МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДОНЕЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ВАСИЛЯ СТУСА
ФАКУЛЬТЕТУ ІНФОРМАЦІЙНИХ І ПРИКЛАДНИХ ТЕХНОЛОГІЙ

В. С. Ткаченко, Ю. В. Рассохіна, В. Г. Крижановський

ПРИКЛАДНА КРИПТОЛОГІЯ

Конспект лекцій
(друге видання, доповнене)

Вінниця
2025

УДК 004.056.55:003.26(075.8)
П 759

Рекомендовано до друку вченою радою
факультету інформаційних і прикладних технологій
Донецького національного університету імені Василя Стуса
(протокол № 4(8) від 25.02.2025)

Укладачі:
Ткаченко В. С., доцент кафедри фундаментальної та прикладної хімії Донецького національного університету імені Василя Стуса;
Рассохіна Ю. В., проф. кафедри фундаментальної та прикладної хімії Донецького національного університету імені Василя Стуса;
Крижановський В. Г., проф. кафедри прикладної математики та кібербезпеки Донецького національного університету імені Василя Стуса.

Рецензенти:
Ніколюк П. К., доктор фіз.-мат. наук, професор, проф. кафедри інформаційних технологій Донецького національного університету імені Василя Стуса;
Штовба С. Д., доктор техн. наук, професор, проф. кафедри інформаційних технологій Донецького національного університету імені Василя Стуса.

П 759 Прикладна криптологія: конспект лекцій для здобувачів вищої освіти ОС «Бакалавр» спеціальності 125 Кібербезпека та захист інформації. 2 вид., доп. / уклад. В. С. Ткаченко, Ю. В. Рассохіна, В. Г. Крижановський. Вінниця: ДонНУ імені Василя Стуса, 2025. 84 с.

У конспекті лекцій розглянуто математичні основи й історію криптології, класифікацію та структуру основних систем шифрування (а саме симетричне шифрування) в інформаційних системах. Конспект укладено відповідно до навчальної програми дисципліни «Прикладна криптологія» та рекомендовано для здобувачів освіти спеціальності 125 Кібербезпека та захист інформації.

УДК 004.056.55:003.26(075.8)

ЗМІСТ

ТЕМА 1. ОСНОВНІ ПОНЯТТЯ ТА ВИЗНАЧЕННЯ (ЛЕКЦІЯ 1)

1. Предмет, мета та завдання курсу «Прикладна криптологія».
2. Роль криптології у сучасному суспільстві.
3. Криптографічний захист інформації і актуальні проблеми, з ним пов'язані.
4. Основна термінологія курсу.

1. Предмет, мета та завдання курсу «Прикладна криптологія»

Предметом навчальної дисципліни є:

– загальні принципи побудови систем криптографічного захисту інформації;

– ключові системи, криптоалгоритми і криптопротоколи, що становлять основу криптографічного захисту інформації в сучасних комп'ютерних мережах, і їх криптографічні властивості;

– загальні підходи щодо вибору параметрів криптосистем, алгоритми їх побудови і тестування;

– загальна методика організації підсистем криптографічного захисту інформації в комп'ютерних системах і мережах.

Мета вивчення навчальної дисципліни полягає в тому, щоб навчитись вирішувати питання відповідності засобів криптографічного захисту інформації, що плануються до впровадження, сучасним вимогам, а також розвинути навички практичного застосування стандартних підходів для забезпечення належного рівня криптографічного захисту інформації мережі зв'язку.

Завданнями навчальної дисципліни є формування наступних навичок:

– вміти характеризувати принципи побудови сучасних криптографічних систем, орієнтуватися в термінології і формулюваннях теоретичних результатів щодо їх стійкості;

– вміти надати рекомендації щодо використання стандартних криптоалгоритмів і криптопротоколів для забезпечення належного рівня захисту інформації;

– вміти характеризувати суттєві параметри та потенційні слабкості типових криптографічних алгоритмів і протоколів;

– вміти використовувати і реалізовувати стандарти в галузі криптографічного захисту інформації;

– вміти аналізувати перспективи розвитку криптографії та криптоаналізу.

2. Роль криптології у сучасному суспільстві

З використанням інтернету та інших відкритих технологічних рішень у якості каналу передачі даних особливого значення набуває питання забезпечення конфіденційності, цілісності та достовірності переданої інформації.

Проблема захисту інформації шляхом перетворення, що виключає її прочитання сторонньою особою, хвилювала людський розум з давніх часів. Історія криптографії – майже ровесниця історії людської мови. Спочатку писемність сама по собі була криптографічного системою, оскільки у древніх суспільствах нею володіли лише обрані (наприклад, Священні книги Стародавнього Єгипту, Стародавньої Індії). Із поширенням писемності криптографія стала формуватися як самостійна наука. Перші криптосистеми зустрічаються вже на початку нашої ери. Так, Цезар у своєму листуванні використовував більш-менш систематичний шифр, який отримав його ім'я.

Бурхливий розвиток криптографічні системи отримали в роки Першої та Другої світових воєн. Поява в повоєнний час обчислювальних пристроїв прискорила розробку та вдосконалення криптографічних методів, і це вдосконалення триває і дотепер.

Чому проблема використання криптографічних методів в інформаційних системах сьогодні особливо актуальна? З одного боку, розширилося використання комп'ютерних мереж, по яких передаються великі обсяги інформації державного, військового, комерційного і приватного характеру, що не допускає можливості доступу до неї сторонніх осіб. З іншого, – поява нових потужних комп'ютерів, технологій мережевих і нейронних обчислень зробила можливою дискредитацію криптографічних систем, які ще недавно вважалися такими, що не розкриваються.

Проблемою захисту інформації шляхом її перетворення займається криптологія (*kryptos* – «таємний», *logos* – «наука»).

Криптологія традиційно поділяється на ***криптографію*** і ***криптоаналіз.*** Методи криптографії орієнтовані на створення систем захисту інформації. До області криптоаналізу належать підходи до здійснення несанкціонованого доступу аналітичним шляхом до інформації, захищеної криптографічними методами. Криптоаналітики досліджують питання стійкості абстрактних і реальних криптосистем. Обидва напрями криптології взаємодіють, доповнюючи один одного, що дає змогу вирішити низку зазначених актуальних завдань і впровадити у практику ефективні системи криптографічного захисту інформації.

3. Криптографічний захист інформації і актуальні проблеми, з ним пов'язані

Незважаючи на неперервний розвиток та вдосконалення систем криптографічного захисту інформації, існують причини, через які розроблені криптосистеми треба розглядати як потенційно ненадійні:

1. Відсутність повних формальних критеріїв якості криптосистем, а також неможливість на практиці виконання (в повному обсязі) вимог, з огляду на які розробники обґрунтовували висновок про достатній рівень захисту інформації.

2. У разі масового застосування криптосистем проявляються невраховані малоймовірні ситуації, в яких стійкість криптосистеми знижується.

3. Додатковим джерелом потенційної ненадійності криптосистем є можливість створювати криптосистеми зі свідомо внесеними слабкостями.

З цих положень можна зробити висновки про наявність факторів, ***специфічних для практичної криптології:***

1. Неможливість допомоги ззовні для розвитку криптографічного потенціалу держави.

2. Неминучість прихованого змагання між криптографами і криптоаналітиками, а також у параметрах обчислювальної потужності в глобальному масштабі.

3. Необхідність уніфікації криптосистем до стандартів, прийнятих у розвинених країнах (політичний і економічний фактори).

4. Необхідність випереджаючого оволодіння технологіями, які забезпечували б захист інформації на сучасному науково-технічному рівні в умовах обмеженого доступу до сучасних розробок і проєктів.

4. Основна термінологія курсу

Криптографія – дисципліна, що включає принципи, засоби і математичні методи перетворення інформації з метою приховування змісту або структури даних, а також для захисту їх від несанкціонованого використання або підробки.

Основним методом криптографії є *шифрування*.

Шифруванням називається взаємно-однозначне перетворення повідомлення з метою приховування його змісту від сторонніх осіб.

Оригінальним текстом повідомлення є відкритий текст.

Результатом шифрування є шифрований текст *(шифротекст)*.

Під час шифрування можуть використовуватися секретні параметри – *ключі*.

Відкритий текст складається з елементів, які визначаються шифроперетворенням.

Елемент – це найменша частина даних (набір бітів), яка може бути зашифрована. Елементам відкритого тексту відповідають елементи шифротексту.

Для передачі по каналах зв'язку шифротекст поєднується разом з додатковими даними (наприклад, адресами кореспондентів) і компонується у послідовність, що називається *криптограмою*.

Особа, на адресу якої була спрямована криптограма, виконує над нею ряд певних операцій, що дають змогу їй відновити відкритий текст повідомлення. Ця процедура називається *розшифруванням криптограми*.

Створення системи секретного зв'язку включає в себе, крім іншого, розробку власне криптосистеми, під якою, в широкому сенсі, розуміються документи, пристрої, обладнання та відповідні методи, використання яких (у сукупності), забезпечує засоби шифрування і розшифрування (відновлення початкового повідомлення).

У вузькому сенсі під криптосистемою розуміють так звану систему шифрування, в межах якої розглядають лише методи, що забезпечують засоби шифрування і розшифрування під час обміну інформацією.

Якщо кому-небудь зі сторонніх осіб вдається шляхом аналізу криптограми отримати відкритий текст, то кажуть, що криптограма була дешифрована (розкрита). Зауважимо, що дешифрування криптограми не обов'язково вимагає знання ключа.

Існують системи, які теоретично неможливо дешифрувати. Всі інші системи мають певні слабкості, тому можливість дешифрування або значного ослаблення шифру не можна не брати до уваги.

З одного боку, для стійкості криптосистеми важливо знати, наскільки всебічно і ретельно враховані всі особливості роботи такої системи в умовах, за яких вона застосовується. З іншого боку, під час впровадження засобів криптографічного захисту необхідно зберігати баланс між їх вартістю і необхідною стійкістю.

Питання для самоконтролю

1. Що є предметом вивчення дисципліни «Прикладна криптологія»?
2. У чому полягає актуальність використання криптографічного захисту інформації?
3. Що називають криптоаналізом?
4. Що називають шифруванням?
5. Що таке «відкритий текст»?
6. У чому різниця між розшифруванням та дешифруванням?

ТЕМА 2. ОСНОВНІ ЕТАПИ РОЗВИТКУ КРИПТОГРАФІЇ (ЛЕКЦІЯ 2)

1. Наївна криптографія.
2. Формальна криптографія.
3. Математична криптографія.

1. Наївна криптографія

Про важливість збереження інформації в таємниці знали вже у стародавні часи, коли з появою писемності з'явилася і небезпека прочитання її небажаними особами. Навіть більше, спочатку писемність сама по собі була криптографічною системою, оскільки у древніх суспільствах нею володіли лише обрані. Із поширенням писемності криптографія стала формуватися як самостійна наука. У документах стародавніх цивілізацій – Індії, Єгипту, Месопотамії – є відомості про системи і способи складання шифрованих текстів.

Для наївної криптографії (до початку XVI ст.) характерним є використання будь-яких, зазвичай примітивних, способів заплутування супротивника щодо змісту переданих повідомлень. На початковому етапі для захисту інформації використовувалися методи кодування і стеганографії, які споріднені, але не тотожні криптографії. Шифрувальні системи зводилися до використання перестановки або заміни букв на різні символи (інші букви, знаки, малюнки, числа тощо). Одні й ті самі методи шифрування використовувалися повторно, ключі були короткими, використовувалися примітивні способи перетворення вихідної інформації в зашифроване повідомлення. Це давало змогу, один раз встановивши алгоритм шифрування, швидко розшифровувати повідомлення.

Точний час виникнення способів обміну таємною інформацією встановити неможливо. Історики вважають, що перші протокриптографічні прийоми з'явилися у Стародавньому Єгипті приблизно 4 тис. років тому. Писарі, які складали опис життя правителів, прагнули надати стандартним ієрогліфам на пам'ятниках і гробницях незвичайного вигляду, щоб надати написам урочистого й більш шанобливого стилю. Жреці використовували цей самий прийом під час переписування релігійних текстів, щоб ці тексти здавалися простому народу таємничішими та викликали більше поваги.

Зі збільшенням кількості написів на стінах храмів люди втрачали до них інтерес. Єгиптологи вважають, що писарі тоді стали ще більше видозмінювати деякі знаки в прагненні пробудити інтерес і привернути увагу населення. Ці модифікації в жодному разі не були кодами або шифрами, але вони містили в собі два основні принципи криптології, а саме: зміну шрифту і приховування вмісту тексту. Безперечних доказів, що вказують на широке використання модифікацій ієрогліфів для приховування дипломатичних, торговельних або військових планів у Стародавньому Єгипті, немає.

Більш явні приклади залишились від цивілізацій Месопотамії – від вавилонян, ассирійців, халдеїв, які використовували особливу систему письма – клинопис. У 1500 р. до н. е. на глиняній табличці був записаний рецепт глазурі для гончарних виробів. На ньому знаки, що визначають необхідні інгредієнти, були навмисно перемішані. Отже, ми маємо право стверджувати, що ця табличка є найбільш раннім відомим секретним написом.

Приблизно з 500 р. до н. е. в Індії також широко застосовувалися секретні написи, зокрема в донесеннях шпигунів. Методи засекречування інформації включали в себе фонетичну заміну, коли приголосні і голосні мінялися місцями, використовувалися перевернуті букви і запис тексту під випадковими кутами.

Одним із перших документально зафіксованих прикладів використання таємних шифрів є шифр Цезаря, що полягає в заміні кожної букви вихідного тексту на іншу, віддалену від неї в алфавіті на певну кількість позицій. Інший шифр, квадрат Полібія, авторство якого приписується грецькому письменникові Полібію, є шифром простої однозначної заміни. У квадрат виписувалися літери алфавіту (для грецького алфавіту розмір становив 5×5). Кожна літера вихідного тексту замінювалося на пару цифр – номер рядка і стовпця, на перетині яких стояла буква початкового тексту, який треба було зашифрувати.

Із VIII ст. н. е. розвиток криптографії відбувається в основному в арабських країнах. Вважається, що арабський філолог Халіль аль-Фараxіфі першим звернув увагу на можливість використання стандартних фраз відкритого тексту для дешифрування. Він припустив, що першими словами в листі на грецькій мові візантійського імператора будуть «В ім'я Аллаха», що дало йому змогу прочитати решту повідомлення. Пізніше він написав книгу з описом методу – «Кітаб аль-Муамма» («Книга таємної мови»). У 855 р. з'являється «Книга про великі прагнення людини розгадати загадки древньої писемності» арабського вченого Абу Бакр Ахмед ібн Алі Ібн Вахшія ан-Набаті, одна з перших книг із криптографії з описами декількох шифрів, зокрема із застосуванням декількох алфавітів. Також до IX ст. належить перша відома згадка про частотний криптоаналіз – у книзі Аль-Кінді «Манускрипт про дешифрування криптографічних повідомлень». У 1412 р. виходить 14-томна енциклопедія Шихаба ал-Калкашанді «Субхі ал-Ааша», один з розділів якої з назвою «Щодо приховування в буквах таємних повідомлень» містив опис семи шифрів заміни та перестановки, частотного криптоаналізу, а також таблиці частоти появи літер арабською мовою на основі тексту Корану. Саме арабські вчені того часу сформулювали поняття алгоритм і шифр.

Навіть у стародавні часи існували найпростіші криптографічні пристрої.

Грецький поет Архілох у VII ст. до н. е. згадує пристрій під назвою сцитала. Достовірно відомо, що сцитала використовувалася у війні Спарти проти Афін у кінці V ст. до н. е. Цей пристрій являє собою циліндр (іноді жезл командувача) і

вузьку смужку пергаменту, що обмотують навколо жезла по спіралі. Текст писався на пергаментній стрічці по довжині палички. Після того, як довжина палички закінчувалася, вона поверталася, і текст писався далі, поки або не закінчувався текст, або не закінчувалася стрічка. Для розшифровки отримувач використовував паличку такого самого діаметру, на яку він намотував пергамент, щоб прочитати повідомлення. Античні греки, зокрема спартанці, використовували цей шифр для зв'язку під час військових кампаній. Однак такий шифр може бути легко зламаний. Наприклад, метод злому сцитали був запропонований ще Арістотелем. Він полягає в тому, що, не знаючи точного діаметру палички, можна використовувати конус, який має змінний діаметр, і переміщувати пергамент із повідомленням по його довжині доти, поки текст не почне читатися – у такий спосіб дешифрується діаметр сцитали.

Іншим широко відомим криптографічним пристроєм захисту інформації був «диск Енея» – інструмент для захисту інформації, придуманий Енеєм Тактиком у IV ст. до н. е. Пристроєм був диск діаметром 13–15 см і товщиною 1–2 см з проробленими в ньому отворами, кількість яких дорівнювала кількості букв в алфавіті. Кожному отвору ставилася у відповідність конкретна буква. У центрі диска знаходилася котушка з намотаною на неї ниткою. Механізм шифрування був дуже простий. Для того, щоб зашифрувати послання, необхідно було по черзі протягувати вільний кінець нитки через отвори, які позначають літери вихідного, не зашифрованого, повідомлення. Отже, сам диск із протягнутою в його отвори ниткою і був зашифрованим посланням. Одержувач повідомлення послідовно витягував нитку з кожного отвору, в такий спосіб отримуючи послідовність літер. Але ця послідовність була зворотною відносно вихідного повідомлення, тобто він читав повідомлення навпаки. Зашифроване повідомлення було доступне до прочитання будь-кому, хто зміг заволодіти диском. На цей випадок було передбачене швидке знищення зашифрованої інформації. Для цього було достатньо витягнути всю нитку за один з її кінців або зламати диск, просто наступивши на нього. Насправді «диск Енея» не можна назвати справжнім криптографічним інструментом, оскільки прочитати повідомлення міг будь який охочий. Але цей пристрій став прабатьком першого криптографічного інструменту, винахід якого також належить Енею – «Лінійки Енея».

Лінійка Енея являла собою пристрій, що мав отвори, кількість яких дорівнювала кількості букв алфавіту. Кожен отвір було позначено своєю буквою, причому літери по отворах розташовувалися у довільному порядку. До лінійки була прикріплена котушка з намотаною на неї ниткою. Поряд із котушкою був проріз. Під час шифрування нитка протягувалася через проріз, а потім через отвір, відповідний першій літері тексту. Під час цього на нитці зав'язувався вузлик у місці проходження її через отвір; потім нитка поверталася у проріз і аналогічно зашифрову-

валася друга та всі інші букви тексту. Після закінчення шифрування нитка витягувалася і передавалася одержувачу повідомлення. Одержувач, маючи ідентичну лінійку, протягував нитку через проріз до отворів, які визначаються вузлами, і відновлював вихідний текст по буквах отворів. Такий шифр є одним із прикладів шифру заміни: коли літери замінюються на відстані між вузликами з урахуванням проходження через проріз. Ключем шифру був порядок розташування букв по отворах у лінійці. Стороння особа, яка отримала б нитку (навіть маючи лінійку, але без нанесених на ній букв), не змогла б прочитати передане повідомлення.

2. Формальна криптографія

Етап формальної криптографії (кінець XV – початок XX ст.) пов'язаний з появою формалізованих і стійких до «ручного криптоаналізу» шифрів. У європейських країнах це відбулося в епоху Відродження, коли розвиток науки й торгівлі призвів до необхідності використання нових надійних способів захисту інформації.

До кінця XIV ст. між італійськими містами-державами в листуванні вже застосовувалися «номенклатори» (лат. *nomen* – «ім'я» і *calator* – «раб», «слуга»). Вони складалися з кодових позначень для складів, слів та імен, а також алфавітів шифрозамін. До XIX ст. номенклатори залишалися широко використовуваною системою приховування змісту повідомлень.

Симеон де Крема був першим (1401), хто використовував омофонічні таблиці для приховування частоти появи голосних у тексті за допомогою більш ніж однієї шифрозаміни.

Засновником західноєвропейської криптографії називають вченого епохи Відродження Леона Баттіста Альберті. Вивчивши методи зламу шифрів, що використовувалися в Європі (по суті, моноалфавітних шифрів однозначної заміни), він спробував створити шифр, який був би стійкий до частотного криптоаналізу. Його «Трактат про шифри» був представлений у папську канцелярію в 1466 р. і вважається першою в Європі науковою працею з криптографії. Альберті запропонував замість одного секретного алфавіту використовувати два або більше, перемикаючись між ними за якимось правилом (тобто запропонував поліалфавітні шифри). Однак він так і не зміг оформити своє відкриття в реально працюючу систему. Це було зроблено його послідовниками (наприклад, шифр Віжинера).

Іншою роботою, в якій узагальнено і сформульовано відомі на той момент алгоритми шифрування, є праця «Поліграфія» (1518) німецького абата Йоганна Трисемуса (Тритемія). Він першим помітив, що шифрувати можна і по дві букви за раз – використовувати біграми (хоча перший біграмний шифр Playfair був запропонований лише в XIX ст.).

У 1550 р. італійський математик Джироламо Кардано запропонував нову техніку шифрування – решітка Кардано. Цей спосіб поєднував у собі стеганографію

та криптографію. Важко було зрозуміти, що отримане повідомлення містить зашифрований текст, а розшифрувати його, не маючи ключа (решітки), в той час було практично неможливо. Цей шифр вважають першим транспозиційним шифром, або, як ще його називають, геометричним шифром, заснованим на розташуванні букв у шифротексті.

Значний поштовх розвитку криптографії дав винахід телеграфу. Передача даних перестала бути секретною, і повідомлення теоретично міг перехопити хто завгодно. Інтерес до криптографії зріс, зокрема і серед простого населення, внаслідок чого було багато спроб створити індивідуальні системі шифрування. Перевага телеграфу була особливо помітною у військовій сфері, де командувач повинен був віддавати негайні накази, а також отримувати інформацію з місць подій. Це послугувало поштовхом до розвитку військових, польових шифрів.

У 1863 р. Фрідріх Касіскі (англ. Friedrich Kasiski) опублікував метод, згодом названий його ім'ям, що давав змогу швидко й ефективно розкривати практично будь-які шифри того часу, зокрема й поліалфавітні. Метод складався з двох частин – визначення періоду шифру і дешифрування тексту з використанням частотного криптоаналізу.

У 1883 р. голландець Огюст Керкгоффс опублікував працю під назвою «Військова криптографія». У ній він описав шість вимог (що наведені нижче), які має задовольняти захищена система. До деяких із них варто ставитися критично.

1. Шифр не має розкриватись математичними методами або фізично.

2. Система не має бути секретною на випадок, якщо вона потрапить в руки ворога.

3. Ключ має бути простим, зберігатися у пам'яті без запису на папері, а також легкозмінним за необхідності.

4. Зашифрований текст має без проблем передаватися телеграфом.

5. Апарат для шифрування можна транспортувати, робота з ним не має вимагати допомоги кількох осіб.

6. Апарат для шифрування має бути відносно простим у використанні, не вимагати значних розумових зусиль або дотримання великої кількості правил.

Відомий «принцип Керкгоффса» являє собою правило розробки криптографічних систем, згідно з яким у засекреченому вигляді тримається тільки певний набір параметрів алгоритму, ключ, а сам алгоритм шифрування має бути відкритим. Інакше кажучи, під час оцінки надійності шифрування необхідно передбачати, що противник знає про використовувану систему шифрування все, крім застосовуваних ключів.

У 1920 р. вийшла монографія американського криптографа Вільяма Ф. Фрідмана «Індекс збігу і його застосування у криптографії» (англ. «Index of Coincidence and Its Applications in Cryptography»). Робота вийшла у відкритій пресі, незважаючи на те, що була виконана в межах військового замовлення. Двома роками пізніше Фрідман ввів у науковий обіг терміни «криптологія» і «криптоаналіз».

Перед початком Другої світової війни провідні світові держави мали електромеханічні шифрувальні пристрої, результат роботи яких вважався на той час невразливим до розкриття. Ці пристрої ділилися на два типи – роторні машини і машини на цівочних дисках. До першого типу належить «Енігма», що використовувалася військами Німеччини та її союзників, другого типу – американська М-209.

3. Математична криптографія

Після Першої світової війни уряди країн засекретили всі роботи в галузі криптографії. На початку 1930-х рр. остаточно сформувалися розділи математики, що є основою для майбутньої науки: загальна алгебра, теорія чисел, теорія ймовірностей і математична статистика. До кінця 1940-х рр. побудовано перші програмовані машини, закладено основи теорії алгоритмів, кібернетики. Проте, після Першої світової війни і до кінця 1940-х рр. у відкритій пресі було опубліковано зовсім небагато робіт і монографій. Найбільший прогрес у криптографії досягається у військових відомствах.

Ключовим пунктом у розвитку криптографії є фундаментальна праця Клода Шеннона «Теорія зв'язку в секретних системах» (англ. Communication Theory of Secrecy Systems) – секретна доповідь, представлена автором у 1945 р., і опублікована ним у «Bell System Technical Journal» у 1949 р. В цій роботі, на думку багатьох криптографів, був вперше показаний ***підхід до криптографії як до математичної науки.***

У 1960-х рр. почали з'являтися різні блокові шифри, яким притаманна більша криптостійкість, порівняно з результатом роботи роторних машин. Такі шифри передбачали обов'язкове використання цифрових електронних пристроїв – ручні або напівмеханічні способи шифрування вже не використовувалися.

У 1967 р. виходить книга Девіда Кана «Зломщики кодів». Хоча книга не містила яких-небудь нових відкриттів, вона детально описувала наявні на той момент результати в області криптографії, великий історичний матеріал, включно з успішними випадками використання криптоаналізу. Книга мала помітний комерційний успіх і познайомила з криптографією десятки тисяч людей. З цього моменту почали потроху з'являтися роботи і у відкритій пресі.

Приблизно в цей самий час Хорст Фейстель переходить із Військово-повітряних сил США на роботу в лабораторію корпорації IBM. Там він займається розробкою нових методів криптографії і розробляє сітку Фейстеля, яка є основою багатьох сучасних шифрів, в тому числі шифру Lucifer, що став прообразом шифру DES – в минулому стандарту шифрування США, першого у світі відкритого державного стандарту на шифрування даних. На основі сітки Фейстеля були створені й інші блокові шифри, в тому числі TEA (1994), Twofish (1998), IDEA (2000), а також радянський шифр ГОСТ 28147-89, що довгий час використовувався у пострадянських країнах.

У 1976 р. було опубліковано роботу Уітфілда Діффі і Мартіна Хеллмана «Нові напрями у криптографії» (англ. «New Directions in Cryptography»). Ця робота відкрила нову область у криптографії – криптографію з відкритим ключем. Також у роботі містився опис алгоритму Діффі–Хеллмана–Меркла, який давав змогу сторонам згенерувати секретний ключ, використовуючи відкритий канал зв'язку. Але першою реальною криптосистемою з відкритим ключем вважають алгоритм RSA (названий за ім'ям авторів – Рон Рівест (R. Rivest), Аді Шамір (A. Shamir) і Леонард Адлеман (L. Adleman)). Опублікована у серпні 1977 р. робота дала змогу сторонам обмінюватися секретною інформацією, не маючи заздалегідь обраного секретного ключа. Варто зазначити, що і алгоритм Діффі–Хеллмана–Меркла, і RSA були вперше відкриті в англійських спецслужбах, але не були ані опублікованими, ані запатентованими через секретність.

Створення *асиметричних криптосистем* підштовхнуло математиків і криптоаналітиків до вивчення способів факторизації, дискретного логарифмування, операцій над еліптичними кривими у скінченному полі та інших математичних основ криптографії.

Відносно новим методом є ймовірнісне шифрування, яке запропонували Шафі Гольдвассер (Goldwasser) і Сільвіо Мікелі (Micali). Шифрування було назване «ймовірнісним» через те, що один і той самий вхідний текст для шифрування з використанням одного і того самого ключа може перетворюватися в абсолютно різні шифротексти.

Чарльз Беннет (Charles Bennet) і Жиль Брассар (Gilles Brassard), спираючись на роботу Стівена Уіснера (Stephen Wiesner), розробили теорію квантової криптографії, яка базується радше на аналогії з законами квантової фізики, ніж на математиці. Процес відправлення та прийому інформації описується за допомогою фотонів. Заснована на принципах квантової механіки, ця система, на відміну від звичайної криптографії, теоретично дає змогу гарантовано захистити інформацію від зловмисника, навіть якщо той володіє найсучаснішою технологією і необмеженими обчислювальними потужностями. Проте наразі розробляються тільки прототипи квантових криптосистем.

Застосування криптографії у вирішенні питань автентифікації, перевірки цілісності даних, передачі конфіденційної інформації по каналах зв'язку тощо стало невід'ємним атрибутом інформаційних систем. У сучасному світі криптографія знаходить безліч різних застосувань – вона використовується в системах мобільного зв'язку, в цифровому телебаченні, під час підключення до Wi-Fi, для захисту квитків від підробок на транспорті, в банківських операціях, у системах електронних платежів та ін.

Питання для самоконтролю

1. Назвіть основні етапи розвитку криптографії.

2. Коли та де, за версією істориків, з’явились перші прообрази криптографічних прийомів?

3. Які особливості характеризують період наївної криптографії?

4. Наведіть приклади шифрів періоду формальної криптографії.

5. У чому полягає принцип та вимоги до шифрів Керкгоффса?

6. У якій роботі був вперше показаний підхід до криптографії як до математичної науки?

7. Наведіть приклади шифрів на основі сітки Фейстеля.

ТЕМА 3. ПОВТОРЕННЯ ОКРЕМИХ МАТЕМАТИЧНИХ ВИЗНАЧЕНЬ ТА ПРАВИЛ (ЛЕКЦІЯ 3)

1. Множини та алгебраїчні операції.
2. Довідка з теорії груп.
3. Багаточлени, дії над ними та алгоритм Евкліда.
4. Наведені нижче математичні визначення, теореми та правила дають змогу розуміти опис сучасних алгоритмів математичної криптографії.

1. Множини та алгебраїчні операції

Нехай X – це довільна множина. Бінарною алгебраїчною операцією (або законом композиції) на X називається відображення $f: X \times X \to X$, якщо будь-якій впорядкованої парі $(a, b) \in X$ ставиться у відповідність певний елемент $c = f(a, b)$, що належить цій самій множині. Іноді замість $f(a, b)$ пишуть $c = afb$, а ще частіше бінарну операцію на X позначають якимось спеціальним символом: $*$, $\circ$, $\cdot$ або $+$.

На множині X може бути задано загалом декілька різних операцій. Бажаючи виділити одну з них, скажімо, $\circ$, використовують дужки $(X,\circ)$, і кажуть, що операція визначена на алгебраїчній структурі. Операції в алгебраїчній структурі можуть бути узгоджені, тобто одна операція може задовольняти деякі вимоги, відносно того, як її застосування впливає на результати застосування іншої операції.

2. Довідка з теорії груп

Бінарна операція $*$ на множині X називається асоціативною, якщо $(a * b) * c = a * (b * c)$ для всіх $a, b, c \in X$. Вона також називається комутативною або абелевою, якщо $a * b = b * a$.

Ті ж назви присвоюються і відповідній алгебраїчній структурі $(X,*)$. Вимоги асоціативності і комутативності є незалежними.

Множина X із заданою на ній бінарною асоціативною операцією називається *напівгрупою*.

Елемент $e \in X$ називається *одиничним* (або *нейтральним*) відносно даної бінарної операції $*$, якщо $e * x = x * e$ для всіх $x \in X$. Якщо e_1 – ще один одиничний елемент, тоді $e_1 = e * e_1 = e_1 * e = e$. Отже, в алгебраїчній структурі $(X,*)$ може існувати не більше одного одиничного елементу.

Напівгрупу $(X,*,e)$ з одиничним елементом e прийнято називати *моноїдом*.

Елемент a моноїда $(M,\cdot,e)$ називається *оберненим*, якщо знайдеться елемент $b \in M$, для якого $a \cdot b = b \cdot a = e$. Обернений (зворотний) до a елемент позначається через a^{-1}. Таке позначення є коректним, тому що існує лише єдиний зворотний елемент. Дійсно, $a \cdot b_1 = b_1 \cdot a = e \Rightarrow b_1 = e \cdot b_1 = (b \cdot a) \cdot b_1 = b \cdot (a \cdot b_1) = b \cdot e = b$. Запис операції в вигляді $a \cdot b$ називається мультиплікативним, а сама операція називається множенням.

Зауважимо, що під час мультиплікативного запису знак операції часто опускається: $a \cdot b = ab$.

Моноїд G, всі елементи якого є оборотними, називається *групою*. Інакше кажучи, група передбачає виконання таких аксіом:

1) на множині G визначена бінарна операція $f: (x, y) \to xy$;

2) операція f асоціативна;

3) у множині G існує нейтральний щодо f елемент;

4) для кожного елементу $x \in G$ існує обернений (зворотний) елемент.

Група називається *скінченною,* якщо число її елементів буде скінченним. Для позначення числа елементів у групі G використовуються позначення $|G|$, $\#G$, $\mathrm{Card}\ G$.

Кількість елементів скінченної групи називається її *порядком*.

Підмножина групи називається підгрупою групи, якщо вона також є групою. Аналогічно визначаються підструктури інших алгебраїчних структур.

Підмножина H групи $(G,*)$ є підгрупою групи G тоді і тільки тоді, якщо обернений елемент до будь-якого елементу з H, а також добуток будь-яких двох елементів з H, належать H.

Теорема Лагранжа. Порядок скінченної групи ділиться на порядок будь-якої її підгрупи.

Група $(G,*)$ називається *комутативною,* якщо $\forall a, b \in G\ a * b = b * a$. Для комутативної групи часто використовується запис операції у вигляді $a + b$. У цьому випадку операція називається *додаванням*. У загальному випадку групи виду $(G, +)$ називають *адитивними*.

Зауваження. Нехай n – натуральне число. Тоді для $a \in (G,\cdot)$ запис a^n позначає добуток однакових множників: $a^n = aa \ldots a$. Вважаючи, що $a^{-n} = a^{-1}a^{-1} \ldots a^{-1}$ і $a^0 = e$, отримуємо можливість оперувати з показниками за звичайними правилами алгебри.

Аналогічним способом для адитивної групи, виходячи з суми n однакових доданків: $[n] \cdot a = a + a + \cdots + a$, можна ввести операцію множення елементу групи на ціле число.

Групи $(G,*)$ і $(G_1,\circ)$ *гомоморфні,* якщо існує відображення $f: G \to G_1$, таке, що для $\forall a, b \in G\ f(a * b) = f(a) \circ f(b)$. Відображення f у цьому випадку називається *гомоморфізмом груп*.

Ядром гомоморфізму $f: G \to G_1$ називається множина $\mathrm{Ker} f \subset G$, що є прообразом одиничного елементу $e_1 \in G_1$.

Групи $(G,*)$ та $(G_1,\circ)$ є *ізоморфними*, якщо для них існує гомоморфізм f з $(G,*)$ у $(G_1,\circ)$, причому відображення є взаємно однозначним.

Відображення f є *автоморфізмом* групи $(G,*)$, якщо відображення – ізоморфізм $f: G \to G$.

Відображення f є *ендоморфізмом* групи $(G,*)$, якщо відображення $f: G \to G$ – гомоморфізм.

Асоціативним кільцем називається множина $D = (D, +, \cdot)$ з двома операціями, які називаються додаванням та множенням, і для яких виконуються такі аксіоми:

1. Асоціативність щодо додавання: $a + b + c = (a + b) + c = a + (b + c)$.
2. Комутативність щодо додавання: $a + b = b + a$.
3. Можливість розв'язання рівняння $a + x = b$ для всіх $a, b \in D$.
4. Асоціативність щодо множення: $a \cdot b \cdot c = (a \cdot b) \cdot c = a \cdot (b \cdot c)$.
5. Дистрибутивність щодо добутку ліворуч: $a \cdot (b + c) = a \cdot b + a \cdot c$.
6. Дистрибутивність щодо добутку праворуч: $(b + c) \cdot a = b \cdot a + c \cdot a$.

Зазвичай під терміном «кільце» розуміється асоціативне кільце.

Кільце називається неасоціатівним, якщо операція множення не є асоціативною. Кільце називається комутативним, якщо операція множення комутативна.

Аналогічно до групи, в кільці існує одиничний елемент щодо операції додавання. Він називається нулем і часто записується як число нуль.

Одиничний елемент e за множенням з властивістю $ae = ea,\ \forall a \in D$, не обов'язково має існувати. У разі його відсутності в кільці може існувати кілька односторонніх одиничних елементів, лівих або правих (але не одночасно). Множення на ці елементи з відповідної сторони не впливає на результат операції. Якщо в кільці відсутній одиничний елемент за множенням, то кільце називається кільцем без одиниці.

Прикладом комутативного кільця без одиниці є множина парних чисел зі звичайними операціями додавання і множення.

У кільці з одиницею можливе існування елементу a^{-1}, зворотного (оберненого) до елементу a, за умови $a^{-1}a = aa^{-1} = e$. Такі елементи називаються оборотними.

Множина оборотних елементів кільця з одиницею становить мультиплікативну групу кільця. Множина всіх одиниць кільця D називається групою одиниць D^* і позначається або $U(D)$.

Прикладом комутативного кільця з одиницею є множина цілих чисел. Група одиниць цього кільця складається з двох елементів: ± 1.

Полем F називається комутативне кільце, для будь-якого ненульового елементу якого існує зворотний (обернений).

Зауваження. Нехай $a, b \in F,\ b \neq 0$. Тоді діленням елементу a на елемент b називається операція $(a, b) \to ab^{-1}$.

Прикладом поля може слугувати поле звичайних раціональних чисел (дробів). Це поле містить нескінченну кількість елементів. Класичними прикладами є також поле дійсних чисел і поле комплексних чисел.

Існують поля, що складаються зі скінченного числа елементів. Такі поля називають *полями Галуа*. Виявляється, що число q елементів скінченного поля завжди є степенем деякого простого числа p: $q = p^a$. Поле Галуа, що складається з q елементів, позначається F_q або $GF(q)$. Оскільки мультиплікативна група F_q^* поля F_q складається з $q-1$ елементів, то $\forall a \in F_q^*$, $a^{q-1} = 1$.

Адитивна група $GF(p^a)$ поля має фундаментальну особливість: результат складання будь-якого елементу поля із самим з собою p разів дорівнює нулю. Число $p \neq 0$ називається *характеристикою поля,* якщо результат додавання з p одиниць дорівнює нулю, і p – мінімальне число з такою властивістю. Характеристика поля позначається $\mathrm{char}(F)$.

3. Многочлени, дії над ними та алгоритм Евкліда

Многочлен (синоніми: багаточлен, поліном) над полем F – це функція вигляду $f(x) = a_0x^n + a_1x^{n-1} + \cdots + a_n$, де $a_i \in F, i = 0,1 \ldots, n, a_0 \neq 0$. Ціле число $n \geq 0$ називається степенем многочлена і позначається $\deg f$. Числа a_i називаються коефіцієнтами. Областю зміни аргументу є $\mathbb{R}$. Природно, множення і додавання є операціями в полі. Константи (елементи поля) є многочленами нульового степеня.

Аналогічно визначається многочлен над комутативним кільцем.

Якщо $a_0 = 1$, то багаточлен називається звідним (нормованим, унітарним). Будь-який багаточлен над полем можна звести до нормованого, помноживши його на a_0^{-1}, але в кільці це не діє, оскільки не для всіх елементів існують зворотні.

Многочлен $g(x)$ називається дільником многочлена $f(x)$, якщо існує многочлен $h(x)$, такий, що $f(x) = g(x)h(x)$, $f, g, h \in F[x]$.

Спільним дільником двох многочленів називається многочлен, який ділить обидва зазначені многочлени. Дільники многочленів визначаються з точністю до константи.

Найбільшим спільним дільником $d = (f, g)$ двох многочленів $f, g \in F[x]$ над полем F називається багаточлен $d \in F[x]$, такий, що для будь-якого спільного дільника h багаточленів f та g ділить d.

Зазвичай в якості (f, g) вибирається нормований многочлен.

Многочлен ненульового степеня називається незвідним, якщо він ділиться тільки на константи і сам на себе.

Пригадаємо, що числа $1,2,3, \ldots$ називаються натуральними. Число 0, а також числа виду $\pm a$, де a – натуральне число, називаються цілими числами. Відношення двох цілих чисел називається раціональним дробом і є записом результату ділення одного числа на інше. Ділення на нуль не визначене. Безліч раціональних дробів є полем. Позначення області раціональних чисел – $\mathbb{Q}$.

Простим числом називається натуральне число, у якого є два (і тільки два) нерівні натуральні дільники.

Основна теорема арифметики: кожне натуральне число єдиним, з точністю до порядку співмножників, способом представляється у вигляді добутку ступенів простих чисел.

Найбільшим спільним дільником двох цілих чисел a та b називається найбільше ціле число, яке ділить як a, так і b. Позначення: (a, b) або НСД(a, b). Якщо НСД$(a, b) = 1$, то числа називаються *взаємно простими*.

Найменшим спільним кратним натуральних чисел a та b називається найменше натуральне число, НСК (a, b), що ділиться як на a, так і на b.

Очевидно, що НСК$(a, b) = \frac{ab}{(a,b)}$.

Найбільший спільний дільник двох натуральних чисел a, b $(a > b)$ обчислюється за допомогою *алгоритму Евкліда*. У цьому алгоритмі основну роль відіграє операція ділення чисел із залишком, тобто представлення числа у вигляді $a = kb + r, b > r \geq 0$.

Запишемо числа a та b. Знайдемо залишок r_1 від ділення a на b, запишемо його слідом за a, b: a, b, r_1. В отриманому переліку розглянемо останні два числа.

Знайдемо залишок r_2 від ділення першого з них на друге: $b = k_1 r_1 + r_2$, допишемо r_2 в список: a, b, r_1, r_2. Діємо далі аналогічно, поки вперше (на k-му кроці) не виникне ситуація, коли $r_k = 0$. Тоді $(a, b) = r_{k-1}$.

Алгоритм є коректним, тому що елементи списку зменшуються на кожному кроці роботи. До того ж немає необхідності вести весь список повністю, достатньо зберігати два останніх члена.

Схема алгоритму Евкліда може бути застосована також і для многочленів $a(x)$ і $b(x)$ над полем F.

Операція ділення з остачею відповідає представленню у вигляді $a(x) = k(x)b(x) + r(x)$, $\deg b > \deg r \geq 0$. Якщо вперше на k-му кроці виявляється, що $r_k = 0$, процес обчислення залишків від ділення зупиняється і записується $\big(a(x), b(x)\big) = r_{k-1}(x)$.

Питання для самоконтролю

1. Що називають бінарною операцією?
2. Дайте визначення групи, моноїда, поля.
3. Які аксіоми виконуються для групи?
4. Що таке асоціативне кільце?
5. Які поля називають полями Галуа?
6. Дайте визначення многочлена над полем.
7. Опишіть схему алгоритму Евкліда. Чи можна її застосовувати для многочленів над полем?

ТЕМА 4. МАТЕМАТИЧНА ДОВІДКА З МОДУЛЬНОЇ АРИФМЕТИКИ (ЛЕКЦІЯ 4)

1. Лишки за модулем m. Відношення порівняння.
2. Розширений алгоритм Евкліда.

1. Остача за модулем m. Відношення порівняння

Кожне ціле число a можна розділити з остачею на натуральне число m:

$$a = km + r,\ 0 \le r < m.$$

Остача r від ділення числа називається лишком (у даному випадку – лишком числа a за модулем m). Операція, що зіставляє числу його лишок за модулем, називається діленням за модулем.

Два цілі числа a та b є порівнянними за модулем m, якщо їх різниця ділиться на m. Аналогією є тіло, що рухається по колу. Під час руху по колу тіло періодично потрапляє в одну й ту саму точку кола, хоча проходить різний шлях. Довжини шляхів можна порівняти за модулем довжини окружності кола.

Відношення порівняння може записуватись у вигляді: $a \equiv b \bmod m$, $a \equiv b(m)$.

Замість знака порівняння часто використовується знак рівності. До того ж, якщо це не викликає непорозумінь за контекстом задачі, вказування модуля може бути необов'язковим.

Відповідно до цього визначення, числа, що мають однакові лишки від ділення на m, можна порівняти за модулем m.

Якщо не визначено інше, то стандартні значення лишків за модулем m належать множині $0, 1, \dots m-1$ (система найменших невід'ємних лишків).

Лишок за модулем двох доданків дорівнює додаванню лишків, звідному, якщо необхідно, ще раз за тим самим модулем. Аналогічну властивість має лишок добутку. Отже, знаходження лишків великих чисел можна звести до роботи з числами, що за абсолютною величиною не перевищують квадрата необхідного модуля.

Відношення порівняння дає змогу розбити всі множини цілих чисел на класи. Елементи одного й того самого класу мають однакові лишки за модулем. Очевидно, що класи не перетинаються, тому кожному класу відповідає один найменший додатний або нульовий лишок, і навпаки. Якщо ми виділимо деякі два класи A та B, то лишок суми або добутку за модулем m елементів $a \in A$ та $b \in B$ не залежить від вибору цих елементів, а залежить лише від вибору класів. Тому ми можемо побудувати *кільце,* елементами якого є класи, а операції виконуються через дії над найменшими додатними або нульовими лишками, які належать відповідним класам.

Наприклад, якщо $c \equiv a + b(\bmod m)$, то клас, що містить c, є множиною виду $C = \{c, c+m, c+2m \dots\}$. Оскільки множина Z цілих чисел є кільцем, то наша відповідність між класами і лишками $c = \phi(C)$ є гомоморфізмом кілець. Образ

цього гомоморфізму (кільце лишків за модулем m) називається фактор-кільцем кільця цілих чисел за модулем m і зазвичай позначається Z/mZ. Нулем у кільці лишків у кільці Z/mZ є клас $0 = \{0, m, 2m \ldots\}$.

2. Розширений алгоритм Евкліда

Під час зашифровування інформації використовуються взаємно однозначні перетворення. Зокрема, під час використання операції зведення за модулем необхідне знаходження зворотних (обернених) елементів. Для знаходження обернених елементів у кільці Z/mZ можна використати розширений алгоритм Евкліда.

Цей алгоритм призначений для пошуку цілочисельного розв'язку x, y, d рівняння $xa + yb = d, a > b, d = (a, b)$, де a і b також цілі. Загальний випадок зводиться до вирішення подібних рівнянь за умови додатних a і b.

Розглянемо схему розширеного алгоритму Евкліда на прикладі чисел 15 і 25.

Оскільки має виконуватися нерівність $a > b$, то приймемо такі позначення: $r_0 = 25,\ r_1 = 15$.

Випишемо послідовність рядків:

$a_0 = a = 25$	$a_0 = ax_0 + by_0, x_0 = 1, y_0 = 0$
$a_1 = b = 15$	$a_1 = ax_1 + by_1, x_1 = 0, y_1 = 1$
$a_2 = a_0 - a_1q_1, q_1 = 1, a_2 = 10$	$a_2 = ax_2 + by_2, x_2 = 1, y_2 = -1$
$a_3 = a_1 - a_2q_2, q_2 = 1, a_3 = 5$	$a_3 = ax_3 + by_3, x_3 = -1, y_3 = 2$
$a_4 = a_2 - a_3q_3, q_3 = 2, a_4 = 0$	

Оскільки ми отримали на певному кроці $a_i = 0$, то виписуємо значення з попереднього кроку, у нашому випадку $a_3 = 5$. Це і буде НСД(25,15).

Лишки за модулем часто можна побачити у математичному описі шифрів гамування, комбінованих шифрів, про які буде йтися у наступних лекціях.

Питання для самоконтролю

1. Що називають лишком числа a за модулем m?
2. Сформулюйте поняття класу елементів.
3. Поясніть на прикладі розширений алгоритм Евкліда.

ТЕМА 5. КЛАСИФІКАЦІЯ ШИФРІВ (ЛЕКЦІЯ 5)

1. Основні елементи криптосистеми, модель криптосистеми Шеннона.
2. Вимоги до криптосистем.
3. Класифікація шифрів.

1. Основні елементи криптосистеми, модель криптосистеми Шеннона

Модель системи секретного зв'язку К. Шеннона була запропонована в його роботі «Теорія зв'язку в секретних системах», опублікованій у 1949 р.

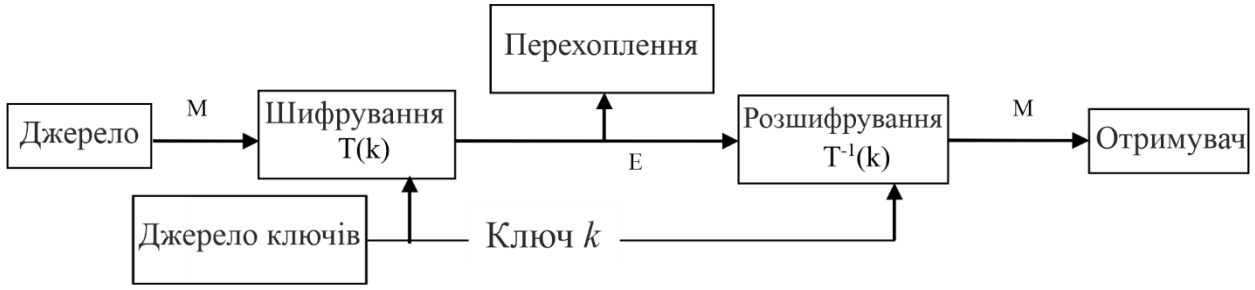

Рис 1. Модель Шеннона

За Шенноном, криптографічна система є параметричним сімейством оборотних відображень множини можливих повідомлень у множину криптограм. Кожне відображення є шифруванням. Вибір відображення проводиться за допомогою випадкового секретного параметра – ключа.

Ключ також дає змогу обрати зворотне перетворення (розшифрування). Ключ повинен бути доступний відправнику і одержувачу в необхідний момент.

Передбачається, що ключі розсилаються безпечним методом, що не обговорюваним у межах моделі.

Така система секретного зв'язку має назву *симетричної криптосистеми* (симетрія користувачів щодо знання секрету).

Модель противника у системі є такою. Доступним для перехоплення є шифротекст, що передбачається відомим алгоритмом шифрування і його параметрами, за винятком ключа (принцип відкритості, загальнодоступності системи).

Практична стійкість шифроперетворення полягає у трудомісткості отримання відкритого тексту аналітично, без знання ключа, за допомогою найкращих з наявних сьогодні алгоритмів.

За допомогою шифрування за описаною моделлю вирішується завдання забезпечення конфіденційності даних. Під час цього по каналу зв'язку передається не сама захищається інформація, а результат її перетворення, і для супротивника виникає складне завдання розкриття шифру.

Розкриття або злам шифру – це процес отримання інформації з шифрованого повідомлення без знання застосованого для шифрування ключа.

Здатність шифру протистояти атакам на нього називають *криптографічною стійкістю* (криптостійкістю) шифру. Під атакою на шифр розуміють спробу зламу цього шифру.

Поняття стійкості шифру є основним для криптографії. Хоча якісно зрозуміти його доволі легко, але отримання строгих доказових оцінок стійкості для кожного конкретного шифру – проблема невирішена. Тому стійкість конкретного шифру оцінюється тільки шляхом різних спроб його зламу і залежить від кваліфікації криптоаналітиків, які атакують шифр. Таку процедуру іноді називають перевіркою стійкості. Вона робиться за припущення, що супротивник знає алгоритм перетворення, але не знає ключа. Супротивник також може знати деякі характеристики відкритих текстів, наприклад, загальну тематику повідомлень, їх стиль, деякі стандарти, формати та інше.

Однак крім перехоплення шифру, супротивник може намагатися отримати інформацію багатьма іншими способами. Найбільш відомим із таких способів є агентурний, коли противник будь-яким шляхом схиляє до співпраці одного із законних користувачів і за допомогою цього агента отримує доступ до інформації, яку захищають. У такій ситуації криптографія безсила.

Супротивник може спробувати не одержати, а знищити або модифікувати інформацію в процесі її передачі. Це зовсім інший тип загроз для інформації, що відрізняється від перехоплення і зламу шифру. Для захисту від таких загроз розробляються специфічні методи.

Не існує єдиного шифру, придатного на всі випадки. Вибір способу шифрування залежить:

– від виду інформації, яку захищають (документальна, телефонна, телевізійна, комп'ютерна та ін.);

– від обсягу і необхідної швидкості передачі шифрованої інформації;

– від цінності інформації, яку захищають (наприклад, державні, військові та ін. таємниці повинні зберігатися десятиліттями, а біржова інформація може вже через кілька годин бути розголошена);

– від можливостей власників секретної інформації, а також від можливостей супротивника (одна справа – протистояти одинаку, а інша справа – потужній державній структурі).

Процес криптографічного захисту даних може здійснюватися як програмно, апаратно, так і іншими способами. Апаратна реалізація відрізняється істотно більшою вартістю, однак їй властиві і переваги: висока продуктивність, простота, захищеність та ін. Програмна реалізація більш практична, допускає відому гнучкість у використанні.

2. Вимоги до криптосистем

Для сучасних криптографічних систем можна сформулювати такі вимоги:

– складність і трудомісткість процедур шифрування і дешифрування повинні визначатися залежно від необхідного рівня захисту інформації (необхідно забезпечити надійний захист інформації);

– часові і вартісні витрати на захист інформації повинні бути прийнятними за заданого рівня її секретності (витрати на захист не повинні бути надмірними);

– процедури шифрування і дешифрування не повинні залежати від довжини повідомлення;

– кількість усіх можливих ключів шифру має бути такою, щоб їх повний перебір за допомогою сучасних інформаційних технологій (зокрема і розподілених обчислень) був неможливий за прийнятний для супротивника час;

– будь-який ключ із безлічі можливих має забезпечувати надійний захист інформації;

– незначна зміна ключа повинна призводити до істотної зміни вигляду зашифрованого повідомлення;

– надлишковість повідомлень, яку вносять у процесі шифрування, повинна бути якомога меншою (хорошим вважається результат, коли довжина шифрограми не перевищує довжину вихідного тексту);

– зашифроване повідомлення повинно зчитуватись тільки за наявності ключа.

3. Класифікація шифрів

Існує кілька варіантів класифікацій шифрів.

За кількістю та послідовністю використання ключів розрізняють одноключові (симетричні), двоключові (асиметричні з відкритим ключем), багатоключові.

За особливостями процедур перетворення вхідного повідомлення існують:

– шифри перестановки (одноразової та багаторазової);

– шифри заміни або підстановки, які поділяються на однозначну заміну: регулярну (поліграмну або асиметричну детерміновану), або нерегулярну, і багатозначну заміну: омофонічні та поліалфавітні (адитивні, квантові, ймовірнісні асиметричні) шифри;

– комбіновані шифри.

За кількістю символів вхідного повідомлення, які одночасно застосовуються в алгоритмі шифрування, розрізняють блокові та потокові шифри.

За стійкістю існують шифри нестійкі, або обмеженого використання, та необмеженого використання, практично та абсолютно стійкі.

Питання для самоконтролю:

1. Опишіть модель системи секретного зв’язку К. Шеннона.
2. Сформулюйте визначення симетричної криптосистеми.
3. Від чого може залежати вибір способу шифрування?
4. Які вимоги висувають для криптосистем?
5. Наведіть приклади класифікації шифрів за особливостями процедур перетворення вхідного повідомлення та за кількістю символів вхідного повідомлення.

ТЕМА 6. КЛАСИЧНІ СИМЕТРИЧНІ КРИПТОСИСТЕМИ (ЛЕКЦІЯ 6)

1. Шифри перестановки.
2. Стеганографічні шифри.
3. Шифри заміни.
4. Шифри гамування.

1. Шифри перестановки

Під час шифрування перестановкою символи шифротексту переставляються за певним правилом у межах блоку цього тексту. Шифри перестановки є найбільш простимі і, ймовірно, найдавнішими шифрами.

Першим найпростішим криптографічним пристроєм, що реалізує шифр перестановки, є древній шифр «сцитала» – на циліндр або жезл намотувалась стрічка, на ній записувався текст. Для дешифрування такого шифру потрібно не тільки знати правило шифрування, але і володіти ключем у вигляді циліндра певного діаметра. Ідея цього шифру отримала розвиток у методах шифрування за допомогою спеціальних таблиць.

З початку епохи Відродження у шифрах перестановки застосовуються шифрувальні таблиці, які, по суті, задають правила перестановки літер у повідомленні. Як ключ у шифруючих таблицях використовуються:

– розмір таблиці;
– слово або фраза, що задають перестановку;
– особливості структури таблиці.

Одним з найбільш примітивних табличних шифрів перестановки є проста перестановка, для якої ключем слугує розмір таблиці. Оригінальний текст повідомлення записується в таблицю по стовпчиках. Для отримання шифротексту вміст таблиці зчитують по рядках. Під час дешифрування ці дії виконують у зворотному порядку. Ключем шифру є розмір таблиці – кількість рядків і стовпців. Якщо текст не може бути розміщений у таблиці повністю, його розбивають на блоки довжиною відповідно до розмірів таблиці, додаючи за необхідності певну кількість символів. Оскільки кількість варіантів різних ключів у таких алгоритмах є відносно невеликою, стійкість такого алгоритму шифрування є невисокою.

Дещо більшу стійкість має метод, що полягає у одиночній перестановці за ключем. Цей метод відрізняється від попереднього тим, що стовпчики таблиці переставляються за ключовим словом, фразою або набором чисел довжиною в рядок таблиці.

Розглянемо приклад.

1. Нам потрібно зашифрувати повідомлення «Інформаційна безпека дає захист» методом табличної перестановки з ключовим словом.

2. Обираємо ключове слово (ключ) «Травень» та нумеруємо стовпчики у порядку абетки. Основна умова добору ключового слова – відсутність літер, що повторюються.

3. Записуємо відкрите повідомлення у рядки без пробілів.

Т	*Р*	*А*	*В*	*Е*	*Н*	*Ь*
6	*5*	*1*	*2*	*3*	*4*	*7*
І	**Н**	**Ф**	**О**	**Р**	**М**	**А**
Ц	**І**	**Й**	**Н**	**А**	**Б**	**Е**
З	**П**	**Е**	**К**	**А**	**Д**	**А**
Є	**З**	**А**	**Х**	**И**	**С**	**Т**

4. Виписуємо у рядок літери зі стовпчиків за порядком абетки та отримуємо криптограму:

ФЙЕАОНКХРААИМБДСНІПЗІЦЗЄАЕАТ

5. Під час розшифровки підставляємо ключове слово, нумеруємо його літери, перераховуємо кількість знаків у шифровці, розділяємо на довжину ключа та виписуємо у таблицю.

Ключ можна задати просто цифрами, але його важче запам’ятати, ніж слово. Тому історично склалося так, що ключ прийнято називати «ключовим словом».

Для забезпечення додаткової стійкості можна повторно зашифрувати повідомлення, яке вже пройшло шифрування. Такий метод шифрування називається подвійною перестановкою.

Кількість варіантів подвійної перестановки швидко зростає за збільшення розміру таблиці. Однак і подвійна перестановка не відрізняється високою стійкістю і порівняно просто «зламуються» за будь-якого розміру таблиці шифрування.

До шифрів перестановки належить і шифр Кардано. У середині XVI ст. італійським математиком Кардано була висунута ідея використання частини самого переданого відкритого тексту в якості ключа, і новий спосіб шифрування отримав назву «решітка Кардано». Для її виготовлення брався квадрат картону, в якому за спеціальним правилом прорізали «вікна». Під час шифрування решітка накладалась на аркуш паперу, і відкритий текст вписувався у вікна. Потім решітка повертalася на 90 градусів, і знову вписувався текст, всього 4 рази.

Головна вимога до «решітки Кардано» – під час всіх поворотів картонки вікна не повинні були потрапляти на одне й те саме місце паперу. Порожні місця заповнювалися довільними літерами, а потім шифрограма виписувалася по рядках. Решітка Кардано давала змогу здійснювати перестановку букв доволі швидко.

Висновок: шифрування перестановкою полягає в тому, що символи вхідного тексту переставляються за певним правилом у межах деякого блоку цього тексту.

За достатньої довжини блоку, в межах якого здійснюється перестановка, і складного порядку (маршруту) перестановки можна досягти прийнятної для простих практичних застосувань стійкості шифру.

2. Стеганографічні шифри

Ідея шифру-решітки, запропонованого Кардано, лежить і в основі знаменитого шифру Рішельє, в якому зашифрований текст зовні мав вигляд осмисленого повідомлення, що не має жодного стосунку до переданої інформації. Зі щільного матеріалу вирізали прямокутник розміром, наприклад, 7×10; у ньому проробллялися вікна (на рисунку-прикладі вони заштриховані).

I	.	L	O	V	E	.	Y	O	U
I	.	H	A	V	E	.	Y	O	U
D	E	E	P	.	U	N	D	E	R
M	Y	.	S	K	I	N	.	M	Y
L	O	V	E	.	L	A	S	T	S
F	O	R	E	V	E	R	.	I	N
H	Y	P	E	R	S	P	A	C	E

Рис. 2. Приклад шифру Рішельє

Секретний текст вписувався в ці вікна, потім решітка знімалася, і комірки, що залишились, заповнювалися так, щоб виходило повідомлення, яке має зовсім інший сенс. Сувору команду англійською мовою: «YOU KILL AT ONCE» за допомогою таких перетворень можна заховати в «невинний» текст любовного змісту: «I LOVE YOU. I HAVE YOU. DEEP UNDER MY SKIN. MY LOVE. LASTS LEVER IN HYPERSPACE ».

Такі шифри належать вже до стеганографічних методів захисту інформації, які передбачають приховування самого факту передачі інформації. До них належать, але безпосередньо не стосуються криптографії, методи тайнопису (невидимі чорнила, які проявляються внаслідок спеціальної обробки, мікроточки в тексті книги та ін., приховування інформації на зображеннях або інших графічних елементах).

Стеганографічні методи у вигляді замаскованих закладок використовуються для захисту програмних продуктів від несанкціонованого копіювання.

3. Шифри заміни

У шифрі простої заміни кожен символ вихідного тексту замінюється символами того самого алфавіту однаково протягом усього тексту. Шифри простої заміни називають також шифрами одноалфавітної (одноабеткової) підстановки.

Одним з перших шифрів простої заміни вважається так званий квадрат Полібія. За два століття до нашої ери грецький полководець та історик Полібій винайшов з метою шифрування квадратну таблицю розміром 5×5, заповнену літерами алфавіту у випадковому порядку.

Під час шифрування в цьому квадраті знаходили наступну літеру відкритого тексту і записували в шифртекст літеру, розташовану нижче неї у тому самому стовпці. Якщо літера тексту знаходилася у нижньому рядку таблиці, то для шифртексту брали верхню літеру з того самого стовпця. Концепція квадрата Полібія виявилася плідною і далі знайшла застосування в криптосистемах.

Шифр Цезаря є окремим випадком шифру простої заміни (одноалфавітної підстановки). Під час шифрування вихідного тексту кожна літера замінялася на іншу літеру того самого алфавіту шляхом зміщення за алфавітом від вихідної букви на літеру, посунуту на K позицій у абетці. Після досягнення кінця алфавіту виконувався циклічний перехід до його початку. Цезар використовував шифр заміни за зміщення $K = 3$. Наприклад, послання Цезаря VENI VIDI VICI виглядало б е зашифрованому вигляді так: YHQL YLGL YLFL.

Водночас такий шифр заміни можна задати таблицею підстановок, що містить відповідні пари букв відкритого тексту і шифротексту.

Перевагою системи Цезаря є простота шифрування і дешифрування, що обумовлює її застосування навіть у складних сучасних шифрах у якості складового елемента.

Недоліками шифру Цезаря є:

– кількість можливих ключів K є малим (не більшим за кількість літер алфавіту);

– зберігається алфавітний порядок у послідовності заміни літер;

– за зміни значення K змінюються тільки початкові позиції такої послідовності, і достатньо розшифрувати заміну однієї літери, щоб визначити всі інші заміни;

– шифр Цезаря легко розкривається на основі аналізу частот появи літер у шифртексті, оскільки підстановки, що виконуються за шифром Цезаря, не маскують частоти появи різних букв вихідного відкритого тексту.

4. Шифри гамування

Подальший розвиток шифрування розвивається за допомогою теорії множин – вводиться застосування поняття групи, підгрупи, циклічної групи та підгрупи. Все це відбувається у Європі в період епохи Відродження. Значний внесок у розвиток криптоаналітики в цей час зробили видатні математики Джироламо Кардано, Франсуа Вієт та Джованні Соро. Спочатку криптологи не здогадувалися, що криптоаналітики оволоділи частотним аналізом криптограм, і продовжували використовувати

одноабетковий шифр замін (зрозуміло, що криптоаналітики не афішували свої методи роботи). Але поступово вони зрозуміли нестійкість такого шифру, і почалася боротьба за підвищення стійкості.

Перша робота в цьому напрямі виникла приблизно в 1460 р. у флорентійського вченого-енциклопедиста Леона Батисто Альберті. Він здогадався, що для підвищення стійкості шифру треба ввести багатоабетковий шифр. Вчений запропонував три шифроабетки, якими відповідно до ключа та алгоритму шифрування складалася шифрограма. Але він не довів свою ідею до цілісної системи.

Одним із варіантів підвищення стійкості шифру була ідея виконання заміни шляхом накладання на відкрите повідомлення *гами* – деякої псевдовипадкової послідовності, яка формується на основі секретного ключа, у найпростішому варіанті – це додавання ключового слова за заздалегідь обраним законом. Взагалі накладання гами може бути реалізоване як будь-яка арифметична операція.

Вперше метод такого шифрування був використаний монахом з Вюрцбурга Трітемієм (шифр Трітемія). Такі шифри почали називати гаманізованими – від слова «гама» (інколи рондомізованими – від «рондо», тобто «повторення»). Також використовуються терміни «шифри гамування» та «шифри рондомування». Далі теорію розвинув італійський вчений Джованні Порту. Ці ідеї розвивалися у XVI ст.

Остаточно ідеї Альберті, Трітемія та Порту були доведені до закінченої теорії та практичного створення надзвичайно досконалого шифру французьким дипломатом Блезу де Віженером. Саме на його честь шифр отримав назву «квадрат Віженера».

Отже, були розроблені класичні методи створення шифрів. Ці методи з деякими ускладненнями використовуються і до теперішнього часу.

Водночас варто зазначити, що у багатьох випадках сама теорія чисел складалася, як то кажуть, «за перебігом» розробки нових методів шифрування та дешифрування тими математиками, які займалися питаннями криптографії та криптоаналізу.

Принцип шифру Віженера полягає у тому, що для латинської абетки використовується 26 шифроабеток, які створюються зсувом вліво кожної наступної шифроабетки відносно попередньої. Так утворюється квадрат з відкритої абетки та 26 таємних абеток. Цей винахід Віженер оприлюднив у 1586 р. у своєму «Трактаті про шифри». Але, як не дивно, криптографи того часу не скористалися шифром Віженера, оскільки його вважали занадто складним для шифрування. Лише з винаходом телеграфічного зв'язку шифр Віженера почав використовуватися всіма криптографами у всіх державах.

Навіть більше, цей шифр вважався таким, що не піддається розкритю, до 1854 р., коли видатному англійському криптоаналітику Чарльзу Беббіджу не вдалося знайти механізм його зламу. Чарльз Беббідж вважається «батьком» сучасного криптоаналізу, оскільки він встановив надзвичайно важливу залежність: для роз-

шифрування шифру необхідно знайти довжину ключа, а це можна зробити, встановивши періодичність у зашифрованому повідомленні. Після виявлення довжини ключа аналіз ведеться як для звичайного одноабеткового шифру, але для кожної абетки окремо.

Загалом шифр на основі гамування, на відміну від шифру на основі таблиці заміни, є стійким до статистичного аналізу. Однак такий шифр є вразливим до атак на генератор гами, зокрема до атаки нав'язування зловмисником відкритих повідомлень.

Також до шифрів гамування можна віднести шифр Вернама (англійською One-time pad – шифр одноразових блокнотів). Шифр Вернама – це система симетричного шифрування, винайдена у 1917 р. співробітниками AT&T Мейджором Джозефом Моборном і Гільбертом Вернамом. Шифр Вернама є системою шифрування, для якої доведена абсолютна криптографічна стійкість.

Для утворення шифротексту відкритий текст об'єднується операцією «XOR» з ключем (ключ генерується такої самої довжини, що і повідомлення, використовується один раз, тому отримав назву одноразового блокнота або шифроблокнота). Ключ повинен мати три критично важливі властивості:

1) бути справді випадковим;

2) збігатися за розміром із заданим відкритим текстом;

3) застосовуватися тільки один раз.

Вернамом було створено телеграфний апарат, який виконував цю операцію автоматично – треба було тільки подати на нього стрічку з ключем. Хоч і не був шифрувальником, Вернам правильно помітив важливу властивість свого шифру: кожна стрічка повинна використовуватися тільки один раз і після цього знищуватися. Це важко застосувати на практиці – тому апарат був перероблений на кілька закільцьованих стрічок із взаємно простими періодами.

Питання для самоконтролю

1. У чому полягає основна ідея шифрів перестановки? Наведіть приклади шифрів.
2. Що може виконувати роль ключа у шифрах із шифрувальною таблицею?
3. Наведіть приклади стеганографічного захисту інформації.
4. У чому полягає основна ідея шифрів заміни? Наведіть приклади шифрів.
5. В чому полягає основна ідея шифрів гамування? Наведіть приклади шифрів.

ТЕМА 7. МАТЕМАТИЧНИЙ АНАЛІЗ ПРОСТИХ ШИФРІВ (ЛЕКЦІЯ 7)

1. Алфавіт криптосистеми.
2. Система шифрування Цезаря.
3. Підстановки, шифр Віженера.
4. Система Хілла.
5. Одноразові блокноти, шифр Вернама.

1. Алфавіт криптосистеми

Під час криптографічних перетворень як звичайний текст, так і шифротекст утворюються з букв, включених у скінченний набір символів, які називаються алфавітом або абеткою.

Джерело інформації генерує повідомлення (звичайний текст) із деяких скінченних алфавітів X: $\{x_1, x_2, \dots x_m\}$ з основою алфавіту m. Шифротекст або шифрограма утворюються загалом з алфавіту Y: $\{y_1, y_2, \dots, y_n\}$, який може не збігатися з алфавітом X.

Під час виконання криптографічних перетворень зручно замінити букви алфавіту цілими числами. Заміна букв традиційного алфавіту цифрами дає змогу математично чітко сформулювати основні поняття та прийоми криптографічних перетворень і спростити здійснення необхідних алгебраїчних маніпуляцій. Водночас ми продовжимо використовувати алфавіт природної мови в якості ілюстрацій.

Отже, криптографія використовує алфавіт, який містить m букв природної або технічної мови у вигляді цифр:

$$Z_m = \{0,1,2.3, \dots . m-1\}.$$

Приклади алфавітів:

– алфавіт Z_{33} – 33 букви українського алфавіту;
– алфавіт Z_{26} – 26 букв англійського алфавіту;
– алфавіт Z_{256} – символи, що входять до стандартних кодів ASCII і KOI-8;
– двійковий алфавіт – $Z_2 = \{0,1\}$;
– 8-ковий алфавіт або 16-ковий алфавіт.

Об'єднання n букв з алфавіту Z називається n-грамою. Об'єднавши n букв з алфавіту Z_m, отримуємо алфавіт $Z_{m,n}$, що містить множину n-грам з алфавіту Z_m. Наприклад, англійський алфавіт $Z_{26} = \{ ABCDEFGH \dots WXYZ\}$ з об'ємом $m =$ 26 букв дає змогу генерувати алфавіт з $26^2 = 676$ біграм, або $26^3 = 17\,576$ триграм.

Отже, криптографічне перетворення E – це узагальнено:

$$E = \{E^{(n)}: 1 < n < \infty\}, \text{ де } E^{(n)}: Z_{m,n} \to Z_{m,n}.$$

Перетворення $E^{(n)}$ визначає, як кожна n-грама відкритого тексту x замінюється n-грамою шифротексту y, тобто:

$$y = E^n(x), x, y \in Z_{m,n}.$$

У даному випадку врахована вимога взаємної унікальності перетворення E^n на множині $Z_{m,n}$.

Криптографічну систему можна інтерпретувати як набір криптографічних перетворень $E = \{E_k : k \in K\}$, що визначається параметром k – ключем. Набір ключових значень утворює ключовий простір K.

2. Система шифрування Цезаря

Розглядаючи алфавіт криптосистеми як множину цілих чисел Z_m, ми можемо записати функцію шифрування E_k для $k = 3$ в простому шифрі Цезаря у вигляді:

$$E_k : x \to (x + 3) \bmod m, \forall x \in Z_m,$$

де x – числовий код літери відкритого тексту; $x + 3$ – числовий код відповідної літери шифротексту. Для операції додавання за модулем див. лекцію 3.

На відміну від шифру Цезаря, система шифрування Цезаря формується набором одноалфавітних замін, визначених функціями шифруванняE_k для різних значень ключа k, де k приймає значення від 0 до m, m є основою алфавіту.

Відповідно до цієї системи буква $x \in Z_m$ відкритого тексту перетворюється на букву $y \in Z_m$ шифротексту за таким правилом:

$$E_k : y = (x + k)\ mod\ m,$$

де x – числовий код літери відкритого тексту; y – числовий код відповідної літери шифротексту.

Концепція, закладена в системі шифрування Цезаря, виявилася дуже вдалою, про що свідчать її численні модифікації.

Система шифрування Цезаря використовувала тільки адитивні властивості множини цілих чисел Z_m, тобто розглядалася як група з операцією додавання. З огляду на множину цілих чисел Z_m з двома операціями додавання і множення за модулем m, яка є кільцем, можна отримати систему заміщення, яка називається *афінною системою шифрування Цезаря*.

Визначимо в такій системі перетворення $E_{a,b} : Z_m \to Z_m$:

$$E_{a,b}(x) = ax + b\ mod\ m,$$

де ключ $k = (a, b)$ – пара цілих чисел, які задовольняють умови $0 \le a, b < m$, та $\text{НСД}(a, m) = 1$.

Під час цього перетворення літера, яка відповідає цифрі x у чистому тексті, замінюється літерою шифротексту, що відповідає числовому значенню:

$$y = (ax + b)\ mod\ m.$$

Варто зазначити, що перетворення $E_{a,b}(x)$ є взаємно однозначним відображенням на множині Z_m тільки у випадку, якщо $\text{НСД}(a, m) = 1$, тобто a і m мають бути взаємно простими числами.

Якщо така вимога не виконується, то можлива ситуація, коли однією літерою шифротексту можуть бути зашифровані дві різні літери (це приведе до двозначності розшифровки), а в шифротексті будуть відсутні деякі літери, оскільки за такого перетворення в них не відображатимуться деякі літери відкритого тексту.

Така система шифрування належить до класу шифрів на основі аналітичних перетворень зашифрованих даних. Перевагою є зручне управління ключами – ключі шифрування і дешифрування представлені в компактному вигляді, у вигляді пари чисел (a, b). Порівняно з простою системою шифрування Цезаря, кількість можливих ключів є набагато більшою, а алфавітний порядок слів не зберігається під час шифрування.

Афінна система використовувалася на практиці кілька століть тому, і сьогодні її застосування обмежується переважно ілюстраціями основних криптологічних правил.

3. Підстановки, шифр Віженера

Під підстановкою множини M ми розуміємо взаємно-однозначне відображення цієї множини на саму себе:

$$p\colon M \to M,$$

тобто зіставлення p кожному елементу m з M деякого образу $p(m)$, причому кожен елемент з M є образом тільки одного елементу. Елемент $p(m)$ позначають також через pm.

У разі нескінченної множини M підстановки називають також перетвореннями.

Якщо множина M має скінчену кількість елементів, і її елементи пронумеровані числами $0, 1, 2, \ldots, m-1$, то кожну підстановку можна повністю описати схемою, в якій під кожним номером k вказується номер $p(k)$ елементу, що є прообразом елементу з номером k.

Під добутком $\pi_1\pi_2$ двох підстановок π_1 і π_2 розуміється підстановка, яку ми отримуємо, здійснюючи спочатку підстановку π_2, а потім застосовуючи до результату підстановку π_1:

$$\pi_1\pi_2(m) = \pi_1\big(\pi_2(m)\big).$$

Для підстановок виконується закон асоціативності:

$$(p_1p_2)p_3 = p_1(p_2p_3).$$

Тотожною або одиничною підстановкою є таке відображення π_0, за якого кожен об’єкт переводиться сам у себе:

$$p_0(m) = m.$$

Тотожна підстановка має властивість одиничного елементу групи:

$$\forall p_0\colon pp_0 = p.$$

Зворотною підстановкою p^{-1} до підстановки p є така підстановка, яка переводить $p(m) \to m$:

$$p^{-1}p(m) = p_0(m) = m, \text{ а також } p^{-1}p = p_0.$$

Сукупність підстановок довільної множини M утворює групу, оскільки для неї є справедливими аксіоми групи 1–4. Скінченна множина M з π елементів є також симетричною групою.

Таке відображення (підстановка) використовується для математичного опису шифру заміни – як простої, так і складної (багатоалфавітної). Під час шифрування заміною символи відкритого тексту замінюються символами того самого або іншого алфавіту за заздалегідь встановленими правилами.

Виконаємо математичний аналіз шифру простої заміни. Ключем шифру є підстановка π в алфавіті Z_m, що визначає правило заміни під час шифрування літери x відкритого тексту (x – ціле число) на літеру $\pi(x)$ шифртексту:

$$\pi: x \to \pi(x).$$

Підстановка π є взаємно однозначним відображенням з Z_m на Z_m.

Множина π всіх підстановок на Z_m є симетричною групою і має такі властивості: замкненість (добуток підстановок $\pi_1\pi_2$ є підстановкою), асоціативність, існування одиничного і зворотного елементів.

Для багатоалфавітної підстановки E_k ключ підстановки K являє собою послідовність підстановок з множини П:

$$K = \{(\pi_0, \pi_1, \dots, \pi_{n-1}), \pi \in \text{П}\},\ 1 \le n < \infty.$$

Ця підстановка шифрує: n-граму $(x_0, x_1, x_2, \dots, x_{n-1})$ відкритого тексту в n-граму $(y_0, y_1, y_2, \dots, y_{n-1})$ шифртексту відповідно до формули:

$$y_i = \pi_i(x_i), 0 \le i < n, n = 1,2,3,\dots,$$

при $n \to \infty$ ми наближаємося до теоретично стійкої одноразової системи шифрування.

Відзначимо характерні особливості підстановки E_k:

– відкритий текст шифрується літера за літерою;

– i-та літера шифротексту є функцією тільки i-тої компоненти π_i ключа k і i-тої літери x_i відкритого тексту.

Зауваження: криптографічне перетворення E_k буде одноалфавітною підстановкою, якщо значення π_i однакові для кожного $i = 0,1,2,\dots$; інакше воно буде багатоалфавітною підстановкою.

Опишемо тепер шифр Віженера. Нехай ключова послідовність системи Віженера має довжину r, тоді ключ r-алфавітної підстановки, який є рядком літер або цифр, можна представити у вигляді послідовності підстановок:

$$\pi = (\pi_0, \pi_1, \dots, \pi_{r-1}).$$

Функція шифрування Віженера E_π: $x \to y$ перетворює відкритий текст $x = (x_0, x_1, x_2, \dots, x_{n-1})$ у шифротекст $y = (y_0, y_1, y_2, \dots, y_{n-1})$ за таким правилом:

$$y = (y_0, y_1, y_2, \dots, y_{n-1}) = (\pi_0, \pi_1, \dots, \pi_{n-1}), \text{ де } \pi_i = \pi_{(i \bmod r)}.$$

4. Система Хілла

Криптосистема Хілла і її окремий випадок – шифр Плейфера (Playfair) – також належить до класу шифрів, заснованих на аналітичних перетвореннях шифрованих даних, як і афінна система шифрування. Вони засновані на підстановці не окремих символів, а n-грам (шифр Хілла) або 2-грам (шифр Плейфера). За більш високої криптостійкості вони значно складніші для реалізації і вимагають доволі великої кількості ключової інформації.

Алгебраїчний метод, що узагальнює афінну підстановку Цезаря для шифрування n-грам, був сформульований Лестером С. Хіллом.

Шифрування ведеться шляхом виконання множення вектора на матрицю. Матриця є ключем шифрування. Відкритий текст розбивається на n-грами – блоки довжиною n, що дорівнює розмірності матриці, і кожна n-грама розглядається як вектор.

Множина цілих чисел Z_m, для якої визначені операції додавання і множення за модулем m, є кільцем. Якщо модуль m є простим числом, то існує обернена величина будь-якого ненульового елемента з Z_m.

Множина всіх n-грам $x = (x_0, x_1, x_2, \dots, x_{n-1})$ з компонентами з кільця Z_m утворює векторний простір $Z_{m,n}$ над кільцем Z_m. Кожна n-грама x є вектором. У векторному просторі $Z_{m,n}$ для векторів x визначені операції додавання і віднімання за модулем n, а також скалярне множення вектора на елемент $\boldsymbol{x}$ кільця Z_m. Додавання і скалярне множення є операціями, що задовольняють комутативний, асоціативний і дистрибутивний закони.

Базисом для векторного простору $Z_{m,n}$ є набір векторів з $\{\boldsymbol{x}^i: 0 \le i < L\}$, які є лінійно незалежними і породжують $Z_{m,n}$. Тобто будь-який вектор x є лінійною комбінацією векторів базису:

$$\boldsymbol{x} = t_0 \cdot \boldsymbol{x}^0 + t_1 \cdot \boldsymbol{x}^1 + \cdots + t_{L-1} \cdot x^{(L-1)} \bmod m.$$

Ключова матриця $\boldsymbol{T}$ є квадратною матрицею розмірності n та вигляду $\boldsymbol{T} = \{\gamma_{i,j}\}, i, j = 0,1, \dots, n-1$. Вона задає відображення, що є лінійним перетворенням:

$$\boldsymbol{T}: Z_{m,n} \to Z_{m,n}, \quad T: \boldsymbol{x} \to \boldsymbol{y}; \quad \boldsymbol{y} = \boldsymbol{T}\boldsymbol{x},$$

де $y_i = \sum_{j=0}^{n-1} t_{ij} x_j \bmod m$.

Це перетворення задовольняє умову лінійності:

$$\boldsymbol{T}(t \cdot \boldsymbol{x} + s \cdot \boldsymbol{y}) = t \cdot \boldsymbol{T}\boldsymbol{x} + s \cdot \boldsymbol{T}\boldsymbol{y} \bmod m$$

для всіх $s, t \in Z_m$ і $\boldsymbol{x}, \boldsymbol{y} \in Z_{m,n}$.

Для дешифрування шифротексту необхідно виконати зворотне перетворення:

$$\boldsymbol{x} = \boldsymbol{T}^{-1}\boldsymbol{y}.$$

Для того, щоб лінійне перетворення $\boldsymbol{T}$, задане своєю матрицею, могло бути криптографічним перетворенням, необхідно, щоб воно було оборотним (або невиродженим), тобто має існувати зворотне лінійне перетворення:

$$\boldsymbol{T}^{-1}: Z_{m,n} \to Z_{m,n}, \quad \boldsymbol{T}\boldsymbol{T}^{-1} = \boldsymbol{T}^{-1}\boldsymbol{T} = \boldsymbol{I},$$

де $\boldsymbol{I}$ – одинична матриця.

Для цього необхідно, щоб образи $\{\boldsymbol{T}\boldsymbol{x}^i: 0 \le i < n\}$ векторів базису $\{\boldsymbol{x}^{(i)}: 0 \le$ $\le i < n\}$ простору n-грам вихідного тексту $Z_{m,n}$ були базисом простору криптограми $Z_{m,n}$.

Відомо, що кожен базис для $Z_{m,n}$ містить n лінійно незалежних векторів, і будь-який набір з n лінійно незалежних векторів над $Z_{m,n}$ є базисом.

Якщо $\{\boldsymbol{x}^{(i)}: 0 \le i < n\}$ лінійно незалежні над $Z_{m,n}$, тоді їх образи $\{\boldsymbol{T}\boldsymbol{x}^i: 0 \le$ $\le i < n\}$ лінійно незалежні над $Z_{m,n}$ тільки в тому випадку, коли визначник матриці $det\boldsymbol{T}$ не ділиться на будь-яке просте p, яке ділить m.

Приклад

Розглянемо шифрування біграм ($n = 2$) для латинського алфавіту $m = 26$ з $Z_{26,2}$ за допомогою матриці $\boldsymbol{T} = \begin{bmatrix} 3 & 3 \\ 2 & 5 \end{bmatrix}$.

Визначник цієї матриці $\det \boldsymbol{T} = 9 = 1\ mod\ 2 = 9 \bmod 13$, тому існує зворотне перетворення $\boldsymbol{T}^{-1} = \begin{bmatrix} 15 & 17 \\ 20 & 9 \end{bmatrix}$, таке, що $\boldsymbol{T}\boldsymbol{T}^{-1} = \boldsymbol{T}^{-1}\boldsymbol{T} = \boldsymbol{I}$.

Використовуємо це перетворення для шифрування тексту:

PAYMOREMONEY.

Розіб'ємо його на біграми і замінимо кожну букву її чисельним еквівалентом:

$\boldsymbol{x}_1(PA) = (15,0);$ $\boldsymbol{x}_2(YM) = (24,12);$ $\boldsymbol{x}_3(OR) = (14,17);$

$\boldsymbol{x}_4(EM) = (4,12);$ $\boldsymbol{x}_5(ON) = (14,13);$ $\boldsymbol{x}_6(EY) = (4,24).$

Шифрування здійснюється множенням матриці T на відповідні вектори:

$$\boldsymbol{y}_1 = \begin{bmatrix} 3 & 3 \\ 2 & 5 \end{bmatrix} \cdot \begin{vmatrix} 15 \\ 0 \end{vmatrix} = \begin{vmatrix} 19 \\ 4 \end{vmatrix};\ y_2 = \begin{bmatrix} 3 & 3 \\ 2 & 5 \end{bmatrix} \cdot \begin{vmatrix} 24 \\ 12 \end{vmatrix} = \begin{vmatrix} 4 \\ 4 \end{vmatrix};\ y_3 = \begin{bmatrix} 3 & 3 \\ 2 & 5 \end{bmatrix} \cdot \begin{vmatrix} 14 \\ 17 \end{vmatrix} = \begin{vmatrix} 15 \\ 9 \end{vmatrix};$$

$$\boldsymbol{y}_4 = \begin{bmatrix} 3 & 3 \\ 2 & 5 \end{bmatrix} \cdot \begin{vmatrix} 4 \\ 12 \end{vmatrix} = \begin{vmatrix} 22 \\ 16 \end{vmatrix};\ y_5 = \begin{bmatrix} 3 & 3 \\ 2 & 5 \end{bmatrix} \cdot \begin{vmatrix} 14 \\ 13 \end{vmatrix} = \begin{vmatrix} 3 \\ 15 \end{vmatrix};\ y_6 = \begin{bmatrix} 3 & 3 \\ 2 & 5 \end{bmatrix} \cdot \begin{vmatrix} 4 \\ 24 \end{vmatrix} = \begin{vmatrix} 6 \\ 24 \end{vmatrix}.$$

Замінюючи в біграмах тексту числа на відповідні літери, отримуємо шифротекст:

TE EE PJ WQ DP GY.

Система Хілла є одноалфавітною в широкому сенсі слова.

5. Одноразові блокноти, шифр Вернама

Найважливішим для розвитку криптографії було сформульоване К. Шенноном твердження про існування та єдиність абсолютно стійкого шифру. Таким шифром є шифр із застосуванням так званої стрічки одноразового використання, в якій відкритий текст «об'єднується» з повністю випадковим ключем такої самої довжини. Цей результат був доведений К. Шенноном за допомогою розробленого ним теоретико-інформаційного методу дослідження шифрів. Обговоримо особливості будови абсолютно стійкого шифру і можливості його практичного використання.

Нехай $\{K_i;\ 0 < i < n\}$ – незалежні випадкові змінні, що набувають рівноймовірних значень на множині Z_m:

$$P\{(K_0, K_1, \ldots, K_{n-1}) = (k_0, k_1, \ldots, k_{n-1})\} = \left(\frac{1}{m}\right)^n.$$

Одноразова система шифрування перетворює вихідний текст $(x_0, x_1, \ldots, x_{n-1})$ у шифрований текст $(y_0, y_1, \ldots, y_{n-1})$ за допомогою підстановки Цезаря:

$$y_i = E_k(x_i) = (k_i + x_i)\ mod\ m, i = 0, \ldots, n-1.$$

Для такої системи підстановки використовують також термін «одноразова стрічка» і «одноразовий блокнот». Простір ключів до системи одноразової підстановки складається з векторів $(k_0, k_1, \ldots, k_{n-1})$ і містить m^n точок.

Процес шифрування фактично являє собою накладення білого шуму у вигляді нескінченного ключа на вихідний текст, який змінює статистичні характеристики мови джерела. Системи одноразового використання теоретично не розшифровуються, оскільки не містять достатньої інформації для відновлення тексту (без знання ключа).

Підкреслимо, що для абсолютної стійкості істотною є кожна з таких вимог до стрічки одноразового використання:

1) повна випадковість і рівноймовірність ключа (це, зокрема, означає, що ключ не можна виробляти за допомогою будь-якого детермінованого пристрою);

2) рівність довжини ключа і довжини відкритого тексту;

3) одноразове використання ключа.

У разі порушення хоча б однієї з цих умов шифр перестає бути абсолютно стійким, і з'являються принципові можливості для його розкриття (хоча вони можуть бути важко реалізованими). Але саме ці умови і роблять абсолютно стійкий шифр дуже дорогим і непрактичним. Перш ніж користуватися таким шифром, ми маємо забезпечити всіх абонентів достатнім запасом випадкових ключів і виключити можливість їх повторного застосування.

Простим прикладом реалізації абсолютно стійкого шифру (за умови використання ключа, рівного довжині вихідного тексту) є шифр, запропонований в 1926 р. співробітником фірми AT&T Вернамом.

Шифр використовує двійкове подання символів вихідного тексту з використанням телеграфного коду Бодо, де кожна літера вихідного тексту переводиться у п'ятибітовий символ з використанням телетайпа фірми AT&T і записується на паперовій перфострічці. У такий самий спосіб записувався на перфострічці і ключ. Під час шифрування, яке могло бути виконане простим накладенням двох перфострічок однакової довжини одна на одну, ключ додавався до початкового тексту шляхом побітового додавання $\oplus$ за модулем двох символів відкритого тексту і ключа:

$$y_i = x_i \oplus k_i, i = 1, \ldots, n.$$

Тут $x_1, x_2, \ldots, x_n$ – відкритий текст, $k_1, k_2, \ldots, k_n$ – ключ $y_1\ y_2 \ldots y_n$ – шифрований текст.

Дешифрування виконується аналогічно (накладенням перфострічок) і складається з додавання за модулем двох символів шифртексту з тією самою послідовністю ключів:

$$y_i \oplus k_i = x_i \oplus k_i \oplus k_i = x_i, i = 1, \ldots, n.$$

За такої умови послідовності ключів $k_i, i = 1,2, \ldots, n,$ які використовувалися під час шифрування і дешифрування тексту, у разі додавання за модулем 2 компенсують одна одну, внаслідок чого відновлюються символи x вихідного тексту.

Однак під час реалізації методу Вернама виникають серйозні проблеми, пов'язані з необхідністю доставки одержувачу такої самої послідовності ключів, як і у відправника, або з необхідністю безпечного зберігання ідентичних послідовностей ключів у відправника і одержувача. Ці недоліки системи шифрування Вернама були подолані за допомогою шифрування методом гамування.

Хоча одноразова система шифрування і виявилася непридатною на практиці через неможливість практичної реалізації всіх вимог, що забезпечують її теоретичну стійкість, ідея, яка лежить у її основі, знайшла практичне застосування й широко використовується зараз у системі шифрування, що отримала назву методу гамування.

Гама шифру – це псевдовипадкова послідовність, вироблена за певним алгоритмом для шифрування відкритих даних і дешифрування зашифрованих даних. Вона відіграє роль ключа в одноразовій системі шифрування. Інакше кажучи, вона не задовольняє ані вимогу випадковості, оскільки використовується детермінований алгоритм для її вироблення, ані вимогу нескінченної довжини, оскільки всі псевдовипадкові послідовності мають скінченний період. Проте за умови правильно обраного алгоритму генерації гами шифру можна отримати метод шифрування з хорошою практичної стійкістю, достатньою для вирішення реальних завдань захисту інформації.

Один і той самий алгоритм генерації гами шифру виконується і під час пересилання, і з боку одержувача інформації. Обидва мають однакову псевдовипадкову послідовність, що використовується для шифрування і дешифрування. Водночас фактичний (що підлягає передачі) обсяг ключової інформації є дуже малим і складається з декількох чисел, які задають значення параметрів алгоритму та нульового елементу послідовності.

Процес шифрування за цим методом називають гамуванням. Він полягає в генерації гами шифру і її накладенні на вихідний відкритий текст за певним законом, наприклад, з використанням операції додавання за модулем 2.

Шифрування ведеться або посимвольно, або шляхом шифрування даних, об'єднаних у блоки. Під час шифрування блоками відкритий текст m розбивають

на блоки m_i однакової довжини. Гама шифру виробляється у вигляді послідовності блоків $\gamma_i, i = 1, \ldots, M$, аналогічної довжини. Рівняння зашифрування можна записати у вигляді:

$$c_i = \gamma_i \oplus m_i, i = 1, \ldots, M,$$

де c_i – i-й символ (блок) шифротексту; γ_i – i-й символ (блок) гами шифру; m_i – i-й символ (блок) відкритого тексту; M – кількість символів (блоків) відкритого тексту.

Процес дешифрування зводиться до повторної генерації гами шифру з боку одержувача інформації за тим самим алгоритмом і накладення цієї гами на зашифровані дані. Рівняння дешифрування має вигляд:

$$m_i = \gamma_i \oplus c_i, i = 1,2, \ldots, M.$$

Отриманий зашифрований текст є доволі важким для розкриття в тому випадку, якщо гама шифру не містить повторюваних послідовностей. В ідеалі гама шифру повинна змінюватися випадковим непередбачуваним способом для кожного слова відкритого тексту. Якщо період гами перевищує довжину всього тексту і невідома жодна частина вихідного тексту, то шифр можна розкрити тільки прямим перебором ключа. Тобто криптостійкість методу визначається періодом гами.

Слабке місце методу гамування в тому, що він стає безсилим, якщо зловмиснику стає відомим фрагмент вихідного тексту довжиною більш ніж період гами і відповідна йому шифрограма. Простим вирахуванням за модулем він знаходить ключ і по ньому відновлюється вся послідовність. Криптоаналітика також може частково або повністю відновити ключ на основі припущень про зміст вихідного тексту. Так, якщо більшість повідомлень, які надсилаються, починається зі слів «ЦІЛКОМ ТАЄМНО», то криптоаналіз усього тексту значно полегшується. Також багато текстових редакторів, наприклад, Word, вставляють у початок файла стандартну службову інформацію, що знижує криптостійкість під час шифрування цих файлів за методом гамування.

Питання для самоконтролю

1. Що називають алфавітом криптосистеми?
2. Що низивають n-грамою?
3. Наведіть формулу, що описує систему шифрування Цезаря.
4. Опишіть математичні властивості операції підстановки.
5. Опишіть шифр Віженера у термінах підстановок.
6. Наведіть приклад шифрування за допомогою шифру Хілла.
7. Яка математична операція лежить в основі шифру Вернама?

ТЕМА 8. ШИФР DES (ЛЕКЦІЇ 8 ТА 9)

1. Мережа (сітка) Фейстеля.
2. Стандарт шифрування DES, основні властивості.
3. Режими роботи шифру ECB та CBC.

1. Мережа (сітка) Фейстеля

Мережею (іноді «сіткою», Feistel's network) Фейстеля називається метод оборотних перетворень тексту, за якого значення, обчислене від однієї з частин тексту, накладається на інші частини. Часто структура мережі виконується так, що для шифрування і дешифрування використовується один і той самий алгоритм – відмінність полягає тільки в порядку використання ключа.

Незалежні потоки інформації, які породжені вхідним блоком, називаються гілками мережі. Величини V_i називаються параметрами мережі, зазвичай це функції від ключа. Функція F називається функцією шифрування, або ж твірною функцією. Дія, що складається з одноразового обчислення функції F і подальшого накладання її результату на іншу гілку з обміном їх місцями, називається циклом або раундом мережі Фейстеля.

На рис. 3 показана структура шифру, запропонованого Фейстелем. На вхід алгоритму шифрування подаються блок відкритого тексту довжиною 64 біти і ключ V. Блок відкритого тексту поділяється на дві рівні частини $X1$ і $X2$, які послідовно проходять через 16 раундів обробки (число раундів відповідає шифру DES, у загальному випадку кількість раундів може відрізнятись), а потім об'єднуються знову для отримання блоку шифрованого тексту відповідної довжини. Для раунду i в якості вхідних даних виступають $X1^i$ і $X2^i$, які отримані на виході попереднього раунду, і підключ V_i, який обчислюється за загальним ключем V. Зазвичай усі підключі V_i відрізняються як від загального ключа V, так і один від одного.

Всі раунди обробки проходять по одній і тій самій схемі. Спочатку для правої половини блоку даних виконується операція підстановки. Вона полягає в застосуванні до лівої половини блоку даних деякої функції раунду F і подальшому додаванні отриманого результату з правою половиною блоку даних за допомогою операції XOR. Для всіх раундів функція F має одну і ту саму структуру, але залежить від параметра – підключа раунду V_i. Після підстановки виконується перестановка, що являє собою обмін місцями двох половин блоку даних.

Процес дешифрування мережі Фейстеля принципово не відрізняється від процесу шифрування. Застосовується той самий алгоритм, але на вхід подається шифрований текст, а підключі V_i використуються е зворотній послідовності: для першого раунду береться підключ V_n, для другого – V_{n-1}, і так далі, доти, поки не буде вве-

дений ключ V_1 для останнього раунду. Така властивість цієї схеми шифрування виявляється дуже зручною, тому що для дешифрування не потрібно вводити інший алгоритм, відмінний від алгоритму шифрування.

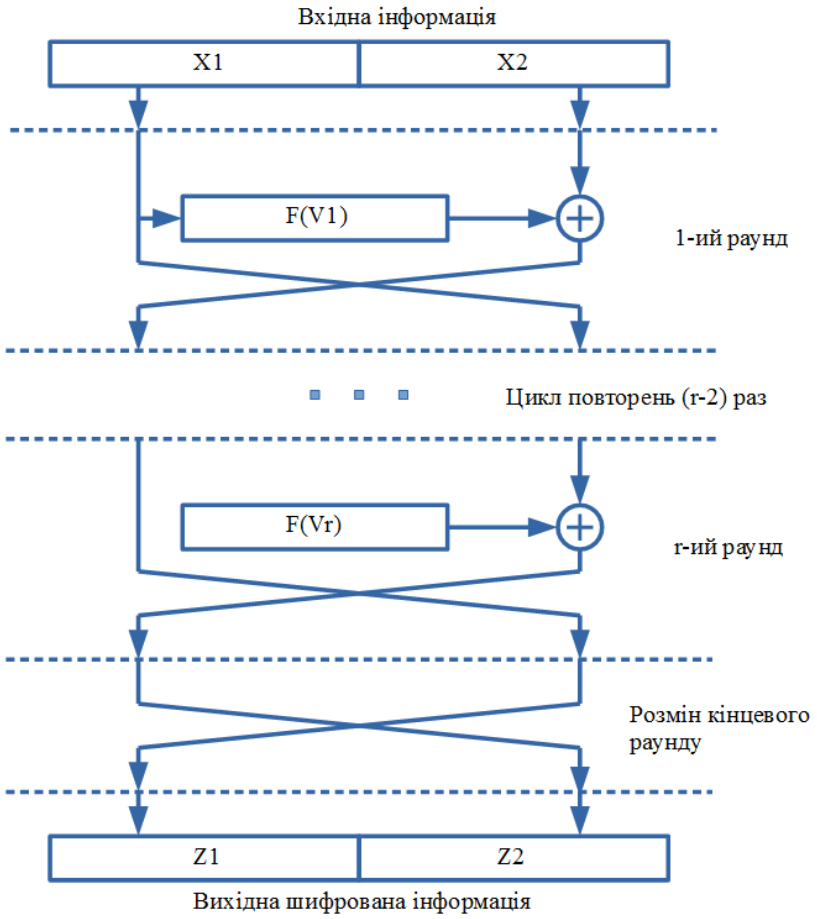

Рис. 3. Класична мережа Фейстеля

Мережа Фейстеля надійно зарекомендувала себе як крипостійка схема проведення криптоперетворень і є невід'ємною частиною стандарту шифрування DES.

2. Стандарт шифрування DES, основні властивості

Стандарти щодо захисту даних від несанкціонованого доступу були потрібні для шифрування, встановлення автентичності особистості і даних (автентифікації), контролю доступу, надійного зберігання і передачі даних. Внаслідок співпраці трьох організацій США – Національного бюро стандартів (NBC), Управління національної безпеки (NSA) і фірми IBM – подібний стандарт, який отримав назву DES, був розроблений і опублікований у 1975 р. у спеціальному виданні Federal Register. Після дворічних випробувань з метою пошуку в алгоритмі DES «таємної лазівки», а також з економічних питань (зокрема щодо встановлення довжини ключа) було прийнято рішення залишити стандарт без змін.

Ефективна довжина ключа у 56 бітів цілком задовольняла потенційних користувачів на найближчі 15–20 років, оскільки загальна кількість ключів у цьому випадку оцінювалась цифрою орієнтовно у $7{,}6^{1016}$. DES став одним з перших «відкритих» шифроалгоритмів. Усі схеми, які використовуються для його реалізації, були опубліковані і ретельно перевірені. Секретним був тільки ключ, за допомогою якого здійснюється кодування і декодування інформації.

Алгоритм DES базується на науковій роботі Шеннона 1949 р., яка зв'язала криптографію з теорією інформації. Шеннон виділив два загальні принципи, що використовуються в практичних шифрах: розсіювання та перемішування. Розсіюванням він назвав поширення впливу одного знаку відкритого тексту на нескінченну кількість знаків шифртексту, що дає змогу приховати статистичні властивості відкритого тексту. Під перемішуванням Шеннон розумів використання взаємозв'язку статистичних властивостей відкритого і шифрованого тексту. Але шифр повинен не тільки ускладнювати розкриття, а й забезпечувати легкість шифрування і дешифрування за відомого секретного ключа. Тому була прийнята ідея використовувати комбінацію простих шифрів, кожен із яких робить невеликий внесок у значне сумарне розсіювання і перемішування.

Основні переваги алгоритму DES є такими:

– використовується тільки один ключ довжиною 56 бітів;

– якщо повідомлення зашифроване за допомогою одного пакету, для розшифровки ви можете використовувати будь-який інший;

– відносна простота алгоритму забезпечує високу швидкість обробки інформації;

– доволі висока стійкість алгоритму.

DES здійснює шифрування 64-бітових блоків даних за допомогою 56-бітового ключа. Розшифрування в DES є операцією зворотного шифрування і виконується шляхом повторення операцій шифрування у зворотній послідовності.

Процес шифрування полягає в початковій перестановці бітів 64-бітового блоку, 16 циклах шифрування і, нарешті, зворотній перестановці бітів.

Усі таблиці, наведені нижче, є стандартними, а отже, повинні включатися в реалізацію алгоритму в незмінному вигляді. Всі перестановки і коди в таблицях підібрані розробниками так, щоб максимально ускладнити процес розшифровки шляхом підбору ключа. Структура алгоритму DES неведена на рис. 4 та рис. 5.

Нехай з файла зчитано черговий 8-байтовий блок T, який перетворюється за допомогою матриці початкової перестановки IP (табл. 1) так: біт 58 блоку T стає бітом 1, біт 50 – бітом 2 тощо, що дає в результаті: $T(0) = IP(T)$. Отримана послідовність бітів $T(0)$ розділяється на дві послідовності по 32 біти кожна: $L(0)$ – ліві або старші біти, $R(0)$ – праві або молодші біти.

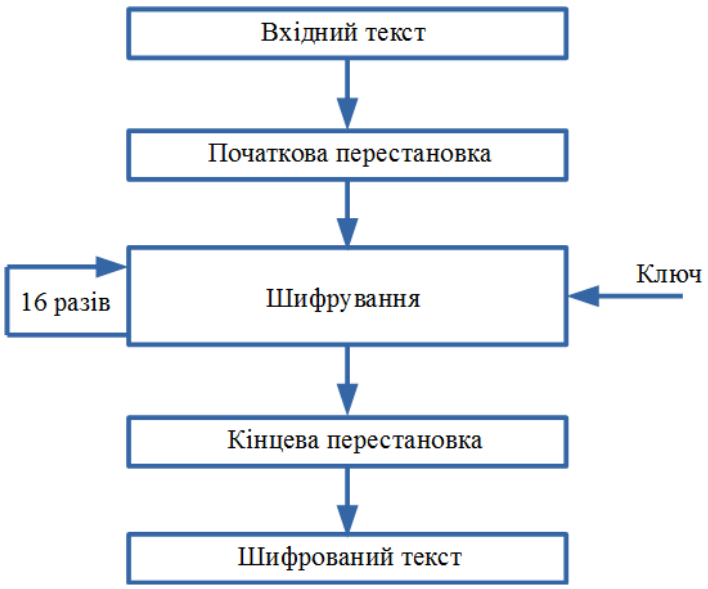

Рис. 4. Схема шифрування в алгоритмі DES

Таблиця 1. Матриця початкової перестановки IP

58	50	42	34	26	18	10	02
60	52	44	36	28	20	12	04
62	54	46	38	30	22	14	06
64	56	48	40	32	24	16	08
57	49	41	33	25	17	9	01
59	51	43	35	27	19	11	03
61	53	45	37	29	21	13	05
63	55	47	39	31	23	15	07

Функція f називається функцією шифрування. На вхід подаються такі аргументи: 32-бітова послідовність $R(i-1)$, отримана на $(i-1)$-ій ітерації, і 48-бітовий ключ $K(i)$, який є результатом перетворення 64-бітового ключа K. Детально функція шифрування і алгоритм отримання ключів К(i) описані нижче.

На 16-й ітерації отримують послідовності $R(16)$ і $L(16)$ (без перестановки), які об'єднуються в 64-бітову послідовність $R(16)L(16)$. Потім позиції бітів цієї послідовності переставляють відповідно до матриці фінальної зворотної перестановки IP^{-1} (табл. 2).

Процес розшифрування даних є інверсним відносно процесу шифрування. Всі дії повинні бути виконані у зворотному порядку. Це означає, що дані, які розшифровуються, спочатку переставляються відповідно до матриці IP^{-1}, а потім над послідовністю біт $R(16)L(16)$ виконуються ті самі дії, що і в процесі шифрування, але у зворотному порядку.

Таблиця 2. Матриця зворотної перестановки IP^{-1}

40	08	48	16	56	24	64	32
39	07	47	15	55	23	63	31
38	06	46	14	54	22	62	30
37	05	45	13	53	21	61	29
36	04	44	12	52	20	60	28
35	03	43	11	51	19	59	27
34	02	42	10	50	18	58	26
33	01	41	9	49	17	57	25

Ітеративний процес розшифрування може бути описаний такими формулами:

$$R(i-1) = L(i), i = 1,2,\ldots,16.$$

На 16-й ітерації отримують послідовності $L(0)$ і $R(0)$, які конкатенуються в 64-бітову послідовність $L(0)R(0)$.

Потім позиції бітів цієї послідовності переставляють відповідно до матриці IP. Результатом такої перестановки є вихідна 64-бітова послідовність.

Тепер розглянемо функцію шифрування $f(R(i-1), K(i))$.

Для обчислення значення функції f використовуються такі функції-матриці:

– E-розширення 32-бітової послідовності до 48-бітової;

– $S1, S2, \ldots, S8$-перетворення 6-бітового блоку в 4-бітовий;

– P-перестановка біт в 32-бітової послідовності.

Функція розширення E визначається табл. 3. Зверніть увагу, що початок кожного блоку – це два біти блоку попереднього (за винятком першого та останнього блоку-рядка).

Таблиця 3. Функція розширення E

32	01	02	03	04	05
04	05	06	07	08	09
08	09	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	01

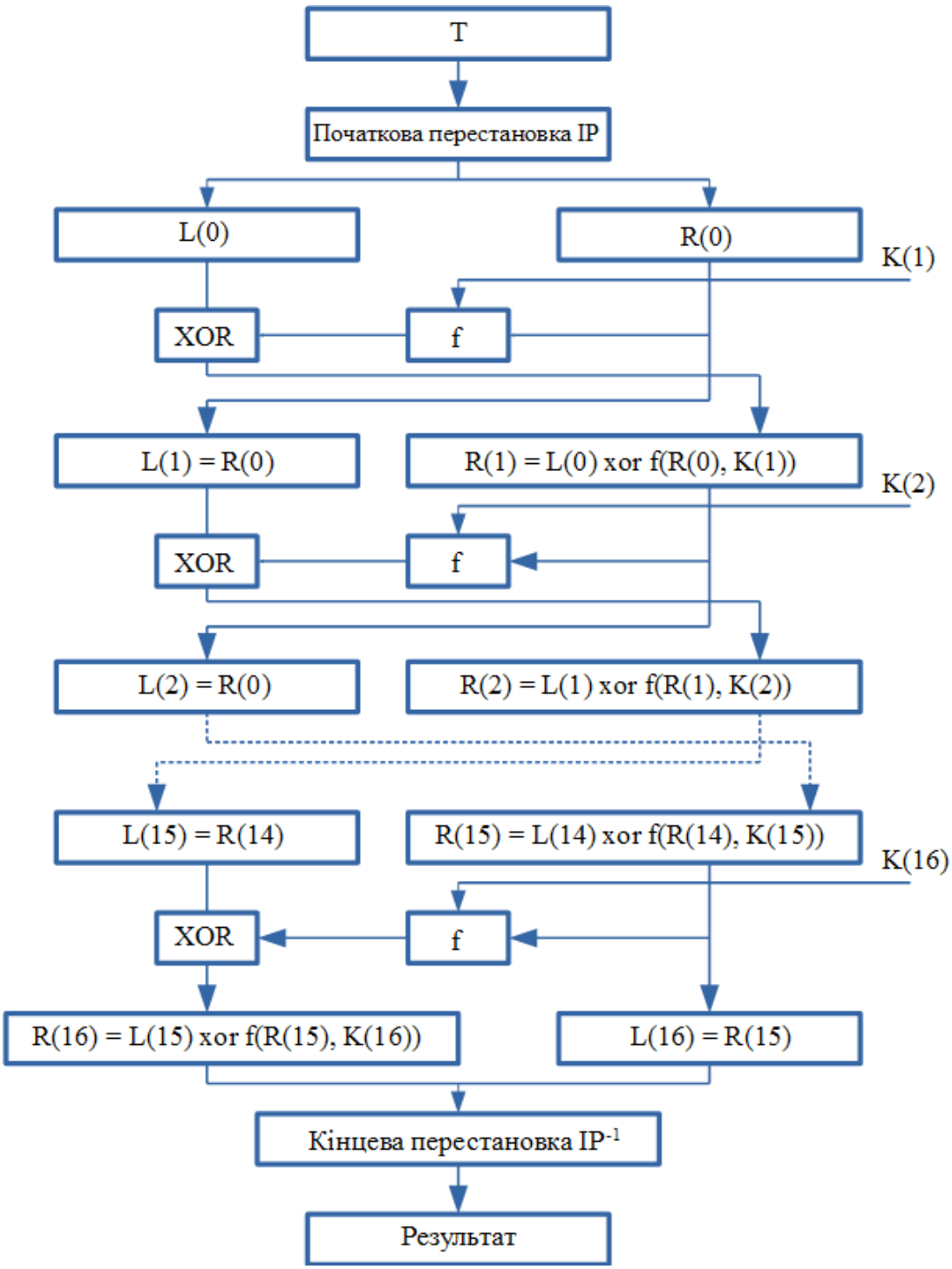

Рис. 5. Структура алгоритму шифрування DES

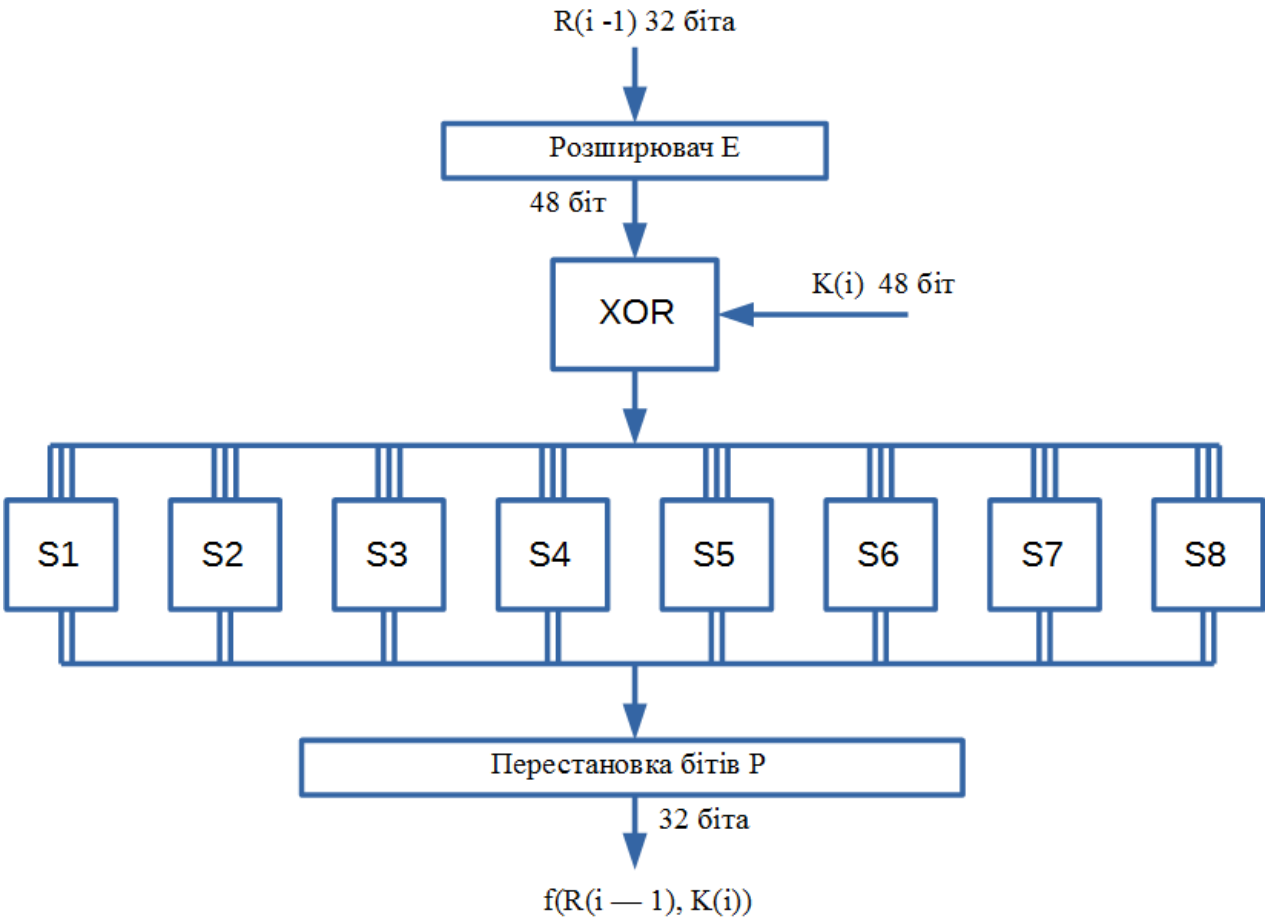

Рис. 6. Обчислення функції f(R(i-1),K(i))

Результатом функції $E(R(i-1))$ є 48-бітова послідовність, яка складається за модулем 2 (операція XOR) із 48-бітовим ключем $K(i)$. Виходить 48-бітова послідовність, яка розбивається на вісім 6-бітових блоків $B(1), B(2), B(3), B(4), B(5), B(6), B(7), B(8)$. Тобто:

$$E(R(i-1)) \text{ XOR } K(i) = B(1)B(2)\dots B(8).$$

Функції $S1$, $S2$, ... , $S8$ визначаються табл. 4.

До табл. 4 потрібні додаткові пояснення. Нехай на вхід функції-матриці Sj надходить 6-бітовий блок $B(j) = b1b2b3b4b5b6$, тоді двобітове число $b1b6$ вказує номер рядка матриці, а $b2b3b4b5$ – номер стовпчика. Результатом $Sj(B(j))$ буде 4-бітовий елемент, розташований на перетині зазначених рядка і стовпця.

Наприклад, $B(1) = 011011$. Тоді $S1(B(1))$ розташований на перетині рядка 1 і стовпця 13. У стовпці 13 рядка 1 задано значення 5. Тому $S1(011011) = 0101$.

Застосувавши операцію вибору до кожного з 6-бітових блоків $B(1)$, $B(2), \dots, B(8)$, отримуємо 32-бітову послідовність $S1(B(1))$ $S2(B(2))$ $S3(B(3))\dots S8(B(8))$.

Таблиця 4. Функції перетворення S1, S2, ..., S8

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	
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	
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	S8
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	

Нарешті, для отримання результату функції шифрування треба переставити біти цієї послідовності. Для цього застосовується функція перестановки P (табл. 5). У вхідній послідовності біти переставляється так, щоб біт 16 став бітом 1, а біт 7 – бітом 2 тощо.

Отже,

$$f(R(i-1), K(i)) = P(S1(B(1)), \ldots S8(B(8))).$$

Таблиця 5. Функція перестановки P

16	07	20	21
29	12	28	17
01	15	23	26
05	18	31	10
02	08	24	14
32	27	03	09
19	13	30	06
22	11	04	25

Щоб завершити опис алгоритму шифрування даних, треба навести алгоритм отримання 48-бітових ключів $K(i), i = 1\ldots16$. На кожній ітерації використовується нове значення ключа $K(i)$, яке обчислюється з початкового ключа K. Ключ K являє собою 64-бітовий блок з вісьмома бітами контролю по парності, які розташовані в позиціях 8, 16, 24, 32, 40, 48, 56, 64.

Для видалення контрольних бітів і перестановки інших використовується функція G початкової підготовки ключа (табл. 6).

Таблиця 6. Матриця G первинної підготовки ключа

57	49	41	33	25	17	09
01	58	50	42	34	26	18
10	02	59	51	43	35	27
19	11	03	60	52	44	36
63	55	47	39	31	23	15
07	62	54	46	38	30	22
14	06	61	53	45	37	29
21	13	05	28	20	12	04

Результат перетворення $G(K)$ розбивається на два 28-бітові блоки $C(0)$ і $D(0)$, причому $C(0)$ буде складатися з бітів 57, 49, ..., 44, 36 ключа K, а $D(0)$ буде складатися з бітів 63, 55, ..., 12, 4 ключа K. Після визначення $C(0)$ і $D(0)$ рекурсивно визначаються $C(i)$ та $D(i)$, $i = 1 \ldots 16$. Для цього застосовують циклічний зсув ліворуч на один або два біти залежно від номера ітерації, як показано в табл. 7.

Таблиця 7. Таблиця зсувів для обчислення ключа

Номер ітерації	Зсув (біт)	Номер ітерації	Зсув (біт)
01	1	09	1
02	1	10	2
03	2	11	2
04	2	12	2
05	2	13	2
06	2	14	2
07	2	15	2
08	2	16	1

Отримане значення знову «перемішується» відповідно до матриці H у табл. 8.

Таблиця 8. Матриця H кінцевої обробки ключа

14	17	11	24	01	05
03	28	15	06	21	10
23	19	12	04	26	08
16	07	27	20	13	02
41	52	31	37	47	55
30	40	51	45	33	48
44	49	39	56	34	53
46	42	50	36	29	32

Ключ $K(i)$ буде складатися з бітів 14, 17, ..., 29, 32 послідовності C(i)D(i). Отже:

$$K(i) = H(C(i)D(i)).$$

Відновлення вихідного тексту здійснюється за таким самим алгоритмом, але спочатку використовуєте ключ $K(15)$, потім – $K(14)$ тощо.

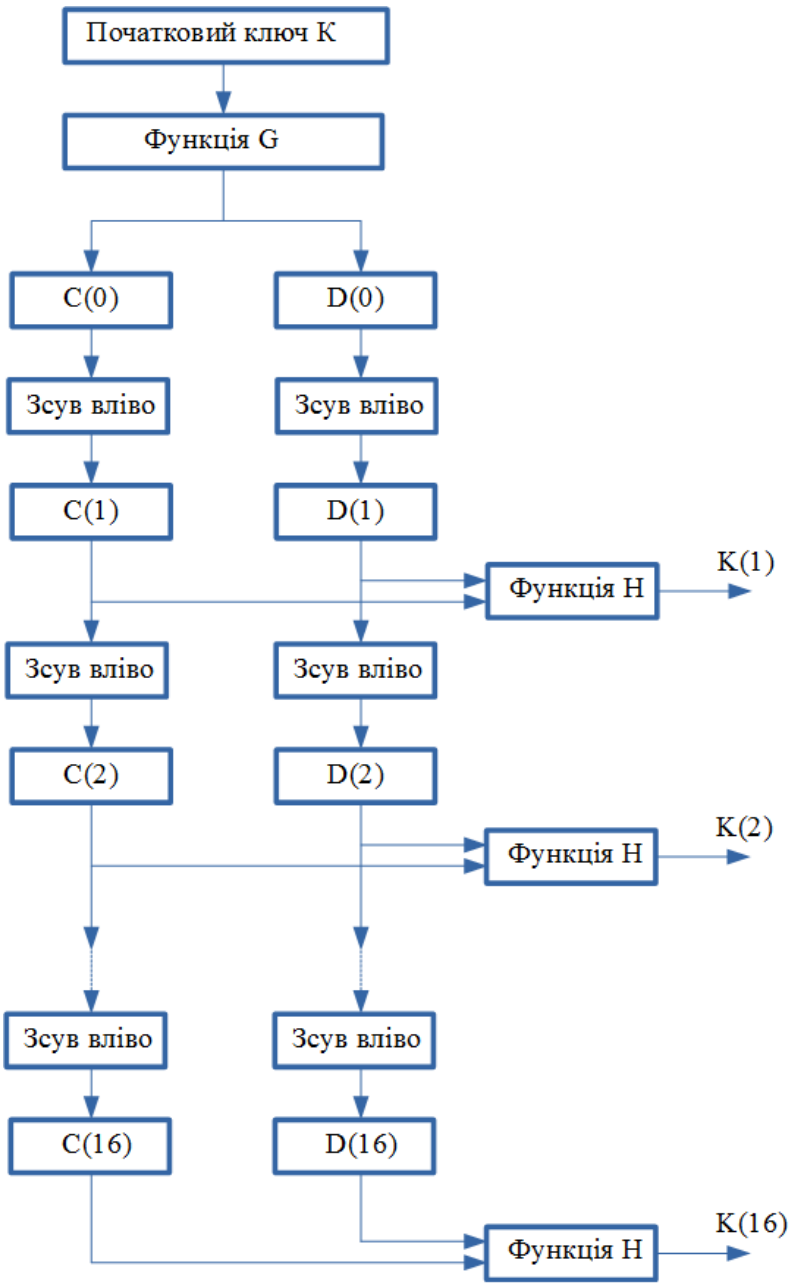

Рис. 7. Блок-схема алгоритму обчислення ключа $K(i)$

3. Режими роботи алгоритму ECB та CBC

Для найбільш повного задоволення всіх вимог, що ставляться до комерційних систем шифрування, реалізовано кілька режимів роботи алгоритму DES. Найбільш широкого поширення набули режими:

– електронний шифроблокнот (Electronic Codebook) – ECB;
– поєднання цифрових блоків (Cipher Block Chaining) – CBC;
– цифровий зворотний зв'язок (Cipher Feedback) – CFB;
– зовнішній зворотний зв'язок (Output Feedback) – OFB.

DES-ECB. У цьому режимі початковий файл M розбивається на 64-бітові блоки (по 8 байтів): $M = M(1)\,M(2)\ldots M(n)$. Кожен із цих блоків кодується не-

залежно з використанням одного й того самого ключа шифрування. Основна перевага цього алгоритму – простота реалізації. Недоліком є відносно слабка стійкість проти кваліфікованих криптоаналітиків.

Зокрема, не рекомендується використовувати цей режим роботи для шифрування EXE-файлів, оскільки перший блок – заголовок файла – є цілком вдалим початком для зламу всього шифру.

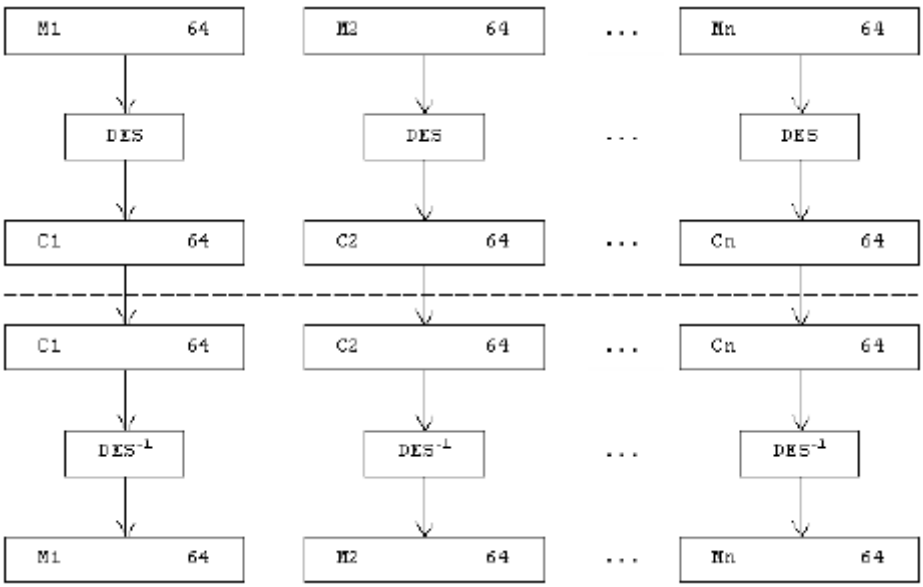

Рис. 8. Робота алгоритму DES в режимі ECB

DES-CBC. У цьому режимі початковий файл M, як і в режимі ECB, розбивається на 64-бітові блоки: $M = M(1)\ M(2)\dots M(n)$. Перший блок $M(1)$ складається за модулем 2 з 64-бітовим початковим вектором IV, який змінюється щодня і тримається в секреті. Отримана сума потім шифрується з використанням ключа DES, відомого і відправнику, і одержувачу інформації.

Отриманий 64-бітовий блок шифртексту $C(1)$ складається за модулем 2 з другим блоком початкового тексту, результат шифрується і виходить другий 64-бітовий блок шифртексту $C(2)$ тощо. Процедура повторюється доти, поки не будуть оброблені всі блоки початкового тексту.

Отже, для всіх $i = 1\dots n$ блок шифртексту $C(i)$ визначається так: $C(i) = DES(M(i)\ \text{XOR}\ C(i-1))$, $C(0)$ – початкове значення шифру, яке дорівнює початковому вектору. Розшифрування виконується так:

$$M(i) = C(i-1)\ \text{XOR}\ DES^{-1}(C(i)).$$

Перевага цього режиму полягає в тому, що він не допускає накопичення помилок під час передачі даних. Блок $M(i)$ є функцією лише $C(i-1)$ і $C(i)$. Тому помилка під час передачі призведе до втрати тільки двох блоків початкового тексту.

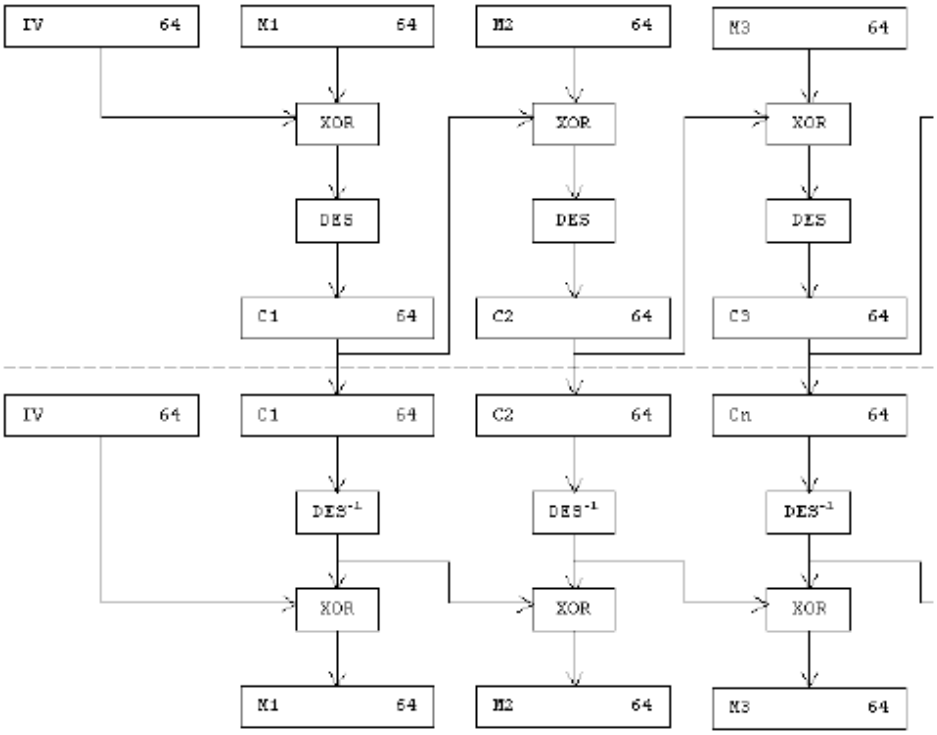

Рис. 9. Робота алгоритму в режимі CBC

Питання для самоконтролю

1. Наведіть блок-схему сітки Фейстеля.
2. Який розмір блоку вхідного потоку в алгоритмі DES?
3. Який ефективний розмір ключа використовується в DES?
4. Які два загальні принципи практичних шифрів використано в DES?
5. Наведіть схему шифрування (загальний порядок операцій) алгоритму DES.
6. Опишіть процедуру S перетворення b-блоків (перетворення 6-бітового блоку в 4-бітовий).
7. Опишіть процедуру генерації ключів в алгоритмі DES.
8. Які існують найбільш поширені режими роботи алгоритму DES?
9. Опишіть роботу алгоритму DES в режимі ECB.

ТЕМА 9. ШИФР AES (ЛЕКЦІЇ 10 ТА 11)

1. Історія виникнення AES.
2. Термінологія алгоритму.
3. Математичні основи AES.
4. Алгоритм AES, шифрування.
5. Дешифрування.

1. Історія виникнення та основні поняття AES

У 1998 р. NIST (National Institute of Standards and Technology) оголосив конкурс на створення алгоритму симетричного шифрування. Алгоритм отримав назву AES (Advanced Encryption Standard). Його планувалося прийняти як стандарт Сполучених Штатів Америки замість застарілого на той час стандарту DES (Digital Encryption Standard), що був американським стандартом з 1977 р. Необхідність у прийнятті нового стандарту була викликана невеликою довжиною ключа DES (56 бітів), що давало змогу успішно застосовувати метод прямого перебору ключів для зламу DES. До того ж архітектура DES була орієнтована на апаратну реалізацію, і програмна реалізація алгоритму на платформах з обмеженими ресурсами не давала достатньої швидкодії. Модифікація TripleDES мала вже достатню довжину ключа, але водночас була ще й повільною.

Вимоги до кандидатів на новий стандарт були такими:

– це мав бути блоковий шифр;

– довжина блоку мала дорівнювати 128 бітам;

– ключі мали бути довжиною 128, 192 і 256 бітів.

Додатково кандидатам рекомендувалося:

– використовувати операції, які легко реалізуються як апаратно (в мікрочипах), так і програмно (на персональних комп’ютерах і серверах);

– орієнтуватися на 32-розрядні процесори;

– не ускладнювати без необхідності структуру шифру для того, щоб усі зацікавлені сторони були в змозі самостійно провести незалежний криптоаналіз алгоритму і переконатися, що в ньому не закладено жодних недокументованих можливостей.

До того ж алгоритм мав поширюватися по всьому світу на неексклюзивних умовах і без плати за користування патентом.

Алгоритм AES насамперед повинен був пропонувати високу ступінь захисту, мати просту структуру високу продуктивність. На рівні внутрішньої архітектури він повинен володіти надійністю, достатньою для того, щоб протистояти майбутнім спробам його зламу.

Було проведено конкурс серед алгоритмів шифрування на роль AES. 2 жовтня 2000 р. було оголошено, що переможцем конкурсу став алгоритм Rijndael, і поча-

лася процедура стандартизації. 28 лютого 2001 р. було опубліковано проєкт, а 26 листопада 2001 р. AES був прийнятий як стандарт FIPS 197.

AES не є тотожним Rijndael, оскільки оригінальний алгоритм Rijndael підтримує більш широкий діапазон довжин ключів і блоків. В AES розмір ключа фіксований і залежить від модифікації, а в Rijndael підтримуються різні довжини ключів – від 128 до 256 бітів, з кроком 32 біти. AES оперує блоком даних з 16 байтів (128 бітів), а Rijndael дає змогу обирати розмір блоку.

Rijndael – швидкий і компактний алгоритм із простою математичною структурою. Він продемонстрував хорошу стійкість до атак, під час яких намагаються декодувати зашифроване повідомлення, аналізуючи зовнішні прояви алгоритму, в тому числі рівень енергоспоживання і час виконання. Алгоритм відрізняє «внутрішній паралелізм» – блоки вхідного потоку можуть оброблятись паралельно і незалежно один від одного, що дає змогу забезпечити ефективне використання апаратних ресурсів.

У подальшому матеріалі під AES будемо розуміти Rijndael з ключем в 128 бітів і блоком даних у 16 байтів.

2. Термінологія алгоритму

У табл. 9 в алфавітному порядку наведені основні позначення функцій і параметрів криптоперетворень AES.

Таблиця 9. Параметри алгоритму, символи і функції

Позначення	Сенс позначення
AddRoundKey()	Перетворення в шифрування і дешифрування, в якому раундовий ключ додається до state-форми з використанням операції XOR
InvMixColumns()	Перетворення в дешифрування, яке є інверсією MixColumns
InvShiftRows()	Перетворення в дешифрування, яке є інверсією ShiftRows
InvSubBytes()	Перетворення в дешифрування, яке є інверсією SubBytes
K	Ключ шифрування – масив з 128 бітів або 16 байтів
MixColumns()	Перетворення в процесі шифрування, пов’язане з перестановкою стовпців state
Rcon()	Масив раундових констант-слів
RotWord()	Функція, що використовується в операції розширення ключа, отримує на вхід 4-байтове слово і виконує циклічну перестановку
ShiftRows()	Перетворення в процесі шифрування, яке виконується над state, циклічна перестановка 3 нижніх рядків state
SubBytes()	Перетворення в процесі шифрування, яке виконується над state і полягає у заміні кожного байта з використанням таблиці заміни (S-box)
Nk	Довжина ключа в словах, для AES – 4 слова
Nb	Довжина блоку в словах, для AES – 4 слова
Nr	Число раундів під час шифрування – 10
XOR	Операція над бітами
⊗	Операція над бітами
·	Множення у скінченному полі

3. Математичні основи AES

Алгоритм AES оперує байтами, тобто числами від 0 до 255 або 8-розрядними двійковими числами. Всі байти в AES інтерпретуються як елементи скінченного поля $F(2^8)$. Тут число 2^8 вказує на кількість елементів скінченного поля.

Скінченне поле F – це множина довільних елементів (членів множини), що містить скінченну кількість таких членів. Для елементів поля визначені дві бінарні операції: додавання двох елементів поля і множення двох елементів поля, результатом яких для будь-яких двох елементів поля є теж елемент поля.

Серед елементів поля є елемент «нуль». Його додавання до будь-якого елементу дає в результаті цей самий елемент, а його множення на будь-який елемент поля дає «нуль». Також серед елементів поля є елемент «одиниця», множення на який будь-якого елементу дає в результаті цей самий елемент. Для кожного елементу поля існує «мультиплікативно зворотний» елемент, множення елементу на зворотний до нього дає в результаті елемент «одиниця».

Елементи скінченного поля можуть «додаватись» і «помножуватись», але ці операції відрізняються від операцій, які використовуються для простих чисел.

Додавання виконується за допомогою операції побітового додавання за модулем 2. Операція позначається XOR або $\oplus$ і виконується над двійковим представленням числа порозрядно так, що $1 \oplus 1 = 0, 1 \oplus 0 = 0 \oplus 1 = 1, 0 \oplus 0 = 0$.

Для двох байтів $A = \{a_7, a_6, a_5, a_4, a_3, a_2, a_1, a_0\}$, де $a_i -$ i-й розряд двійкового представлення числа $A \in [0..255]$, і $B = \{b_7, b_6, b_5, b_4, b_3, b_2, b_1, b_0\}$ сума буде дорівнювати байту $C = \{c_7, c_6, c_5, c_4, c_3, c_2, c_1, c_0\}$, де $c_i = a_i \oplus b_i$.

Для множення використовується поліноміальне представлення двійкового числа. Так, байт $A = \{a_7, a_6, a_5, a_4, a_3, a_2, a_1, a_0\}$ може бути представлений у вигляді полінома:

$a(x) = a_7 \cdot x^7 + a_6 \cdot x^6 + a_5 \cdot x^5 + a_4 \cdot x^4 + a_3 \cdot x^3 + a_2 \times \cdot x^2 + a_1 \cdot x^1 + a_0 \cdot x^0$,

де a_i набуває значення 0 або 1.

У поліноміальному представленні множення (позначене ·) у скінченному полі $F(2^8)$ виконується між поліномами за модулем полінома ступеня 8, який неможливо привести в полі F. Поліном неможливо привести в полі F, якщо він ділиться тільки на себе і на «одиницю» поля, тобто одиничний для такого поля елемент. Для алгоритму AES в полі F неможливо привести поліном:

$$m(x) = x^8 + x^4 + x^3 + x + 1.$$

Результатом множення двох байтів A і B є остача від ділення добутку поліноміального представлення $a(x)$ для A на поліноміальне представлення $b(x)$ для байта B на поліном $m(x)$.

Наприклад, обчислимо добуток:

$$57_8 \cdot 83_8.$$

Число 57_8 у двійковому форматі може бути представлене у вигляді 1010111, отже поліноміальним представленням числа 57_8 є:

$$x^6 + x^4 + x^2 + x + 1,$$

поліноміальним представленняя числа 83_8 (у двійковій системі 10000011) є:

$$x^7 + x + 1.$$

Результат множення поліномів буде таким:

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

Підсумовування коефіцієнтів за однакових ступенів проводиться за правилами додавання в полі, тобто в нашому випадку за модулем 2, наприклад:

$$1 \cdot x^7 + 1 \cdot x^7 = (1 \oplus 1) \cdot x^7 = 0 \cdot x^7,$$

отримаємо:

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

Обчислюємо лишок від ділення добутку на $m(x)$, тоді отримаємо:

$$(x^{13} + x^{11} + x^9 + x^8 + x^6 + x^5 + x^4 + x^3 + 1) \bmod (x^8 + x^4 + x^3 + x + 1) =$$
$$= x^7 + x^6 + 1,$$

що є поліноміальних поданням числа 11000001_2 або 193_8. Отже,

$$57 \cdot 83 = 193.$$

Ділення за модулем $m(x)$ гарантує, що результат буде двійковим поліномом ступеня меншим за 8, а отже може бути представлений у вигляді байта. На відміну від операції додавання, немає іншої простої операції на рівні бітів, щоб виконати саме множення.

4. Алгоритм AES, шифрування

Для шифрування необхідний ключ. Розмір ключа в алгоритмі AES 128 бітів, зазвичай ключ представляють як матрицю 4×4 байти. Схема шифрування наведена нижче.

На початку процесу шифрування вхідні дані розбиваються на блоки розміром 16 байтів або 128 бітів. Якщо повний розмір даних не кратний 16 байтам, то дані доповнюються до розміру, кратного 16 байтам.

Першим кроком шифрування можна назвати початкову підготовку потоку. Вхідний блок у 128 бітів розбивається на байти по 8 бітів, причому порядок бітів у байті починається з кінця, як показано на схемі.

Підготовлений блок (поданий у вигляді байтів) в алгоритмі AES називається state і зазвичай представляється у вигляді матриці 4×4 байти (див. рис. 10). Операція шифрування кожного блоку даних проводиться незалежно від вмісту інших блоків. Після закінчення шифрування блоку матриця заповнюється наступною порцією даних і процес повторюється.

№ біта вхідного потоку	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	…
№ байта	0								1								2		
№ бітів у байт	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	…

S_{00}	S_{01}	S_{02}	S_{03}
S_{10}	S_{11}	S_{12}	S_{13}
S_{20}	S_{21}	S_{22}	S_{23}
S_{30}	S_{31}	S_{32}	S_{33}

Рис. 10. Матриця state

Кожен блок шифрується в кілька етапів – раундів. Схема криптоперетворень може бути записана так:

1. Розширення ключа KeyExpansion.
2. Початкова операція – AddRoundKey – додавання основного ключа.
3. 9 раундів з чотирьох кроків кожен:
   3.1. SubBytes – заміна байтів state по таблиці замін.
   3.2. ShiftRows – циклічний зсув рядків state.
   3.3. MixColumns – перестановка стовпчиків state.
   3.4. AddRoundKey – додавання раундового ключа.
4. Заключний 10-й раунд:
   4.1. SubBytes – заміна байтів state по таблиці замін.
   4.2. ShiftRows – циклічний зсув рядків state.
   4.3. AddRoundKey – додавання раундового ключа.

Далі докладно описано кожне з перетворень.

Перетворення SubBytes

Перетворення SubBytes – це нелінійна заміна байтів, що проводиться над кожним байтом state з використанням таблиці заміни S-box (див. табл. 10). Метою застосування таблиці замін є ускладнення лінійного і диференціального криптоаналізу.

Таблиця замін в алгоритмі AES є фіксованою. У таблиці числа представлені у шістнадцятковій системі числення. В цій системі числення будь-яке значення байта представимо не більше ніж двома 16-ми розрядами.

Заміна байта по таблиці S-box проводиться так:

1. Байт Z перетворюється в шістнадцяткову систему числення, наприклад, $Z = XYh$, X – старший розряд, Y – молодший розряд. Якщо старшого розряду немає, то він замінюється нулем.

2. В S-box обирається рядок X і стовпець Y.

3. Значення Z' на перетині рядка X і стовпця Y таблиці S-box використовується як заміна Z.

Наприклад, для значення байта $Z = 9Ah$ заміною відповідно до таблиці S-box буде $Z' = B8h$. Повний процес SubBytes полягає в заміні всіх 16 байтів матриці state (рис. 11).

Таблиця 10. Таблиця заміни байтів S-box алгоритму AES

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

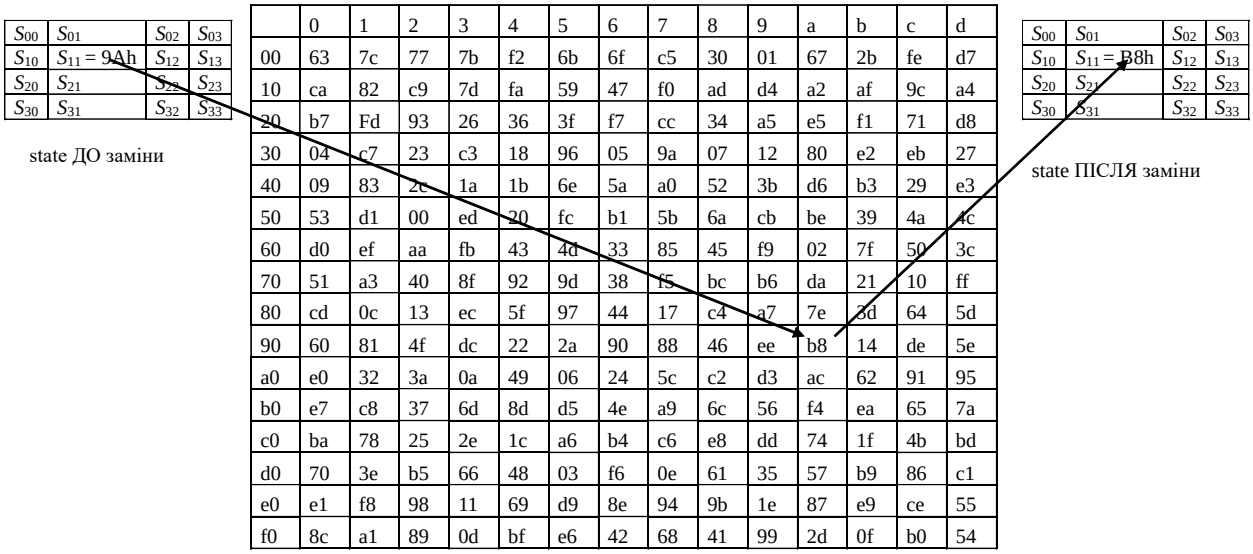

Рис. 11. Приклад заміни SubBytes

Перетворення ShiftRows

У перетворенні ShiftRows байти в останніх трьох рядках state циклічно зміщуються вліво на різну кількість байтів. Рядок 1 (нумерація рядків починається з нуля, див. рис. 12) зміщується на один байт, рядок 2 – на два байти, рядок 3 – на три байти. На рис. 12 проілюстровано застосування перетворення ShiftRows до state.

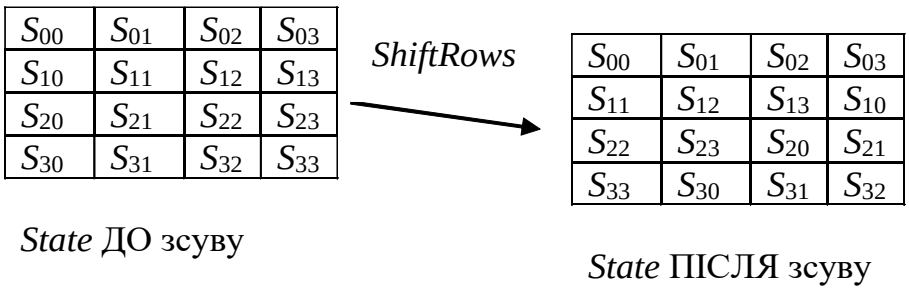

Рис. 12. Операція ShiftRows

Перетворення MixColumns

У перетворенні MixColumns – перемішуванні стовпчика – стовпці state розглядаються як поліноми над полем $F(2^8)$ і множаться за модулем $x^4 + 1$ на сталий поліном:

$$a(x) = 3x^3 + 1 \cdot x^2 + 1 \cdot x^2 + 2.$$

Процес множення поліномів еквівалентний матричному множенню:

$$\begin{bmatrix} S'_{0c} \\ S'_{1c} \\ S'_{2c} \\ S'_{3c} \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \cdot \begin{bmatrix} S_{0c} \\ S_{1c} \\ S_{2c} \\ S_{3c} \end{bmatrix},$$

де c – номер стовпця масиву state і $0 \le c \le 3$. Операція додавання і множення чисел виконується за правилами, описаними у пункті «Математичні основи AES».

В результаті такого множення байти стовпчика c $\{S_{0c}, S_{1c}, S_{2c}, S_{3c}\}$ замінюються відповідно на байти:

$$S'_{0c} = (2 \cdot S_{0c}) \oplus (3 \cdot S_{1c}) \oplus S_{2c} \oplus S_{3c};$$
$$S'_{1c} = S_{0c} \oplus (2 \cdot S_{1c}) \oplus (3 \cdot S_{2c}) \oplus S_{3c};$$
$$S'_{2c} = S_{0c} \oplus S_{1c} \oplus (2 \cdot S_{2c}) \oplus (3 \cdot S_{3c});$$
$$S'_{3c} = (3 \cdot S_{0c}) \oplus S_{1c} \oplus S_{2c} \oplus (2 \cdot S_{3c}).$$

Перетворення застосовується до кожного з чотирьох стовпців state.

Перетворення AddRoundKey

У перетворенні AddRoundKey раундовий ключ RK додається до state за допомогою порозрядного XOR. Кожен раундовий ключ складається з 16 байтів розширеного ключа. Байти раундового ключа записуються в матрицю 4×4, подібну до state. Кожен байт раундового ключа підсумовується з відповідним байтом зі state, як показано на рис. 13.

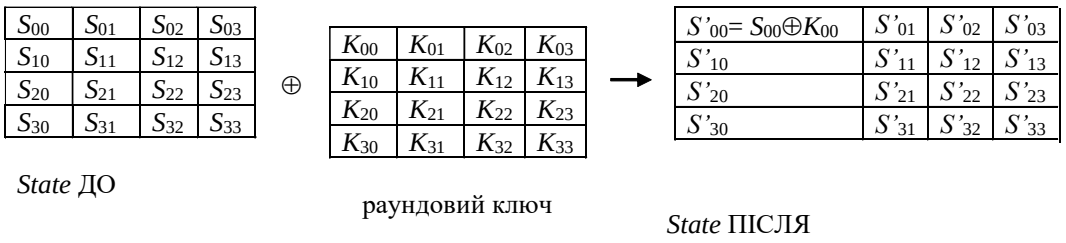

Рис. 13. Додавання раундового ключа

Розширення ключа KeyExpansion

Виконується розширення ключа, щоб створити масив W для раундових ключів. Розширений ключ W складається зі слів (4 байти на слово), що позначаються нижче як w_i, де i лежить у діапазоні [0 ... 44]. Повна довжина розширеного ключа дорівнює 1 408 бітам, по 128 бітів на кожен раунд.

Розширений ключ W містить $4 \times (10 + 1)$ слів – початковий ключ у 4 слова та по 4 слова розширеного ключа на кожен з 10-ти раундів.

У процесі розширення ключа використовується масив констант Rcon. Елементи масиву $Rcon$ пронумеровані від 1 до $256 + 3$. Значення елементів масиву визначені так:

$Rcon_1 = 1;$

$Rcon_k = 2 \cdot Rcon_{k-1} = 2k - 1$, для $k = 2,3, \dots 255;$

$Rcon_k = 0$, для $k = 256, 257, 258.$

Розширення ключа можна описати такою послідовністю операцій:

1) 4 слова ключа шифрування докопіюються в перші 4 слова розширеного ключа W: $w_i = k_i$ для $i = 0,1,2,3$.

2) Решта слова розширеного ключа W для $i = 4, 5, \ldots 44$ генеруються так:

– якщо i кратне 4, то $w = \text{SubBytes}(\text{RotByte}(w_{i-1})) \oplus Rcon_{(i/4)}$;

– якщо i не кратне 4, то $w_i = w_{i-4} \oplus w_{i-1}$.

Функція RotByte переставляє чотири байти вихідного слова $\{a_0, a_1, a_2, a_3\}$ за допомогою циклічної перестановки, перетворюючи в слово $\{a_3, a_1, a_2, a_0\}$. Функція SubBytes застосовує до кожного з чотирьох байтів слова заміну по таблиці S-box, як це описано вище.

Вибір раундового ключа

Раундовий ключ RK для раунду k вибирається з розширеного ключа W як слова з w_{4k} по $w_{4(k+1)}$.

5. Розшифрування

Всі перетворення шифрування однозначні, а отже, – мають зворотне перетворення, тобто можуть бути інвертовані і виконані у зворотному порядку, щоб виконати дешифрування для алгоритму AES.

Схема криптоперетворень може бути записана так:

1. Розширення ключа KeyExpansion.
2. 9 раундів з чотирьох кроків кожен.
   2.1. AddRoundKey – підсумовування з раундовим ключем.
   2.2. InvMixColumns – зворотна перестановка стовпців state.
   2.3. InvShiftRows – зворотний циклічний зсув рядків state.
   2.4. InvSubBytes – зворотна заміна байтів state по таблиці замін.
3. Заключний 10-й раунд.
   3.1. AddRoundKey – підсумовування з раундовим ключем.
   3.2. InvShiftRows – зворотний циклічний зсув рядків state.
   3.3. InvSubBytes – зворотна заміна байтів state по таблиці замін.

Далі докладно описано кожне з перетворень.

Перетворення InvMixColumns

Перетворення InvMixColumns є зворотним для перетворення MixColumns. У перетворенні InvMixColumns стовпці стану (state) розглядаються як поліноми над полем $F(2^8)$ і множаться за модулем $x^4 + 1$ зі сталим поліномом $d(x) = a^{-1}(x)$. У полі $F(2^8)$:

$$d(x) = 0Bh \cdot x^3 + 0Dh \cdot x^2 + 09h \cdot x + 0Eh.$$

Перетворення InvShiftRows

Перетворення InvShiftRows зворотне до перетворенnz ShiftRows. Байти останніх трьох рядів масиву state циклічно зсуваються вправо. Рядок 1 (нумерація з нуля) зміщується на 1 байт, рядок 2 – на 2 байти, рядок 3 – на 3 байти.

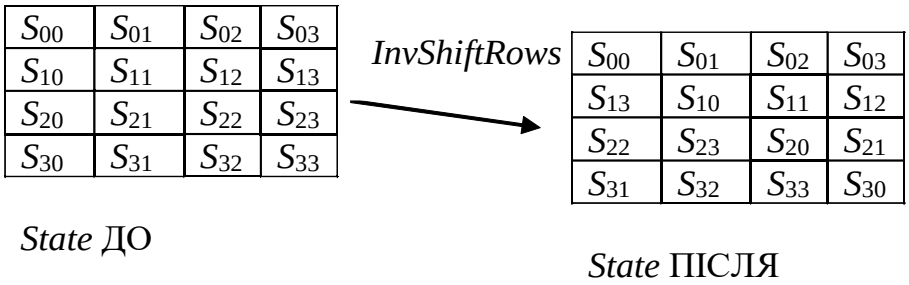

Рис. 14. Перетворення InvShiftRows

Перетворення InvSubBytes

Перетворення InvSubBytes виконує зворотну заміну байтів за допомогою зворотної таблиці замін, назвемо її InvS-box. Таблиця замін наведена нижче.

Таблиця 11. Таблиця InvS-*box* для зворотної заміни байтів

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

Перетворення зворотне AddRoundKey

Перетворення AddRoundKey вже описане у попередніх розділах, це перетворення є зворотним для самого себе, оскільки використовує операцію XOR.

Питання для самоконтролю

1. Які умови висувались до розробки AES?
2. Які основні функції використовуються в алгоритмі AES?
3. Опишіть підготовку вхідного потоку (довжину блоку, розбиття на байти, порядок бітів у байті) в AES.
4. Над яким полем проводяться основні математичні операції алгоритму AES?
5. Що являє собою матриця форми (матриця state)?
6. У чому полягає зміст операції SubBytes?
7. Які сталі поліноми використовуються в MixColumns8?
8. Як формуються раундові ключі?
9. Як відбувається розшифрування повідомлення? Наведіть порядок операцій.

ТЕМА 10. ДЕРЖАВНИЙ СТАНДАРТ ШИФРУВАННЯ, ШИФР «КАЛИНА» (ЛЕКЦІЯ 13)

1. Передумови виникнення.
2. Особливості архітектури.
3. Блок-схема шифру.

1. Передумови виникнення

В Україні з 1990 р. як основний шифр використовувався ГОСТ 28147-89 (або ДСТУ ГОСТ 28147:2009). Шифр забезпечував практичну стійкість, але для нього були розроблені теоретичні методи криптоаналізу, зі складністю значно меншою, ніж повний перебір ключів. З погляду продуктивності, ГОСТ 28147-89 істотно поступається сучасним аналогам, як-от AES, що призводить до ускладнення та подорожчання засобів криптографічного захисту за інших однакових характеристик.

З огляду на позитивний світовий досвід проведення криптографічних конкурсів, Державна служба спеціального зв'язку та захисту інформації України у 2007–2010 рр. успішно провела національний відкритий конкурс, за результатами якого було відзначено алгоритм «Калина», який став, після додаткових досліджень, основою національного стандарту.

Завдання розробки було складним і суперечливим. З одного боку, новий стандарт криптографічного перетворення має забезпечувати високий рівень стійкості із достатнім запасом для застосування протягом кількох десятиліть. З іншого, порівняння нового стандарту виконувалось не лише із застарілим ГОСТ 28147-89, а й з AES, в якому вже було досягнуто низки екстремальних показників швидкодії, а деякі компромісні рішення призвели до низки теоретичних атак.

2. Особливості архітектури

Під час розробки національного стандарту ДСТУ 7624:2014 (блоковий шифр «Калина» та режими його роботи) було прийнято рішення забезпечити прозорість проєктування та використовувати консервативний підхід із застосуванням добре досліджених конструкцій, які забезпечують запас стійкості для безпечного застосування алгоритму в умовах суттєвого прогресу криптоаналітичних технік та засобів обробки даних.

Національний стандарт підтримує розмір блоку і довжину ключа шифрування 128, 256 і 512 бітів (довжина ключа дорівнює розміру блоку або в два рази перевищує його), забезпечуючи нормальний, високий та надвисокий рівень стійкості. Різні варіанти забезпечують гнучкість вибору параметрів для розробників систем криптографічного захисту, що дає змогу отримати найвищий рівень швидкодії та найбільший запас стійкості перетворення.

Високорівнева конструкція використовує добре досліджену Square-подібну SPN-структуру, що застосовується в алгоритмах AES/Rijndael, Whirlpool, «Стрибог»,

«Kuznechik» та багатьох інших. Циклове перетворення побудовано на базі таблиць підстановки (S-блоків) та множення на МДР-матрицю над скінченним полем, що забезпечує необхідні криптографічні властивості. Застосування саме такої конструкції дає змогу забезпечити доведену стійкість до диференційного, лінійного та інших видів криптоаналізу, одночасно забезпечуючи ефективну реалізацію на широкому спектрі програмних і програмно-апаратних платформ. Під час вибору розміру МДР-матриці було взято до уваги розмір кешу L1 сучасних та перспективних на момент розробки алгоритму процесорів, що дало змогу оптимізувати швидкодію програмної реалізації шифру.

Альтернативний варіант, який розглядався під час розробки раундової функції – це ARX-перетворення (Addition-Rotation-XOR). Цей підхід реалізований у шифрі SPECK (розроблений Агентством національної безпеки США та переданий у відкритий доступ), у блокових алгоритмах, на основі яких побудоване сімейство хеш-функцій SHA-0,1,2 та ін. Перевагою підходу є компактність і швидкодія перетворення. Але водночас є й суттєвий недолік, пов'язаний із відсутністю методів, що дають змогу виконати суворе аналітичне обґрунтування криптографічної стійкості таких рішень.

У якості схеми отримання та перетворення ключів було запропоновано нову конструкцію з такими властивостями:

– забезпечення криптографічної стійкості до відомих методів аналізу; відсутність «слабких» ключів, які можуть погіршити властивості перетворення;

– зручність програмної та програмно-апаратної реалізації (для формування циклових ключів застосовуються тільки операції, що використовуються під час шифрування);

– висока обчислювальна складність відновлення ключа шифрування за одним або декількома раундовими ключами.

Остання властивість забезпечує додатковий захист до атак на реалізацію, коли зловмисник намагається атакувати інженерні рішення. Ця особливість є істотною перевагою для низки додатків, зокрема під час реалізації шифрування на смарт-картах, USB-токенах та ін., коли ключ прошитий у пристрої і має бути захищений від зовнішнього доступу (наприклад, у модулях доступу до платних цифрових ТБ-каналів та ін.).

Кількість раундів шифрування залежить від довжини ключа: 10 циклів для 128-бітового, 14 циклів для 256-бітового та 18 циклів для 512-бітового ключа шифрування.

Порівняно з іншими алгоритмами на основі Square-подібної SPN-структури, блоковий шифр «Калина» має такі суттєві конструктивні відмінності:

– початкове та кінцеве забілювання з використанням додавання за модулем 264 для підвищення складності криптоаналітичних атак;

– застосування чотирьох різних S-блоків (замість одного) з властивостями для захисту від атак алгебри, і під час порівняння характеристик з іншими блоковими

і потоковими шифрами забезпечують найбільшу нелінійність булевих функцій (104), що дає додатковий запас стійкості перетворення;

– нову односпрямовану схему формування циклових ключів, що забезпечує захист від атак, ефективність програмної та програмно-апаратної реалізації, разом із додатковою стійкістю до методів аналізу спеціального виду.

Оцінка криптографічної стійкості до диференційного, лінійного, алгебраїчного, інтегрального та інших методів аналізу (практичний критерій) показала, що шифр є стійким при 6 циклах для 128-бітового блоку, 7 циклах для 256-бітового і 9 циклах для 512-бітового блоку, цикл забезпечує експоненціальне зростання складності криптоаналізу). Отже, шифр, що містить 10, 14 і 18 циклів для блоку 128, 256 і 512 бітів відповідно, забезпечує захист від розглянутих видів аналізу і має суттєвий запас стійкості.

3. Блок-схема шифру

Блок-схема алгоритму може здатися схожою на AES. Але у шифрі «Калина» попередньої підготовки потоку як такої немає, що дало змогу збільшити швидкодію алгоритму. Потік розбивається на блоки, до блоку додається раундовий ключ за модулем 2, далі за допомогою таблиці S-box формується state-форма (нелінійна операція SubBytes). Виконується зсув рядків (ShiftRows) та перестановка стовпчиків (MixColumns являє собою, по суті, множення на фіксовану матрицю над полем $GF(2^8)$), сформованого незвідним поліномом 8-го порядку. Також виконується додавання ключа за модулем 2^{64}.

Перетворення зашифрування описується функцією, що складається з низки перетворень, загального вигляду:

$$T_{l,k}^{(K)} = \eta_l^{(K_t)} \circ \psi_l \circ \tau_l \circ \pi_l' \prod_{\nu=1}^{t-1} \left(\kappa^{(K_\nu)} \circ \psi_l \circ \tau_l \circ \pi_l'\right) \circ \eta_l^{(K_0)}.$$

Перетворення $\eta_l^{(K_0)}$ – це додавання до матриці стану раундового ключа за модулем 2^{64}.

За блок-схемою на рис. 16 відбувається розгортання ключів. Передбачається використання описаних у специфікації алгоритму констант та циклічних зсувів.

Стандартні режими роботи алгоритму визначаються ISO/IEC 10116:2006, основними з яких є режими ECB, CBC, CFB, OFB, CTR, а також режим імітовставки CMAC.

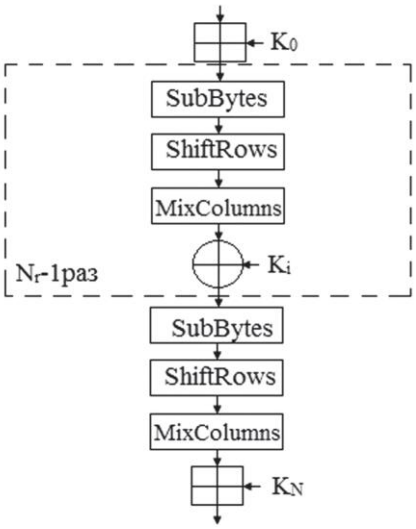

Рис. 15. Схема шифру «Калина»

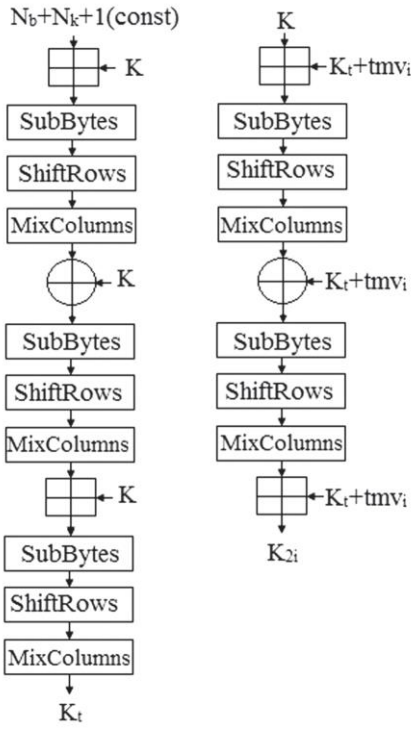

Рис. 16. Схема розгортання ключів шифру «Калина»

Питання для самоконтролю

1. Який стандарт ДСТУ описує алгоритм блокового шифрування та можливі режими його роботи?

2. Які розміри ключа та вхідного блоку підтримує національний стандарт із шифрування?

3. Які відмінності має «Калина» у порівнянні з більшістю шифрів Square-архітектури?

4. Наведіть блок-схему алгоритму.

5. Наведіть блок-схему генерації ключів.

ТЕМА 11. ПОТОКОВІ ШИФРИ (ЛЕКЦІЯ 14)

1. Загальна характеристика потокових шифрів.
2. Шифри з самосинхронізацією.
3. Синхронні шифри.
4. Генератори псевдовипадкових послідовностей, основні принципи.
5. Алгоритми RC4 та A5.

1. Загальна характеристика потокових шифрів

Потокові шифри, по суті, є шифрами гамування і перетворюють звичайний текст у шифр біт за бітом, потоком. У найбільш простому вигляді їх можна описати так: генератор послідовностей ключів виробляє біти гами шифру $k_1, k_2, \ldots, k_i, \ldots$. Ця ключова послідовність додається за модулем 2 з послідовністю вихідних бітів $p_1, p_2 \ldots, p_i, \ldots$ для отримання шифротексту c_i:

$$c_i = p_i \oplus k_i.$$

Отримувач повідомлення додає шифротекст за модулем 2 до ідентичної гами для отримання вихідного тексту:

$$c_i \oplus k_i = p_i \oplus k_i \oplus k_i = p_i.$$

Стійкість системи повністю залежить від внутрішньої структури генератора послідовностей ключів. Якщо генератор виробляє послідовність із невеликим періодом, то стійкість системи буде невеликою. Навпаки, якщо генератор виробляє нескінченну послідовність дійсно випадкових бітів, то ми отримуємо «одноразову стрічку» з ідеальною стійкістю.

На практиці для генерування ключової послідовності використовуються генератори псевдовипадкових послідовностей.

2. Шифри із самосинхронізацією

У 1946 р. основна ідея шифрів із самосинхронізацією (або Cipher-Text Auto Key (CTAK)) була запатентована в США. Основна ідея полягає в тому, що внутрішній стан генератора є функцією фіксованої кількості попередніх бітів зашифрованого потоку. Оскільки внутрішній стан залежить тільки від бітів p шифротексту, генератор на приймальній стороні синхронізується з передавальною стороною після отримання p-біта.

Реалізація цього підходу полягає в такому. На початку кожного повідомлення є випадковий заголовок довжиною n бітів. Цей заголовок шифрується і відправляється в лінію. На приймаючій стороні заголовок розшифровується. Перший результат дешифрування буде неправильним, але після обробки n бітів заголовка обидва генератори будуть синхронізовані.

Недоліком цієї системи є поширення помилок. Якщо один біт містить помилку, генератор на іншій стороні виведе n неправильних бітів послідовності ключів,

поки помилковий біт не буде витіснений з пам’яті, що призведе до помилкового дешифрування n бітів вхідного тексту.

До того ж такі шифри вразливі до атак «відтворення». Зловмисник пише певну кількість бітів шифротексту. Потім, пізніше, він замінює біти трафіка на записані, тобто «відтворює» їх. Після певної кількості «сміття», поки приймальна сторона не буде синхронізована, старий шифротекст буде розшифровуватись нормально.

Шифри із самосинхронізацією, хоч і є потоковими, можуть бути реалізованими як блокові шифри, що використовуються в режимі зворотного зв’язку за зашифрованим текстом. Під час цьому за один раз може шифруватись довільна кількість бітів, що є меншою або дорівнює довжині блоку.

Найпростіші шифрувальні пристрої із самосинхронізацією і шифруванням з використанням генератора псевдовипадкових ключів можуть бути реалізовані на основі N-розрядного регістру зсуву з лінійним зворотним зв’язком – LFSR (Linear Feedback Shift Register) або РЗЛЗЗ. Ці пристрої називаються скремблерами, а сам процес перетворення – скремблюванням.

На рис. 17 показаний приклад шифрування двійкової послідовності 11100101010110 з використанням гами, сформованої 4-розрядним LFSR, за початкового стану, що дорівнює 1001. Зашифрована послідовність має вигляд 01001010010010.

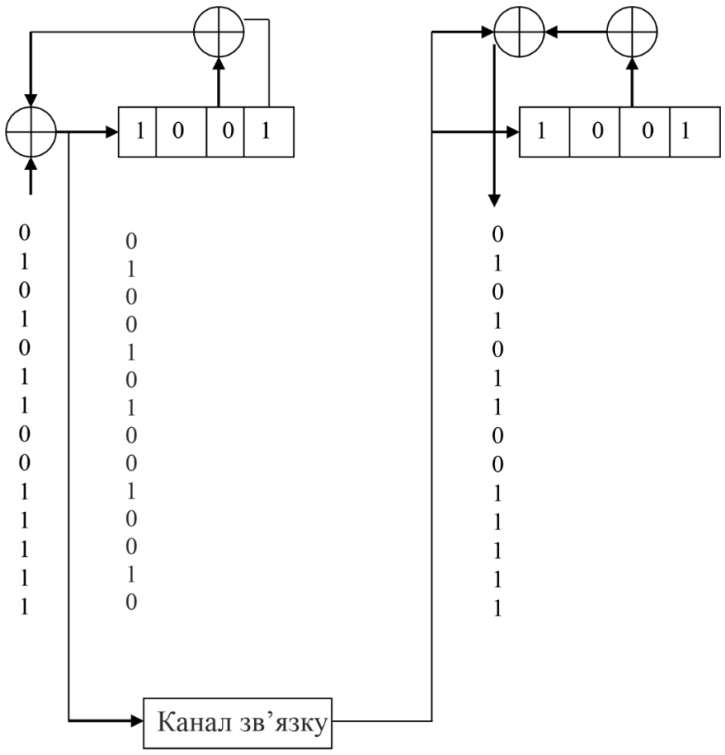

Рис. 17. Потокове шифрування із самосинхронізацією з LFSR

3. Синхронні шифри

У синхронних потокових шифрах гама формується незалежно від вхідної послідовності, кожен елемент (біт, символ, байт тощо) якої у такий спосіб шифрується незалежно від інших елементів (рис. 18).

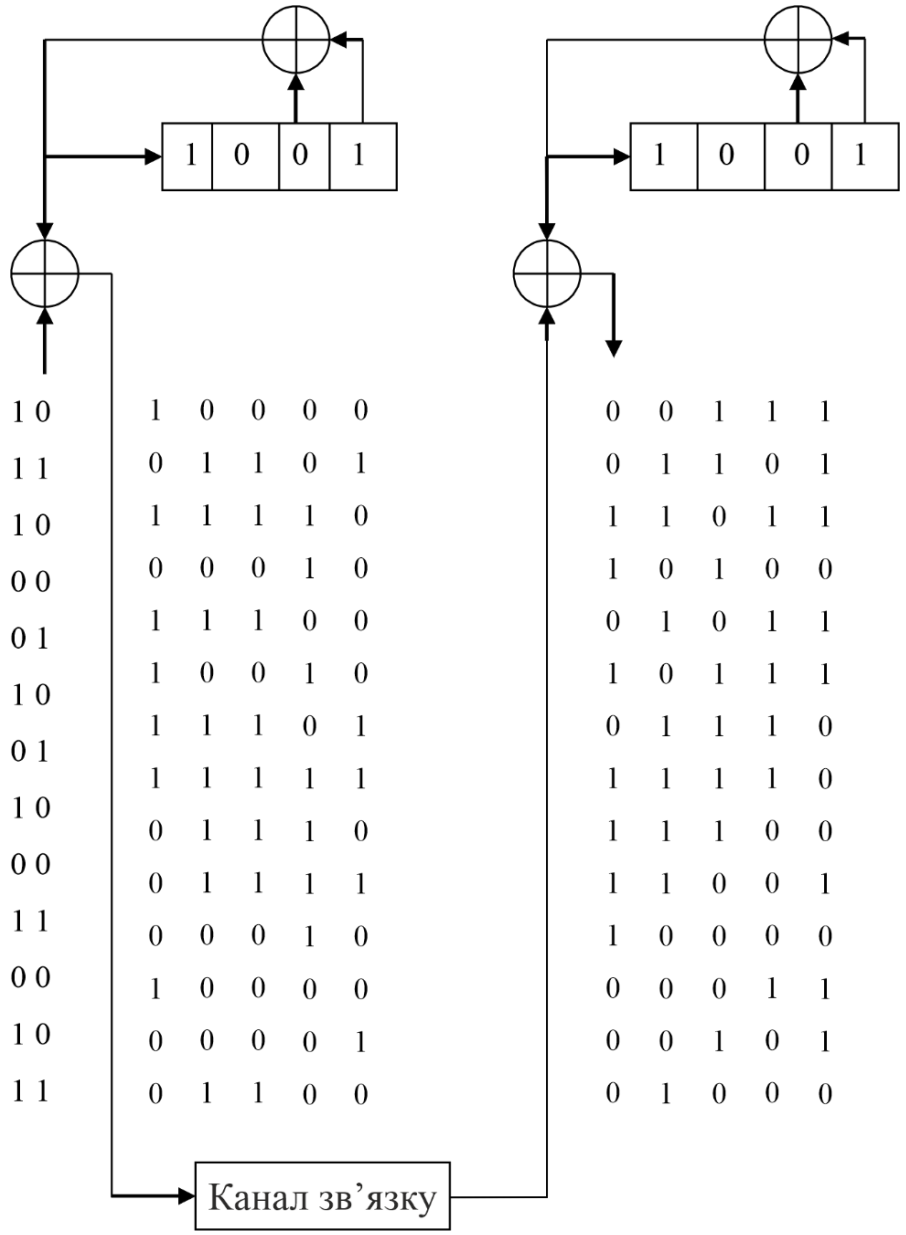

Рис. 18. Синхронне потокове шифрування з LFSR

Основною складністю такого підходу є необхідність синхронізації ключових генераторів на передавальній і приймальній сторонах.

Позитивною властивістю шифрів синхронного потоку є відсутність ефекту поширення помилок. Один заявлений біт під час транзиту спотворить лише один біт тексту під час розшифрування.

Синхронні шифри також захищають від вставок або вилучень сегментів шифротексту з потоку.

Однак такі шифри є вразливими для зміни окремих бітів. Якщо зловмисник знає початковий вхідний текст, він може змінити біти в потоці шифротексту, що шифр буде розшифрований так, як потрібно зловмиснику.

4. Генератори псевдовипадкових послідовностей: основні принципи побудови

Для генерації ключової інформації та/або гами необхідні алгоритми, основані на псевдовипадкових послідовностях (ПВП) спеціального типу.

Загальною вимогою до них є наближення статистичних властивостей ПВП до властивостей теоретичних випадкових послідовностей. Іншою вимогою є вимога щодо неможливості відновлення секретних параметрів алгоритму, а також неможливості продовження відрізків ПВП у довільному напрямку, без знання початкового стану генератора.

Такі генератори називають криптографічно стійкими, або криптографічними генераторами ПВП.

Для побудови криптографічних генераторів часто використовуються функції, що за властивостями наближаються до односпрямованих, у тому числі функції, що реалізуються схемами криптоалгоритмів, наприклад, блокових шифрів.

Криптографічно стійкі генератори ПВП з використанням блокових шифрів можуть формуватися:

- на основі шифрування детермінованої послідовності;
- на основі шифрування послідовності з випадковим параметром;
- на основі поліпшення (модифікації) вихідної послідовності шифру;
- комбінуванням згаданих методів.

Наприклад, блоковий шифр у режимі лічильника перешифровує на ключі K послідовність блоків виду $B_{i+1} = E_K[B_i + 1]$, де B_0 – довільно обраний (несекретний) блок. Вихідна послідовність складається з блоків $B_2, B_3, ...$, секретність основана на секретності ключа.

Для генерації сеансових ключів, рандомізаторів або псевдовипадкових чисел зручно використовувати стандарт ANSI X9.17, який оснований на перешифруванні послідовності з випадковим параметром. Іншим прикладом стандарту генераторів ПВП є FIPS-186.

5. Алгоритми RC4 та A5

Класичним прикладом потокового шифру є алгоритм RC4. Це шифр змінної довжини ключа, розроблений у 1987 р. Роном Рейвестом для RSA Data Security, Inc. З 1994 р. шифр став доступним для незалежного аналізу.

Алгоритм працює в режимі OFB. Ключова послідовність не залежить від вихідного тексту. Структура алгоритму включає блок заміни розмірами 8×8 з

елементами $S_0, S_1 \ldots, S_{255}$. Блок заміни – це змінна довжина ключа, залежна від перестановки чисел $0,1, \ldots, 255$. Є два лічильники i і j, початкове значення яких дорівнює 0. Для створення псевдовипадкового байта виконуються такі дії:

$$i = (i + 1) \bmod 256,$$
$$j = (j + S_i) \bmod 256.$$

Далі виконується така перестановка S_i і S_j:

$$t = (S_i + S_j) \bmod 256,$$
$$k = S_t.$$

Потім байт k додається за модулем 2 до байта вхідного коду для отримання зашифрованого байта.

Ініціалізувати блок заміни також просто. Спочатку він заповнюється лінійно: $S_0 = 0, S_1 = 1$, ..., $S_{255} = 255$. Потім ключ заповнюється ще одним 256-байтовим масивом, водночас ключ можна повторити необхідну кількість разів для заповнення всього масиву: $k_0, k_1, \ldots, k_{255}$. Лічильник j встановлюється на 0. Після цього виконуються такі дії:

$$j = (j + k_i + S_i) \bmod 256$$

для всіх i від 0 до 255. Після цього переставляються місцями S_i і S_j.

Шифрування за допомогою цього алгоритму є приблизно у 10 разів швидшим за шифрування за алгоритмом DES.

Одним з найбільш відомих серед потокових алгоритмів є алгоритм (точніше, сімейство алгоритмів) A5.

A5 – це потоковий шифр, що використовується в системах GSM (Group Special Mobile) для шифрування зв'язку між абонентом і базовою станцією.

A5 використовує три LFSRs довжиною 19, 22 і 23 з модифікованими скороченими поліномами зворотного зв'язку, тобто з поліномами, що мають невелику кількість ненульових коефіцієнтів. Вихід гама-генератора – це вихід із додаванням за модулем 2. На вхід генератора отримуються послідовності з виходів трьох LFSR, початкове заповнення яких визначається секретним ключем.

Для управління синхронізацією використовуються $C1$, $C2$ і $C3$ біти з виходів LFSR. У кожному такті відбувається зсув не менше двох LFSR. Якщо $C1 = C2 = C3$, то зсуваються всі три регістри, інакше – зсуваються два регістри i і j, для яких виконується $c_i = c_j$.

Існує кілька версій алгоритму A5: основна версія A5/1, навмисно ослаблена версія алгоритму A5 – алгоритм A5/2 і більш сучасна версія A5/3 (алгоритм Касумі). Зупинимося докладніше на версії A5/1.

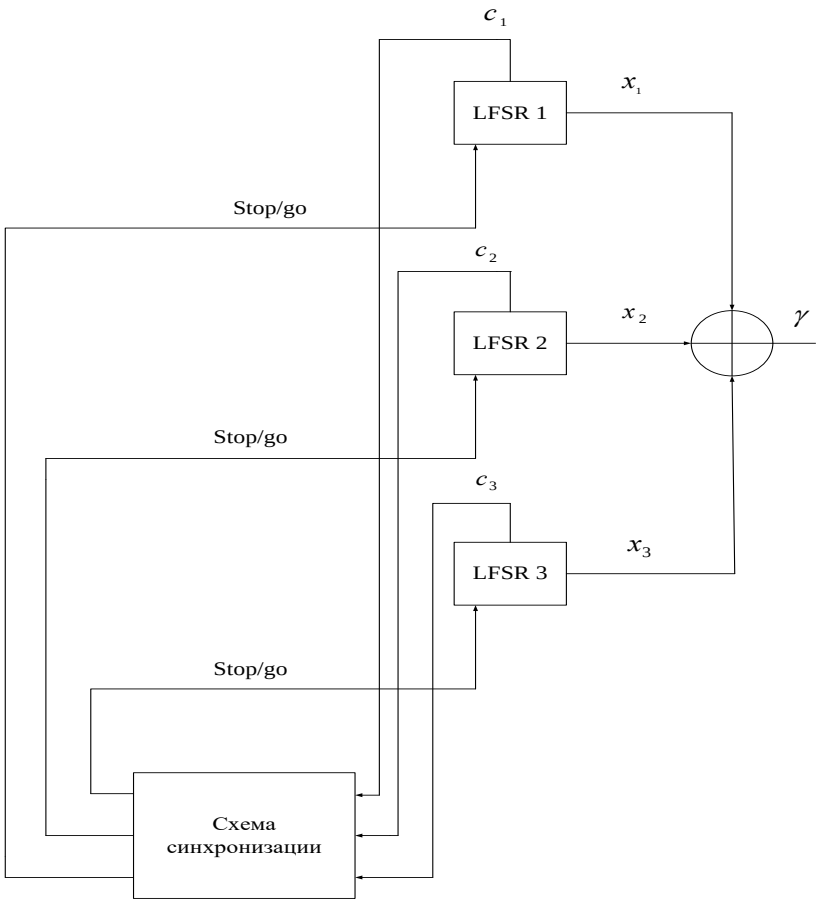

Рис. 19. Гама-генератор алгоритму А5

Кожен кадр шифрується за допомогою секретного ключа шифрування Ks і прохідного порядкового номера наступного кадру. Генератор ПВП А5/1 складається з трьох коротких LFSRs , позначених як LFSR-l, LFSR-2 і LFSR-3.

Твірні поліноми цих регістрів мають форму:

LFSR-1: $x^{19} + x^{18} + x^{17} + x^{14} + 1$;

LFSR-2: $x^{22} + x^{21} + 1$;

LFSR-3: $x^{23} + x^{22} + x^{21} + x^8 + 1$.

Вихідні біти знімаються зі старших розрядів регістрів, а потім застосовується XOR над вихідними бітами всіх трьох бітів гами шифру. Регістри працюють за принципом stop-and-go, що забезпечується за допомогою спеціальної функції **majority,** яка вводиться значеннями регістрових бітів: біт $C1$ (восьма цифра) для LFSR-1, біт $C2$ (десята цифра) для LFSR -2 і $C3$ (десята цифра) для LFSR-3.

Функція **majority** виглядає так:

$$\text{majority}(x_1, x_2, x_3) = x_1x_2 + x_1x_3 + x_2x_3.$$

На кожному кроці шифру зміщуються два або три регістри. Отже, кожен регістр зміщується в один такт алгоритму зі ймовірністю $3/4$ і не рухається зі ймовірністю $1/4$.

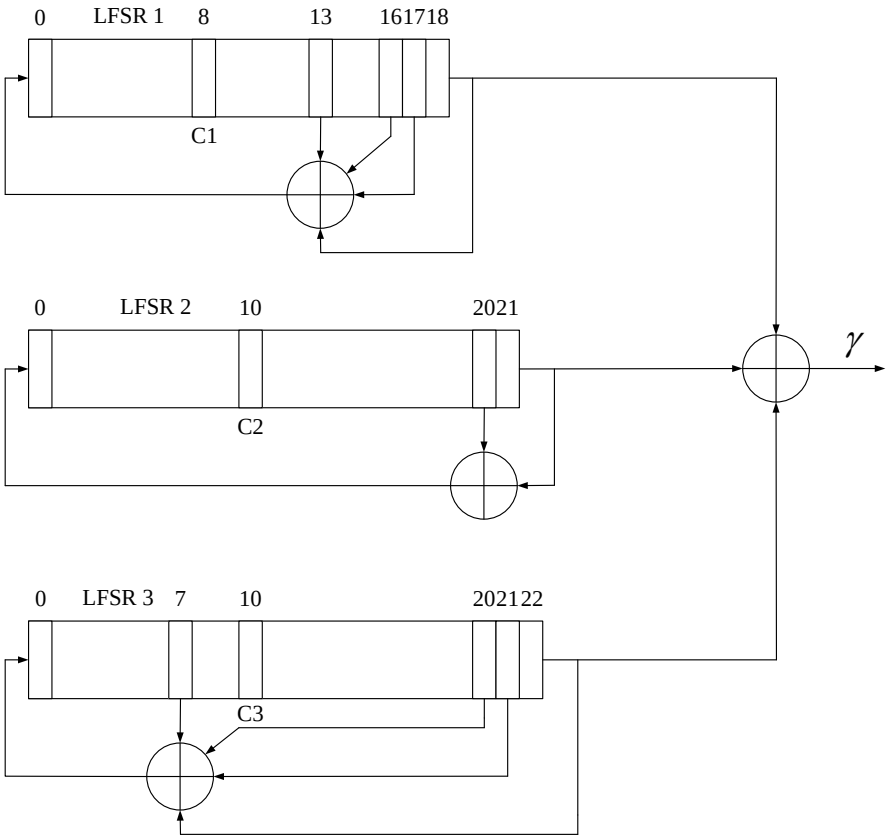

Рис. 20. Схема роботи регістрів алгоритму A5/1

Питання для самоконтролю

1. Які шифри називають потоковими? Де вони використовуються?
2. Опишіть приклад шифру із самосинхронізацією.
3. Наведіть приклад синхронного шифру.
4. Де в криптографії використовуються генератори ПВП?
5. Які вимоги висуваються до криптографічних генераторів ПВП?
6. Яка основна сфера використання алгоритмів A5?

ДОДАТОК А

ПОСТКВАНТОВІ СИСТЕМИ ШИФРУВАННЯ ТА СИМЕТРИЧНІ КРИПТОСИСТЕМИ

Зараз існує дуже багато побоювань щодо нових технічних можливостей для розкриття шифросистем. Найбільше обговорюють розвиток квантових комп'ютерів, і це не випадково, бо саме для квантових алгоритмів математично доведено можливість розв'язання задачі факторизації (розкладання на множники) добутку двох простих чисел та задачі виконання перетворення Фур'є з набагато більшою швидкістю, ніж на класичних комп'ютерах. А саме на складності виконання таких операцій і побудовано багато систем шифрування, переважно асиметричних.

У повідомленні компанії OffSec (https://www.offsec.com/blog/post-quantum-cryptography/) йдеться про сучасний стан (грудень 2024 р.) цієї проблеми. Далі наведемо основні висновки, які можна зробити з того факту, що NIST прийняв перші стандарти постквантового (тобто такого, яке можна використовувати в умовах майбутнього застосування кантових комп'ютерів) шифрування, та навіщо це робиться.

Уявіть, що зараз 2040 р., і в мережі з'явиться перший квантовий комп'ютер, здатний здійснювати справжні криптографічні атаки. Уряди, компанії та інші особи чи організації можуть мати дані з минулого, захищені доквантовим (класичним комп'ютерним) шифруванням, і деякі з цих даних все ще можуть мати певну цінність для когось у 2040 р. Якщо зібрати класично захищені дані, пізніше, якщо можливо, проти нього використовуються квантові криптографічні атаки, і зловмисник може отримати доступ до секретів.

Цей сценарій називається атакою «збирай зараз, розшифруй пізніше». Якщо дані будуть зібрані, наприклад, з доступу, отриманого внаслідок іншої атаки, навіть якщо ми не зможемо атакувати конфіденційність даних за допомогою сучасних (класичних) комп'ютерів у 2024 р., ми зможемо атакувати їх за допомогою квантових комп'ютерів у майбутньому.

Сучасні квантові комп'ютери вдосконалюються і можуть виконувати певні обчислення. Факторизація цілих чисел – це завдання, яке можна виконати більш ефективно за допомогою квантових комп'ютерів, порівняно з класичними комп'ютерами. З удосконаленням квантових комп'ютерів зростає їх здатність вирішувати великі проблеми факторизації, що ставить деякі класичні криптографічні алгоритми під загрозу зламу або ослаблення.

І алгоритми RSA, і EC використовують цілу факторизацію великих чисел як частину складних математичних завдань, що робить їх безпечними. RSA широко використовується для ідентифікації, тоді як EC використовується як для ідентифікації, так і для обміну ключами Діффі–Хеллмана в сучасних системах.

Тому 2 серпня 2016 р. NIST оголосив про запит на подання алгоритмів шифрування, які обробляють відомі теоретичні майбутні квантові атаки. Тепер, у 2024 р., 13 серпня NIST опублікував перший набір стандартів постквантової криптографії (PQC).

Це стандарти FIPS 203, 204 і 205. Цю групу стандартів і алгоритмів разом із FIPS 206, який незабаром з'явиться, можна використовувати, щоб підготувати нас до майбутніх теоретичних атак квантових комп'ютерів. Через вектор атаки «збирай зараз, розшифровуй пізніше» виникає відчуття терміновості для кращого захисту конфіденційних довготривалих секретів.

Механізм інкапсуляції ключів модульної решітки (ML-KEM) – це техніка, яку ми можемо використовувати для захисту ефемерних секретів TLS від атак типу «зберіть зараз, розшифруйте пізніше». Сьогодні ми можемо застосувати його в гібридній криптографічній системі за допомогою розширення TLSv1.3. Модульно-решіткова криптографія ML-KEM базується на математичній задачі «Навчання з помилками» (LWE), застосованій до геометричних структур решітки. Простий спосіб уявити собі це полягає в тому, що точки на решітках є відкритим і закритим ключами, а зв'язок між точками є частиною проблеми LWE. Потім ми використовуємо ці ключі для шифрування інших відкритих ключів, які використовуються для отримання загального секрету, або іншим способом безпосередньо інкапсулюємо спільний секрет. Тоді спільний секрет є ключем, який використовується в симетричному шифрі, такому як-от AES256, для шифрування фактичних даних.

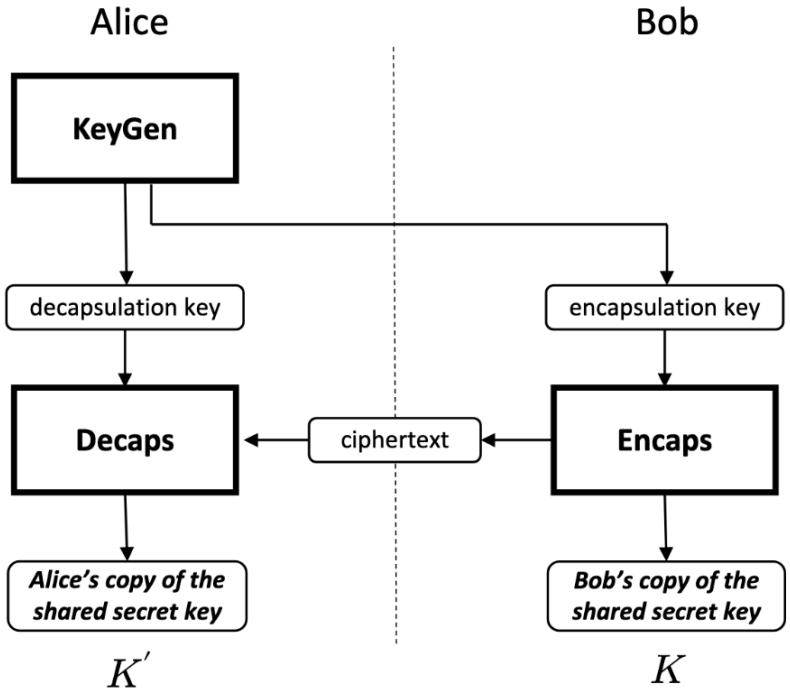

Рис. Д1. Спрощена схема заснування ключів із використанням методу KEM

Модульно-решітковий алгоритм цифрового підпису (ML-DSA) є підходом до цифрових підписів. Ми можемо використовувати Dilithium (FIPS 204 – ML-DSA (CRYSTALS-Dilithium – відповідний стандарт NIST)) для створення підписів, які здатні витримати класичні та квантові криптографічні атаки. Сертифікати PQC TLS (HTTPS) можуть використовувати Dilithium для криптографічної ідентифікації. Модульно-решіткова криптографія ML-DSA базується на підході «Fiat-Shamir with Aborts».

Цей підхід Fiat-Shamir with Aborts створює неінтерактивну перевірку за допомогою хеш-функції, і вона перериває операцію, якщо використовується недійсний матеріал. Це допомагає підвищити безпеку системи та може зменшити використання мережі чи інших ресурсів.

Деякі класичні алгоритми можуть витримати атаки

Не всі класичні шифри здаються зламаними в квантовому світі, але багато хто з них зламаний, і отримані прогалини в безпеці мають широкий теоретичний вплив. У математичної теорії сьогодні є криптографічні компоненти, які, ймовірно, витримають. Багато класичних алгоритмів використовуються в алгоритмах PQC. До них належать симетричне шифрування AES256 і розширені функції виведення SHAKE (XOF). SHAKE є частиною сімейства функцій SHA-3 (див. також FIPS 202, щоб дізнатися більше про SHAKE). TLS може використовувати AES256, але головною проблемою на стороні TLS є EC, який використовується в обміні ефемерними ключами, тобто як виводиться спільний секрет. Тоді цей спільний секрет є симетричним секретним ключем у шифруванні AES256 корисних даних сеансу. Тож те, що ми маємо AES256, не означає, що ми гарантуємо захист.

Складання плану адаптації PQC

Світу знадобиться багато років, щоб краще протистояти вектору атак «збирай урожай зараз, розшифровуй пізніше». Одна з перших речей, яку ми повинні зробити, – це отримати інвентаризацію всіх наших криптографічних компонентів. Потім ми хочемо виміряти та задокументувати кожен із цих компонентів, не допускаючи витоку та не відкриваючи їх. Вимірювання повинні включати термін придатності або вік матеріалу, а також міцність біта й алгоритми. В ідеалі ми також знаємо, які клієнти чи ділові партнери покладаються на цей матеріал.

Звідси ми можемо визначити пріоритетність списку криптографічних компонентів на основі найбільшої потреби в PQC. TLS, пов’язаний з інтернетом, є однією з областей, яка, ймовірно, переміститься вгору в списку пріоритетів. Це відбувається тому, що у нас працює та доступний гібридний механізм PQC, а дані зашифрованого тексту переміщуються в інтернеті, виходячи за межі нашого контролю. Навіть якщо він не проходить через інтернет, оскільки TLS (HTTPS) прохо-

дить через наші маршрутизатори, брандмауери, проксі-сервери та численні фізичні носії, ризик набагато більший. Системи, як-от медичні, фінансові та людські ресурси, можуть бути пріоритетними через характер ідентифікаційної інформації, яка переміщується між пристроями цих систем. Поточне вирішення цього полягає у використанні гібридного PQC для розшарування класичних і квантових алгоритмів. Це досягається для TLS за допомогою розширення TLSv1.3.

TLSv1.3 розроблено за модульним принципом, тому цей механізм розширення можна використовувати для змішування нового коду з додаванням додаткових шифрів. Тепер, у 2024 р., використовується KEM для захисту EC, а все інше залишається класичним. Існують підходи до гібридного TLSv1.3, деякі з них нагадують FIPS, а інші розходяться, їх приймають і постачають IBM, AWS, Cloudflare та інші. Гібрид використовує кілька методів обміну ключами, тому, якщо EC зламано, інкапсуляція Kyber може захистити його.

Після гібридного TLS ще одним аспектом, який, ймовірно, має вищий пріоритет у списку, є довготривалі ідентифікаційні дані, які, як очікується, триватимуть через 10 років або довше. Ці додаткові довготривалі ідентифікатори можуть бути компонентами PKI, як-от сертифікати підпису центру сертифікації, наприклад, для підпису коду або іншого використання, де кореневий ключ може ніколи не змінюватися. Для цих випадків довготривалої ідентичності сума технологічного боргу набагато різноманітніша, ніж ситуація TLS. Ми можемо використовувати сертифікати ML-DSA для вирішення цих довготривалих ідентифікацій. Щоб усунути цю прогалину в PQC, потрібно буде розробити та прийняти більше програмного забезпечення, у деяких випадках спеціально для інструментів і засобів автоматизації, що використовуються в організації.

Деякі аспекти міграції PQC, ймовірно, займуть майже десять років, і не всі збираються це робити. Є багато складних систем, які є центральними для безпеки, які в ідеалі потрібно замінити до квантової ери. Ймовірно, це буде надзвичайно дорога подорож для всього світу. Але якщо ми добре плануємо, ці витрати повинні бути зменшені.

Crypo Agility

Криптографічна гнучкість – це властивість систем використовувати модульні криптографічні компоненти та ефективно їх змінювати. Багато сучасних систем не є такими.

З моменту прийняття стандартів HTTPS (SSL, потім TLS) ми пережили багато великих криптографічних міграцій. І багатьом організаціям знадобилося 8 років, щоб перейти з TLSv1.0 і SHA1. Міграція PQC буде справді багатьма міграціями в різних сферах, з кількома фазами впровадження та підтримки змішування з класичними алгоритмами.

Щоб заощадити гроші та зменшити ризик, ми використовуємо шаблон проєктування «швидкість криптовалюти». Ми можемо застосувати це до наших програмних систем, вибравши бібліотеки та фреймворки, які дають змогу додавати та змінювати модульні компоненти для криптографічних операцій, без необхідності виконувати повну міграцію системи чи переписувати всі програмні системи одночасно для впровадження нової криптографії.

Ми не хочемо помилково послабити нашу безпеку, зробивши криптографію настільки гнучкою, що зловмисник зможе нею маніпулювати. Натомість ми хочемо скомпілювати наші програми за допомогою модульних бібліотек для керування доступними можливостями та алгоритмами. Звичайно, є ще CI/CD та атаки на ланцюги поставок, які слід розглянути. Вибір і перевірка наших ланцюгів постачання для криптографічних компонентів також буде важливим завданням, оскільки ми плануємо впровадження PQC.

Висновок

Цілочисельна факторизація вважається достатньо великою проблемою для криптографічного захисту від квантових комп'ютерних атак. Це означає, що багато наявних (класичних) криптографічних механізмів мають ризики. RSA і EC знаходяться в групі ризику, тоді як інші можуть мати часткове пошкодження. Більшість класичного сильного симетричного шифрування та хеш-функцій мають деяку теоретичну стійкість до квантових атак, хоча ми не можемо знати напевно. Так само ми не знаємо напевно, чи витримають алгоритми PQC, але вони є найкращими, які ми маємо сьогодні.

Водночас класичні симетричні алгоритми шифрування з достатньо довгими ключами поки вважаються достатньо стійкими для розшифровки квантовими комп'ютерами.

Інкапсуляція ключів у гібридних криптосистемах є, мабуть, найбільш зрілим PQC, доступним для виробничого використання сьогодні з погляду впровадження програмного забезпечення. Наскрізні системи PQC (повне впровадження всіх компонентів) можуть зайняти багато років, щоб втілити їх у реальність. Витрати та ризики високі, тож зараз чудовий час для розумного планування та проєктування, щоб нам було легше вносити ці зміни в майбутньому.

СПИСОК РЕКОМЕНДОВАНОЇ ЛІТЕРАТУРИ

Базова

1. Інформаційна безпека / Ю. Я. Бобало, І. В. Горбатий, М. Д. Кіселичник та ін. Львів: Львівська політехніка, 2019. 580 с.

2. Schneier B. Applied Cryptography. Protocols, Algorithms and Source Code in C. Indianapolis: John Wiley & Sons, 2016. 712 p.

3. Alko R. Meijer Algebra for Cryptologists. Luxembourg: Springer, 2016. 301 p.

4. Rubinstein-Salzedo S. Cryptography. Luxembourg: Springer, 2018. 271 p.

5. Свейгарт Э. Криптографія та злам шифрів на Python. Київ: Діалектика, 2020. 512 с.

6. Щур Н. О. Прикладна криптологія. Методичні рекомендації до виконання лабораторних робіт. Житомир: Державний університет «Житомирська політехніка», 2021. 104 с.

7. Криптологія: опорний конспект лекцій / уклад.: Д. Вербівський, Б. Якимчук. Житомир: Вид-во ЖДУ ім. Івана Франка, 2023. 173 с.

Допоміжна

1. Stinson D. Cryptography: Theory and Practice – Chapman & Hall/CRC, 2002. 360 p.

2. Anderson R. J. Security Engineering: A guide to building dependable distributed systems. John Wiley & Sons Inc, New York, 2008. 1080 p.

3. Mollin Richard A. An Introduction to Cryptography (Discrete Mathematics and Its Applications). Chapman & Hall/CRC, 2000. 392 p.

4. Історія криптології та секретного зв’язку / уклад. В. В. Гребенніков. Київ: Вид. «КНТ», 2023. 800 с.

Навчальне видання

Ткаченко Віра Сергіївна
Рассохіна Юлія Валентинівна
Крижановський Володимир Григорович

ПРИКЛАДНА КРИПТОЛОГІЯ

Конспект лекцій
(друге видання, доповнене)

Редактор О. А. Солдатова
Технічний редактор Т. О. Важеніна-Гопрак

Підписано до друку 09.10.2025.
Формат 60×84/16. Папір офсетний.
Друк – цифровий. Умовн. друк. арк. 4,88.
Тираж 30. Зам. 44.

Донецький національний університет імені Василя Стуса
21021, м. Вінниця, 600-річчя, 21.
Свідоцтво про внесення суб'єкта видавничої справи
до Державного реєстру
серія ДК № 5945 від 15.01.2018