділу 2

1. У чому полягає алгоритм одноразового блокноту?
2. Що являє собою операція XOR?
3. Які переваги і недоліки шифрування методом одноразового блокноту?
4. До яких шифрів належить стандарт шифрування даних DES?
5. Якою повинна бути довжина ключа у шифрі DES?
6. З яких кроків складається алгоритм шифрування DES?
7. Скільки разів виконується перетворення Фейстеля над блоком у DES?
8. Яка довжина блоку у алгоритмі IDEA?
9. Опишіть кроки шифрування за алгоритмом IDEA.
10. Яка довжина блоку в AES?
11. Як називають матрицю проміжного результату при шифруванні за допомогою алгоритму AES?
12. Опишіть операцію підстановки байтів у алгоритмі AES.
13. Опишіть операцію зсуву рядків у алгоритмі AES.
14. Опишіть операцію перемішування стовпців у алгоритмі AES.
15. Опишіть операцію додавання раундового ключа у алгоритмі AES.
16. Які особливості дешифрування за алгоритмом AES?
17. Від чого залежить кількість раундів шифрування за алгоритмом «Калина»?
18. Яка довжина ключа в алгоритмі «Калина»?
19. Як генерується допоміжний ключ в алгоритмі «Калина»?
20. Яким чином генеруються ключі з парними індексами в алгоритмі «Калина»?
21. Яким чином генеруються ключі з непарними індексами в алгоритмі «Калина»?
22. Скільки рядків має матриця стану в алгоритмі «Калина»?
23. Скільки таблиць замін використовується в криптографічному алгоритмі перетворення даних «Калина»?
24. Які особливості дешифрування за алгоритмом «Калина»?
25. Назвіть основні режими роботи блокових симетричних алгоритмів шифрування.

Тести до розділу 2

1. *Ключовий потік – це…*
   а) бітова послідовність, що визначається за допомогою ключа шифру
   б) бітова послідовність, що не залежить від ключа шифру
   в) бітова послідовність потоку відкритих даних, що потрібно зашифрувати
   г) бітова послідовність, що забезпечує захист даних від випадкових завад при передачі по каналах зв’язку
   д) бітова послідовність, що використовується для формування секретних ключів фіксованої довжини
2. *З якою метою використовують генератори псевдовипадкових чисел при потоковому шифруванні?*

а) для захисту інформації від випадкових перешкод при передачі

б) для захисту інформації від модифікації

в) для стискання інформації при шифруванні

г) для формування закритих та відкритих ключів

д) для отримання нескінченної ключової послідовності на основі ключа малої довжини

3. *Який з нижченаведених шифрів має теоретичну (абсолютну) стійкість?*
   а) шифр Віженера

   б) DES

   в) шифр Хілла

   г) AES

   д) шифр Вернама

4. *У шифрі Вернама (одноразового блокноту) виконується операція…*
   а) побітовий зсув

   б) додавання по модулю 2

   в) множення по модулю 2

   г) заміна байтів

   д) перестановка символів

5. *Відкритий виглядає як 00101010. Чому буде дорівнювати шифротекст, якщо ключ одноразового блокноту 10001010?*
   а) 10100000

   б) 11100011

   в) 11111111

   г) 10101010

   д) немає правильної відповіді

6. *Матриця стану алгоритму RC4 має розмір…*
   а) 128 байт

   б) 255 байт

   в) 256 байт

   г) 512 байт

   д) 1024 байти

7. *Довжина раундових ключів DES складає…*
   а) 56 біт

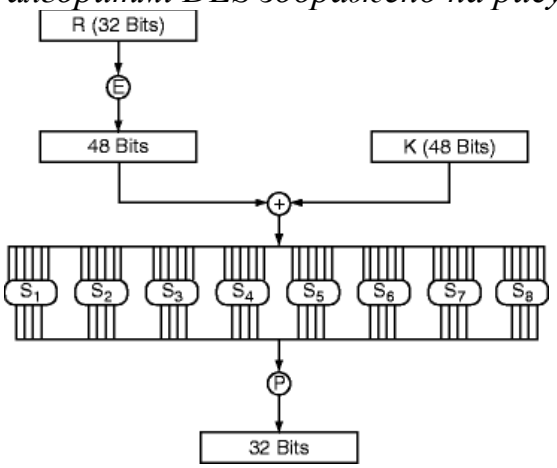
   б) 128 біт

   в) 256 біт

   г) 192 біти

д) 48 біт

8. *Скільки раундів виконується шифрування за алгоритмом DES?*
   а) 8
   б) 12
   в) 10
   г) 16
   д) 14
9. *Яке перетворення у алгоритмі DES зображено на рисунку?*



   а) початкова перестановка
   б) функція Фейстеля
   в) кінцева перестановка
   г) раунд зашифрування
   д) генерація ключів
10. *Скільки S-боксів у DES?*
   а) 1
   б) 4
   в) 6
   г) 7
   д) 8
11. *Довжина ключа IDEA складає…*
   а) 56 біт
   б) 128 біт
   в) 256 біт
   г) 192 біти
   д) 48 біт
12. *Скільки потрібно підключів на один раунд шифрування у алгоритмі IDEA?*
   а) 1
   б) 3

в) 4

г) 6

д) 8

13. Який многочлен у полі GF(28) відповідає байту F8 у 16-вій системі числення?

а) $x^6 + x^5 + x^3 + x^2 + x + 1$

б) $x^7 + x^5 + x^3 + x^2 + 1$

в) $x^8 + x^7 + x^6 + x^5 + x^4$

г) $x^7 + x^6 + x^5 + x^4 + x^3 + x^2 + x + 1$

д) $x^7 + x^6 + x^5 + x^4 + x^3$

14. Яку кількість слів містить розширений ключ AES-128?

а) 36 слів

б) 68 слів

в) 52 слова

г) 60 слів

д) 44 слова

15. Наведіть наступні чотири дії в порядку, за яким відбувається дешифрування за алгоритмом AES:

4 – Операція, обернена до перемішування стовпців (InvMixColumns);

1 – Зсув рядків в зворотному порядку (InvShiftRows);

3 – Додавання раундового ключа (AddRoundKey);

2 – Обернена операція до підстановки байтів (InvSubBytes)

а) (1, 2, 4, 3)

б) (3, 4, 2, 1)

в) (3, 2, 1, 4)

г) (1, 2, 3, 4)

д) (2, 4, 1, 3)

16. Скільки рядків має матриця стану у алгоритмі «Калина»?

а) 6

б) 12

в) 4

г) 8

д) 10

17. Скільки разів виконується операція зашифрування у процесі формування допоміжного ключа в алгоритмі «Калина»?

а) 4

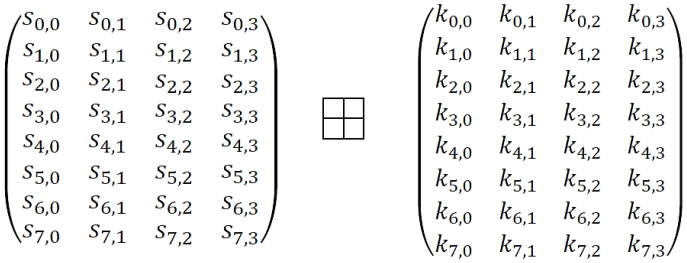
б) 5

в) 3

г) 2

д) 10

18. *Яке перетворення алгоритму «Калина» зображено на рисунку?*



а) Зсув рядків

б) Підстановка байтів

в) Додавання раундового ключа за модулем 2

г) Перемішування стовпців

д) Додавання раундового ключа за модулем 264

19. *У чому найбільша перевага режиму зв'язування блоків (CBC)?*

а) існує можливість шифрувати блоки довжиною менше n біт

б) блок зашифрованого тексту залежить від усіх попередніх блоків

в) помилки в одному блоці не поширюються на інші блоки

г) не потрібен вектор ініціалізації (IV)

д) існує можливість паралельно шифрувати різні блоки тексту

20. *Вектор ініціалізації у режимі зв'язування блоків (CBC) застосовують...*

а) для відмежування першого блока шифрованого тексту

б) для визначення розмірів першого блока відкритого тексту

в) для спеціальної обробки останнього блока шифрованого тексту

г) для підвищення швидкості шифрування

д) при шифруванні першого блоку відкритого тексту

Задачі до розділу 2

1. Здійсніть S-перетворення блоку 111001 з використанням третього S-боксу DES.
2. Здійсніть S-перетворення блоку 101101 з використанням восьмого S-боксу DES.
3. Для заданого півблоку в алгоритмі DES виконайте його додавання із ключем та завершіть S-перетворення:

⊕	01001001010011000011101110011111000011110011101 0
	01101010111111111010101111010101001111000000001
	?

S1	S2	S3	S4	S5	S6	S7	S8
0010	**?**	0101	0001	0101	0011	**?**	0101

4. В алгоритмі AES знайдіть наступне слово ключа дев’ятого раунду, якщо дано ключ восьмого раунду:

90	40	34	10
F3	DD	F0	49
52	90	C9	66
4C	13	DF	4C

5. В алгоритмі AES знайдіть 2 слово ключа четвертого раунду, якщо дано ключ третього раунду:

FA	E4	1F	53
49	D2	58	9A
E9	65	94	48
C1	2C	87	3E

6. В алгоритмі AES обчисліть:

02	03	01	01
01	02	03	01
01	01	02	03
03	01	01	02

·

63	37	7B	BC
84	51	8E	CF
AC	FD	D1	CA
83	B5	FB	AD

=

7E	D5	55	4E
1C	3C	EF	D1
3A	43	**?**	10
90	84	3F	9B

7. В алгоритмі AES обчисліть:

02	03	01	01
01	02	03	01
01	01	02	03
03	01	01	02

·

7B	D5	27	99
44	26	1B	3A
12	98	89	2D
F1	4B	31	FA

=

D9	08	D8	B0
34	61	A0	60
13	05	66	EC
22	4C	99	**?**

8. В алгоритмі «Калина» виконайте заміну байтів:

5A	8E
89	9F
20	5F
CB	87
94	80
C2	87
8F	B3
2C	AF

РОЗДІЛ 3. АСИМЕТРИЧНІ КРИПТОГРАФІЧНІ СИСТЕМИ

3.1. КРИПТОСИСТЕМИ З ВІДКРИТИМ КЛЮЧЕМ

Якими б не були надійними та швидкими симетричні криптографічні системи – їх слабким місцем, під час практичної реалізації, є проблема обміну ключами. Для вирішення цієї і ряду інших проблем були запропоновані криптосистеми з відкритим ключем, які називають також асиметричними криптосистемами.

Концепція криптографії з відкритим ключем була висунута Вітфілдом Діффі (Whitfield Diffie) та Мартіном Хелманом (Martin Hellman), і окремо Ральфом Мерклом (Ralph Merkle). У *асиметричних криптосистемах* для шифрування використовується один, відкритий (публічний, загальнодоступний) ключ, а для дешифрування – інший, закритий (секретний, приватний). Закритий ключ та відкритий ключ – це два великі числа, обчислені на основі деякого асиметричного алгоритму. Відкритий може бути доступним будь-якому учаснику процесу інформаційного обміну. При чому, знання відкритого ключа не дозволяє обчислити відповідний закритий ключ.

Ідея криптографії з відкритим ключем дуже тісно пов'язана з ідеєю *однобічних функцій*, тобто таких функцій $f(x)$, що по відомому x досить просто знайти значення $f(x)$, тоді як визначити x з $f(x)$складно (рис. 3.1).

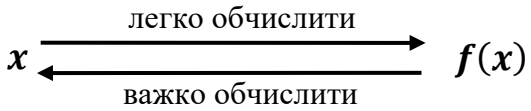


Рис. 3.1. Схема роботи однобічних функцій

Також використовуються *однобічні функції з лазівкою*. Лазівка – це певний секрет, що допомагає розшифрувати. Тобто існує такий y, що знаючи $f(x)$, можна обчислити x.

3.1.1. КРИПТОСИСТЕМА МЕРКЛА-ХЕЛМАНА

Першим алгоритмом для шифрування із відкритим ключем став алгоритм Меркла-Хелмана або алгоритм рюкзака. Безпека цього, а також інших ранцевих алгоритмів, ґрунтується на складній задачі пакування рюкзака. Із часом було виявлено, що алгоритм рюкзака Меркла-Хелмана не є криптостійкий. Проте цей

алгоритм ілюстративний і на його прикладі, як правило, розглядають основні поняття та інструменти асиметричної криптографії.

Задача рюкзака: Дано набір предметів різної маси. Чи можна покласти деякі із цих предметів у рюкзак так, щоб маса рюкзака дорівнювала певному значенню? Більш формально, дано набір значень $M_1, M_2, \dots M_n$ і сума S. Необхідно знайти значення b_i, такі що $S = M_1 b_1 + M_2 b_2 + \dots + M_n b_n$, де $b_i \in \{0,1\}$, тобто b_i може бути або нулем, або одиницею. Одиниця вказує на те, що предмет кладуть у рюкзак, а нуль – що не кладуть.

Наприклад, маси предметів можуть мати значення 1, 5, 6, 11, 14 і 20. Ми можемо спакувати рюкзак так, щоб його маса дорівнюватиме 22, використавши маси 5, 6 і 11. Неможливо впакувати рюкзак так, щоб його маса дорівнювала 24.

В основі алгоритму Меркла-Хеллмана лежить ідея шифрування повідомлення як розв'язання задачі рюкзака. При зашифруванні предмети до рюкзака вибирають за допомогою блока відкритого тексту. Довжина блока в бітах дорівнює кількості предметів, з яких вибирають. Біти блока відкритого тексту відповідають значенням b_i, а шифротекстом кожного блока є отримана маса. Таким чином $M_1, M_2, \dots M_n$ – маси предметів, $b_1, b_2, \dots b_n$ – відкритий текст (у бінарному вигляді), S – шифротекст.

Приклад 3.1:

Зашифруємо чотири блоки відкритого тексту по шість бітів у кожному, якщо маси предметів, з яких деякі кладуть у рюкзак мають значення: 1, 5, 6, 11, 14, 20:

Відкритий текст	111001	010110	000000	011000
Рюкзак	1 5 6 11 14 20	1 5 6 11 14 20	1 5 6 11 14 20	1 5 6 11 14 20
Шифротекст	1+5+6+20=32	5+11+14=30	0=0	5+6=11

Таким чином, у розглянутому прикладі відкритому бінарному тексту 111001 010110 000000 011000 відповідає шифротекст: 32, 30, 0, 11.

Очевидно, що зашифрування виконати легко. А от під час розшифрування для кожного блока шифртексту потрібно розв'язувати складну задачу пакування рюкзака. Звичайно, у розглянутому простому прикладі цю задачу розв'язати відносно просто. Адже предметів для вибору є небагато, і перебором швидко можна встановити набір предметів, сума мас яких становитиме задану масу

рюкзака. Але якщо кількість предметів для вибору велика, процедура дешифрування різко ускладнюється. Час, необхідний для розв'язання задачі пакування рюкзака, у загальному випадку зростає експоненційно зі збільшенням кількості предметів, з яких вибирають.

Загалом існує дві різні задачі рюкзака: легка та складна. Якщо перелік мас предметів являє собою суперзростаючу послідовність, то задачу рюкзака легко розв'язати. Якщо перелік мас предметів являє собою нормальну послідовність, то задачу рюкзака розв'язати важко. Легку задачу можна перетворити у складну.

Суперзростаюча послідовність – це послідовність, у якій кожної член більший за суму усіх попередніх членів. Наприклад, послідовність {1, 3, 6, 13, 27, 52} є суперзростаючою.

Нормальна послідовність – це послідовність, що містить довільні елементи. Наприклад, послідовність {1, 3, 4, 9, 15, 25} не є суперзростаючою, тобто вона нормальна.

Алгоритм розв'язання задачі суперзростаючого рюкзака

1. Повну масу рюкзака порівнюємо з найбільшим числом послідовності.

2. Якщо повна маса менша за це число, то його не кладемо у рюкзак.

3. Якщо повна маса більша або дорівнює цьому числу, то воно кладеться у рюкзак. Зменшуємо масу рюкзака на це значення.

4. Переходимо до наступного по величині числа послідовності.

5. Будемо повторювати, поки процес не закінчиться. Якщо повна маса зменшиться до нуля, то розв'язок знайдений.

Приклад 3.2:

Повна маса рюкзака – 70, послідовність мас {2, 3, 6, 13, 27, 52}

1. Найбільша маса – 52 < 70 ⇨ кладемо 52 у рюкзак.

2. Віднімаємо: 70 – 52=18.

3. Наступна маса – 27 > 18 ⇨ 27 у рюкзак не кладемо.

4. Маса 13 < 18 ⇨ кладемо 13 у рюкзак.

5. Віднімаємо: 18 – 13=5.

6. Наступна маса – 6 > 5 ⇨ 6 не кладемо у рюкзак.

Продовження цього процесу покаже, що й 2, і 3 кладемо у рюкзак, і повна маса зменшується до 0, що повідомляє про знайдений розв'язок.

Відкритий текст: 110101.

Криптосистема Меркла-Хелмана ґрунтується на розглянутих властивостях. *Відкритий ключ* являє собою нормальну послідовність, яку легко використовувати для шифрування, але складно для дешифрування повідомлень. *Закритий ключ* є суперзростаючою послідовністю, яку можна використовувати для дешифрування повідомлень за описаним вище алгоритмом розв'язання задачі суперзростаючого рюкзака.

Створення відкритого ключа із закритого

1. Генерується суперзростаюча послідовність.

2. Обирається число m (модуль), більше за суму усіх чисел послідовності.

3. Знаходиться n взаємно просте з m.

4. Усі значення суперзростаючої послідовності множаться по модулю m на число n.

Приклад 3.3:

Дано закритий ключ – суперзростаюча послідовність {2, 3, 6, 13, 27, 52}, m = 105, n = 31.

Нормальною послідовністю буде:

$2 \cdot 31 \; mod \; 105 \; = 62;$

$3 \cdot 31 \; mod \; 105 \; = 93;$

$6 \cdot 31 \; mod \; 105 \; = 81;$

$13 \cdot 31 \; mod \; 105 \; = 88;$

$27 \cdot 31 \; mod \; 105 \; = 102;$

$52 \cdot 31 \; mod \; 105 \; = 37;$

Відкритий ключ – {62, 93, 81, 88, 102, 37}.

Шифрування у криптосистемі Меркла-Хелмана

1. Розбити повідомлення на блоки, рівні по довжині кількості елементів послідовності рюкзака.

2. Вважати, що у відкритому тексті одиниця вказує на присутність члена послідовності, а нуль – на його відсутність.

3. Обчислити повні маси рюкзака – по одному для кожного блоку повідомлення.

Приклад 3.4:

Дано повідомлення в бінарному виді 011000110101101110, відкритий ключ – послідовність {62, 93, 81, 88, 102, 37}

Зашифруємо повідомлення 011000 110101 101110.

011000 відповідає 93 + 81 = 174;

110101 відповідає 62 + 93 + 88 + 37 = 280;

101110 відповідає 62 + 81 + 88 + 102 = 333;

Шифротекст: послідовність {174, 280, 333}.

Дешифрування у криптосистемі Меркла-Хелмана

Обчислити повні маси рюкзака – по одному для кожного блоку повідомлення.

1. Спочатку визначають n^{-1}, таке що $n(n^{-1}) \equiv 1\ (mod\ m)$.

2. Кожне значення шифротексту множиться на $n^{-1}\ mod\ m$.

3. Одержати значення відкритого тексту за допомогою закритого ключа – одиниця вказує на присутність члена послідовності, а нуль – на його відсутність.

Приклад 3.5:

Дано шифротекст {174, 280, 333}, закритий ключ – {2, 3, 6, 13, 27, 52}, m =105, n = 31. У нашому випадку n^{-1} дорівнює 61, тому значення шифротекста помножимо на 61 за модулем 105.

$174 \cdot 61\ mod\ 105\ = 9 = 3 + 6$, що відповідає 011000;

$80 \cdot 61\ mod\ 105\ = 70 = 2 + 3 + 13 + 52$, що відповідає 110101;

$333 \cdot 61\ mod\ 105\ = 48 = 2 + 6 + 13 + 27$, що відповідає 101110.

Відкритий текст: 011000 110101 101110.

3.1.2. АЛГОРИТМ ШИФРУВАННЯ RSA

Найбільш простим для розуміння та реалізації є алгоритм з відкритим ключем RSA, названий на честь трьох авторів – Рона Рівеста (Ron Rivest), Аді Шаміра (Adi Shamir) і Леонарда Едлмана (Leonard Adleman).

Безпека RSA заснована на складності розкладання на множники великих чисел. Відкритий і закритий ключі є функціями двох великих простих чисел розрядністю 100…200 десяткових цифр і навіть більше. Відновлення відкритого тексту за шифртекстом та відкритим ключем є рівнозначне до розкладання числа на два великі прості множники.

Генерація ключів

1. Вибираються два великих випадкових простих числа, p і q (для максимальної безпеки p і q варто обирати рівної довжини).

2. Обчислюється добуток (модуль системи): $n = p \cdot q$.

3. Обчислюється функція Ейлера $\varphi(n) = \varphi(pq) = (p-1)(q-1)$. Результат розрахунку даної функції дорівнює кількості додатних чисел, які не більше n і взаємно прості з n.

4. Випадковим чином вибирається число e (ключ шифрування), таке що $1 < e < \varphi(n)$ та взаємно просте з $\varphi(n)$.

5. За допомогою розширеного алгоритму Евкліда знаходиться число d (ключ дешифрування), таке що $ed \equiv 1(mod\ \varphi(n))$.

6. Пара $(e,\ n)$ публікується у якості *відкритого ключа*.

7. Пара $(d,\ n)$ виконує роль *закритого ключа* і тримається таємниці.

Два простих числа p і q більше не потрібні. Проте вони не повинні бути розкриті.

Шифрування

Для шифрування повідомлення *M* воно спочатку розбивається на цифрові блоки, менші n (для двійкових даних вибирається найбільший степінь числа 2, менший n). Зашифроване повідомлення *C* буде складатися із блоків c_i. Формула шифрування виглядає наступним чином: $c_i = m_i^e\ mod\, n$.

Дешифрування

Для дешифрування повідомлення візьмемо кожний зашифрований блок c_i і обчислимо: $m_i = c_i^d\ mod\, n$.

Приклад 3.6:

Зашифруємо повідомлення КНИГА, що складається із символів українського алфавіту та представляється як послідовність цілих чисел M = 14 17 10 3 0.

Для простоти обчислень будемо використовувати невеликі числа, проте пам'ятаємо, що на практиці застосовують дуже великі прості числа. Оберемо $p = 3$ і $q = 11$, тоді $n = p \cdot q = 3 \cdot 11 = 33$.

Обчислимо $\varphi(33) = 2 \cdot 10 = 20$.

Виберемо (випадково) $e = 3$ та перевіримо виконання умов: $1 < 3 < 20$, НСД(3, 20) =1.

Визначимо d – ключ дешифрування з рівняння $3d \equiv 1(mod\ 20)$.

Для розв'язання рівняння використаємо *розширений алгоритм Евкліда*:

1) послідовно виконуємо ділення з остачею попереднього значення r_{i-1} на наступне r_i, у відповідністю з рівністю $r_{i-1} = r_i q_{i+1} + r_{i+i}$ (якщо $r_i = 1$, тоді зупиняєио процес);

2) використовуємо рекурентне співвідношення $u_{i+1} = u_{i-1} - q_{i+1}u_i$;

3) використовуємо рекурентне співвідношення $v_{i+1} = v_{i-1} - q_{i+1}v_i$;

4) щоб почати процес виконання алгоритму, використовуємо значення $r_0 = 20$, $r_1 = 3$, $u_0 = 1$, $u_1 = 0$, $v_0 = 0$, $v_1 = 1$.

i	r_i	q_i	u_i	v_i
0	20		1	0
1	3		0	1
2	20 *mod* 3 =2	20 *div* 3 = 6	1-6·0=1	0-6·1=-6
3	3 *mod* 2 =1	3 *div* 2 = 1	0-1·1=-1	1-1·(-6)=7

Виконуємо перевірку $3 \cdot 7\ mod\ 20 = 1$. Таким чином, $d = 7$.

Опублікуємо відкритий ключ $(e,\ n)\ =\ (3, 33)$.

Зберігаємо в таємниці секретний ключ $(d,\ n)\ =\ (7, 33)$.

Зашифруємо повідомлення M = 14 17 10 3 0, що складається із п'яти блоків m_i:

$c_1 = 14^3\ mod\ 33 = \big((14^2\ mod\ 33) \cdot (14^1\ mod\ 33)\big)\ mod\ 33 = (31 \cdot 14)\ mod\ 33 =$
$= 434\ mod\ 33 = 5;$
$c_2 = 17^3\ mod\ 33 = \big((17^2\ mod\ 33) \cdot (17^1\ mod\ 33)\big)\ mod\ 33 = (25 \cdot 17)\ mod\ 33 =$
$= 425\ mod\ 33 = 29;$
$c_3 = 10^3\ mod\ 33 = 1000\ mod\ 33 = 10;$

$c_4 = 3^3 \, mod \, 33 = 27 \, mod \, 33 = 27;$
$c_5 = 0^3 \, mod \, 33 = 0 \, mod \, 33 = 0.$

Шифротекст: C = 5 29 10 27 0.

Для дешифрування потрібно також виконати піднесення до степеня, використовуючи ключ дешифрування 7:

$m_1 = 5^7 \, mod \, 33 = ((5^4 \, mod \, 33) \cdot (5^3 \, mod \, 33)) \, mod \, 33 = (31 \cdot 26) \, mod \, 33 =$
$= 806 \, mod \, 33 = 14;$
$m_2 = 29^7 \, mod \, 33 = ((29^4 \, mod \, 33) \cdot (29^3 \, mod \, 33)) \, mod \, 33 =$
$= (((29^2))^2 \, mod \, 33) \cdot (29^2 \, mod \, 33) \cdot (29 \, mod \, 33)) \, mod \, 33 = (25 \cdot 16 \cdot 29) \, mod \, 33$
$= 11600 \, mod \, 33 = 17;$
$m_3 = 10^7 \, mod \, 33 = ((10^4 \, mod \, 33) \cdot (10^3 \, mod \, 33)) \, mod \, 33 =$
$= (((10^2))^2 \, mod \, 33) \cdot (10^2 \, mod \, 33) \cdot (10 \, mod \, 33)) \, mod \, 33 = (1 \cdot 1 \cdot 10) \, mod \, 33 =$
$= 10 \, mod \, 33 = 10;$
$m_4 = 27^7 \, mod \, 33 = ((27^4 \, mod \, 33) \cdot (27^3 \, mod \, 33)) \, mod \, 33 =$
$= (((27^2))^2 \, mod \, 33) \cdot (27^2 \, mod \, 33) \cdot (27 \, mod \, 33)) \, mod \, 33 = (9 \cdot 3 \cdot 27) \, mod \, 33 =$
$= 729 \, mod \, 33 = 3;$
$m_5 = 0^7 \, mod \, 33 = 0 \, mod \, 33 = 0.$

Відкритий текст: M = 14 17 10 3 0 $\Rightarrow$ КНИГА.

3.1.3. АЛГОРИТМ ШИФРУВАННЯ ЕЛЬ-ГАМАЛЯ

Алгоритм шифрування Ель-Гамаля (ElGamal) – криптосистема з відкритим ключем, заснована на складності обчислення дискретних логарифмів в скінченному полі. Шифр була запропонована американським вченим єгипетського походження Тахером Ель-Гамалем у 1984 році.

Генерація ключів

1. Генерується просте випадкове число p.

2. Вибирається генератор g, таке що $1 < g < p - 1$ та $g^{p-1} mod \, p = 1$.

3. Вибирається випадкове число x, таке що $1 < x < p - 1$.

4. Обчислюється $y = g^x \, mod \, p$.

5. *Відкритими* даними є p, g, y.

6. *Закритим* ключем є x.

Шифрування

Повідомлення *M* шифрується таким чином:

Вибирається сесійний ключ – випадкове число k, таке що $1 < k < p - 1$.

Потім обчислюються $a = g^k \, mod \, p$ та $b = y^k M \, mod \, p$.

Пара чисел $(a, \, b)$ є шифротекстом.

Дешифрування

Для дешифрування $(a,\ b)$ обчислюється:

$M = b(a^x)^{-1}\ mod\ p$ або $M = b(a^x)^{-1}\ mod\ p = b \cdot a^{(p-1-x)}\ mod\ p$.

Приклад 3.7:

Зашифруємо повідомлення $M = 5$.

Спершу згенеруємо ключі шифрування. Нехай $p = 11,\ g = 2$.

Виберемо $x = 8$ – випадкове ціле число x таке, що таке що $1 < x < p - 1$.

Обчислимо $y = g^x\ mod\ p = 2^8\ mod\ 11 = 3$.

Отже, відкритим даними є трійка є $11,\ 2$ та 3, закритим ключем є число $x = 8$.

Для шифрування вибираємо випадкове ціле число $k = 9$ таке, що $1 < k < p - 1$.

Обчислюємо $a = g^k mod\ p = 2^9 mod\ 11 = 512\ mod\ 11 = 6$.

Обчислюємо $b = y^k M\ mod\ p = 3^9 \cdot 5\ mod\ 11 = 19683 \cdot 5\ mod\ 11 = 9$.

Пара $(6,\ 9)$ є шифротекстом.

Шифротекст $(6,\ 9)$, закритий ключ $x = 8$.

Для дешифрування обчислюємо M за формулою:

$$M = b(a^x)^{-1} mod\ p = b \cdot a^{(p-1-x)}\ mod\ p = 9 \cdot\ 6^{(11-1-8)} mod\ 11 = 5.$$

Отримали початкове повідомлення $M = 5$.

3.1.4. АЛГОРИТМ ОБМІНУ КЛЮЧАМИ ДІФФІ-ХЕЛМАНА

Протокол обміну ключами Діффі-Хелмана дозволяє двом сторонам отримати спільний секретний ключ, використовуючи незахищений від прослуховування, але захищений від модифікації канал зв'язку. Отриманий ключ можна використовувати для симетричного шифрування повідомлень. Алгоритм заснований на складності обчислень дискретних логарифмів.

Припустимо, користувачі A і B мають намір обмінятися ключами за алгоритмом Діффі-Хелмана, суть якого полягає в наступному (рис. 3.2):

1. A і B спільно обирають просте число p і ціле число g таке, що $1 < g < p - 1$ і g є первісним коренем p.

Первісним коренем за модулем p називається таке число g, що при піднесення до степеню $g^i mod\ p$ всі його степені $i \in \{1, \dots, p-1\}$ за модулем p пробігають по всім числам взаємно простим із p.

Нехай $p = 5$. Усі взаємно прості числа з p: 1, 2, 3, 4.

*Елементи **2** та **3** є первісними коренями 5.*

1
$1^1 mod\ 5 = 1$
$1^2 mod\ 5 = 1$
$1^3 mod\ 5 = 1$
$1^4 mod\ 5 = 1$
2
$2^1 mod\ 5 = 2\ mod\ 5 = \mathbf{2}$
$2^2 mod\ 5 = 4\ mod\ 5 = \mathbf{4}$
$2^3 mod\ 5 = 8\ mod\ 5 = \mathbf{3}$
$2^4 mod\ 5 = 16\ mod\ 5 = \mathbf{1}$
3
$3^1 mod\ 5 = 3 \bmod 5 = \mathbf{3}$
$3^2 mod\ 5 = 9 \bmod 5 = \mathbf{4}$
$3^3 mod\ 5 = 27 \bmod 5 = \mathbf{2}$
$3^4 mod\ 5 = 81 \bmod 5 = \mathbf{1}$
4
$4^1 mod\ 5 = 4 \bmod 5 = 4$
$4^2 mod\ 5 = 16 \bmod 5 = 1$
$4^3 mod\ 5 = 64 \bmod 5 = 4$
$4^4 mod\ 5 = 256 \bmod 5 = 1$

2. Користувач A вибирає випадкове ціле число $x < p$, обчислює $x_A = g^x\ mod\ p$ та відправляє його користувачеві B.
3. Користувач B вибирає випадкове ціле число $y < p$, обчислює $y_B = g^y\ mod\ p$ та відправляє його користувачеві A.
4. Користувач A обчислює закритий ключ за формулою $k_A = y_B^x\ mod\ p$.
5. Користувач B обчислює закритий ключ за формулою $k_B = x_A^y\ mod\ p$.

Ці дві формули обчислення дають однакові результати. Відкритими параметрами є: p, g, x_A та y_B. Закриті параметри: x, y.

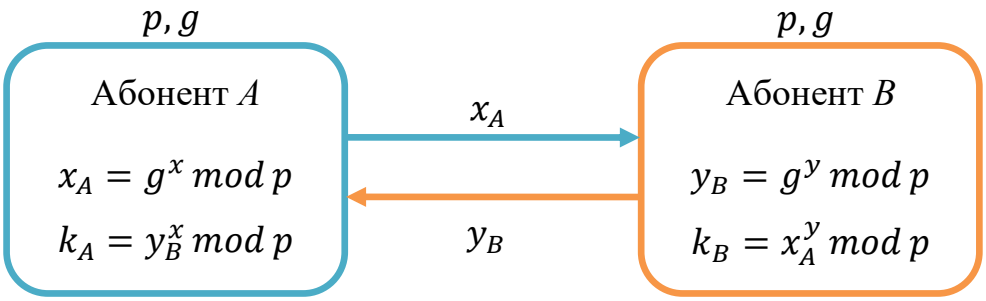


Рис. 3.2. Схема обміну ключами Діффі-Хелмана

Приклад 3.8:

1. Нехай p =11, g =2.
2. x = 4, обчислимо $x_A = 2^4 \, mod \, 11 = 16 \, mod \, 11 = 5$.
3. y = 6, обчислимо $y_B = 2^6 \, mod \, 11 = 64 \, mod \, 11 = 9$.
4. $k_A = 9^4 \, mod \, 11 = (9^2)^2 \, mod \, 11 = 4^2 \, mod \, 11 = 16 \, mod \, 11 = 5$.
5. $k_B = 5^6 \, mod \, 11 = (5^3)^2 \, mod \, 11 = 4^2 \, mod \, 11 = 16 \, mod \, 11 = 5$.

Секретний ключ, обчислений обома сторонами – 5.

3.2. КРИПТОГРАФІЧНІ ХЕШ-ФУНКЦІЇ

3.2.1. ПОНЯТТЯ ТА ВЛАСТИВОСТІ ХЕШ-ФУНКЦІЙ

Хеш-функція являє собою функцію, математичну або іншу, що отримує на вхід рядок змінної довжини і перетворює його в рядок фіксованої, зазвичай меншої, довжини. Такі перетворення ще називають *функціями згортки*, а їх результати – *хешем*, *хеш-значенням* або *дайджестом повідомлення*.

Алгоритм хешування – послідовність математичних перетворень, в результаті яких з деякої двійкової послідовності змінної довжини отримується унікальна двійкова послідовність фіксованої довжини.

Практично усі хеш-функції працюють за одним і тим самим принципом. Вхідне значення (повідомлення, файл тощо) розглядається як послідовність *n*-бітних блоків. Кожен блок обробляється послідовно один за визначеним алгоритмом хешування, а в результаті обробки останнього блоку отримується *m*-бітове хеш-значення.

Застосування хеш-функцій:

✓ ***перевірка цілісності повідомлень та файлів*** – порівнюючи хеш-значення повідомлень, обчислені до та після надсилання, можна визначити, чи були внесені будь-які зміни до повідомлення або файлу;

✓ ***верифікація пароля*** – паролі користувачів зберігаються у базі даних не у відкритому вигляді, а у вигляді хешів та не можуть бути відновлені зі збережених хеш-значень;

✓ ***генерація і перевірка цифрового підпису*** – обчислення хешу дозволяє виявити найменші зміни у документі та перевірити дійсність підпису.

Основні властивості криптографічної хеш-функції

1. *Детермінованість* – для однакових повідомлень *M* функція має повертати однакові хеш-значення *h*.

2. *Односторонність* – за значенням *h* неможливо відновити *M*.

3. *Наявність лавинного ефекту* – будь-які, навіть незначні, зміни у повідомленні *M* призводять до значних змін у хеш-значенні *h*.

4. *Відсутність колізій (унікальність хеша)* – ймовірність співпадіння хеш-значень двох різних повідомлень повинна бути надзвичайно малою.

5. *Висока швидкість роботи*.

Криптографічні хеш-функції широко використовуються в таких протоколах безпеки, як SSL/TLS та SSH, та в інших додатках, які покладаються на цілісність даних. Криптовалюти використовують алгоритми хешування для оновлення блокчейна новими блоками захищених та перевірених даних транзакцій.

Існує багато алгоритмів хешування із різними характеристиками (довжина блоку, довжина хешу, кількість раундів тощо). Вибір тієї чи іншої хеш-функції визначається специфікою вирішуваної задачі. Приклади найбільш популярних хеш-функцій наведено в таблиці 3.1.

Таблиця 3.1. Деякі хеш-функції та їх характеристики

Хеш-функція	**Рік**	**Розробники**	**Довжина блоку**	**Довжина хешу**	**Кількість раундів**
MD5	1992	Ronald Rivest	512	128	64
RIPEMD	1992	The RIPE Consortium	512	128	48
SHA-1	1995	NSA	512	160	80
SHA-256	2002	NSA	512	256	64
SHA-512	2002	NSA	1024	512	80
Whirlpool	2004	Vincent Rijmen, Paulo Barreto	512	512	10
BLAKE-256	2008	Jean-Philippe Aumasson, Luca Henzen, Willi Meier, Raphael C.-W. Phan	512	256	14
Купина	2014	ПАТ «Інститут інформаційних технологій»	512	від 8 до 512 біт	10 або 14

3.2.2. ХЕШ-ФУНКЦІЯ SHA-256

SHA-256 є окремим випадком хеш-функції із сімейства криптографічних хеш-функцій SHA-2 (Secure Hash Algorithm Version 2) опублікованим АНБ США у 2002 році. Наприклад, у twitter SHA-256 використовується для збереження паролів користувачів.

Довжина вхідних даних: до $2^{64} - 1$ біт.

Довжина хешу: 256 біт.

Довжина блоку: 512 біт.

Кількість циклів: 64.

Алгоритм SHA-256

1. *Попередня обробка*, що полягає у доповненні початкового повідомлення та його розбиття на блоки по 512 біт.

2. *Ініціалізація значень хешу*. Використовуються константи, що представляють собою перші 32 біта дробових частин квадратних коренів перших 8 простих чисел: 2, 3, 5, 7, 11, 13, 17, 19):

```
h0 := 0x6a09e667
h1 := 0xbb67ae85
h2 := 0x3c6ef372
h3 := 0xa54ff53a
h4 := 0x510e527f
h5 := 0x9b05688c
h6 := 0x1f83d9ab
h7 := 0x5be0cd19
```

.

3. *Створення масиву констант* k [0..63]. Використовуються ще 64 константи – це перші 32 біта дробових частин кубічних коренів перших 64 простих чисел (2 – 311):

```
k[0..63] :=
   0x428a2f98, 0x71374491, 0xb5c0fbcf, 0xe9b5dba5, 0x3956c25b, 0x59f111f1, 0x923f82a4, 0xab1c5ed5,
   0xd807aa98, 0x12835b01, 0x243185be, 0x550c7dc3, 0x72be5d74, 0x80deb1fe, 0x9bdc06a7, 0xc19bf174,
   0xe49b69c1, 0xefbe4786, 0x0fc19dc6, 0x240ca1cc, 0x2de92c6f, 0x4a7484aa, 0x5cb0a9dc, 0x76f988da,
   0x983e5152, 0xa831c66d, 0xb00327c8, 0xbf597fc7, 0xc6e00bf3, 0xd5a79147, 0x06ca6351, 0x14292967,
   0x27b70a85, 0x2e1b2138, 0x4d2c6dfc, 0x53380d13, 0x650a7354, 0x766a0abb, 0x81c2c92e, 0x92722c85,
   0xa2bfe8a1, 0xa81a664b, 0xc24b8b70, 0xc76c51a3, 0xd192e819, 0xd6990624, 0xf40e3585, 0x106aa070,
   0x19a4c116, 0x1e376c08, 0x2748774c, 0x34b0bcb5, 0x391c0cb3, 0x4ed8aa4a, 0x5b9cca4f, 0x682e6ff3,
   0x748f82ee, 0x78a5636f, 0x84c87814, 0x8cc70208, 0x90befffa, 0xa4506ceb, 0xbef9a3f7, 0xc67178f2
```

.

4. *Основний цикл*. Наступні кроки будуть виконуватися для кожного 512-бітного блоку вхідних даних. На кожній ітерації циклу буде змінюватися значення h0 – h7:

Крок 1. Блок ділиться на 16 слів (кожне слово – 32 біти) та записується у масив w[0..15];

Крок 2. Додається (в кінець масиву) ще 48 слів, ініціалізованих нулями, щоб отримати масив w[0..63];

Крок 3. Нульові елементи w[16..63] замінюються на нові за алгоритмом:

```
for i from 16 to 63
    s0 := (w[i-15] rightrotate  7) xor (w[i-15] rightrotate 18) xor (w[i-15] rightshift  3)
    s1 := (w[i- 2] rightrotate 17) xor (w[i- 2] rightrotate 19) xor (w[i- 2] rightshift 10)
    w[i] := w[i-16] + s0 + w[i-7] + s1
```

;

Крок 4. Ініціалізація змінних a, b, c, d, e, f, g, h поточними значенням хешу відповідно h0, h1, h2, h3, h4, h5, h6, h7:

```
a := h0
b := h1
c := h2
d := h3
e := h4
f := h5
g := h6
h := h7
```

;

Крок 5. Виконується цикл стиснення, який буде змінювати усі значення від a до h наступним чином:

```
for i from 0 to 63
    S1 := (e rightrotate 6) xor (e rightrotate 11) xor (e rightrotate 25)
    ch := (e and f) xor ((not e) and g)
    temp1 := h + S1 + ch + k[i] + w[i]
    S0 := (a rightrotate 2) xor (a rightrotate 13) xor (a rightrotate 22)
    maj := (a and b) xor (a and c) xor (b and c)
    temp2 := S0 + maj

    h := g
    g := f
    f := e
    e := d + temp1
    d := c
    c := b
    b := a
    a := temp1 + temp2
```

;

Крок 6. До значень h0…h7 додаються відповідні змінні a…h (за модулем 232):

```
h0 := h0 + a
h1 := h1 + b
h2 := h2 + c
h3 := h3 + d
h4 := h4 + e
h5 := h5 + f
h6 := h6 + g
h7 := h7 + h
```

5. *Фінальний хеш* є конкатенацією:

```
digest := hash := h0 append h1 append h2 append h3 append h4 append h5 append h6 append h7
```

3.2.3. ХЕШ-ФУНКЦІЯ «КУПИНА» (ДСТУ 7564:2014)

У грудні 2014 року прийнято національний стандарт ***ДСТУ 7564:2014*** (введений в дію 1 квітня 2015 р.), що базується на криптографічній функції хешування «Купина». Хеш-функція «Купина» використовується зокрема й для створення та перевірки електронного цифрового підпису, що визначений у *ДСТУ 4145*.

Довжина вхідних даних: до $2^{96} - 1$ біт .

Довжина хешу: від 8 до 512 біт. Варіант, який повертає n біт, позначається як «Купина-n».

Довжина блоку: 512 біт для $8 \le n \le 256$ (8 стовпців у матриці стану) або 1024 біт для $256 < n \le 512$ (16 стовпців у матриці стану).

Кількість ітерацій: 10 для $8 \le n \le 256$, 14 для $256 < n \le 512$.

Алгоритм «Купина»

1. *Розбиття повідомлення на блоки*. Повідомлення M розбивають на t блоків $m_1, m_2, \dots, m_t$ завдовжки l бітів кожен.

2. *Доповнення останнього блоку*. В кінець повідомлення M довжини N додається додаткова інформація, яка містить одиничний біт «1», d нульових біт, які визначаються за формулою: $d = (-N - 97) mod\ l$.Після цього додають ще 96 бітів, в яких міститься значення довжини повідомлення N.

3. *Обчислення хеш-значення* за такою ітеративною процедурою (рис.3.3):

$$h_0 = IV,$$

$$h_i = T_l^{\oplus}(h_{i-1} \oplus m_i) \oplus T_l^{+}(m_i) \oplus h_{i-1}, \text{ де } i = 1, \dots, t,$$

$$H(IV, M) = R_{l,n}\left(T_l^{\oplus}(h_t) \oplus h_t\right).$$

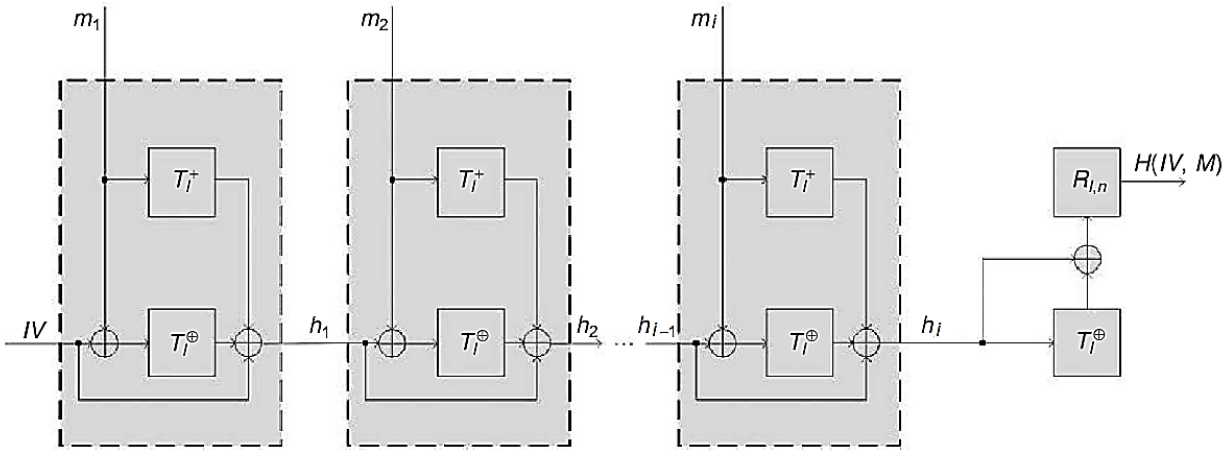


Рис. 3.3. Структура хеш-функції «Купина»

4. *Завершальне перетворення* результату хешування являє собою функцію $R_{l,n}(x)$, що повертає n старших біт з вхідного блоку x довжиною l біт ($n < l$), де результат записується в молодші n біт обчисленого значення.

Перетворення $T_l^{\oplus}$ та T_l^{+} виконуються над матрицею стану, кожним елементом якої є один байт та містять такі операції:

1. Додавання з константами ітерації;
2. Підстановка байтів;
3. Зсув рядків;
4. Перемішування стовпців.

Розглянемо кожен з чотирьох етапів детальніше.

Додавання з константами ітерації

Побітове додавання за модулем 2 ($T_l^{\oplus}$) або за модулем 2^{64} (T_l^{+}) до елементів матриці внутрішнього стану та відповідних констант ітерацій (табл. 3.2).

Таблиця 3.2. Константи ітерацій хеш-функції «Купина»

Внутрішній стан 512 біти

$$C^i = \begin{bmatrix} 00\oplus i & 10\oplus i & 20\oplus i & 30\oplus i & 40\oplus i & 50\oplus i & 60\oplus i & 70\oplus i \\ 00 & 00 & 00 & 00 & 00 & 00 & 00 & 00 \\ 00 & 00 & 00 & 00 & 00 & 00 & 00 & 00 \\ 00 & 00 & 00 & 00 & 00 & 00 & 00 & 00 \\ 00 & 00 & 00 & 00 & 00 & 00 & 00 & 00 \\ 00 & 00 & 00 & 00 & 00 & 00 & 00 & 00 \\ 00 & 00 & 00 & 00 & 00 & 00 & 00 & 00 \\ 00 & 00 & 00 & 00 & 00 & 00 & 00 & 00 \end{bmatrix};$$

Внутрішній стан 1024 біти

$$C^i = \begin{bmatrix} 00\oplus i & 10\oplus i & 20\oplus i & 30\oplus i & 40\oplus i & 50\oplus i & \dots & f0\oplus i \\ 00 & 00 & 00 & 00 & 00 & 00 & \dots & 00 \\ 00 & 00 & 00 & 00 & 00 & 00 & \dots & 00 \\ 00 & 00 & 00 & 00 & 00 & 00 & \dots & 00 \\ 00 & 00 & 00 & 00 & 00 & 00 & \dots & 00 \\ 00 & 00 & 00 & 00 & 00 & 00 & \dots & 00 \\ 00 & 00 & 00 & 00 & 00 & 00 & \dots & 00 \\ 00 & 00 & 00 & 00 & 00 & 00 & \dots & 00 \end{bmatrix};$$

Внутрішній стан 512 біти *Внутрішній стан 1024 біти*

$$C^i = \begin{bmatrix} f3 & f3 & f3 & f3 & f3 & f3 & f3 & f3 \\ f0 & f0 & f0 & f0 & f0 & f0 & f0 & f0 \\ f0 & f0 & f0 & f0 & f0 & f0 & f0 & f0 \\ f0 & f0 & f0 & f0 & f0 & f0 & f0 & f0 \\ f0 & f0 & f0 & f0 & f0 & f0 & f0 & f0 \\ f0 & f0 & f0 & f0 & f0 & f0 & f0 & f0 \\ f0 & f0 & f0 & f0 & f0 & f0 & f0 & f0 \\ 70 \oplus i & 60 \oplus i & 50 \oplus i & 40 \oplus i & 30 \oplus i & 20 \oplus i & 10 \oplus i & 00 \oplus i \end{bmatrix}; \quad C^i = \begin{bmatrix} f3 & f3 & f3 & f3 & f3 & f3 & \dots & f3 \\ f0 & f0 & f0 & f0 & f0 & f0 & \dots & f0 \\ f0 & f0 & f0 & f0 & f0 & f0 & \dots & f0 \\ f0 & f0 & f0 & f0 & f0 & f0 & \dots & f0 \\ f0 & f0 & f0 & f0 & f0 & f0 & \dots & f0 \\ f0 & f0 & f0 & f0 & f0 & f0 & \dots & f0 \\ f0 & f0 & f0 & f0 & f0 & f0 & \dots & f0 \\ f0 \oplus i & e0 \oplus i & d0 \oplus i & c0 \oplus i & b0 \oplus i & a0 \oplus i & \dots & 00 \oplus i \end{bmatrix}.$$

Підстановка байтів

Кожен байт матриці стану замінюється відповідно до заданої таблиці підстановки алгоритму «Калина» (табл. 2.15). Заміна одного байту полягає у виборі з таблиці підстановки нового значення за адресою, яку задає поточне значення байту.

Зсув рядків

Рядки стану циклічно зсувають праворуч на різну кількість байтів, залежно від розміру блока (рис. 3.4):

Номер рядка	Значення зсуву, байтів	
	Внутрішній стан 512 біти	*Внутрішній стан 1024 біти*
0	0	0
1	1	1
2	2	2
3	3	3
4	4	4
5	5	5
6	6	6
7	7	11

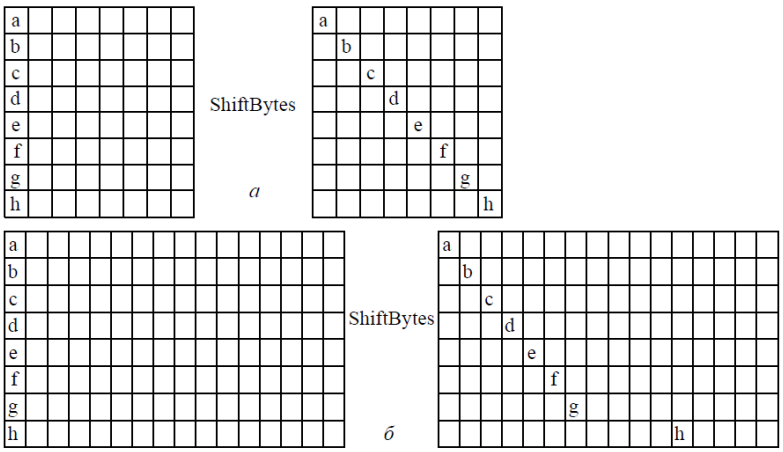


Рис. 3.4. Зсув рядків: а – 512-бітового та б – 1024-бітового внутрішнього стану

Перемішування стовпців

Як і в алгоритмі «Калина» відбувається множення стовпців стану як многочленів над полем $GF(2^8)$ на фіксований многочлен $c(x)$:

$$c(x) = 01_{16} \cdot x^7 + 05_{16} \cdot x^6 + 01_{16} \cdot x^5 + 08_{16} \cdot x^4 + 06_{16} \cdot x^3 + 07_{16} \cdot x^2 + 04_{16} \cdot x + 01_{16}.$$

Або у матричному вигляді:

$$\begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \end{pmatrix} = \begin{pmatrix} 01 & 01 & 05 & 01 & 08 & 06 & 07 & 04 \\ 04 & 01 & 01 & 05 & 01 & 08 & 06 & 07 \\ 07 & 04 & 01 & 01 & 05 & 01 & 08 & 06 \\ 06 & 07 & 04 & 01 & 01 & 05 & 01 & 08 \\ 08 & 06 & 07 & 04 & 01 & 01 & 05 & 01 \\ 01 & 08 & 06 & 07 & 04 & 01 & 01 & 05 \\ 05 & 01 & 08 & 06 & 07 & 04 & 01 & 01 \\ 01 & 05 & 01 & 08 & 06 & 07 & 04 & 01 \end{pmatrix} \cdot \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \\ a_7 \end{pmatrix}.$$

3.3. ЦИФРОВИЙ ПІДПИС

Із широким розповсюдженням у сучасному світі електронних форм документів, у тому числі і конфіденційних, та засобів їхньої обробки, особливо актуальним є питання автентифікації, ідентифікації та неспростовності електронної документації. Для захисту від підробки, перевірки цілісності даних та достовірності джерела повідомлення використовують цифровий підпис.

Електронний підпис – електронні дані, які додаються підписувачем до інших електронних даних або логічно з ними пов'язуються і використовуються ним як підпис.

(Електронний) цифровий підпис – вид електронного підпису, отриманого за результатом криптографічного перетворення набору електронних даних, який додається до цього набору або логічно з ним поєднується і дає змогу підтвердити його цілісність та ідентифікувати підписувача.

Існує декілька алгоритмів побудови цифрового підпису (ЦП). Найбільш ефективним та найпоширенішим у застосуванні на даний момент є алгоритм ЦП на основі асиметричних криптосистем з використанням хеш-функцій.

Хеш-функція *H*, яка використовується у алгоритмі ЦП, призначена для того, щоб стиснути повідомлення *M* довільної довжини до двійкового хеш-значення *h(M)* фіксованої довжини. Хешування не входить до складу алгоритму ЦП, тому в схемі може бути використана будь-яка надійна хеш-функція.

Етапи цифрового підпису

1. *Генерація пари ключів.* За допомогою алгоритму генерації ключів створюється пара ключів – закритий (для створення підпису) та відкритий (для перевірки підпису).
2. *Формування підпису.* Для заданого електронного документа за допомогою деякої хеш-функції обчислюється хеш-значення, після чого воно зашифровується із використанням закритого ключа підписувача. Зашифрований дайджест і є ЦП для даного документу.
3. *Перевірка (верифікація) підпису.* Для отриманого документу одержувач знову обчислює його хеш-значення, після чого за допомогою відкритого ключа підписувача дешифрує ЦП. Якщо хеші рівні – підпис справжній.

Управлянням ключами займаються центри сертифікації ключів (ЦСК), що забезпечують:

- доступ користувача до справжнього відкритого ключа іншого користувача;
- захист ключів від підміни зловмисником;
- організацію відкликання ключа у випадку його компрометації.

Сертифікат відкритого ключа – електронний документ, який засвідчує належність відкритого ключа фізичній або юридичній особі.

3.3.1. АЛГОРИТМ ЦИФРОВОГО ПІДПИСУ RSA

Для створення підпису повідомлення M спочатку необхідно за допомогою деякої хеш-функції обчислити хеш-значення $h(M)$.

Далі за алгоритмом RSA генеруються ключі $(e,\ n)$ і $(d,\ n)$.

ЦП повідомлення $h(M)$ буде мати вигляд: $S = h(M)^d mod\ n.$

Тепер кожний, хто має відкритий ключ підписувача повідомлення, може перевірити дійсність підпису. Для цього необхідно знайти результат хешування прийнятого повідомлення M за допомогою тієї самої хеш-функції $h'(M)$ та порівняти його із $s^e mod\ n = h(M)$. Якщо дайджести рівні – підпис дійсний.

Приклад 3.9:

З використанням алгоритму RSA підписати та перевірити підпис повідомлення M хеш-значення, якого $h(M) = 88$.

Оберемо $p = 17$ і $q = 11$, тоді $n = p \cdot q = 17 \cdot 11 = 187$.

Обчислимо $\varphi(187) = 16 \cdot 10 = 160$.

Виберемо відкритий ключ $e = 7$ та перевіримо виконання умов: $1 < 7 < 160$, НСД(7, 160) =1.

Знайдемо закритий ключ $d = 23$ за розширеним алгоритмом Евкліда з рівняння $7d \equiv 1(mod\ 160)$.

Обчислимо підпис за допомогою закритого ключа підписувача:

$$s = h(M)^d mod\ n = 88^{23} mod\ 187 = 11.$$

Для перевірки підпису повідомлення M одержувачу потрібно знову обчислити його хеш-значення $h(M) = 88$ та порівняти із значенням, отриманим за допомогою відкритого ключа підписувача:

$$s^e mod\ n = 11^7 mod\ 187 = 88.$$

В даному випадку будемо вважати, що підпис справжній.

3.3.2. АЛГОРИТМ ЦИФРОВОГО ПІДПИСУ ЕЛЬ-ГАМАЛЯ

Як правило, спочатку потрібно за допомогою деякої хеш-функції знайти дайджест $h(M)$ для повідомлення M.

Для генерації пари ключів спочатку вибирається просте число p та числа g (первісний корінь за модулем p) й x (закритий ключ). Обидва ці числа повинні бути менше p. Після чого обчислюється $y = g^x\ mod\ p$ (відкрити ключ).

Виберемо сесійний ключ – випадкове число k, таке що $1 < k < p - 1$ та обчислимо $r = g^k mod\ p$. Після чого обчислимо $s = k^{-1}(h(M) - xr) mod\ p - 1$.

Отже, підписом повідомлення M являється пара $(r,\ s)$.

Випадкове значення k повинне зберігатися в секреті і не повинно дублюватися для різних підписів. Для перевірки підпису потрібно використати відкриті параметри $(p,\ g,\ y)$ та переконатися, що $g^{h(M)} \equiv y^r r^s (mod\ p)$.

Приклад 3.10:

З використанням алгоритму Ель-Гамаля підписати та перевірити підпис повідомлення M хеш-значення, якого $h(M) = 14$.

Виберемо $p = 19$ і $g = 10$. Нехай $x = 16$ – закритий ключ. Обчислимо відповідний відкритий ключ $y = g^x mod\ p = 10^{16} mod\ 19 = 4$.

Виберемо $k = 5$ (сесійний ключ), такий що $1 < 5 < 18$.

Визначимо, що $d = 23$ (закритий ключ) за розширеним алгоритмом Евкліда з рівняння $7d \equiv 1(mod\ 160)$.

Обчислимо підпис:

$$r = 10^5 mod\ 19 = 3;$$

$$s = 5^{-1}(14 - 16 \cdot 3) mod\ 18 == -374\ mod\ 18 = 4;$$

$5 \cdot ? \equiv 1\ mod\ 18 \rightarrow 5^{-1} mod\ 18 = 11$ *(за розширеним алгоритмом Евкліда)*.

Приймається $(M, 3,\ 4)$. Обчислимо ліву та праву частину рівняння $g^{h(M)} \equiv y^r r^s (mod\ p)$ за модулем p:

$$g^{h(M)} mod\ p = 10^{14} mod\ 19 = 16;$$

$$y^r r^s\ mod\ p\ = 4^3 \cdot 3^4 mod\ 19 = 16.$$

Можна зробити висновок, що підпис дійсний.

3.3.3. СТАНДАРТ ЦИФРОВОГО ПІДПИСУ DSS

Національний інститут стандартів і технології США (NIST) розробив федеральний стандарт цифрового підпису DSS (Digital Signature Standard). Для створення цифрового підпису використовується алгоритм DSA (Digital Signature Algorithm).

Як хеш-алгоритм стандарт передбачає використання алгоритму SHA-1 (Secure Hash Algorithm).

Генерація ключів

1. Генерується просте число p, таке що $2^{L-1} < p < 2^L$, $512 \le L \le 1024$ і L кратне 64.

2. Обирається q – простий дільник $p - 1$, таке $2^{159} < q < 2^{160}$.

3. Обчислюється $g = h^{(p-1)/q}\ mod\ p$, де h будь-яке ціле число таке, що $0 \le h \le p - 1$ та $h^{(p-1)/q}\ mod\ p > 1$.

4. Вибирається x – випадкове ціле число, таке що $0 < x < q$.

5. Обчислюється $y = g^x \bmod p$.

6. x і y є закритим і відкритим ключами, відповідно.

Вибирається випадкове ціле число k – разовий секретний ключ, де $0 < k < q$. Підпис повідомлення M із використанням закритого ключа підписувача виглядає наступним чином: $r = (g^k mod\ p)\ mod\ q$ та $s = k^{-1}(h(M) + xr) mod\ q$,

де $h(M)$ – значення хеш-функції SHA-1 від повідомлення M. Підписом для повідомлення M є пара $(r,\ s)$.

Перевірка підпису із використанням відкритого ключа підписувача виглядає наступним чином: обчислюється $w = s^{-1}\ mod\ q$, $u_1 = (h(M)w)\ mod\ q$, $u_2 = (rw)\ mod\ q$, $v = \big((g^{u_1}y^{u_2})\ mod\ p\big) mod\ q$. Підпис дійсний, якщо $v = r$.

Приклад 3.11:

Підписати та перевірити підпис повідомлення M хеш-значення, якого $h(M) = 3$.

Виберемо $p = 23$, тоді $p - 1 = 23 - 1 = 22$, $q = 11$ – це простий дільник $p - 1$. Нехай $h = 2$, обчислимо $g = h^{(p-1)/q}\ mod\ p = 2^{22/11} mod\ 23 = 4$.

Закритий ключ: $x = 5$.

Відкритий ключ: $y = g^x \bmod p = 4^5 mod\ 23 = 1024\ mod\ 23 = 12$.

Виберемо $k = 3$ (сесійний ключ), такий що $1 < 3 < 11$.

Обчислимо підпис:

$$r = (4^3 mod\ 23)\ mod\ 11 = 18\ mod\ 11 = 7;$$

$$s = 3^{-1}(3 + 5 \cdot 7) mod\ 11 = 4 = 152\ mod\ 11 = 9\ ;$$

$3 \cdot ?\ \equiv 1\ mod\ 11 \rightarrow 3^{-1} mod\ 11 = 4$ *(за розширеним алгоритмом Евкліда).*

Приймається $(M, 7,\ 9)$. Проводяться наступні обчислення:

$$w = s^{-1}\ mod\ q = 9^{-1}\ mod\ 11 = 5;$$

$$u_1 = 3 \cdot 5\ mod\ 11 = 4;$$

$$u_2 = 7 \cdot 5\ mod\ 11 = 2;$$

$$v = \big((4^4 \cdot 12^2)\ mod\ 23\big) mod\ 11 = 18\ mod\ 11 = 7.$$

Оскільки $v = r$, то можна зробити висновок, що підпис дійсний.

3.4. ОСНОВИ КРИПТОГРАФІЇ НА ЕЛІПТИЧНИХ КРИВИХ

Криптографія на еліптичних кривих (elliptic curve cryptography, ECC) вивчає асиметричні криптосистеми, засновані на еліптичних кривих над скінченими полями. Їх безпека, як правило, базується на складності розв'язання задачі дискретного логарифмування в групі точок еліптичної кривої над скінченним полем. Використання еліптичних кривих у криптографії було незалежно запропоновано Нілом Кобліцом (Neal Koblitz) та Віктором Міллером

(Victor Miller) у 1985 році. З 1998 року використання еліптичних кривих для вирішення криптографічних завдань було закріплено в стандартах США ANSI X9.62 і FIPS 186-2 (FIPS 186-3 з 2009 року). В Україні на рівні національного стандарту (ДСТУ 4145-2002) прийнято алгоритм цифрового підпису, що ґрунтується на еліптичних кривих.

Криптосистеми на еліптичних є важливим елементом сучасних реалізацій багатьох протоколів безпеки, зокрема протоколу TLS, що використовується для передачі даних між вузлами комп'ютерної мережі. TLS дозволяє здійснювати автентифікацію сторін, забезпечувати конфіденційність даних, контролювати їх цілісність за допомогою кодів автентифікації повідомлень.

Основною перевагою криптосистем на еліптичних кривих у порівнянні із звичайними асиметричними алгоритмами є те, шо вони забезпечують еквівалентний захист за меншої довжини ключа (табл. 3.3).

Таблиця 3.3. Порівняння звичайних асиметричних алгоритмів та криптосистем на еліптичних кривих

Ступінь захисту (на кожен біт ключа)	*Мінімальна довжина ключа (в бітах)*	
	RSA/DSA/DH	**ECC**
80	1024	160
112	2048	224
128	3072	256
192	7680	384
256	15360	512

Розглянемо рівняння еліптичної кривої у спрощеному вигляді (рівняння Вейєрштрасса):

$$y^2 = x^3 + ax + b \tag{3.1}$$

Залежно від значень параметрів a і b еліптичні криві можуть приймати на площині різні форми. Так як $y = \pm\sqrt{x^3 + ax + b}$, то графік кривої симетричний відносно *Ox*.

Дискримінант рівняння: $D = \left(\frac{a}{3}\right)^3 + \left(\frac{b}{2}\right)^2$.

- $D < 0$ – три різних дійсних корені (рис. 3.5, графік 1);
- $D = 0$ – три дійсних корені, два з яких однакові (рис. 3.5, графік 2 – сингулярна крива, такі криві виключають з розгляду);
- $D > 0$ – один дійсний корінь та два комплексних (рис. 3.5, графік 3).

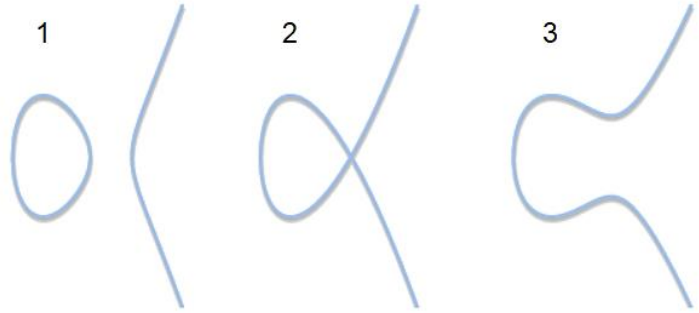


Рис. 3.5. Варіанти еліптичних кривих при $D < 0$, $D = 0$ та $D > 0$

У реальних криптосистемах використовуються еліптичні криві над скінченним полем p, що описуються рівнянням:

$$y^2 \equiv x^3 + ax + b \ (mod\ p), \tag{3.2}$$

де $(x,\ y)$ – точки еліптичної кривої,

$a,\ b$ – параметри кривої,

p – просте число ($p \neq 2, p \neq 3$).

При цьому параметри кривої a та b мають задовольняти умову:

$$4a^3 + 27b^2 \neq 0 \ (mod\ p).$$

Позначимо через $E_p(a,\ b)$ множину точок еліптичної кривої. У множину точок еліптичної кривої також включається нескінченно віддалена точка O.

Точка належить еліптичній кривій, якщо пара чисел $(x,$ у$)$ задовольняє рівнянню (3.2).

Кількість точок кривої називається *порядком кривої*.

Приклад 3.12:

Множина точок $E_5(2,\ 1)$ еліптичної кривої $y^2 \equiv x^3 + 2x + 1 \ (mod\ 5)$, складається з 6 точок, а також точки O. Порядок кривої – 7. На рис.3.6 зображено усі точки, що задовольняють рівнянню кривої.

solve	$y^2 \equiv x^3 + 2x + 1 \pmod 5$

Solutions in the least residue system:

$x \equiv 0, y \equiv 1 \pmod 5$

$x \equiv 0, y \equiv 4 \pmod 5$

$x \equiv 1, y \equiv 2 \pmod 5$

$x \equiv 1, y \equiv 3 \pmod 5$

$x \equiv 3, y \equiv 2 \pmod 5$

$x \equiv 3, y \equiv 3 \pmod 5$

Рис. 3.6. Точки, що належать еліптичній кривій $y^2 \equiv x^3 + 2x + 1\ (mod\ 5)$

3.4.1. ОПЕРАЦІЇ НАД ТОЧКАМИ ЕЛІПТИЧНИХ КРИВИХ

Оберненою точкою до $P(x, y)$ називають точку еліптичної кривої, що симетрична відносно осі Ox (рис. 3.7) та позначають $-P(x, -y)$. Варто зауважити, що $-P$ має належати $E_p(a, b)$.

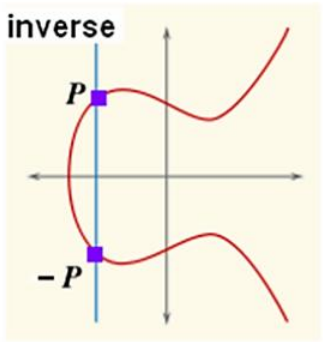


Рис. 3.7. Обернена точка еліптичної кривої

Приклад 3.13:

Якщо $P(3, 2)$ – точка еліптичної кривої $y^2 \equiv x^3 + 2x + 1\ (mod\ 5)$, то точка $-P\ (3, -2)$. Проте $-2\ mod\ 5 = 3$, тому $-P\ (3, 3)$.

Додавання точок. Візьмемо дві різні точки $P(x_1, y_1)$ та $Q(x_2, y_2)$, які належать E_p і проведемо через них пряму. Ця пряма обов'язково перетне криву в третій точці R. Проведемо через точку R вертикальну пряму до перетину з кривою у точці $-R = P + Q$. Отже, сумою двох точок P та Q буде точка, обернена до третьої точки перетину еліптичної кривої і прямої, що проходить через задані точки (рис. 3.8).

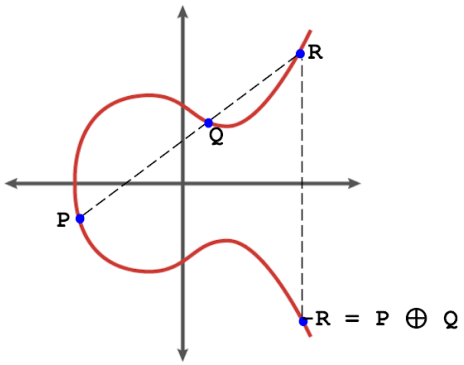


Рис. 3.8. Додавання точок еліптичної кривої

Подвоєння точки. Якщо дві точки $P(x_1,\ y_1)$ та $Q(x_2,\ y_2)$ співпадають, то $P+Q=P+P$, що рівнозначно подвоєнню точки $2P=-R$. При $P=Q$ січна перетворюється на дотичну, тому точка $2P$ є оберненою до точки R (рис. 3.9).

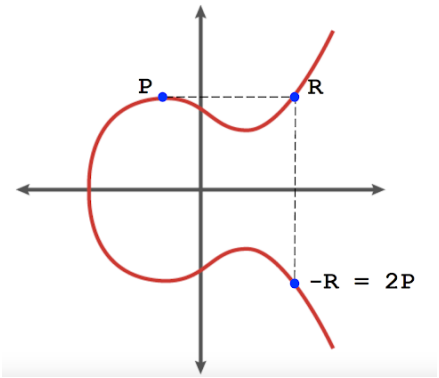


Рис. 3.9. Подвоєння точки еліптичної кривої

Координати $-R(x_3,\ y_3)$ визначаються за формулами, де λ – кутовий коефіцієнт січної, що проведена через точки $P(x_1,\ y_1)$ та $Q(x_2,\ y_2)$.

Додавання точок (якщо $P \neq Q$)	***Подвоєння точки (якщо $P = Q$)***
$x_3=\lambda^2-x_1-x_2(mod\ p);$	$x_3=\lambda^2-2x_1\ (mod\ p);$
$y_3=\lambda(x_1-x_3)-y_1(mod\ p);$	$y_3=\lambda(x_1-x_3)-y_1(mod\ p);$
$\lambda=\frac{y_2-y_1}{x_2-x_1}(mod\ p).$	$\lambda=\frac{3x_1^2+a}{2y_1}(mod\ p).$

Приклад 3.14:

Рівняння еліптичної кривої має вигляд:

$$y^2\equiv x^3+x+1\ (mod\ 23), \tag{3.3}$$

Потрібно перевірити чи точки $P(3,\ 10)$ та $Q(9,\ 7)$ належать кривій та знайти $P+Q$.

Підставимо значення $P(3,\ 10)$ та $Q(9,\ 7)$ у рівняння еліптичної кривої (3.3) та переконаємося, що точки належать кривій:

$$10^2 \equiv 3^3 + 3 + 1\ (mod\ 23) \rightarrow 100\ mod\ 23 \equiv 31(mod\ 23);$$

$$7^2 \equiv 9^3 + 9 + 1\ (mod\ 23) \rightarrow 49\ mod\ 23 \equiv 739(mod\ 23).$$

Виконаємо додавання точок $P(3,\ 10)$ та $Q(9,\ 7)$:

$$\lambda = \frac{y_2 - y_1}{x_2 - x_1}(mod\ p) = \frac{7-10}{9-3}(mod\ 23) = -\frac{3}{6}(mod\ 23) = -\frac{1}{2}(mod\ 23) =$$

$$= \frac{22}{2}(mod\ 23) = 11.$$

Знаходимо:

$$x_3 = \lambda^2 - x_1 - x_2(mod\ p) = 121 - 3 - 9(mod\ 23) = 109(mod\ 23) = 17$$

$$y_3 = \lambda(x_1 - x_3) - y_1(mod\ p) = 11(3 - 17) - 10(mod\ 23) = -164(mod\ 23) =$$

$$= 20.$$

Отже $P + Q = (3,\ 10) + (9,\ 7) = (17,\ 20)$.

Приклад 3.15:

Додати точки $P(12,\ 19)$ та $Q(5,\ 4)$ еліптичної кривої 3.1.

$$\lambda = \frac{y_2 - y_1}{x_2 - x_1}(mod\ p) = \frac{4-19}{5-12}(mod\ 23) = \frac{-15}{-7}(mod\ 23) = \frac{15}{7}(mod\ 23).$$

Якщо записати $15 \cdot \frac{1}{7}(mod\ 23) \rightarrow 5 \cdot 7^{-1}(mod\ 23)$, то потрібно знайти обернений елемент, розв'язавши рівняння:

$$7 \cdot Z \equiv 1\ (mod\ 23) \rightarrow Z = 10$$ (*за розширеним алгоритмом Евкліда*).

$$\lambda = 15 \cdot 10\ (mod\ 23) = 12.$$

$$x_3 = \lambda^2 - x_1 - x_2(mod\ p) = 144 - 12 - 5(mod\ 23) = 127(mod\ 23) = 12.$$

$$y_3 = \lambda(x_1 - x_3) - y_1(mod\ p) = 12(12 - 12) - 19(mod\ 23) = 4(mod\ 23) = 4.$$

Отже $P + Q = (12,\ 19) + (5,\ 4) = (12,\ 4)$.

Приклад 3.16:

Дано точку $P(5,\ 4)$ еліптичної кривої 2.1. Знайти $2P$ та $3P$.

$$\lambda = \frac{3x_1^2 + a}{2y_1}(mod\ p) = \frac{3 \cdot 25 + 1}{2 \cdot 4}(mod\ 23) = \frac{76}{2 \cdot 4}(mod\ 23) = \frac{19}{2}(mod\ 23).$$

Знайдемо обернений елемент $2^{-1}(mod\ 23)$, розв'язавши рівняння:

$$2 \cdot Z \equiv 1\ (mod\ 23) \rightarrow Z = 12.$$

$$\lambda = 19 \cdot 12\ (mod\ 23) = 21.$$

$$x_3 = \lambda^2 - 2x_1(mod\ p) = 441 - 10(mod\ 23) = 431(mod\ 23) = 17.$$

$$y_3 = \lambda(x_1 - x_3) - y_1(mod\ p) = 21(5 - 17) - 4(mod\ 23) = -256(mod\ 23) =$$

$$= 20.$$

Отже $2P = (17,\ 20)$.

Далі знайдемо суму точок $P + 2P = (5,\ 4) + (17,\ 20)$.

$$\lambda = \frac{20-4}{17-5}(mod\ 23) = \frac{16}{12}(mod\ 23) = \frac{4}{3}(mod\ 23).$$

Знайдемо обернений елемент, розв'язавши рівняння:

$$3 \cdot Z \equiv 1\ (mod\ 23) \rightarrow Z = 8.$$

$$\lambda = 4 \cdot 8\ (mod\ 23) = 9.$$

$$x_3 = 9^2 - 5 - 17(mod\ 23) = 81 - 22(mod\ 23) = 13.$$

$$y_3 = 9(5 - 13) - 4(mod\ 23) = 9 \cdot (-8) - 4(mod\ 23) = -76(mod\ 23) = 16.$$

Отже $3P = (13,\ 16)$.

Множина точок еліптичної кривої $E_p(a,\ b)$ разом із введеною точкою на нескінченності O утворює комутативну групу щодо операції додавання точок. Для цього виконуються усі необхідні властивості:

1) Якщо P і $Q \in E_p(a,\ b)$, то $P + Q \in E_p(a,\ b)$ – замкнутість;
2) $P + Q = Q + P$ – комутативність;
3) $(P + Q) + R = P + (Q + R)$ – асоціативність;
4) $P + (-P) = O$ – обернений елемент;
5) $P + O = O + P = P$ – нейтральний елемент.

Скалярне множення точки на число. Із попередніх операцій додавання точок та подвоєння точки випливає операція скалярного множення точки на число:

$$2P = P + P$$
$$3P = P + P + P$$
$$\dots$$
$$mP = \underbrace{P + P + P + \cdots + P}_{m \text{ разів}}$$

Скалярне множення є аналогом піднесення до степеню в звичайних асиметричних шифрах. Прямою задачею є обчислення $mP = Q$. Зворотна задача полягає у тому, що знаючи точки P та Q, знайти m важко (дискретне логарифмування у групі точок еліптичної кривої).

Точка $G \in E_p(a,b)$ називається **базовою точкою** підгрупи точок еліптичної кривої $E_p(a,b)$, якщо будь-яка точка P цієї підгрупи може бути подана у вигляді $P = mG$, де $m = 1,\ 2, \ldots n$, де n – порядок підгрупи.

Для базової точки G має місце рівність $nG = O$.

Приклад 3.17:

Точка $G = (0,\ 1)$ є базовою точкою для групи точок еліптичної кривої $y^2 \equiv x^3 + x + 1\ (mod\ 5)$. Вона генерує усі інші точки підгрупи:

$$\boldsymbol{G} = (0,\ 1) \rightarrow \boldsymbol{2G} = (4,\ 2) \rightarrow \boldsymbol{3G} = (2,\ 1) \rightarrow \boldsymbol{4G} = (3,\ 4) \rightarrow \boldsymbol{5G} = (3,\ 1) \rightarrow \boldsymbol{6G} = (2,\ 4) \rightarrow \boldsymbol{7G} = (4,\ 3) \rightarrow \boldsymbol{8G} = (0,\ 4) \rightarrow \boldsymbol{9G} = O.$$

3.4.2. АЛГОРИТМ ДІФФІ-ХЕЛМАНА НА ЕЛІПТИЧНИХ КРИВИХ

1. Абоненти А і В спільно обирають просте число p та параметри еліптичної кривої a та b.
2. У групі точок еліптичної кривої $E_p(a,\ b)$ також обирається спільна базова точка $G = (x, y)$, що має дуже великий порядок n.
3. Абонент *A* обирає $x < n$, обчислює $X_A = xG$ та відправляє його *B*.
4. Абонент *B* обирає $y < n$, обчислює $Y_B = yG$ та відправляє його *A*.
5. Абонент *A* обчислює закритий ключ за формулою $K_A = xY_B$.
6. Користувач *B* обчислює закритий ключ за формулою $K_B = yX_A$.

Приклад 3.18:

1. Нехай абоненти обрали параметри еліптичної кривої $p = 23$, $a = -2$, $b = 15$, тобто $y^2 \equiv\ x^3 - 2x + 15\ (mod\ 23)$.
2. Нехай $G = (4,\ 5)$ – базова очка.
3. Абонент *A* обирає $x = 3$ та обчислює $X_A = 3G = 2G + G = (13,\ 22)$.
4. Абонент *B* обирає $y = 7$ обчислимо $Y_B = 7G = 2G + 4G + G = (17,\ 8)$.
5. Абонент *A* обчислює закритий ключ $K_A = 3Y_B = 2Y_B + Y_B = (15,\ 5)$.
6. Абонент *B* обчислює закритий ключ $K_B = 7X_A = 2X_A + 4X_A + X_A = (15,\ 5)$.

Секретний ключ, обчислений обома сторонами – $(15,\ 5)$.

3.4.3. СТАНДАРТ ЦИФРОВОГО ПІДПИСУ ECDSS

Алгоритм ЦП DSS, який заснований на застосуванні еліптичної кривої називається ECDSS (Elliptic Curve Digital Signature Scheme). Для створення цифрового підпису використовується алгоритм ECDSA (Elliptic Curve Digital Signature Algorithm).

Генерація ключів

1. Обираються просте число p та параметри еліптичної кривої a та b.

2. Обираються базова точка $G = (x, y)$ та n (просте число), таке що $nG = O$.

3. Закритий ключ d – випадкове ціле число, таке що $0 < d \leq n - 1$.

4. Обчислюється відкритий ключ $Q = dG$.

Якщо розмірність n в бітах менше розмірності в бітах хеш-значення $h(M)$, то використовуються тільки ліві біти хеш-значення – z. Вибирається випадкове ціле число k – разовий секретний ключ, де $0 < k \leq n - 1$ та обчислюється $(x_1, y_1) = kG$.

Для підписування хеш-значення повідомлення потрібно обчислити:

$r = x_1 \ mod \ n$ (якщо $r = 0$, то потрібно обрати інше значення k);

$s = k^{-1}(z + dr) mod \ n$ (якщо $s = 0$, то потрібно обрати інше значення k).

Підписом для повідомлення M є пара (r, s).

Отримується (r, s) та підтверджене значення відкритого ключа Q. Обчислюються значення:

$$w = s^{-1} \ mod \ n;$$

$$u_1 = z \cdot w \ mod \ n;$$

$$u_2 = r \cdot w \ mod \ n;$$

$$(x_1, y_1) = u_1 G + u_2 Q.$$

Якщо $(x_1, y_1) = O$ – підпис недійсний. Якщо $r \equiv x_1 mod \ n$ – підпис дійсний.

<u>Приклад 3.19:</u>

Підписати та перевірити підпис повідомлення M хеш-значення, якого $z = 10$.

Нехай $p = 23,\ \ a = -2,\ \ b = 15,$ тобто $y^2 \equiv x^3 - 2x + 15\ (mod\ 23)$. Виберемо базову точку $G = (4,\ 5);\ n = 23.$

Оберемо $d = 3$ – закритий ключ.

Обчислимо $Q = dG = 3G = 2G + G = (13,\ 22)$ – відкритий ключ.

Сесійний ключ: $k = 19$. Обчислимо $kG = 19G = (9,\ 17).$

Підписом хеш-значення повідомлення буде:

$$r = 9\ mod\ 23 = 9;$$

$$s = 19^{-1}(10 + 3 \cdot 9)\ mod\ 23 = 629\ mod\ 23 = 8;$$

$$19^{-1} mod\ 23 = 17$$ *(за розширеним алгоритмом Евкліда).*

Відомо $(9,\ 8)$ та $Q = (13,\ 22)$. Для перевірки підпису проводяться наступні обчислення:

$$w = 8^{-1}\ mod\ 23 = 3;$$

$$u_1 = 10 \cdot 3\ mod\ 23 = 7;$$

$$u_2 = 9 \cdot 3\ mod\ 23 = 4;$$

$$7G + 4Q = (17,\ 8) + (10,\ 2) = (9,\ 17).$$

Оскільки $9 \equiv 9\ mod\ 23$ – підпис дійсний.

Контрольні запитання до розділу 3

1. У чому полягає ідея криптосистеми з відкритим ключем?
2. Хто є основоположниками криптографії з відкритим ключем?
3. Яка основна перевага асиметричних шифрів над симетричними?
4. Що таке одностороння функція?
5. На чому ґрунтується криптостійкість алгоритму шифрування даних RSA?
6. Як знайти n – модуль криптосистеми RSA?
7. Яким чином у алгоритмі RSA отримуються відкритий та закритий ключі?
8. На чому ґрунтується криптостійкість алгоритму шифрування даних Ель-Гамаля?
9. Яким чином у алгоритмі Ель-Гамаля отримуються відкритий та закритий ключі?
10. Опишіть алгоритм шифрування Ель-Гамаля.
11. Опишіть алгоритм обміну ключами Діффі-Хелмана.
12. Що таке первісний корінь за модулем простого числа?
13. Дайте визначення поняттю «хеш-функція».
14. Що таке дайджест повідомлення?
15. Які основні вимоги висуваються до криптографічної хеш-функції?
16. Що являє собою (електронний) цифровий підпис?

17. Опишіть схему створення і перевірки ЦП.
18. Як здійснюється підпис RSA? Яка відмінність підпису RSA від шифру RSA?
19. Як здійснюється підпис Ель-Гамаля?
20. Які переваги мають криптосистеми на еліптичних кривих над звичайними асиметричним алгоритмами?
21. Який загальний вигляд має крива, що використовується в криптографічних системах, заснованих на еліптичних кривих?
22. Дайте визначення порядку групи точок еліптичної кривої.
23. Дайте визначення порядку точки еліптичної кривої.
24. Які основні операції виконуються над точками еліптичних кривих при їх використанні в криптографічних системах?
25. Опишіть алгоритм Діффі-Хелмана на еліптичних кривих.

Тести до розділу 3

1. *Ідея криптосистеми з відкритим ключем полягає у використанні…*
   а) відкритого ключа як для шифрування, так і для дешифрування даних
   б) відкритого ключа для шифрування закритого ключа
   в) закритого ключа для шифрування відкритого ключа
   г) відкритого ключа для шифрування даних та закритого для дешифрування даних
   д) закритого ключа для шифрування даних та відкритого для дешифрування даних
2. *Відкритий текст виглядає як 101001, а маси предметів дорівнюють 2, 5, 8, 17, 33, 70 відповідно. Чому дорівнюватиме вага рюкзака?*
   а) 80
   б) 24
   в) 25
   г) 135
   д) 100
3. *Яка з наведених нижче послідовностей є суперзростаючою?*
   а) 1, 2, 8, 15, 27, 55
   б) 2, 3, 6, 13, 27, 49
   в) 1, 3, 4, 9, 15, 25
   г) 5, 10, 25, 42, 85, 127
   д) 4, 9, 13, 21, 35, 56
4. *Знайдіть n – модуль криптосистеми RSA, якщо $p = 13$ та $q = 17$?*
   а) 30
   б) 4
   в) 221

г) 192

д) 211

5. *Чому дорівнює $\varphi(n)$ в алгоритмі RSA?*

а) $p + q$

б) p/q

в) $p \cdot q$

г) $(p-1) * (q-1)$

д) $(p+1)/(q+1)$

6. *Якщо користувач криптосистеми RSA вибрав для генерації модуля n два числа $p = 7$ та $q = 11$, то яку відкриту експоненту e він може вибрати?*

а) 3

б) 5

в) 7

г) 10

д) 17

7. *Чому буде дорівнювати закритий ключ d в алгоритмі RSA, якщо $\varphi(n) = 20$ та $e = 3$?*

а) 3

б) 5

в) 7

г) 10

д) 17

8. *Яким буде результат шифрування повідомлення M = 4 у криптосистемі RSA, якщо e=3 та n=33?*

а) 16

б) 31

в) 34

г) 44

д) 9

9. *На чому ґрунтується криптостійкість алгоритму шифрування даних Ель-Гамаля?*

а) на складності обчислення дискретного логарифму великих простих чисел

б) на складності піднесення великих простих чисел до степеню

в) на складності розкладання великих чисел на прості множники

г) на складності обчислення коренів від великих простих чисел заданого ступеня

д) на складності множення двох великих простих чисел

10. *Чому буде дорівнювати у криптосистемі Ель-Гамаля відкритий ключ y, якщо $p = 13, g = 7, x = 2$?*
    а) 12
    б) 3
    в) 11
    г) 36
    д) 10
11. *Яким буде результат шифрування повідомлення $M = 3$ із використанням сесійного ключа $k = 2$ у криптосистемі Ель-Гамаля, якщо $p = 7, g = 3, y = 5$?*
    а) (2, 5)
    б) (3, 5)
    в) (2, 3)
    г) (5, 5)
    д) (3, 4)
12. *Яким буде результат дешифрування шифротексту $(a, b) = (6, 2)$ у криптосистемі Ель-Гамаля, якщо $p = 7, g = 3, x = 5$?*
    а) 2
    б) 3
    в) 4
    г) 5
    д) 6
13. *Визначте спільний секретний ключ користувачів A та B, обчислений абонентами у криптосистемі Діффі-Хеллмана, якщо $p = 11, g = 2, x = 4, y = 3$*
    а) 4
    б) 3
    в) 7
    г) 5
    д) 8
14. *Хеш-функція – це математична або інша функція, яка виконує…*
    а) зашифрування повідомлення за допомогою відкритого ключа
    б) дешифрування повідомлення за допомогою закритого ключа
    в) стиснення повідомлення довільної довжини до вдвічі меншого повідомлення
    г) застосування афінних перетворень для шифрування повідомлень

д) перетворення вхідного повідомлення довільної довжини у вихідний бітовий рядок фіксованої довжини

15. *Блоки даних якої довжини обробляються алгоритмом хешування SHA-256?*
    а) 256 біт
    б) 128 біт
    в) 512 біт
    г) 192 біти
    д) 64 біт

16. *Блоки даних якої довжини обробляються криптографічним алгоритмом хешування даних «Купина-512»?*
    а) 1024 біти
    б) 384 біти
    в) 256 біти
    г) 512 бітів
    д) 224 біти

17. *Оберіть найбільш повне визначення цифрового підпису серед наведених*
    а) це підпис, який створюється з використанням електронного засобу
    б) це підпис, що однозначно ідентифікує користувача в системі
    в) це підпис, отриманий за результатом криптографічного перетворення набору даних довільної довжини у вихідний бітовий рядок фіксованої довжини
    г) це сертифікований аналог звичайного рукописного підпису
    д) це підпис, отриманий за результатом криптографічного перетворення набору електронних даних, який додається до цього набору або логічно з ним поєднується

18. *Наведіть наступні чотири дії в порядку, за яким підписувач може створити ЦП повідомлення, а отримувач може перевірити справжність ЦП:*
    1 – шифрування хешу закритим ключем підписувача;
    2 – порівняння надісланого хешу із обчисленим хешем повідомлення;
    3 – дешифрування підпису відкритим ключем підписувача;
    4 – обчислення хешу повідомлення
    а) (4, 3, 1, 2)
    б) (1, 2, 3, 4)
    в) (3, 1, 4, 2)
    г) (3, 2, 4, 1)
    д) (4, 1, 3, 2)

19. *Який загальний вигляд має крива, що використовується в криптографічних системах, заснованих на еліптичних кривих?*
    а) $y^3 \equiv x^2 + ax + b \ (mod\ p)$

б) $y \equiv x^3 + ax + b\ (mod\ p)$

в) $y \equiv x^2 + ax + b\ (mod\ p)$

г) $y^2 \equiv x^3 + ax + b\ (mod\ p)$

д) $y \equiv x^3 + ax^2 + b\ (mod\ p)$

20. *Які основні операції виконуються над точками еліптичних кривих при їх використанні в криптографічних системах?*

а) додавання точок та подвоєння точок

б) додавання та множення точок

в) додавання та віднімання точок

г) множення точок

д) ділення точок

Задачі до розділу 3

1. У алгоритмі рюкзака дано закритий ключ – суперзростаюча послідовність {3, 9, 17, 32}. Обчисліть відкритий ключ – нормальну послідовність, якщо $m = 64$ та $n = 37$ та зашифруйте відкритий текст 0110 1011 1110.
2. Визначте спільний секретний ключ користувачів А та В, обчислений абонентами у криптосистемі Діффі-Хеллмана, якщо $p = 127$, $g = 12$, $x = 13$, $y = 15$.
3. Виконайте зашифрування повідомлення $M = 7$ за алгоритмом RSA, якщо $p = 23$ та $q = 17$. Самостійно оберіть відкритий ключ e та обчисліть закритий ключ d.
4. Виконайте зашифрування повідомлення $M = 5$ за алгоритмом Ель-Гамаля, якщо $p = 19$ та $g = 3$. Самостійно оберіть закритий ключ x, сесійний ключ k та обчисліть відкритий ключ y.
5. Виконайте дешифрування шифротексту $C = 72$ за алгоритмом RSA, якщо $p = 11$ та $q = 23$, а $d = 49$.
6. Виконайте формування ЦП хеш-значення повідомлення $h(M) = 6$ за алгоритмом RSA, якщо $p = 11$ та $q = 19$. Самостійно оберіть відкритий ключ e та обчисліть закритий ключ d.
7. Виконайте формування ЦП хеш-значення повідомлення $h(M) = 8$ за алгоритмом Ель-Гамаля, якщо $p = 17$ та $g = 5$. Самостійно оберіть закритий ключ x, сесійний ключ k та обчисліть відкритий ключ y.
8. Для повідомлення $M = 8$ виконайте перевірку його ЦП (r=15, s=22) за алгоритмом Ель-Гамаля, якщо $p = 29$ та $g = 8$, відкритий ключ $y = 13$, сесійний ключ $k = 9$.
9. Виконайте додавання двох точок $P(1, 2)$ та $Q(4, 2)$ еліптичної кривої $E_{11} = (1, 2)$.
10. Для еліптичної кривої $E_{13} = (1,1)$ обчисліть $2P$, якщо $P(1, 4)$.

ІСТОРИЧНА ДОВІДКА

…Історія криптології – це історія людства…
Девід Канн

1900 р. до н. е.	Єгиптяни почали використовували нестандартні ієрогліфічні символи замість звичайних ієрогліфів
1500 р. до н. е.	Один з найстаріших шифрованих текстів з Месопотамії являв собою глиняну табличку, що містила рецепт виготовлення глазурі в гончарному виробництві
VIII ст. до н. е.	У Греції остаточно сформувалося європейське алфавітне письмо. Почали розвиватися латинська та слов'янська абетки, що послугувало потужним поштовхом до розвитку криптології
VI ст. до н.е.	Священні тексти стародавніх іудеїв шифрувалися за алгоритмом простої заміни ***атбаш***
V ст. до н.е.	Перші відомості про використання шифрів у військовій справі пов'язані із ім'ям спартанського полководця *Лісандра*. Він використовував шифр ***сцитала***
IV ст. до н.е.	З ім'ям грецького полководця *Енея*, пов'язують винайдений ним шифрувальний пристрій – ***диск Енея***
II ст. до н.е.	Грецький історик *Полібій* описав свою криптосистему на основі шифрувальної таблиці – ***квадрат Полібі***
I ст. до н.е.	Юлій Цезар використовував шифр для секретного листування, що названий на його честь ***шифр Цезаря***. Це один із найбільш поширених шифрів моноалфавітної заміни.
VI – X ст.ст. н.е.	«Темні століття» в Європі. Криптографія прийшла в занепад. В ній бачили елементи чаклунства, ототожнюючи з чорною магією
VIII ст. н.е.	В арабському світі криптографія не тільки не занепала, але продовжувала успішно розвиватися. У праці арабського вченого *Абу ан-Набаті* вперше змістовно описано кілька алгоритмів шифрування. Робота іншого арабського вченого *Абу-Юсуф Аль-Кінді* містить першу відому писемну згадку про ***криптоаналіз***
XIII ст. н.е	Перша відома європейська книга, в якій розповідається про використання криптографії, була написана англійським монахом та енциклопедистом *Роджером Беконом*. У його *«Таємних дослідах та недійсності магії»* наведено сім способів зберігання повідомлення в таємниці
XIV ст. н.е	Розвиток криптології в пізнє середньовіччя і ранній Новий час було прямо пов'язане з розквітом дипломатії. Співробітник таємної канцелярії папської курії *Чікко Сімонеті* пише книгу про системи тайнопису
XV ст. н.е	Секретар папи Климентія XII *Габріель де Левінда* написав «*Трактат про шифри»* (перший європейський підручник по криптографії), в якому виклав метод розшифрування шифрів заміни, оснований на підрахунку частот літер
1466 р.	Вчений епохи Відродження *Леона Батіста Альберті*, відомий як «батько європейської криптографії», у своїй праці *«Трактат про*

	шифри» вперше запропонував замість одного секретного алфавіту, використовувати два або більше, перемикаючись між ними за певним правилом
1508 р.	Німецький абат *Йоган Трітемій (Трісемус)* написав трактат *«Поліграфія»*, у якій висвітлив дві новаторські пропозиції в області криптографії: запропонував шифр ***Аве Марія*** і ***таблицю Трітемія***
1553 р.	Подальший крок у розвитку запропонованого Трітеміем способу шифрування був зроблений італійцем *Джованні Белазо*. У брошурі «*Шифр сеньйора Белазо*». У ***шифрі Белазо*** ключем є пароль – фраза або слово, які легко запам'ятати
1563 р.	Італійський вчений *Джованні де ла Порта* опублікував книгу *«Про таємне листування»*. У цій книзі він запропонував нову систему шифру періодичної лозунгової заміни, яку пізніше назвали на його честь – ***шифр Порта***
1550 р.	Перебуваючи на службі у папи римського, італійський математик *Джероламо Кардано*, запропонував шифр перестановки названий ***решітка Кардано*** або ***трафарети***. Цей спосіб поєднував в собі як стеганографію (мистецтво прихованого листа), так і криптографію
1580 р.	Відомий англійський філософ та вчений *Френсіс Бекон* у своїй роботі запропонував двійковий спосіб кодування латинського алфавіту, за принципом аналогічний тому, що зараз використовується в комп'ютерах
1585 р.	Ґрунтуючись на ідеях Альберті, Трітемія, Белазо та Порта, свій шифр створив французький посол в Римі *Блез де Віженер* та описав його у своїй книзі «*Трактат про шифри»*. ***Шифр Віженера*** протягом 350 років вважався однією з найбільш надійних систем
XVII ст.	Ера «чорних кабінетів». У цей період у багатьох державах Європи отримали розвиток дешифрувальні підрозділи, що називалися «чорними кабінетами»
1628 р.	Перший з «чорних кабінетів» створений у Франції з ініціативи кардинала *Ришельє* при дворі короля Людовика XIII. Його очолив перший професійний криптограф Франції *Антуан Россіньоль*
1795 р.	Державний діяч, а потім і третій президент США *Томас Джефферсон* розробив перший циліндричний шифрувальний пристрій відомий як ***дисковий шифр*** або ***шифратор Джефферсона***
1854 р.	Англійський професор математики *Чарльз Бебідж* вперше дав строгу математичну формалізацію основних понять криптографії. Він запропонував алгоритм криптоаналізу поліалфавітних шифрів. Також Бебідж одним з перших математиків почав застосовувати алгебру в галузі криптографії
1854 р.	Англійський фізик *Чарльз Уітстон* описав, а міністр пошти при королеві Вікторії *Ліон Плейфер* домігся застосування британськими збройними силами нового шифру, як його пізніше назвуть – ***шифр Плейфера***
1863 р.	Офіцер пруської армії, майор *Фрідріх Казіскі* опублікував книгу під назвою *«Мистецтво тайнопису і дешифрування»*, в якій було

	описано метод зламу поліалфавітного шифру на прикладі шифру Віженера, який раніше вважався незламним
1883 р.	*Огюст Керкгоффс*, нідерландський криптограф, опублікував велику наукову працю під назвою *«Військова криптографія»*, у якій проводився порівняльний аналіз шифрів
1917 р.	Інженери телеграфної компанії AT&T *Джозеф Моборн* та *Гілберт Вернам* винайшли ***шифр одноразового блокноту*** для автоматичного шифрування телеграфних повідомлень
1917 р.	У всіх провідних країнах починають з'являтися електромеханічні шифрувальні машини. Найвідомішою стала роторна машина ***Енігма***, винайдена німецькими інженерами *Артуром Шербіусом* та *Річардом Ріттером* (деякі дослідники вважають, що машина розроблена винахідником *Едвардом Хеберном*)
1918 р.	Вийшла монографія американського криптографа *Вільяма Фрідмана «Індекс збігу і його застосування в криптографії»*
1920 р.	Фрідман ввів у науковий обіг терміни «криптологія» та «криптоаналіз»
1929 р.	Американський математик *Лестер С. Хілл* винайшов шифр, заснований на застосуванні лінійної алгебри (множення матриць) – ***криптосистема Хілла***
1940 р.	Групою вчених на чолі із видатним англійським математиком *Аланом Тюрінгом* була сконструйована перша електронно-обчислювальна машина з метою зламу шифру Енігми
1949 р.	Американський вчений *Клод Шеннон* у статті *«Теорія зв'язку в секретних системах»* сформулював та довів математичними засобами необхідні і достатні умови забезпечення секретності системи шифрування
1967 р.	Виходить книга *Девіда Кана «Зломщики кодів».* Хоча книга не містила нових відкриттів, вона детально описувала наявні на той момент результати в області криптографії
1970 р.	Вперше ідея захисту інформації за допомогою квантових об'єктів була запропонована *Стівеном Візнером*
1975 р.	Розроблений американський стандарт шифрування ***DES*** (*Data Encryption Standard*). Один з його авторів, *Хорст Фейстель* (співробітник IBM), описав модель блокових шифрів, на основі якої були побудовані інші, більш стійкі симетричні криптосистеми
1976 р.	Поява ***асиметричних криптосистем***, які не вимагали передачі секретного ключа між сторонами. Американськими вченими *Уітфілдом Діффі* та *Мартіном Хелманом* опубліковано роботу під назвою *«Нові напрямки в криптографії»*, в якій вперше сформульовані принципи обміну шифрованими повідомленнями без обміну секретним ключем
1977 р.	*Рон Рівест, Аді Шамір* і *Леонард Едлман* відкрили систему ***RSA*** – перший практичний асиметричний шифр, стійкість якого була заснована на проблемі факторизації великих простих чисел

1978 р.	Особистий внесок до ідеї асиметричних криптосистем незалежно опублікував американський криптограф *Ральф Меркл*
1978 р.	Створено ***алгоритм рюкзака*** (криптосистема Меркла-Хелмана)
1984 р.	*Чарльз Беннет*, *Томас Ватсон* та *Жиль Брасар*, спираючись на роботу Стівена Візнера, запропонували передавати секретний ключ з використанням квантових об'єктів
1985 р.	Асиметрична ***криптосистема Ель-Гамаля*** була запропонована єгипетським криптографом *Тахером Ель-Гамалем*
1986 р.	Незалежно один від одного, *Ніл Коблiц* та *Віктор Міллер* запропонували використовувати ***еліптичні криві*** для побудови криптосистем з відкритим ключем
1987 р.	Американським криптографом *Роном Рівестом* розроблено ***RC4*** – потоковий шифр, що довгий час застосовувався в таких мережевих протоколах, як WEP, WPA і TLS (до 2015 року)
1988 р.	Прийнято **X.509** – стандарт, який визначає формати даних та процедури розподілу відкритих ключів за допомогою сертифікатів з цифровими підписами
1991 р.	*Ксуеджа Лай* та *Джеймс Мессі* розробили симетричний блоковий алгоритм шифрування ***IDEA*** (*International Data Encryption Algorithm*)
1991 р.	Створено ***DSS*** (*Digital Signature Standard*) – американський стандарт цифрового підпису, в основі якого лежить ***DSA*** (*Digital Signature Algorithm*)
1991 р.	*Роном Рівестом* розроблено широко відомий криптографічний алгоритм хешування ***MD5*** (*Message Digest 5*), проте на сьогоднішній день цей алгоритм не вважається надійним
1994 р.	*Пітер Шор* розробляє алгоритм, який дозволяє квантовим комп'ютерам швидко виконувати розкладання великих цілих чисел на множники
1995 р.	Компанія *Netscape* випустила ***SSL*** (*Secure Sockets Layer*) – криптографічний протокол, який забезпечує встановлення безпечного з'єднання між клієнтом і сервером. Наступником *SSL* став ***TLS*** (Transport Layer Security (1999 рік)
1995 р.	Стандартизовано ***ECDSA*** (*Elliptic Curve Digital Signature Algorithm)* – алгоритм цифрового підпису, який заснований на застосуванні еліптичних кривих
1995 р.	Опубліковано ***SHA-1*** (*Secure Hash Algorithm-1*) – це криптографічна хеш-функція, результатом роботи якої є 160-бітове хеш-значення
1996 р.	*Джефрі Хофштейн*, *Джил Пайфер* та *Джозеф Х. Сільверман* розробили ***NTRU*** (*Nth Degree Truncated Polynomial Ring Units*) – криптосистему із відкритим ключем на основі решіток. Цей напрям заснований на завданні дискретної оптимізації, а саме на пошуку найкоротшого шляху на багатовимірній решітці
1997 р.	*Національний інститут стандартів і технологій США* (NIST) оголосив конкурс на новий блоковий симетричний стандарт шифрування

2000 р.	*У* Бельгії розпочато трирічний європейський проект ***NESSIE*** (*New European Schemes for Signatures, Integrity, and Encryption*), метою якого був відбір криптографічних алгоритмів, на основі яких теоретично повинні формуватися майбутні криптостандарти Європи
2001 р.	Прийнято новий національний стандарт блокового шифрування США – ***AES*** (*Advanced Encryption Standard*), в основу якого ліг переможець міжнародного конкурсу алгоритм ***Rijndael***. Автори шифру Rijndael – бельгійські криптографи *Вінсент Реймен* та *Йоан Дамен*
2002 р.	Опубліковано ***SHA-2*** (*Secure Hash Algorithm-2*) – сімейство криптографічних хеш-функцій, що включає в себе алгоритми SHA-224, SHA-256, SHA-384, SHA-512, SHA-512/256 та SHA-512/224
2004 р.	Стартує європейський проект ***eSTREAM***, спрямований на відбір нових потокових алгоритмів для широкого використання
2007 р.	Оголошено конкурс NIST на розробку нового алгоритму хешування, оскільки кілька алгоритмів вже були атаковані, у тому числі були опубліковані серйозні атаки проти алгоритму SHA-1
2009 р.	Запущено мережу ***Bitcoin*** – криптографічно безпечна децентралізована система однорангових (Peer To Peer, P2P) електронних платежів, яка дозволяє здійснювати транзакції з використанням віртуальної валюти
2012 р.	Переможцем конкурсу на розробку нового алгоритму хешування став алгоритм ***Keccak,*** розроблений групою авторів на чолі з *Йоаном Даменом*. Цей хеш-алгоритм заснований на конструкції *sponge* (губка), яка є новим способом проектування хеш-функцій
2015 р.	Алгоритм Keccak стандартизовано як новий алгоритм хешування **SHA-3**
2016 р.	NIST оголосив конкурс на розробку найбільш оптимальних ***постквантових алгоритмів*** асиметричного шифрування та цифрового підпису
2020 р.	Оголошені алгоритми-фіналісти конкурсу NIST на розробку постквантових алгоритмів. Очевидними лідерами серед запропонованих криптосистем є *алгоритми постквантової криптографії на решітках*

СПИСОК ВИКОРИСТАНИХ ТА РЕКОМЕНДОВАНИХ ДЖЕРЕЛ

1. Бабенко Т.В. Криптологія у прикладах, тестах і задачах: навч. посібник / Т.В.Бабенко, Г.М.Гулак, С.О.Сушко, Л.Я.Фомичова. – Д.: Національний гірничий університет, 2013. – 318 с.
2. Блінцов В. С. Математичні основи криптології + CD : Навчальний посібник для студ. вищих навч. закл. / В. С. Блінцов, Ю. Л. Гальчевський. – Миколаїв : Національний ун-т кораблебудування ім. адмірала Макарова, 2006. – 232 с.
3. Бобало Ю. Я. Інформаційна безпека: навч. посібник / Ю. Я. Бобало, І. В. Горбатий, М. Д. Кіселичник та ін.; за заг. ред. д-ра техн. наук, проф. Ю. Я. Бобала та д-ра техн. наук, доц. І. В. Горбатого. – Львів : Видавництво Львівської політехніки, 2019. – 580 с.
4. Богуш В. М. Криптографічні застосування елементарної теорії чисел : Навч. посібник / В. М. Богуш, В. А. Мухачов. – К. : Державний ун-т інфо-рмаційно-комунікаційних технологій, 2006. – 126 с.
5. Болотов А. А. Элементарное введение в эллиптическую криптографию. Протоколы криптографии на эллиптических кривых / Болотов А. А., Гашков С. Б., Фролов А. Б. – М.: КомКнига, 2006. – 280 с.
6. Бурячок В.Л. Інформаційна та кібербезпека: соціотехнічний аспект: підручник / В. Л. Бурячок, В. Б. Толубко, В. О. Хорошко, С. В. Толюпа; за заг. ред. д-ра техн. наук, професора В. Б. Толубка. – К. : ДУТ, 2015. – 288 с.
7. Глинчук Л.Я. Криптологія: навч.-метод. посіб. / Л. Я. Глинчук – Луцьк: Вежа-Друк, 2014. – 164 с.
8. Горбенко І. Д. Захист інформації в інформаційно-телекомунікаційних системах: Навч. посіб. для студ. Ч. 1. Криптографічний захист інформації / І. Д. Горбенко, Т. О. Гріненко. – Х. : Харк. нац. ун-т радіоелектрон., 2004. – 368 с.
9. Горбенко І. Д. Прикладна криптологія: Теорія. Практика. Застосування / І. Д. Горбенко, Ю. І. Горбенко. – Харків: Форт, 2013. – 880 с.
10. Грайворонський М. В. Безпека інформаційно-комунікаційних систем: Підручник / М. В. Грайворонський, О. М. Новіков. – К. : Видавнича група BHV, 2009. – 608 с.
11. Грищук Р. В. Основи кібернетичної безпеки : монографія / Р. В. Грищук, Ю. Г. Даник; за заг. ред. проф. Ю. Г. Даника. – Житомир : ЖНАЕ, 2016. – 636 с.
12. ДСТУ 7624:2014. Інформаційні технології. Криптографічний захист інформації. Алгоритм симетричного блокового перетворення. – Введ. 01–07–2015. – К. :Мінекономрозвитку України, 2015.
13. Задірака В.К. Комп’ютерні технології криптографічного захисту інформації на спеціальних цифрових носіях: навч. посіб. / В. К. Задірака, А. М. Кудін, В. О. Людвиченко, О. С.Олексюк. – К. -Тернопіль: Підручники і посібники, 2007. – 272 с.
14. Козіна Г.Л. Криптографія від історії до сучасних стандартів: навч.посібник / Г. Л. Козіна. – Запоріжжя : НУ «Запорізька політехніка», 2020. – 192 с.
15. Корченко О. Г. Прикладна криптологія: системи шифрування : підручник / О. Г. Корченко, В. П. Сіденко, Ю. О. Дрейс. – К. : ДУТ, 2014. – 448 с.

16. Маркова І.І. Захист інформації. Криптографічні методи: Підручник для вищих навчальних закладів. / І.І. Маркова, А.І. Рибак, Ю.С. Ямпольський. – Одеса, 2001. – 175 с.
17. Методи та алгоритми симетричної криптографії: Навч. посіб. / Кузнецов О. О., Євсеев С. П., Смірнов О. А., Мелешко Є. В., Король О. Г. – Кіровоград: Вид. КНТУ, 2012. – 316 с.
18. Фергюсон Н. Практическая криптография / Фергюсон Н., Шнайер Б. – М.: Диалектика, 2005. – 424 с.
19. Штанько С.В. Эллиптические кривые в криптографии // Проблемы информационной безопасности. Компьютерные системы. 2003. № 2. С. 65 – 74
20. Alfred J. Menezes. Handbook of Applied Cryptography/ Alfred J. Menezes, Paul C. van Oorschot, and Scott A. Vanstone. Publisher: CRC Press, 2001. – 780 pages
21. Bruce Schneier. Applied cryptography: protocols, algorithms, and source code in C / 2nd ed. – New York : JohnWiley & Sons, Inc.,1995. – 792 pages.
22. Jean-Philippe Aumasson. Serious Cryptography: A Practical Introduction to Modern Encryption Paperback. Kindle Edition, 2017. – 313 pages.
23. Rivest R. How to leak a secret / Ronald L. Rivest, Adi Shamir, Yael Tauman // Proceedings of Asiacrypt 2001 – Springer-Verlag, 2001. – V 2248 of LNCS, pp. 552-565.
24. The CrypTool Portal [Електронний ресурс]. — Режим доступу : https://www.cryptool.org/en
25. The GNU Privacy Guard [Електронний ресурс]. – Режим доступу: https://gnupg.org/

ДЛЯ НОТАТОК

Навчальне видання

ОСНОВИ КРИПТОЛОГІЇ

Навчальний посібник

Підготували
Щур Наталія Олександрівна
Покотило Олександра Андріївна

Редактори
Комп’ютерне верстання –
Свідоцтво про реєстрацію № __ від _____ 202_ року
Підписано до друку __.__.21. Формат 60×84/16.
Ум. друк. арк. ____. Зам. __ офс.

Безкоштовно