МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ ВИЩИЙ НАВЧАЛЬНИЙ ЗАКЛАД
«УЖГОРОДСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ»
КАФЕДРА КОМП'ЮТЕРНИХ СИСТЕМ І МЕРЕЖ

ЗАХИСТ ІНФОРМАЦІЇ В КОМП'ЮТЕРНИХ СИСТЕМАХ

Підручник
для студентів спеціальності
123 «комп'ютерна інженерія»

Ужгород - 2021

УДК 004.056.55, 003.26

Підручник з курсу «Захист інформації в комп’ютерних системах» призначено для студентів інженерно-технічного факультету ДВНЗ «УжНУ» спеціальності 123-«комп’ютерна інженерія».

Укладачі: Гапак О. М. – канд. пед. наук, доцент кафедри комп’ютерних систем та мереж ДВНЗ «УжНУ», Балога С.І., канд. фіз.-мат. наук, доцент кафедри комп’ютерних систем та мереж ДВНЗ «УжНУ»

Рецензент: Глебена М. І. – канд. фіз.-мат. наук, доцент кафедри системного аналізу та теорії оптимізації ДВНЗ «УжНУ».

Відповідальний за випуск – Горват П.П., канд. фіз.-мат. наук, доцент, завідувач кафедри комп’ютерних систем та мереж ДВНЗ «УжНУ».

Підручник розглянуто та схвалено на засіданні кафедри комп’ютерних систем та мереж, протокол № 11 від 20.05.2021 року та методичної комісії інженерно-технічного факультету протокол №4 від 24.05.2021 року.

видавництво

ЗМІСТ

ВСТУП……5
1. ЗМІСТ БАЗОВИХ ПОНЯТЬ ЗАХИСТУ ІНФОРМАЦІЇ В КОМП'ЮТЕРНИХ СИСТЕМАХ……..6
1.1. Предмет захисту……..6
1.2. Об'єкт захисту……7
1.3. Погрози безпеки інформації в комп'ютерних системах……..7
2. БАЗОВІ СХЕМИ АТАК ТА ОРГАНІЗАЦІЯ КАНАЛІВ ВИТОКУ ІНФОРМАЦІЇ. КЛАСИФІКАЦІЯ ЗЛОВМИСНИКІВ……14
2.1. Класифікація атак.……14
2.2. Базові схеми атак.……..14
2.3. Організація каналів витоку інформації……...15
2.4. Класифікація зловмисників.……16
2.5. Основні напрямки комп'ютерних злочинів.……...18
3. ОРГАНІЗАЦІЯ СИСТЕМИ ЗАХИСТУ ІНФОРМАЦІЇ……...21
3.1. Політика безпеки.……..21
3.2. Побудова системи захисту……...24
3.3. Основні підсистеми комплексу засобів захисту……27
4. МАТЕМАТИЧНІ МОДЕЛІ БЕЗПЕКИ……...29
4.1. Види математичних моделей безпеки.……29
4.2. Моделі дискреційної політики безпеки……..29
4.3. Моделі політики мандатного доступу.……37
5 ОРГАНІЗАЦІЙНО- ТЕХНІЧНІ ЗАДАЧІ ЗАБЕЗПЕЧЕННЯ ЗАХИСТУ ІНФОРМАЦІ……40
5.1. Загальні поняття……40
5.2. Критерії оцінювання захищених комп'ютерних систем……41
5.3. Законодача і нормативна база захисту інформації в України……44
5.4. Міжнародний стандарт ISO/IEC 15408……50
6. ІДЕНТИФІКАЦІЯ ТА АУТЕНТИФІКАЦІЯ……53
6.1. Паролі.……53
6.2. Ідентифікація і перевірка достовірності.……55
6.3. Біометрична ідентифікація й аутентифікація користувача.……..59
6.4. Взаємна перевірка дійсності користувачів.……...61
Тестові завдання для розділів 1-6……...62
7. ОСНОВИ КРИПТОГРАФІЧНОГО ЗАХИСТУ ІНФОРМАЦІЇ……65
7.1. Історія виникнення криптографії.…….....65
7.2. Принципи криптографічного захисту інформації……70
7.3. Основи теорії К. Шеннона……...75
8. КЛАСИЧНІ МЕТОДИ СИМЕТРИЧНОГО ШИФРУВАННЯ.……...77
8.1. Основні поняття і визначення.……..77
8.2. Шифри перестановок.…….....78
8.3. Шифри підстановок…….....81
8.4. Гамування…….....86
8.5. Аналітичні методи…….....87

9. СИМЕТРИЧНІ КРИПТОАЛГОРИТМИ…….....88
9.1. Блокові складені шифри…….....88
9.2. Алгоритм DES…….....90
9.3. Основні режими роботи алгоритму DES…….....98
9.4. Області застосування алгоритму DES…….....98
9.5. Стійкість DES…….....102
9.6. Модифікації DES…….....103
9.7. Алгоритм шифрування IDEA...…….....107
9.8. ГОСТ 28147-89...…….....109
9.9. AES...…….....111
9.10. Стандарти шифрування України…….113
10. ПОТОКОВІ ШИФРИ...…….....…….....119
10.1. Основні відомості...…….....119
10.2. Алгоритм RC4..…….....122
10.3. Потокові шифри на базі регістрів зсуву……...122
11. ГЕНЕРАТОРИ ВИПАДКОВИХ І ПСЕВДОВИПАДКОВИХ ПОСЛІДОВНОСТЕЙ……...……125
11.1. Загальні відомості…….125
11.2. Генератори випадкових послідовностей..……125
11.3. Генератори псевдовипадкових послідовностей..……127
12. АСИМЕТРИЧНІ КРИПТОСИСТЕМИ……..139
12.1. Основні поняття…….139
12.2. Елементи теорії чисел..……..141
12.3. Алгоритм RSA..……..143
12.4. Схема шифрування Ель-Гамаля……..145
13. ЕЛЕКТРОННИЙ ЦИФРОВИЙ ПІДПИС……...146
13.1. Проблема аутентифікації даних і електронний цифровий підпис…..146
13.2. Хеш-функція…….147
13.3. Алгоритми електронного цифрового підпису……...151
14. УПРАВЛІННЯ КЛЮЧАМИ……158
14.1. Генерування ключів..…….158
14.2. Накопичення ключів...……..159
14.3. Розподіл ключів..……...160
14.4. Сертифікати ключів..……161
14.5. Відкритий розподіл ключів Діффі-Хеллмана..……...166
15. ОСНОВИ КРИПТОАНАЛІЗУ……169
15.1. Основні принципи криптоаналізу……170
15.2. Універсальні методи криптозахисту……..172
15.3. Первинний аналіз шифровки……173
15.4. Конкретні приклади криптоаналізу……..174
Тестові завдання для розділів 7-15…….179
ЛІТЕРАТУРА 184

ВСТУП

Задача захисту інформації в комп'ютерних системах на сьогоднішній день є актуальною внаслідок широкої розповсюдженості таких систем, розширення комп'ютерних мереж, якими передаються великі обсяги інформації. Забезпечення безпечної діяльності комп'ютерних систем необхідне для будь-яких підприємств і установ, починаючи від державних організацій і закінчуючи невеликими приватними фірмами, незалежно від виду їх діяльності. Розходження полягає лише в засобах, методах та в обсязі забезпечення безпеки.

Цей підручник призначений для студентів спеціальності «комп'ютерна інженерія». В ньому відображено основні розділи нормативного курсу «Захист інформації в комп'ютерних системах»: основні засади інформаційної безпеки, класифікація та принципи побудови систем захисту, методи і засоби захисту інформації в комп'ютерних системах.

Для освоення даної дисципліни необхідні знанні із курсів: «Програмування», «Системне програмування», «Дискретна математика», «Лінійна алгебра та аналітична геометрія», «Теорія інформації і кодування», «Теорія ймовірності і математична статистика.

Студенти при вивченні дисципліни повинні сформувати погляд на захист інформації і криптографію як на систематичну науково-практичну діяльність, що носить прикладний характер. Сформувати базисні теоретичні поняття, що лежать в основі процесу захисту інформації.

У результаті вивчення дисципліни студент повинен знати: особливості інформації як об'єкту захисту, перелік та особливості основних загроз інформації в комп'ютерних системах, підходи до розробки політики безпеки комп'ютерної системи, основні принципи захисту інформації, порядок формування комплексу засобів захисту інформації, принципи управління доступом до захищених ресурсів комп'ютерної системи, криптографічні методи і засоби захисту інформації, способи та засоби захисту інформації від витоку технічними каналами, порядок визначення вимог щодо захисту інформації в комп'ютерній системі, основні положення нормативної бази системи захисту інформації в комп'ютерних системах. Вміти: застосовувати математичні методи описання і дослідження криптосистем; оцінювати криптографічну стійкість шифрів.

Структура та перелік викладених розділів та їх наповнення повністю відповідають освітній програмі – «комп'ютерні системи та мережі».

1. ЗМІСТ БАЗОВИХ ПОНЯТЬ ЗАХИСТУ ІНФОРМАЦІЇ В КОМП'ЮТЕРНИХ СИСТЕМАХ

1.1. Предмет захисту

Питання інформаційної безпеки займають особливе місце в зв'язку із зростаючою роллю в житті суспільства і вимагають до себе все більше уваги. Як відомо, всі виробничі процеси мають в своєму складі матеріальну і нематеріальну складові. Перша – це необхідне для виробництва устаткування, матеріали, енергія і т.д. Друга – технологія виробництва. В останні роки з'явилось багато галузей виробництва, які майже на 100% складаються із однієї інформації. Наприклад, дизайн, створення програмного забезпечення, реклама та інше. Із підвищенням важливості та цінності інформації відповідно росте і роль її захисту.

Тому **предметом захисту** є інформація, яка зберігається, обробляється, передається в комп'ютерних системах (КС).

Актуальність і важливість проблеми забезпечення безпеки інформаційних технологій обумовлені наступними причинами:

– різке збільшення потужності сучасних комп'ютерів при одночасному спрощенні їх експлуатації;

– різке збільшення обсягів інформації, що накопичується, зберігається, обробляється за допомогою КС;

– високі темпи росту парку персональних комп'ютерів (ПК), що знаходяться в експлуатації в самих різних сферах діяльності;

– різке розширення кола користувачів, що мають безпосередній доступ до обчислювальних ресурсів і масивів даних;

– бурхливий розвиток програмних засобів, що не задовольняють навіть мінімальним вимогам безпеки;

– повсюдне поширення мережних технологій і розвиток глобальної мережі Інтернет.

Інформація не матеріальна, зберігається і передається за допомогою матеріальних носіїв, будь-який матеріальний об'єкт містить інформацію про себе чи інший об'єкт. Вона має такі властивості:

1. **Цінність** інформації, яка визначається степенню її корисності для власника.

2. **Конфіденційність** інформації – це статус, що вказує на необхідні введення обмеження на коло суб'єктів, що мають доступ до даної інформації. Якщо доступ до інформації обмежений, то вона конфіденційна. Конфіденційна інформація може містити державну або комерційну таємницю.

3. **Цілісність** інформації – це властивість бути незмінним в сематичному змісті при функціонуванні системи в умовах випадкових чи навмисних перекручувань або впливів, що руйнують.

4. **Доступність** – це властивість бути доступним для авторизованих законних суб'єктів.

5. **Достовірність** визначається потрібною для власника точністю відображати об'єкти і процеси зовнішнього світу в певних часових і просторових рамках. Якщо інформація спеціально викривлена, то вона називається дезінформацією.

6. **Своєчасність** – відповідність цінності і достовірності певному часовому періоду. Дана властивість виражається відношенням:

$$C(t) = C_0 e^{\frac{-2.3t}{\tau}},$$

де C_0 – цінність інформації в момент її виникнення, t – час від моменту виникнення інформації до часу її оцінки, τ – час від моменту виникнення інформації до її старіння [1].

1.2. Об'єкт захисту

Об'єктом захисту інформації є комп'ютерна система або автоматизована система обробки інформації (АСОІ).

Під безпекою АСОІ розуміють її захищеність від випадкового або навмисного втручання в нормальний процес її функціонування, а також від спроб розкрадання, зміни або руйнування її компонентів.

Природа впливів на АСОІ може бути найрізноманітнішою (стихійні лиха, вихід з ладу елементів, помилки персоналу, проникнення зловмисника).

Інформаційна безпека досягається проведенням відповідного рівня політики безпеки. Під **політикою інформаційної безпеки** розуміють сукупність норм, правил, практичних рекомендацій, що регламентують роботу засобів захисту АСОІ від заданої множини погроз безпеки. **Система захисту інформації** – сукупність правових, організаційних мір, технічних, програмних і криптографічних засобів і методів, що забезпечують захищеність системи.

Необхідно навести два важливі висновки:

1. Трактовка проблем, зв'язаних із інформаційною безпекою, для різних категорій суб'єктів відрізняється (в одному випадку – „нехай краще зламається, ніж ворог дізнається хоча б один секретний біт", а в іншому – „нема у нас секретів, лише все б працювало").

2. Інформаційна безпека не зводиться винятково до захисту від несанкціонованого доступу (НСД), це принципово ширше поняття. Суб'єкт інформаційних відносин може понести збитки не лише від НСД, а й від поломки системи, що викликала перерву в роботі. Тому в більшості випадків захист від НСД стоїть не на першому місці.

1.3. Погрози безпеки інформації в комп'ютерних системах

Погроза інформації – це потенційно можлива подія, що приводить до наслідків порушення цілісності, доступності та конфіденційності інформації.

Порушення безпеки – реалізація даної погрози безпеки.

Атака (вторгнення, напад) – це дія, що починається зловмисником, і полягає у пошуку і використанні вразливостей КС для реалізації погрози безпеки.

Вразливість КС – це деяка невдала властивість системи, що уможливлює виникнення і реалізацію погрози.

Протидія погрозам безпеки є метою захисту АСОІ. Безпечна або захищена система – це система із засобами захисту, що успішно і ефективно протистоїть погрозам безпеки.

Розрізняють основні типи погроз:

– погрози порушення конфіденційності інформації (відбуваються кожний раз, коли отриманий НСД);

– погрози порушення цілісності інформації (цілісність може бути порушена навмисно зловмисником, а також у результаті об'єктивних впливів з боку середовища);

– погрози порушення працездатності системи (відмовлення в обслуговуванні, блокування авторизованого користувача).

Погроза безпеки поділяється на об'єктивну і суб'єктивну.

Об'єктивна погроза – незалежна від людини.

Суб'єктивна погроза – навмисна і ненавмисна, залежна від людини.

Ненавмисні суб'єктивні погрози.

1. Ненавмисні дії людей, персоналу, що приводять до часткового або повного виходу комп'ютерних систем з ладу.

2. Неправомірне використання устаткування або зміни режимів робіт цього устаткування.

3. Ненавмисне псування носіїв інформації.

4. Нелегальне використання неліцензійних програм і устаткування.

5. Порушення атрибутів, правил, розмежувань і т.д.

6. Неправильне проектування архітектури системи, у яку заздалегідь закладені „люки», „лази" і „дірки" для стороннього порушника. Ігнорування сторонніх порушень.

7. Вхід у комп'ютерну систему в обхід установлених правил.

8. Введення помилкових даних та інші.

Основні навмисні погрози.

1. Фізичне руйнування системи.

3. Дезорганізація комп'ютерної або обчислювальної системи.

4. Впровадження в число персоналу «зацікавленої» людини.

5. Навмисна погроза: шантаж, погроза персоналу комп'ютерної системи.

6. Застосування засобів технічних розвідок: фотографування, підслуховування і т.д.

7. Перехоплення побічних електромагнітних випромінювань і наведень.

8. Перехоплення переданих даних по каналу зв'язку і т.д.

9. Розкрадання носіїв інформації.

10. Незаконне одержання правил і атрибутів розмежування доступу.

11. Незаконне використання терміналів законних користувачів з метою проникнення в систему під іменем санкціонованого користувача.

12. Розкриття шифрів криптозахисту санкціонованих користувачів.

13. Упровадження закладок типу „троянських коней”, „жучків”, „пасток” і т.д. з метою впровадження недекларованого програмного забезпечення санкціонованих користувачів.

14. Незаконне підключення до ліній передачі даних з метою роботи між рядків, видаючи себе за законного користувача та інші.

Класифікація погроз інформаційної безпеки

Класифікація всіх можливих погроз інформаційної безпеки комп’ютерної системи може бути проведена за рядом базових ознак [1].

1. За природою виникнення.

Природні погрози – погрози, викликані впливами на КС і її компоненти об'єктивних фізичних процесів або стихійних природних явищ, що не залежать від людини.

Штучні погрози – погрози інформаційної безпеки КС, викликані діяльністю людини.

2. За ступенем навмисності прояву.

Погрози випадкової дії і/або погрози, викликані помилками або недбалістю персоналу. Погрози, не зв'язані з навмисними діями зловмисників і реалізовані у випадкові моменти часу, називають випадковими або ненавмисними. Реалізація погроз цього класу приводить до найбільших втрат інформації (до 80 % збитку). При цьому може відбуватися знищення, порушення цілісності, доступності і конфіденційності інформації, наприклад:

− прояв помилок програмно-апаратних засобів КС;

− некомпетентне використання, настроювання або неправомірне відключення засобів захисту персоналом служби безпеки;

− ненавмисні дії, що приводять до часткової або повної відмови системи або руйнуванню апаратних, програмних, інформаційних ресурсів системи (ненавмисне псування устаткування, видалення, перекручування файлів з важливою інформацією або програм, у тому числі системних і т.д.);

− неправомірне включення устаткування або зміна режимів роботи пристроїв і програм;

− ненавмисне псування носіїв інформації;

− пересилання даних по помилковій адресі абонента (пристрою);

− введення помилкових даних;

− ненавмисне ушкодження каналів зв’язку.

Погрози навмисної дії, наприклад:

− традиційне або універсальне шпигунство і диверсії (підслуховування, візуальне спостереження; розкрадання документів і машинних носіїв, розкрадання програм і атрибутів системи захисту, підкуп і шантаж співробітників, збір і аналіз відходів машинних носіїв, підпали, вибухи);

− несанкціонований доступ до інформації (реалізується за допомогою відсутності системи розмежування доступу, збоями або відмовою технічних засобів),

− побічні електромагнітні випромінювання і наведення;

− несанкціонована модифікація структур (алгоритмічної, програмної, технічної);

− інформаційні інфекції (шкідливі програми).

3. За безпосереднім джерелом погроз.

Погрози, безпосереднім **джерелом яких є природнє середовище** (стихійні лиха, магнітні бурі, радіоактивне випромінювання і т.д.).

Погрози, джерелом яких є людина, наприклад:

− впровадження агентів у число персоналу системи (у тому числі, можливо, і в адміністративну групу, що відповідає за безпеку);

− вербування (шляхом підкупу, шантажу і т.д.) персоналу або окремих користувачів, що мають визначені повноваження;

− погроза несанкціонованого копіювання секретних даних користувачем КС;

− розголошення, передача або втрата атрибутів розмежування доступу (паролів, ключів шифрування, ідентифікаційних карток, пропусків і т. д.).

Погрози, безпосереднім джерелом яких є санкціоновані програмно-апаратні засоби, наприклад:

− запуск технологічних програм, здатних при некомпетентному користуванні викликати втрату працездатності системи (зависання або зациклення) або необоротні зміни в системі (форматування або реструктуризацію носіїв інформації, видалення даних і т. д.);

− виникнення відмови в роботі операційної системи.

Погрози, безпосереднім джерелом яких є несанкціоновані програмно-апаратні засоби, наприклад:

− нелегальне впровадження і використання неліцензійних програм (ігрових, навчальних, технологічних і інших, що не є необхідними для виконання зловмисником своїх службових обов'язків) з наступною необґрунтованою витратою ресурсів (завантаження процесора, захоплення оперативної пам'яті і пам'яті на зовнішніх носіях);

− зараження комп'ютера вірусами з деструктивними функціями.

4. За положенням джерела погроз.

Погрози, джерело яких розташоване поза контрольованою зоною території (приміщення), на якій знаходиться КС, наприклад:

− перехоплення побічних електромагнітних, акустичних і інших випромінювань приладів і ліній зв'язку, а також наведень активних випромінювань на допоміжні технічні засоби, що безпосередньо не беруть участь в обробці інформації (телефонні лінії, мережі харчування, опалення);

− перехоплення даних, переданих по каналах зв'язку, і їхній аналіз з метою визначення протоколів обміну, правил входження в зв'язок і авторизації користувача і подальших спроб їхньої імітації для проникнення в систему;

− дистанційна фото- і відеозйомка.

Погрози, джерело яких розташоване у межах контрольованої зони території (приміщення), на якій знаходиться КС, наприклад:

− розкрадання виробничих відходів (роздруківок, записів, списаних носіїв інформації і т.д.);

− відключення або вивід з ладу підсистем забезпечення функціонування обчислювальних систем (електроживлення, охолодження і вентиляції, ліній зв'язку і т.д.);

− застосування підслухуючих пристроїв.

Погрози, джерело яких має доступ до периферійних пристроїв КС.

Погрози, джерело яких розташоване в КС, наприклад:

− проектування архітектури системи і технології обробки даних, розробка прикладних програм, що становлять небезпеку для працездатності системи і безпеки інформації;

− некоректне використання ресурсів КС.

5. За ступенем залежності від активності КС.

Погрози, що можуть виявлятися незалежно від активності КС, наприклад:

− розкриття шифрів криптоаналізу інформації;

− розкрадання носіїв інформації (магнітних дисків, стрічок, мікросхем пам'яті, запам'ятовуючих пристроїв і комп'ютерних систем).

Погрози, що можуть виявлятися тільки в процесі автоматизованої обробки даних (наприклад, погрози виконання і поширення програмних вірусів).

6. За ступенем впливу на КС.

Пасивні погрози, що при реалізації нічого не змінюють у структурі і змісті КС, наприклад: погроза копіювання секретних даних.

Активні погрози, що при впливі вносять зміни в структуру і зміст КС, наприклад:

− впровадження апаратних спецвкладок, програмних „закладок” і „вірусів”, тобто таких ділянок програм, що не потрібні для виконання заявлених функцій, але дозволяють перебороти систему захисту, таємно і незаконно здійснити доступ до системних ресурсів з метою реєстрації і передачі критичної інформації або дезорганізації функціонування системи;

− дії по дезорганізації функціонування системи (зміна режимів роботи пристроїв або програм, страйк, саботаж персоналу, постановка могутніх активних радіоперешкод на частотах роботи пристроїв системи і т.д.);

− погроза навмисної модифікації інформації.

7. За етапами доступу користувачів або програм до ресурсів КС.

Погрози, що можуть виявлятися на етапі доступу до ресурсів КС (наприклад, погрози несанкціонованого доступу в КС).

Погрози, що можуть виявлятися після дозволу доступу до ресурсів КС (наприклад, погрози несанкціонованого або некоректного використання ресурсів КС).

Несанкціонований доступ є найбільш розповсюдженим і різноманітним видом комп'ютерних порушень. Суть НСД складається в одержанні користувачем (порушником) доступу до об'єкта з порушенням правил розмежування доступу, установлених відповідно до прийнятої в організації політики безпеки. НСД використовує будь-яку помилку в системі захисту і можливий при нераціональному виборі засобів захисту, їх некоректній установці і настроюванню. НСД може бути здійснений як штатними засобами АСОІ, так і спеціально створеними апаратними і програмними засобами.

8. За способом доступу до ресурсів КС.

Погрози, спрямовані на використання прямого стандартного шляху доступу до ресурсів. Наприклад:

− Незаконне одержання паролів і інших реквізитів розмежування доступу (агентурним шляхом, використовуючи недбалість користувачів, підбором, імітацією інтерфейсу системи і т.д.) з наступним маскуванням під зареєстрованого користувача – «маскарад».

Перехоплення паролів здійснюється спеціально розробленими програмами. При спробі законного користувача ввійти в систему програма-перехоплювач імітує на екрані дисплея введення імені і пароля користувача, що відразу пересилаються власникові програми-перехоплювача, після чого на екран виводиться повідомлення про помилку і керування повертається операційній системі. Користувач припускає, що припустився помилки при введенні пароля. Він повторює введення й одержує доступ у систему. Власник програми-перехоплювача, що одержав ім’я і пароль законного користувача, може тепер використовувати їх у своїх цілях. Існують і інші способи перехоплення паролів.

«Маскарад» – це виконання яких-небудь дій одним користувачем від імені іншого користувача, що володіє відповідними повноваженнями. Метою «маскараду» є приписування яких-небудь дій іншому користувачеві або присвоєння повноважень і привілеїв іншого користувача. Прикладами реалізації „маскараду” є: вхід у систему під ім’ям і паролем іншого користувача (цьому „маскарадові” передує перехоплення пароля); передача повідомлень у мережі від імені іншого користувача; «маскарад» особливо небезпечний у банківських системах електронних платежів, де неправильна ідентифікація клієнта через „маскарад” зловмисника може привести до великих збитків законного клієнта банку; несанкціоноване використання терміналів користувачів, що мають унікальні фізичні характеристики, такі як номер робочої станції в мережі, фізична адреса, адреса в системі зв’язку, апаратний блок кодування і т.д.

Погрози, спрямовані на використання схованого нестандартного шляху доступу до ресурсів КС, наприклад:

− вхід у систему в обхід засобів захисту (завантаження сторонньої операційної системи зі змінних магнітних носіїв і т.д.);

− погроза несанкціонованого доступу до ресурсів КС шляхом використання недокументованих можливостей ОС.

9. За поточним місцем розташування інформації, збереженої й оброблюваної в КС.

Погрози доступу до інформації на зовнішніх запам'ятовуючих пристроях (наприклад, погроза несанкціонованого копіювання секретної інформації з твердого диска).

Погрози доступу до інформації в оперативній пам'яті, наприклад:

− читання залишкової інформації з оперативної пам'яті;

− читання інформації з областей оперативної пам'яті, використовуваних операційною системою (у тому числі підсистемою захисту) або іншими користувачами, в асинхронному режимі, використовуючи недоліки мультизадачних КС і систем програмування;

− погроза доступу до системної області оперативної пам'яті зі сторін прикладних програм.

Погрози доступу до інформації, що циркулює в лініях зв'язку, наприклад:

− незаконне підключення до ліній зв'язку з метою роботи „між рядками" з використанням пауз у діях законного користувача від його імені з наступним уведенням помилкових повідомлень або модифікацією переданих повідомлень;

− незаконне підключення до ліній зв'язку з метою прямої підміни законного користувача шляхом його фізичного відключення після входу в систему й успішної аутентифікації з наступним уведенням дезінформації і нав'язуванням помилкових повідомлень.

Погрози доступу до інформації, відображуваної на терміналі або друку на принтері, наприклад, погроза запису відображуваної інформації на приховану відеокамеру.

2. БАЗОВІ СХЕМИ АТАК ТА ОРГАНІЗАЦІЯ КАНАЛІВ ВИТОКУ ІНФОРМАЦІЇ. КЛАСИФІКАЦІЯ ЗЛОВМИСНИКІВ

2.1. Класифікація атак

Наведемо спрощену класифікацію, яка відображає найбільш типові атаки на АСОІ [3; 6].

– Віддалене проникнення або віддалене керування комп'ютером через мережу. Програми, що реалізують даний тип атак: NetBus, BackOrifice.

– Локальне проникнення, отримання НСД до вузлів, на яких ініційовані атаки. Програма, що реалізує даний тип атак – GetAdmin.

– Віддалена відмова в обслуговуванні або порушення функціонування системи, перевантаження комп'ютера через мережу. Програми, що реалізують даний тип атак: Teardrop, trin00.

– Локальна відмова в обслуговуванні. Приклад – відкрито багато вікон великого розміру, як наслідок перевантаження процесора.

– Сканування мережі. Аналіз топології мережі та активних сервісів, доступних для атаки. Така атака може бути створена за допомогою утиліти nmap.

– Використання сканерів уразливостей. Вони призначені для пошуку вразливостей на локальних та віддалених комп'ютерах. Такі сканери системнв адміністратори використовують як діагностичні інструменти. Програми, що реалізують даний тип атак: SATAN, SystemScanner, Xspider, nessus.

– Злам паролів. Приклади програм: L0phtCrack – Windows, Crack – Unix, AZPR – zip, rar архівів.

– Пасивне прослуховування мережі. Пасивна атак спрямована на розкриття конфіденційних даних. Приклади програм: tcpdump, Microsoft Network Monitor, LanExplorer.

2.2. Базові схеми атак

– Підміна (фальсифікація) об'єктів. Атаку здійснюють, скориставшись слабкістю системи аутентифікації.

– Модифікація даних. Навмисне псування даних. Атаки можуть бути спрямовані на інформаційні об'єкти, що перебувають у стані зберігання і в мережі.

– Відмова від авторства. Порушник має змогу відмовитись від здійснених ним дій. Причина – слабкість механізмів реєстрації подій та механізмів аутентифікації.

– Перехват та розголошення інформації.

– Відмова в обслуговуванні.

– Підвищення привілеїв.

2.3. Організація каналів витоку інформації

Основними причинами витоку інформації є: невиконання персоналом норм, вимог, правил експлуатації КС; помилки при проектуванні КС і систем захисту КС; ведення зловмисниками технічної і агентурної розвідки.

Види витоку інформації:

- розголошення;
- НСД;
- отримання захищеної інформації засобами розвідки.

Несанкціонований доступ у телекомунікаційні системи призводить до значних матеріальних втрат. Для успішної боротьби з порушеннями інформаційної безпеки користувач повинен знати всі канали витоку інформації.

Аналіз показує, що шляхи несанкціонованого одержання інформації дуже різноманітні і багаточисленні:

- несанкціоноване підключення до апаратури і ліній зв'язку;
- перехоплення електромагнітних випромінювань;
- примусове електромагнітне опромінення (підсвічування) ліній зв'язку;
- застосування пристроїв, що підслухують (закладань);
- перехоплення акустичних випромінювань;
- дистанційне фотографування;
- розкрадання носіїв інформації і виробничих відходів;
- зчитування даних у масивах інших користувачів;
- читання залишкової інформації в пам'яті системи після виконання санкціонованих запитів;
- копіювання носіїв інформації з подоланням заходів захисту;
- маскування під зареєстрованого користувача;
- використання програмних пасток,
- використання недоліків мов програмування й операційних систем;
- вмикання в бібліотеки програм спеціальних блоків типу «троянський кінь»;
- злочинне виведення з ладу механізмів захисту;
- впровадження і використання комп'ютерних вірусів.

Близьким по характеру є і перелік загроз безпеки інформації.

Канал витоку інформації – сукупність джерела інформації, матеріального носія або засобу розповсюдження сигналу, який проводить дану інформацію і засобів виділення інформації із сигналу чи носія.

Узагальнена схема каналів витоку інформації:

1. **Електромагнітний та електричний канал**. Причиною його появи є електромагнітне поле, яке зв’язане із проходження електричного струму в апаратних компонентах КС. Цей канал в свою чергу поділяється на: радіоканал (високочастотне випромінювання), низькочастотний, мережний (наведення на мережу електроспоживання), заземлення (наведення на проводи заземлення), лінійний (наведення на лінії зв’язку між комп’ютерами). Зазвичай засоби розвідки розташовані за межами контрольованої зони.

2. **Акустичний канал** – зв’язаний із розташуванням звукових хвиль у повітрі або пружних коливань в інших середовищах, які виникають при роботі пристроїв відображення інформації.

3. **Візуальний (оптичний) канал** – зв’язаний із можливістю візуального спостереження зловмисником за роботою пристроїв відображення інформації без проникнення в приміщення, де розташовані компоненти системи. Використовуються фото-, відео-камери.

4. **Інформаційний канал** – зв’язаний з доступом до елементів КС, до носіїв інформації, до самої інформації (результатів введення та виведення), до програмного забезпечення. Може бути розділений на наступні канали: локальна мережа, машинні носії, термінальні та периферійні пристрої, комп’ютерні лінії зв’язку.

5. **Перехоплення мовної інформації.**

Розмова кількох осіб**:** підслуховування, спрямований мікрофон, використання закладних пристроїв, цифрові диктофони, віброакустичний сигнал, гідроакустичний сигнал (в батарею, зливний бачок), мімічний канал (читання по губах), акустоелектричний канал (дифузор динаміка).

Розмова по телефону**:** все перераховане вище та сигнал в лінії зв’язку (котушка індуктивності) і закладний пристрій в телефонному апараті.

2.4. Класифікація зловмисників

Порушник (зловмисник) – фізична особа, яка порушує політику безпеки системи [3].

Можливості здійснення шкідливих впливів у великому ступені залежать від статусу зловмисника стосовно КС. Зловмисником може бути:

− **розробник КС**, який володіє найбільш повною інформацією про програмні й апаратні засоби КС і має можливість упровадження «закладок» на етапах створення і модернізації систем, але не одержує доступу на експлуатовані об’єкти КС;

− **співробітник з числа обслуговуючого персоналу.** Найбільш небезпечний клас – робітники служби безпеки інформації, системні і прикладні програмісти, інженерно-технічний персонал;

− **користувач**, який має загальне представлення про структуру КС і механізми її захисту, але може здійснювати збір інформації методами традиційного шпигунства і спробами НСД;

− **стороння особа**, яка може здійснювати дистанційні методи шпигунства і диверсійну діяльність.

До **основних напрямків реалізації** зловмисником інформаційних погроз відносяться:

− безпосереднє звертання до об’єктів доступу;

− створення програмних і технічних засобів, що виконують звертання до об’єктів доступу в обхід засобів захисту;

− модифікація засобів захисту, що дозволяє реалізувати погрози інформаційної безпеки;

− впровадження в технічні засоби КС програмних або технічних механізмів, що порушують передбачувану структуру і функції КС.

До числа **основних методів реалізації погроз** інформаційної безпеки відносяться:

− визначення зловмисником типу і параметрів носіїв інформації;

− одержання зловмисником інформації про програмно-апаратне середовище, тип і параметри засобів обчислювальної техніки, тип і версії операційної системи, складі прикладного програмного забезпечення;

− одержання зловмисником детальної інформації про функції, що виконує КС;

− одержання зловмисником даних про системи захисту;

− визначення способу представлення інформації;

− визначення зловмисником змісту даних, оброблюваних у КС, на якісному рівні (моніторинг дешифрування повідомлень);

− розкрадання (копіювання) машинних носіїв інформації, що мають конфіденційні дані;

– використання спеціальних технічних засобів для перехоплення побічних електромагнітних випромінювань і наведень – конфіденційні дані перехоплюються зловмисником шляхом зміни інформативних сигналів з електромагнітного випромінювання і наведень по ланцюгах харчування засобів обчислювальної техніки, що входить в КС;

– знищення засобів обчислювальної техніки і носіїв інформації;

– несанкціонований доступ користувача до ресурсів КС шляхом подолання систем захисту з використанням спецзасобів, прийомів, методів;

– несанкціоноване перевищення користувачем своїх повноважень;

– несанкціоноване копіювання програмного забезпечення;

– перехоплення даних, переданих по каналах зв’язку;

– візуальне спостереження – конфіденційні дані зчитуються з екранів, терміналів, роздруківок у процесі їхньої друку і т.д.;

– розкриття представлення інформації (дешифрування даних);

– внесення користувачем несанкціонованих змін програмно-апаратні компоненти КС і оброблюваних даних;

– установка і використання позаштатного апаратного і/або програмного забезпечення;

− зараження програмними вірусами;

– внесення перекручувань у представлення даних, знищення на рівні представлення, перекручування інформації при передачі по лініях зв'язку;

− упровадження дезінформації;

− виведення з ладу машинних носіїв інформації без знищення інформації (виведення з ладу електронних блоків твердих дисків і т.д.);

− прояв помилок проектування і розробки апаратних програмних компонентів КС;

− обхід (відключення) механізмів захисту – завантаження зловмисником нештатної операційної системи з дискети, використання режимів програмно-апаратних компонент КС і т.д.

− перекручування відповідності синтаксичних і семантичних конструкцій мови – установлення нових значень слів, виражень і т.д.;

− заборона на використання інформації – наявна інформація яким-небудь причинам не може бути використана.

2.5. Основні напрямки комп'ютерних злочинів

Комп'ютерні злочини – це передбачені карним законом суспільно небезпечні дії, у яких машинна інформація є об'єктом злочинного зазіхання. У даному випадку як предмет або знаряддя злочину є машинна інформація, комп'ютер, комп'ютерна система або комп'ютерна мережа [3]. Комп'ютерні злочини умовно можна підрозділити на дві великі категорії:

– злочини, зв'язані з втручанням у роботу комп'ютерів;

– злочини, що використовують комп'ютери як необхідні технічні засоби.

2.5.1. Основні види злочинів, зв'язані із втручанням у роботу комп'ютерів.

– **Несанкціонований доступ до інформації, що зберігається в комп'ютері.** НСД здійснюється, як правило, з використанням чужого імені, зміною фізичних адрес технічних пристроїв, використанням інформації, що залишилася після рішення задач, модифікацією програмного й інформаційного забезпечення, розкраданням носія інформації, установкою апаратури запису, що підключається до каналів передачі даних.

Хакер, комп'ютерний пірат – особа, що робить систематичні несанкціоновані доступи в комп'ютерні системи і мережі з метою розваги, шахрайства або нанесення збитку (у тому числі і шляхом поширення комп'ютерних вірусів). З одного боку «хакер», це людина, що прекрасно знає комп'ютер і пише гарні програми, а з іншого боку – незаконно проникаючий у комп'ютерні системи з метою одержання інформації.

Англійське дієслово «to **hack**» стосовно до комп'ютерів може означати дві речі – зламати систему або полагодити її. В основі цих дій лежить загальна основа: розуміння того, як улаштований комп'ютер, і програми, що на ньому працюють. Таким чином, слово хакер сполучає в собі принаймні два значення: одне – забарвлене негативно (зломщик), інше – нейтральне або навіть хвалебне («ас», «майстер»). Іншими словами, хакерів можна розділити на «поганих» і «гарних».

Гарні хакери рухають технічний прогрес і використовують свої знання й уміння на користь людства. Ними розроблене велике число нових технічних і програмних систем. Їм протистоять «погані» – вони читають чужі листи, крадуться чужі програми і всі доступні способи шкодять прогресивному людству.

Поганих хакерів можна умовно розділити на чотири групи. Перша, що складається в основному з молоді, – люди, що зламують комп'ютерні системи просто заради власного задоволення. Вони не наносять шкоди, а таке заняття досить корисне для них самих – згодом з них виходять чудові комп'ютерні фахівці. Друга група – пірати. Вони зламують захист комп'ютерів для

викрадення нових програм і іншої інформації. Третя група – хакери, що використовують свої пізнання дійсно на шкоду всім і кожному. Вони знищують комп'ютерні системи, у які їм удалося прорватися, читають чужі листи, а потім знущаються з їх авторів. Є і ще одна група – хакери, що полюють за секретною інформацією за замовленням.

– **Введення в програмне забезпечення «логічних бомб»**, що спрацьовують при виконанні визначених умов і частково або цілком виводять з ладу комп'ютерну систему.

«Тимчасова бомба» – різновид «логічної бомби», що спрацьовує по досягненні визначеного моменту часу.

Спосіб «троянський кінь» полягає в таємному введенні в чужу програму таких команд, що дозволяють здійснювати нові, що не планувалися власником програми, функції, але одночасно зберігати і колишню працездатність. Комп'ютерні програмні тексти звичайно складні, вони складаються із сотень, тисяч, а іноді і мільйонів команд. Тому троянський кінь з декількох десятків команд навряд чи може бути виявлений, якщо, звичайно, немає підозр щодо цього. Але й в останньому випадку експертам-програмістам буде потрібно багато днів і тижнів, щоб знайти його. Є ще один різновид троянського коня, його особливість полягає в тому, що в частину програми, де вставляються не команди, власне, що виконують «брудну» роботу, а команди, що формують ці команди і після виконання знищуються. У цьому випадку програмістові, що намагається знайти „троянського коня", необхідно шукати не самого коня, а команди, які його формують.

– **Розробка і поширення комп'ютерних вірусів.**

– **Злочинна недбалість у розробці, виготовленні й експлуатації програмно-обчислювальних комплексів, що привела до тяжких наслідків.**

Особливістю комп'ютерної необережності є те, що безпомилкових програм у принципі не буває. Якщо проект практично в будь-якій області техніки можна виконати з величезним запасом надійності, то в області програмування така надійність досить умовна, а в ряді випадків майже не досяжна.

– **Підробка комп'ютерної інформації.**

Очевидно, цей вид комп'ютерної злочинності є одним з найновішим. Він є різновидом несанкціонованого доступу з тією різницею, що користуватися ним може, як правило, не сторонній користувач, а сам розроблювач, що має досить високу кваліфікацію.

Ідея злочину полягає в підробці вихідної інформації комп'ютерів з метою імітації працездатності великих систем, складовою частиною яких є комп'ютер. При досить спритно виконаній підробці найчастіше вдається продати замовникові свідомо несправну продукцію. До підробки інформації можна віднести також підтасування результатів виборів, голосування, референдумів і т.д.

– **Розкрадання комп'ютерної інформації.**

Якщо «звичайні» розкрадання підпадають під дію існуючого карного закону, то проблема розкрадання інформації значно більш складна. Присвоєння

машинної інформації, у тому числі програмного забезпечення, шляхом несанкціонованого копіювання не кваліфікується як розкрадання, оскільки розкрадання сполучене з вилученням цінностей з фондів організації.

2.5.2. Злочини, для яких комп'ютер є засобом досягнення мети.

– **Розробка складних математичних моделей**, вхідними даними в яких є можливі умови проведення злочину, а вихідними даними - рекомендації з вибору оптимального варіанту дій злочинця.

– **Злочини з загальною назвою – «повітряний змій».**

У найпростішому випадку потрібно відкрити в двох банках по невеликому рахунку. Далі гроші переводяться з одного банку в інший і назад та поступово підвищуються суми. Хитрість полягає в тому, щоб до того, як у банку виявиться, що доручення про перевід не забезпечено необхідною сумою, приходило б повідомлення про перевід у цей банк, такий, що загальна сума покривала б вимоги про перший перевід. Цей цикл повторюється велике число раз («повітряний змій» піднімається усе вище і вище) доти, поки на рахунку не виявляється пристойна сума (фактично вона постійно «перескакує» з одного рахунка на інший, збільшуючи свої розміри). Тоді гроші швидко знімаються, а власник рахунка зникає. Цей спосіб вимагає дуже точного розрахунку, але для двох банків його можна зробити і без комп'ютера. На практиці в таку гру включають велику кількість банків, так сума накопичується швидше і число доручень про перевід не досягає підозрілої частоти. Але керувати цим процесом можна тільки за допомогою комп'ютера.

2.5.3. Класифікація комп'ютерних злочинів

Закордонними фахівцями розроблені різні класифікації способів здійснення комп'ютерних злочинів. У 1991 році даний кодифікатор був інтегрований в автоматизовану систему пошуку і в даний час доступний Національному Центральному Бюро Інтерполу більш ніж 100 країн [1].

Усі коди, що характеризують комп'ютерні злочини, мають ідентифікатор, що починається з букви Q. Для характеристики злочину можуть використовуватися до п'яти кодів, розташованих у порядку спадання значимості заподіяного.

3. ОРГАНІЗАЦІЯ СИСТЕМИ ЗАХИСТУ ІНФОРМАЦІЇ

3.1. Політика безпеки

Основним призначенням КС є переробка (збір, збереження, обробка і передача) інформації, тому проблема забезпечення інформаційної безпеки є для КС центральною. Забезпечення безпеки припускає організацію протидії будь-якому несанкціонованому вторгненню в процес функціонування КС, а також спробам модифікації, розкрадання, виведення з ладу або руйнування її компонентів, тобто захист усіх компонентів КС – апаратних засобів, програмного забезпечення, даних і персоналу.

Існують два підходи до проблеми забезпечення безпеки КС: фрагментарний і комплексний.

Фрагментарний підхід спрямований на протидію чітко визначеним погрозам у заданих умовах. Як приклади реалізації такого підходу можна вказати окремі засоби керування доступом, автономні засоби шифрування, спеціалізовані антивірусні програми і т.д. Перевагою такого підходу є висока вибірковість до конкретної погрози. Істотним недоліком даного підходу є відсутність єдиного захищеного середовища обробки інформації. Фрагментарні міри захисту інформації забезпечують захист конкретних об'єктів КС тільки від конкретної погрози. Навіть невелика видозміна погрози веде до втрати ефективності захисту.

Комплексний підхід орієнтований на створення захищеного середовища обробки інформації у КС, що поєднує в єдиний комплекс різнорідні міри протидії погрозам. Організація захищеного середовища обробки інформації дозволяє гарантувати визначений рівень безпеки КС, що є безсумнівною перевагою комплексного підходу. До недоліків цього підходу відносяться: обмеження на свободу дій користувачів КС, велика чутливість до помилок установки і настроювання засобів захисту, складність керування. Комплексний підхід застосовують для захисту великих організацій або невеликих КС, що виконують відповідальні задачі або обробляють особливо важливу інформацію. Порушення безпеки інформації в КС великих організацій може нанести величезний матеріальний збиток як самим організаціям, так і їх клієнтам. Тому такі організації змушені приділяти особливу увагу гарантіям безпеки і реалізовувати комплексний захист. Комплексного підходу дотримують більшість державних і великих комерційних підприємств і установ. Цей підхід знайшов своє відображення в різних стандартах. Комплексний підхід до проблеми забезпечення безпеки заснований на розробленій для конкретної КС політиці безпеки. **Політика безпеки** являє собою набір норм, правил і практичних рекомендацій, на яких будується керування, захист і розподіл інформації в КС. Політика безпеки регламентує ефективну роботу засобів захисту КС, вона охоплює всі особливості процесу обробки інформації, визначаючи поводження системи в різних ситуаціях.

Політика безпеки реалізується за допомогою адміністративно-організаційних мір, фізичних і програмно-технічних засобів і визначає архітектуру системи захисту. Для конкретної організації політика безпеки повинна носити індивідуальний характер і залежати від конкретної технології обробки інформації і використовуваних програмних і технічних засобів. Політика безпеки визначається способом керування доступом, що визначає порядок доступу до об'єктів системи. Розрізняють два основних види політики безпеки: **виборча** (дискреційна) і **повноважна** (мандатна).

Основними функціями системи розмежевання доступу є:

- реалізація прав розмежування доступу суб'єктів та їх процесів до даних;
- реалізація прав розмежування доступу суб'єктів та їх процесів до пристроїв створення твердих копій;
- ізоляція програм процесу, виконуваного в інтересах суб'єкта, від інших суб'єктів;
- управління потоками даних з метою запобігання запису даних на носії невідповідного рівня;
- реалізація правил обміну даними між суб'єктами для автоматизованої системи і засобів обчислювальної техніки, які побудовані за мережними принципами.

Виборча політика безпеки заснована на виборчому способі керування доступом. Виборче (або дискреційне) керування доступом характеризується заданою адміністратором множиною дозволених відносин доступу (наприклад, у виді трійок <об'єкт, суб'єкт, тип доступу>). Звичайно для опису властивостей виборчого керування доступом застосовують математичну модель на основі матриці доступу. Матриця доступу являє собою матрицю, у якій стовпець відповідає об'єктові системи, а рядок-суб'єктові. На перетинанні стовпця і рядка матриці вказується тип дозволеного доступу суб'єкта до об'єкта. Звичайно виділяють такі типи доступу суб'єкта до об'єкта, як «доступ на читання», «доступ на запис», «доступ на виконання» і т.д. Матриця доступу є самим простим підходом до моделювання систем керування доступом, вона є основою для більш складних моделей, що більш адекватно описують реальні КС.

Виборча політика безпеки широко застосовується в КС комерційного сектора, тому що її реалізація відповідає вимогам комерційних організацій по розмежуванню доступу і підзвітності, а також має прийнятну вартість.

Повноважна політика безпеки заснована на повноважному способі керування доступом. Повноважне (або мандатне) керування доступом характеризується сукупністю правил надання доступу, визначених на множині атрибутів безпеки суб'єктів і об'єктів, наприклад, в залежності від мітки конфіденційності інформації і рівня доступу користувача. Повноважне керування доступом має на увазі, що:

- усі суб'єкти й об'єкти системи однозначно ідентифіковані;

- кожному об’єктові системи привласнена мітка конфіденційності інформації, що визначає цінність інформації, що утримується в ньому;
- кожному суб’єктові системи привласнений визначений рівень допуску, що визначає максимальне значення мітки конфіденційності інформації об'єктів, до яких суб’єкт має доступ.

Чим важливіший об’єкт, тим вище його мітка конфіденційності. Тому найбільш захищеними виявляються об'єкти з найбільш високими значеннями мітки конфіденційності. Основним призначенням мандатної політики безпеки є регулювання доступу суб’єктів системи до об’єктів з різними рівнями конфіденційності, запобігання витоку інформації з верхніх рівнів посадової ієрархії на нижні, а також блокування можливих проникнень з нижніх рівнів на верхні.

Крім двох основних видів поліки безпеки розглядають ще декілька видів: політика безпеки інформаційних потоків, ролева політика безпеки, політика ізольоваго програмного середовища.

Політика безпеки інформаційних потоків.

Крім керування доступом суб’єктів до об’єктів системи проблема захисту інформації має ще один аспект. Для одержання інформації про який-небудь об’єкт системи зовсім необов'язково шукати шляхи несанкціонованого доступу до нього. Необхідну інформацію можна одержати, спостерігаючи за обробкою необхідного об'єкта, тобто використовуючи канали витоку інформації. У системі завжди існують інформаційні потоки. Тому адміністраторові необхідно визначити, які інформаційні потоки в системі є «легальними», тобто не ведуть до витоку інформації, а як-ведуть до витоку. Тому виникає необхідність розробки правил, що регламентують керування інформаційними потоками в системі. Звичайне керування інформаційними потоками застосовується в рамках виборчої або повноважної політики, доповнюючи їх і сприяючи підвищенню надійності системи захисту.

Ролева політика безпеки – це самостійна політика безпеки, яка базується на дискреційній або мандатній політиці безпеки. Відповідно до політики ролевого розмежування доступу права доступу суб’єктів формуються згідно із їх повноважними обов’язками, тобто ролями. Дану політику безпеки застосовують у мережних операційних системах, великих системах управління базами даних.

Політика ізольованого програмного середовища визначає безпечний порядок взаємодії суб’єктів системи, який унеможливлює модифікацію її параметрів. Вся множина інформаційних потоків поділяється на дві підмножини, що не перетинаються – потоки несанкціонованого доступу, які підлягають фільтрації та потоки легального доступу.

Виборче і повноважне керування доступом, а також керування інформаційними потоками є тим фундаментом, на якому будується вся система захисту.

3.2. Побудова системи захисту

Під системою захисту КС розуміють єдину сукупність правових і морально-етичних норм, адміністративно-організаційних мір, фізичних і програмно-технічних засобів, спрямованих на протидію погрозам КС з метою зведення до мінімуму можливості збитку.

Процес побудови системи захисту включає наступні етапи:

- аналіз можливих погроз КС;
- планування системи захисту;
- реалізація системи захисту;
- супровід системи захисту.

Етап аналізу можливих погроз КС необхідний для фіксації стану АСОІ – визначення конфігурації апаратних і програмних засобів, технології обробки інформації і виявлення впливів, що діють на компоненти системи. Практично неможливо забезпечити захист АСОІ від усіх впливів, оскільки неможливо цілком установити всі погрози і способи їхніх реалізацій. Тому з усієї множини ймовірних впливів вибирають тільки такі впливи, що можуть реально відбутися і завдати серйозної шкоди.

На етапі планування формулюється система захисту як єдина сукупність мір протидії погрозам різної природи .

За способами здійснення всі міри забезпечення безпеки комп'ютерних систем підрозділяють на:

1. **Нормативно-правові (законодавчі).**
2. **Морально-етичні.**
3. **Адміністративні (організаційні).**
4. **Фізичні (технічні).**
5. **Апаратно-програмні.**

Перераховані міри безпеки КС можна розглядати як послідовність бар’єрів або рубежів захисту інформації. Для того щоб добратися до інформації, що захищається, потрібно послідовно перебороти кілька рубежів захисту. Розглянемо їх докладніше.

Перший рубіж захисту, що встає на шляху людини, яка намагається здійснити НСД до інформації, є правовим. Цей аспект захисту інформації зв'язаний з необхідністю дотримання юридичних норм при передачі й обробці інформації. До правових мір захисту інформації відносяться діючі в країні закони, укази й інші нормативні акти, що регламентують правила роботи з інформацією обмеженого використання і відповідальності за їхні порушення. Цим вони перешкоджають несанкціонованому використанню інформації і є стримуючим чинником для потенційних порушників.

Другий рубіж захисту утворюють морально-етичні міри. Етичний момент у дотриманні вимог захисту має досить велике значення. Дуже важливо, щоб люди, що мають доступ до комп’ютерів, працювали в здоровому морально-етичному кліматі. До морально-етичних мір протидії відносяться норми поводження, що традиційно склалися або складаються в суспільстві в міру поширення комп’ютерів у країні. Ці норми здебільшого не є обов'язковими, як

законодавчо затверджені, але їхнє недотримання звичайне веде до падіння престижу людини, групи осіб або організації. Морально-етичні норми бувають як неписаними (наприклад, загальновизнані норми чесності, патріотизму і т.д.), так і оформленими в якийсь звід правил або розпоряджень. Наприклад, «Кодекс професійного поводження членів Асоціації користувачів ЕОМ США» розглядає як неетичні дії, що навмисне або ненавмисно: порушують нормальну роботу комп'ютерних систем; викликають невиправдані витрати ресурсів (машинного часу, пам’яті, каналів зв’язку); порушують цілісність інформації; порушують інтереси інших законних користувачів і т.д.

Третім рубежем, що перешкоджає неправомірному використанню інформації, є адміністративні міри. Адміністратори всіх рангів з урахуванням правових норм і соціальних аспектів визначають адміністративні міри захисту інформації.

Адміністративні міри захисту відносяться до мір організаційного характеру. Вони регламентують:

- процеси функціонування КС;
- використання ресурсів КС;
- діяльність її персоналу;
- порядок взаємодії користувачів із системою, для того щоб найбільшою мірою утруднити або виключити можливість реалізації погроз безпеки.

Адміністративні міри включають:

- розробку правил обробки інформації в КС;
- сукупність дій при проектуванні й устаткуванні обчислювальних центрів і інших об'єктів КС (облік впливу стихії, пожеж, охорона приміщень);
- сукупність дій при підборі і підготовці персоналу (перевірка нових співробітників, ознайомлення їх з порядком роботи з конфіденційною інформацією, з мірами відповідальності за порушення правил її обробки; створення умов, при яких персоналові було б невигідно припускатися зловживань і т.д.);
- організацію надійного пропускного режиму;
- організацію обліку, збереження, використання і знищення документів і носіїв з конфіденційною інформацією;
- розподіл реквізитів розмежування доступу (паролів, повноважень і т.п.);
- організацію схованого контролю за роботою користувачів і персоналу КС;
- сукупність дій при проектуванні, розробці, ремонті і модифікації устаткування і програмного забезпечення (сертифікація використовуваних технічних і програмних засобів, строге санкціонування, розгляд і твердження всіх змін, перевірка на задоволення вимогам захисту, документальна фіксація змін і т.д.).

Важливо відзначити, що, поки не будуть реалізовані дійові заходи адміністративного захисту ЕОМ, інші міри будуть, безсумнівно, неефективні.

Адміністративно-організаційні міри захисту можуть здатися нудними і рутинними в порівнянні з морально-етичними і позбавленими конкретності в порівнянні з апаратно-програмними. Однак вони являють собою могутній бар'єр на шляху незаконного використання інформації і надійну базу для інших рівнів захисту.

Четвертим рубежем є фізичні міри захисту. До фізичних мір захисту відносяться різного роду механічні, электро- і електронно-механічні пристрої або спорудження, спеціально призначені для створення фізичних перешкод на можливих шляхах проникнення і доступу потенційних порушників до компонентів системи й інформації, що захищається.

П'ятим рубежем є апаратно-програмні засоби захисту. До них відносяться різні електронні пристрої і спеціальні програми, що реалізують самостійно або в комплексі з іншими засобами наступні способи захисту: ідентифікацію (розпізнавання) і аутентифікацію (перевірка дійсності) суб'єктів (користувачів, процесів) КС; розмежування доступу до ресурсів КС; контроль цілісності даних; забезпечення конфіденційності даних; реєстрацію й аналіз подій, що відбуваються в КС; резервування ресурсів і компонентів КС.

Більшість з перерахованих способів захисту реалізується криптографічними методами захисту інформації.

При проектуванні ефективної системи захисту варто враховувати ряд принципів, що відображають основні положення по безпеці інформації. До числа цих принципів відносяться наступні:

- Економічна ефективність. Вартість засобів захисту повинна бути менше, ніж розміри можливого збитку.
- Мінімум привілеїв. Кожен користувач повинний мати мінімальний набір привілеїв, необхідний для роботи.
- Простота. Захист тим більше ефективний, чим легше користувачеві з нею працювати.
- Відключення захисту. При нормальному функціонуванні захист не повинний відключатися. Тільки в особливих випадках співробітник зі спеціальними повноваженнями може відключити систему захисту.
- Відкритість проектування і функціонування механізмів захисту. Фахівці, що мають відношення до системи захисту, повинні цілком уявляти собі принципи її функціонування й у випадку виникнення складних ситуацій адекватно на них реагувати.
- Загальний контроль. Будь-які виключення з множини контрольованих суб’єктів і об’єктів захисту знижують захищеність автоматизованого комплексу обробки інформації.
- Незалежність системи захисту від суб’єктів захисту. Особи, що займалися розробкою системи захисту, не повинні бути в числі тих, кого ця система буде контролювати.
- Звітність і підконтрольність. Система захисту повинна надавати доказу коректності своєї роботи.

- Відповідальність. Мається на увазі особиста відповідальність осіб, що займаються забезпеченням безпеки інформації.
- Ізоляція і поділ. Об'єкти захисту доцільно розділяти на групи таким чином, щоб порушення захисту в одній із груп не впливало на безпеку інших груп.
- Повнота і погодженість. Надійна система захисту повинна бути цілком специфікована, опротестована і погоджена.
- Параметризація. Захист стає більш ефективним і гнучким, якщо він допускає зміну своїх параметрів з боку адміністратора.
- Принцип ворожого оточення. Система захисту повинна проектуватися в розрахунку на вороже оточення. Розроблювачі повинні виходити з припущення, що користувачі мають найгірші наміри, що вони будуть робити серйозні помилки і шукати шляхи обходу механізмів захисту.
- Залучення людини. Найбільш важливі і критичні рішення повинні прийматися людиною.
- Відсутність зайвої інформації про існування механізмів захисту. Існування механізмів захисту повинний бути по можливості приховано від користувачів, робота яких повинна контролюватися.

Результатом етапу планування є розгорнутий план захисту АСОІ, що містить перелік компонентів, що захищаються, АСОІ і можливих впливів на них, ціль захисту інформації в АСОІ, правила обробки інформації в АСОІ, що забезпечують її захист від різних впливів, а також опис планованої системи захисту інформації.

Сутність **етапу реалізації** системи захисту полягає в установці і настроюванні засобів захисту, необхідних для реалізації запланованих правил обробки інформації.

Заключний етап супроводу полягає в контролі роботи системи, реєстрації подій, що відбуваються в ній, їхньому аналізі з метою виявлення порушень безпеки, корекції системи захисту.

3.3. Основні підсистеми комплексу засобів захисту

Комплекс засобів захисту (КЗЗ) – це сукупність підсистем, які забезпечують необхідні сервіси безпеки. На рис.3.1 показано типову структуру КЗЗ [3].

Така структура може бути реалізована повністю або частково на всіх рівнях КС (застосування-додатки, бази даних, операційні системи, мережі).

Реєстрація і облік – всіх подій в КС, які пов’язані з безпекою та подальшим аналізом цих подій.

Керування доступом: дискреційне, мандатне.

Ідентифікація + аутентифікація = авторизації: ідентифікація – присвоєння унікального ідентифікатора (пароль та інше); аутентифікація – перевірка запропонованого ідентифікатора; авторизація – надання користувачу визначених повноважень у системі.

Забезпечення цілісності – контроль здійснюється під час запуску системи чи програми на виконання. За особливо критичними інформаційними та програмними ресурсами контроль здійснюється неперервно.

Криптографія – шифрування та дешифрування, хешування, цифровий електронний підпис, генерація та розподіл ключів.

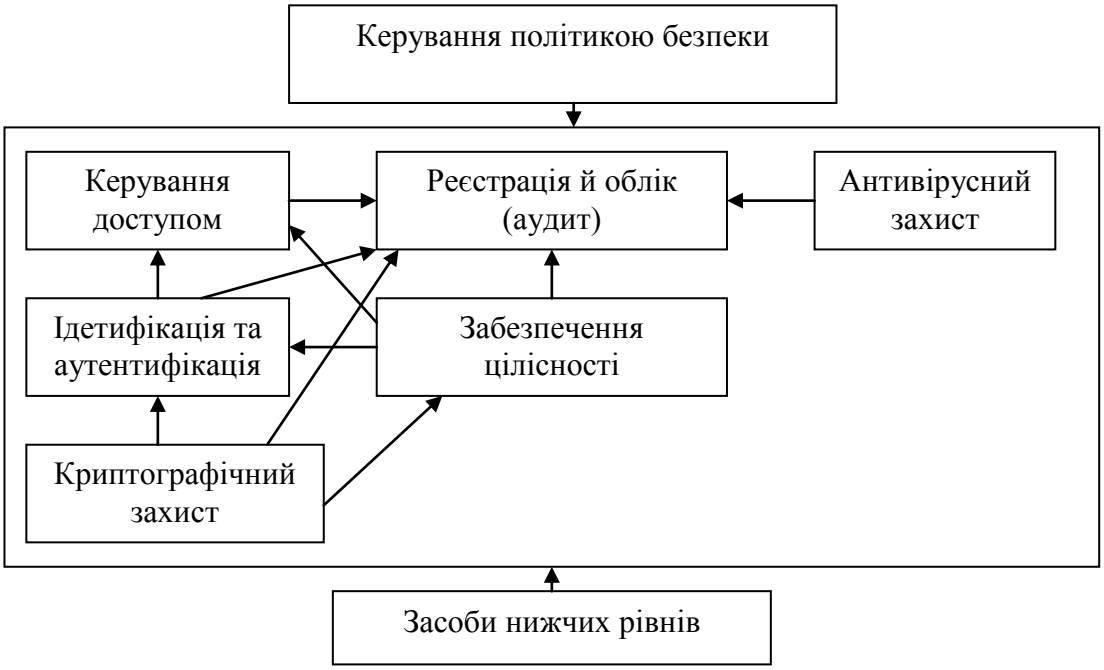

Рисунок 3.1 – Структура комплексу засобів захисту

4. МАТЕМАТИЧНІ МОДЕЛІ БЕЗПЕКИ

4.1. Види математичних моделей безпеки

Формальне визначення політики безпеки називають математичною моделлю безпеки.

Згідно вимог нормативних документів у галузі захисту інформації в інформаційних системах, системи захисту інформації будують на основі математичних моделей захисту інформації. Використання цих моделей дозволяє теоретично обґрунтувати відповідність системи захисту інформації вимогам заданої політики безпеки. Формальна теорія захисту інформації почала розвиватися відносно недавно, але сьогодні існує багато математичних моделей, які описують різні аспекти безпеки і надають доказову теоретичну базу для побудови сучасних систем захисту інформації [3].

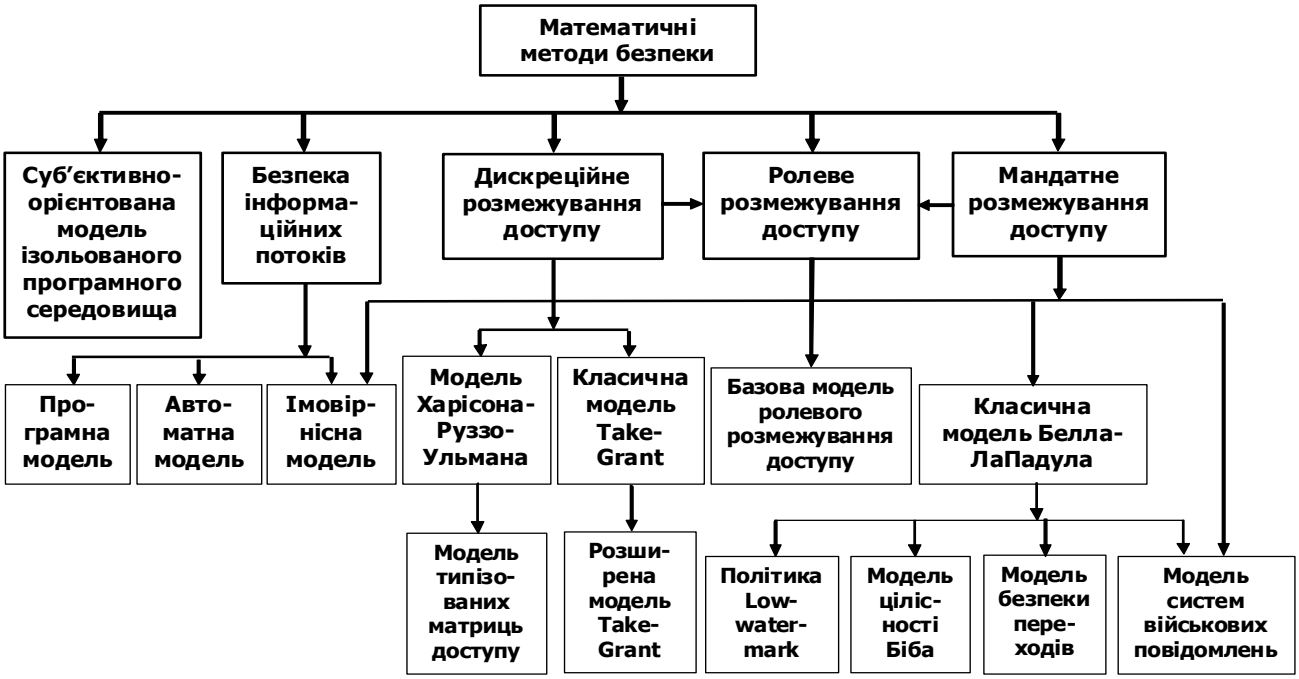

Рисунок 4.1 – Основні види математичних моделей безпеки

4.2. Моделі дискреційної політики безпеки

Політика дискреційного доступу охоплює найбільшу сукупність моделей розмежування доступу, реалізованних в більшості захищених КС, і історично є першою проробленою в теоретичному і практичному плані.

Перші роботи по моделях дискреційного доступу до інформації в КС з'явилися ще в 60-х роках і докладно представлені в літературі. Найбільш відомі з них - модель **АДЕПТ-50** (кінець 60-х років), **п'ятимірний простір Хартсона** (початок 70-х років), модель **Харіссона-Руззо-Ульмана (HRU)** (середина 70-х років), **модель Take-Grant** (1976 р.). Авторами і дослідниками цих моделей був зроблений значний внесок у теорію безпеки комп'ютерних систем, а їх роботи заклали основу для подальшого створення та розвитку захищених КС.

Моделі дискреційного доступу безпосередньо грунтуються і розвивають суб'єктно-об'єктну модель КС як сукупність деяких множин взаємодіючих

елементів. Множина безпечних доступів у моделях дискреційного доступу визначається дискретним набором трійок «Користувач (суб'єкт)-потік (операція)-об'єкт» [3].

Модель Хартсона

Модель дискреційного доступу, запропонована Хартсоном найбільш наочно у формальному плані ілюструє дискреційний принцип розмежування доступу, виражений мовою реляційної алгебри. Наведемо її основні положення в короткому викладі.

1. Система представляється сукупністю п'яти множин: множина користувачів *U*; множина ресурсів *R*; множини станів *S*; множина встановлених повноважень *A*; множина операцій *E*.

2. Область безпеки представляється декартовим добутком: $A \times U \times E \times R \times S$.

3. Користувачі подають запити на доступ до ресурсів, здійснення яких переводить систему в новий стан. Запити на доступ представляються чотиривимірними кортежами: $q = (u, e, R', s)$, де $u \in U$, $e \in E$, $s \in S$, $R' \in R$ (R'– потрібний набір ресурсів). Таким чином, запит на доступ являє собою підпростір чотиривимірної проекції простору безпеки. Запит задовольняється, якщо він повністю входить в область безпеки.

4. Процес організації доступу алгоритмічно описується наступним чином.

4.1. Визначити з *U* ті групи користувачів, до яких належить *u*. Потім вибрати з *A* ті специфікації, яким відповідають виділені групи користувачів. Цей набір повноважень *F(u)* визначає привілей користувача *u*.

4.2. Визначити з множини *A* набір повноважень *P=F(e)*, які встановлюють ***e*** як основну операцію. Набір повноважень *P=F(e)* визначає привілеї операції ***e***.

4.3. Визначити з множини *A* набір повноважень *P=F(R')*, які дозволяють доступ до набору ресурсів *R'*. Набір повноважень *P=F(R')* визначає привілеї ресурсів *R'*. Повноваження, які є загальними для всіх трьох привілеїв, утворюють так званий домен повноважень запиту *D(q): D(q)=F(u)∩F(e)∩F (R')*.

4.4. Переконатися, що запитуваний набір ресурсів *R'* повністю міститься в домені запиту *D (q)*, тобто будь-який *r* з набору *R'* хоча б один раз присутній серед елементів *D(q)*.

4.5. Здійснити розбиття *D(q)* на еквівалентні класи так, щоб в один клас потрапляли повноваження (елементи *D(q)*), коли вони специфікують один і той же ресурс r з набору *R'*. У кожному класі зробити операцію логічного АБО елементів *D(q)* з урахуванням типу операції *e*. У результаті формується новий набір повноважень на кожну одиницю ресурсу, зазначеного в *D(q)-F(u,q)*. Набір *F(u,q)* називається фактичною привілеєю користувача u по відношенню до запиту *q*.

4.6. Обчислити умову фактичного доступу, відповідну запиту *q*, через операції логічного АБО за елементами повноважень *F(u,q)* і запитуваним ресурсом *r* з набору *R'*, і отримати набір *R''* – набір фактично доступних за запитом ресурсів.

4.7. Оцінити умову фактичного доступу і прийняти рішення про доступ: дозволити доступ, якщо R'' і R' повністю перекриваються або відмовити в доступі в іншому випадку.

Зауважимо, що при всій своїй наочності модель Хартсона володіє одним, але істотним недоліком – безпека системи на основі даної моделі строго не доведена. Користувачі, здійснюючи законний доступ до ресурсів, можуть змінювати стан системи, в тому числі, змінювати і множину ресурсів R. Тим самим може змінюватися сама область безпеки. Чи збережеться в такому випадку безпека системи? Модель відповіді на дане питання не дає [3].

Модель ADEPT-50

Вперше була опублікована у 1970 році. Розглядає чотири типи об'єктів: користувачі (u); завдання (j); термінали (t); файли (f).

Кожний з об'єктів описується кортежем (A, C, F, M), який містить основні параметри безпеки:

- Компетенція (A) – скалярна величина – елемент з набору ієрархічно впорядкованих характеристик цінності, наприклад, грифів таємності.
- Категорія (C) – дискретний набір (множина) рубрик. Категорії не залежать від рівня компетенції.
- Повноваження (F) – множина користувачів, що мають право доступу до цього об'єкта.
- Режим (M) – набір (множина) дозволених видів доступів.

Правила функціонування системи в ADEPT-50:

$U=\{u_i,\ i=1,\ldots,n\}$ – множина усіх відомих системі користувачів.

$F(i)=\{u_j,\ j=1,\ldots,m\le n\}$ – множина користувачів, що мають право використовувати об'єкт i. Об'єктом може бути завдання, термінал, або файл.

Правила, за якими функціонує система Σ:

Користувач u_i отримує доступ до системи Σ тільки тоді, коли $u_i \in U$.

Користувач u_i отримує доступ до термінала t_j тільки тоді, коли $u_i \in F(t_j)$.

Користувач u_i отримує доступ до файлу f_k тільки тоді, коли $u_i \in F(f_k)$, $A(j_l)\ge A(f_k)$, $C(j_l)\supseteq C(f_k)$, $M(j_l)\supseteq M(f_k)$ [6].

Таблиця 4.1

Матриця визначення параметрів безпеки

Об'єкт	***A***	***C***	***F***	***M***
Користувач u_i	Константа	Константа	$\{u_i\}$	Константа
Термінал t_j	Константа	Константа	$F(t_j)$	Константа
Завдання j_l	$\min(A(u_i), A(t_j))$	$C(u_i)\cap C(t_j)$	$F(j_l)$	$M(ui)\cap M(tj)$
Файл f_k	Константа	Константа	$F(f_k)$	Константа
Новий файл $f=(f_1,f_2)$	$\max(A(f_1), A(f_2))$	$C(f_1)\cup C(f_2)$	$F(j_l)$	$M(f_1)\cap M(f_2)$

Модель HRU

Модель Харіссона-Руззо-Ульмана (HRU – M. Harrison, W. Ruzzo, J. Ullman) вперше запропонована у 1971 році, формальний опис моделі з'явився у 1976 р. Модель HRU використовується для аналізу системи захисту, що реалізує дискреційну політику безпеки та її матриці доступів.

Система захисту є кінцевим автоматом, що функціонує відповідно до певних правил переходу. Стан системи визначається її матрицею доступів. Зміна стану (перехід) – це внесення змін до матриці доступів. Будемо називати системою автомат, що побудований відповідно до положень моделі HRU.

Позначення: O – множина об'єктів системи; S – множина суб'єктів системи, причому $S \subseteq O$; де o та s – об'єкти та суб'єкти, відповідно; R – множина прав доступу суб'єктів до об'єктів (наприклад, read, write, own); M – матриця доступу. Рядки матриці доступів відповідають суб'єктам, стовпчики матриці доступів відповідають об'єктам та суб'єктам, елементи матриці $M[s,o] \subseteq R$ – права доступу суб'єкта s до об'єкта o.

Таблиця 4.2

Примітивні оператори

Примітивний оператор моделі HRU	**Умови виконання**	**Новий стан системи**
«Внести» право $r \in R$ в $M[s,o]$	$s \in S$, $o \in O$	$S'=S$, $O'=O$, $M'[s,o] = M[s,o] \cup \{r\}$, $\forall (s',o') \neq (s,o) \Rightarrow M'[s',o']=M[s',o']$
«Видалити» право $r \in R$ із $M[s,o]$	$s \in S$, $o \in O$	$S'=S$, $O'=O$, $M'[s,o]=M[s,o] \setminus \{r\}$, $\forall (s',o') \neq (s,o) \Rightarrow M'[s',o']=M[s',o']$
«Створити» суб'єкт s'	$s' \notin S$	$S'=S \cup \{s'\}$, $O'=O \cup \{s'\}$, $\forall (s,o) \in S \times O \Rightarrow M'[s,o]=M[s,o]$, $\forall o \in O' \Rightarrow M'[s',o]=\varnothing$, $\forall s \in S' \Rightarrow M'[s,s']=\varnothing$
«Створити» об'єкт o'	$o' \notin O$	$S'=S$, $O'=O \cup \{o'\}$, $\forall (s,o) \in S \times O \Rightarrow M'[s,o]=M[s,o]$, $\forall s \in S' \Rightarrow M'[s,o']=\varnothing$
«Знищити» суб'єкт s'	$s' \in S$	$S'=S \setminus \{s\}$, $O'=O \setminus \{s'\}$, $\forall (s,o) \in S' \times O' \Rightarrow M'[s,o]=M[s,o]$
«Знищити» об'єкт o'	$o' \in O$, $o' \notin S$	$S'=S$, $O'=O \setminus \{o'\}$, $\forall (s,o) \in S' \times O' \Rightarrow M'[s,o]=M[s,o]$

Команди

З примітивних операторів можна складати команди. Команди описують перехід $Q \vdash cQ'$ системи із стану $Q=(S,O,M)$ в результуючий стан $Q' =(S',O',M')$. Новий стан відрізняється від попереднього принаймні одним компонентом. Кожна команда складається з двох частин: умови, за якими виконується команда та послідовність примітивних операторів.

Запис команди:

command C($x1$, …, xk)
if ($r1 \in M[xs1,xo1]$) and … and ($rm \in M[xsm,xom]$) then
α1; …
αn;
end.

Тут $x_1,...,x_k$ – формальні параметри команди (ідентифікатори об'єктів і суб'єктів), $r_1,\ldots,r_m \in R$ – права доступу, $\alpha_1,\ldots,\alpha_n$ – послідовність примітивних операторів. Умови в тілі команди не обов'язкові.

Означення 1

Витік права $r \in R$ є можливим за результатом виконання команди C, якщо при переході системи зі стану Q у стан Q' виконується такий примітивний оператор, що вносить право r в елемент матриці доступів M, який до того (тобто, у стані Q) не містив це право.

Означення 2

Початковий стан Q_0 називається безпечним відносно деякого права $r \in R$, якщо з Q_0 неможливий перехід системи у такий стан Q, в якому може виникнути витік права r.

Означення 3

Система називається монооперaційною, якщо кожну команду виконує лише примітивний оператор.

Теорема 1

Існує алгоритм, що дозволяє здійснювати перевірку, чи є вихідний стан монооперaційної системи безпечним для даного права $r \in R$.

Теорема 2

Задача перевірки довільних систем алгоритмічно нерозв'язувана: тобто, для загального випадку доведено негативний результат. А саме, не існує алгоритму, який може для довільної системи, її початкового стану Q_0 і загального права r вирішити, чи є ця конфігурація безпечною. Доведення спирається на властивості машини Тьюринга, за допомогою якої моделюється послідовність переходів системи зі стану в стан.

Приклади систем, для яких задача є розв'язуваною:

1. Монооперaційні системи.

Неможливо реалізувати вимоги деяких політик безпеки. Наприклад, неможливо надавати суб'єктам специфічні права на створювані ними об'єкти (не існує оператора, який би і створював об'єкт, і надавав права на нього).

2. Системи, в яких відсутня операція «створити».

3. Монотонні і моноумовні системи. Тобто, системи, в яких немає операторів «знищити» і «видалити» і в частині умов кожної команди присутнє не більш як одне речення.

Задача безпеки для системи зі скінченною множиною суб'єктів розв'язувана, але обчислювально складна [3].

Модель Take-Grant

У багатьох моделях дискреційного доступу виявилась проблема несанкціонованого поширення прав доступу. У 1976 році була запропонована модель Take–Grant. Як основний елемент цієї моделі використовуються графи доступів і правила їх перетворень. Призначенням моделі Take–Grant є аналіз шляхів розповсюдження прав доступу по вихідному графу прав доступу у системах дискреційного розмежування доступу. Модель допускає наявність прав доступу не лише у суб'єктів до об'єктів, але і в об'єктів до об'єктів. У

подальшому з'явилась розширена модель Take–Grant, яка стала одним із методів дослідження захищених систем [3].

Основні елементи моделі Take–Grant*: O* – множина об'єктів системи; $S \subseteq O$ – множина суб'єктів системи; *E* – множина встановлених прав доступу суб'єкта *x* до об'єкта *y* із правом α, де $\alpha \in R = \{r_1, r_2, ..., r_m\} \cup \{t, g\}$ – множина видів прав доступу, *t* (*take*) – повноваження брати права доступу, *g* (*grant*) – повноваження надавати права доступу.

Основу моделі Take–Grant складає скінченний орієнтований граф без петель $G \subseteq (S,O,E)$, який визначає поточні доступи в системі. У цьому графі елементи множин *S i O* є вершинами графу, які позначаються як: ⊗ – об'єкти (елементи множини $O \backslash S$), ● – суб'єкти (елементи множини *S*). Елементи множини $E \subseteq O \times O \times R$ – ребра графу, кожне з яких помічено непустою підмножиною множини видів прав доступу *R*, $\alpha_i \in R$.

Переходи із стану до стану

Порядок переходу із стану до стану системи визначається операціями або правилами перетворення графу доступів. Перетворення графа *G* в граф *G'* у результаті виконання правила *op* позначимо через $G \vdash op\ G'$. У класичній моделі Take–Grant розглядаються чотири **де-юре** правила перетворення графа. Виконання кожного з правил може бути ініційовано лише суб'єктом, який є активною компонентою системи:

- Правило *take*(α,*x*,*y*,*z*) – право брати права доступу.
- Правило *grant*(α,*x*,*y*,*z*) – право надавати права доступу.
- Правило *create*(β,*x*,*y*) – право створювати новий об'єкт.
- Правило *remove*(α,*x*,*y*) – право видалити права доступу на об'єкт.

Правило *take*(α,*x*,*y*,*z*)

Нехай $x \in S, y, z \in O$ – різні вершини графу *G*, $\beta \subseteq R$, $\alpha \subseteq \beta$.

Правило визначає порядок одержання нового графа доступів *G'* з графу *G*

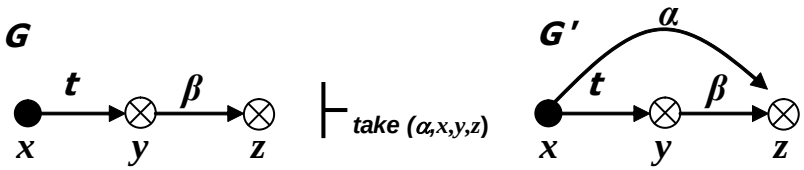

Рисунок 4.2 – Суб'єкт *x* бере в об'єкта *y* права $\alpha \subseteq \beta$ на об'єкт *z*.

Правило *grant*(α,*x*,*y*,*z*)

Нехай $x \in S, y, z \in O$ – різні вершини графу *G*, $\beta \subseteq R$, $\alpha \subseteq \beta$. Правило визначає порядок одержання нового графу доступів *G'* із графу *G*.

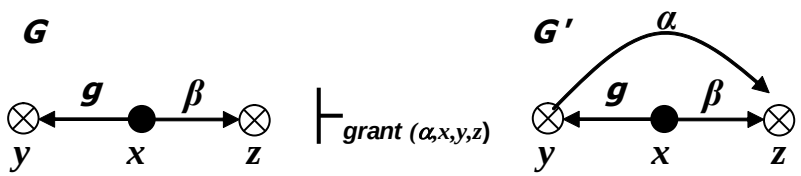

Рисунок 4.3 – Суб'єкт *x* надає об'єкту *y* права $\alpha \subseteq \beta$ на об'єкт *z*.

Правило *create*(β,*x*,*y*)

Нехай $x \in S$, $\beta \subseteq R$, $\beta \neq \varnothing$. Правило визначає порядок одержання нового графу G' з графу G; $y \in O$ – новий об'єкт або суб'єкт

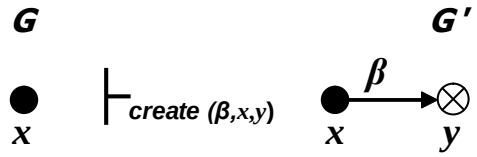

Рисунок 4.4 – Суб'єкт *x* створює новий β – доступний об'єкт *y*.

Правило *remove*(α,*x*,*y*)

Нехай $x \in S$, $y \in O$ – різні вершини графу G, $\beta \subseteq R$, $\alpha \subseteq \beta$. Правило визначає порядок одержання нового графа G' з графа G

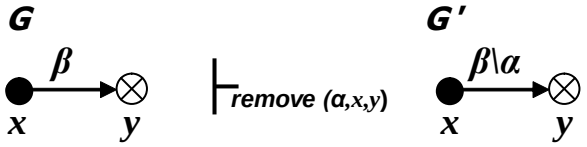

Рисунок 4.5 – Суб'єкт *x* видаляє право доступу α на об'єкт *y*

Таблиця 4.3

Умови застосування де-юре правил *take*, *grant*, *create*, *remove*

Правила де-юре моделі *Take-Grant*	**Вихідний стан $G=(S,O,E')$**	**Результуючий стан $G'=(S',O',E')$**
take(α,*x*,*y*,*z*)	$x\in S$, $(x,y,t)\in E$, $(y,z,\beta)\in E$, $x\neq z$, $\alpha\subseteq\beta$	$S'=S$, $O'=O$, $E'=E\cup\{(x,z,\alpha)\}$
grant(α,*x*,*y*,*z*)	$x\in S$, $(x,y,g)\in E$, $(x,z,\beta)\in E$, $y\neq z$, $\alpha\subseteq\beta$	$S'=S$, $O'=O$, $E'=E\cup\{(y,z,\alpha)\}$
create(β,*x*,*y*)	$x\in S$, $y\notin O$	$O'=O\cup\{y\}$, $S'=S\cup\{y\}$, якщо *y* суб'єкт, $E'=E\cup\{(x,y,\beta)\}$
remove(α,*x*,*y*)	$x\in S$, $y\in O$, $(x,z,\beta)\in E$, $\alpha\subseteq\beta$	$S'=S$, $O'=O$, $E'=E\setminus\{(x,y,\alpha)\}$

Розширена модель Take-Grant

У розширеній моделі розглядаються шляхи і вартості виникнення інформаційних потоків у системах із дискреційним розмежуванням доступу. У даній моделі розглядають не лише правила де-юре, які є формальним змістом моделі і обов'язкові до виконання, але і так звані правила де-факто, які дозволяють аналізувати фактичну ситуацію у системі.

У класичній моделі Take-Grant розглядають права доступу до об'єкта: *r (read), w (write), t (take), g (grant*) та чотири правила де-юре перетворення графа доступів: *take*, *grant*, *create*, *remove*. У розширеній моделі Take-Grant додатково розглядаються шість правил **де-факто** перетворення графа доступів: *post; spy; find; pass; n1; n2*.

Правила де-факто вводять для пошуку шляхів виникнення інформаційних потоків у системі, які не регламентовані правилами де-юре. У результаті застосування до графу доступу правил де-факто, до нього додаються уявні дуги, що позначаються *r* або *w*, але зображуються пунктиром. Разом із дугами графа, що відповідають реальним правам доступу *r* і *w* (суцільними дугами), уявні дуги вказують на наявність та напрям інформаційних каналів між об'єктами в системі. До уявних дуг не можна застосовувати правила де-юре перетворення графу доступів, оскільки вони позначають наявність і напрям каналів, які неможливо брати і передавати іншим об'єктам у системі.

Правило *post*

Нехай суб'єкт *x* має право r на об'єкт *y i* суб'єкт *z* має право *w* на об'єкт *y*.

Суб'єкт *x*, переглядаючи інформацію в об'єкті *y*, може шукати в ньому інформацію із *z* або про *z*. Таким чином, у системі може виникнути інформаційний канал від суб'єкта *z* до суб'єкта *x* (об'єкт як поштова скринька). Для реалізації такого каналу необхідна участь принаймні двох суб'єктів.

Правило *spy*

Нехай суб'єкт *x* має право *r* на суб'єкт *y* і суб'єкт *y* має право *r* на об'єкт *z*.

Суб'єкт *x*, переглядаючи інформацію в сб'єкті *y*, може шукати в ньому інформацію із *z* або про *z*. Таким чином, у системі може виникнути інформаційний канал від об'єкта *z* до суб'єкта *x* (шпигунство). Для реалізації такого каналу необхідна участь принаймні двох суб'єктів.

Правило *find*

Нехай суб'єкт *x* має право *w* на суб'єкт *y* та суб'єкт *y* має право *w* на об'єкт *z*.

Інформація, записана суб'єктом *x* в *y*, може бути переписана суб'єктом *y* в *z*. Таким чином, у системі може виникнути інформаційний канал від суб'єкта *x* до об'єкта *z*. Для реалізації такого каналу необхідна участь принаймні двох суб'єктів.

Правило *pass*

Нехай суб'єкт *y* має право *r* на об'єкт *z* і суб'єкт *y* має право *w* на об'єкт *x*.

Інформація, прочитана суб'єктом *y* в *z*, може бути записана в *x*. Таким чином, у системі може виникнути інформаційний канал від об'єкта *z* до об'єкта *x*. Для реалізації трьох з чотирьох розглянутих вище правил необхідна співпраця двох суб'єктів.

Правило *n1*

Нехай суб'єкт *x* має право *w* на суб'єкт *y*.

Таким чином, у системі може виникнути інформаційний потік від суб'єкта *x* до суб'єкта *y*. Де-факто це еквівалентно тому, що суб'єкт *y* має право *r* на суб'єкт *x* .

Правило *n2*

Нехай суб'єкт *x* має право *r* на суб'єкт *y*.

Таким чином, у системі може виникнути інформаційний потік від суб'єкта *y* до суб'єкта *x*. Де-факто це еквівалентно тому, що суб'єкт *y* має право *w* на суб'єкт *x*. Таким чином, аналізуючи результат застосування правил де-факто достатньо розглядати лише право *r* або лише право *w*.

4.3. Моделі політики мандатного доступу

Вихідним поштовхом до розробки мандатної політики послужили проблеми з контролем поширення прав доступу, і, в особливості, проблема «троянських» програм у системах із дискреційним доступом. Дослідники і критики дискреційної політики, розуміючи, що основна проблема дискреційних моделей полягає у відсутності контролю за інформаційними потоками, відзначили наступне:

1. Розмежування доступу і порядок роботи з конфіденційними «паперовими» документами організуються на основі парадигми градації довіри певним групам працівників у відношенні державних секретів певної міри важливості. З цією метою вводиться система рівнів безпеки, або інакше рівнів секретності.

Працівники з найвищим рівнем безпеки (рівнем довіри), можуть працювати з документами найвищого ступеня секретності. На більш низькому рівні довіри, тобто на більш низькому рівні безпеки, вводяться обмеження відносно роботи з документами більш високого рівня секретності і т. д., відповідно, всі працівники отримують допуск до роботи з секретними документами певного рівня, а документи позначаються спеціальною міткою, що відображає вимоги до рівня безпеки при роботі з ними – гриф секретності.

2. Критерієм безпеки є неможливість отримання інформації із документів певного рівня безпеки працівником, чий рівень безпеки, тобто рівень довіри, нижче, ніж рівень безпеки відповідних документів.

Даний критерій безпеки фактично означає заборону певних інформаційних потоків, які трактуються як небезпечні і недопустимі. Крім того, були проаналізовані правила і система призначень, змін, заборон, допусків співробітників до роботи з секретними документами, правила створення, знищення документів, привласнення чи зміни грифів їх секретності, в тому числі і розсекречення, а також інші особливості роботи з секретними документами. Зокрема було зазначено, що правила отримання доступу до документів розрізняються в залежно від характеру роботи з ними – вивчення (читання) або зміна (створення, знищення, внесення доповнень, редагування, тобто запис в них). На цій основі було «виявлено» два основних правила, які гарантують безпеку: NRU і NWD.

Правило 1. (No read up (NRU) – немає читання вгору). Працівник не має права знайомитися з документом (читати), гриф секретності (рівень безпеки) якого вище його ступеня допуску (рівня безпеки).

Правило 2. (**No write down** (NWD) – немає запису вниз). Працівник не має права вносити інформацію (писати) свого рівня безпечності в документ з більш низьким рівнем безпеки (з більш низьким грифом секретності).

Модель Белла-ЛаПадули

Першою формальною моделлю мандатного доступу є модель, розроблена ще в 1972-1975 р.р. американськими фахівцями – співробітниками MITRE Corporation Девідом Беллом і Леонардом ЛаПадулою. Названа в честь них модель зіграла величезну методологічну роль у розвитку теорії комп'ютерної безпеки. У моделі аналізуються умови, при яких неможливе створення інформаційних потоків від суб'єктів з більш високим рівнем доступу до суб'єктів з більш низьким рівнем доступу.

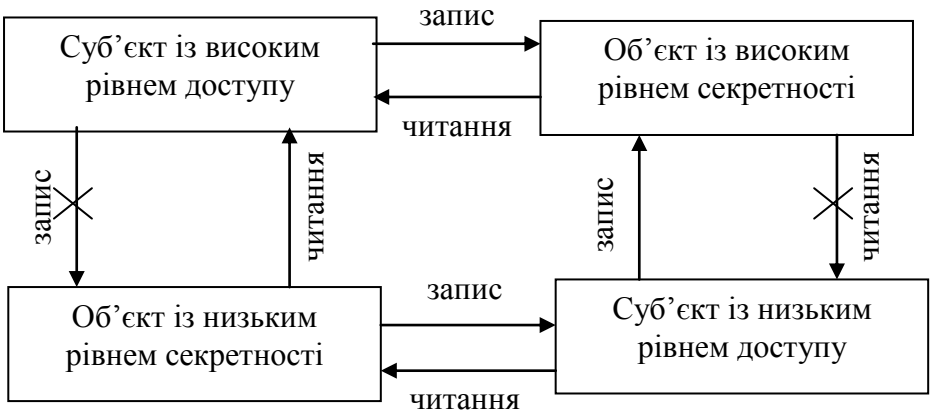

Рисунок 4.6 – Взаємодія суб'єктів та об'єктів системи

Основні положення моделі Белла-ЛаПадули.

Модель системи $\Sigma(v_0,Q,FT)$ представляється сукупністю: множини об'єктів O доступу; множини суб'єктів S доступу; множини прав доступу R (у класичній моделі Белла-ЛаПадули всього два елементи – read і write); матриці доступу $A[s,o]$; решітки Λ_L рівнів безпеки L суб'єктів і об'єктів системи; функції F_L: $S \cup O \rightarrow L$, що відображає елементи множин S і O на множину L; множини станів системи V, яке визначається множиною впорядкованих пар (F_L,A); початкового стану $v_0 \cup V$; набору запитів Q суб'єктів на доступ (здійснення операцій) до об'єктів, виконання яких переводить систему в новий стан; функції переходів FT: $(V \times Q) \rightarrow V^*$, яка переводить систему з одного стану V в інший V^* при виконанні запитів з Q [3].

Означення 1. Стан називається безпечним з читання (або просто безпечним - ss) тоді і тільки тоді, коли для кожного суб'єкту, що здійснює в цьому стані доступ читання до об'єкту, рівень безпеки цього суб'єкта домінує над рівнем безпеки цього об'єкта: $\forall s \in S, \forall o \in O, read \in A[s,o] \rightarrow F_{L(s)} \geq F_{L(o)}$.

Означення 2. Стан називається безпечним по запису (або *-безпечним) тоді і тільки тоді, коли для кожного суб'єкта, який здійснює в цьому стані доступ запису до об'єкта, рівень безпеки цього об'єкта домінує над рівнем безпеки цього суб'єкта: $\forall s \in S, \forall o \in O, write \in A[s,o] \rightarrow F_{L(o)} \geq F_{L(s)}$.

Означення 3. Стан системи безпечний тоді і тільки тоді, коли вона безпечна і за читанням, і за записом.

У літературі існують позначення : *ss*-безпека – безпека за читанням або проста безпека; *-безпека – безпека за записом; *ds*-безпека – дискреційна безпека за матрицею доступу.

Означення 4. (Критерій безпеки в моделі Белла-лапаДули)

Система $\Sigma(v_0,Q,FT)$ безпечна тоді і тільки тоді, коли вона має одразу три властивості *ss*, ***, *ds*.

На основі даного критерію Белла і ЛаПадула довели теорему, яка отримала назву «Основної теореми безпеки» (Basic Security Theorem – BST).

Теорема BST

Система $\Sigma(v_0,Q,FT)$ безпечна тоді і тільки тоді, коли: стан v_0 безпечний і функція переходів *FT* така, що для будь-якого стану *v*, який досягається із v_0 при виконанні кінцевої послідовності запитань з множини *Q* таких, що при $FT(v,q)=v^*$, де $v=(F_L,A)$ і $v^*=(F_L^*,A^*)$, переходи системи зі стану в стан підпорядковані наступним обмеженням для $s\in S$ *і для* $o\in O$:

– якщо $read \in A^*[s,o]$ ³ $read \in A[s,o]$, òî $F_{L^*(s)} \geq F_{L^*(o)}$;

– якщо $read \in A[s,o]$ ³ $F_{L^*(s)} < F_{L^*(o)}$, òî $read \notin A^*[s,o]$;

– якщо $write \in A^*[s,o]$ ³ write $\notin A[s,o]$, òî $F_{L^*(s)} \geq F_{L^*(o)}$;

– якщо $write \in A[s,o]$ ³ $F_{L^*(s)} < F_{L^*(o)}$, òî write $\notin A^*[s,o]$.

Модель Белла-ЛаПадули зіграла величезну роль у розвитку теорії комп'ютерної безпеки, і її положення були введені в якості обов'язкових вимог до систем, що обробляють інформацію, яка містить державну таємницю, в стандартах захищених КС, в тому числі, у відомій «Оранжевій книзі» (1983 р.). Однією з сильних сторін моделі Белла-ЛаПадули є автоматичне вирішення проблеми «троянських» програм. Відомі основні розширення моделі: модель із безпечною функцією переходу; уповноважені суб'єкти; груповий доступ – Low-Watermark [3].

5. ОРГАНІЗАЦІЙНО- ТЕХНІЧНІ ЗАДАЧІ ЗАБЕЗПЕЧЕННЯ ЗАХИСТУ ІНФОРМАЦІЇ

5.1. Загальні поняття

Так як незнання закону не звільняє від відповідальності за його порушення, ця тема присвячена саме юридичним питанням захисту інформації. Дану тему рекомендується особливо уважно прочитати хакерам і іншим аматорам чужих секретів, щоб вони через незнання не порушили якого-небудь закону, а якщо вже і порушили, то точно знали, який саме.

Хто, від кого або від чого, як і яку інформацію повинний захищати?

Хто – адміністративні, технічні, силові, юридичні і правоохоронні служби держави або недержавних організацій в особі служб інформаційної безпеки різних рівнів, відповідних відділів міліції, державних контролюючих органів і судів.

Від кого – від іноземних або вітчизняних, приватних або державних, юридичних або фізичних осіб, що не мають права легального доступу до інформації, але прагнучих оволодіти такою інформацією з метою нанесення збитку її власникові або для витягу особистої вигоди для себе. Цікаво відзначити той факт, що досить часто держава виключає себе зі списку можливих зловмисників, ставлячи свою цікавість до чужих секретів вище бажання своїх громадян зберегти ці секрети.

Від чого – конкретна шкідлива дія порушника може бути спрямована проти одного з трьох властивостей інформації:

– доступності інформації (інформація і її автоматизовані служби, що надають, повинні бути готові до обслуговування запиту до них завжди, коли в цьому виникне необхідність);

– конфіденційності інформації (засекречена інформація повинна бути доступна тільки тому, кому вона призначена);

– цілісності інформації (у будь-який момент часу інформація повинна бути достовірною і захищеною від можливих навмисних перекручувань або збоїв апаратури);

Як – шляхом створення надійних систем збереження самої інформації, грамотного адміністрування готових систем, прийняття охоронно-режимних, слідчо-оперативних і профілактичних мір до порушників безпеки.

Яку – насамперед акцентуємо той факт, що всілякі законодавчі акти різних країн у сфері захисту інформації спрямовані на захист не всієї інформації, а тільки особливої інформації, що позначається спеціальним грифом – грифом таємності або знаком інтелектуальної власності. З юридичної точки зору, ознакою таємності конкретного документа може вважатися не тільки відповідний значок у верхньому правому куті документа, але також і сам значеннєвий зміст документа. Грифи можуть привласнювати державні чиновники компетентних відомств або патентні бюро.

За важливістю на Україні й у Росії інформацію звичайно відносять до одного з п’яти класів таємності [5].

Таблиця 5.1

Грифи таємності

Гриф таємності інформації	Термін дії грифу
Знак інтелектуальної власності	не визначений
Не для друку (НДД)	не визначений
Для службового користування/ Конфіденційна	не визначений
Секретна (С)	5 років
Цілком секретна (ЦС)	10 років
Особливої важливості (ОВ)	30 років

Кожен клас (табл. 5.1.) припускає, що важливість середньостатистичного документа (середня вартість, що можуть заплатити за нього зацікавлених осіб) зростає на порядок у порівнянні з вартістю аналогічного документа з попереднього класу. Найменший гриф НДД звичайно привласнюється документам, що містять інформацію, вартість якої еквівалентна одному дневі роботи кваліфікованого співробітника. У різних країнах у різних організаціях і фірмах у залежності від профілю їхньої роботи і рівня бюрократизму, можуть бути інші критерії оцінок.

Для роботи з електронними документами різного ступеня таємності повинні використовуватися комп'ютерні системи різного рівня безпеки. Однак тому що саме поняття «рівень безпеки» занадто неконкретне і воно не може бути обмірюване у формальних одиницях виміру, то для прояснення ситуації необхідно штучно ввести шкалу оцінок. Різні організації пропонують свої системи класифікацій рівнів безпеки КС.

5.2. Критерії оцінювання захищених комп'ютерних систем

Загальновизнаною на сьогоднішній день є класифікація національного комітету з комп'ютерної безпеки США. Через те, що вона стала загальноприйнятим світовим стандартом і лежить в основі всіх інших класифікацій, приведемо її цілком нижче. Історично першим оцінним стандартом, що одержав широке поширення і зробив величезний вплив на базу стандартизації у багатьох країнах, став стандарт Міністерства оборони США «Критерії оцінювання захищених комп'ютерних систем» (Trusted Computer System Evaluation Criteria – TCSEC).

Дана праця, названа найчастіше по кольорі обкладинки «**Оранжевою книгою**», була вперше опублікована у серпні 1983 року. В ній мова йде не про безпечні, а про довірені системи, тобто системи, яким можна надати визначений ступінь довіри.

«Оранжевою книгою» пояснюється поняття безпечної системи. Очевидно, однак, що абсолютно безпечних систем не існує, це абстракція. Є зміст оцінювати лише ступінь довіри, який можна надати тій або іншій системі.

У «Оранжевій книзі» довірена система визначається як «система, що використовує достатні апаратні і програмні засоби, щоб забезпечити одночасну

обробку інформації різного ступеня таємності групою користувачів без порушення прав доступу». Ступінь довіри оцінюється за двома основними критеріями – політика безпеки та рівень гарантування.

Політика безпеки – набір законів, правил і норм поводження, що визначають, як організація обробляє, захищає і поширює інформацію. Зокрема, правила визначають, у яких випадках користувач може оперувати конкретними наборами даних. Чим вище ступінь довіри до системи, тим суворіша і багатогранніша повинна бути політика безпеки. У залежності від сформульованої політики можна вибирати конкретні механізми забезпечення безпеки. Це активний аспект захисту, що включає в себе аналіз можливих погроз і вибір мір протидії.

Рівень гарантування – міра довіри, що може бути надана архітектурі і реалізації інформаційної системи. Довіра безпеки може виникати як з аналізу результатів тестування, так і з перевірки (формального чи ні) загального задуму і реалізації системи в цілому й окремих її компонент. Рівень гарантованості показує, наскільки коректні механізми, що відповідають за реалізацію політики безпеки. Це пасивний аспект захисту.

5.2.1. Механізми безпеки

Згідно з «Оранжевою книгою», політика безпеки повинна обов'язково містити в собі наступні елементи: довільне керування доступом*;* безпека повторного використання об'єктів*;* мітки безпеки*;* примусове керування доступом.

Довільне керування доступом (дискреційним) – це метод розмежування доступу до об'єктів, заснований на обліку особистості суб'єкта або групи, у яку суб'єкт входить. Довільність керування полягає в тому, що деяка особа (звичайно власник об'єкта) може за своїм розсудом надавати іншим суб'єктам або відбирати в них права доступу до об'єкта.

Безпека повторного використання об'єктів – важливе доповнення засобів керування доступом, що охороняє від випадкового або навмисного витягу конфіденційної інформації з «сміття». Це повинно стосуватися областей оперативної пам'яті (зокрема, для буферів з образами екрана, розшифрованими паролями і т.д.), для дискових блоків і магнітних носіїв у цілому.

Операційна гарантованість – це спосіб переконатися в тому, що архітектура системи і її реалізація дійсно реалізують обрану політику безпеки.

5.2.2. Класи безпеки

У стандарті TCSEC визначається чотири групи, що відповідають різному ступеню захищеності (довіри) – D, C, B і A.

Група D (мінімальний захист) має лише один клас – **D1,** призначений для систем, визнаних незадовільними.

В міру переходу від рівня C до A до систем пред'являються усе більш жорсткі вимоги. Рівні C і B підрозділяються на класи (C1, C2, B1, B2, B3) з поступовим зростанням ступеня довіри.

Група C (дискреційний захист).

Клас C1 – системи цього класу містять засоби контролю і керування доступом, які дають змогу визначати обмеження для окремих користувачів, що

надає їм можливість захищати свою приватну інформацію від інших користувачів.

Клас С2 (на додаток до С1). Системи цього класу здійснюють більш гнучке керування доступом, ніж системи класу С1, застосовуючи засоби індивідуального контролю за діями користувачів, реєстрації, обліку подій і виділення ресурсів. Саме вимогам цього класу відповідають операційні системи (ОС) і СКБД універсального призначення. Серед ОС можна назвати Windows NT, HP-UX, Irix, Solaris, СКБД – Oracle, Informix та деякі інші.

Група В (мандатний захист). Слід відмітити, що у США законодавство вимагає обов'язкового застосування цих систем, сертифікованих відповідно до групи В, в огранах державного управління, оборонному відомстві, спецслужбах тощо. До таких систем належать Trusted Irix, Trusted Solaris та інші.

Клас В1 (на додаток до С2). Системи цього класу мають відповідати всім вимогам, що висуваються до класу С2, та підтримувати визначену неформально модель безпеки, маркування даних, мандатне керування доступом.

Класу В2 відповідає КС, яка підтримує формально визначену і чітко документовану модель безпеки, що передбачає дискреційне і мандатне керування доступом до всіх суб'єктів. Крім того, ця система повинна здійснювати контроль прихованих каналів витоку інформації та виявляти у структурі ядра захисту критичні з точки зору безпеки елементи.

Щоб система відповідала **класу В3**, її ядро захисту має містити монітор взаємодій, який контролював би всі типи доступу суб'єктів до об'єктів і який неможливо обійти. КС класу **В3** повинні містити засоби відновлення роботоздатності системи.

Група А (верифікований захист). Ця група містить лише один клас **А1**. Системи цього класу еквівалентні системам класу В3, до них не висувають додаткових вимог. Під час розроблення систем класу А1, на відміну від класу В3, застосовують формальні методи верифікації; це надає більшої впевненості у тому, що буде отримано коректну реалізацію функцій захисту.

Звичайно, на адресу «Критерії оцінювання захищених комп'ютерних систем» можна висловити цілий ряд серйозних зауважень: таких, наприклад, як повне ігнорування проблем, що виникають у розподілених системах; часто сама ОС може належати до класу С2, а дискреційне керування доступом у ній належить до класу В3, керування безпекою – до класу В2. Проте, варто підкреслити, що публікація «Оранжевої книги» без усякого перебільшення стала епохальною подією в області інформаційної безпеки. З'явився загальновизнаний понятійний базис, без якого навіть обговорення проблем інформаційної безпеки було б скрутним.

Відзначимо, що величезний ідейний потенціал «Оранжевої книги» поки багато в чому залишається незатребуваним. Насамперед це стосується концепції технологічної гарантованості, що охоплює весь життєвий цикл системи – від вироблення специфікацій до фази експлуатації. При сучасній технології програмування результуюча система не містить інформації, що є

присутнім у вихідних специфікаціях, губиться інформація про семантику програм.

На основі TCSEC було розроблено інші стандарти інформаційної безпеки, відтак з'явилася ціла низка супутніх документів TCSEC.

Крім американської класифікації захищених систем на сьогоднішній день існують також канадська, японська, французька, українська і російська класифікації.

5.3. Законодача і нормативна база захисту інформації в України

5.3.1. Закони України

Перш ніж говорити про Україну, відзначимо, що відповідальність користувача за правопорушення при роботі із конфіденційною інформацією, комп'ютерами і комп'ютерною інформацією в розвитих країнах узаконена вже давно. Коротко перелічимо деякі закони, що найбільше яскраво висвітлюють дану юридичну сферу:

У США – «Закон про безпеку комп'ютерних систем» – 1987р., «Акт про зловживання з використанням ЕОМ» – 1986р., «Закон про свободу інформації», «Закон про захист обчислювальних засобів невеликих фірм і освіти», «Закон про дотримання таємниць» – 1974р., «Закон про фінансові таємниці», «Закон про авторське право», «Конституція США» – 1787р.

У Канаді – «Закон про комп'ютерні й інформаційні злочини» – 1985р.

В Франції – «Закон про інформатику, картотеки і |.:свободах» – 1978р.

У Великобританії – «Закон про захист інформації» – 1984р.

У Німеччині – «Закон про подальший розвиток електронної обробки і захисту даних» – 1990р., «Федеральний закон про охорону даних» – 1978р.

Оскільки масова інформатизація і комп'ютеризація почалася в Україні з запізненням років на 15, а традиція поважати чужі секрети і шанувати чужу інтелектуальну власність не з'явилася і понині, то і законодавство в даній сфері помітно слабкіше західного. Така ситуація характерна для більшості країн колишнього СРСР, і правові акти в цій сфері в колишніх союзних республік досить схожі. Оскільки в Україні законів про захист інформації не дуже багато, ми маємо можливість конспективно привести головні думки кожного з них [3; 5].

Закон України **«Про інформацію»** №2657 від 2.10.1992 р. Декларує загальні принципи одержання, використання, поширення і збереження інформації, закріплює право особистості на інформацію у всіх сферах життя. Визначає статуси учасників інформаційних відносин, регулює доступ до інформації і забезпечує її охорону, захищає особистість і суспільство від недостовірної інформації. Документ є ідеологічною основою для всіх наступних документів у сфері захисту інформації.

Закон України **«Про державну таємницю»** №3855-12 від 21.01.1994 р. Регламентує інформаційні відносини, зв'язані з доступом до державної секретної інформації:

– Державну таємницю можуть складати відомості зі сфери оборони, економіки, міждержавних відносин, державної безпеки й охорони правопорядку.

– Забороняється засекречувати в будь-якій формі відомості про стихійні нещастя, катастрофи, екологію, рівні добробуту народу, стані правопорядку, про незаконні дії органів влади, а також будь-які інші зведення, засекречування яких порушує конституційні права і волю громадян.

– Інформація засекречується державними експертами з питань секретів, якими є президент, спікер парламенту, прем'єр-міністр, а також спеціально призначені президентом посадові особи.

– Недержавні структури, що мають намір працювати з інформацією, що складає державну таємницю, повинні ліцензуватися Державним комітетом з питань держсекретів.

При дотриманні цілого ряду формальностей людина по службовій необхідності може одержати допуск до інформації, що складає державну таємницю. При цьому вона перевіряється компетентними органами на благонадійність, обмежується в деяких правах і дає підписку про «нерозголошення отриманих відомостей». У виняткових випадках допуск може бути оформлений на іноземця.

Закон України «**Про захист інформації в інформаційно-телекомунікаційних системах**» №81/94-ВС від 05.07.1994 р. Дає визначення основних термінів у сфері автоматизованих систем (АС), і регламентує відносини між сторонами в процесі обробки інформації в АС, установлює загальні вимоги до захисту інформації в АС, порядок міждержавних відносин у сфері захисту інформації в АС:

– Суб'єктами відносин у процесі обробки інформації в АС є власник інформації, власник АС, користувач інформації, користувач АС.

– Після процесу обробки інформації в АС, отриманий продукт є власністю користувача АС, що її обробив, якщо інше не передбачене договором між ним і власником інформації.

– Якщо власником оброблюваної інформації є держава, то власник АС повинний забезпечити захист інформації відповідно до державних стандартів.

– Власник АС є її адміністратором і організатором роботи користувачів АС. Власник АС зобов'язаний інформувати власника інформації і користувача АС про методи спілкування з АС, а також методи захисту інформації в його системі.

– Захист інформації в АС забезпечується програмними, апаратними й іншими засобами, що повинні сертифікуватися, якщо на них планується обробляти державну інформацію секретного характеру.

– Особи, винні у втраті інформації внаслідок неграмотної побудови систем захисту АС, несуть дисциплінарну, адміністративну, кримінальну відповідальність або матеріально компенсують нанесений збиток.

Закон України **«Про електронні документи і електронний документообіг»** – 2003 р., Закон України **«Про електронний цифровий підпис»** – 2003 р., Закон України **«Про телекомунікації»** – 2004 р.

Положення **«Про технічний захист інформації в Україні»**, затверджене Указом Президента України від 27.09.1999р. №1229/99, розкриває порядок технічного захисту інформації (ТЗІ), що є важливою для держави, суспільства й особистості, охорона якої забезпечується державою відповідно до законодавства. Під технічним захистом інформації розуміється діяльність, спрямовану на забезпечення інженерно-технічними мірами конфіденційності, цілісності і доступності інформації.

Головним органом по здійсненню державної політики технічного захисту інформації визначений Департамент спеціальних телекомунікаційних систем і. захисту інформації (ДСТСЗІ) Служби безпеки України, що є спадкоємцем Державного комітету по питаннях державних секретів, що виконував до цього функції технічного захисту. Даним положенням регламентуються всі основні питання, зв'язані з організацією технічного захисту інформації в органах державної влади, органах місцевого самоврядування, органах керування Збройних Сил України і т.д. (органах, у відношенні яких здійснюється ТЗІ).

У положенні визначені завдання ДСТСЗІ, завдання органів, у відношенні яких здійснюється ТЗІ, причому до сфери компетенції останніх відноситься забезпечення технічного захисту інформації відповідно до вимог нормативно-правових актів з питань ТЗІ, видання в межах своїх повноважень нормативно-правових актів сфери ТЗІ, здійснення контролю за станом ТЗІ.

До основних завдань інших суб'єктів системи ТЗІ, визначених у Положенні, відносяться дослідження погроз інформації на об'єктах, функціонування яких зв'язане із інформацією, що підлягає охороні; створення і виробництво засобів забезпечення ТЗІ; розробка, упровадження, супровід комплексів ТЗІ; підвищення кваліфікації фахівців сфери ТЗІ.

Розробка, впровадження, атестація й експлуатація комплексів ТЗІ для власних потреб здійснюється відповідними підрозділами органів, у відношенні яких здійснюється ТЗІ або підприємствами, установами, організаціями, на які у встановленому порядку покладене забезпечення ТЗІ, при наявності в них відповідного дозволу. Оцінка захищеності інформації здійснюється шляхом атестації або експертизи комплексів ТЗІ, а також інспекційних перевірок.

Положення «**Про порядок здійснення криптографічного захисту інформації в Україні»**, затверджене Наказом Президента України № 505 від 22.05.98 р. (зі змінами, внесеними Наказами від 15.09.98 р. № 1019 і № 1229 від 27.09.99 р.) визначає порядок здійснення криптографічного захисту інформації й обмеженим доступом, розголошення якої може заподіяти шкода державі, суспільству або особистості.

– Державну політику в сфері криптографічного захисту реалізує Департамент спеціальних телекомунікаційних систем і захисту інформації Служби безпеки України (далі Департамент).

– Державні організації створюють, купують, продають, ввозять, вивозять, використовують криптосистеми за узгодженням з Департаментом. Подібна діяльність недержавних структур, що мають відповідну ліцензію, здійснюється на основі діючого законодавства.

– Інформація, що містить державну таємницю, повинна захищатися дозволеними криптосистемами, що знаходяться в державній власності і мають сертифікат відповідності. Службова інформація може захищатися криптосистемами будь-якої власності.

– Користувачі допускаються до роботи з криптосистемами тільки при наявності в них допуску до державного таємниці.

– Міністерства й інші органи центральної влади видають свої інструкції на основі даного Положення.

Відповідальність за порушення Положення настає **відповідно** до діючого законодавства.

5.3.2. Кримінальний кодекс України має ряд статей, регламентуючих відповідальність за розголошення або крадіжку секретної інформації як звичайними, так і електронними засобами.

До найбільш характерних особливостей комп'ютерних злочинів можна віднести такі: складність визначення розміру збитків, значні фінансові витрати на проведення розслідування. До зазначеного слід додати невелику кількість фахівців, які мають необхідні для їх розкриття навички та кваліфікацію, а також неврегульованість законом багатьох питань. У залежності від виду правопорушень особи, винні у вчиненні комп'ютерних злочинів несуть кримінальну, адміністративну, цивільно-правову, дисциплінарну або матеріальну відповідальність.

Комп'ютерний злочин – це передбачена законом суспільно небезпечна дія, що посягає на встановлений в суспільстві порядок інформаційних відносин та скоюється з використанням електронно-обчислювальних машин (комп'ютерів), систем та комп'ютерних мереж. Тобто, об'єктом злочину виступають інформаційні відносини у суспільстві, що охороняється законом, а предметом – електронно-обчислювальні машини (комп'ютерів), системи та комп'ютерні мережі, а також комп'ютерна інформація, що обробляється за їх допомогою.

Новий Кримінальний Кодекс України був прийнятий 5 квітня 2001 року, який набув чинності з 1 вересня 2001 року. Він містить, на відміну від попереднього, окремий розділ (розділ 16) «Злочини у сфері використання електронно-обчислювальних машин (комп'ютерів), систем та комп'ютерних мереж» – повністю присвячений класифікації та встановленню відповідальності за вчинення комп'ютерних злочинів.

До розділу входять три статті:

Стаття 361. Незаконне втручання в роботу електронно-обчислювальних машин (комп'ютерів), систем та комп'ютерних мереж.

Незаконне втручання, що привело до перекручення чи знищення інформації, а також розповсюдження комп'ютерного вірусу шляхом застосування програмних і технічних засобів, призначених для незаконного проникнення в ці машини. Втручання в мережу, що привело до перекручення, блокування або порушення порядку маршрутизації. Карається визначеним шрафом або виправними роботами на певний період.

Стаття 362. Викрадення, привласнення, вимагання комп'ютерної інформації або заволодіння нею шляхом шахрайства чи зловживання

службовим становищем. Карається визначеним шрафом або виправними роботами на певний період.

Стаття 363. Порушення правил експлуатації автоматизованих електронно-обчислювальних систем.

Порушення правил, яке спричинило викрадення, перекручування, знищення, незаконного копіювання комп'ютерної інформації, засобів її захисту або істотне порушення роботи таких машин. Карається визначеним шрафом або виправними роботами на певний період.

Стаття 57. Шпигунство. Передача або збирання з метою передачі іноземній державі, іноземній організації або їхнім представникам відомостей, що містять державну або військову таємницю, здійснену іноземцем або особою без громадянства на шкоду інтересам України, карається позбавленням волі на певний термін з конфіскацією майна.

Стаття 67. Розголошення державної таємниці.

Розголошення відомостей, що складають державну таємницю, особою, якій ці відомості були довірені або стали відомі по службовій необхідності, при відсутності ознак державної зради або шпигунства, карається позбавленням волі.

Стаття 136. Порушення авторських прав. Випуск під своїм ім'ям або інше присвоювання авторства на чужий здобуток науки, культури і мистецтва, незаконне копіювання і поширення такого добутку – карається виправними роботами або штрафом.

Стаття 137. Порушення прав на об'єкт права інтелектуальної власності. Присвоєння авторства на чуже відкриття, винахід, корисну модель, промисловий зразок або раціоналізаторську пропозицію або розголошення їхнього змісту без згоди автора до їхньої офіційної публікації – карається виправними роботами або штрафом.

Стаття 198-1. Порушення роботи автоматизованих систем. Навмисне вторгнення в роботу АС, що привело до змін або знищенню інформації або носіїв Інформації, або поширення програмних і технічних засобів, призначених для незаконного проникнення в АС і здатних привести до зміни або знищення інформації або її носіїв – карається позбавленням волі, або штрафом.

Необхідно підкреслити і наявність певних невирішених на сьогодні проблем стосовно визначення «комп'ютерні злочини» та їх класифікації у чинному законодавстві.

У КК України не чітко виписано такі правопорушення:

а) виробництво, продаж, постачання, імпорт, розповсюдження:

– пристроїв (і комп'ютерні програми, що розроблені чи адаптовані повністю або вибірково) з метою здійснення неправомірного доступу до комптерних систем, неправомірного перехоплення та модифікації даних або втручання в роботу КС;

– комп'ютерних паролів, кодів доступу або аналогічних даних, які надають можливість неправомірного доступу до всієї КС або будь-якої її частини з наміром використання їх для скоєння правопорушень;

б) володіння зазначеними у попередньому пункті атрибутами з наміром використання їх у подальшому для неправомірного доступу до комп'ютерних систем або неправомірного перехоплення даних.

5.3.3. Нормативні документи системи технічного захисту інформації

Крім законів та положень впровадженно багато нормативних документів, які регламентують певні аспекти діяльності у сфері захисту інформації [6]. До них належать:

НД ТЗІ 1.1-002-99: Загальні положення по захисту інформації в комп'ютерних системах від несанкціонованого доступу.

НД ТЗІ 1.1-003-99: Термінологія в області захисту інформації в комп'ютерних системах від несанкціонованого доступу.

НД ТЗІ 2.5-004-99: Критерії оцінювання захищеності інформації в комп'ютерних системах від несанкціонованого доступу.

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

НД ТЗІ 3.7-001-99: Методичні вказівки з розробки технічного завдання на створення комплекної системи захисту інформації в автоматизованій системі.

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

НД ТЗІ 3.6-001-2000: Технічний захист інформації. Комп'ютерні системи. Порядок створення, впровадження, супроводження та модернізації засобів технічного захисту інформації від несанкціонованого доступу.

НД ТЗІ 2.5-008-02: Вимоги із захисту конфіденційної інформації від несанкціонованого доступу під час оброблення в автоматизованих системах класу 2.

НД ТЗІ 2.5-010-03: Вимоги до захисту інформації веб-сторінки від несанкціонованого доступу.

НД ТЗІ 3.7-003-05: Порядок проведення робіт із створення комплекної системи захисту інформації в інформаційно-телекомунікаційній системі.

Нормативний документи НД ТЗІ 2.5-004-99 «Критерії оцінювання захищеності інформації в комп'ютерних системах від несанкціонованого доступу» містить специфікації вимог до реалізації послуг безпеки. Вони утворюють систему із п'яти груп. Це критерії: **конфіденційності (К), цілісності (Ц), доступності (Д), спостережності (Н), гарантій (Г).** Критерії перших чотирьох груп – це функціональні критерії, що визначають, які послуги безпеки здатна надавати оцінювана система. До надання кожної із послуг висувається ряд вимог, за якими визначається рівень надання цієї послуги. Множина рівнів послуг, які надає система, складає функціональний профіль. Остання група – визначає рівень адектватності та коректності реалізації послуг безпеки, тобто фактично визначає рівень довіри до реалізації цих функцій.

Об'єктом оцінювання є комп'ютерна система, тобто сукупність апаратних та програмних засобів, поданих на оцінювання захищеності інформації, яку вони обробляють.

Згідно із 5 критеріями розрізнять рівні цих послуг.

К: КД-1, КД-2, КД-3, КД-4 (довірча конфіденційність); КА-1, КА-2, КА-3, КА-4 (адміністративна конфіденційність); КО-1 (повторне використання об'єктів); КК-1, КК-2, КК-3 (аналіз прихованих каналів); КВ-1, КВ-2, КВ-3, КВ-4 (конфіденційність під час обміну).

Ц: ЦД-1, ЦД-2, ЦД-3, ЦД-4 (довірча цілісність); ЦА-1, ЦА-2, ЦА-3, ЦА-4 (адміністративна цілісність); ЦО-1, ЦО-2 (відкат); ЦВ-1, ЦВ-2, ЦВ-3 (цілісність під час обміну).

Д: ДР-1, ДР-2 (використання ресурсів); ДС-1, ДС-2, ДС-3 (стійкість до відмов); ДЗ-1, ДЗ-2, ДЗ-3 («гаряча» заміна); ДВ-1, ДВ-2, ДВ-3 (відновлення після збоїв).

Н: НР-1, НР-2, НР-3, НР-4, НР-5 (реєстрація); НИ-1, НИ-2, НИ-3 (ідентифікація й автентифікація); НК-1, НК-2 (достовірний канал); НО-1, НО-2, НО-3 (розподіл обов'язків); НЦ-1, НЦ-2, НЦ-3 (цілісність комплексу засобів захисту); НТ-1, НТ-2, НТ-3 (самотестування); НВ-1, НВ-2, НВ-3 (ідентифікація й автентифікація під час обміну); НА-1, НА-2 (автентифікація відправника);НП-1, НП-2 (автентифікація одержувача).

Г: Г1 – Г7 (1 – найнижчий, 7 – найвищий).

У документі НД ТЗІ 2.5-005-99 «Класифікація автоматизованих систем і стандартні функціональні профілі захищеності оброблюваної інформації від несанкціонованого доступу» визначено три класи автоматизованих систем:

– Одномашинний однокористувацький комплекс (клас 1).
– Локалізований багатомашинний багатокористувацький комплекс (клас 2).
– Розподілений багатомашинний багатокористувацький комплекс (клас 3).

Для даних класів реалізовано стандартні функціональні профілі: конфіденційність (К), цілісність (Ц), доступність (Д), конфіденційність і цілісність (КЦ), конфіденційність і доступність (КД), цілісність і доступність (ЦД), конфіденційність, цілісність і доступність (КЦД). У рамках кожного профілю запропоновано кілька різних профілів, що відрізняються рівнем захищеності.

Наприклад, для АС класу 2 визначено такий стандартний профіль для технології, що вимагає підвищених вимог до забезпечення конфіденційності, цілісності і доступності оброблюваної інформації:

2.КЦД.2а={КД-2, КА-2, КО-1, ЦД-1, ЦА-2, ЦО-1, ДР-1, ДС-1, ДЗ-1, ДВ-1, НР-2, НК-1, НЦ-2, НТ-2, НИ-2, НО-2}.

Мінімальним достатнім рівнем гарантій реалізації комплекної системи захисту АС класу 2 є рівень Г-2.

5.4. Міжнародний стандарт ISO/IEC 15408

Роботу над новим стандартом із оцінювання безпеки інформаційних технологій (ІТ) було розпочато 1990 року під крівництвом Міжнародної організації зі стандартизації (International Organization for Standartization – ISO). У 1993 році організації зі стандиртизації та забезпечення безпеки США,

Канади, Великої Британії, Франції, Німеччини, Нідерландів об'єднали свої зусилля щодо створення єдиних критеріїв оцінювання безпеки інформаційних технологій і розробили у рамках цього проекту документ, який отримав назву «**Загальні критерії»** (ЗК). У 1999 році було прийнято Міжнародний стандарт **ISO/IEC 15408 «Критерії оцінювання безпеки інформаційних технологій»** (Evaluation Criyeria for IT Security). У 2005 році з'явилася нова версія ISO/IEC 15408: 2005.

Стандарт ISO/IEC 15408 регламентує усі стадії розробки, кваліфікаційного аналізу та експлуатації продуктів інформаційних технологій. Він пропонує досить складну концепцію процесу розробки і кваліфікаційного аналізу продуктів ІТ. У застосуванні до оцінки безпеки продуктів інформаційних технологій (ПІТ) стандарт є по суті метазасобом, що задає систему понять, в термінах яких повинна проводитись оцінка. Стандарт містить відносно повний каталог вимог безпеки (функціональних та гарантій), але не надає конкретних наборів вимог та критеріїв для тих чи інших типів ПІТ, виконання яких необхідно перевіряти.

Вимоги та критерії формулюються у профілях захисту та завданнях із безпеки. Саме офіційно прийняті профілі захисту утворюють побудовану на основі ЗК нормативну базу, що використовується на практиці у сфері інформаційної безпеки.

Згідно із концепцією «Загальних критеріїв» вимоги до безпеки об'єкта оцінювання поділяють на дві категорії: функіональні вимоги (вимоги до підсистеми ідентифікації, автентифікації, реєстрації); вимоги адекватності або гарантованості.

У стандарті використано єдину термінологію для визначення функціональних вимог та вимог гарантованості та представлені таблиці зв'язку класів, вимог, кількості сімейств та опису функціональних вимог:

– клас – найбільш загальна група вимог безпеки;
– сімейство – член класу, який визначає групу вимог, що забезпечують виконання певної частини цілей безпеки;
– компонент – член сімейства, який визначає мінімальний набір вимог безпеки для включеннядо структур, визначених у «Загальних критеріях»;
– елемент – неподільна складова компонента.

Структура «Загальних критеріїв»

Розділ **Представлення і загальна модель** – визначає загальну концепцію й принципи оцінки безпеки ІТ і подає загальну модель оцінки, а також конструкції для:

– формування задач захисту ІТ;
– вибору й визначення вимог безпеки ІТ;
– опису специфікацій високого рівня для продуктів і систем.

Розділ **Вимоги до функцій безпеки** – встановлює набір функціональних компонентів як стандартний шлях формулювання функціональних вимог до об'єктів оцінки.

Розділ **Вимоги гарантій безпеки** – включає компоненти вимог гарантій оцінки, а також рівні гарантій оцінки, які визначають ранжирування за ступенем задоволення вимог

Розділ **Визначені профілі захисту** (був передбачений з самого початку, але після версії 1.0 ЗК був винесений за межі стандарту) – містить приклади профілів захисту, що включають функціональні вимоги безпеки та вимоги гарантій оцінки, що були ідентифіковані у вихідних критеріях (ITSEC, CTCPEC, FC, TCSEC), а також інші профілі.

Задачі захисту (Security Objectives) – потреба споживачів продукту ІТ:

– у протистоянні заданій множині загроз безпеці;

– у необхідності реалізації політики безпеки.

Профіль захисту (Protection Profile)

– Спеціальний нормативний документ, що містить: задачі захисту, функціональні вимоги, вимоги адекватності, їхнє обґрунтування.

– Служить керівництвом для розроблювача при створенні завдання з безпеки.

Завдання з безпеки (Security Target)

– Спеціальний нормативний документ, що містить: задачі захисту, функціональні вимоги, вимоги адекватності, загальні специфікації засобів захисту, їхнє обґрунтування.

– У ході кваліфікаційного аналізу служить як опис продукту ІТ

6. ІДЕНТИФІКАЦІЯ ТА АУТЕНТИФІКАЦІЯ

6.1. Паролі

На сьогоднішній день пароль є найбільш прийнятним і тому найбільше часто використовуваним засобом установлення дійсності, заснованим на знаннях суб'єктів доступу.

У будь-якій критичній системі помилки людини-оператора є чи ледве не найдорожчими і розповсюдженими. У випадку криптосистем, непрофесійні дії користувача зводять нанівець самий стійкий криптоалгоритм, коректну його реалізацію і застосування. У першу чергу це зв'язано з вибором паролів. Очевидно, що короткі або осмислені паролі легко запам'ятовуються людиною, але вони набагато простіші для розкриття. Використання довгих і безглуздих паролів, безумовно, краще з погляду криптостойкості, але людина звичайно не може їх запам'ятати і записує, що легко знаходть зловмисник. Саме з того, що недосвідчені користувачі звичайно вибирають або короткі, або осмислені паролі, існують два методи їхнього розкриття: атака повним перебором і атака по словнику.

Захищеність пароля при його підборі залежить, у загальному випадку, від швидкості перевірки паролів і від розміру повної множини можливих паролів, що, у свою чергу, залежить від довжини пароля і розміру застосовуваного алфавіту символів. Крім того, на захищеність сильно впливає реалізація парольного захисту. У зв'язку з різким ростом обчислювальних потужностей атаки повним перебором мають набагато більше шансів на успіх, ніж раніше. Крім того, активно використовуються розподілені обчислення, це дозволяє багаторазово скоротити час злому.

Чим більша довжина пароля, тим більшу безпеку буде забезпечувати система, тому що будуть потрібні великі зусилля для його відгадування. Це можна представити в термінах очікуваного часу розкриття пароля або очікуваного безпечного часу. Очікуваний безпечний час ($T_б$) — половина добутку числа можливих паролів і часу, необхідного для того, щоб спробувати кожен пароль з послідовності запитів. Представимо це у виді формули:

$$T_б = \frac{A^S \bullet t}{2},$$

де t – час, необхідний на спробу введення пароля, рівний E/R; E – число символів у переданому повідомленні при спробі одержати доступ (включаючи пароль і службові символи); R – швидкість передачі (символи/хв) у лінії зв'язку; S – довжина пароля; A – число символів в алфавіті, з яких складається пароль. Якщо після кожної невдалої спроби підбору автоматично передбачається десятисекундна затримка, то безпечний час різко збільшується.

Для перевірки уразливості паролів використовуються спеціальні контролери паролів. Наприклад, відомий контролер Кляйна, здійснює спроби злому пароля шляхом перевірки використання як пароля вхідного імені користувача, його ініціалів і їхніх комбінацій, перевірки використання слів з

різних словників, починаючи від найбільш уживаних, перевірки різних перестановок слів. Перевірка паролів в обчислювальних мережах за допомогою контролера Кляйна показала досить високі результати – більшість користувачів використовують прості паролі.

Приведений аналіз дозволяє сформулювати наступні правила зниження уразливості паролів і спрямовані на протидію відомим атакам на них:

– символи пароля при їхньому введенні не повинні з'являтися в явному виді;

– обмежується кількість спроб уведення пароля;

– при передачі по каналах зв’язку паролі повинні шифруватися;

– паролі повинні зберігатися в пам’яті тільки в зашифрованому вигляді у файлах, недоступних користувачам;

– користувач повинний мати можливість самому змінювати пароль;

– адміністратор не повинний знати паролі користувачів, хоча може їх змінювати;

– паролі повинні періодично мінятися;

– установлюються терміни дії паролів, після закінчення яких треба зв’язатися з адміністратором.

– розширювати застосовуваний у паролі алфавіт – використовувати прописні і малі літери латинського, російського, українського алфавітів, цифри і знаки;

– не використовувати в паролі осмислені слова;

– не використовувати повторювані групи символів;

– не застосовувати паролі довжиною менш 6–8 символів, тому що запам’ятати їх не представляє великої праці, а пароль саме потрібно запам'ятовувати, а не записувати. По тій же причині не має змісту вимагати довжину неосмисленого пароля більш 15 символів, тому що запам’ятати його нормальній людині практично неможливо;

– не використовувати той самий пароль у різних системах, тому що при компрометації одного пароля постраждають усі системи;

– перевіряти паролі перед їхнім використанням контролерами паролів.

Для складання пароля можна дати рекомендації, якими користуватися треба дуже обережно:

– вибирати кілька рядків з пісні або поеми і використовувати першу (або другу) букву кожного слова — при цьому пароль повинний мати велику довжину (більш 15 символів), інакше потрібно змінювати регістри букв, застосовувати латинські букви замість російських або навпаки, можна вставляти цифри і знаки;

– замінювати в слові із семи-восьми букв одну приголосну й одну або дві голосні на знаки або цифри. Внаслідок чого отримаємо слово-абракадабру, що звичайно вимовне і тому легко запам’ятовується.

6.2. Ідентифікація і перевірка достовірності

З кожним об’єктом КС зв’язана деяка інформація, що однозначно ідентифікує його. Це може бути число, рядок символів, алгоритм, що визначає даний об’єкт. Цю інформацію називають ідентифікатором об’єкта. Якщо об’єкт має деякий ідентифікатор, зареєстрований у мережі, він називається законним (легальним) об’єктом; інші об’єкти відносяться до незаконних (нелегальних).

Ідентифікація об’єкту – одна з функцій підсистеми захисту. Ця функція виконується в першу чергу, коли об’єкт робить спробу отримати доступ до КС. Якщо процедура ідентифікації завершується успішно, даний об’єкт вважається законним для даної системи. Наступний крок – **аутентифікації** об'єкта (перевірка дійсності об’єкта). Ця процедура встановлює, чи є даний об’єкт саме тим, ким він себе повідомляє. Після того як об’єкт ідентифікований і підтверджена його дійсність, можна установити сферу його дії і доступні йому ресурси КС. Таку процедуру називають наданням повноважень (**авторизацією**). Перераховані три процедури ініціалізації є процедурами захисту і відносяться до одного об’єкта КС.

Під час захисту каналів передачі даних аутентифікація об’єктів означає взаємне встановлення дійсності об’єктів, що зв’язуються між собою по лініях зв’язку. Процедура підтвердження дійсності виконується звичайно на початку сеансу в процесі встановлення з’єднання абонентів. Термін «з’єднання» указує на логічний зв’язок між двома об’єктами мережі. Ціль даної процедури – забезпечити впевненість, що з’єднання встановлене з законним об’єктом і всією інформацією дійде до місця призначення.

Після того, як з’єднання встановлене, необхідно забезпечити виконання вимог захисту при обміні повідомленнями:

а) одержувач повинний бути упевнений у дійсності джерела даних;

б) одержувач повинний бути упевнений у дійсності переданих даних;

в) відправник повинний бути упевнений у доставці даних одержувачу;

г) відправник повинний бути упевнений у дійсності доставлених даних.

Для виконання вимог а) і б) засобом захисту є **цифровий підпис**. Для виконання вимог в) і г) відправник повинний одержати **повідомлення про вручення** за допомогою пошти, що засвідчує це. Засобом захисту в такій процедурі є цифровий підпис підтверджуючого відповідного повідомлення, що у свою чергу є доказом пересилання вихідного повідомлення.

Якщо ці чотири вимоги реалізовані в КС, то гарантується захист даних при їхній передачі по каналі зв’язку і забезпечується функція захисту. У цьому випадку відправник не може заперечувати ні факту посилки повідомлення, ні його змісту, а одержувач не може заперечувати ні факту одержання повідомлення, ні дійсності його змісту.

Є декілька способів підтвердження достовірності користувача:

1) **Наперед визначена інформація,** що знаходиться в користувача: пароль, особистий ідентифікатор, номер, домовленість про використання спеціальних закодованих фраз.

2) **Елементи апаратного забезпечення**, що знаходяться в розпорядженні користувача: ключі, магнітні картки, мікросхеми.

3) **Характерні особисті особливості користувача**: відбитки пальців, малюнок сітківки ока, розміри фігури, тембр голосу та інші більш складні медичні і біохімічні властивості.

4) **Характерні прийоми і риси поведінки користувача в режимі реального часу**: особливості динаміки, стиль роботи на клавіатурі, швидкість читання, вміння використовувати маніпулятори та інше.

5) **Звички, навички, і знання користувача, обумовлені освітою, культурою, навчанням, виховання** та інше.

Типові схеми ідентифікації й аутентифкації користувача

Розглянемо структури даних і протоколи ідентифікації й аутентифікації користувача. Припустимо, що в комп'ютерній системі зареєстровано n користувачів. Нехай i-й аутентифікуючий об'єкт i-го користувача містить два інформаційних поля: ID_i – незмінний ідентифікатор i-го користувача, що є аналогом імені і використовується для ідентифікації користувача; K_i – аутентифікуюча інформація користувача, що може змінюватися і служить для аутентифікації (наприклад, пароль P_i.=K_i). Описана структура відповідає практично будь-якому ключовому носієві інформації, використовуваному для впізнання користувача. Наприклад, для носіїв пластикових карт виділяється незмінна інформація ID_i, первинної персоналізації користувача й об'єкт у файловій структурі карти, що містить K_i . Сукупну інформацію в ключовому носії можна назвати первинною аутентифікуючою інформацією i-го користувача. Очевидно, що внутрішній аутентифікуючий об'єкт не повинний існувати в системі тривалий час (більше часу роботи конкретного користувача). Для тривалого збереження варто використовувати дані в захищеній формі.

Схема 1. У комп'ютерній системі виділяється об'єкт-еталон для ідентифікації й аутентифікації користувачів. Структура об'єкта-еталона для схеми 1 показана в табл.6.1. Тут $E_i=F(ID_i, K_i)$, де F-функція, що має властивість «невідновленості» значення K_i по E_i і ID_i .

Таблиця 6.1.

Структура об'єкта-еталона для схеми 1

Номер користувача	Інформація для ідентифікації	Інформація для аутентифікації
1	ID_1	E_1
2	ID_2	E_2
N	ID_N	E_N

Протокол ідентифікації й аутентификації для схеми 1.

1. Користувач пред'являє свій ідентифікатор ID.

2. Якщо ID не збігається з жодним ID_i, зареєстрованим у комп'ютерній системі, то ідентифікація відкидається – користувач не допускається до роботи, інакше існує $ID_i = ID$ і користувач пройшов ідентифікацію.

3. Суб'єкт аутентифікації запитує в користувача його аутентифікатор K_i.

4. Суб'єкт аутентифікації обчислює значення $Y=F(ID_i, K_i)$.

5. Суб'єкт аутентифікації робить порівняння значень Y і E_i. Під час збігу цих значень установлюється, що даний користувач успішно аутентифікований у системі. Інформація про цього користувача передається в програмні модулі, що використовують ключі користувачів (тобто в систему шифрування, розмежування доступу і т.д.). У протилежному випадку аутентифікація відкидається – користувач не допускається до роботи.

Дана схема ідентифікації й аутентифікації користувача може бути модифікована і має кращі характеристики в порівнянні зі схемою 1.

На відміну від схеми 1, у схемі 2 значення $E_i = F(S_i, K_i)$, де S – випадковий вектор, що задається при створенні ідентифікатора користувача, тобто при створенні рядка, необхідної для ідентифікації й аутентифікації користувача; F – функція, що має властивість «невідновності» значення K_i за E_i і S_i.

Схема 2. У комп'ютерній системі виділяється модифікований об'єкт-еталон, структура якого показана в табл.6.2.

Таблиця 6.2.

Структура модифікованого об'єкта-еталона

Номер користувача	Інформація для ідентифікації	Інформація для аутентифікації
1	ID_1, S_1	E_1
2	ID_2, S_2	E_2
N	ID_N, S_N	E_N

Протокол ідентифікації й аутентифікації для схеми 2.

1. Користувач пред'являє свій ідентифікатор ID.

2. Якщо ID не збігається з жодним ID_i, зареєстрованим у комп'ютерній системі, то ідентифікація відкидається – користувач не допускається до роботи, інакше існує $ID_i=ID$ і установлюється, що користувач, що називався користувачем пройшов ідентифікацію.

3. За ідентифікатором ID_i виділяється вектор S_i.

4. Суб'єкт аутентифікації запитує в користувача аутентифікатор K.

5. Суб'єкт аутентифікації обчислює значення $Y= F(S_i, K)$.

6. Суб'єкт аутентифікації робить порівняння значень Y і E_i. При збігу цих значень установлюється, що даний користувач успішно аутентифікований у системі. У противному випадку аутентифікація відкидається – користувач не допускається до роботи.

Друга схема аутентифікації застосовується в ОС UNIX. Як ідентифікатор ID використовується ім'я користувача – login, у якості аутентифікатора K_i – пароль користувача – password, функція F являє собою алгоритм шифрування DES. Еталони для ідентифікації й аутентифікації утримуються у файлі Etc/passwd.

Слід зазначити, що необхідною вимогою стійкості схем аутентифікації до відновлення інформації K_i є випадковий рівноймовірний вибір K_i із множини можливих значень. Системи парольної аутентифікації мають знижену стійкість, оскільки в них вибір аутентифікуючої інформації походить з відносно невеликої множини осмислених слів. Потужність цієї множини визначається ентропією відповідної мови.

Особливості застосування пароля для аутентифікації користувача

Традиційно кожен законний користувач комп'ютерної системи одержує ідентифікатор і/або пароль. На початку сеансу роботи користувач пред'являє свій ідентифікатор системі, що потім запитує в користувача пароль.

Найпростіший метод підтвердження дійсності з використанням пароля заснований на порівнянні пароля, що представляється користувачем, P_A з вихідним значенням P_A', що зберігається в комп'ютерному центрі (рис.6.1). Оскільки пароль повинний зберігатися в таємниці, він повинний шифруватися перед пересиланням по незахищеному каналі. Якщо значення P_A і P_A' збігаються, то пароль P_A вважається справжнім, а користувач – законним.

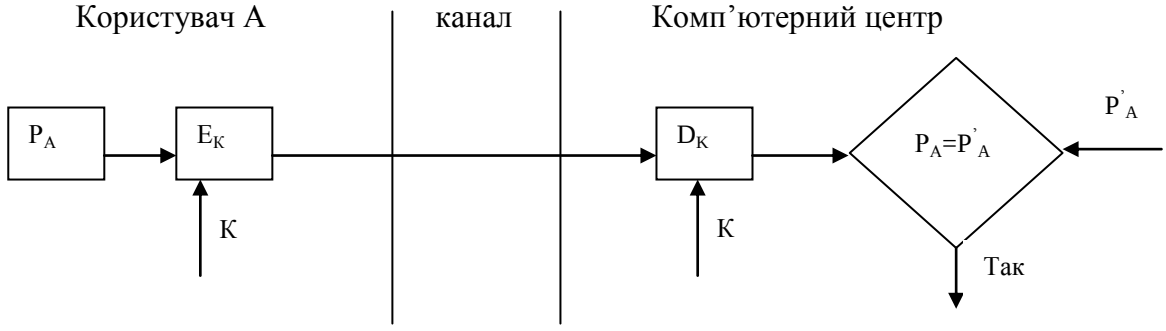

Рисунок 6.1 – Схема простої аутентифікації за допомогою пароля

Якщо хто-небудь, що не має повноважень для входу в систему, довідається яким-небудь образам пароль і ідентифікаційний номер законного користувача, він одержує доступ у систему.

Зауваження. Користувачу може бути заданий список паролів. Користувач використовує перший пароль при першому входженні в систему, другий при другому і т. д. При кожному входженні він перевіряє чи не використовував його пароль хто-небудь. Ця модель називається моделлю із змінним паролем.

Недоліки: користувач повинен пам'ятати довгий список паролів або тримати його при собі; у випадку помилки, користувач не знає повинний він використовувати той же пароль чи наступний.

Іноді одержувач не повинний розкривати вихідну відкриту форму пароля. У цьому випадку відправник повинний пересилати замість відкритої форми пароля відображення пароля, одержуване з використанням односторонньої функції $\alpha(\cdot)$ пароля. Це перетворення повинне гарантувати неможливість розкриття супротивником пароля за його відображенням, тому що супротивник наштовхується на нерозв'язну числову задачу. Наприклад, функція $\alpha(\cdot)$ може бути визначена в такий спосіб: $\alpha(P) = E_P(ID)$, де P – пароль відправника; ID –

ідентифікатор відправника; E_P – процедура шифрування, виконувана з використанням пароля P в якості ключа. Такі функції особливо зручні, якщо довжина пароля і ключа однакові. У цьому випадку підтвердження дійсності за допомогою пароля складається з пересилання одержувачеві відображення $\alpha(P)$ і порівняння його з попередньо обчисленим і збереженим еквівалентом $\alpha'(P)$.

На практиці паролі складаються тільки з декількох букв, щоб дати можливість користувачам запам'ятати їх. Короткі паролі уразливі до атаки повного перебору усіх варіантів. Для того щоб запобігти такій атаці, функцію $\alpha(P)$ визначають інакше, а саме, $\alpha(P) = E_{P \oplus K}(ID)$, де K и ID – відповідно ключ і ідентифікатор відправника. Очевидно, значення $\alpha(P)$ обчислюється заздалегідь і зберігається у виді $\alpha'(P)$ в ідентифікаційній таблиці в одержувача (рис.6.2). Підтвердження дійсності складається з порівняння двох відображень пароля $\alpha(P_A)$ і $\alpha'(P_A)$ і визнання пароля P_A, якщо ці відображення рівні Звичайно, кожний, хто одержить доступ до ідентифікаційної таблиці, може незаконно змінити її вміст, не побоюючись, що ці дії будуть виявлені.

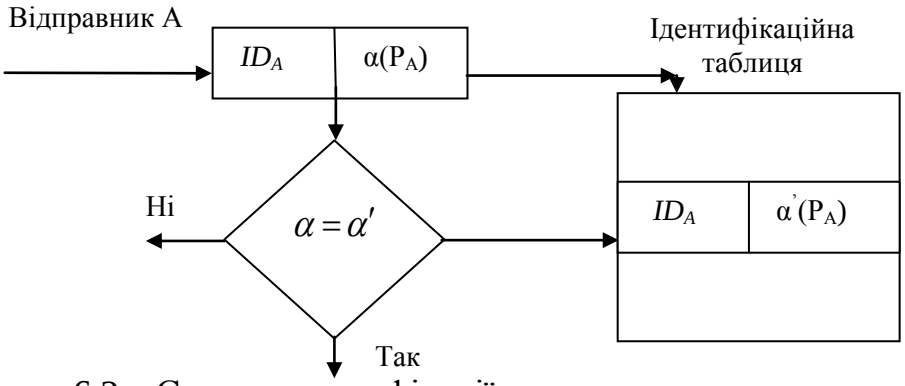

Рисунок 6.2 – Схема аутентифікації за допомогою пароля з використанням ідентифікаційної таблиці

6.3. Біометрична ідентифікація й аутентифікація користувача

Процедури ідентифікації й аутентифікації користувача можуть базуватися не тільки на секретній інформації, якою володіє користувач (пароль, секретний ключ, персональний ідентифікатор і т.д.). Останнім часом усе більше поширення одержує біометрична ідентифікація й аутентифікація користувача, що дозволяє впевнено ідентифікувати потенційного користувача шляхом виміру фізіологічних параметрів і характеристик людини, особливостей його поводження.

Відзначимо основні переваги біометричних методів ідентифікації й аутентифікації користувача в порівнянні з традиційними :

– високий ступінь вірогідності ідентифікації за біометричними ознаками через їхню унікальність;

– невіддільність біометричних ознак від дієздатної особистості;

– труднощі фальсифікації біометричних ознак.

Біометричні ознаки, що можуть бути використані при ідентифікації потенційного користувача: візерунок райдужної оболонки і сітківки око,

відбитки пальців, геометрична форма руки, форма і розміри особи, особливості голосу, біомеханічні характеристики рукописного підпису, біомеханічні характеристики «клавіатурного почерку».

Під час реєстрації користувач повинний продемонструвати один або кілька разів свої характерні біометричні ознаки. Ці ознаки (відомі як справжні) реєструються системою як контрольний «образ» законного користувача. Цей образ користувача зберігається в електронній формі і використовується для перевірки ідентичності кожного, хто видає себе за відповідного законного користувача. У залежності від збігу або розбіжності сукупності пред'явлених ознак із зареєстрованими в контрольному образі їх що пред'явив визнається законним користувачем (збіг) чи ні (розбіжність).

Системи ідентифікації за візерунком райдужної оболонки і сітківки ока можуть бути розділені на два класи:

– використання малюнка райдужної оболонки ока;

– використання малюнка кровоносних судин сітківки ока.

Оскільки імовірність повторення даних параметрів дорівнює 10^{-78}, ці системи є найбільш надійними серед усіх біометричних систем. Такі засоби ідентифікації застосовуються там, де потрібен високий рівень безпеки (наприклад, у США в зонах військових і оборонних об'єктів). Індивідуальність розташування кровоносних судин доведена в 1935 році.

Системи ідентифікації за відбитками пальців є найпоширенішими. Одна з основних причин широкого поширення таких систем полягає в наявності великих банків даних по відбитках пальців. Основними користувачами подібних систем в усім світі є поліція, різні державні і деякі банківські організації.

Системи ідентифікації за геометричною формою руки використовують сканери форми руки і зазвичай встановлюються на стінах. Слід зазначити, що переважна більшість користувачів надають перевагу системі саме цього типу, а не описаній вище.

Системи ідентифікації за особою і за особливостями голосу є найбільш доступними через їхню дешевину, оскільки більшість сучасних комп’ютерів мають відео- і аудіозасоби. Системи даного класу широко застосовуються при віддаленій ідентифікації суб’єкта доступу в телекомунікаційних мережах.

Сучасні методи аналізують декілька характеристик мови: період висоти тону, відносний спектр амплітуди, резонансні частоти. Недолік – вплив по сторонніх шумів, нежить, настрій, психологічний фактор.

Системи ідентифікації особистостей за динамікою рукописного підпису враховують інтенсивність кожного зусилля що підписує, частотні характеристики написання кожного елемента підпису і накреслення підпису в цілому. Перетворювачі можуть бути встановлені в ручці та під пластинкою, на якій пишуть.

Одна із останні інтелектуальних ручок SmartPen розроблена бельгійськими дослідниками. Ручка містить мікродатчики для зняття трьохмірних динамічних параметрів, процесор для обробки даних, перо-датчик

і систему захисту від радіоперехоплень. Нею можна писати по звичайному паперу.

Системи ідентифікації за біомеханічними характеристиками «клавіатурного почерку» ґрунтуються на тому, що моменти натискання і відпускання клавіш при наборі тексту на клавіатурі істотно відрізняються в різних користувачів. Цей динамічний ритм набору («клавіатурний почерк») дозволяє побудувати досить надійні засоби ідентифікації. У випадку виявлення зміни клавіатурного почерку користувача йому автоматично забороняється робота на КС.

Застосування біометричних параметрів при ідентифікації суб'єктів доступу автоматизованих систем поки не одержало належного нормативно-правового забезпечення, зокрема у виді стандартів. Тому застосування систем біометричної ідентифікації допускається тільки в автоматизованих системах, що обробляють і зберігають персональні дані, що складають комерційну і службову таємницю.

6.4. Взаємна перевірка дійсності користувачів

Звичайно сторони, що вступають в інформаційний обмін, мають потребу у взаємній перевірці дійсності (аутентифікації) один одного. Цей процес взаємної аутентифікації виконують на початку сеансу зв’язку.

Для перевірки дійсності застосовують наступні способи:

– механізм запиту-відповіді;

– механізм оцінки часу («часовий штемпель»).

Механізм запиту-відповіді полягає в наступному. Якщо користувач *A* хоче бути упевненим, що повідомлення, одержувані ним від користувача *B*, не є помилковими, він включає в повідомлення, що посилається для *B,* непередбачений елемент-запит *X* (наприклад, деяке випадкове число). При відповіді користувач *B* повинний виконати над цим елементом деяку операцію (наприклад, обчислити деяку функцію *f(X)*). Це неможливо здійснити заздалегідь, тому що користувачеві *B* невідомо, яке випадкове число *X* прийде в запиті. Одержавши відповідь з результатом дій *B*, користувач може бути упевнений, що *B* – справжній. Недолік цього методу – можливість установлення закономірності між запитом і відповіддю.

Механізм оцінки часу має на увазі реєстрацію часу для кожного повідомлення. У цьому випадку кожен користувач мережі може визначити, наскільки «застаріло» повідомлення, що прийшло, і вирішити не приймати його, оскільки воно може бути помилковим.

В обох випадках для захисту механізму контролю варто застосовувати шифрування, щоб бути упевненим, що відповідь послана не зловмисником.

При використанні оцінок часу виникає проблема припустимого часового інтервалу затримки для підтвердження дійсності сеансу. Адже повідомлення з «тимчасовим штемпелем» у принципі не може бути передане миттєво. Крім того, комп'ютерний годинник одержувача і відправника не можуть бути абсолютно синхронізовані. Яке запізнювання «штемпеля» є підозрілим?

Тестові завдання для розділів 1-6

1. На сьогоднішній день під системою захисту комп’ютерної інформації розуміють:
   1) сукупність методів криптографічного перетворення вихідних даних з метою отримання зашифрованих даних;
   2) сукупність правових, організаційних, технічних, програмних і криптографічних засобів і методів, що забезпечують захищеність системи;
   3) захист від навмисних спроб людини отримати доступ до цієї інформації або модифікувати її;
   4) сукупність методів, програм і алгоритмів, що дозволяють забезпечити надійний захист від несанкціонованого доступу до інформації;
   5) розробку методів і засобів перетворення інформації.
2. Вкажіть, що не належить до традиційних напрямків захисту комп'ютерної інформації:
   1) криптографія;
   2) антивірусний захист;
   3) лінійне програмування;
   4) захист від несанкціонованого копіювання;
   5) мережний захист.
3. Що є об’єктом захисту інформації?
   1) комп’ютерна система або автоматизована система обробки даних;
   2) обчислювальні мережі;
   3) системи управління базами даних;
   4) пам’ять ЕОМ;
   5) інша відповідь.
4. Що є предметом захисту в комп’ютерних системах?
   1) електронні та електромеханічні пристрої, а також машинні носії;
   2) інформація;
   3) системи передачі даних;
   4) комп'ютерна система;
   5) оперативна пам’ять.
5. Канал витоку інформації, зв’язаний із розташуванням звукових хвиль у повітрі або пружних коливань у інших середовищах, які виникають при роботі пристроїв відображення інформації називається:
   1) інформаційним;
   2) акустичним;
   3) оптичним;
   4) електромагнітним;
   5) візуальним.
6. Під загрозою безпеки інформації розуміють:
   1) атаку на інформацію з боку зловмисника;
   2) потенційно можливу подію, процес або явище, які можуть призвести до знищення, втрати цілісності, конфіденційності або доступності інформації;
   3) несанкціонований доступ до інформації, який може привести до порушення цілісності системи комп’ютерної безпеки;
   4) несанкціоноване копіювання інформації;
   5) все перераховане вище.

7. Вся множина потенційних загроз безпеці інформації в КС може бути розділена на наступні класи:
1) випадкові і навмисні загрози;
2) потенційні та не потенційні загрози;
3) можливі та неможливі загрози;
4) передбачувані та непередбачувані загрози;
5) інша відповідь.

8. До основних видів випадкових загроз не належать:
1) стихійні лиха і аварії;
2) збої і відмови технічних засобів;
3) помилки при розробці комп'ютерних систем;
4) електромагнітні випромінювання та наведення;
5) алгоритмічні і програмні помилки.

9. До основних видів навмисних загроз не належать:
1) шпигунство і диверсії;
2) несанкціонований доступ до інформації;
3) електромагнітні випромінювання та наведення;
4) шкідливі програми;
5) стихійні лиха і аварії.

10. Яка з математичних моделей реалізує мандатну політику безпеки?
1) Adept-50;
2) HRU;
3) Take-Grant;
4) Белла-ЛаПадули;
5) Хартсона.

11. Мандатна політика безпеки заснована на …:
1) матриці доступу;
2) виборчому способі керування доступом;
3) повноважному способі керування доступом;
4) комплексному підході;
5) інша відповідь.

12. Розташуйте етапи процесу побудови системи захисту (супровід (С), планування (П), реалізація (Р), аналіз можливих погроз(А)) у порядку їх реалізації:
1) ПАРС;
2) АРПС;
3) АСРП;
4) ПРАС;
5) АПРС.

13. TCSEC – це:
1) критерії захищеності інформації в КС від несанкціонованого доступу;
2) критерії оцінювання безпеки інформаційних технологій;
3) федеральні критерії безпеки інформаційних технологій;
4) критерії оцінювання захищених комп'ютерних систем;
5) «Загальні критерії».

14. ISO/IEC 15408 – це:
1) критерії захищеності інформації в КС від несанкціонованого доступу;
2) критерії оцінювання безпеки інформаційних технологій;
3) федеральні критерії безпеки інформаційних технологій;

4) критерії оцінювання захищених комп'ютерних систем;
5) «Оранжева книга».

15. Критерії захищеності інформації в КС від несанкціонованого доступу в Україні задаються документом:
1) FCITS;
2) ISO/IEC;
3) НД ТЗУ;
4) TCSEC;
5) ITSEC.

16. Згідно із стандартом TCSEC група С відповідає:
1) дискреційному захисту;
2) мандатному захисту;
3) дискреційному і мандатному захисту;
4) мінімальному захисту;
5) верифікованому захисту.

17. Згідно із стандартом TCSEC група А відповідає:
1) дискреційному захисту;
2) мандатному захисту;
3) дискреційному і мандатному захисту;
4) мінімальному захисту;
5) верифікованому захисту

18. Усі коди, що характеризують комп'ютерні злочини, мають ідентифікатор, що починається з букви:
1) A;
2) Q ;
3) B;
4) S;
5) C.

19. Чинні у країні закони, укази, нормативні акти, що регламентують правила взаємодії з інформацією обмеженого використання і відповідальність за їх порушення, які відіграють роль стримуючого чинника для потенційних порушень відносять до заходів захисту інформаційних систем:
1) правових;
2) морально-етичних;
3) адміністративних;
4) фізичних;
5) організаційних.

20. Процес, у результаті якого користувач повідомляє системі за запитом свої унікальні дані називається:
1) аутентифікацією;
2) ідентифікацією;
3) класифікацією;
4) модернізацією;
5) аудитом.

7. ОСНОВИ КРИПТОГРАФІЧНОГО ЗАХИСТУ ІНФОРМАЦІЇ

7.1. Історія виникнення криптографії

Проблеми захисту інформації хвилювали людство з незапам'ятних часів. Необхідність захисту інформації виникла з потреб таємної передачі, як військових, так і дипломатичних повідомлень. Наприклад, античні спартанці шифрували свої військові повідомлення. У китайців простий запис повідомлення за допомогою ієрогліфів робив його таємним для чужоземців.

Для позначення всієї області секретного зв'язку використовується термін «криптологія», що походить від грецьких коренів «cryptos» – таємний і «logos» – повідомлення. Криптологія досить чітко може бути розділена на два напрямки: криптографію і криптоаналіз.

Задача криптографа – забезпечити конфіденційність і аутентичність переданих повідомлень. Термін криптографія ввів Д. Валліс.

Задача криптоаналітика – «зламати» систему захисту, розроблену криптографами. Він намагається розкрити зашифрований текст або видати підроблене повідомлення за справжнє .

Метою криптографії є приховання вмісту повідомлень за рахунок їх шифрування, на відміну від цього, при стеганографії, приховується сам факт існування таємного повідомлення. Стеганографія – із грецького «тайнопис». Історично цей напрям з'явився першим, але потім у багатьох випадках був витісненим криптографією. Розвиток засобів обчислювальної техніки за останнє десятиліття дав новий поштовх для розвитку комп'ютерної стеганографії, з'явилось багато нових областей застосування. Повідомлення приховуються у цифрові дані, які мають аналогову природу, це – текст, звук, зображення, відео. Методи стеганографії дозволяють не тільки приховано передавати дані, але й успішно вирішувати завдання завадостійкості аутентифікації, захисту інформації від несанкціонованого копіювання, відстеження поширення інформації мережами зв'язку, пошуку інформації в мультимедійних базах даних.

Перші канали зв'язку були дуже простими. Їх організовували, використовуючи надійних кур'єрів. Безпека таких систем зв'язку залежала як від надійності кур'єра, так і від його здатності не попадати в ситуації, при яких могло мати місце розкриття повідомлення.

За твердженням спеціалістів – криптографія за віком ровесниця єгипетських пірамід. У документах Стародавніх цивілізацій: Індії, Єгипту, Месопотамії – є відомості про системи і способи створення шифрованих листів. У стародавніх релігійних книгах Індії вказується, що сам Будда знав декілька способів письма, серед яких були присутні шифри перестановок (в сучасній класифікації). Один із стародавніх шифрованих текстів в Месопотамії (20 ст. до н.е.) являє собою глиняну табличку, яка містить рецепт виготовлення глазурі в гончарному виробництві, в якому ігноруються деякі голосні і приголосні і використовуються числа замість імен.

Так ще в 5-му, 6-му століттях до нашої ери греки застосовували спеціальний шифрувальний засіб. За описанням, він складався з двох паличок однакової довжини і товщини. Одну паличку залишали собі, а другу віддавали від'їжджаючому. Ці палички називали **скіталами**. Коли товаришам потрібно було повідомити якусь таємницю, вони вирізали довгу і вузьку смужку папіруса, намотували її на скіталу, не залишаючи ніяких проміжків, так щоб вся поверхня скітали була зайнята цією смужкою. Потім, залишаючи папірус так як він є, писали на ньому повідомлення, далі знімали смужку і без палички передавали адресатові. Так як букви на ній були розкидані, то прочитати написане можна було лише за допомогою такої ж скітали.

Аристотелю належить спосіб дешифрування цього шифру. Необхідно виготовити довгий конус, і починаючи з основи, обмотувати його смужкою із шифрованим повідомленням, поступово зсуваючи її до вершини. В якийсь момент почнуть читатись кусочки повідомлення. Так можна визначити розмір скітали.

У Стародавній Греції (2 ст. до н.е.) був відомий шифр, що називався квадрат Полібія. Цей засіб представляв собою квадрат 5 на 5, стовпці і рядки якого були пронумеровані цифрами від 1 до 5. В кожну клітинку цього квадрату записувалась одна буква (в грецькому алфавіті одна клітина залишалась пустою, а в латинському – в одну клітинку записувались дві букви i, j). У результаті кожній букві відповідала пара чисел і шифроване повідомлення перетворювалось в послідовність пар чисел [4, 10].

Приклад

13 34 22 24 44 34 15 42 22 34 43 45 32.

Це повідомлення записане за допомогою квадрата Полібія, в якому букви розташовані в алфавітному порядку.

Таблиця 7.1

Заміна букв

	1	2	3	4	5
1	a	f	l	q	v
2	b	g	m	r	w
3	c	h	n	s	x
4	d	i,j	o	t	y
5	e	k	p	u	z

„Cogito, ergo sum" – „ Я думаю, значить існую".

В 1-му ст.н.е. Ю. Цезарь під час війни з Галами, переписуючись зі своїми друзями в Римі, заміняв в повідомленні першу букву латинського алфавіту на четверту, другу – на п'яту і т. д., а останню на третю.

ABCDEFGHIJKLMNOPQRSTUVWXYZ

DEFGHIJKLMNOPQRSTUVWXYZABC

Обидва ці методи належать до класу шифру, що називається підстановка чи проста заміна.

Розвитком шифру простої заміни став шифр Б. Віженера (XVI ст., Франція). Криптографією займалася багато відомих математиків, таких як

Ф. Вієт, Д. Кардано, В. Лейбніц і, нарешті, Ф. Бекон, який запропонував двійкове кодування латинського алфавіту. У Росії самостійна криптографічна служба була уперше організована Петром I, який під впливом спілкування з В. Лейбніцом заснував цифрову палату для розвитку і використання криптографії.

Промислова революція в розвинених країнах привела до створення шифрувальних машин. У кінці XVIII століття Т. Джефферсоном (майбутнім третім президентом США) були винайдені шифруючі колеса. Першу практично працюючу шифрувальну машину запропонував у 1917 році Г. Вернам. У тому ж році була винайдена роторна шифрувальна машина, що згодом випускалася фірмою Сименс під назвою «Енігма» (загадка) – основний супротивник криптографів Союзних держав у роки Другої світової війни.

Створення сучасних комп'ютерних систем і поява глобальних комп'ютерних мереж радикально змінила характер і діапазон проблем захисту інформації. У широко комп'ютеризованому і інформатизованому сучасному суспільстві володіння реальними цінностями, керування ними, передача цінностей або доступ до них часто засновані на неопредметненій інформації, тобто на інформації, існування якої не обов'язково зв'язується з яким-небудь записом на фізичному носії. Тому досить важливо створювати і застосовувати ефективні засоби для реалізації всіх необхідних функцій, зв'язаних із забезпеченням конфіденційності і цілісності інформації.

Оскільки інформація може бути дуже коштовною або особливо важливою, можливі різноманітні зловмисні дії стосовно комп'ютерних систем, що зберігають, обробляють або передають таку інформацію. Наприклад, порушник може спробувати видати себе за іншого користувача системи, підслухати канал зв'язку або перехопити і змінити інформацію, якою обмінюються користувачі системи. Порушником може бути і користувач системи, що відмовляється від повідомлення, у дійсності сформованого ним, або який намагається стверджувати, що ним отримане повідомлення, яке в дійсності не передавалося. Зловмисник може спробувати розширити свої повноваження, щоб одержати доступ до інформації, до якої йому наданий тільки частковий доступ, або спробувати зруйнувати систему, несанкціоновано змінюючи права інших користувачів.

Для вирішення зазначених і інших подібних проблем не існує якогось одного технічного прийому або засобу. Однак, загальним у вирішенні багатьох з них є використання криптографії і криптоподібних перетворень інформації.

Протягом більш ніж тисячолітньої історії криптографії вона являла собою постійно оновлюючий й вдосконалюючий набір технічних прийомів шифрування і розшифрування, що зберігалися в строгому секреті.

Період розвитку криптології з старих часів до 1949 року прийнято називати ерою донаукової криптології, оскільки досягнення тих часів були засновані на інтуїції і не підкріплювалися доказами. Криптологією займалися тоді майже винятково як мистецтвом, а не як наукою. Звичайно, це не означає, що історія криптології тих часів не представляє для нас ніякого інтересу. Лише з початком другої світової війни криптологічні служби держав, що воювали

усвідомили, що математики можуть внести вагомий вклад у розвиток криптології. Зокрема, в Англії в цей час був закликаний на службу як фахівець з криптології Алан Тьюринг.

Публікація в 1949 році статті К. Шеннона «Теорія зв'язку в секретних системах» стала початком нової ери наукової криптології із секретними ключами. У цій блискучій роботі К. Шеннон зв'язав криптографію з теорією інформації [15].

Із середини сімдесятих років (у зв'язку з винаходом систем з відкритим ключем) криптографія не тільки перестала бути секретним зводом прийомів шифрування-розшифрування, але і почала оформлятися в нову математичну теорію. За останні двадцять років відбулося значне підвищення активності в області розвитку криптографії та її застосування для вирішення проблем захисту інформації. Це викликано широким визнанням крайньої необхідності в засобах забезпечення захисту інформації у всіх областях діяльності широко інформатизованого людського співтовариства й обумовлено появою таких нових фундаментальних ідей, як асиметрична криптографія, стійкі протоколи, надійність яких заснована на гарантованій складності розв'язання математичних задач, і т.д.

У сучасний час відомо велику кількість криптографічних методів захисту інформації. Класифікація методів шифрування може бути здійснена за наступними ознаками:

– за типом ключів (симетричні і асиметричні);

– за розміром блоку інформації (потокові і блокові);

– за характером дії над даними (метод перестановки, метод заміни, аналітичні методи, методи гамування, комбіновані методи).

У криптографічній системі перетворення шифрування може бути симетричним або асиметричним щодо перетворення розшифрування. Відповідно розрізняють два класи криптосистем:

– симетричні одноключеві криптосистеми (із секретним ключем);

– асиметричні двоключеві криптосистеми (з відкритим ключем).

Сучасні симетричні одноключеві криптоалгоритми базуються на принципах, викладених у згаданій роботі К. Шеннона. До них відносяться закордонні криптоалгоритми DES, IDEA і вітчизняний криптоалгоритм, описаний у стандарті ГОСТ 28147-89 та інші. Схеми реалізації цих криптоалгоритмів відкрито опубліковані і ретельно проаналізовані багатьма дослідниками. У цих криптосистемах секретним є тільки ключ, за допомогою якого здійснюється шифрування і дешифрування інформації. Дані криптосистеми можуть використовуватися не тільки для шифрування, але і для перевірки дійсності (аутентифікації) повідомлень.

Появі нового напрямку в криптології – асиметричної криптографії з відкритим ключем – сприяли дві проблеми, що не вдавалося вирішити в рамках класичної симетричної одноключевої криптографії. Перша з цих проблем зв'язана з поширенням секретних ключів. Наявність секретного ключа, відомого тільки одержувачеві повідомлення і його відправникові, сторіччями вважалося неодмінною умовою безпечної передачі інформації. Але при

використанні симетричних криптосистем із секретними ключами вимагають рішення наступні питання: 1) як передати учасникам обміну інформацією змінювані секретні ключі, що потрібні їм для виконання цього обміну? 2) як учасники обміну зможуть переконатися в цілісності того, що вони одержали? Друга з цих проблем зв'язана з формуванням електронного цифрового підпису. Наприкінці листа або іншого авторизованого документа відправник звичайно ставить свій підпис. Подібна дія переслідує дві мети: по-перше, одержувач може переконатися в дійсності листа, звіривши підпис з наявним у нього зразком; по-друге, особистий підпис є юридичним гарантом авторства документа. Цей аспект особливо важливий при заключені різного роду торговельних угод, складанні зобов'язань, доручень і т.д.

Підробити підпис людини на папері зовсім не просто, а скопіювати ланцюжок цифр на ЕОМ – нескладна операція. Як у такому випадку гарантувати дійсність і авторство електронних повідомлень? У той же час існує багато додатків, що вимагають достовірного цифрового підпису для цифрової інформації, яка б виконувала всі ті задачі, що виконує підпис, поставлений на документі рукою.

Обидві ці проблеми здавалися нерозв'язними. Однак вони були успішно вирішені за допомогою криптографії з відкритими ключами. В опублікованій у 1976 р. статті «Нові напрямки в криптографії» У. Діффі і М. Хеллман уперше показали, що секретний зв'язок можливий без передачі секретного ключа між відправником і одержувачем. В основі цього криптографічного методу лежать так звані однобічні функції: при заданому значенні x відносно просто обчислити значення $f(x)$, однак, знаючи $y=f(x)$, визначити по в значення x надзвичайно важко.

В асиметричних криптосистемах з відкритим ключем використовуються два ключі, принаймні, один із яких неможливо обчислити з іншого. Один ключ використовується відправником для шифрування інформації; інший - одержувачем для дешифрування одержуваних шифротекстів. Звичайно в додатках один ключ повинний бути несекретним, а інший-секретним.

А) Якщо ключ дешифрування неможливо одержати з ключа шифрування за допомогою обчислень, то таємність інформації, зашифрованої на несекретному (відкритому) ключі, буде забезпечена. Однак цей ключ повинний бути захищений від підміни або модифікації, інакше відправник може бути обдуреним і буде виконувати зашифрування на підробленому ключі, відповідно ключ дешифрування якого відомий супротивникові. Для того, щоб забезпечити секретність інформації, ключ розшифрування одержувача повинний бути секретним і фізично захищеним від підміни. Так працює канал забезпечення конфіденційності (таємності) інформації.

Б) Якщо ж, навпаки, неможливо одержати ключ шифрування з ключа дешифрування, то ключ дешифрування може бути несекретним, а секретний ключ шифрування можна використовувати для формування електронного цифрового підпису під повідомленням. У цьому випадку, якщо результат розшифрування цифрового підпису містить аутентифікаційну інформацію (заздалегідь погоджену законним відправником інформації з потенційним

одержувачем), цей підпис засвідчує цілісність повідомлення, отриманого від відправника. Так працює канал аутентифікації повідомлення.

Поява нових інформаційних технологій і інтенсивний розвиток комп'ютерних мереж залучають усе більшу увагу користувачів до глобальної мережі Internet. Багато компаній і організації підключають сьогодні свої локальні мережі до мережі Internet, щоб скористатися її ресурсами і перевагами. Бізнесмени і державні організації використовують Internet у різних цілях, включаючи обмін електронною поштою, поширення інформації серед зацікавлених осіб і т.д. Підключення до Internet дає великі переваги, однак при цьому виникають серйозні проблеми з забезпеченням інформаційної безпеки підключеної локальної або корпоративної мережі. У силу відкритості своєї ідеології Internet надає зловмисникам багато можливостей для вторгнення у внутрішні мережі підприємств і організацій з метою розкрадання, перекручування або руйнування важливої і конфіденційної інформації. Розв’язання задач по захисту внутрішніх мереж від найбільш ймовірних атак через Internet може бути покладене на міжмережні екрани, іноді називані брандмауерами або firewall. Застосовуються і програмні методи захисту, до яких відносяться захищені криптопротоколи SSL і SKIP.

Важливим додатком, що потребує ефективні засоби захисту інформації, є електронні платіжні системи. У цих системах у якості універсального платіжного засобу використовуються банківські пластикові карти. Для забезпечення надійної роботи електронна платіжна система повинна бути надійно захищена. З погляду інформаційної безпеки в системах електронних платежів існує ряд потенційно уразливих місць, зокрема, пересилання платіжних і інших повідомлень між банками, між банком і банкоматом, між банком і клієнтами. Для забезпечення захисту інформації на окремих вузлах системи електронних платежів повинні бути реалізовані наступні механізми захисту: керування доступом на віконних системах, забезпечення цілісності і конфіденційності повідомлень, взаємна аутентифікація абонентів, гарантії доставки повідомлення і т. д. Якість вирішення зазначених проблем істотно залежить від раціональності вибору криптографічних засобів при реалізації механізмів захисту [9; 10; 15].

7.2. Принципи криптографічного захисту інформації

Криптографія являє собою сукупність методів перетворення даних, спрямованих на те, щоб зробити ці дані марними для супротивника. Такі перетворення дозволяють вирішити дві головні проблеми захисту даних: проблему конфіденційності (шляхом позбавлення супротивника можливості витягти інформацію з каналу зв’язку) і проблему цілісності (шляхом позбавлення супротивника можливості змінити повідомлення так, щоб змінився його зміст, або ввести помилкову інформацію в канал зв’язку).

Крім забезпечення конфіденційності інформації, криптографію часто використовують для вирішення інших завдань:

– перевірки справжності інформації – отримувач повідомлення повинен мати можливість встановити джерело інформації, а супротивник – неспроможний замаскуватися під когось іншого;

– цілісності інформації – отримувач повідомлення повинен мати можливість перевірити, чи не було це повідомлення змінено у процесі кореспонденції, а супротивник – неспроможний видати фальшиве повідомлення за справжнє;

– невідмови від авторства – відправник повідомлення у подальшому не повинен мати можливості неправдиво відмовитися від надісланого повідомлення.

Криптосистема – набір інструкцій, апаратні засоби, комплекс програм комп'ютера, що дозволяють зашифрувати відкритий текст і розшифрувати шифротекст різними способами, один із яких вибирається за допомогою конкретного ключа.

До складу криптосистеми чи шифру входять нижченаведені об'єкти:

1) Абетка A, в якій записані відкриті тексти – упорядкований набір слів із літер абетки. Будь-яке повідомлення є словом в абетці A, а множина слів A^* – простором відкритих текстів.

2) Абетка B, в якій записані криптограми. Будь-яка криптограма є словом в абетці B. Множина B^* у цьому разі називається простором криптограм. Для деяких криптосистем $A = B$.

3) Криптоалгоритм – алгоритм криптографічного перетворення, тобто математична функція, за допомогою якої шифрується та дешифрується інформація.

4) Ключ – змінний елемент шифру, застосованого для шифрування окремого повідомлення або дешифрування криптограми. Множина всіх можливих ключів називається простором ключів даного шифру. Аби задати ключ (вибрати ключ із простору ключів), слід задати конкретні таємні значення (стан) параметрів криптоалгоритму, що забезпечує вибір серед багатьох можливих варіантів одного варіанта перетворення відкритого тексту на шифротекст чи навпаки. Усі сучасні шифри базуються на принципі Керкхоффса, згідно з яким секретність шифру забезпечується секретністю ключа, а не секретністю алгоритму шифрування.

Відзначимо деякі основні характеристики шифрів:

– стійкість шифру – здатність шифру протистояти всіляким несанкціонованим атакам на нього. Це поняття – центральне для криптографії. Якщо наявність криптограми у супротивника не зменшує невизначеності можливого вибору відкритого тексту, що відповідає криптограмі, то такий шифр називають теоретично стійким;

– ефективність шифру, яка визначається швидкістю шифрувальних і дешифрувальних відображень;

– довжина ключа, точніше, об'єм ключового простору;

– простота виконання операцій шифрування та дешифрування;

– збільшення кількості помилок (для деяких шифрів помилка в одній літері при шифруванні викликає значну кількість помилок у дешифрованому

тексті). При виборі шифру для зв'язку намагаються мінімізувати цю властивість шифру;

– завадостійкість шифру – здатність шифру при дешифруванні криптограми протидіяти утворенню помилок, що виникли під час шифрування через перешкоди на лініях зв'язку;

– імітостійкість шифру – здатність шифру протидіяти спробам супротивника нав'язати абонентові неправдиву інформацію шляхом спотворення криптограми на каналах зв'язку.

Про подання інформації у цифровій формі. Традиційно при використанні сучасних засобів зв'язку для передачі повідомлення або комп'ютерного шифрування інформація подається у цифровій формі, коли кожен символ тексту замінюється його номером у абетці. Наприклад, слово «аналіз» згідно з табл. 7.2 для української абетки (нумерація починається з нуля) буде подано як 00 17 00 15 11 09.

Таблиця 7.2

Заміна букв на цифри

00	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16
а	б	в	г	ґ	д	е	є	ж	з	и	і	ї	й	к	л	м
17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	
н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ь	ю	я	

Часто номери букв записують не в десятковій системі числення, а в двійковій. Слову «аналіз» у цьому разі відповідатиме двійкове слово

00000 010001 000000 001111 001011 001001,

де кожен блок з шести цифр є номером букви у двійковому записі. Отже, будь-який текст можна подати у двійковій формі за допомогою лише двох символів – 0 і 1, які називають бітами. Також використовують таблицю ASCII кодів для перетворення символів тексту в біти.

У зарубіжній криптографічній літературі літерою *M* (від слова message – повідомлення) прийнято позначати повідомлення відкритого тексту, що підлягає шифруванню, *C* – криптограму (від слова ciphertext), *E* (encrypt) і *D* (decrypt) – математичні функції, за допомогою яких виконуються процеси шифрування та дешифрування відповідно.

Відправник генерує відкритий текст вихідного повідомлення M, що повинне бути передано законному одержувачеві по незахищеному каналі. За каналом стежить перехоплювач з метою перехопити і розкрити передане повідомлення. Для того щоб перехоплювач не зміг довідатися зміст повідомлення *M*, відправник шифрує його за допомогою оборотного перетворення $E_к$ і одержує шифротекст (або криптограму) $C=E_к(M)$, що відправляє одержувачеві.

Законний одержувач, прийнявши шифротекст *C*, розшифровує його за допомогою оберненого перетворення $D = E_K^{-1}$ і одержує вихідне повідомлення у виді відкритого тексту *M*: $D_K(C) = E_K^{-1}(E_K(M)) = M$. Узагальнена схема криптографічної системи, що забезпечує шифрування переданої інформації, показана на рис. 7.1.

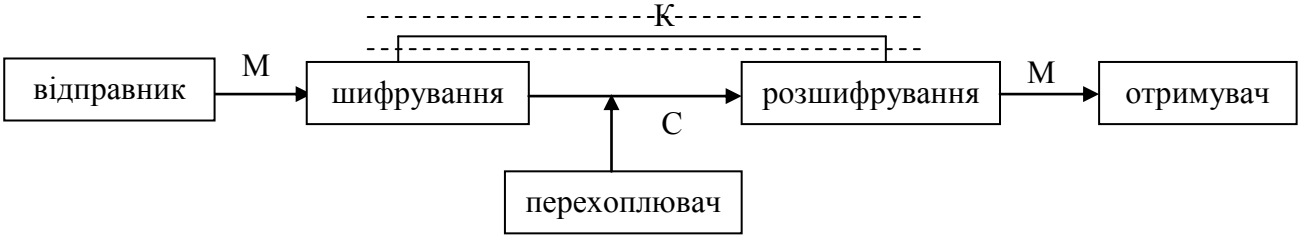

Рисунок 7.1 – Узагальнена схема криптографічної системи

Перетворення E_K вибирається із сімейства криптографічних перетворень, що називають криптоалгоритмами. Параметр, за допомогою якого обирається використовуване перетворення, називається криптографічним ключем *K*.

Криптографічна систем це однопараметричне сімейство (E_K), де $k \in \overline{K}$ зворотних перетворень $E_K : \overline{M} \to \overline{C}$ з простору $\overline{M}$ повідомлень відкритого тексту в простір $\overline{C}$ шифрованих текстів. Параметр *K* вибирається з скінченої множини $\overline{K}$, що називається простором ключів.

Схема симетричної криптосистеми з одним секретним ключем була показана на рис. 7.1. У ній використовуються однакові секретні ключі в блоці шифрування і блоці розшифрування.

Узагальнена схема асиметричної криптосистеми з двома різними ключами *K1* і *K2* показана на рис. 7.2. У цієї криптосистеми один із ключів є відкритим, а інший - секретним.

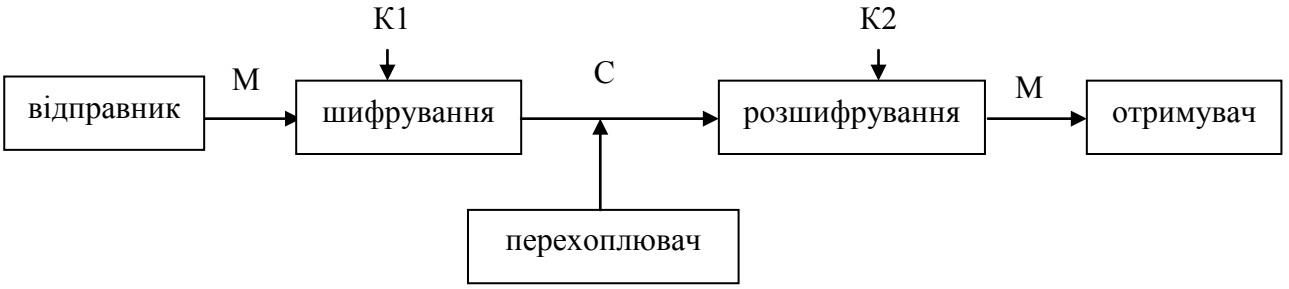

Рисунок 7.2 – Узагальнена схема асиметричної криптосистеми

У симетричної криптосистеми секретний ключ треба передавати відправникові й одержувачеві по захищеному каналі поширення ключів, наприклад такому, як кур'єрська служба. На рис.7.1 цей канал показаний «екранованою» лінією. Існують і інші способи розподілу секретних ключів, вони будуть розглянуті пізніше. В асиметричної криптосистемі передають по незахищеному каналі тільки відкритий ключ, а секретний ключ зберігають на місці його генерації.

На рисунку 7.3 показаний потік інформації в криптосистемі у випадку активних дій перехоплювача. Активний перехоплювач не тільки зчитує всі шифртексти, передані по каналі, але може також намагатися змінювати них за своїм розсудом.

Будь-яка спроба з боку перехоплювача розшифрувати шифротекст *C* для одержання відкритого тексту *M* або зашифрувати свій власний текст *M'* для одержання правдоподібного шифротекста *C*, не маючи справжнього ключа, називається криптоаналітичною атакою.

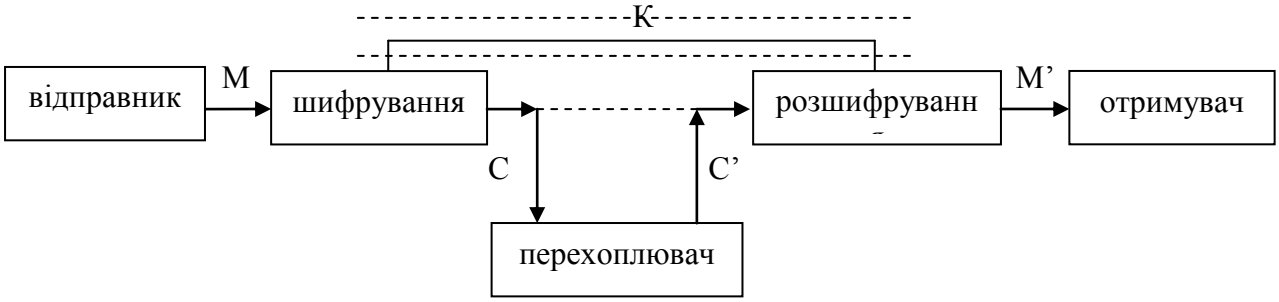

Рисунок 7.3 – Потік інформації в криптосистемі у випадку активних дій перехоплювача

Деякі відомості з теорії складності задач

Теорія складності – це методика аналізу обчислювальної складності різних математичних (зокрема криптографічних) методів та алгоритмів. У криптографії цю теорію використовують як основу для визначення надійності криптографічних алгоритмів, що дає змогу проаналізувати, чи можна зламати той чи інший шифр за прийнятний час. Очевидно, складність обчислювального алгоритму залежить від обчислювальних потужностей, необхідних на його виконання. Найчастіше складність алгоритму характеризують двома параметрами: T – часова складність та S – просторова складність або вимоги до пам’яті комп’ютера.

З’ясуємо поняття часової складності. Нехай A – алгоритм розв’язку деякого класу задач, n – розмірність окремої задачі цього класу (масив або довжина вхідної послідовності, іншими словами, величина, що характеризує розмір вхідних даних). Функцію $f_A(n)$ назвемо робочою, якщо вона визначає верхню межу максимальної кількості операцій (додавання, множення, порівняння і т. д.), які має виконати алгоритм A під час розв’язку задачі розмірності n. Тоді часова складність залежить від того члена розкладання робочої функції, який із зростанням n зростатиме швидше інших (члени розкладання, порядок яких нижче, ігноруються). У багатьох криптографічних та математичних задачах порядок величини обчислювальної складності алгоритмів визначається стандартно за допомогою символу «o». Отже, для визначення часової складності алгоритму не обов’язково знати точний термін виконання різних операцій, кількість бітів, потрібних для зображення змінних, навіть швидкість процесора. З цієї оцінки наочно видно, як розмірність вхідних даних впливає на вимоги до часу виконання та об’єму пам’яті. Так, якщо часова складність одного алгоритму $T = o(n)$, то подвоєння розмірності вхідних даних відповідно подвоює і час виконання алгоритму, а якщо часова складність деякого іншого алгоритму дорівнює $T = o(2^n)$, то подвоєння часу виконання викличе навіть додавання лише одного біта до вхідних даних.

Наведемо відому класифікацію алгоритмів за їх часовою складністю:

– алгоритм називається сталим, якщо його складність не залежить від n;

– алгоритм називається поліноміальним за часом, якщо його часова складність дорівнює $o(n^m)$, де m – константа, тобто робоча функція $f_A(n)$

поліноміального алгоритму є поліномом $P_m(n)$ степеня m. Зокрема, алгоритм лінійний, якщо його складність $T=o(n)$, квадратичний – у випадку $T=o(n^2)$ і т.д.;

– алгоритми, складність яких дорівнює $o(t^{g(n)})$, де $t>1$ –константа, $g(n)$ – деяка поліноміальна функція від n, називаються експоненціальними;

– експоненціальні алгоритми, складність яких є $o(c^{g(n)})$, де із зростанням n функція $g(n)$ зростає швидше константи c, називаються супер-поліноміальними.

Якщо сучасні комп'ютери здатні сприймати поліноміальні алгоритми для задач великої розмірності, то експоненціальні алгоритми часто стають каменем спотикання. Тому в ідеальному випадку кожен криптограф хотів би стверджувати, що його шифр можна зламати тільки за допомогою алгоритму, який характеризується експоненціальною часовою складністю. Із збільшенням n часова складність алгоритмів може стати настільки великою, що почне заважати практичній реалізації алгоритму. Для ілюстрації цього у табл. 7.3 наведено час виконання алгоритмів різних класів при розмірі вхідних даних $n=10^6$.

Таблиця 7.3

Час виконання алгоритмів

Клас алгоритмів	Складність	Кількість операцій комп'ютера	Час зі швидкістю 10^6 операцій за секунду
Сталі	$o(1)$	1	1 мкс
Лінійні	$o(n)$	10^6	1с
Квадратичні	$o(n^2)$	10^{12}	11,6 доби
Кубічні	$o(n^3)$	10^{18}	32000 років
Експоненціальні	$o(2^n)$	10^{301030}	У 10^{301006} разів довше за термін існування Всесвіту

Зауваження 1. Часова складність силової атаки пропорційна кількості можливих ключів, яка експоненціально залежить від довжини ключа. Якщо n – довжина ключа, то складність такого зламування дорівнює $o(2^n)$.

Зауваження 2. Усі криптографічні системи мають бути ефективними, тобто всі обчислення законним користувачем за криптографічним алгоритмом повинні виконуватися за поліноміальний час. Проблема обґрунтування стійкості криптографічної системи зводиться до доведення відсутності поліноміального алгоритму розв'язання задачі, яка виникає у супротивника при намаганні зламати алгоритм. На жаль, сучасний стан теорії складності не дозволяє доводити над поліноміальні нижні оцінки складності алгоритмів.

7.3. Основи теорії К. Шеннона

Клод Шеннон, а точніше його книга «теорія зв'язку в секретних системах», зробила визначальний внесок у сучасну криптографічну науку. Вважають, що зазначена праця визначила онови та сформулювала сутність сучасної криптографії. Автор розглядав модель, в якій джерело повідомлень утворює відкритий текст M. Джерело ключів генерує K. Шифратор перетворює M за допомогою K у текст С: $C = E_K(M)$. Дешифратор, отримавши C,

використовує оператор: $M = E^{-1}{}_K(C)$. Задачею криптоаналітика є отримання тексту і ключа на основі аналізу шифротексту (рис.7.4).

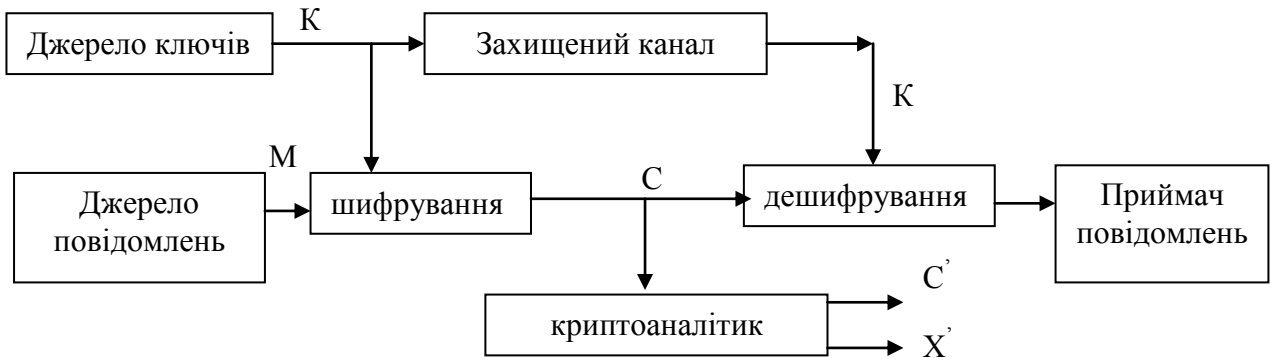

Рисунок 7.4 – Модель обміну повідомленнями

К. Шеннон розглянув питання теоретичної і практичної секретності.

Для визначення теоретичної секретності він сформулював такі питання:

– наскільки стійка система, якщо криптоаналітик противника необмежений часом і володіє всіма необхідними засобами для аналізу криптограм;

– чи має криптограма єдиний розв'язок; який об'єм шифротексту необхідно перехопити крипто аналітику, щоб розв'язок був єдиним.

Для відповіді на ці питання криптолог ввів поняття **цілковитої секретності** за допомогою наступної умови: для будь-яких *C* апостеріорні ймовірності рівні апріорним ймовірностям, тобто перехоплення зашифрованого повідомлення не дає крипто аналітику противника ніякої інформації:

$$P_C(M) = P(M)P_M(C) / P(C),$$

де *P(M)* – апріорна ймовірність повідомлення *M*; $P_M(C)$ – умовна ймовірність криптограми *C* при умові, вибрано повідомлення *M*, тобто сума всіх тих ключів, які переводять повідомлення *M* в криптограму *C*; *P(C)* – ймовірність отримання криптограми *C*; $P_C(M)$ – апостеріорна ймовірність події *M* при умові, що перехоплена криптограма *C*. Для цілковитої секретності значення $P_C(M)$ та *P(M)* повинні бути рівні для всіх *M* і *C*.

Існують криптосистеми, для яких будь-який об'єм перехопленої інформації недостатній для знаходження шифрувального відображення, причому це не залежить від обчислювальної потужності обладнання криптоаналітика. Шифри такого типу називаються безумовно стійкими (за К. Шенноном – ідеально секретними). Криптосистеми, які використовують рівномірний випадковий ключ, який має туж саму довжину , що і відкритий текст називають шифрами із одноразовим блокнотом (на практиці такі шифри не зручні). Шифри, стійкість яких грунтується на значній обчислювальній складності розв'язку криптографічної задачі називають умовно стійкими (практично стійкими за К. Шенноном). Шифри, які грунтуються на обчислювально нездійснених задачах, називають обчислювально стійкими і отримали найбільшу практичну розповсюдженість [9].

8. КЛАСИЧНІ МЕТОДИ СИМЕТРИЧНОГО ШИФРУВАННЯ

8.1. Основні поняття і визначення

Шифрування – перетворюючий процес, при якому вихідний текст замінюється шифрованим текстом.

Дешифрування – обернений процес до шифрування, при якому на основі ключа дешифрований текст перетворюється у вихідний.

Ключ – інформація, необхідна для безперешкодного шифрування чи розшифрування текстів. Зазвичай, ключ представляє собою ряд символів того ж алфавіту, в якому набрана інформація.

Текст – набір елементів алфавіту, що має певний логічний зміст.

Алфавіт – скінчена множина використовуваних для шифрування інформації знаків.

В якості прикладів алфавітів можна привести наступні: алфавіт Z_{34} – 33 букви українського алфавіту + пробіл; алфавіт Z_{33} – 32 букви українського алфавіту + пробіл; алфавіт Z_{256} – символи, що входять в стандартний код ASCII; алфавіт Z_{27} – 26 букв англійського алфавіту + пробіл; бінарний – Z_2 – 0 і 1 та ін.

У загальному виді деякий алфавіт можна представити так – $\sum = \{a_0, a_1, ..., a_м\}$.

Поєднуючи за визначеним правилом букви з алфавіту Е, можна створити нові алфавіти: алфавіт $\sum^2$, що містить $м^2$ біграм $a_0a_0, a_1a_1 \quad a_ma_m$, алфавіт $\sum^3$, що містить $м^3$ триграм $a_0a_0a_0, a_1a_1a_1 \quad a_ma_ma_m$.. У загальному випадку, поєднуючи по n букв, одержуємо алфавіт $\sum^n$, який містить m^n n –грам.

Наприклад: англійський алфавіт – $\sum$ = {ABCDEFGH ... WXYZ}, обсягом m =26 букв дозволяє згенерувати алфавіт з 26^2=676 біграм *AA, AB XZ, ZZ,* алфавіт з 26^3 = 17576 триграм *AAA, AAB ZZX, ZZZ* і т.д.

При виконанні криптографічних перетворень корисно замінити букви алфавіту цілими числами 0,1,2,3,.... Це дозволяє спростити виконання необхідних алгебраїчних маніпуляцій. Наприклад, можна установити взаємно однозначну відповідність між російським алфавітом – $\sum$ = *{АБВГДЕ... ЮЯ}* і множиною цілих Z_{32} = {0,1,2,3,...,31}; між англійським алфавітом – $\sum$ = *{ABCDEF...YZ}* і множиною цілих Z_{26} = {0,1,2,3,...,25}. Надалі буде звичайно використовуватися алфавіт Z_m = {0, 1,2.3, ...m-1}, що містить m «букв» (у виді чисел). Заміна букв традиційного алфавіту числами дозволяє більш чітко сформулювати основні концепції і прийоми криптографічних перетворень.

Основною характеристикою шифру є **криптостійкість,** що визначає його стійкість до розкриття методами криптоаналізу. Звичайно ця характеристика визначається інтервалом часу, необхідним для розкриття шифру. **Ефективність криптолгоритму** – відношення часових затрат криптоаналітика на розкриття шифру до часових затрат криптографа на створення шифровки.

До шифрів, що використовуються для криптографічного захисту інформації, ставиться ряд вимог:

– зашифрований текст повинен піддаватися читанню тільки при наявності ключа дешифрування;

– число операцій, необхідних для розшифрування шляхом повного перебору, повинно виходити за межі можливостей сучасних комп'ютерів та мати часовий запас, з врахуванням технічного прогресу;

– знання алгоритму шифрування для хакер не повинно впливати на надійність системи захисту (все залежить від ключа);

– невелика зміна ключа чи вихідного тексту повинна приводити до суттєвої зміни зашифрованого тексту;

- алгоритм повинен, по можливості, допускати програмну та апаратну реалізаціюта інші.

Тією чи іншою мірою цим вимогам відповідають симетричні шифри.

Процеси шифрування і розшифрування здійснюються в рамках деякої криптосистеми (набір інструкцій, апаратні засоби, комплекс програм, що дозволяють зашифровувати відкритий текст та розшифровувати шифротекст різним способом за допомогою ключа. Характерною рисою симетричної криптосистеми є застосування того самого секретного ключа як при шифруванні, так і при розшифрування повідомлень.

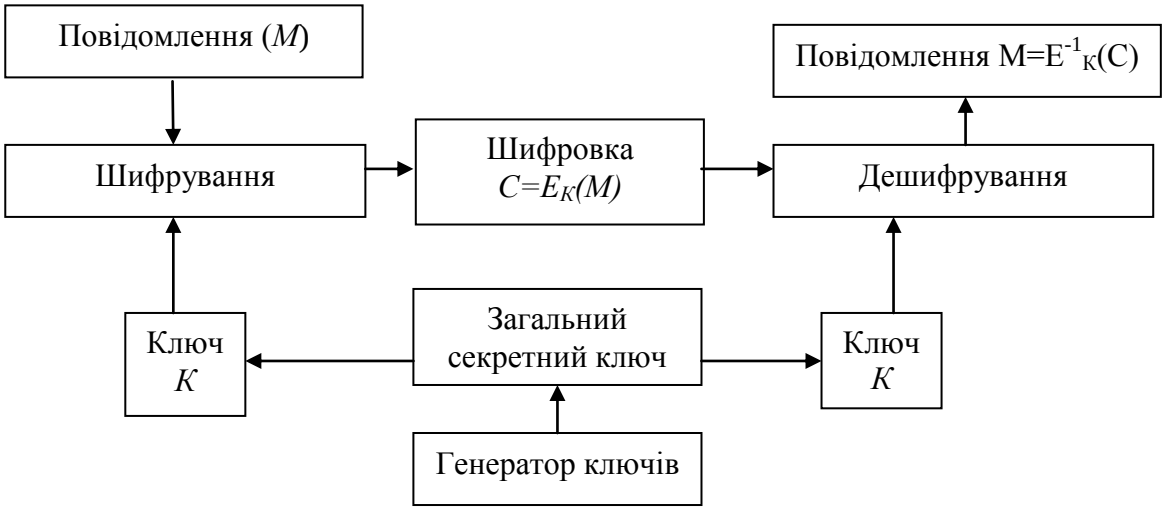

Рисунок 8. 1 – Використання симетричного методу шифрування

Далі ми будемо розглядати класичні симетричні методи шифрування. Як вже зазначалось до них відносяться :**шифри перестановок, шифри підстановок, шифри гамування, шифри аналітичних перетворень, комбіновані шифри** [3, 4].

8.2 Шифри перестановок

Шифрування перестановкою полягає в тому, що символи тексту переставляються за визначеним правилом у межах деякого блоку цього тексту. При достатній довжині блоку, у межах якого здійснюється перестановка, і складному неповторюваному порядку перестановки можна досягти прийнятної для простих практичних додатків стійкості шифру. Шифри перестановки є найпростішими і, імовірно, самими давніми шифрами.

8.2.1. Шифр частоколу

Наприклад, зашифруємо слово «*за шифрування*» із висотою частоколу 2. Для цього запишемо так:

а и р в н я
з ш ф у а н

Тепер зчитуємо спочатку верхній рядок, а потім нижній, отже зашифроване слово: «*аирвнязшфуан*».

8.2.2. Табличні або матричні

З початку епохи Відродження (кінець XIV сторіччя) початку відроджуватися і криптографія. Поряд із традиційними застосуваннями криптографії в політику, дипломатії і військовій справі з'являються й інші задач-захист інтелектуальної власності від переслідувань інквізиції або запозичень зловмисників. У розроблених шифрах перестановки того часу застосовуються таблиці, що шифрують, що у сутності задають правила перестановки букв у повідомленні. Як ключ у табличних методах використовують: розмір таблиці; слово або фраза, що задають перестановку; особливості структури таблиці.

Проста перестановка

Одним із самих примітивних табличних шифрів перестановки є проста перестановка, для якої ключем служить розмір таблиці. Цей метод шифрування подібний із шифром скітала.

Наприклад, повідомлення «*завтра пишемо контрольну роботу*» записується в таблицю по черзі по стовпцях. Результат заповнення таблиці з 5 рядків і 7 стовпців показаний у табл.8.1.

Таблиця 8.1

Заповнення таблиці з 5 рядків і 7 стовпців

з	р	ш	к	р	н	б
а	а	е	о	о	у	о
в	п	м	н	л	р	т
т	и	о	т	ь	о	у

Після заповнення таблиці текстом повідомлення по стовпцях для формування шифротекста зчитують вміст таблиці по рядках. Якщо шифротекст записувати групами по читири букви, виходить таке шифроване повідомлення:

«*зрик рнба аеоо товп мнлр ттио тьоу*»

Природно, відправник і одержувач повідомлення повинні заздалегідь домовитися про загальний ключ у виді розміру таблиці. При розшифруванні дії виконують у зворотному порядку.

Перестановка за ключем

Дещо більшою стійкістю до розкриття володіє метод шифрування, називаний одиночною перестановкою за ключем. Цей метод відрізняється від попередніх тем, що стовпці таблиці переставляються по ключовому слову, фразі або наборові чисел довжиною в рядок таблиці. Цей шифр використовували під час Першої світової війни.

Застосуємо як ключ, наприклад, слово ПЕЛІКАН, а текст повідомлення візьмемо з попереднього прикладу. Дві таблиці, заповнені текстом повідомлення і ключовим

словом, при цьому перша таблиця відповідає заповненню до перестановки, а друга таблиці – заповненню після перестановки.

Таблиця 8.2

Таблиця, заповнена ключовим словом и текстом повідомлення

п	**е**	**л**	**і**	**к**	**а**	**н**
7	2	5	3	4	1	6
з	р	ш	к	р	н	б
а	а	е	о	о	у	о
в	п	м	н	л	р	т
т	и	о	т	ь	о	у

Таблиця 8.3

Перестановка стовпців

1	2	3	4	5	6	7
н	р	к	р	ш	б	з
у	а	о	о	е	о	а
р	п	н	л	м	т	в
о	и	т	ь	о	у	т

У верхньому рядку лівої таблиці записаний ключ, а номери під буквами ключа визначені в залежності з нормальним порядком відповідних букв ключа в алфавите. Якщо б в ключі зустрілись одинакові букви , вони були б пронумеровані зліва на право. У правій таблиці стовпці переставлені в залежності з впорядкованими номерами букв ключа. При зчитуванні вмісту правої таблиці по рядках и запису шифротекста группами по пять букв отримаємо зашифроване повідомлення:

«нркр шбзу аоое оарп нлмт воит ьоут»

Подвійна перестановка

Для забезпечення додаткової таємності можна повторно зашифрувати повідомлення, що вже пройшло шифрування. Такий метод шифрування називається подвійною перестановкою**.** У випадку подвійної перестановки стовпців і рядків таблиці перестановки визначаються окремо для стовпців і окремо для рядків. Спочатку в таблицю записується текст повідомлення, а потім по черзі переставляються стовпці, а потім рядки. При розшифруванні порядок перестановок повинний бути зворотним.

Приклад виконання шифрування методом подвійної перестановки показаний таблиці 8.4. Ключем до шифру подвійної перестановки служить послідовність номерів стовпців і номерів рядків вихідної таблиці (у нашому прикладі послідовності 4132 і 3142 відповідно). Повідомлення – «*прилітаю восьмого»*.

Таблиця 8.4

Приклад виконання шифрування методом подвійної перестановки

	4	1	3	2
3	п	р	и	л
1	і	т	а	ю
4	в	о	с	ь
2	м	о	г	о

	1	2	3	4
3	р	л	и	п
1	т	ю	а	і
2	о	о	г	м
4	о	ь	с	в

	1	2	3	4
1	т	ю	а	і
2	о	о	г	м
3	р	л	и	п
4	о	ь	с	в

Якщо зчитувати шифртекст із правої таблиці по рядках блоками по чотири букви, то вийде наступне: „*тюаї оогм рлип оьсв*”.

Число варіантів подвійної перестановки швидко зростає при збільшенні розміру таблиці: для таблиці 3х3 – 36 (3!х3!) варіантів; для таблиці 4х4 – 576 варіантів і так далі. Однак подвійна перестановка не відрізняється високою стійкістю і порівняно просто «зламується» при будь-якому розмірі таблиці шифрування.

Магічні квадрати

У середні століття для шифрування перестановкою застосовувалися магічні квадрати. Квадрат з прорізаними віконечками, який у вигляді маски накладався на таблицю такого ж розміру, а у віконця записувався текст повідомлення. При цьому після одного заповнення квадрат повертався на 90^0 і так три рази. Після повороту прорублені віконця були над незаповненими клітинами. Після третього повороту всі віконця були заповнені. Є ще поняття псевдомагічних квадратів – після трьох поворотів залишались пусті клітини, які заповняли довільним чином. Для організації шифропередачі на основі цього методу, сторонам потрібно мати два ідентичні квадрати.

Зараз магічними квадратами називають квадратні таблиці записаними в їхні клітки послідовними натуральними числами, починаючи від 1, що дають у сумі по кожнім стовпці, кожному рядкові і кожній діагоналі те саме число.

Таблиця 8.5

Заповнення таблиці

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

→

о	и	р	м
і	о	с	ю
в	т	а	ь
л	г	о	п

«прилітаю восьмого» – «оирм іосю втаь лгоп».

8.3. Шифри підстановок

Шифрування заміною (підстановкою) полягає в тому, що символи тексту заміняються символами того ж або іншого алфавіту відповідно до заздалегідь обумовленої схеми заміни. Вони поділяються на **моноалфавітні, багатоалфавітні (поліалфавітні), гомофонічні.**

8.3.1. Моноалфавітні

Шифр Цезаря. При шифруванні вихідного тексту кожна буква замінюється на іншу букву того ж алфавіту шляхом зміщення на *K* букв. При досяганні кінця алфавіту здійснюється циклічний перехід до початку. Головне, щоб адресант повідомлення знав величину і напрямок зсуву. Існує лише *N!* моноалфавітних підстановок.

Переваги: простота шифрування та дешифрування.

Недоліки: не маскуються частоти появи різних букв вихідного тексту; зберігається алфавітний порядок букв; кількість ключів невелика.

Криптоаналітична атака починається із підрахунку частот появи символів і порівняння з розподілом частот в деякому алфавіті.

Модифікація шифру Цезаря із ключовим словом

Ключове слово використовується для зміщення та зміни порядку символів (бажано, щоб букви були різними, якщо вони повторюються, то їх можна випускати). Ключове слово підписується під буквами алфавіту, починаючи з тієї букви числовий код якої співпадає із вибраним K, $0 \le K \le 25$ – для англійського алфавіту.

Наприклад ключеве слово – *дипломат* і K=5. Тоді зсув буде наступним:

a bcde**fghijklm**nopqrstuvwxyz

vwxyz**diplomat**bcefghijknqrsu

Тепер «*send more money*» – «*hzby tcgd tcbzs*».

Квадрат Полібія

Уперше даний квадрат був розроблений для грецького алфавіту. Для українського алфавіту квадрат може мати вигляд вказаний у таблиці 8.6.

Таблиця 8.6

Заповнення таблиці символами у довільному порядку

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

Для зашифрування повідомлення квадрат використовується так: кожна літера відкритого тексту замінюється на літеру, що знаходиться під нею, якщо літера стоїть в останньому рядочку, то вона замінюється такою, що стоїть у першому рядочку саме над нею.

Отже «*зустріч перенесемо на завтра» – «бр;.жзгїужуду;удшдябяь.жя*».

Модифікація даної таблиці – таблиця Трісімуса, де букви розташовуються в залежності від ключа. Ключ першим вписується в таблицю, а потім всі наступні букви, що не входять до ключа.

Шифр Плейфера (або play-fair)

Цей шифр теж використовувався в першій світовій війні. Застосовується на підстановці не окремих символів, а пари символів (2-грами). Алфавіт розташовується в таблиці в довільному порядку. Вихідний текст розбивається на 2-грами (кількість букв парна), біграми не повинні містити однакові букви. Якщо дві букви не належать одному рядку чи стовпцю , то знаходяться букви в кутках прямокутника, що і визначає пару букв. Якщо дві букви належать одному рядку чи стовпцю, то пишуться букви, що знаходяться (**справа**) **або** (**під**) ними. Шифрування повідомлення із попереднього прикладу на основі таблиці 6 буде наступним:

«*зу ст рі чп ер ен ес ем он аз ав тр а.» – «кн кх цж гє уж уд уг ще шж яі ія .у ;ю»*.

Афінна підстановка

Розглянемо на прикладі: m=26 (англійський алфавіт), a=3, b=5 – вибрані числа, так щоб виконувались умови: $0 \le a,b \le m$, $HCD(a,m)=1$. Заміна

здійснюється за формулою: $E_{a,b}(t) = at + b(\mathrm{mod}\, m)$. Перевіримо НСД(3, 26)=1. Тому зсув відбувається за формулою $3t+5$, де $t \in (0,25)$.

Таблиця 8.7

Заміна символів

t	0	1	2	3	4	5	…..	25
3t+5	5	8	11	14	17	20	…..	2
Початкова	A	B	C	D	E	F	….	Z
Після заміни	F	I	L	..	..	..	..	C

Текст «*hope*» – «*avyk*».

Переваги: зручне керування ключами, ключі представляються у вигляді пари чисел *a, b*. Недоліки: легко знайти формулу.

8.3.2. Поліалфавітні

Шифри складної заміни називають багатолфавітними шифрами, так як для шифрування кожного символу вихідного повідомлення застосовується свій шифр простої заміни. Багато алфавітна підстановка послідовно і циклічно змінює використовувані алфавіти. Ефект багато алфавітної підстановки полягає в тому, щоб забезпечити маскування справжньої статистики мови, так як конкретний символ із вихідного алфавіту може бути перетворений в декілька різних символів інших алфавітів.

Багато алфавітні шифри замни запропонував і ввів в практику криптографії Леон Батис Альберті, який був відомим архітектором і теоретиком мистецтва криптології. Його книга «Трактат про шифр» написана в 1566 році, представила собою першу в Європі наукову працю по криптології. Крім шифру багатоалфавітної підстановки він описав пристрій із обертанням коліс для його реалізації. Криптологи всього світу вважають його основоположником криптології.

Система шифрування Віжінера

Система шифрування Віжінера вперше була опублікована в 1586 році. Свою назву вона отримала за іменем французького дипломата Блеза Віжінера, який розвивав і вдосконалював криптографію. У процесі шифрування і дешифрування використовувалася таблиця Віжінера (квадрат), що утворена наступним чином: у перший рядок виписується весь алфавіт, в наступних рядках вводиться зсув на 1 літеру вліво, так отримується квадратна таблиця; щоб зашифрувати повідомлення, вибирають лозунг, який підписують під текстом; потім у стовпчику шукаємо літеру повідомлення, а у рядку – ключа, на їх перетині знаходиться наша підстановка.

Також цю криптосистему можна розглядати як шифр гамування із використанням періодичної гами малого періоду.

Шифрування можна записати: Ci=(Mi+Ki(mod u))mod L, де C_i, M_i – числові еквіваленти символів криптограми та початкового тексту, K_i – числовий еквівалент букви ключа, L – потужність алфавіту, u – довжина ключа.

Розшифрування запишемо у вигляді: Mi=(Ci-Ki(mod u))mod L.

Багатопетлева поліалфавітна підстановка є цікавим шифром заміни. У багатопетлевому шифрі використовується не один, а декілька ключів шифрування. Їх називають петлевими або первинними ключами. Багатопетлевий шифр описується формулою

Ci =(Mi+K1,i*mod*u1+K2,i*mod*u2+...+Kj,i*mod*uj+...+Kg,I*mod*ug)*mod*L

g - кількість петель шифру, uj - довжина j-ого первинного ключа.

Складений ключ дорівнює сумі первинних ключів. На відміну від шифру Віженера складений ключ багатопетлевих підстановок не є осмисленим словом і має набагато більший період.

Приклад: Зашифруємо повідомлення «*переходьте до виконання плану номер два*» на ключі *резидент*.

п е р е х о д ь т е д о в и к о н а н н я п л а н у н о м е р д в а

р е з и д е н т р е з и д е н т р е з и д е н т р е

е і ш л щ у с п з і й ч є л ь є г е х ц г ф ю т г ш х ч р і г ц т е

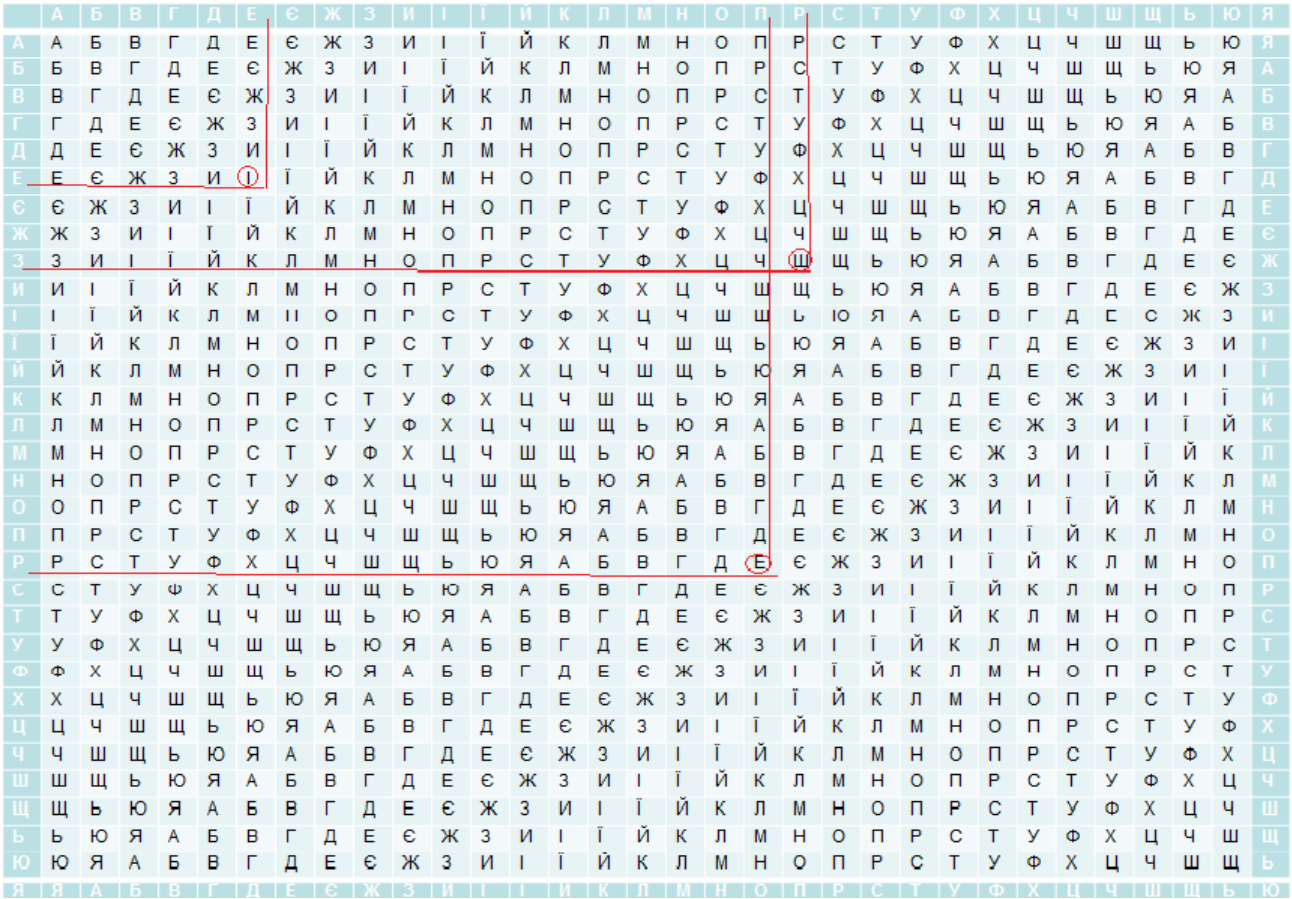

Рисунок 8.2 – Таблиця Віжінера

Система шифрування методом Вернама

Система Вернама є частинним випадком шифру Віжінера при m=2 (алфавіт 0 і 1). Версія цього шифру запропонована в 1926 році Гілбертом Вернамом, співробітником фірми AT&T США. Кожен символ вихідного відкритого тексту із англійського алфавіту $\{A,B,C,\ldots,Z\}$, що розширений 5 допоміжними знаками (пробіл і т. д), спочатку кодувався в 5-бітовий блок телеграфного коду Бодо. Далі випадкова послідовність ключів $k_0,k_1,\ldots$ наперед

записувалась н паперовій стрічці. $\oplus-$ операція дадавання за модулем 2. Схема передачі з використанням шифру Вернама показана на рис.8.3.

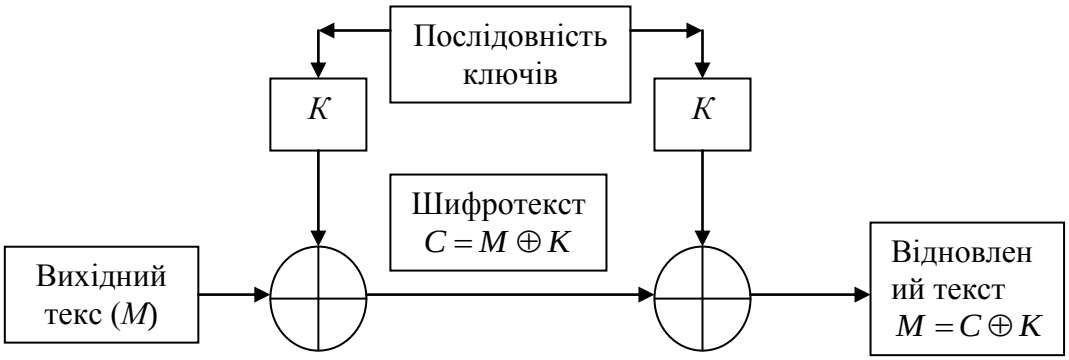

Рисунок 8.3 – Система шифрування за методом Вернама

Шифрування:		
Текст	11001110101	$-M$
Ключ	10101010101	$-K$
Сума	21102120202	
Сума за модулем 2	01100100000	$-C$
Дешифрування:		
Шифр	01100100000	$-C$
Ключ	10101010101	$-K$
Сума	11201110101	
Сума за модулем 2	11001110101	$-M$

Роторні машини

У 20-х роках XX століття було винайдено електромеханічні пристрої шифрування, що автоматизують процес шифрування. Принцип роботи таких машин базується на багатоалфавітній заміні символів початкового тексту довгим ключем згідно з версією шифру Віженера. Це американська машина **SIGABA** (М-134), англійська **TYPEX**, німецька **ENIGMA**, японська **PURPLE** були роторними машинами.

Гомофонічні

Одному елементу ставиться у відповідність декілька символів шифротексту. Цей метод застосовується для викривлення статистичних властивостей тексту.

Таблиця 8.8

Заміна символів гомо фонічної підстановки

Алфавіт тексту	а	б	...	і	ж	з	...	м	н	...
Алфавіт шифру	17	23		97	47	76		32	55	
	31	44		51	67	19		28	84	
	48	63		15	33	59		61	34	

Приклад: «*заміна*» – «*761732975531*».

Таким чином, при гомофонічній підстановці кожна буква відкритого тексту замінюється по черзі цифрами відповідного стовпця.

8.4. Гамування

Шифрування гамування полягає в том, що символи тексту складаються із символами деякої випадкової послідовності, іменованою гамою шифру. Стійкість шифрування визначається в основному довжиною (періодом) неповторюваної частини гами шифру. Оскільки за допомогою комп'ютера можна згенерувати практично нескінченну гаму шифру, те даний спосіб є одним з основних для шифрування інформації в автоматизованих системах. Перед шифруванням відкритий текст поділяється на блоки однакової довжини $T_0^{(i)}$, зазвичай по 64 біти. Гама шифру виробляється у вигляді послідовності блоків аналогічної довжини $Г_ш^{(i)}$.

$$T_ш^{(i)} = Г_ш^{(i)} \oplus T_0^{(i)},\ i=1, ..к$$ – процес шифрування;

$$T_0^{(i)} = Г_ш^{(i)} \oplus T_ш^{(i)},\ i=1, ..к$$ – процес дешифрування,

де $T_ш^{(i)}$ – i-й блок шифру; $T_0^{(i)}$ – i-й блок відритого тексту; $Г_ш^{(i)}$ – i-й блок гами; $к$ – кількість блоків.

Отриманий цим методом шифротекст є надто складним для розкриття, оскільки ключ є постійно змінним. По суті, гама шифру має змінюватися випадково для кожного зашифровуваного блока. Якщо період гами перевищує довжину всього зашифровуваного тексту і зловмисникові невідома жодна з частин вхідного тексту, то такий шифр можна розкрити тільки безпосереднім перебиранням усіх варіантів ключа. У цьому випадку криптостійкість шифру визначається довжиною ключа.

Одноразова система шифрування (одноразовий блокнот)

Майже всі застосовувані на практиці шифри характеризуються як умовно надійні, оскільки вони можуть бути розкриті за наявності необмежених обчислювальних можливостей. Абсолютно надійні шифри не можна зруйнувати навіть за використанням необмежених обчислювальних можливостей. Існує єдиний такий шифр, застосовуваний на практиці - одноразова система шифрування. Характерною рисою цієї системи є одноразове використання ключової послідовності.

Одноразову систему винайдено 1917 року американцями

Дж. Моборном та Г. Вернамом. Використовують одноразовий нотатник з відривними листками, на кожному з листів надруковано таблицю випадкових чисел. Нотатник використовується в двох екземплярах. Для кожного символу використовується свій ключ, після того як таблицю використано, її вилучають. Шифрування нового повідомлення починається з нового листа.

Цей шифр є абсолютно надійним, якщо набір ключів справді випадковий. Теоретично доведено, що одноразові системи є нерозкривними системами.

Однак можливості використання таких систем є обмеженими практичними аспектами - ключова послідовність довжиною вихідного повідомлення, повинна передаватись одержувачеві заздалегідь якимось секретним каналом, це є проблемою при опрацюванні великих об'ємів інформації.

8.5. Аналітичні методи

Шифрування аналітичним перетворенням полягає в тому, що текст перетвориться за деяким аналітичним правилом.

Наприклад, можна використовувати правило множення вектора на матрицю, причому множена матриця є ключем шифрування (тому її розмір і зміст повинні зберігатися в секреті), а символами множеного вектора послідовно служать символи зашифрованого тексту

Шифр Хілла

Перетворення: $y_i = T \cdot x_i (\mathrm{mod}\, m)$, де T – лінійне перетворення, що описується матрицею. *T* повинне мати обернене перетворення, а також необхідно щоб визначник *T* не ділився на будь-яке просте p, яке ділить *m* (розмір алфавіту).

Розглянемо цей метод на прикладі. Нехай *m*=26=13*2, $T = \begin{pmatrix} 3 & 3 \\ 2 & 5 \end{pmatrix}$ – матриця перетворення, *Det(T)*=9, що не ділиться ні на 2, ні на 13, а також існує обернене перетворення (бо визначник не рівний нулю).

Спочатку текст «*paymoremoney*» розбивають на біграми (можна і на триграми і т.д.), потім кожну букву в біграмі замінюють її порядковим номером в алфавіті, починаючи з нуля.

p a	y m	o r	e m	o n	e y
15 0	24 12	14 17	4 12	14 13	4 24

$$y_1 = \begin{pmatrix} 3 & 3 \\ 2 & 5 \end{pmatrix} \cdot \begin{pmatrix} 15 \\ 0 \end{pmatrix} = \begin{pmatrix} 45 \\ 30 \end{pmatrix} \mathrm{mod}\, 26 = \begin{pmatrix} 19 \\ 4 \end{pmatrix};$$

$$y_2 = \begin{pmatrix} 3 & 3 \\ 2 & 5 \end{pmatrix} \cdot \begin{pmatrix} 24 \\ 12 \end{pmatrix} = \begin{pmatrix} 4 \\ 4 \end{pmatrix};$$

$$y_3 = \begin{pmatrix} 3 & 3 \\ 2 & 5 \end{pmatrix} \cdot \begin{pmatrix} 14 \\ 17 \end{pmatrix} = \begin{pmatrix} 15 \\ 9 \end{pmatrix};$$

$$y_4 = \begin{pmatrix} 3 & 3 \\ 2 & 5 \end{pmatrix} \cdot \begin{pmatrix} 4 \\ 12 \end{pmatrix} = \begin{pmatrix} 22 \\ 16 \end{pmatrix};$$

$$y_5 = \begin{pmatrix} 3 & 3 \\ 2 & 5 \end{pmatrix} \cdot \begin{pmatrix} 14 \\ 13 \end{pmatrix} = \begin{pmatrix} 3 \\ 15 \end{pmatrix};$$

$$y_6 = \begin{pmatrix} 3 & 3 \\ 2 & 5 \end{pmatrix} \cdot \begin{pmatrix} 4 \\ 24 \end{pmatrix} = \begin{pmatrix} 6 \\ 24 \end{pmatrix};$$

Отже, шифротекст . згідно тієї ж таблиці – «*teeeyiwqdygy*».

Розшифрування здійснюють в зворотньому порядку за формулою

$x_i = T^{-1} \cdot y_i (\mathrm{mod}\, m)$

При складанні комбінованих шифрів необхідно проявляти обережність, так як неправильний вибір складових шифрів може привести до вихідного тексту.

9. СИМЕТРИЧНІ КРИПТОАЛГОРИТМИ

9.1. Блокові складені шифри

Вище вже йшлося про можливість зламування шифру простої заміни у випадку, коли розмір шифрованого тексту значно перевищує розміри абетки відкритого та шифрованого текстів (шифр простої заміни не піддається зламуванню лише при шифруванні повідомлень в одне-два слова). Звідси випливають такі два підходи до підсилення шифру заміни: 1) збільшення абетки (реалізується у шифрах багатозначної заміни, у кодах та сучасних блокових шифрах, коли відкритий текст поділяють на блоки і виконують просту заміну блоків); 2) зменшення кількості знаків, що шифруються за однією заміною (реалізується у потокових шифрах).

Шифр Фейстеля. За Шенноном трійка множин $A^*\times K\to B^*$ з функцією $E: A^*\times K\to B^*$ тоді являтиме собою шифр, якщо ця функція сюр'єктивна і при будь-якому фіксованому ключі $k \in K$ функція E ін'єктивна.

Нехай множини відкритих текстів $A^*=\{x_1;x_2;\dots;x_n\}$ та криптограм $B^*=\{y_1;y_2;\dots;y_n\}$ збігаються, тобто $A^*=B^*=M$. На множині ключів $k_j \in K_i$ визначимо сімейство сюр'єктивних та ін'єктивних функцій f_j і далі за допомогою композиції шифрів побудуємо шифр, заданий відображенням

$E: M\times\{K_1\times K_2\times\dots\times K_r)\to M,$ де $E=f_r o f_{r-1} o \dots o f_2 o f_1$. Такий шифр називають складеним (композиційним), самі функції (перетворення) f_i – *і-им* раундом шифрування, ключ κ_i – і-им раундовим ключем. Ключем складеного шифру вважається вектор $(k_1,k_2,\dots,k_n) \in K_1\times K_2\times\dots\times K_n$.

Зауваження 1. У деяких випадках раундові ключі утворюються з ключа всієї криптосистеми завдяки алгоритму створення раундових ключів (при цьому розмір ключа системи значно менший за сумарний розмір усіх раундових ключів).

Звичайно, абеткою, на якій діє блоковий шифр, є множина двійкових векторів-блоків однакової довжини відкритого тексту. Ідея створення складених блокових шифрів належить К. Шеннону. Загальний принцип побудови шифрувальних перетворень блокових шифрів – принцип перемішування. Під перемішуванням розуміють таке відображення векторного простору самого на себе, коли кожна компактна область простору (по можливості) відображається у більшу, некомпактну область. Проілюструємо це поняття прикладом. Розглянемо перетворення $y = x \oplus a$, де $\oplus$ – операція покоординатного додавання векторів за модулем 2; a – (1,1,1); x – двійковий вектор, що належить множині векторів: $x_0 = (0,0,0); x_1 = (0,0,1); x_2 = (0,1,0); x_3 = (0,1,1); x_4 = (1,0,0); x_5 = (1,0,1); x_6 = (1,1,0); x_7 = (1,1,1)$.

Визначимо:

$$x_0 \oplus a = (0,0,0)+(1,1,1) = (1,1,1) = x_7 \qquad x_4 \oplus a = (1,0,0)+(1,1,1) = (0,1,1) = x_3$$
$$x_1 \oplus a = (0,0,1)+(1,1,1) = (1,1,0) = x_6 \qquad x_5 \oplus a = (1,0,1)+(1,1,1) = (0,1,0) = x_2$$
$$x_2 \oplus a = (0,1,0)+(1,1,1) = (1,0,1) = x_5 \qquad x_6 \oplus a = (1,1,0)+(1,1,1) = (0,0,1) = x_1$$
$$x_3 \oplus a = (0,1,1)+(1,1,1) = (1,0,0) = x_4 \qquad x_7 \oplus a = (1,1,1)+(1,1,1) = (0,0,0) = x_0$$

Введене перетворення не є розмішувальним, бо, наприклад, компактна область векторів x_1,x_2,x_3 відобразилася в іншу компактну область x_6,x_5,x_4. Якщо ж розглядати вектори як числа та застосувати підстановку $\begin{pmatrix}01234567\\17546032\end{pmatrix}$, то компактна область x_0, x_1, x_2 відобразиться в некомпактну область x_1, x_7, x_5. Тому ця підстановка є перемішувальною.

К. Шеннон запропонував перемішувати вихідні дані за допомогою суперпозиції деяких простих некомутуючих перетворень. Саме такий підхід став у наш час магістральним при побудові блокових шифрів, коли до блоків відкритого тексту багаторазово застосовують деякі базові перетворення двох типів: 1) криптографічно складні локальні перетворення над окремими частинами блоків тексту та 2) прості перетворення, завдяки яким ці блоки переставляються між собою. Перетворення першого типу називаються перемішувальними, а другого типу –розсіювальними. Алгоритм шифрування виконує декілька раундів (ітерацій). Раундові шифри складаються з перетворень першого та другого типів (операції заміни двійкових кодів невеликої розрядності, перестановки елементів двійкових кодів, арифметичні та логічні операції над двійковими кодами). Схеми, що реалізують ці перетворення, називаються *SP*-мережами (від перших літер англійських слів substitution – підстановка та permutation –перестановка). Саме так при шифруванні за допомогою SP-мереж приховується статистична залежність між літерами відкритого тексту і ускладнюється залежність між ключем та шифротекстом.

Зауваження 2. Перевага складених шифрів – у симетричності щодо шифрування та дешифрування, тому їх можна реалізовувати на одному пристрої. Перехід від шифрування до дешифрування забезпечується заміною послідовності раундових ключів на обернену.

Для побудови блокових складених шифрів часто використовують конструкцію під назвою *петлі Фейстеля,* схема якою зображена на рис. 1. Нехай вихідний блок даних *B* (відкритого тексту *x* чи закритого тексту *y)* має довжину *n.* Визначимо і-ий раунд шифрування блока таким чином. Розіб'ємо блок на два півблоки: *L* (лівий старший) і *R* (правий молодший) удвічі менших розмірів, тобто $B = (L; R);\ |B| = n;\ |L| = |R| = n/2$ [10].

Зашифруємо лівий старший півблок *L* із залученням правого молодшого півблока *R,* використовуючи деяку функцію f_i, що залежить від раундового ключа k_i і бінарну операцію – ○, яка має обернену операцію – ●. Для підготовки до наступного аналогічного $(i+1)$-го раунду поміняємо місцями лівий та правий півблок. Отже, функція шифрування *i*-го раунду набуде вигляду $F_i(x)=F_i(L;R)=(R;L\circ f_i(R))$. Обернене перетворення дешифрування – це $F_i^{-1}(y)=F_i^{-1}(L;R)=(R;L\bullet f_i(R))$.

Блокові шифри, побудовані за таким принципом, відрізняються довжиною блоків, функцією f_i та кількістю раундів. У більшості з них розрядність вихідного блока дорівнює 64 бітам, а операції ○ і ● – це побітове додавання векторів за модулем 2 (операція *XOR).*

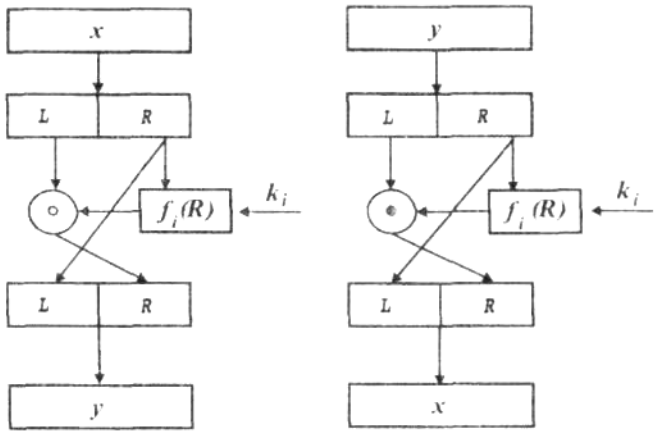

Рисунок 9.1 – Петля Фейстеля

9.2. Алгоритм DES

У кінці 60-х IBM почала науково-дослідний проект у галузі комп'ютерної криптографії, який очолив Хорст Фейстель. У результаті роботи над проектом до 1971 року був створений алгоритм під кодовою назвою LUCIFER, який продали банку Ллойда (Lloyd's of London) для використання в системі управління оборотом готівкових коштів, теж розробленій IBM. Шифр LUCIFER являв собою блоковий шифр Фейстела, що оперував блоками розміром 64 біти, використовуючи ключ довжиною 128 бітів. Базуючись на багатообіцяючих результатах проекту LUCIFER, IBM взяла курс на створення комерційного варіанта шифру, який, в ідеалі, можна було б розмістити в одній мікросхемі. Цей напрямок очолили Уолтер Тачман і Карл Мейер, залучивши не тільки спеціалістів з IBM, але й консультантів і технічних спеціалістів. У результаті їхніх зусиль була створена вдосконалена версія шифру LUCIFER, що мала більшу криптостійкість, але зменшений до 56 бітів ключ, щоб алгоритм міг вміститися в одну мікросхему.

У 1973 році Національне бюро стандартів оголосило конкурс на найкращий проект по створенню загальнодержавного стандарту шифрування. Компанія IBM подала на конкурс результати проекту Тачмана-Мейера, їхній алгоритм виявився безумовно найкращим з усіх запропонованих, і в 1977 році він був затверджений ж стандарт шифрування даних – DES(Data Encription Standard).

Але перш ніж стати офіційним стандартом, запропонований IBM алгоритм зазнав жорстокої критики, яка не вщухає до сьогодні. Нападок зазнають, в основному, дві особливості шифру. По-перше, в початковому алгоритмі LUCIFER фірми IBM використовувалися ключі довжиною 128 бітів, а в запропонованому стандарті довжина ключа зменшена до 56 бітів. Критики побоювалися (і побоюються досі), що такий розмір ключа занадто малий для того, щоб шифр міг гарантовано протистояти спробам криптоаналізу з простим перебором всіх можливих варіантів (сьогодні, з бурхливим розвитком комп'ютерної техніки, реалізація такої атаки стала фактом). Другий серйозний випад критиків спрямований проти факту засекреченості внутрішньої структури DES, а саме структури S-матриць. Тому користувачі не могли бути впевнені в тому, що у внутрішній структурі DES немає якихось дефектів, за допомогою яких спеціалісти

АНБ могли б розшифрувати повідомлення, не маючи ключа. Подальші дослідження, зокрема останні праці з різницевого криптоаналізу, дозволяють із більшою впевненістю стверджувати, що DES має досить надійну внутрішню структуру. Більше того, за ствердженнями учасників проекту з боку IBM, єдиними змінами, внесеними в представлений ними проект стандарту, були запропоновані АНБ зміни в структурі S-матриць із метою укріплення ймовірних слабких місць алгоритму, знайдених у ході оцінки проекту.

Так чи інакше, DES побачив світ і став дуже популярним, особливо у фінансових застосуваннях. В 1994 році NIST (National Institute of Standards and Technology) продовжив використання DES як федерального стандарту ще на п'ять років і рекомендував його до застосування в комерційних структурах.

Алгоритм DES також використовує комбінацію підстановок і перестановок. DES здійснює шифрування 64-бітових блоків даних за допомогою 64-бітового ключа, у якому значущими є 56 біт (інші 8 біт-перевірочні біти для контролю на парність). Дешифрування в DES є операцією, зворотною шифруванню, і виконується шляхом повторення операцій шифрування в зворотній послідовності. Узагальнена схема процесу шифрування в алгоритмі DES показана на рис.9.1. Процес шифрування полягає в початковій перестановці бітів 64-бітового блоку, шістнадцятьох циклах шифрування і, нарешті, у кінцевій перестановці бітів.

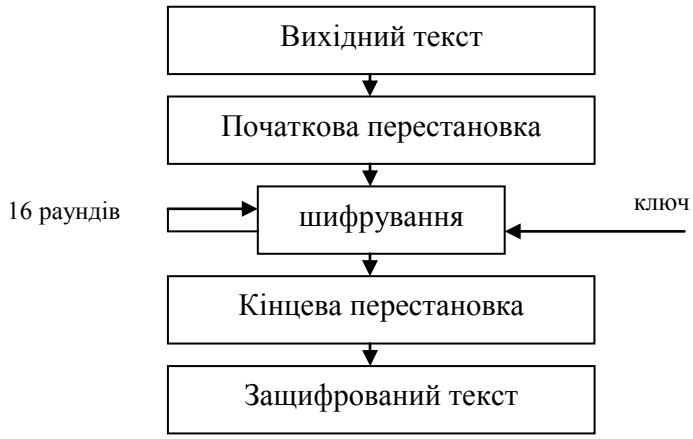

Рисунок 9.2 – Узагальнена схема шифрування в алгоритмі DES

Слід відразу зазначити, що всі таблиці, що приводяться, є стандартними і повинні включатися в реалізацію алгоритму DES у незмінному виді.

Усі перестановки і коди в таблицях підібрані розроблювачами таким чином, щоб максимально утруднити процес розшифровки шляхом підбору ключа. При описі алгоритму DES (рис. 2) застосовані наступні позначення:

– *L* і *R*-послідовності бітів (ліва – left) і права – right);

– *LR* – конкатенація послідовностей *L* і *R*, тобто така послідовність бітів, довжина якої дорівнює сумі довжин *L* і *R*; у послідовності *LR* біти послідовності *R* слідують за бітами послідовності *L*;

– ⊕ – операція побітового додавання за модулем 2.

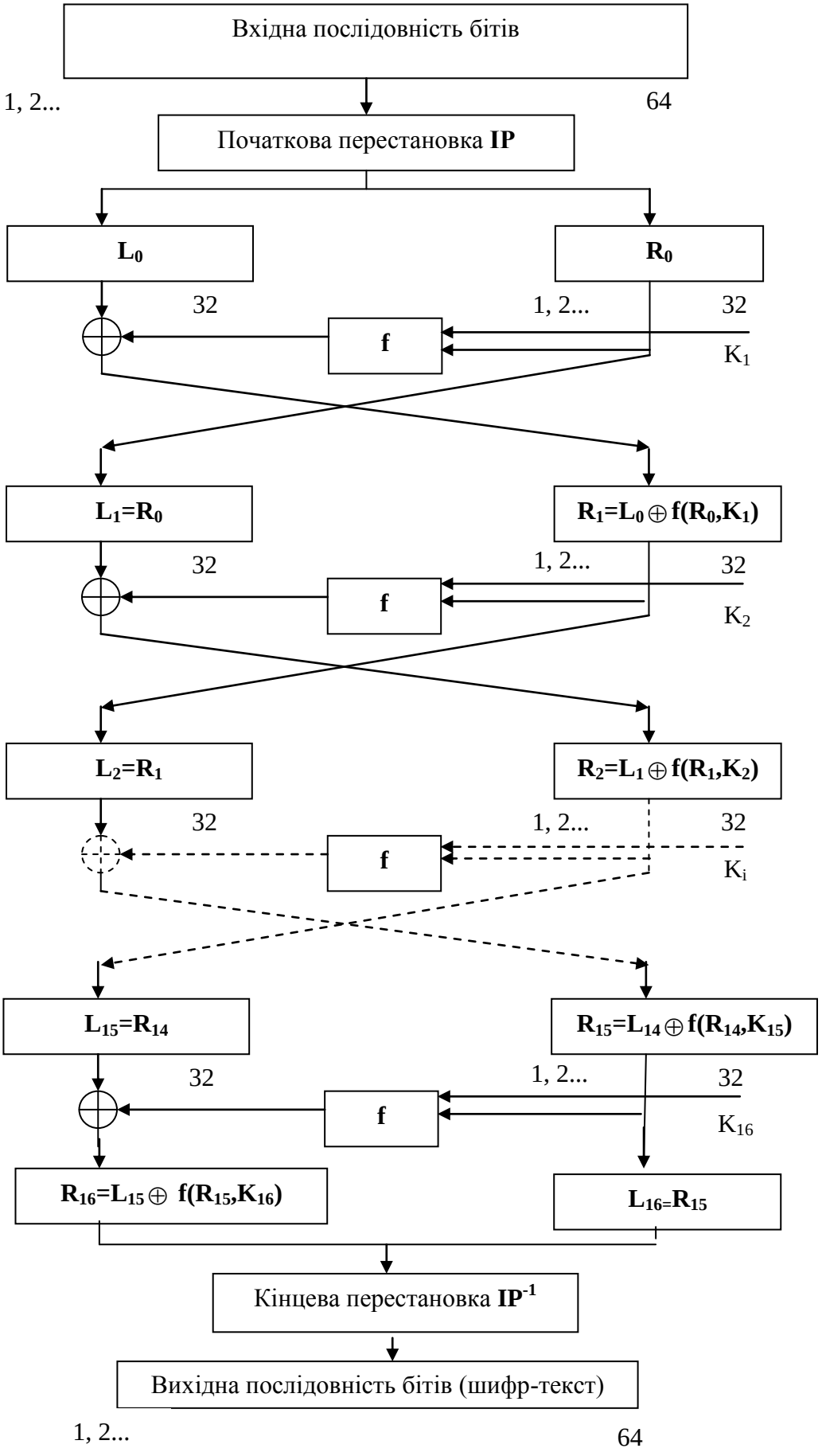

Рисунок 9.3 – Структура алгоритму DES

Нехай з файлу вхідного тексту зчитаний черговий 64-бітовий (8-байтовий) блок *T*. Цей блок *T* перетвориться за допомогою матриці початкової перестановки *IP* (табл.2). Біти вхідного блоку *T* (64 біти) переставляються відповідно до матриці *IP*: біт 58 вхідного блоку *T* стає бітом 1, біт 50-битому 2 і т.д. Цю перестановку можна описати вираженням $T_0=IP(T)$. Отримана послідовність бітів T_0 розділяється на дві послідовності: L_0- ліві або старші біти, R_0 – праві або молодші біти, кожна з яких містить 32 біти.

Таблиця 2

Матриця початкової перестановки *IP*

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

Таблиця 3

Матриця оберненої перестановки IP^{-1}

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

Потім виконується ітеративний процес шифрування, що складає з 16 кроків (циклів). Нехай T_i – результат *i*-ї ітерації: $T_i = L_i R_i$, де $L_i = t_1 t_2 ... t_{32}$ (перші 32 біти); $R_i = t_{33}t_{34} ... t_{64}$ (останні 32 біти).

Функція f називається функцією шифрування. Її аргументами є послідовність R_{i-1}, одержувана на попередньому кроці ітерації, і 48-бітовий ключ K_i, що є результатом перетворення 64-бітового ключа шифру *K*. На останньому кроці ітерації одержують послідовності R_{16} і L_{16} (без перестановки місцями), що з'єднуються в 64-бітову послідовність $R_{16}L_{16}$.

По закінченні шифрування здійснюється відновлення позицій бітів за допомогою матриці зворотної перестановки IP^{-1}(табл. 3).

Процес розшифрування даних є інверсним стосовно процесу шифрування. Усі дії повинні бути виконані в зворотному порядку. Це означає, що дані, що розшифровуються, спочатку переставляються відповідно до матриці IP^{-1}, а потім над послідовністю бітів $R_{16}L_{16}$ виконуються ті ж дії, що й у процесі шифрування, але в зворотному порядку. Таким чином, для процесу розшифрування з переставленим вхідним блоком $R_{16}L_{16}$ на першій ітерації використовується ключ K_{16}, на другій ітерації – K_{15} і т.д. На 16-й ітерації використовується ключ K_1. На останньому кроці ітерації будуть отримані

послідовності L_0 и R_0 які перетворюються в 64-бітову послідовність L_0R_0. Потім у цій послідовності 64 біти переставляються відповідно до матриці IP. Результат такого перетворення – вихідна послідовність бітів (розшифроване 64-бітове значення).

Тепер розглянемо, що ховається під перетворенням *f*. Схема обчислення функції шифрування $f(R_{i-1}K_i)$ показана на рис.9.4.

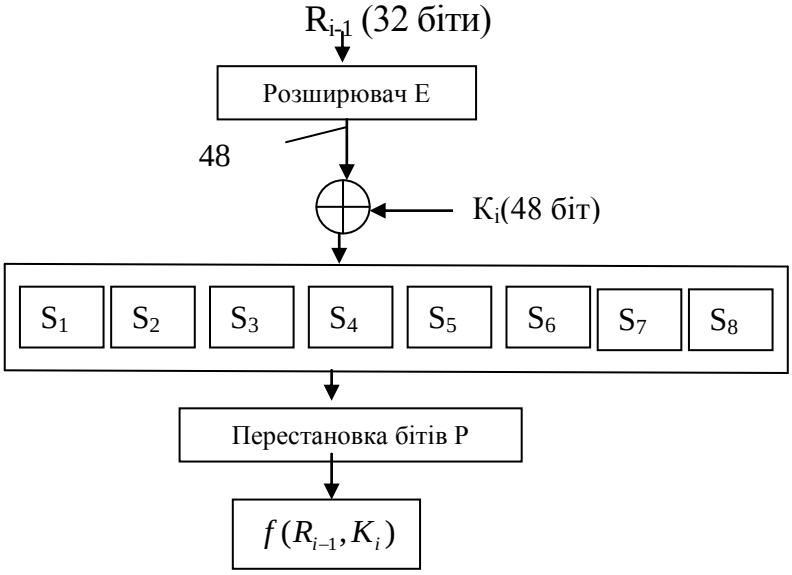

Рисунок 9.4 – Схема обчислення функції шифрування *f*

Для обчислення значення функції *f* використовуються:

– функція *E* (розширення 32 біт до 48);

– функція S_1 S_2,..., S_8 (перетворення 6-бітового числа в 4-бітове);

– функція *P* (перестановка бітів у 32-бітовій послідовності).

Приведемо визначення цих функцій.

Аргументами функції шифрування *f* є R_{i-1}(32 біти) і K_i (48 біт). Результат функції $E(R_{i-1})$ є 48-бітове число. Функція розширення *E*, що виконує розширення 32 біт до 48 (приймає блок з 32 біт і породжує блок з 48 біт), визначається із табл.9.4.

Таблиця 9.4

Функція розширення *E*

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

Потім здійснюється перетворення $E(R_{s-1}) \oplus K_i = B_1B_2...B_8$. Кожний з цих блоків використовується як номер елементу у функціях-матрицях S_1, S_2...S_8, що містять 4-бітові значення (табл.9.5). Слід зазначити, що вибір елементу в матриці S_i здійснюється досить оригінальним способом. Нехай на вхід матриці S_i надходить 6-бітовий блок $B_i = b_1b_2b_3b_4b_5b_6$ тоді двобітове число b_1b_6 вказує номер

рядка матриці, а чотирибітове число $b_2b_3b_4b_5$ – номер стовпця. Наприклад, якщо на вхід матриці S_i надходить 6-бітовий блок $B_i = b_1b_2b_3b_4b_5b_6 =$ 100110, то 2-бітове число $b_1b_6 = 10_{(2)} = 2_{(10)}$ вказує рядок з номером 2 матриці S_i а 4-бітове число $b_2b_3b_4b_5 = 0011_{(2)} = 3_{(10)}$ вказує стовпець з номером 3 матриці S_i. Це означає, що в матриці S_i блок B_i = 100110 вибирає елемент на перетинанні рядка з номером 2 і стовпця з номером 3, тобто елемент $8_{(10)} = 1000_{(2)}$.

Таблиця 9.5

Функції перетворення S_1, S_2....S_8

	НОМЕР СТОВПЦЯ																
	0	1	2	**3**	4	5	6	7	8	9	10	11	12	13	14	15	
НОМЕР РЯДКА 0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7	S_1
1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8	
2	4	1	4	**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	S_2
0	15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10	
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	S_3
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	
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	S_4
0	7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15	
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	S_5
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	
1	14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6	S_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	
1	10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8	S_7
2	9	14	15	5	2	8	12	3	7	0	4	10	1	13	1	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	S_8
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	
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	

У результаті одержуємо $S_1(B_1)$ $S_2(B_2)$ $S_3(B_3)$... $S_8(B_8)$, тобто 32-бітовий блок (оскільки матриці S_i містять 4-бітові елементи). Цей 32-бітовий блок перетвориться за допомогою функції перестановки бітів P (табл.6).

Таким чином, функція шифрування $f(R_{i-1}, K_i) = P(S(B_1),...,S_8(B_8))$.

Як неважко помітити, на кожній ітерації використовується нове значення ключа K_i, (довжиною 48 біт). Нове значення ключа K_i обчислюється з початкового ключа K (рис.5). Ключ K являє собою 64-бітовий блок з 8 бітами контролю парності, розташованими в позиціях 8,16, 24, 32, 40, 48, 56, 64. Для

видалення контрольних бітів і підготовки ключа до роботи використовується функція **G** первісної підготовки ключа (табл. 9.6).

Таблиця 9.6

Функція *P* перестановки бітів

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

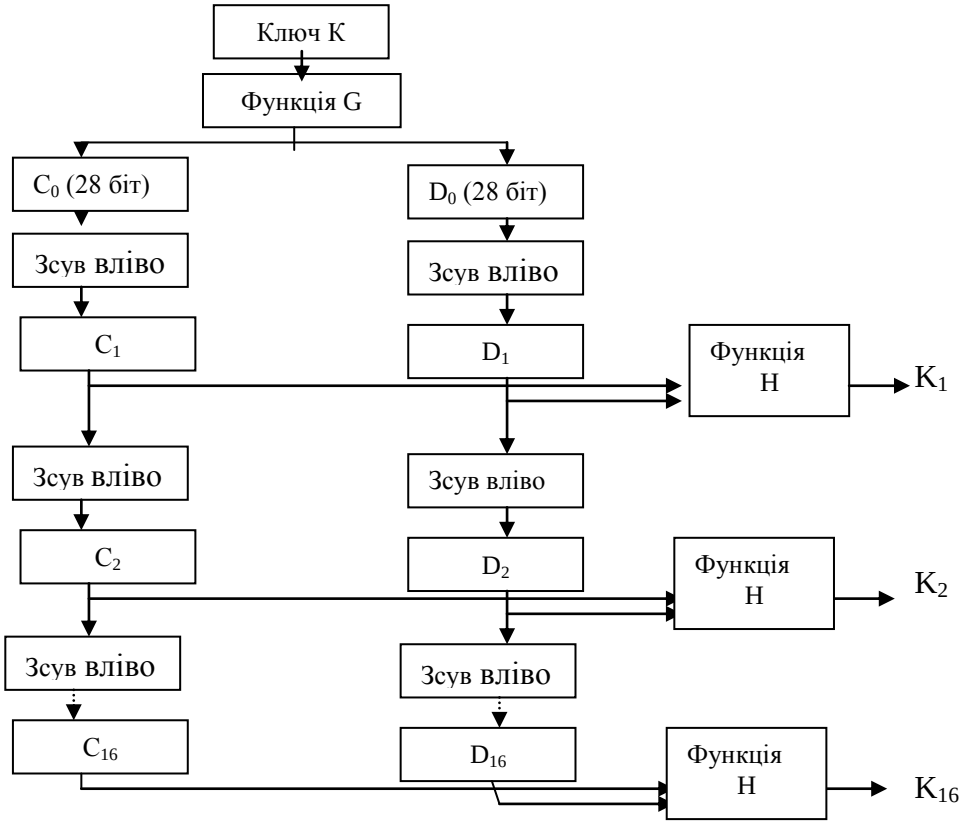

Рисунок 9.5 – Схема алгоритму обчислення ключів K_i

Таблица 9.7

Функция *G* початкової підготовки ключа (представлена виборка 1)

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

Табл. 7 розділена на дві частини. Результат перетворення *G(K)* розбивається на дві половини C_0 і D_0 по 28 біт кожна. Перші чотири рядки матриці *G* визначають, як вибираються біти послідовності C_0 (першим бітом C_0 буде біт 57 ключа шифру, потім біт 49 і т.д., а останніми бітами-биті 44 і 36 ключа). Наступні чотири рядки матриці *G* визначають, як вибираються біти послідовності D_0 (тобто послідовність D_0 буде складатися з бітів 63, 55, 47, 12, 4 ключі шифру). Як видно з табл. 7, для генерації послідовностей C_0 і D_0 не використовуються біти 8,16, 24, 32, 40, 48, 56 і 64 ключа шифру. Ці біти не впливають на шифрування і можуть служити для інших цілей (наприклад, для контролю парності). Таким чином, у дійсності ключ шифру є 56-бітовим.

Після визначення C_0 і D_0 рекурсивно визначаються C_i і D_i, i=1,2, ...,16. Для цього застосовуються операції циклічного зсуву вліво на один або два біти в залежності від номера кроку ітерації, як показано в табл. 9.8. Операції зсуву виконуються для послідовностей C_i і D_i незалежно. Наприклад, послідовність C_3 виходить за допомогою циклічного зсуву в ліво на дві позиції послідовності C_2, а послідовність D_3 за допомогою зсуву вліво на дві позиції послідовності D_2, C_{16} і D_{16} виходять із C_{15} і D_{15} за допомогою зсуву вліво на одну позицію.

Таблиця 9.8

Таблиця зсуву для обчислення ключа

Номер ітерації	Кількість зсувів вліво	Номер ітерації	Кількість зсувів вліво
1	1	9	1
2	1	10	2
3	2	11	2
4	2	12	2
5	2	13	2
6	2	14	2
7	2	15	2
8	2	16	1

Ключ K_i, обумовлений на кожнім кроці ітерації, є результат вибору конкретних бітів з 56-бітової послідовності C_iD_i і їхньої перестановки. Іншими словами, ключ $K_i=H(C_iD_i)$, де функція *H* визначається матрицею, що завершує обробку ключа (табл. 9.9). Як випливає з табл. 9, першим бітом ключа K_i, буде 14-й біт послідовності C_iD_i, другим-17-й біт, 47-м бітом ключа K_i, буде 29-й біт C_iD_i, а 48-м бітом – 32-й біт C_iD_i.

Таблиця 9.9

Функція *H* кінцевої обробки ключа (представлена виборка 2)

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

9.3. Основні режими роботи алгоритму DES

Алгоритм DES цілком підходить як для шифрування, так і для аутентифікації даних. Він дозволяє безпосередньо перетворювати 64-бітовий вхідний відкритий текст у 64-бітовий вихідний шифрований текст, однак дані рідко обмежуються 64 розрядами. Щоб скористатися алгоритмом DES для рішення різноманітних криптографічних задач, розроблені чотири робочі режими:

– електронна кодова книга ECB (Electronic Code Book);
– зчеплення блоків шифру CBC (Cipher Block Chaining);
– зворотний зв'язок за шифром CFB (Cipher Feed Back);
– зворотний зв'язок за виходом OFB (Output Feed Back).

9.3.1. Режим «Електронна кодова книга»

Довгий файл розбивають на 64-бітові відрізки (блоки) по 8 байтів. Кожний з цих блоків шифрують незалежно з використанням того самого ключа шифрування (рис.9.6).

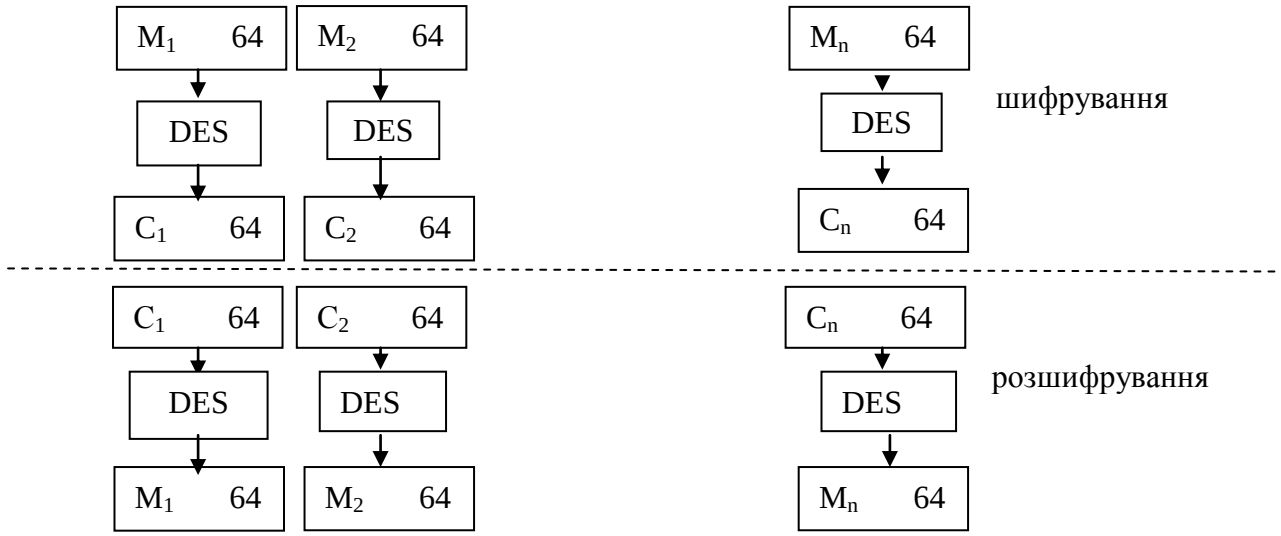

Рисунок 9.6 – Схема алгоритму DES в режимі електронної кодової книги

Основна властивість – простота реалізації. Відносно слабка стійкість проти кваліфікованих криптоаналітиків. Через фіксований характер шифрування при обмеженій довжині блоку 64 біти можливе проведення криптоаналізу «зі словником». Блок такого розміру може повторитися в повідомленні внаслідок великої надмірності в тексті природної мови. Це приводить до того, що ідентичні блоки відкритого тексту в повідомленні будуть представлені ідентичними блоки шифртексту, що дає криптоаналітику деяку інформацію про зміст повідомлення.

9.3.2. Режим «Зчеплення блоків шифру»

У цьому режимі вихідний файл M розбивається на 64-бітні блоки: $M=M_1M_2...M_n$. Перший блок M_1, додається за модулем 2 із 64-бітним початковим вектором *IV*, що міняється щодня і тримається в секреті (рис.7). Отримана сума потім шифрується з використанням ключа DES, відомого і відправникові, і одержувачеві інформації. Отриманий 64-бітовий шифр C_1 додається за модулем 2 із другим блоком

тексту, результат шифрується і виходить другий 64-бітовий шифр C_2, і т.д. Процедура повторюється доти, поки не будуть оброблені всі блоки тексту.

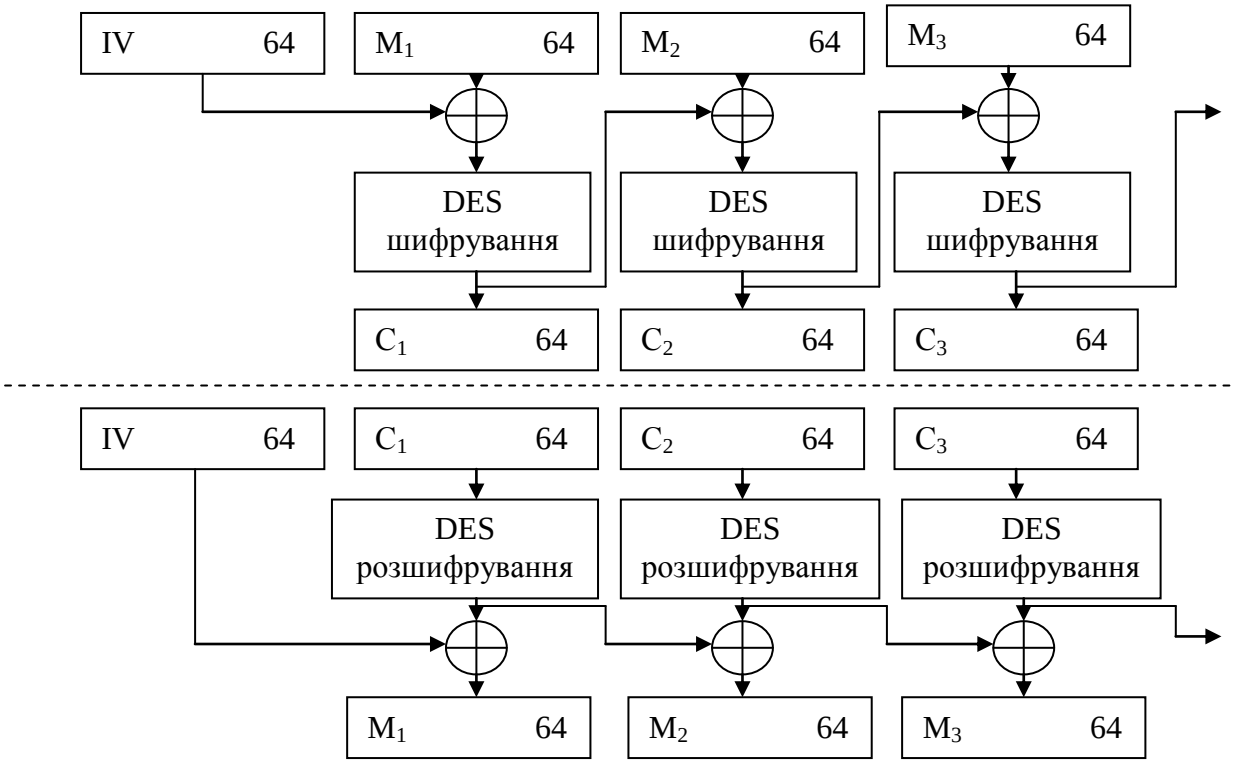

Рисунок 9. 7 – Схема алгоритму DES у режимі зчеплення блоків шифру

Таким чином, для всіх $i=1 \ldots n$ (n – число блоків) результат шифрування C_i та розшифрування M_i визначається в такий спосіб: $C_i = E_K(C_{i-1} \oplus M_i)$, $M_i = D_K(C_i) \oplus M_{i-1}$, де $C_0 = IV$ – початкове значення шифру, рівне початковому векторові (векторові ініціалізації). Очевидно, що останній 64-бітовий блок шифротексту є функцією секретного ключа, початкового вектора і кожного біта відкритого тексту незалежно від його довжини. Цей блок шифротекста називають кодом аутентифікації повідомлення (КАС). Код КАС може бути легко перевірений одержувачем, що володіє секретним ключем і початковим вектором, шляхом повторення процедури, виконаної відправником. Сторонній, однак, не може здійснити генерацію КАС, що сприйнявся б одержувачем як справжній, щоб додати його до помилкового повідомлення, або відокремити КАС від справжнього повідомлення для використання його зі зміненим або помилковим повідомленням. Перевага даного режиму в тім, що він не дозволяє накопичуватися помилкам при передачі. Блок M_i, є функцією тільки C_{i-1} і C_i, тому помилка при передачі приведе до втрати тільки двох блоків вихідного тексту.

9.3.3. Режим «Зворотного зв'язку за шифром», близький до CBC, перетворює блочний шифр на потоковий, що самосинхронізується. Дія дуже схожа; зокрема, розшифрування CFB майже тотожне до шифрування CBC виконаного навпаки: $C_i = E_K(C_{i-1}) \oplus M_i$, $M_i = D_K(C_{i-1}) \oplus C_i$, $C_0 = IV$.

Описаний вище найпростіший спосіб використання CFB ані трохи не більш самосинхронізовний ніж інші режими на кшталт CBC. Якщо губиться цілий блок шифротекту, то обидва CBC і CFB синхронізуються, тоді як втрата

одного байту або біту остаточно відкине розшифрування. Для уможливлення синхронізації після втрати одного байту або біту, за раз треба шифрувати один байт або біт. CFB можна таким чином використовувати разом з зсувним регістром на вході до блочного шифру. Подібно до режиму CBC, зміни в відкритому тексті поширюються на весь шифротекст, і шифрування не можна упаралелити. Також подібно до CBC, розшифрування упаралельнюване. При розшифруванні, зміна одного біту в шифротексті впливає на два блоки відкритого тексту: однобітова зміна в відповідному блоці відкритого тексту і повне пошкодження наступного блоку. Подальші блоки розшифровуються нормально. Через це, якщо зловмисник знає відкритий текст, він може змінити біти в блоці шифротексту як він захоче і призвести до отримання потрібного йому блоку відкритого тексту, наступний блок буде перетасовано, але на той час вже може бути заподіяна шкода. CFB разом з режимами потокових шифрів (OFB) поділяє дві переваги над CBC режимом: блок шифр використовується лише в напрямку шифрування, і повідомлення не потребує доповнення до розміру кратного розміру блоку (хоча також можна використати викрадення шифротексту і таким чином зробити доповнення необов'язковим).

9.3.4. Режим «Зворотний зв’язок за виходом»

Режим зворотного зв’язку за виходом утворює з блочного шифру синхронний потоковий шифр. Він утворює потік ключа, який потім додають за модулем 2 із блоками відкритого тексту для утворення шифротескту. Як і передбачає ім’я, OFB використовує попередній вихід блочного шифру, а не попередній шифротекст як CFB, що робить шифротекст незалежним від відкритого тексту і шифротексту, саме через це OFB є синхронним потоковим шифром. Так само як і з іншими потоковими шифрами, обернення біту в відкритому тексті призводить до обернення відповідного біту в шифротексті. Ця властивість дозволяє багатьом кодам виправлення помилок виконуватись нормально навіть при застосуванні до шифрування. Через симетричність операції XOR, шифрування і розшифрування однакові: $C_i = M_i \oplus O_i, M_i = C_i \oplus O_i, O_i = E_K(S_i), S_i = O_{i-1}, S_0 = IV$.

Для кожного нового повідомлення треба використовувати новий *IV*, інакше повідомлення будуть шифруватись за тим самим потоком ключа. *IV* можна відправляти у відкритому вигляді, однак, якщо зловмисник підмінив *IV*, тоді на відміну від CFB режиму, отримати відкритий текст з шифротексту неможливо. Кожен блок на виході залежить від усіх попередніх, отже неможливе паралельне виконання. Однак, через те, що відкритий текст і шифр використовуються лише в фінальному XOR, операції блочного шифру можна виконати наперед, що дозволить виконання фінального кроку паралельно щойно відкритий або шифротекст стане доступним. Можна отримати потік ключа для режиму OFB із рядком нулів на вході. Це може бути корисним, бо уможливлює використання швидкого апаратного забезпечення, що втілює CBC для шифрування в режимі OFB. На відміну від CFB, помилка в одному біті відкритого тексту зачіпає лише відповідний біт в шифротексті, наступні блоки залишаються неушкодженими. Ця властивість разом із можливістю високої швидкодії OFB робить його підхожим для шифрування потоків даних на

кшталт голосу і відео, особливо на каналах с завадами де поширення помилок може легко перетворити зашифроване передавання в майже неможливе. З іншого боку OFB вимагає від двох учасників бути синхронізованими. Як варіант розв'язання цієї проблеми пропонується відправляння синхронізаційних сигналів в узгоджені проміжки часу.

9.4. Області застосування алгоритму DES

Кожному з розглянутих режимів (ECB, CBC, CFB, OFB) властиві свої переваги і недоліки, що обумовлює області їхнього застосування. Режим ECB добре підходить для шифрування ключів: режим CFB, як правило, призначається для шифрування окремих символів, а режим OFB нерідко застосовується для шифрування в супутникових системах зв’язку.

Режими CBC і CFB придатні для аутентифікації даних. Ці режими дозволяють використовувати алгоритм DES для:

– інтерактивного шифрування при обміні даними між терміналом і головної ЕОМ;

– шифрування криптографічного ключа в практиці автоматизованого поширення ключів;

– шифрування файлів, поштових відправлень, даних супутників і інших практичних задач.

Спочатку стандарт DES призначався для шифрування і дешифрування даних ЕОМ. Однак його застосування було узагальнено і на аутентифікацію.

Звичайні коди, що виявляють помилки, непридатні, тому що якщо алгоритм утворення коду відомий, супротивник може виробити правильний код після внесення змін у дані. Однак за допомогою алгоритму DES можна утворити криптографічну контрольну суму, що може захистити як від випадкових, так і навмисних, але несанкціонованих змін даних. Цей процес описує стандарт для аутентифікації даних ЕОМ (FIPS 113). Суть стандарту полягає в тому, що дані зашифровуються в режимі зворотного зв’язку по шифротексту (режим CFB) або в режимі зчеплення блоків шифру (режим CBC), у результаті чого виходить остаточний блок шифру, що представляє собою функцію всіх розрядів відкритого тексту. Після цього повідомлення, що містить відкритий текст, може бути передано з використанням обчислювального кінцевого блоку шифру, що служить в якості криптографічної контрольної суми. Ті самі дані можна захистити, користуючись як шифруванням, так і аутентифікацією. Дані захищаються від ознайомлення шифруванням, а зміни виявляються за допомогою аутентифікації. Алгоритм аутентифікації можна застосувати як до відкритого, так і до зашифрованого тексту. При фінансових операціях, коли в більшості випадків реалізуються і шифрування, і аутентифікація, остання застосовується і до відкритого тексту.

Шифрування й аутентифікацію використовують для захисту даних, що зберігаються в ЕОМ. У багатьох ЕОМ паролі зашифровують необоротним образом і зберігають у пам’яті машини. Коли користувач звертається до ЕОМ і вводить пароль, останній зашифровується і порівнюється зі значенням, що

зберігається. Якщо обидві зашифровані величини однакові, користувач одержує доступ до машини, у противному випадку випливає відмовлення.

Нерідко зашифрований пароль виробляють за допомогою алгоритму DES, причому ключ покладається рівним паролеві, а відкритий текстові-кодові ідентифікації користувача. За допомогою алгоритму DES можна також зашифрувати файли ЕОМ для їхнього збереження. Одним з найбільш важливих застосувань алгоритму DES є захист повідомлень електронної системи платежів (ЭСП) при операціях із широкою клієнтурою і між банками. Алгоритм DES реалізується в банківських автоматах, терміналах у торговельних центрах, автоматизованих робочих місцях і головних ЕОМ. Діапазон даних, що захищаються ним, досить широкий – від оплат $50 до переводів на багато мільйонів доларів. Гнучкість основного алгоритму DES дозволяє використовувати його в найрізноманітніших областях застосування електронної системи платежів.

9.5. Стійкість DES

Обговорюючи DES, неможливо обминути тему безпеки цього алгоритму та можливих атак на нього. Багаторічний досвід експлуатації DES і його відкритість зумовили те, що DES став одним із найпопулярніших алгоритмів із погляду перевірки тих чи інших методів криптоаналізу. За весь час існування алгоритму на нього було здійснено багато атак; при цьому уважно вивчалися та враховувалися його слабкості, виявлені за довгий термін експлуатації. Треба мати на увазі, що деякі атаки можна реалізувати, лише виходячи з припущення, що зловмисник володіє певними обчислювальними (або часовими) ресурсами. У більшості випадків такі спроби мають лише теоретичний характер, однак не виключено, що з розвитком комп'ютерної техніки та криптології як науки ці атаки можна буде реалізувати на практиці.

До основних недоліків DES відносять такі:

– наявність «слабких» ключів (4 слабких та 12 напівслабких), викликана тим, що для генерування ключової послідовності виконується два незалежних регістри зсуву. Прикладом слабкого ключа може служити 1F1F1F1F0E0E0E0E. При цьому результатом генерування будуть ключові послідовності, однакові з вихідним ключем, в усіх 16 раундах. Існують також різновиди слабких ключів, що дають усього чотири ключові послідовності. Існують також «зв’язані» ключі, які отримуються один з одного інверсією одного біта;

– невелика довжина ключа в 56 бітів. При сучасному рівні розвитку комп'ютерних засобів ця довжина ключа не може забезпечувати потрібного захисту для деяких типів інформації;

– надмірність ключа, що має біти контролю парності. Біхам і Шамір запропонували досить ефективну атаку на реалізацію DES у смарт-картах або банківських криптографічних модулях, що використовують EEPROM-пам'ять для зберігання ключів. Наявність бітів контролю парності дозволяє відновити ключ при втраті частини ключа, викликаній втратою інформації в комірках пам'яті;

– використання статичних підстановок у *S* блоках, що, незважаючи на велику кількість раундів, дозволяє криптоаналітикам атакувати цей алгоритм.

Зауважимо, що авторам не відомі успішні атаки на 16-раундовий DES, які використовують останній факт, однак успішні атаки на 8-раундовий DES трапляються. Так, М. Хеллман запропонував атаку на основі робочої станції SUN-4, яка визначає 10 бітів ключа за 10 с. У випадку вибору 512 відкритих текстів імовірність успіху становить 80%, а при виборі 768 відкритих тактів - 95%. Відновивши 10 бітів ключа, решту можна знайти методом повного перебору наступних 46 бітів.

Отже, враховуючи вищезазначене, можна стверджувати, що сьогодні використання DES для критичної, з погляду секретності, інформації є досить небезпечною справою. Криптологи давно намагалися модифікувати DES із метою збільшення його криптостійкості. Розглянемо такі модифікації, як 3DES, DESX і S-DES.

9.6. Модифікації DES

Потрійний DES

Найвідомішою модифікацією DES є потрійний DES (3DES), один з варіантів якого визначається формулою:

$C = 3DES_{K1K2K3}(M) = DES_{K3}(DES_{K2}^{-1}(DES_{K1}(M)))$.

Ум цій схемі використовуються три ключі *K1, K2, K3,* сумарна довжина яких складає $56 \cdot 3 = 168$ бітів. 64-бітний блок відкритого повідомлення *M* спочатку шифрується на ключі *K1* потім розшифровується на ключі *K2* і знову зашифровується на *K3*. Причиною того, що на другому кроці використовується DES_{K2}^{-1}, а не DES_{K2}, є сумісністю з DES. Дійсно, якщо $K = K1 = K2 = K3$, то $3DES_K = DES_K$. Причиною того, чому обрано саме три ітерації, а не дві, є існування атаки «зустріч посередині» на подвійний DES.

Розшифрування інформації в 3DES використовується обернено: $M = DES_{K3}^{-1}(DES_{K2}(DES_{K1}^{-1}(C)))$.

Отже, кожен блок повідомлення *M* обробляється DES-машиною тричі на різних ключах, що, звичайно, поліпшує криптостійкість. Однак проблемою 3DES є його повільність – швидкість його втричі менша за DES. У багатьох випадках це неприпустимо, особливо для шифрування каналів зв'язку.

DESX

У 1984 р. Рон Рівест запропонував інший варіант модифікації DES, позбавлений цього недоліку 3DES.

DESX (DES extended – розширений DES) можна визначити так: $C = DESX_{KK1K2} = K2 \oplus DES_K(K1 \oplus M)$.

Отже, 64-бітний блок повідомлення *M* підсумовується за *mod 2* з першим «зашумлюючим» ключем *K1* потім обробляється DES-машиною з ключем *K,* після чого підсумовується за *mod 2* з другим «зашумлюючим» ключем *K2*. Таким чином, DESX має усього на дві операції XOR більше за оригінальний DES-алгоритм, що не вимагає значних витрат часу. Водночас наявність цих двох

операцій XOR значно поліпшує стійкість до повного перебирання ключів (загальна довжина ключа дорівнює $K+K1+K_2$=56+64+64=184 біти), а також робить його стійкішим до диференційного та лінійного криптоаналізів, збільшуючи необхідну кількість спроб із вибраним текстом до 2^{60}.

S-DES

Спрощений DES – це алгоритм шифрування, який має, скоріше, навчальне, ніж практичне значення. За своїми властивостями він подібний до DES, але має значно менше параметрів. Цей алгоритм приймає на вході 8-бітний блок відкритого тексту та 10-бітний ключ, а на виході генерує 8-бітний блок шифрованого тексту. При розшифруванні на вхід алгоритму подається 8-бітний блок шифротексту і 10-бітний ключ, а на виході генерує 8-бітний блок відкритого тексту. Алгоритм шифрування передбачає послідовне виконання п’яти операцій: початкової перестановки *IP;* раундової функції, що складається з перестановок і підстановок; перестановки *SW,* коли дві половинки блока по 4 біти переставляються місцями; ще одного застосування раундової функції; і, нарешті, перестановки IP^{-1} оберненої до початкової. Послідовне використання кількох перестановок і підстановок значно ускладнюють криптоаналіз. Раундова функція приймає на вході не лише блок тексту, а й 8-бітний цикловий підключ, який утворюється з 10-бітного ключа. Блок-схему алгоритму подано на рис. 8. З цього рисунка видно, що, оскільки це симетричний криптоалгоритм, він використовує для шифрування та розшифрування той самий ключ. Тому ключ має бути як на передавальній, так і на приймальній стороні. З цього ключа на певних етапах шифрування та розшифрування генеруються два 8-бітних раундових підключів.

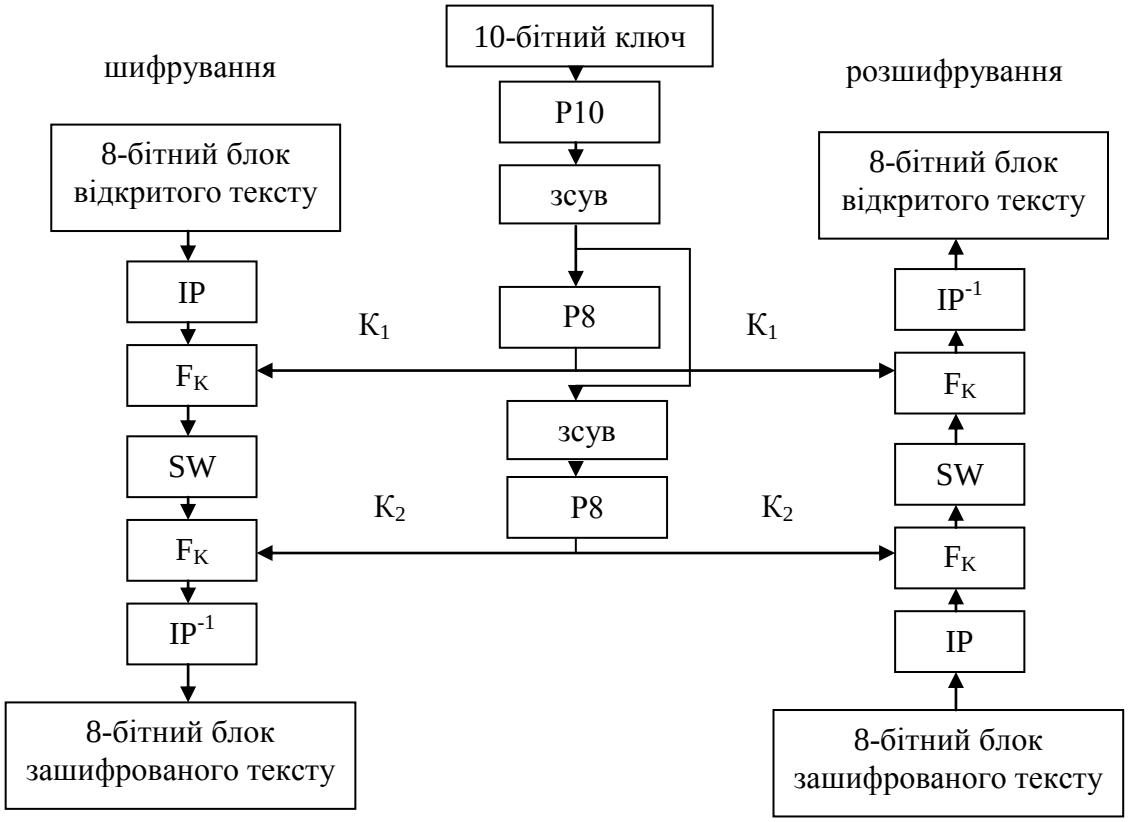

Рисунок 9.8 – Схема спрощеного алгоритму S-DES

Процедура генерування раундових підключів

1. Спочатку біти ключа переставляються так. Якщо 10-бітний ключ подати у вигляді $k_1, k_2 \ldots, k_{10}$, то перестановка *P10* задається у вигляді таблиці

Перестановка P10									
3	5	2	7	4	10	1	9	8	6

Ця таблиця символізує позицію біта вхідних даних у вихідній послідовності: першим стає 3-й біт, другим – 5-й, третім – 2-й і т.д. Наприклад, ключ (1010000010) відповідно до цієї перестановки перетворюється в послідовність (1000001100).

2. Ключ розділяється на дві 5-бітні половини. Окремо перша половина й окремо друга піддаються циклічному зсуву ліворуч на одну позицію. У нашому прикладі в результаті буде отримана послідовність (00001 11000).

3. Отримана послідовність піддається перестановці P8, у результаті якої з 10-бітного ключа обирається 8-бітна послідовність за таким правилом

Перестановка P8							
6	3	7	4	8	5	10	9

У результаті цієї операції ми отримаємо перший раундовий підключ (K_1). У нашому прикладі він буде мати вигляд (10100100).

4. Для генерування другого раундового підключа K_2 необхідно повернутися на крок назад, до двох 5-бітних рядків до застосування *P8* та виконати для кожного з цих рядків циклічний зсув ліворуч на дві позиції. У нашому прикладі значення підключів (00001 11000) перетворяться у (00100 00011).

5. Нарешті, застосувавши до цієї послідовності перестановку *P8*, отримаємо другий раундовий підключ K_2 Для нашого прикладу результатом буде (01000011).

Шифрування S-DES

Початкова перестановка IP							
2	6	3	1	4	8	5	7

На завершальній стадії алгоритму виконується обернена перестановка $IP^{1.}$

Кінцева перестановка IP^{-1}							
4	1	3	5	7	2	8	6

1. Початкова й кінцева перестановки *(IP* та IP^{-1}*)*. На вхід алгоритму подається 8-бітний блок відкритого тексту, до якого застосовується початкова перестановка *IP*. Можна пересвідчитися, що ці дві таблиці дійсно обернені одна до одної, тобто $IP^{-1}(IP(M))=M$.

2. Раундова функція *F*. Розіб'ємо вхідний блок тексту після *IP*-перестановки на два 4-бітні півблоки. Лівий 4-бітний блок позначимо *L*, а правий – *R*. Тоді циклову функцію можна записати у вигляді формули $F_K(L, R)=(L \oplus F(R,K_i), R)$, де K_i означає цикловий підключ.

На вході циклова функція отримує 4-бітне значення (n_1,n_2,n_3,n_4), тобто праву половину вхідного блока. Перша операція –операція розширення та перестановки, її можна також зобразити так (4,1,2,3,2,3,4,1).

Зручніше цю операцію зобразити у вигляді матриці: $\begin{pmatrix} n_4 & n_1 & n_2 & n_3 \\ n_2 & n_3 & n_4 & n_1 \end{pmatrix}$.

До цього значення додається 8-бітний підключ за допомогою операції XOR. Це можна зобразити так: $\begin{pmatrix} n_4 + k_1 & n_1 + k_2 & n_2 + k_3 & n_3 + k_4 \\ n_2 + k_5 & n_3 + k_6 & n_4 + k_7 & n_1 + k_8 \end{pmatrix}$.

Перейменуємо отримані елементи: $\begin{pmatrix} p_{00} & p_{01} & p_{02} & p_{03} \\ p_{10} & p_{11} & p_{12} & p_{13} \end{pmatrix}$.

Перші чотири біти (тобто перший рядок цієї матриці) далі подаються на вхід модуля заміни (*S*-матриці) S_0 , на виході якого отримується 2-бітна послідовність. Другий рядок матриці подається на вхід другого модуля заміни S_1, на виході якого також отримується 2-бітна послідовність. Модулі S_0 і S_1, задаються так:

$$S_0 = \begin{pmatrix} 1 & 0 & 3 & 2 \\ 3 & 2 & 1 & 0 \\ 0 & 2 & 1 & 3 \\ 3 & 1 & 3 & 1 \end{pmatrix} \quad S_1 = \begin{pmatrix} 1 & 1 & 2 & 3 \\ 2 & 0 & 1 & 3 \\ 3 & 0 & 1 & 0 \\ 2 & 1 & 0 & 3 \end{pmatrix}.$$

Рядки та стовпчики нумеруються, починаючи з нуля.

Ці модулі заміни працюють так. Перший і четвертий біти вхідної послідовності вважаються двійковим представленням номера рядка, а другий і третій - номерами стовпця. Елемент, що знаходиться на перетині цих рядка і стовпця, задає двобітне вихідне значення. Наприклад, якщо (p_{00},p_{03})=(00) та (p_{01},p_{02})=(10), то вихідні два біти задаються значенням, яке знаходиться на перетині 0-го рядка та 3-го стовпців, тобто це буде число 3, а у двійковому представленні – 11.

Аналогічну операцію виконують і з другим рядком.

Після застосування матриць заміни результат піддають перестановці *P4* за таким законом: (2, 4, 3, 1). Результат перестановки *P4* і буде результатом функції F_K. Отримана послідовність бітів додається за модулем 2 з лівою половиною *L* вхідного блока і буде новою лівою половиною. Права половина передається на вихід циклу без змін.

3. Перестановка підблоків. Як бачимо, за один раунд раундовою функцією обробляється лише ліва половина відкритого тексту, права половина залишається без змін. Для того, щоби зашифрувати й праву половину, використовується другий цикл, однак на його вхід треба подати переставлені підблоки: *L* і *R* поміняти місцями. Для цього й служить функція *SW*- перемикач блоків.

Після переставляння підблоків один раунд алгоритму закінчено.

До переставлених підблоків знову застосовується циклова функція, як це описано вище. При другому викликові раундової функції розширення з перестановкою модулі S_0 і S_1 , P_4 залишаються тими ж, тільки використовується підключ K_2. По закінченні другого раунду виконується IP^{-1}-перестановка, й роботу алгоритму закінчено, тобто на виході маємо зашифрований текст.

Розшифрування зашифрованого тексту

Як видно з рис.9. 8, розшифрування зашифрованого тексту виконується аналогічно шифруванню, за винятком того, що ключі подаються у зворотному порядку.

9.7. Алгоритм шифрування IDEA

Алгоритм IDEA (International Data Encryption Algorithm) є блоковим шифром. Він оперує 64-бітовими блоками відкритого тексту. Безсумнівною перевагою алгоритму IDEA є те, що його ключ має довжину 128 біт. Той самий алгоритм використовується і для шифрування, і для розшифрування.

Перша версія алгоритму IDEA була запропонована в 1990 р., її автори-Х. Лей і Дж. Мэсси. Первісна назва алгоритму PES (Proposed Encryption Standard). Поліпшений варіант цього алгоритму, розроблений у 1991р., одержав назву IPES (Improved Proposed Encryption Standard). У 1992 р. IPES змінив своє ім'я на IDEA. Як і більшість інших блокових шифрів, алгоритм IDEA використовує при шифруванні процеси змішування і розсіювання, причому всі процеси легко реалізуються апаратними і програмними засобами [10].

В алгоритмі IDEA використовуються наступні математичні операції:

$\oplus$ – порозрядне додавання за модулем 2.

$\boxplus$ – додавання беззнакових цілих за модулем 2^{16} (модуль 65536);

$\odot$– множення цілих за модулем (2^{16} +1) (модуль 65537), розглянутих як беззнакові цілі, за винятком того, що блок з 16 нулів розглядається як 2^{16}.

Всі операції виконуються над 16-бітовими субблоками. Комбінування цих трьох операцій забезпечує комплексне перетворення входу, істотно утрудняючи криптоаналіз IDEA у порівнянні з DES, що базується винятково на операції $\oplus$.

Загальна схема алгоритму IDEA приведена на рис.9.

64-бітовий блок даних поділяється на чотири 16-бітових субблоки. Ці чотири субблоки стають входом у перший цикл алгоритму. Усього виконується вісім циклів. Між циклами другий і третій субблоки міняються місцями. У кожнім циклі має місце наступна послідовність операцій:

1) $\odot$ – множення субблока X_1 і першого підключа.
2) $\boxplus$ – додавання субблока X_2 і другого підключа.
3) $\boxplus$ – додавання субблока X_3 і третього підключа.
4) $\odot$ – множення субблока X_4 і четвертого підключа.
5) $\oplus$ – додавання результатів кроків (1) і (3).
6) $\oplus$ – додавання результатів кроків (2) і (4).
7) $\odot$ – множення результату кроку (5) і п'ятого підключа.
8) $\boxplus$ – додавання результатів кроків (6) і (7).
9) $\odot$ – множення результату кроку (8) із шостим підключем.
10) $\boxplus$ – додавання результатів кроків (7) і (9).
11) $\oplus$ – додавання результатів кроків (1) і (9).
12) $\oplus$ – додавання результатів кроків (3) і (9).
13) $\oplus$ – додавання результатів кроків (2) і (10).
14) $\oplus$ – додавання результатів кроків (4) і (10).

Виходом циклу є чотири субблока, що одержують як результати виконання кроків (11), (12), (13) і (14). У завершення циклу переставляють місцями два

внутрішніх субблока (за винятком останнього циклу), і в результаті формується вхід для наступного циклу.

Після восьмого циклу здійснюють заключне перетворення виходу:

1) ⊙ - множення субблока X_1 і першого підключа.
2) ⊞ - додавання субблока X_2 і другого підключа.
3) ⊞ - додавання субблока X_3 і третього підключа.
4) ⊙ - множення субблока X_4 і четвертого підключа.

Нарешті, ці результуючі чотири субблока $Y_1...Y_4$ знову поєднують для одержання блоку шифртекста.

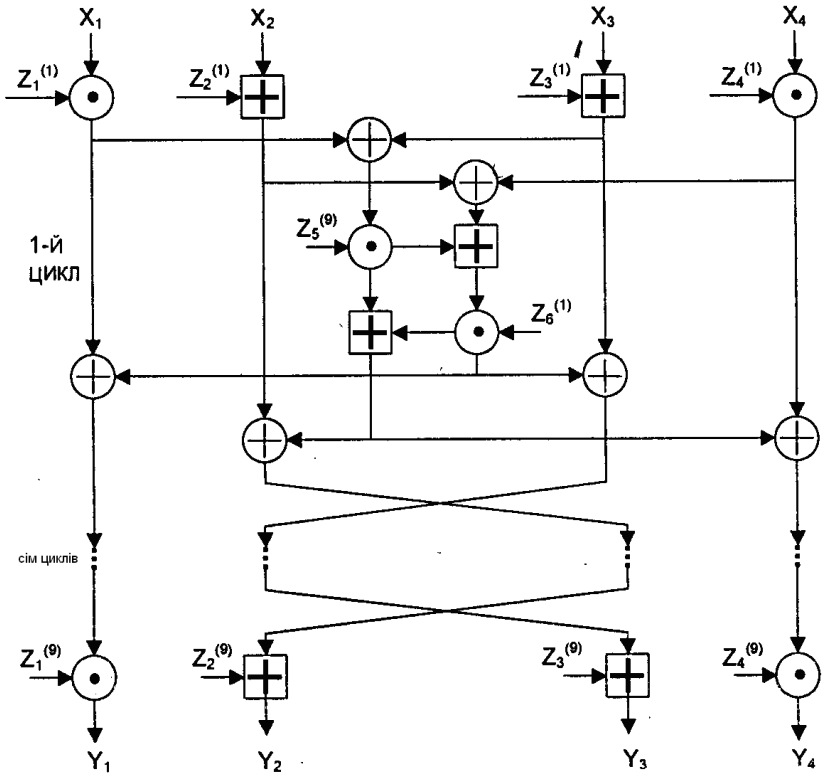

Рисунок 9.9 – Схема алгоритму IDEA (режим шифрування)

Створення підключів Z_i також відносно нескладно. Алгоритм використовує всього 52=6*8+4 підключі (по шість для кожного з восьми циклів і ще чотири для перетворення виходу). Спочатку 128-бітовий ключ поділяють на вісім 16-бітових підключів. Це перші вісім підключів для алгоритму (шість підключів-для першого циклу і перші два для другого циклу). Потім 128-бітовий ключ циклічно зсувається вліво на 25 біт і знову ділиться на вісім підключів. Перші чотири з них використовують в другому циклі; останні в третьому циклі. Ключ знову циклічно зсувають вліво ще на 25 біт для одержання наступних восьми підключів і т.д., поки виконання алгоритму не завершиться.

Розшифрування здійснюють аналогічним способом, за винятком того, що порядок використання підключів стає зворотним.

Алгоритм IDEA може працювати в будь-якому режимі блокового шифру, передбаченому для алгоритму DES. Алгоритм IDEA володіє багатьма перевагами перед алгоритмом DES. Він значно безпечніше алгоритму DES, оскільки 128-бітний ключ алгоритму IDEA удвічі більше ключа DES.

Внутрішня структура алгоритму IDEA забезпечує кращу стійкість до криптоаналізу. Існуючі програмні реалізації алгоритму IDEA приблизно удвічі швидше реалізацій алгоритму DES.

9.8. ГОСТ 28147-89

Стандарти шифрування Радянського Союзу було прийнято в 1989 році. В основі лежить DES – подібний алгоритм симетричного шифрування. Враховані недоліки DES. Має також відкрито опублікований алгоритм шифрування. На сьогодні залишається одним з найнадійніших алгоритмів шифрування, призначений для апаратної та програмної реалізації. Стандарт обов'язковий для організацій, підприємств і установ, що застосовують криптографічний захист даних, збережених і переданих у межах КС, в окремих обчислювальних комплексах.

ГОСТ є 64 бітним алгоритмом із 256-бітним ключем. У процесі роботи алгоритму на 32 етапах послідовно виконується простий алгоритм шифрування. При описі алгоритму використовуються наступні позначення: L і R – послідовність бітів (по 32); $A \oplus B$ – додавання за модулем 2; $A[+]B$ – додавання за модулем 2^{32} ($a[+]b=a+b$, $a+b<2^{32}$, $a[+]b=a+b-2^{32}$, $a+b>2^{32}$); $A\{+\}B$ – додавання за модулем 2^{32}-1, аналогічно як вказано вище [9; 10].

Для різних вимог до шифротексту і різних довжин текстів розроблений ряд режимів шифрування. Алгоритм передбачає чотири режими роботи:

- шифрування у режимі електронної кодової книги (ECB);
- шифрування у режимі гамування;
- шифрування у режимі гамування зі зворотним зв'язком;
- вироблення імітовставки.

Режим електронної кодової книги

Для шифрування текст (64 біти) розбивається на ліву L (32 біти) і праву R (32 біти) частини. Запишемо блок-схему 1-го етапу роботи алгоритму в режимі електронної кодової книги (ECB).

На i-ому етапі: $\begin{cases} L_i = R_{i-1} \\ R_i = L_{i+1} \oplus f(R_{i-1}, K_i) \end{cases}$. На 32-ому етапі: $\begin{cases} L_{32} = L_{31} \oplus f(R_{31}, K_1) \\ R_{32} = R_{31} \end{cases}$.

Функція f містить операції: [+], S, зсув вліво. S-блоки є секретними – додаткові ключі. Стандарт ГОСТу визначає не спосіб генерації S-блоків, а лише спосіб їх зображення. Виробник створює перестановки S-блоку самостійно за допомогою генератора псевдовипадкових послідовностей. S-блоки повинні час від часу змінюватися.

256 бітний ключ розбивається на 8·32 бітні підключі: *k1...k8*. На кожному етапі використовується свій ключ: k_1 k_2 k_3 k_4 k_5 k_6 k_7 k_8 k_1 k_2 k_3 k_4 k_5 k_6 k_7 k_8 k_1 k_2 k_3 k_4 k_5 k_6 k_7 k_8 k_8 k_7 k_6 k_5 k_4 k_3 k_2 k_1. Для розшифрування використовуються ключі у зворотньому порядку. Розшифрування реалізується аналогічно, в зворотньому порядку.

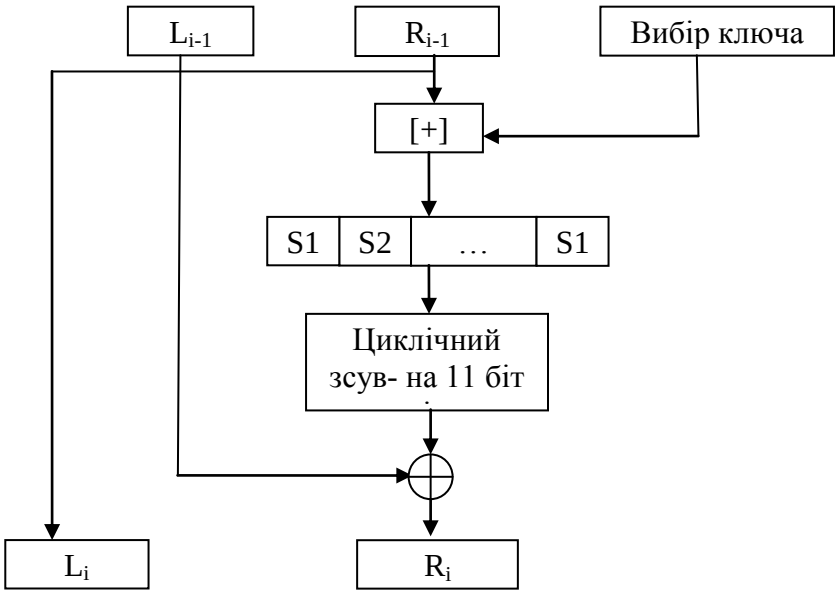

Рисунок 9.10 – Схема алгоритму ГОСТ 28147-89 (режим ECB)

Режим гамування

Відкриті блоки даних розбиваються на 64 біти (M_i), які шифруються шляхом порозрядного додавання за модулем 2 з гамою шифру Γ_i, яка вибирається блоками M_i по 64 біти: $C_i = M_i \oplus \tilde{A}_i = ECB(Y_{i-1} \oplus C2, Z_{i-1}\{+\}C1) \oplus M_i$.

Y_i, Z_i – визначаються ітераційно слідуючим чином:

$(Y_0, Z_0) = (A(S), S)$, $(Y_i, Z_i) = (Y_{i-1}[+]C2, Z_{i-1}\{+\}C1)$, $i = \overline{1,m}$.

A – функція шифрування в режимі простої заміни, S – 64-розрядний двійковий вектор (послідовність), що називається синхронопосиланням, не є секретним елементом шифру. *C1, C2* – 32-розрядні константи, які є стандартом ГОСТу.

Режим гамування з оберненим зв'язком

$C_1 = A(S) \oplus M_1 = \tilde{A}_1 \oplus M_1$, $C_i = A(\tilde{A}_{i-1}) \oplus M_i$ $\quad i = \overline{2,m}$.

Режим вироблення імітовставки

Справжність всіх прийнятих шифрованих повідомлень у всіх режимах додатково підтверджується за допомогою протоколу імітовставки.

Імітовставка – це блок із p біт, який утворюється перед шифруванням всього повідомлення або паралельно із шифруванням окремих блоків. Параметр p вибирається у відповідності із рівнем захищеності. Для отримання імітовставки відкриті дані представлені у виді блоків по 64 біти. Перший блок відкритих даних M_1 піддається перетворенню, що відповідає першим 16-ти циклам режиму ECB, причому в якості ключа використовується той же ключ, що і для шифрування. Отримане число сумується з відкритим блоком M_2, і сума знову піддається 16-циклам шифрування в режимі ECB. Дана процедура використовується для всіх блоків відкритого тексту. Із 64 біт отриманого числа вибирається p біт. Імітовставка передається по каналу зв'язку після зашифрованих даних. На іншій стороні із прийнятого повідомлення виробляється імітовставка і порівнюється із отриманою.

Таблиця 9.10

Порівняння криптостійкості DES і ГОСТ-89

DES використовує складну процедуру для генерації підключів і ключів	у ГОСТ процедура дуже проста
у DES – 56 бітний ключ	256 бітний ключ+ S-секретні блоки
у DES для S-блоків 6 біт вхід →4 біт вихід	4 біт вхід → 4 біт вихід менший, розмір S-блока
Використовуються нерегулярні перестановки – P-блоки	11 бітний циклічний зсув
16 етапів	32 етапи

ГОСТ складний для лінійного та диференціального криптоаналізу, із-за довжини ключа та кількості етапів. Інші частини ГОСТ слабкіші за DES. *S*-блоки слабші, але секретні. Нема перестановки із розширенням. Розробники ГОСТу намагалися досягти рівноваги між безпекою і ефективністю. Вони змінили ідеологію DES так, щоб створити алгоритм, який більше підходить для програмної реалізації. Менш впевнені в безпеці алгоритму, компенсували довжиною ключа, *S*-блоками та кількістю ітерацій. Наведемо порівняння стійкості DES і ГОСТу до повного перебору ключів: Нехай зловмисник володіє обчислювальними потужностями, здатними використати $W=10^9$ спроб ключів за одну секунду, тоді ймовірність успішної атаки $P_k(T) \approx \frac{T \cdot W}{k}$, де T – час, витрачений на підбір ключів C; W – кількість спроб за 1 секунду; k – потужність простору ключів [9].

Таблиця 9.11

Порівняння стійкості

Час	DES	ГОСТ
1 рік	$P_k(1) = \frac{3 \cdot 10_c^7 \cdot 10^9}{2^{56}} \approx 0{,}44$	$P_k(1) = \frac{3 \cdot 10_c^7 \cdot 10^9}{2^{256}} = 2{,}72 \cdot 10^{-61}$
2 роки	0,88	$5{,}4 \cdot 10^{-61}$
10 років	1	$2{,}72 \cdot 10^{-60}$

9.9. AES

У кінці 1996 року Національним інститутом стандартів США було оголошено конкурс на створення нового загальнонаціонального стандарту, який повинен замінити DES. Розробленому стандарту присвоїли робочу назву AES (Advanced Encryption Standard). 2 жовтня 2000 року прийнято рішення, і як запропонований варіант стандарту – алгоритм Rijndael (Вінсент Рійман і Йоан Даймен). Rijndael використовує кінцеву групу точок еліптичної кривої. Для розуміння даного алгоритму необхідні знання із теорії чисел [9].

В алгоритмі AES використовується поле Галуа $GF(2^8)$, як розширення GF(2), побудованого за многочленом $m(x) = x^8 + x^4 + x^3 + x + 1$. Цей многочлен вибраний з міркувань ефективності представлення елементів поля. Цей

алгоритм є блоковим шифром зі змінною довжиною блоку і змінною довжиною ключа 128, 192, 256.

Вхідні M подаються у виді прямокутної матриці байтів $4\times n$, де $n=4,6,8$ в залежності до розміру блока. Наприклад, 128 біт=матриця байтів 4×4 (16×8біт=128біт). Раундова функція $F(x_i)$ складається з чотирьох етапів.

Побайтова заміна (BS): кожен байт замінюється на обернений до в нього в полі GF (2^8); над кожним байтом здійснюється афінне перетворення в полі GF (2), задане рівнянням:

$$\begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \\ y_7 \end{bmatrix} = \begin{bmatrix} 10001111 \\ 11000111 \\ 11100011 \\ 11110001 \\ 11111000 \\ 01111100 \\ 00111110 \\ 00011111 \end{bmatrix} \cdot \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}$$

SR – зсув рядків: 1-й не зсувається, 2-й на зсувається на 1 байт, 3-й – на 2 байти, 4-й – на 3 байти.

MC – операція над стовпчиками. Кожен стовпчик множимо за правилом поля Галуа на точках еліптичної кривої на фіксовану матрицю:

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

AK (Add Round Key) – додавання раундового ключа. Просте побітне додавання кожного байта масиву і відповідного байта ключа.

Ключі ітерацій отримуються із ключа шифру за допомогою алгоритму обробки ключа, а саме розширення і вибору. Як стандарт Rijndael криптологи пропонують використовувати 128 бітний блок тексту. Якщо ключ брати 128 біт, то використовувати 10 раундів, для 192 біт ключа – 12 раундів, для 256 біт ключа – 14 раундів.

Криптостійкість Rijndael

Дослідження показали, що криптоаналіз 16-раундового DES у принципі реальний, однак вимагає великої кількості вихідних даних. Криптоаналіз при 20-24 циклах стає навіть теоретично неможливим.Алгоритм ГОСТу має 32 раунди. Як ми бачимо, його криптостійкість має значний запас і сьогодні. За оцінками розробників Rijndael, вже на чотирьох раундах шифрування цей алгоритм досить стійкий для сучасних використань. Теоретичною межею вважається 6-8 раундів. Отже, 10-14 раундів, передбачених в цьому алгоритмі, мають значний запас щодо криптостійкості.

До недоліків алгоритму можна віднести нетрадиційність, недостатньо вивчену схему, яка може містити приховані вразливості, невідомі на час конструювання алгоритму.

9.10. Стандарти шифрування України

На сьогоднішній день в Україні використовуються такі алгоритми симетричного блокового шифрування:

- **ДСТУ ГОСТ 28147:2009** Система обробки інформації. Захист криптографічний. Алгоритм криптографічного перетворення (ГОСТ 28147-89). У 2009 році ГОСТ 28147-89 перевиданий в Україні під назвою ДСТУ ГОСТ 28147:2009,
- **AES** (у складі операційних систем загального призначення),
- **RC4** (іноземні реалізації засобів захисту веб-з'єднань відповідно до протоколів SSL/TLS),
- **Triple DES** (Національний банк України, іноземні реалізації засобів захисту мережевого трафіка IPsec),
- **ДСТУ 7624:2014** («Калина»).

ДСТУ ГОСТ 28147:2009 використовується для здійснення швидкісного симетричного шифрування в Україні. За показниками криптостійкості він відповідає задовільному рівню. Крім того, даний стандарт шифрування поступається перспективним шифрам за швидкодією вдвічі або більше. ДСТУ ГОСТ 28147:2009 використовується лише на регіональному рівні. Більшість держав практично відмовилися від його застосування.

Перевагами стандарту ДСТУ ГОСТ 28147:2009 є: загальна відомість шифру, оскільки він був добре досліджений міжнародною спільнотою протягом більше ніж 20 років; прийнятний рівень швидкодії (32-бітові платформи), достатньо зручний для апаратної реалізації, зокрема для малоресурсної (lightweight) криптографії; вузли заміни (S-блоки) з хорошими властивостями забезпечують практичну стійкість шифру.

Недоліками ДСТУ ГОСТ 28147:2009 є: наявність теоретичних атак зі складністю, значно меншою повного перебору ключів; великі класи слабких ключів; використання вузлів заміни спеціального виду дозволяє зменшити рівень стійкості до реалізації практичних атак (виключно на основі шифр-текстів) із використанням одного персонального комп'ютера; швидкодія на сучасних системах суттєво нижча порівняно з іншими блоковими шифрами.

AES (Advanced Encryption Standard, також відомий під назвою Rijndael) – це симетричний алгоритм блочного шифрування, який був прийнятий урядом США як національний стандарт шифрування, оскільки став фіналістом конкурсу. Розмір блоку даних шифрування становить 128 біт, а розмір ключа – 128/192/256 біт. Підтримка AES введена фірмою Intel у сімейство процесорів x86, починаючи з Intel Core i7-980X Extreme Edition, а потім на процесорах Sandy Bridge.

Переваги AES: найбільш досліджений у світі криптографічний алгоритм, висвітлений у відкритих публікаціях; забезпечує високу практичну стійкість, включений до набору Suite-B Агентства національної безпеки США, дозволений для захисту інформації з обмеженим доступом уряду США; ефективний для реалізації на 32-бітових платформах; наявність низки

апаратних акселераторів (включаючи старші моделі процесорів загального призначення.

Недоліки алгоритму шифрування AES: відомі теоретичні атаки зі складністю, меншою, ніж повний перебір; не може в повній мірі використати можливості 64-бітових платформ; відносна застарілість (розроблений у 1997 році, у конкурсі NIST SHA-3 перевага віддана рішенням із архітектурою, що значно відрізняється від AES); відсутність довіри до іноземних апаратних реалізацій AES (у тому числі набору інструкцій AES-NI процесорів x86 і x86_64) на основі даних Е. Сноудена. Світові лідери IT-індустрії почали поступово відмовляться від AES, наприклад, компанія Google у 2014 році впровадила на заміну алгоритм ChaCha20 для захисту каналів зв'язку мобільних пристроїв на базі операційної системи Android .

RC4 (Rivest Cipher 4 або Ron's Code) – потоковий шифр, який широко застосовується в комп'ютерних мережах (наприклад, у протоколах SSL і TLS, алгоритмах забезпечення безпеки бездротових мереж WEP і WPA). Також RC4 використовується в різних системах захисту інформації. Цей алгоритм шифрування даних був розроблений компанією RSA Security. Для його використання необхідно мати ліцензію. Основними перевагами алгоритму шифрування RC4 є: висока швидкість роботи, змінний розмір ключа.

RC4 досить уразливий, якщо: використовуються не випадкові або пов'язані ключі, один ключовий потік застосовується двічі. Ці фактори, а також спосіб використання можуть зробити криптосистему небезпечною (наприклад, WEP). Алгоритм шифрування даних RC4 використовується в таких криптосистемах та протоколах: WEP, BitTorrent protocol encryption, Microsoft point-to-point encryption, браузер Opera Mini, протокол SSL (варіативно), протокол SSH (варіативно), протокол RDP, Kerberos (варіативно), SASL mechanism digest-MD5 (варіативно), формат PDF, Skype (in modified form). Зараз не рекомендується використовувати даний алгоритм шифрування, оскільки було знайдено методи успішної атаки на нього. Тому підтримка RC4 поступово видаляється з різних криптосистем.

Triple DES (3DES) – симетричний блоковий шифр, створений у 1978 році Уітфілдом Діффі, Мартіном Хеллманом і Уолтом Тачманом на основі алгоритму шифрування даних DES. Головною метою розробки було усунення основного недоліку алгоритму шифрування DES, а саме малої довжини ключа в 56 біт, що може бути зламаний методом повного перебору. 3DES поступається за швидкістю DES утричі, але за криптостійкістю він набагато кращий. Час, потрібний для криптоаналізу 3DES, може бути в мільярд разів більшим, ніж необхідний для аналізу DES. Саме тому він використовується частіше, ніж алгоритм шифрування DES. Останній можна легко зламати за допомогою сучасних комп'ютерних технологій. У 1998 році організація Electronic Frontier Foundation, використовуючи спеціальний комп'ютер DES Cracker, розкрила DES за 3 дні.

Симетричний алгоритм шифрування даних 3DES реалізований у багатьох програмних додатках, орієнтованих на роботу з Інтернетом, у тому числі в PGP і S/mime. Triple DES є досить достойною і популярною альтернативою шифру

DES. Потрійний DES використовується в стандартах ISO 8732, ANSI X9.17, а також у Privacy Enhanced Mail для управління ключами. Індустрія електронних платежів використовує 3DES і продовжує активно розробляти та публікувати стандарти, що ґрунтуються на ньому (наприклад, EMV). Компанія Microsoft для захисту даних системи і користувачів за допомогою парольного захисту використовує 3DES є, а саме в додатках Microsoft OneNote, Microsoft Outlook 2007 і Microsoft System Center Configuration Manager 2012. На сьогоднішній день не існує відомих криптографічних атак на 3DES, які можна застосувати на практиці. Але Triple DES потроху виходить з ужитку. На зміну даному алгоритму прийшов новий – AES Rijndael, який було розглянуто вище. Програмно реалізований алгоритм шифрування даних AES працює в шість разів швидше за 3DES. Саме з цієї причини шифр 3DES більше підходить для апаратних реалізацій. Багато систем безпеки продовжують підтримувати як 3DES, так і AES. Але за замовчуванням у них використовується алгоритм AES. Triple DES може підтримуватися для забезпечення сумісності, проте його більше не рекомендують до використання. Переваги Triple DES (3DES, TDEA): відомий шифр, який добре досліджений міжнародною спільнотою протягом більше ніж 30 років; забезпечує припустиму практичну стійкість (2112); поширений у банківських системах, імпортованих або орієнтованих на застарілі стандарти. Недоліки Triple DES: практична стійкість значно нижча теоретичної; наявність класів слабких ключів; швидкодія на сучасних системах суттєво нижча навіть порівняно із ДСТУ ГОСТ 2814:2009 та іншими блоковими шифрами.

ДСТУ 7624:2014 («Калина») - один із основних алгоритмів симетричного блокового шифрування, що використовуються в Україні, який визначає сучасний алгоритм симетричного блокового перетворення для забезпечення конфіденційності й цілісності інформації при її обробці та встановлює режими його роботи. В алгоритмі шифрування даних «Калина» використовуються криптографічні перетворення, які відповідають сучасним вимогам до рівня криптостійкості та швидкодії. Даний стандарт розроблено з урахуванням існуючих та потенційних загроз, подальшого інтенсивного розвитку інформаційних технологій та необхідності активного використання протягом кількох наступних десятиліть [6].

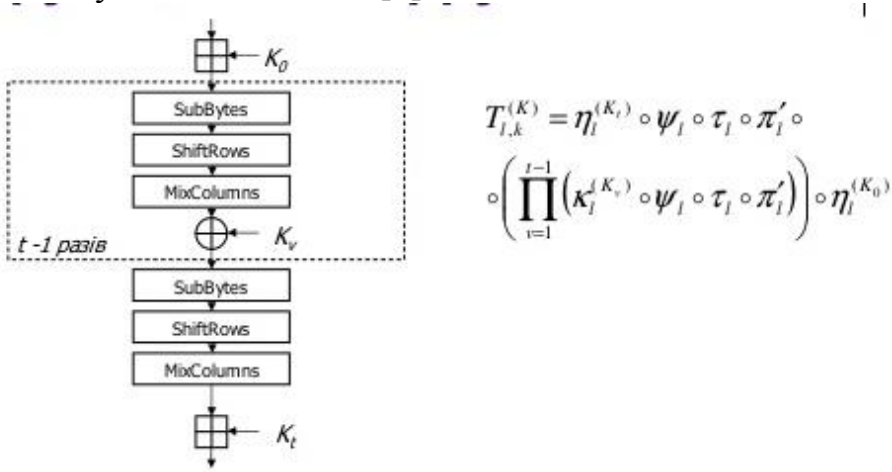

Рисунок 9.11- Функція шифруванн алгоритму «Калина»

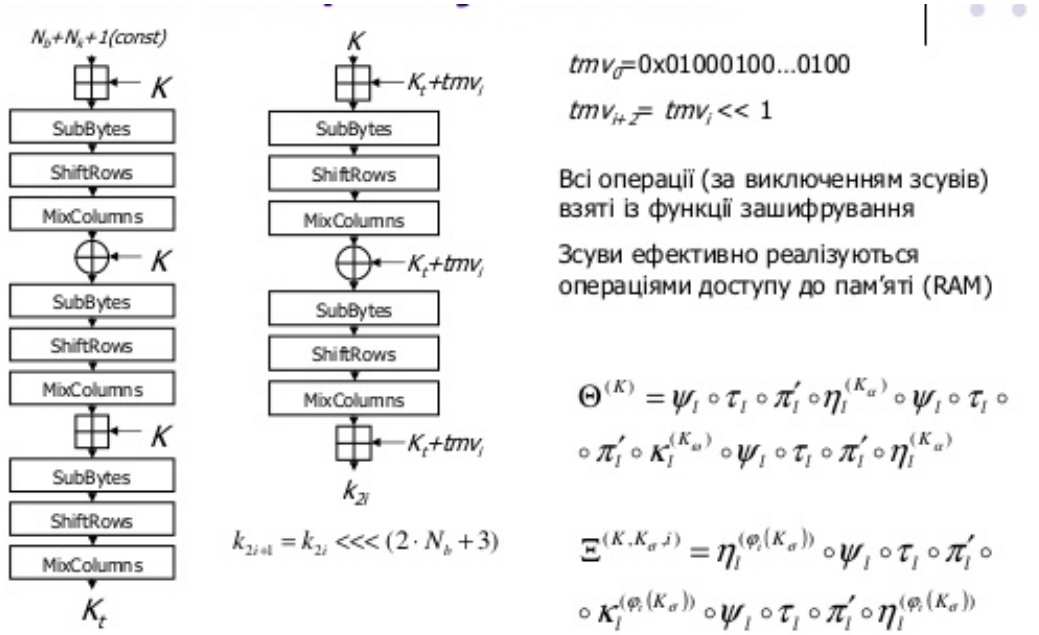

Рисунок 9.12- Формування циклових ключів алгоритму «Калина»

Стандарт блокового симетричного шифрування ДСТУ 7624:2014 визначає десять різних режимів роботи, що широко поширені відповідно до міжнародних стандартів ISO/IEC 10116:2006. Це спрямовано на забезпечення широкого застосування ДСТУ 7624:2014, у тому числі для захисту інформації, що передається комп'ютерними мережами, прозорого шифрування жорстких дисків і змінних носіїв, електронних документів, ключових даних.

Ефективність реалізації систем, засобів та протоколів криптографічного захисту інформації в інформаційно-телекомунікаційних системах різного призначення може бути забезпечена саме наявністю такої кількості режимів роботи алгоритму. До блокового шифру «Калина» ставляться такі вимоги: високий рівень криптографічної стійкості з достатнім запасом у разі появи нових атак протягом тривалого часу; висока швидкодія програмної реалізації на сучасних та перспективних платформах; компактність програмної та програмно-апаратної реалізації; можливість ефективної інтеграції декількох алгоритмів в одному засобі криптографічного захисту; прозорість проектування, консервативний підхід до забезпечення стійкості; вища (або однакова) ефективність порівняно з найкращими світовими рішеннями.

Стандарт симетричного блокового шифрування «Калина» є результатом багаторічної плідної співпраці Державної служби спеціального зв'язку та захисту інформації України та провідних українських вчених. Даний алгоритм шифрування враховує досвід і результати проведення міжнародних та відкритих національних конкурсів криптографічних алгоритмів. Алгоритм ДСТУ 7624:2014 забезпечує досить високий рівень криптостікості порівняно з міжнародним стандартом AES (ISO/IEC 18033-3:2010), оскільки дає можливість застосовувати блок даних і ключ шифрування розміром аж до 512 біт. Крім того, він має аналогічну або навіть більш високу швидкодію на сучасних і перспективних програмних та програмно-апаратних платформах. На даний

момент продовжуються роботи зі стандартизації вітчизняних криптографічних алгоритмів та протоколів.

Оскільки в стандартах симетричного блокового шифрування «Калина» та AES використовуються аналогічні криптографічні перетворення, на наш погляд, буде доцільним порівняти ці два алгоритми.

Основними відмінностями «Калина» від «Rijndael» (AES) є: збільшена кількість циклів шифрування (запас стійкості); використання додавання за модулем 264 і за модулем 2 для введення ключової інформації (захист від алгебричних атак, лінійного та диференціального криптоаналізів, інтерполяційної атаки тощо); використання чотирьох блоків нелінійного перетворення (S-блоків) замість одного (додатковий захист від алгебричних атак, поліпшення властивостей розсіювання алгоритму – покращені статистичні властивості, відповідно, більш високий рівень стійкості до диференціального та лінійного криптоаналізів тощо); використання випадково сформованих чотирьох блоків, відібраних критеріями стійкості до диференціального, лінійного криптоаналізів, ступені нелінійності булевих функцій (на відміну від S-блоку Rijndael/Camellia та інших шифрів, що використовують звернення в полі та, відповідно, квадратичні залежності між входом і виходом, – захист від алгебричних атак); принципово нова схема створення підключів (захист від усіх відомих атак на схеми створення підключів); досить висока продуктивність; можливість відновлення сеансового ключа за окремим підключем (додатковий захист від атак, що виконують відновлення підключів). Усі поліпшення спрямовані на збільшення стійкості та запобігання потенційним вразливостям відносно Rijndael, виявленим в останні роки.

2020 року запроваджено новий Національний стандарт **ДСТУ 9041:2020**.

Його повна назва: **ДСТУ 9041:2020. Інформаційні технології. Криптографічний захист інформації**. Алгоритм шифрування коротких повідомлень, що ґрунтується на скручених еліптичних кривих Едвардса.

Цей алгоритм призначений для шифрування коротких (до 616 біт) повідомлень для будь-яких алгоритмів шифрування, в тому числі визначених національними стандартами України.

Як і стандарт цифрового підпису ДСТУ 4145:2002, новий алгоритм використовує криптографічні перетворення у групі точок еліптичних кривих, використовуючи замість кривих у формі Вейєрштрасса найновітніші розробки у галузі еліптичної криптографії – криві у формі Едвардса. Це дає суттєві переваги у швидкодії більш ніж у 3 рази. Новий стандарт розроблений з урахуванням усіх найсучасніших вимог до стійкості криптографічних алгоритмів. Так, нижня межа стійкості криптосистем у цьому стандарті дорівнює 2127 (≈1042) (це більш ніж у півтора рази вище, ніж у ДСТУ 4145) і можуть бути обрані інші рівні, такі як 2255 (≈1085), 2383 (≈10127) та 2767 (≈10255); крім того, строго обґрунтована його стійкість як до атак на відновлення відкритого тексту, так і до розрізнюючих атак.

Проект алгоритму шифрування, що ліг в основу цього стандарту, пройшов апробацію як в Україні, так і за її межами (Центрально-Європейська

конференція з криптографії (червень 2020 року) – форум ведучих криптологів з усього світу).

Стандарт ДСТУ-9041 узгоджений з усіма діючими в Україні національними стандартами. Новиною стандарту є його сфера застосування – інкапсуляція ключів, найсучасніший математичний апарат, а також новий алгоритм генерації псевдовипадкових послідовностей, який, на відміну від аналогічного алгоритму генерації з ДСТУ 4145, використовує виключно національні криптографічні алгоритми національних стандартів та не містить посилань на відповідні пост-радянські стандарти, термін дії яких вже практично вичерпався.

Стійкість алгоритму буде під загрозою лише тоді, коли з’являться квантові комп’ютери з 700 і більше кубітами (на даний час кількість "робочих" кубітів, які вдалося створити, - близько 50). Його перевагою перед постквантовими алгоритмами є відносно невелика довжина ключа (у десятки або навіть у сотні разів менша, ніж у постквантових алгоритмах)[6].

10. ПОТОКОВІ ШИФРИ

10.1. Основні відомості

Потоковий шифр – це симетричний шифр, в якому кожен символ відкритого тексту перетворюється на символ шифрованого тексту в залежності не тільки від використовуваного ключа, але й від його розташування в потоці відкритого тексту. Потокові шифри являють собою різновид гамування. Генератор ключової послідовності (генератор випадкової чи псевдовипадкової послідовності) видає послідовність бітів $k_1,...k_i,....$ Це послідовність додається за модулем два ($\oplus$) із послідовністю бітів вихідного тексту $x_1,...x_i,...$: $y_i = x_i \oplus k_i$ (рис. 10.1). Стійкість системи цілком залежить від внутрішньої структури генератора ключової інформації: якщо генератор видає послідовність з невеликим періодом, то стійкість системи буде невеликою. Насправді, генератор видає потік бітів, який виглядає випадковим, але в дійсності є детермінованим і може бути точно відтворений на приймаючій стороні. Обов'язково, щоб генератор не видав одну й ту ж послідовність, бо $y_1 = x \oplus k, y_2 = y \oplus k \Rightarrow y_1 \oplus y_2 \Rightarrow x \oplus y$, що розкрити просто. Тому всі генератори передбачають наявність ключа, від якого залежить вихід генератора ключової послідовності і в такому випадку криптоаналіз неможливий. Потокові шифри придатні для шифрування неперервних потоків, наприклад в мережах передачі даних. Вони вирізняються високою швидкістю шифрування, легкою реалізацією [9].

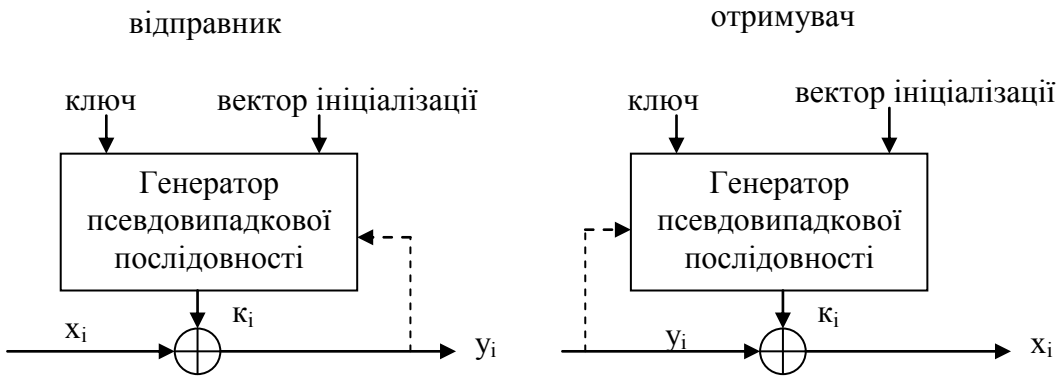

Рисунок 10.1 – Схема потокового шифрування

У 1946 р. – запатентована базова ідея потокових шифрів (шифр з авто ключем) у США (Cipher Text Auto Key – CTAK), або шифри що синхронізуються. Внутрішній стан генератора є функцією фіксованого числа попередніх *n* бітів шифрованого тексту. Кожне повідомлення називають випадковим заголовком (*n* бітів). Цей заголовок шифрується й передається в лінію, на приймаючій стороні він розшифровується. Результат буде неправильним, але після обробки *n* бітів заголовка обидва генератори будуть синхронізовані.

Недолік системи – поширення помилок, при шифруванні одного біту, генератор на приймаючій стороні видасть n неправильних бітів ключової послідовності, що приведе до помилкового розшифрування.

Велику популярність потоковим шифрам принесла робота Клода Шеннона, опублікована в 1949 році, в якій автор довів абсолютну стійкість шифру Вернама (також відомого, як одноразовий блокнот). У шифрі Вернама ключ має довжину, рівну довжині переданого повідомлення, ключ використовується як гама, і якщо кожен біт ключа вибирається випадково, то розкрити шифр неможливо.

Гамування чутливо до синхронізації гами і шифротексту. Якщо в ході передачі буде пошкоджено один символ шифротексту, то це не вплине на правильність дешифрування решти символів. Якщо ж при передачі втратити хоча б один знак криптограми, то весь подальший текст дешифрується неправильно. Тому необхідно забезпечити синхронізацію зашифрування і розшифрування текстів. За способом синхронізації потокові шифри класифікують на 2 типи:

1) Синхронні потокові шифри, в яких знаки згенерованої ключової гами не залежить від відкритого і шифрованого текстів, а залежать тільки від вихідного секретного ключа шифру, тобто $k_i = f(K)$. Головна властивість синхронних шифрів – нерозповсюдження помилок. Помилки відсутні, поки працюють синхронно шифрувальний і дешифрувальний пристрої відправника і одержувача інформації. Тобто поки при дешифрування не виникла ситуація $y_i = x_i + k_j, i \neq j$. Такі перебої називають розсинхронізацією. Її може викликати розбіжність швидкостей зашифрування/розшифрування на різних кінцях каналу зв'язку, випадання знаків при передачі та ін. Якщо так сталося, то треба відновити синхронність у роботі генераторів гами – почати повторне шифрування з реініціалізацією ключа обома користувачами.

Синхронні потокові шифри уразливі до атак на основі зміни окремих біт шифротексту, коли зловмисник може змінити ці біти так, що шифротекст розшифрується так, як це йому вигідно.

Синхронні потокові шифри можна реалізувати у вигляді блокового шифру у режимі OFB (зворотного зв'язку за виходом) або в режимі лічильника.

Синхронні потокові шифри володіють наступними властивостями:

– Вимоги щодо синхронізації. При використанні синхронних потокових шифрів одержувач і відправник повинні бути синхронізовані – тобто виробляти однакові значення ключового потоку для відповідних знаків переданого потоку даних. Якщо синхронізація порушиться (наприклад, внаслідок втрати знака при передачі), процес розшифрування не дасть коректного результату.

– Відсутність розмноження помилок. Зміна знака шифротексту при передачі не викликає помилок при розшифруванні інших знаків шифротексту.

– Властивість активної атаки. Як наслідок першої властивості, будь-яка вставка або видалення символу із шифротексту активним противником призводить до порушення синхронізації і виявляється одержувачем, розшифрованим повідомленням. Як наслідок другої властивості, активний

противник може змінювати символи шифротексту і ці зміни призведуть до відповідних змін у відкритому тексті, одержуваному при розшифруванні. Тому необхідні додаткові механізми, які дозволяють цьому запобігти.

2) Потокові шифри, які самосинхронізуються. У них знаки ключової гами залежать від вихідного секретного ключа шифру $k_i = f(K, y_{i-1}, y_{i-2}, .., y_{i-L})$ (цей варіант відзначений пунктиром на схемі потокового шифрування). У цьому випадку на стороні одержувача генератор почне синхронно працювати з передавальною стороною після отримання L бітів. Ідея запатентована в 1946 р. в США. Використання таких потокових шифрів характерно для військової криптографії і в літературі майже не освітлене. Недолік цих потокових шифрів – поширення помилок, так як спотворення одного біту в процесі передачі шифротексту призведе до спотворення L бітів гами. Ці шифри можна реалізувати у вигляді блокового шифру в режимі CFB (зворотного зв'язку за шифротекстом), при цьому за один раз може шифруватися довільне число бітів, менше або рівне довжині блоку.

Потокові шифри, які самосинхронізуються володіють наступними властивостями:

– Самосинхронізація. Самосинхронізація існує при видаленні або вставці деяких знаків шифртексту, оскільки процес розшифрування залежить від деякого фіксованого числа попередніх знаків шифротексту. Це означає, що у разі видалення знаку з шифртексту спочатку будуть помилки при розшифруванні, а потім все стане добре і помилок не буде.

– Обмежене розмноження помилок. Припустимо, що стан шифру залежить від t попередніх знаків шифртексту. Якщо під час передачі один знак шифротексту був змінений або видалений/вставлений, то при розшифровці буде спотворене не більше t знаків, після яких піде знову нормальний текст.

– Властивість активної атаки. З другої властивості випливає, що будь-яка зміна знаків шифртексту активним противником призведе до того, що кілька знаків шифротексту розшифруються неправильно і це з більшою (порівняно з синхронними шифрами) ймовірністю буде помічено з боку одержувача, розшифрованого повідомлення. Однак у випадку вставки або видалення знаків шифртексту (за властивістю 1) це набагато важче виявити (у порівнянні з синхронними шифрами – де виходить розсинхронізація). Тому необхідні додаткові механізми для контролю цієї ситуації.

– Розсіювання статистики відкритого тексту. Оскільки кожен знак відкритого тексту впливає на весь наступний шифртекст, статистичні властивості відкритого тексту (адже він далеко не випадковий) не зберігатимуться в шифртексті.

Основним недоліком потокових шифрів є необхідність передачі інформації для початкової ініціалізації (синхропосилання) перед заголовком повідомлення, яка повинна бути прийнята до розшифрування будь-якого повідомлення. Це пов'язано з тим, що синхропосилання може також бути і секретним ключем (частиною секретного ключа). Передача синхропосилання у вигляді, в якому воно буде застосовуватися в алгоритмі шифрування по каналу передачі даних може створити загрозу криптографічної стійкості системи, і

тому завжди необхідно застосовувати додатковий ключ, за допомогою якого синхропосилання буде приховуватися, або використовувати в алгоритмі шифрування ключа залежну модифікацію синхропосилання, як це реалізовано в ГОСТ 28147-89.

До теперішнього часу було придумано чимало алгоритмів потокового шифрування, наприклад, такі як: A3, A5, A8, RC4, PIKE, SEAL, eSTREAM та інші.

10.2. Алгоритм RC4

RC4 є потоковим шифром із змінною довжиною ключа. Він розроблений у 1987 р. Роном Рівестом для компанії RSA Data Security, Inc. Упродовж 7 років цей шифр ліцензувався компанією лише за умови нерозголошення. Проте 1994 р. був опублікований анонімно в Internet.

Алгоритм працює в режимі оберненого зв'язку за виходом. Ключова послідовність не залежить від виходу тексту. Структура алгоритму містить блоки заміни розміром $8\times 8: S_0,...S_{255}$. Блок заміни є залежним від ключа. Алгоритм містить два лічильники i, j, що на початку дорівнюють 0.

Для генерування псевдовипадкового байта використовують дії:

$i=(i+1)\bmod 256,\quad j=(j+S_i)\bmod 256,\quad S_i \leftrightarrow S_j,\quad t=(S_i+S_j)\bmod 256,\quad k=S_t$ – ключ, $x\oplus k\to y$ (x – вихідний байт, y – шифр).

Ініціалізація блоку заміни: на початку $S_0=0, S_1=1,...,S_{255}=255$. Ключем заповнюється ще один масив $k_0...\cdot k_{255}$

$j=0$

for $i=0$ *to* 255

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

перестановка S_i і S_j.

Застосування RC4

1. В якості алгоритму шифрування в протоколах безпеки WEP.
2. Для захисту інформації в документах PDF.
3. Один з алгоритмів, застосовуваних у Skype, для захисту трафіку.
4. В якості алгоритмів шифрування документів MS Office та інше.

10.3. Потокові шифри на базі регістрів зсуву

У 1965 році Ернст Селмер, головний криптограф норвезького уряду, розробив теорію послідовності зсувних регістрів. Пізніше Соломон Голомб, математик Агентства Національної Безпеки США, написав книгу під назвою «Shift Register Sequences» («Послідовності зсувних регістрів»), в якій виклав свої основні досягнення в цій галузі, а також досягнення Селмера.

Послідовності регістрів зсуву використовуються як в криптографії, так і в теорії кодування для генерування псевдовипадкових послідовностей. Їх теорія прекрасно опрацьована, потокові шифри на базі зсувних регістрів були робочою конячкою військової криптографії задовго до появи електроніки.

Регістр зсуву являє собою послідовність бітів (кількість бітів визначається довжиною зсувного регістру. Якщо довжина дорівнює *n* бітам, то регістр називається *n*-бітовим регістром зсуву). Одним із найпростіших у реалізації і найпоширеніших є регістри зсуву з лінійним зворотнім зв'язком – LFSR (Linear Feedback Shift Register). Вони знайшли широке використання у різних галузях науки і техніки. Детальніше із LFSR ми познайомимося у розділі 10.

Є кілька причин використання лінійних регістрів зсуву в криптографії:

1. Висока швидкодія криптографічних алгоритмів.

2. Застосування тільки найпростіших операцій додавання і множення, апаратно реалізованих практично у всіх обчислювальних пристроях.

3. Хороші криптографічні властивості (згенеровані послідовності мають великий період і хороші статистичні властивості).

4. Легкість аналізу з використанням алгебраїчних методів за рахунок лінійної структури.

Шифр A5

A5 – це потоковий шифр, використовуваний для шифрування GSM. Це європейський стандарт для цифрових стільникових телефонів. Він використовується для шифрування каналу «телефон - базова станція».

A5 складається з трьох LFSR довжиною 19, 22 і 23 розряди, виходом є XOR трьох регістрів. У A5 використовується змінюване управління тактуванням. Існує тривіальне розкриття такого шифру, тому варіанти A5 із довшими регістрами зсуву є більш безпечними.

SEAL

SEAL (Software-optimized Encryption Algorithm), програмно-оптимізований алгоритм шифрування – симетричний потоковий алгоритм шифрування даних, оптимізований для програмної реалізації.

В 1991 Ральф Меркле (Ralph C. Merkle) описав рентабельність програмно-орієнтованих шифрів. На його думку, найбільш ефективними з них були Khufu, FEAL і RC4. Однак постійно збільшуються потреби клієнтів в надійній криптографії вимагали пошуку нових і доопрацювання старих рішень.

Влітку 1992 почалася розробка першої версії нового програмно-оптимізованого алгоритму SEAL 1.0. Розробники взяли основні ідеї та принцип роботи з блокового шифру Ральфа Меркле Khufu, який видався їм самим досконалим на той момент. Вони вирішили добитися кращих характеристик проекту (в основному швидкості), звузивши коло апаратури, на якій можлива його реалізація. Вибір був зроблений на користь 32-бітних машин мінімум з вісьмома регістрами загального призначення і кешем не менше 8 Кбайт. У березні 1993 було прийняте рішення створити блоковий шифр, але структура з сімейства псевдовипадкових функцій, розроблена до жовтня того ж року, працювала швидше, що схилило розробників до потокового шифрування. Ця структура складалася з чотирьох регістрів, кожен з яких змінював свого «сусіда» в залежності від таблиці, отриманої з ключа. Після деякої кількості таких модифікацій значення регістрів додаються в ключову послідовність, яка зростає з кожною ітерацією до тих пір, поки не досягне певної довжини.

Алгоритм розроблено в IBM Філом Рогевеем (Phil Rogaway) і Доном Копперсмітом (Don Coppersmith) у 1993 році. Він оптимізований і рекомендований для 32-бітних процесорів. Для роботи йому потрібно кеш-пам'ять на кілька кілобайт і вісім 32-бітних регістрів. Швидкість шифрування – приблизно 4 машинних такти на байт тексту. Для кодування і декодування використовується 160-бітний ключ. Щоб уникнути небажаної втрати швидкості з причини повільних операцій обробки ключа, SEAL попередньо виконує з ним кілька перетворень, отримуючи в результаті три таблиці певного розміру. Безпосередньо для шифрування і розшифрування тексту замість самого ключа використовуються ці таблиці. Алгоритм вважається надійним, дуже швидким і захищений патентом США № 5454039 з грудня 1993 року.

У 1996 році Helena Handschuh і Henri Gilbert описали атаки на спрощену версію SEAL 1.0 і на сам SEAL 1.0. Їм знадобилося 2^{30} текстів, кожен довжиною в чотири 32-бітних слова, щоб знайти залежність псевдовипадкової функції від ключа. У результаті, в наступних версіях алгоритму SEAL 3.0 і SEAL 2.0 були зроблені деякі доопрацювання і зміни. Ще SEAL 3.0 і SEAL 2.0 використовували для генерації таблиць алгоритм SHA-1, замість первісного SHA, що зробило їх більш стійкими до криптоаналізу.

11. ГЕНЕРАТОРИ ВИПАДКОВИХ І ПСЕВДОВИПАДКОВИХ ПОСЛІДОВНОСТЕЙ

11.1. Загальні відомості

Генерування випадкових і псевдовипадкових послідовностей є необхідним і надзвичайно важливим елементом криптографічних застосувань. Більше того, від якості генераторів, апаратних чи програмних, багато в чому залежить якість самої криптосистеми. Неякісний генератор випадкових послідовностей може піддаватися атакам зловмисників, бути причиною утворення слабких криптографічних ключів. Тому важливість якісного генерування випадкових криптографічностійких послідовностей важко переоцінити.

Криптографічні послідовності використовують у разі:

- генерування ключів симетричних та асиметричних криптосистем;
- генерування цифрових підписів;
- реалізації переважної кількості криптографічних протоколів;
- аутентифікації, що ґрунтується на криптографічних засобах;
- потокового шифрування;
- інших криптографічних застосувань.

Послідовності можна розділити на два класи: **істинно випадкові** (далі будемо називати їх **випадковими**) та **псевдовипадкові.**

Випадкові послідовності, як правило, породжуються апаратними або комбінованими, програмно-апаратними засобами, які використовують як генеруючий пристрій датчики фізичних величин. Дані, які постачають датчики фізичних величин, обробляються або електронними засобами, або програмними рішеннями, які, власне, й виробляють двійкові послідовності.

Псевдовипадкові послідовності, як правило, генеруються програмно.

Відмінності між істинно випадковими та псевдовипадковими послідовностями дуже значні. Основна відмінність – найважливіша, полягає в тому, що псевдовипадкова послідовність – періодична. Іншою суттєвою відмінністю є те, що за однакових початкових умов псевдовипадкова послідовність також буде однаковою.

Цих два недоліки псевдовипадкових послідовностей необхідно враховувати при проектуванні криптографічних систем, що використовують такі генератори.

Проектування криптографічно стійких псевдовипадкових послідовностей – дуже важлива криптографічна задача, що стала вже цілою індустрією. Прийнято вважати, що стійкий псевдовипадковий генератор, який можна сміливо використовувати в найскладніших криптографічних системах, повинен мати період не менший за 2^{256} [9].

11.2. Генератори випадкових послідовностей

На ринку немає дешевих апаратних засобів генерування справді випадкових послідовностей, що використовують результати вимірювання

певної фізичної величини. Як правило, для побудови таких генераторів можна використати датчик практично довільної фізичної величини, що вимірює її значення з великою точністю:

- температури оточуючого середовища;
- рівня радіоактивності;
- рівня сейсмічної активності;
- рівня освітленості;
- опору еталонного резистора;
- інших фізичних величин.

Схема обробки опитує датчик із деякою періодичністю та використовує отримані дані для формування двійкової випадкової послідовності.

У подальшому сигнал може використовувати програмна (або апаратна) реалізація криптографічного пристрою для різних цілей.

Перевагами фізичних генераторів випадкових послідовностей можна вважати: відсутність періодичності послідовності, що генерується та неможливість отримання однакових послідовностей навіть при однакових зовнішніх умовах.

Отже, проектування та реалізація генератора випадкових послідовностей такої схеми – складна і нетривіальна інженерна задача, причому найскладнішою частиною можна вважати саме датчик фізичної величини, який повинен вимірювати цю величину з максимально можливою точністю. Однак як датчик можна використати будь-який комп'ютер.

Непоганими схемами вважається генерування випадкових послідовностей з використанням клавіатури або мишки. Наприклад, програмне забезпечення (ПЗ) «просить» користувача набирати на клавіатурі будь-яку інформацію (краще беззмістовний набір символів) деякий проміжок часу. ПЗ аналізує дії користувача і може використовувати їх, як мінімум, у два різних способи. Перший спосіб полягає в тому, що для генерування випадкової послідовності використовуються проміжки часу, виміряні або в «тіках» процесора, або в мілісекундах, між послідовними натисканнями на клавіші. Цей час у двійковому представленні обробляється в певний спосіб (наприклад, виконується додавання всіх розрядів за модулем 2) і на вихід подається черговий біт послідовності. При такому використанні не має значення, що саме набирає користувач на клавіатури. Другий спосіб обробки отриманої інформації полягає в тому, що ПЗ використовує саме символи, які набирає користувач. У цьому разі також буде краще, якщо користувач набиратиме невпорядкований набір символів. Двійкове представлення чергового символу піддається деякій обробці (в найпростішому випадку це може бути та сама сума всіх розрядів за модулем 2 чи вибірка молодших або старших розрядів цього представлення з наступним їх додаванням за модулем 2 тощо) та на вихід подається біт (або кілька бітів) випадкової двійкової послідовності.

Перевагами таких способів генерування двійкових послідовностей є відносна простота їх реалізації. Для них, крім комп'ютера, програмного забезпечення та користувача нічого не потрібно. Водночас якість таких послідовностей практично ідентична фізичним генераторам випадкових

послідовностей.

До недоліків таких способів можна віднести неможливість (або принаймні практичну складність) отримання безмежних послідовностей для потокового шифрування, хоча для інших цілей обидва способи достатньо зручні [9].

11.3. Генератори псевдовипадкових послідовностей

Основні вимоги до генераторів псевдовипадкових послідовностей:

1) Необхідно, щоб псевдовипадкові послідовності якнайменше відрізнялися від справді випадкових.

2) Псевдовипадкова послідовність не повинна бути передбачуваною. Це означає, що знаки всіх попередніх бітів послідовності не дозволяє передбачити наступний біт з ймовірністю, більшою за 50% .

3) Довжина періоду повторень не менша ніж 2^{256} .

4) Продуктивність алгоритму – кількість чисел за 1 сек (10^{10} – 10^{12}).

11.3.1. Найпростіші ГПВП (зараз в криптографії не застосовуються):

RAND()

$N_{i+1} = (N_i \cdot 1103515245 + 12345) \bmod 4294967296.$

Недоліки: період дуже малий; на різних машинах одна і та ж послідовність.

FRAC()

$Y_n = FRAC(A \cdot n + B)$, де A, B – початкові дробові числа великої розрядності;

– при раціональному A – множина результату Y_n скінченна.

– при ірраціональному A – множина результату Y_n нескінченна.

$Y_n = FRAC(A \cdot n^2 + B \cdot n + C)$ – для покращення властивостей.

Недоліки: для представлення ірраціональних чисел необхідна велика кількість розрядів, що практично не можливо; для раціональних чисел – малий період; не можна строго визначити нижню межу періоду повторення, бо вона дуже залежить від початкових даних.

Метод квадратів

Перший алгоритмічний метод одержання рівномірно розподілених псевдовипадкових чисел запропонував Джон фон Нейман (один з основоположників кібернетики). Метод одержав назву «метод квадратів».

Суть методу: попереднє випадкове число зводиться у квадрат, а потім з результату витягаються середні цифри. Метод середини квадрата досить добре повинен «перемішувати» попереднє число. Однак він має недоліки: якщо який-небудь член послідовності виявиться рівним нулю, то всі наступні члени також будуть нулями; послідовності мають тенденцію «зациклюватися», тобто зрештою, утворять цикл, що повторюється нескінченне число раз. Властивість «зациклюватися» притаманна всім послідовностям, побудованих за

рекурентною формулою $x_{i+1}=f(x_i)$. Повторюваний цикл називається періодом, довжина періоду в різних послідовностей різна.

Деякі вчені експериментували з методом середин квадратів на початку 50-х років. Працюючи із чотиризначними числами замість восьмизначних, Дж. Е. Форсайт випробував 16 різних початкових значень і виявив, що 12 з них приводять до циклічних послідовностей, що закінчуються циклом 6100, 2100, 4100, 8100, 6100....., у той час два з них приводять до послідовностей, що вироджується в нульові. Більш інтенсивні дослідження, головним чином у двійковій системі числення, провів Н. К. Метрополис. Він показав, що якщо використати 20-розрядне число, то послідовність випадкових чисел, отримана методом середин квадратів, вироджується в 13 різних циклів, причому довжина найбільшого періоду дорівнює 142.

Приклад

– Вибираємо випадкове число – 151.

– Число підносимо до квадрату – $151^2 = 22801$.

– Вибираємо середні розряди і підносимо до квадрату – $280^2 = 78400$

– Аналогічно, $840^2 = 705600$ і т.д.

Отримана послідовність: 151, 22801, 78400, 705600 …

Конгруентний генератор

Одним із найбільш популярних ГПВП є генератори, у яких використовується схема, запропонована Д. Г. Лехмером в 1949 році – лінійний конгруентний метод.

Виберемо чотири «чарівних числа»: m – модуль, $m>0$; a – множник, $0<a<m$; c – приріст; $0<c<m$; x_0 – початкове значення, $0<x_0<m$. Потім одержимо бажану послідовність випадкових чисел (x_n), маючи на увазі $x_{n+1}=(ax_n+c) \bmod m$.

Ця послідовність називається лінійною конгруентною послідовністю. Одержання залишків по модулю m часто нагадує зумовленість, коли кулька попадає в осередок колеса рулетки.

Наприклад, для m=10 і x_0=a=c=7, одержимо послідовність7,6,9,0,7,6,9,0,....

Як показує цей приклад, така послідовність не може бути «випадковою» при деяких наборах чисел m, a, c і x_0. У прикладі ілюструється той факт, що конгруентна послідовність завжди утворить петлі, тобто обов'язково існує цикл, що повторюється нескінченне число раз. Ця властивість є загальною для всіх послідовностей виду $x_{n+1} = f(x_n)$, де f перетворить кінцеву множину саме в себе. Цикли, що повторюються, називаються періодами; довжина періоду послідовності, розглянутої в прикладі, дорівнює 4. Безумовно, послідовності можуть мати довгий період.

Заслуговує на увагу випадок, коли $c=0$, тому що числа, що генеруються, будуть мати менший період, ніж при $c\neq0$. Ми переконаємося надалі, що обмеження $c=0$ зменшує довжину періоду послідовності, хоча при цьому усе ще можливо зробити період досить довгим. В оригінальному методі, запропонованому Д. Г. Лехмером, c вибиралося рівним нулю, хоча він і допускав випадок, коли $c\neq0$, як один з можливих. Той факт, що умова $c\neq0$ може приводити до появи більше довгих періодів, був установлений В. Е. Томсоном

(W. E. Thomson) і незалежно від нього А. Ротенбергом (A. Rotenberg). Багато авторів називають лінійну конгруентну послідовність при $c=0$ мультиплікативним конгруентним методом, а при $c\neq0$ – змішаним конгруентним методом. Можна відразу відкинути випадок, коли $a=1$, при якому послідовність x_n може бути представлена у вигляді $x_n=(x_0+c)mod\ m$ і поводиться явно не як випадкова послідовність. Випадок, коли $a=0$ навіть гірше. Отже, для практичних цілей припускаємо, що $a>2$.

Практична реалізація лінійного конгруентного методу в стандартних бібліотеках різних компіляторів

Лінійний конгруентний метод застосовується в простих випадках і не має криптографічну стійкість. Входить у стандартні бібліотеки різних компіляторів. При практичній реалізації лінійного конгруентного методу вигідно вибирати $m=2e$, де e — число біт у машинному слові. Машинне слово – машиннозалежна й платформозалежна величина, вимірювана в бітах або байтах, рівна розрядності регістрів процесора й/або розрядності шини даних.

У таблиці 11.1 нижче наведені найбільш часто використовувані параметри лінійних конгруентних генераторів, зокрема, у стандартних бібліотеках різних компіляторів.

Таблиця 11.1

Параметри лінійних конгруентних генераторів

Компілятор	*m*	*a*	*c*
Numerical Recipes	232	1664525	1013904223
Borland C/C+ +	232	22695477	1
GNU Compiler Collection	232	69069	5
ANSI C: Open Watcom, Digital Mars, Metrowerks, IBM VisualAge C/C+ +	232	1103515245	12345
Borland Delphi, Virtual Pascal	232	134775813	1
Microsoft Visual/Quick C/ C+ +	232	214013	2531011
Apple CarbonLib	231 - 1	16807	0

Нажаль, такі послідовності за умови знання початкових умов передбачувані, а початкові умови не так складно знайти. Тому конгруентивні генератори не використовуються в криптографії.

11.3.2.Сучасні ГПВП

Існує велика кількість різноманітних методів та принципів генерування ПВП. Розглянемо основні методи отримання ПВП, які найбільше застосовувані у криптографії [8; 11; 16].

Генератор Блюма – Блюма – Шуба (BBS)

Програмний генератор двійкових послідовностей BBS (назву утворено від перших літер його авторів – Ленори та Мануеля Блум та Майка Шуба) вважають одним із найсильніших програмних генераторів псевдовипадкових послідовностей. Він криптографічно стійкий і може мати серйозні криптографічні застосування.

Нехай є два простих числа, p і q, причому $p \equiv q \equiv 3 \bmod 4$. Добуток цих чисел $n = pq$ називається цілим числом Блума. Оберемо ще одне випадкове число x, взаємно просте з n та обчислимо $x_0 \equiv x^2 \bmod n$. Це число вважається стартовим числом генератора. Далі можна обчислити наступні біти послідовності за формулами: $x_i \equiv x^2{}_{i-1} \bmod n$ і $S_i \equiv x_i \bmod 2$. Останнє означає, що як вихід генератора обирається молодший біт числа x. Отже, можемо записати:

$x_0 = x^2 \bmod n$

$for\ i = 1\ to\ \infty$

$x_i = x^2{}_{i-1} \bmod n$

$S_i \equiv x_i \bmod 2.$

Найцікавішою властивістю генератора BBS є те, що для визначення значення *i*-го біта зовсім необов'язково знати всі попередні *i-1* бітів. Для безпосереднього обчислення значення *i*-го біта достатньо знати p і q.

Безпека цієї схеми ґрунтується на складності розкладання n на множники. Число n можна опублікувати, так що кожен зможе генерувати біти за допомогою цього генератора. Однак поки криптоаналітик не розкладе n на множники, він не зможе передбачити вихід генератора. Більше того, генератор BBS непередбачуваний як у правому, так і в лівому напрямках. Це означає, що, отримавши послідовність бітів, криптоаналітик не зможе передбачити ні наступний, ні попередній біти послідовності. Причиною цього є не якийсь заплутаний механізм генерації, а математика розкладання n на множники.

Приклад

$p = 19; q = 23, p = q \equiv 3 \bmod 4, n = 437, x_0 = 233.$

Таблиця 11.2

Генерація чисел генератором BBS

i	0	1	2	3	4	5	6
x_i	101	150	213	358	123	271	25
S_i	1	0	1	0	1	1	1

Цей генератор повільний, але є спосіб його прискорення. Як біти псевдовипадкової послідовності можна використовувати не один молодший біт, а $\log_2 m$ молодших бітів, де m – довжина числа x_i. Порівняна повільність цього генератора не дозволяє використовувати його для потокового шифрування, а от для високонадійних застосувань, як, наприклад, генерування ключів, він вважається кращим за багато інших.

Генератор Блум – Мікалі

Безпека цього генератора базується на проблемах обчислення дискретного логарифма в кінцевому полі.

Для реалізації цього генератора генерують два простих числа: g та p. Зародок x_0 породжує такий процес генерації: $x_{i+1} \equiv g^{x_i} \bmod p$. Виходом генератора буде 1, якщо $x_i < (p-1)/2$, і 0 – якщо ця нерівність не виконується. Якщо модуль p достатньо великий для того, щоб обчислення дискретного логарифму було обчислювально складною задачею, цей алгоритм безпечний.

Генератор RSA

Генератор RSA використовує алгоритм шифрування RSA для утворення псевдовипадкових послідовностей. Застосовується публічний ключ (*e, n*): оберемо випадкове число $x_0 < n$ і обчислимо:

$for\ i = 1 to \infty$

$x_i = (x_{i-1})^e \bmod n$

$B_i \equiv x_i \bmod 2$.

Безпека цього генератора ґрунтується на складності розкладання великого числа на множники, тобто на тій самій задачі, яка лежить в основі криптостійкості самої системи RSA.

Генератори, побудовані на основі регістру зсуву

LFSR

Генератор на основі лінійного регістру зсуву із зворотнім зв'язком LFSR (Linear Feedback Shift Register). Він використовується в криптографії і в теорії кодування. Це пристрій, що складається з регістру зсуву, здатного запам'ятовувати двійкові послідовності кінцевої довжини та схеми, яка реалізує додавання за модулем 2 ($\oplus$) вибраних бітів з регістра (рис 11.1.) [12].

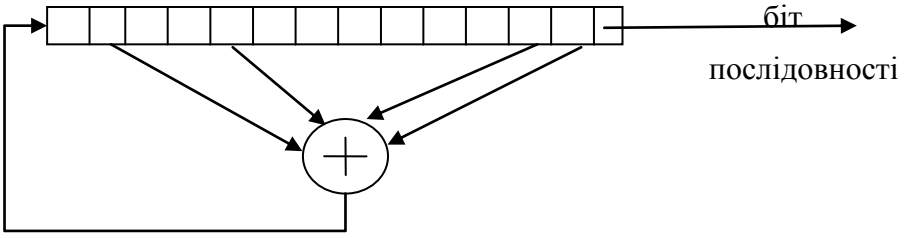

Рисунок 11.1 – Схема роботи генератора LFSR

Початкова послідовність породжується поліномом з коефіцієнтом у двійковому представленні (0; 1):

$P_n = A_n x^n + A_{n-1} x^{n-1} + \ldots + A_1 x + A_0$, де $A_n = A_0 = 1$, інші рівні 0 або 1.

Максимальний період генератора – $2^k - 1$, k – кількість тригерів у регістрі (степінь полінома). P_n не ділиться на ніякий інший поліном; P_n є дільником $x^{n+1} + 1$. Лінійний регістр зсуву працює так:

– виходячи з потрібного періоду генерації знаходять породжуваний поліном потрібного степеня;

– будується регістр зсуву відповідної розрядності;

– виходи тих тригерів, степені яких присутні в поліномі, заводяться на суматор $\oplus$;

– регістр заповнюється довільним початковим значенням;

– додатково на суматор додається константа «1» (0-ва степінь);

– вихід суматора подається на вхід регістру зсуву і виходом є біт справа.

Приклад 1

Припустимо, що ми хочемо побудувати LFSR із періодом 7. Для цього вибираємо поліном $x^3 + x + 1$, період якого становить $2^3 - 1 = 7$. Вхідний потік – 0001 (рис. 11.2).

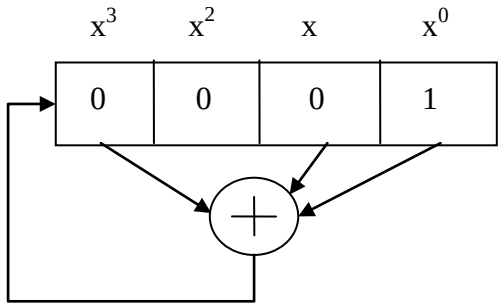

Рисунок 11.2 –Генератор LFSR-1

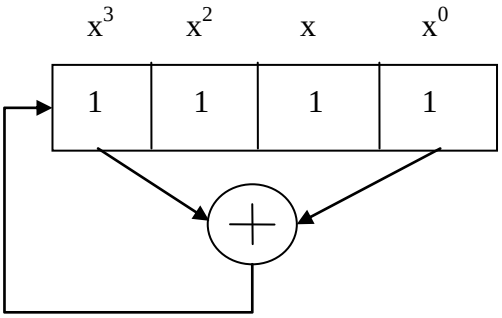

Рисунок 11.3 – Генератор LFSR-2

Таблиця 11.3

Приклад генерування послідовності 1

Вхідна послідовність				Вихідна послідовність
0	0	0	1	1
1	0	0	0	0
1	1	0	0	0
1	1	1	0	0
0	1	1	1	1
0	0	1	1	1
0	0	0	1	1
1	0	0	0	0
1	1	0	0	0
1	1	1	0	0
0	1	1	1	

Вихідна послідовність: 100011 100011 1…

Приклад 2

Твірний поліном – $P_n = x^3 + 1$, вхідний блок – 1111 (рис. 11.3).

Таблиця 11.4

Приклад генерування послідовності 2

Вхідна послідовність				Вихідна послідовність
1	1	1	1	1
0	1	1	1	1
1	0	1	1	1
0	1	0	1	1
1	0	1	0	0
1	1	0	1	1
0	1	1	0	0
0	0	1	1	1
1	0	0	1	1
0	1	0	0	0

Вихідна послідовність: 1111010110…

FCSR

Генератор на основі регістрів зсуву зі зворотним зв’язком та перенесення (Feedback with Carry Shift Register). Він схожий на LFSR, в обох є регістр зсуву та функція зворотного зв’язку, різниця полягає у тому, що у FCSR є також регістр перенесення. Існують два варіанти реалізації FCSR: конфігурація Галуа та Фібоначчі. Ми розглядатимемо конфігурацію Фібоначчі. У порівнянні з

LFSR, замість *xor* над усіма бітами відвідної послідовності, ці біти додаються один з одним і вмістом регістру перенесення. Результат *mod 2* стає новим бітом, результат *div 2* стає новим вмістом регістру перенесення (рис. 11.4) [12].

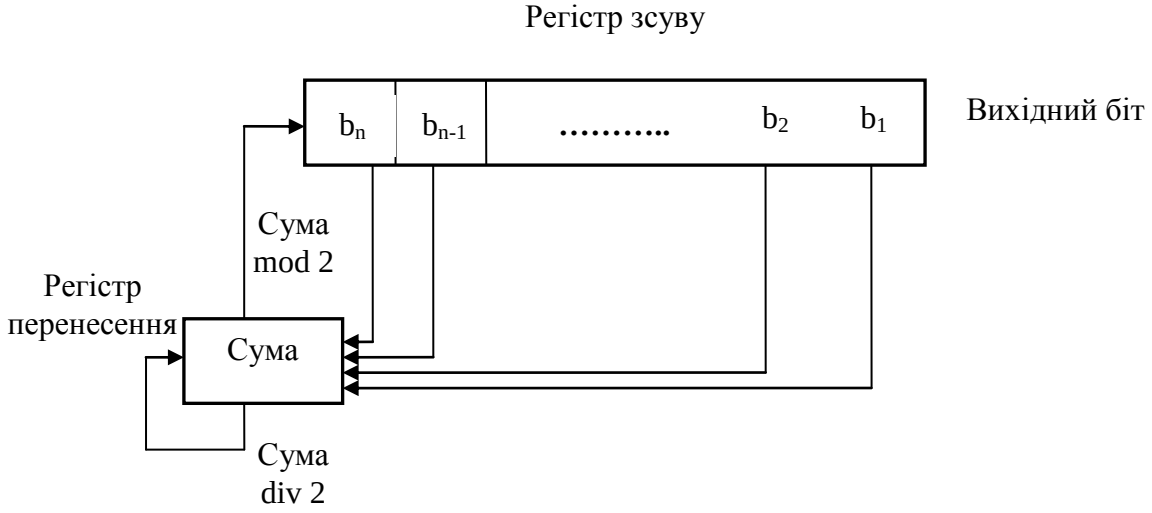

Рисунок 11.4 – FCSR

Існує початкова затримка перш ніж FCSR перейде у циклічний режим. Початковий стан FCSR відповідає ключу потокового шифру. Регістр перенесення не біт, а число, розмір якого повинен бути не менше $\log_2 t$, де t – кількість розгалужень у відвідній послідовності. Якщо є два розгалуження, то регістр перенесення однобітний; якщо три, чотири розгалуження, то регістр перенесення двобітний і т.д. Максимальний період FCSR дорівнює *q-1,* де q – ціле число зв'язку, яке задає розгалуження і визначається за формулою $q = 2q_1 + 2^2 q_2 + ... + 2^n q_n - 1$ (q_i відраховується зліва направо, наприклад, для 4-бітного регістру $b_4 b_3 b_2 b_1$ відраховуємо $q_1 q_2 q_3 q_4$). Крім того, q – просте число, для якого 2 є примітивним коренем. Якщо відвідна послідовність задана у вигляді (*a,b,c,d*), то $q = 2^a + 2^b + 2^c + 2^d - 1$.

Для прикладу побудуємо ПВП на основі FCSR з числом зв'язку q=19. Для того, щоб визначити відвідну послідовність, необхідно розрахувати бінарний склад числа q+1. $20 = 2^4 + 2^2 = 2 \cdot 0 + 2^2 \cdot 1 + 2^3 \cdot 0 + 2^4 \cdot 1$. Відвідна послідовність (4, 2), це означає, що у розгалуженні будуть приймати участь b_3 та b_1 біти регістру зсуву. Так як кількість розгалужень дорівнює 2, то для регістру перенесення необхідно лише один біт. Задамо початковий стан регістру зсуву – (1010) та регістру перенесення – 0. У таблиці 11.5 приведений приклад зміни вмісту регістрів та вихідна послідовність чисел.

Таблиця 11.5

Генерування послідовності

Регістр перенесення	Регістр зсуву	Вихідний біт
0	1010	
0	0101	**0**
1	0010	**1**
0	1001	**0**
0	1100	**1**
0	1110	**0**
0	1111	**0**
1	0111	**1**
1	1011	**1**
1	0101	**1**
1	1010	**1**
0	1101	**0**
1	0110	**1**
1	0011	**0**
1	0001	**1**
1	0000	**1**
0	1000	**0**
0	0100	**0**
0	1010	**0**
0	0101	0
1	0010	1

Як бачимо, період даної послідовності 010100111101011000, а довжина його становить q–1=18.

Для n-бітного LFSR максимальна довжина періоду становить 2^n–1, що суттєво менше довжини періоду n-бітного FCSR.

Наприклад, для 32-розрядного LFSR із твірним поліномом (32,7,5,3,2,1,0), тобто $x^{32}+x^7+x^5+x^3+x^2+x^1+1$, довжина періоду становить 2^{32}–1=4 294 967 295. Для відвідної послідовності (32,31,30,2) 32-розрядного FCSR довжина періоду дорівнює $2^{32}+2^{31}+2^{30}+2^2-2$=7 516 192 770.

Основний підхід до проектування ГПВП на основі FCSR аналогічний як і для LFSR, а саме, використовуються декілька регістрів з різними довжинами, які об’єднані лінійним чи нелінійним способом.

Розглянемо деякі види таких генераторів:

1. Генератор парності. Регістри FCSR або LFSR об’єднані функцією *xor*

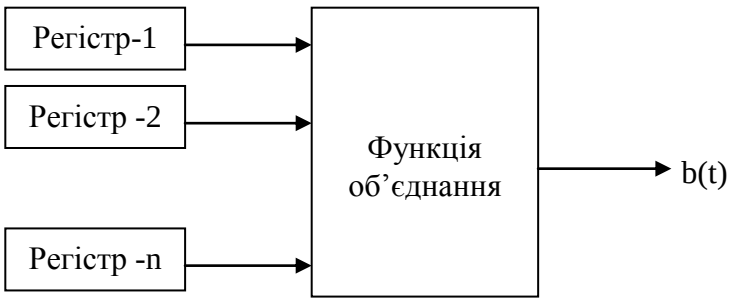

Рисунок 11.5 – Генератор парності

2. Генератор Геффа для регістрів FCSR або LFSR. Регістр-2,..,Регістр-n є входами мультиплексора, а Регістр-1 керує його виходом (рис. 11.6) Для трьох регістрів вихід генератора Геффа можна описати так: $b = a1 \wedge a2 \oplus \neg a1 \wedge a3$, де $a1, a2, a3$ – виходи Регістр-1, Регістр-2, Регістр-3.

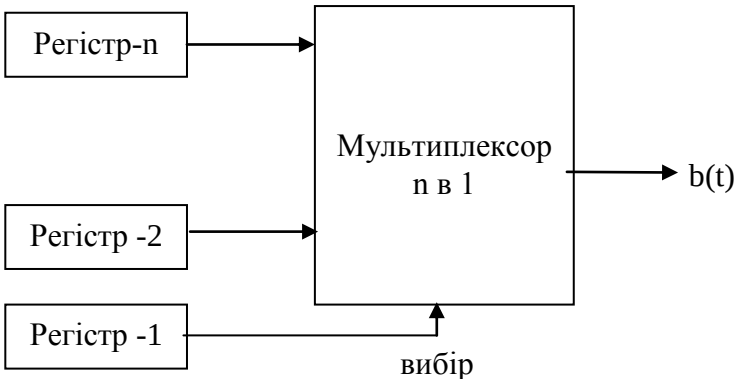

Рисунок 11.6 – Генератор Геффа

3. Пороговий (мажоритарний) генератор.

Для даного генератора необхідно використовувати непарну кількість регістрів. Якщо більше половини вихідних бітів регістра дорівнюють 1, то виходом генератора буде – 1, інакше – 0 (рис.7). Для трьох регістрів вихід генератора можна описати так: $b = a1 \wedge a2 \oplus a1 \wedge a3 \oplus a2 \wedge a3$.

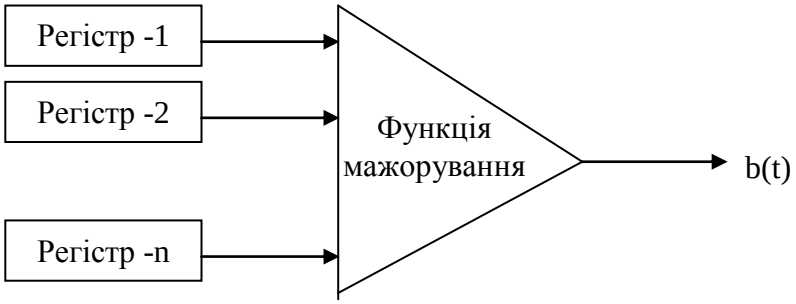

Рисунок 11.7 – Пороговий генератор

4. Генератор «stop-and-go». Використовуються три регістри. Регістр-2 змінює свій стан, якщо вихід Регістр-1 дорівнює 1, Регістр-3 змінює свій стан у протилежному випадку.

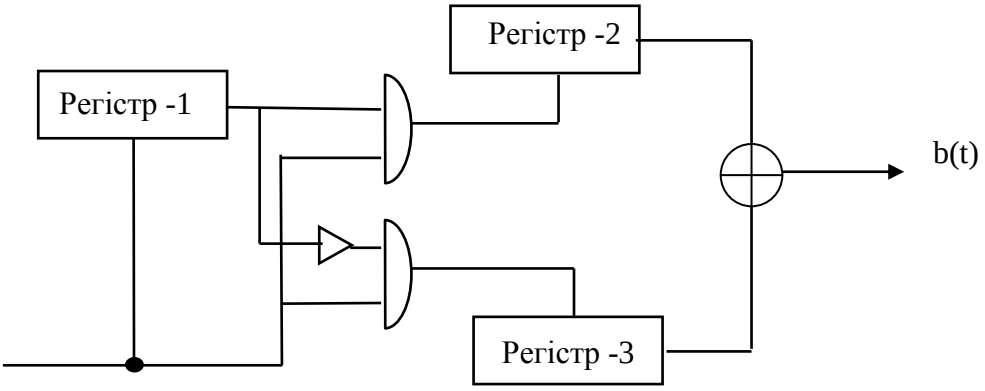

Рисунок 11.8 – Генератор «stop-and-go»

На рис. 11.8 представлено структурну схему модифікованого генератора «stop-and-go», де Регістр-2 змінює свій стан, якщо вихід Регістр-1 дорівнює 1, Регістр-3 змінює свій стан, коли вихід Регістр -1 дорівнює 0. Вихід генератора є *xor* Регістр-2 та Регістр-3. Перспективами подальших досліджень у даному напрямку є використання принципу «stop-and-go» для побудови генератора Голлманна із різною кількістю FCSR та FCSR/LFSR та інші види комбінацій цих генераторів.

Дослідивши послідовності, які побудовані за даним принципом, можемо стверджувати, що довжина періоду генераторів дорівнює $T = HCK(T_1, T_2, ..., T_n)/2$, де $T_1, T_2, ..., T_n$ – довжини періодів відповідно FCSR--1, FCSR-2,…, FCSR-n, НСК – найменше спільне кратне вказаних чисел. У частинних випадках довжина періоду – $T = HCK(T_1, T_2, ..., T_n)$ та в станньому випадку $T = 2 \cdot HCK(T_1, T_2, T_3)$ [14].

11.3.3. Використання симетричного криптоалгоритму

Використання симетричних алгоритмів для генерування псевдовипадкових послідовностей також може бути виправданим, оскільки вони сконструйовані так, щоб максимально перемішувати й розсіювати особливості вхідного тексту. Як показує аналіз, наприклад алгоритму DES, уже при 8 циклах бінарна послідовність на виході практично мало чим відрізняється від випадкової.

Що стосується алгоритму ГОСТ, то навіть у рекомендаціях по його використанню є режим, що називається «гаммуванням зі зворотним зв'язком», де сам алгоритм використовується для генерування т.зв. «гамми», яка побітово накладається на вхідний текст.

Оцінка якості генераторів псевдовипадкових послідовностей

Поверхнева оцінка може бути дана за допомогою програми, що дає спектрограму генерованого випадкового ряду (залежність кількості генерованих на виході значень від самих значень). При чому стартове значення ніякої ролі не відіграє. Має бути пряма лінія. Для перевірки відсутності залежності наступних чисел через попередні, використовують спеціальний кореляційний аналізатор

$F(k) = \sum_{i=k+1}^{N} (z_i \cdot z_{i-k})$, z_i - побудована послідовність; N – досить велике.

– При вдало побудованому генераторі графік $F(k)$ має бути лінією на всьому діапазоні від 1 до *N*.

– Існує думка, що для того щоб покращити генератор, необхідно скласти, помножити декілька генераторів (random $\overset{*}{+}$ random $\overset{*}{+}$ random). Це не зовсім так, навпаки – гірше.

Визначення якості та надійності генераторів випадкових та псевдовипадкових послідовностей – одна з основних задач сучасної прикладної та теоретичної криптографії, тому що вони використовуються для генерації ключів та інших випадкових параметрів криптосистем. Існують спеціальні методики тестування для оцінки якості випадкових послідовностей. На

сьогоднішній день розглядаються групи тестів, які використовуються для аналізу рівня безпеки ГПВП, а також різні методики інтерпретації отриманих результатів. Для дослідження якості генераторів псевдовипадкових чисел використовують дві групи тестів: графічні тести; статистичні тести.

Класифікація методик тестування генераторів псевдовипадкових чисел наведена на рис 9.

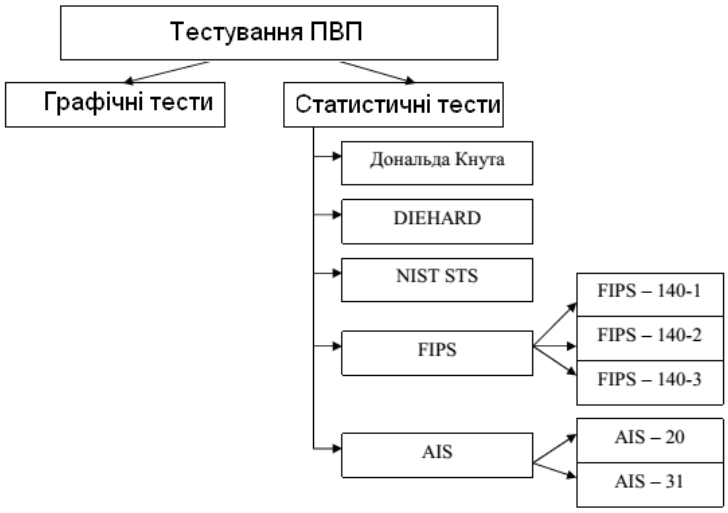

Рисунок 9. – Класифікація методик тестування ПВП

Графічні тести

При графічному тестуванні статистичні властивості послідовностей відображаються у виді графічних залежностей, за виглядом яких роблять висновки про властивості досліджуваної послідовності. Результати графічних тестів інтерпретуються людиною, тому висновки можуть бути неоднозначними. До даної категорії відносяться такі тести:

- гістограма розподілу елементів послідовності,
- розподіл на площині,
- перевірка серій,
- перевірка на монотонність,
- автокореляційна функція,
- графічний спектральний тест.

Статистичні тести

Статистичні тести використовуються для перевірки певної нульової гіпотези H_0 щодо випадковості сформованої послідовності. З цією нульовою гіпотезою пов'язана альтернативна гіпотеза На про те, що послідовність не випадкова. Для кожного тесту, що застосовується, можна зробити висновок щодо прийняття чи відхилення нульової гіпотези, виходячи із сформованої генератором послідовності. При цьому для кожного тесту має бути вибрана адекватна статистика випадковості, на основі якої може бути прийнята або відхилена нульова гіпотеза. Теоретично для нульової гіпотези розподілення статистики визначається математичними методами. Під час проведення тесту розраховується значення тестової статистики, яке порівнюється з критичним. Якщо значення тестової статистики перевищує критичне значення, нульова гіпотеза для випадковості відхиляється, інакше – приймається.

Статистичні тести, на відміну від графічних тестів (результати інтерпретуються користувачами, внаслідок чого можливі відмінності у трактуванні результатів), характеризуються тим, що вони видають чисельну характеристику, яка дозволяє однозначно сказати, пройдено тест чи ні.

Найбільш відомі на сьогоднішній день методики тестування :

- методика тестів Дональда Кнута;
- система статистичного тестування DIEHARD;
- методика NIST STS.

Статистичні тести NIST STS (National Institute of Standards and Technology Statistical Test Suite) використовуються для визначення якісних та кількісних ознак випадковості послідовності чисел. Сьогодні ця методика є найбільш поширеною серед розробників криптографічних засобів захисту інформації. Пакет NIST STS включає в себе 16 статистичних тестів, які розроблені для перевірки гіпотези про випадковість двійкових послідовностей довільної довжини, породжуваних ГПВП. Всі тести спрямовані на виявлення різних дефектів випадковості. Кожен тест подається у розрізі мети, позначення, статистик, опису та правил інтерпретації результатів [15].

12. АСИМЕТРИЧНІ КРИПТОСИСТЕМИ

12.1. Основні поняття

Основна проблема симетричних (закритих) криптосистем – розподіл ключа серед учасників інформаційного обміну. Тому для розповсюдження ключів необхідно використовувати іншу криптосистему або захищений інформаційний канал. Для розв'язання цієї проблеми на основі нових результатів сучасної алгебри запропоновано систему з відкритим ключем (асиметричні криптосистеми). – 70-ті роки.

Асиметричні криптосистеми – ефективні системи криптографічного захисту даних, які також називають криптосистемами з відкритим ключем. В таких системах для зашифровування даних використовується один ключ, а для розшифровування – інший ключ (звідси і назва – асиметричні). Перший ключ є відкритим і може бути опублікованим для використання усіма користувачами системи, які шифрують дані. Розшифровування даних за допомогою відкритого ключа неможливе. Для розшифровування даних отримувач зашифрованої інформації використовує другий ключ, який є секретним. Зрозуміло, що ключ розшифровування не може бути визначеним з ключа зашифровування.

Головне досягнення асиметричного шифрування в тому, що воно дозволяє людям, що не мають наперед наявної домовленості про безпеку, обмінюватися секретними повідомленнями. Необхідність відправникові й одержувачеві погоджувати таємний ключ по спеціальному захищеному каналі цілком відпала. Прикладами криптосистем з відкритим ключем є ElGamal, RSA, Diffie-Hellman і DSA.

Вважається, що ця систем народилася у 1976 році, коли була опублікована стаття американських математиків У. Діффі і М. Хеллмана «Нові напрями у криптографії», в якій вони ввели поняття однобічної (односпрямованої) функції. Перебуваючи під впливом роботи Ральфа Меркле (Ralph Merkle) про поширення відкритого ключа, вони запропонували метод отримання секретних ключів, використовуючи відкритий канал. Цей метод експоненціального обміну ключів, який став відомий як обмін ключами Діффі-Хеллмана, був першим опублікованим практичним методом для встановлення поділу секретного ключа між завіреними користувачами каналу. У 2002 році Хеллман запропонував називати даний алгоритм «Діффі – Хеллмана – Меркле», визнаючи внесок Меркле в винахід криптографії з відкритим ключем. Ця ж схема була розроблена Малькольмом Вільямсоном в 1970-х, але трималася в секреті до 1997 року. Метод Меркле з розповсюдження відкритого ключа був винайдений в 1974 році і опублікований в 1978, його також називають загадкою Меркле.

У 1977 році вченими Рональдом Рівестом (Ronald Linn Rivest), Аді Шамір (Adi Shamir) і Леонардом Адлеманом (Leonard Adleman) з Массачусетського Технологічного Інституту (MIT) був розроблений алгоритм шифрування, заснований на проблемі про розкладанні на множники. Система була названа за першими літерами їхніх прізвищ. Ця ж система була винайдена Клиффордом

Коксом (Clifford Cocks) в 1973 році, що працював в центрі урядового зв'язку (GCHQ). Але ця робота зберігалася лише у внутрішніх документах центру, тому про її існування було не відомо до 1977 року. RSA став першим алгоритмом, придатним і для шифрування, і для цифрового підпису.

Проблема керування ключами була вирішена криптографією з відкритим, або асиметричним, ключем, концепція якої була запропонована Уітфілдом Діффі і Мартіном Хеллманом у 1975 році. Криптографія з відкритим ключем — це асиметрична схема, у якій застосовуються пари ключів: відкритий (public key), що зашифровує дані, і відповідний йому закритий (private key), що їх розшифровує. Ключова пара математично зв'язана, обчислення закритого ключа з відкритого в практичному плані нездійсненна, це досягається за допомогою використання односторонніх функцій. Кожний, у кого є відкритий ключ, зможе зашифрувати дані, але не зможе їх розшифрувати. Тільки людина, яка володіє відповідним закритим ключем може розшифрувати інформацію.

Функція $F(x)$ називається односторонньою (важкооборотною), якщо:

– існує поліноміальний алгоритм обчислення значення функції $F(x)$ для $\forall x \in X$.

– не існує поліноміальний алгоритм обчислення значення оберненої $F^{-1}(x)$, тобто встановити значення аргументу x за відомим значенням y неможливо.

Зауваження: (необхідна досить велика область x, бо коли мала – то можна здійснити метод перебору $\forall x \in X$).

Існує велика кількість таких функцій. Розглянемо деякі з них.

1) Цілочислове множення двох великих чисел: $n = p \cdot q$ – пряма задача. Обернена – розклад числа N на множники. При цілому $n = 2^{664}$ – необхідно 10^{23} операцій, якщо $p \approx q$.

2) Експонента з фіксованою основою і модулем: $f_{A,n}(x) = A^x (\mathrm{mod}\, n)$, де $1 \le x \le n-1$, $1 \le A \le n-1$ – пряма задача. Обернена задача дискретного логарифмування – $a^x = y \Rightarrow x = \log_a y$. При $A = 2^{664}, n = 2^{664} \to 10^{26}$ операцій.

До асиметричних криптосистем ставляться важливі вимоги:

– перетворення відкритого тексту повинно бути незворотнім без можливості його відновлення на публічному ключі;

– обчислення приватного ключа на основі публічного також має бути неможливим на середньому технічному рівні.

Алгоритми шифрування з відкритим ключем застосовують у трьох напрямках:

1) як засоби захисту інформації;
2) як засоби аутентифікації користувача;
3) як засоби розповсюдження ключів.

Як відомо, дані системи повільніші за симетричні. Тому на практиці – асиметричні використовують для шифрування невеликої інформації, симетричні – для великих потоків [9].

12.2. Елементи теорії чисел

12.2.1. Конгруентність

Нехай n – довільне натуральне число, x, y – цілі числа. Будемо називати x і y конгруентними за модулем n, якщо залишки від їх ділення на число n одинакові, тобто: $x \bmod n \equiv y \bmod n$ (наприклад, $2 \bmod 7 \equiv 9 \bmod 7$) [9].

Операція $(\bmod n)$ має такі властивості:

– адитивності: $(a+b) \bmod n = ((a \bmod n)+(b \bmod n)) \bmod n$;

– мультиплікативності: $(a \times b) \bmod n = ((a \bmod n) \times (b \bmod n)) \bmod n$;

– збереження степеня: $(a \bmod n)^k \bmod n = a^k \bmod n$.

12.2.2. Алгоритм Евкліда для розв'язання рівняння $ax = 1 \bmod n$

Представимо: $\frac{a}{n} = \frac{Pk}{Qk}$, причому $P_{-2} = 0,\ P_{-1} = 1,\ Q_{-2} = 1,\ Q_{-1} = 0.$

Для $\forall k \geq 0,\ P_k = q_k P_{k-1} + P_{k-2};\quad Q_k = q_k Q_{k-1} + Q_{k-2}$.

Для обчислення P і Q складається таблиця 12.1.

Таблиця12. 1

Обчислення P і Q

n	-2	-1	0	1	2	3	...	k_{-1}	k
q_n			q_0	q_1	q_2	q_3	...	q_{k-1}	q_k
P_n	0	1	P_0	P_1	P_2	P_3	...	P_{k-1}	P_k
Q_n	1	0	Q_0	Q_1	Q_2	Q_3	...	Q_{k-1}	Q_k

З таблиці отримуємо $\boxed{x = (-1)^{k-1} Q_{k-1}}$ і $y = (-1)^{k-1} P_{k-1}$, при $P_k = a,\ Q_k = n$.

Для обчислення значень таблиці використовують ланцюжкові дроби.

Приклад 1 Розв'язати рівняння $32 \cdot x \bmod 29 = 1$.

$32 = 29 \times \underline{1} + 3 \qquad q_0 = 1$

$29 = 3 \times \underline{9} + 2 \qquad q_1 = 9$

$3 = 2 \times \underline{1} + 1 \qquad q_2 = 1$

$2 = 1 \times \underline{2} + 0 \qquad q_3 = 2$

Таблиця 12.2

Обчислення P і Q

n	-2	-1	0	1	2	3	
q_n			1	9	1	2	
P_n	0	1	1	10	11	32	$P_k = a$
Q_n	1	0	1	9	10	29	$Q_k = b$

Отже, $x = (-1)^2 \cdot 10 = +10;\ y = (-1)^2 \cdot 1 = 11.\ 32 \cdot 10 \bmod 29 = 320 \bmod 29 = 1,\ 32^{-1} = 10$.

12.2.3. Ланцюгові дроби

У схемі асиметричної криптографії використовуються алгебраїчні порівняння.

$НСД(a,n) = 1$.

$\frac{P_k}{q_k} = \frac{n}{a} \Rightarrow \begin{matrix} p_k = n \\ q_k = a \end{matrix}$, $\frac{P_k}{q_k}$ – знаходиться з рекурсивних співвідношень.

$p_k \cdot q_{k-1} - p_{k1} \cdot q_k = (-1)^{k-1}$, $n \cdot q_{k-1} - a \cdot p_{k-1} = (-1)^{k-1}$, $a \cdot p_{k-1} = (-1)^k + n \cdot q_{k-1} / (\mathrm{mod}\, n)$

$a \cdot p_{k-1} = (-1)^k \ (\mathrm{mod}\, n) / \times b(-1)^k$, $\underbrace{a \cdot b(-1)^k \cdot p_{k-1}}_{a \cdot x = b \bmod n} = b \bmod n$ $\boxed{x = (-1)^k \cdot b \cdot p_{k-1} \bmod n}$.

Якщо $\frac{p}{q}$ – нескоротний, то його можна представити у вигляді ланцюгових дробів:

$\frac{p}{q} = a_0 + \frac{r_1}{q}$, $0 < r_1 < q$,

$\frac{q}{r_1} = a_1 + \frac{r_2}{r_1}$, $0 < r_2 < r_1$

…

$\frac{r_{n-1}}{r_n} = a_n$ $\frac{p}{q} = [a_0, a_1, ..., a_n]$ – правильний скінченний ланцюговий дріб

$\frac{p}{q} = a_0 + \cfrac{1}{a_1 + \cfrac{1}{a_2 + \cfrac{1}{\ldots + \cfrac{1}{a_n}}}}$ $\frac{p_0}{q_0}, \frac{p_1}{q_1}, ..., \frac{p_n}{q_n} = \frac{m}{a}$ – підхідні дроби

$$\begin{cases} \frac{p_0}{q_0} = a_0, \frac{p_1}{q_1} = [a_0, a_1].. \\ \frac{p_k}{q_k} = [a_0, a_1, ..., a_k] \\ \begin{cases} p_k = a_k p_{k-1} + p_{k-2} \\ q_k = a_k q_{k-1} + q_{k-2} \end{cases}, k \ge 2 \end{cases} \quad \begin{matrix} p_{-2} = 0 \\ p_{-1} = 1 \\ q_{-2} = 1 \\ q_{-1} = 0 \end{matrix},$$

Приклад 2 Розв’язати рівняння $31x = 19 (\mathrm{mod} 83)$, $НСД(31,83) = 1$.

$\frac{83}{31} = \underline{2} + \frac{21}{31}$ $\frac{31}{21} = \underline{1} + \frac{10}{21}$ $\frac{21}{10} = \underline{2} + \frac{1}{10}$ $\frac{10}{1} = \underline{10}$

Отже, $\frac{83}{31} = [2,1,2,10] = 2 + \cfrac{1}{1 + \cfrac{1}{2 + \cfrac{1}{10}}}$

Таблиця 3

Обчислення p і a

$к$	-2	-1	0	1	2	3
a_k	-	-	2	1	2	10
p_k	0	1	2	3	8	83

$x = (-1)^3 \cdot 19 \cdot 8 (\mathrm{mod} 83) = -152 \bmod 83 = 14$.

12.2.4. Функція Ейлера

$НСД(a,n) = d$

Якщо *b* не ділиться на *d*, то розв'язків нема.

Якщо *b* кратне *d*, тоді рівняння має рівно *d* розв'язків, а саме x_0, $x_0 + \frac{n}{d}, x_0 + 2\frac{n}{d}, \ldots, x_0 + (d_{-1})\frac{n}{d}$, де x_0 – розв'язок із порівнянням: $\frac{a}{d}x = \frac{b}{d}(\mathrm{mod}\frac{n}{d})$.

Якщо *d=1*, то існує один розв'язок, який знаходимо за методом Ейлера $\boxed{x \equiv b \cdot a^{\varphi(n)-1} \bmod n}$.

Властивості функції:

1) $\varphi(p) = p - 1$, *p* – просте число;
2) $\varphi(m_1 \cdot m_2) = \varphi(m_1) \cdot \varphi(m_2)$, m_1, m_2 – прості числа;
3) $\varphi(p^\alpha) = p^\alpha(1 - \frac{1}{p})$, *p* – просте число, $\alpha > 0$;
4) $\varphi(m) = \varphi(m_1 \ldots m_n) = m\prod_{i=1}^{n}(1 - \frac{1}{p_i})$, де $m_i = p_1^{\alpha_1} \ldots p_n^{\alpha_n}$.

Приклад 3 Розв'язати рівняння $7x = 5(\mathrm{mod}9)$.

$НСД(7,9) = 1$, отже, існує один розв'язок $x = b \cdot a^{\varphi(m)-1} \bmod m$, обчислимо функцію Ейлера $\varphi(n) = \varphi(9) = \varphi(3^2) = 9 \cdot (1 - \frac{1}{3}) = 9 \cdot \frac{2}{3} = 6$.

Тому $x = 5 \cdot 7^{6-1} \bmod 9 = 2$ $x = 5 \cdot 7^{6-1} \bmod 9 = 2$.

Приклад 4 Розв'язати рівняння $9x = 7(\mathrm{mod}10)$.

$НСД(9,10) = 1$, $\varphi(10) = \varphi(5 \cdot 2) = \varphi \cdot 1 = \varphi$, $x = 7 \cdot 9^3 \bmod 10 = 5103 \bmod 10 = 3$.

Зауваження

Поняття найбільшого спільного дільника двох чисел (НСД), простого числа, генерації простих чисел та перевірка на простоту чисел широко висвітлені у літературі.

12.3. Алгоритм RSA

Як уже зазначалось, у 1977 році був розроблений алгоритм шифрування, перший повноцінний алгоритм з відкритим ключем, який може бути використаний як в режимі шифрування, так і в режимі цифрового електронного підпису. Надійність заснована на складності факторизації великих чисел [9].

Крок 1. Підготовчі обчислення.

Отримувач генерує два прості числа *p* і *q* (> 128 біт), знаходимо $n = p \cdot q$. Генеруємо число e, взаємно просте із n та $\varphi(n)$. $НСД(e,n) = 1, НСД(e,\varphi(n)) = 1$. Пара (e,n) – публічний ключ. Для отримання приватного ключа *d* необхідно розв'язати рівняння: $(d \cdot e) \bmod \varphi(n)$. Пара (d,n) – приватний ключ.

Крок 2. Розповсюдження ключів.

Для шифрування інформації використовують публічний ключ (хоч можна використати і приватний). Для використання його розміщують на ресурсі, де до нього мають доступ всі учасники обміну інформації. На відміну від

симетричного алгоритму не потрібно для кожної пари учасників генерувати секретний ключ окремо. Єдине, що необхідно зробити – захистити його від підміни. Найпростіше – захистити каталог, де знаходяться ключі, від запису, однак найнадійнішим способом вважається сертифікація ключів. Кожен хто хоче захистити публічний ключ від підміни повинен отримати сертифікат довірчого центру інфраструктур відкритих ключів, який прив'язує ключ до власника. Приватний ключ не розповсюджується. Він використовується для розшифрування та створення ЕЦП і повинен бути відомим лише власнику.

Крок 3. Шифрування інформації на ключі (*e*,*n*): вихідний текст поділяється на блоки довжиною $M_i = \overline{0, N-1}$. Формула шифрування: $C_i = (M_i)^e \bmod n$, де M_i – числове повідомлення чергової літери, C_i – символ криптограми.

Крок 4. Розшифрування інформації на ключі (*d*,*n*): $M_i = (C_i)^d \bmod n$.

Приклад

Необхідно зашифрувати слово «БАНК», яке представимо у числовому вигляді «02 01 17 14».

Для шифрування сформуємо ключі:

$p = 7, q = 11$

$p \cdot q = 77$.,

$\varphi(n) = \varphi(77) = \varphi(7) \cdot \varphi(11) = 6 \cdot 10 = 60$

$(e,n) = (13,77)$ – відкритий ключ, $(d \cdot 13) \bmod 60 = 1, d = 37$, $(37,77)$ – секретний ключ.

$C_1 = 2^{13} \bmod 77 = 30;\ C_2 = 1^{13} \bmod 77 = 1 \quad C_3 = 17^{13} \bmod 77 = 73;\ C_4 = 14^{13} \bmod 77 = 49.$

Отже, отримаємо шифр : «30 01 73 49».

Зашифрований текст пересилається користувачу відкритими каналами.

Розшифрування здійснюється за допомогою секретного ключа $(d,n) = (37,77)$.

$M_1 = 30^{37} \bmod 77 = 2;\quad M_2 = 01^{37} \bmod 77 = 1;\quad M_3 = 73^{37} \bmod 77 = 17;\ M_4 = 49^{37} \bmod 77 = 14$

Отримаємо вихідний текст : «02 01 17 14» –> «БАНК».

Криптостійкість RSA

Задача криптоаналізу для RSA еквівалентна задачі розкладання великого числа *n* на прості множники.

Таблиця 12.4

Часові затрати на розкладання числа *n* на прості множники

Кількість *n*	Значення функції $\alpha(n)$	Кількість MIDS років
512	$6.7 \cdot 10^{19}$	$2.1 \cdot 10^{6}$
576	$1.7 \cdot 10^{21}$	$5.5 \cdot 10^{7}$
960	$3.7 \cdot 10^{28}$	$1.2 \cdot 10^{15}$
1024	$4.4 \cdot 10^{29}$	$1.4 \cdot 10^{16}$

1 MIDS рік дорівнює 1млн. інструкцій за 1 с протягом року, що становить $3.1 \cdot 10^{13}$ інструкцій. Функція $\alpha(n)$ – апроксимація швидкості найкращого алгоритму розкладання чисел на прості множники:

$\alpha(n) = \exp(1+\varepsilon)\cdot\sqrt{(\ln n)\cdot(\ln\ln n)^2}$, де n – кількість двійкових розрядів у числі, ε – нескінчено мала величина. Є інші алгоритми, де $\alpha(n) = e^{2(\ln n)^{1/3}}(\ln\ln n)^{2/3}$ [9].

Автори RSA рекомендують використовувати таку довжину модуля *n:* 768 біт – приватні особи; 1024 біт – для комерційної інформації; 2048 біт – особливо таємної інформації.

Криптоалгоритм RSA реалізований програмно і апаратно (спеціальні процесори). Однак, апаратна реалізація RSA в 1000 раз повільніша ніж апаратна реалізація DES. Програмна реалізація RSA в 100 раз повільніша ніж програмна реалізація DES.

12.4. Схема шифрування Ель-Гамаля

Ця схема запропонована в 1985 році. Її безпека обумовлена складністю обчислень дискретних логарифмів у скінченому колі.

Для реалізації алгоритму вибирають будь-яке велике просте число P (512 – 1024 біт) і велике ціле число $G < P$. Ці числа розповсюджують серед групи користувачів. Вибирають будь-яке випадкове ціле число $X \subset P$ (секретний ключ). Обчислюють $Y = G^X \bmod P$ (відкритий ключ).

Для зашифрування повідомлення M вибирають будь-яке випадкове число $1 < K < P-1$, так щоб $P-1$ і K були взаємно простими $a = G^k \bmod P$, $b = Y^k M \bmod P$. Пара чисел (a,b) є шифротекстом. Довжина шифротексту вдвічі більша за вихідний текст.

Для розшифрування шифру використовують порівняння: $M \cdot a^x = b \bmod P$.

Приклад

Шифрування: Нехай $M = 5$ (повідомлення), $P = 11, G = 2$ секретний ключ $X = 8 < P$.

Обчислимо $Y = G^X \bmod P = 2^8 \bmod 11 = 3 \Rightarrow Y = 3$- відкритий ключ. Вибираємо будь-яке $K < 10$, $K = 9$, $HOD(9,10) = 1$. Обчислимо $a = G^k \bmod P = 2^9 \bmod 11 = 6$, $b = Y^k M \bmod P = 3^9 \cdot 5 \bmod 11 = 9$. Зашифрований текст – $(a,b) = (6,9)$.

Розшифрування: $M \cdot a^x = b \bmod P$ => $M \cdot 6^8 = 9 \bmod 11$ => $M \cdot 1679616 = 9 \bmod 11$ => $M = 5$.

Комбіновані методи

Головне в криптографії з відкритим ключем – потенційно велика безпека. Однак є недоліки: генерація ключів заснована на генерацій нових великих простих чисел, а перевірка на ці властивості займає багато процесорного часу; процес шифрування і дешифрування громіздкі.

Криптосистема з відкритим ключем застосовується для шифрування і передачі та подальшого розшифрування секретного ключа симетричної криптосистеми. Симетрична криптосистема використовується для шифрування і розшифрування самого вихідного відкритого тексту. Асиметрична криптосистема доповнює симетричну криптосистему, підвищує її стійкість.

13. ЕЛЕКТРОННИЙ ЦИФРОВИЙ ПІДПИС

13.1. Проблема аутентифікації даних і електронний цифровий підпис

При обміні електронними документами по мережі зв'язку істотно знижуються витрати на обробку і збереження документів, збільшується швидкість їхнього пошуку. Але при цьому виникає проблема аутентифікації автора документа і самого документа, тобто встановлення дійсності автора і відсутності змін в отриманому документі. У звичайній (паперовій) інформатиці ці проблеми вирішуються за рахунок того, що інформація в документі і рукописному підписі автора жорстко зв'язана з фізичним носієм (папером). В електронних документах на машинних носіях такого зв'язку немає.

Метою аутентифікації електронних документів є їхній захист від можливих видів злочинних дій, до яких відносяться:

– активний перехоплювач-порушник, що підключився до мережі, перехоплює документи (файли) і змінює їх;

– маскарад – абонент *C* посилає документ абонентові *B* від імені абонента *A*;

– ренегатство – абонент *A* заявляє, що не посилав повідомлення абонентові *B*, хоча насправді послав;

– підміна – абонент *B* змінює або формує новий документ і заявляє, що одержав його від абонента *A*;

– повтор – абонент *C* повторює раніше переданий документ, що абонент *A* посилав абонентові *B*.

Ці види злочинних дій можуть завдати істотної шкоди банківським і комерційним структурам, державним підприємствам і організаціям, приватним особам, що застосовують у своїй діяльності комп'ютерні інформаційні технології.

При обробці документів в електронній формі зовсім непридатні традиційні способи встановлення дійсності по рукописному підписі і відбиткові печатки на паперовому документі. Принципово новим рішенням є електронний цифровий підпис (ЕЦП). Цей механізм є винайденням асиметричної криптографії. У. Діффі та М. Хеллман назвали його цифровим підписом.

Електронний цифровий підпис використовується для аутентифікації текстів, переданих по телекомунікаційних каналах. Функціонально вона аналогічна звичайному рукописному підпису і володіє її основними первагами:

1) засвідчує, що підписаний текст виходить від імені, що поставив підпис;

2) не дає самій цій особі можливості відмовитися від зобов'язань, зв'язаних з підписаним текстом;

3) гарантує цілісність підписаного тексту.

Цифровий підпис являє собою відносно невелику кількість додаткової цифрової інформації, переданої разом з текстом, що підписується.

Система ЕЦП включає дві процедури: 1) процедуру постановки підпису; 2) процедуру перевірки підпису. У процедурі постановки підпису використовується секретний ключ відправника повідомлення, у процедурі перевірки підпису – відкритий ключ відправника.

До повідомлення *M* застосовують перетворення за допомогою секретного ключа, що і назвемо цифровим підписом, тобто $S = M^d \bmod n$. Повідомлення M і S

відправляють за призначенням. Отримувач, маючи пару *M* і *S* та відкритий ключ відправника, може перевірити виконання співвідношення $S^e \bmod n = M$.

Така схема приводить до такого:

– отримувач, перевіривши справжність підпису, впевнений в тому, що це повідомлення сформував саме власник приватного ключа;

– відправник не зможе відмовитися від цього листа з тієї ж причини.

Ця схема має суттєвий недолік. ЕЦП має таку ж довжину, що і повідомлення, ним підписане. Отже, каналом зв'язку передається вдвічі більше інформації, ніж потрібно для самого документа. Таким чином, необхідна видозміна цієї схеми – хешування. При формуванні ЕЦП відправник насамперед обчислює хэш-функцию тексту, що підписується – *h(M)*. Обчислене значення *h(M)* це один короткий блок інформації *m*, що характеризує весь текст *M* в цілому. Потім число *m* шифрується секретним ключем відправника. Одержувана при цьому пара чисел є ЕЦП для даного тексту *M*. При перевірці ЕЦП одержувач повідомлення знову обчислює хэш-функцію *m=h(M)* прийнятого по каналу тексту *M*, після чого за допомогою відкритого ключа відправника перевіряє, чи відповідає отриманий підпис обчисленому значенню *m* хэш-функції.

Принциповим моментом у системі ЕЦП є неможливість підробки ЕЦП користувача без знання його секретного ключа підписування.

В якості підписуваного документу може бути використаний будь-який файл. Підписаний файл створюється з непідписаного шляхом додавання в нього однієї або більше електронних підписів.

Кожен підпис містить наступну інформацію: дату підпису; термін закінчення дії ключа даного підпису; інформацію про особу, що підписала файл (П.І.Б. посада, коротке найменування фірми тощо); ідентифікатор, що підписав (ім'я відкритого ключа); власне цифровий підпис [9; 10].

13.2. Хеш-функція

Хеш-функцією назвемо криптографічне перетворення двійкової послідовності довільної довжини у двійкову послідовність фіксованої довжини (стиснення до десятків, сотень біт) [9].

Хеш-функцію використовують для створення стиснутого повідомлення в ЦЕП; захисту паролів; побудови коду аутентифікації та інше.

Вимоги до хеш-функції:

- Хеш-функція повинна бути чутлива до змін в тексті *M* (вставки, перестановки).
- Хеш-функція повинна мати властивість не оберненості, тобто за відомим значенням *h(M)* неможливо визначити *M*.
- Ймовірність того, що значення хеш-функції двох різних документів співпаде, повинна бути дуже мала $h(M') \neq h(M)$.
- Для практичного застосування алгоритми отримування хеш-функції повинні бути застосовані під конкретну апаратуру.

У випадку використання хеш-функції схема цифрового підпису така:

1) Спочатку обчислюють *h(M)*, потім застосовують криптографічне перетворення, наприклад RSA: $S = [h(M)]^d \bmod n$, передача (M,S).

2) На приймаючій стороні $S^e \bmod n = h(M)$, вироблення $h(M')$ із переданого повідомлення *M*. Якщо $h(M') = h(M)$, то отримувач впевнений:

– повідомлення *M* не зазнало ніяких змін під час його передачі по каналу зв'язку;

– *h(M)* – теж не змінилося;

– повідомлення написано власником приватного ключа *d*;

– відправник не зможе відмовитись від факту існування такого документу.

Кінцева довжина хеш-образу від 128 до 256 біт.

13.2.1. Алгоритми хешування сімейства MD

Message Digest – згортка повідомлень, була розроблена Роном Рівестом і утворює на виході 128-бітні образи.

Вимоги до хеш-алгоритмів:

– Безпека: обчислювально неможливо знайти 2 повідомлення з однаковими образами.

– Пряма безпека: Безпека алгоритмів не ґрунтується на жодних припущеннях, наприклад на розкладі числа на прості множники.

– Швидкість: розраховані на швидкісну програмну реалізацію.

– Простота і компактність.

Архітектура: алгоритми оптимізовані для мікропроцесорної архітектури, для більш складних процесорів потрібно внести зміни.

Основні операції: [+] – додавання за модулем 2^{32}, $\oplus$ – додавання за модулем 2 [5].

Алгоритм MD2

1989 р – для 8-розрядних процесорів. На виході 128 біт згортки.

1. Повідомлення доповнюються *i*-бітами, щоб довжина була кратна 16-ти байтам.
2. До повідомлення + 16 байт контрольної суми.
3. Ініціалізується 48-байтний блок: $x_0, x_1, .., x_{47}$, де перші 16 байт $x_0 .. x_{15}$ заповнюються нулями; другі 16 байт $x_{16} .. x_{31}$ – копіюються перші 16 байт повідомлення. треті 16 байт рівні сумі за модулем 2 перших і других 16 байт.
4. Функція стиску виглядає так

t=0
for j=0 to 17
for k=0 to 47
$t = x_t \text{ xor } S_t$
$x_k = t$
t=(t+j)mod 256

S_t – 256 байт випадкового перемішування з цифр числа π (перестановка).

5. Наступні 16 байт копіюються у другі 16 байт блоку, а треті 16 байт утворюються за допомогою *xor* перших і других.

6. Етапи 4-5 повторюються до закінчення відкритого тексту.

7. Вихід – перші 16 байт блоках.

Цей алгоритм є досить повільним у порівнянні з іншими аналогічними алгоритмами і явних недоліків немає.

Алгоритми MD-4 і MD-5

На виході 128 біт. Алгоритми обробляють повідомлення блоками по 512 біт, які розбивають на шістнадцять 32-бітних блоки. Ці алгоритми мало відрізняються, MD-5 є вдосконаленням MD-4.

1. Повідомлення доповняється до величини, що менше на 64 біти за число кратне 512 бітам. Доповнюється 100... До результату додається 64-бітне представлення довжини повідомлення. Якщо довжина повідомлення більша 2^{64}, то використовуються молодші розряди (біти) довжини.

2. Ініціалізують чотири 32-розрядні регістри, які заповнюються початковими значеннями: *a*=01234567, *b*=89abcdef, *c*=fedcba98, *d*=76543210 – змінні значення.

3. Утворення хеш-образу складається з 4-ох етапів (у MD-4 з трьох), кожен з них містить 16 циклів, що використовують 4 нелінійні функції:

$F(x,y,z) = xy\ or\ \overline{xz}$; $G(x,y,z) = xz\ or\ y\bar{z}$; $H(x,y,z) = x\ xor\ y\ xor\ z$; $I(x,y,z) = x\ or\ y\ or\ \bar{z}$.

Процедура обробки:

$FF(a,b,c,d,M_j,S,t_i) = a := b[+]sh_S(F(b,c,d)[+]M_j[+]t_i)$;

$GG(a,b,c,d,M_j,S,t_i) = a := b[+]sh_S(G(b,c,d)[+]M_j[+]t_i)$;

$HH(a,b,c,d,M_j,S,t_i) = a := b[+]sh_S(H(b,c,d)[+]M_j[+]t_i)$;

$II(a,b,c,d,M_j,S,t_i) = a := b[+]sh_S(I(b,c,d)[+]M_j[+]t_i)$.

Тут M_j – 32-бітний блок тексту, t_i – ціла частина $\left[2^{32}\cdot|\sin i|\right]$, [+] – додавання за модуем 2^{32}, sh_s – зсув вліво на *S* позицій.

4. Значення хеш-образу накопичується в регістрах *a,b,c,d*.

Етап 1

FF(a,b,c,d,M0,7,t1), FF(d,a,b,c,M1,12,t2), FF(c,d,a,b,M2,17,t3), FF(b,c,d,a,M3,22,t4), …, FF(b,c,d,a,M15,22,t16).

Етап 2 виконується аналогічно з використанням функції *GG*, послід 5,9,14,20 і в і-му циклі t_{16+i} замість t_i, $M_{(1+5\cdot i)mod16}$ замість M_i, i=1..16.

Етап 3 виконується аналогічно з використанням функції *HH*, послід 4,11,16,23 і в і-му циклі t_{32+i} замість t_i, $M_{(32+3\cdot i)mod16}$ замість M_i, i=1..16.

Етап 4 виконується аналогічно з використанням функції *II*, послід 6,10,15,21 і в і-му циклі t_{48+i} замість t_i, $M_{(3+7\cdot i)mod16}$ замість M_i, i=1..16.

5. Остаточне значення хеш-образу накопичується в регістрах *a,b,c,d* і утворюється їх конкатенація (4x32=128 біт).

Різниця між MD-4 і MD-5:

– У MD-5 – 4 етапи, MD-4 – 3 етапи.

– У MD-5 у кожному циклі використовується унікальна константа, яка додається до результату t_i.

– Функція *G* на 2-ому етапі замінюється: $xy\ or\ xz\ xor\ yz$ на $xz\ or\ y\bar{z}$ (для досягнення меншої симетрії).

– У MD-5 результат кожного етапу додається до результатів попереднього етапу(лавинний ефект).

– Значення зміщення оптимізовано для досягнення лавинного ефекту.

13.2.2. Алгоритм SHA-1

Secure Hash Algorithm використовувався як стандарт в США із 1992 р. іаналогічний як MD-4, але на виході отримуємо 160-бітний образ.

1. Повідомлення доповнюється до величини меншої на 64 біт, до кратної 512 біт + 64 біти довжини повідомлення.

2. П'ять регістрів: *a*=67452301; *b*=efcdab89; *c*=98badcfe; *d*=10325476; *e*=c3d2e1f0.

3. Обробляються блоки тексту по 512 біт у 4 етапи по 20 операцій.

$F_i(x,y,z) = xy\ or\ \bar{x}z, i = \overline{0,19}$; $F_i(x,y,z) = x\ xor\ y\ xor\ z, i = \overline{20,39}$;

$F_i(x,y,z) = xy\ or\ xz\ or\ yz, i = \overline{40,59}$; $F_i(x,y,z) = x\ xor\ y\ xor\ z, i = \overline{60,79}$

512 бітний блок повідомлення перетворюється з 16-ти 32-бітних блоків (M_0-M_{15}) в 80 32-бітні слова (W_0-W_{79}).

$W_i = M_i, i = \overline{0,15}$; $W_i = sh_1(W_{i-3}\ \ xor\ \ W_{i-8}W_{i-14}W_{i-16}), i = \overline{16,79}$

4. Головний цикл:

$for\ i = 0\ to\ 79$

$Temp = sh_5(a)[+]F_i(b,c,d)[+]e[+]W_i[+]K_i$,

$e = d; d = c; c = sh_{30}(b); b = a; a = Temp$

$K_i = 5a827999, i = \overline{0,19};$

$K_i = 6ed9eba1, i = \overline{20,39};$

$K_i = 8f1bbcdc, i = \overline{40,59};$

$K_i = ca62c1db, i = \overline{60,79}.$

5. У кінці кожного циклу отримані *a,b,c,d,e* додають до відповідних початкових даних. Остаточне значення – конкатенація *a,b,c,d,e*.

Головна відмінність з MD-4 – наявність розширювального перетворення та додавання результатів обробки до результату попередніх циклів (як в MD-5), чого нема в MD-4.

Порівняння з MD-5: На 4-му етапі функція 2-го етапу; у MD-5 в кожному циклі використовується унікальна константа, а в $SHA(K_i)$; F_2 така як в MD-4, в MD-5 замінена для кращої асиметрії; постійне значення зсувів, як в MD-4.

13.2.3 RIPEMD (Race Integrity Primitives Evaluation)

RIPEMD-128,160

Розроблений в рамках європейського проекту RIPE, включає в себе паралельні версії MD-4, що відрізняються константами. Для RIPEMD -160 (використовуються функції *mod2*, $mod2^{32}$,*or*,*and*).

13.2.4. Хеш-функція заснована на швидкому перетворенні Фур'є

Вона була запропонована в 1991 році Шнорром і у 1992 році ним же вдосконалена. Повідомлення довільної довжини перетворюється в 128 бітний

стиснений образ. Операції: Дискретне перетворення Фур'є та рекурсія поліномів.

13.2.5 Вітчизняний ГОСТ Р-34.11-94 (на основі блочного шифрування ГОСТ-89)

Повідомлення довільної довжини перетворюється в 256 бітний образ.

Використовується функція стиснення $H_i = f(M_i, H_{i-1})$, де M_i, H_i – 256-бітні величини.

1) Генеруються чотири ключі шифрування K_j, $j = \overline{1,4}$ шляхом лінійного змішування M_i, H_{i-1} і деяких констант C_j

2) Кожен ключ K_j використовують для шифрування 64 біт підслів h_i слова H_{i-1} у режимі простої заміни: $S_j = E_{K_j}(h_j)$ отримуємо $S_4, S_3, S_2 S_1$ – послідовність довжиною 256 біт – тимчасова заміна.

3) H_i – лінійне змішування M_i, H_{i-1}, S.

4) $H = f(Z \; xor \; M', f(L, f(M', H_n)))$ – кінцеве значення.

Z – значення контрольної суми, яка одержується після *mod2* (чи *xor*) всіх блоків повідомлення, H_n – хеш-значення останнього блоку, L – довжина повідомлення, M' – доповнений останній блок.

13.3. Алгоритми електронного цифрового підпису

Технологія застосування системи ЕЦП припускає наявність мережі абонентів, що посилають один одному підписані електронні документи. Для кожного абонента генерується пара ключів: секретний і відкритий. Секретний ключ зберігається абонентом у таємниці і використовується ним для формування ЕЦП. Відкритий ключ відомий всім іншим користувачам і призначений для перевірки ЕЦП одержувачем підписаного електронного документа. Інакше кажучи, відкритий ключ є необхідним інструментом, що дозволяє перевірити дійсність електронного документа й автора підпису. Відкритий ключ не дозволяє обчислити секретний ключ.

13.3.1.Алгоритм цифрового підпису RSA

Першою і найбільш відомою в усьому світі системою ЕЦП стала система RSA, математична схема якої була розроблена в 1977 р. у Массачуссетському технологічному інституті США [9].

Спочатку необхідно обчислити пари ключів (секретний ключ і відкритий ключ). Для цього відправник (автор) електронних документів обчислює два великих простих числа p і q, потім знаходить їхній добуток $n=pq$ і значення функції Ейлера $\varphi(n) = (p-1)(q-1)$. Далі відправник обчислює число e із умов $ÍÑÄ\ (e,n) = 1, ÍÑÄ\ (e,\varphi(n)) = 1$ і число d із умов: $(d \cdot e) \bmod \varphi(n)$. Пара (e,n) – публічний ключ. Пара (d,n) – приватний ключ.

Узагальнена схема формування і перевірки цифрового підпису RSA показана на рис. 13.1.

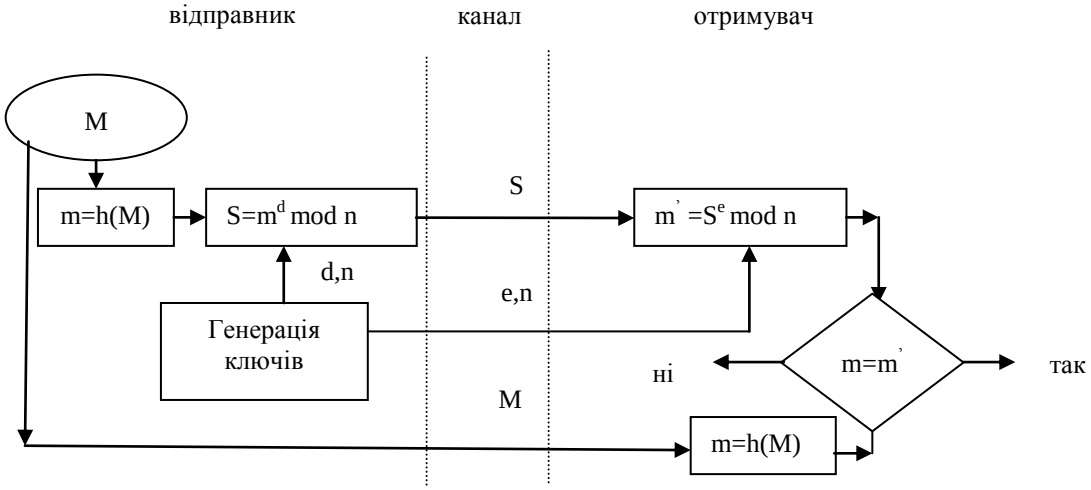

Рисунок 13.1 – Узагальнена схема цифрового підпису RSA

Допустимо, що відправник хоче підписати повідомлення *M* перед його відправленням. Спочатку повідомлення *M* (блок інформації, файл, таблиця) стискають за допомогою деякої хэш-функції у ціле число *m: m = h(M)*. Потім обчислюють цифровий підпис *S* під електронним документом *M*, використовуючи хэш-значення *m* і секретний ключ *d*: $S = m^d (\mathrm{mod} n)$. Пара *(M, S)* передається партнерові-одержувачеві як електронний документ *M*, підписаний цифровим підписом *S*, причому підпис *S* сформований власником секретного ключа *d*.

Після прийому пари *(M, S)* одержувач обчислює хэш-значення повідомлення *M* двома різними способами. Насамперед він відновлює хэш-значення *m'*, застосовуючи криптографічне перетворення підпису *S* з використанням відкритого ключа *e*: $m' = S^e (\mathrm{mod}\, n)$. Крім того, він знаходить результат хешування прийнятого повідомлення *M* за допомогою такої ж хэш-функції: *m = h(M)*. Якщо дотримується рівність обчислених значень, тобто $S^e (\mathrm{mod} n) = h(M)$, то одержувач визнає пару *(M, S)* справжньою.

Доведено, що тільки власник секретного ключа *d* може сформувати цифровий підпис *S* за документом *M*, а визначити секретне число *d* по відкритому числу *e* не легше, ніж розкласти модуль *n* на множники.

Крім того, можна строго математично довести, що результат перевірки цифрового підпису *S* буде позитивним тільки в тому випадку, якщо при обчисленні *S* був використаний секретний ключ *d*, що відповідає відкритому ключеві *e*. Тому відкритий ключ *e* іноді називають «ідентифікатором», що підписав.

Недоліки алгоритму цифрового підпису RSA.

1. При обчисленні модуля *n*, ключів *e* и *d* для системи цифрового підпису RSA необхідно перевіряти велику кількість додаткових умов, що зробити практично важко. Невиконання кожної з цих умов уможливлює фальсифікацію цифрового підпису з боку того, хто знайде таке невиконання. Під час підписання важливих документів не можна допускати таку можливість навіть теоретично.

2. Для забезпечення криптостійкості цифрового підпису RSA стосовно спроб фальсифікації на рівні, наприклад, національного стандарту США на шифрування інформації (алгоритм DES), тобто 10^{18}, необхідно використовувати при

обчисленнях *n*, *d* і *e* цілі числа не менше 2^{512} (або близько 10^{154}) кожне, що вимагає великих обчислювальних витрат, що перевищують на 20...30% обчислювальні витрати інших алгоритмів цифрового підпису при збереженні того ж рівня криптостойкості.

3. Цифровий підпис RSA уразливий до так називаної мультиплікативної атаки. Інакше кажучи, алгоритм цифрового підпису RSA дозволяє зловмисникові без знання секретного ключа *d* сформувати підпис під тими документами, у яких результат хэшування можна обчислити як добуток результатів хэшуваня вже підписаних документів. Припустимо, що зловмистник може сконструювати три повідомлення M_1, M_2, M_3 у яких $m_1=h(M_1)$, $m_2=h(M_2)$, $m_3=h(M_3)$, де $m_3=m_1 \cdot m_2 (mod\ n)$.

Припустимо, що значення двох повідомлень M_1, M_2 отримані законні підписи $S_1= m_1^d (\text{mod } N)$, $S_2= m_2^d (\text{mod } N)$.

Тоді зловмистник може легко обчислити S_3 для M_3: $S_3 = S_1 \cdot S_2 (\text{mod} n)$. Доведемо: $S_3 = m_1^d m_2^d (\text{mod} n) = (m_1 m_2)^d \bmod n = m_3^d \bmod n$.

13.3.2. Алгоритм цифрового підпису Ель-Гамаля (EGSA)

Більш надійний і зручний для реалізації на персональних комп'ютерах алгоритм цифрового підпису був розроблений у 1984 р. американцем арабського походження Тахером Ель Гамалем. У 1991 році НИСТ США обґрунтував перед комісією Конгресу США вибір алгоритму цифрового підпису Ель Гамаля як основу для національного стандарту. Назва EGSA походить від слів El Gamai Signature Algorithm (алгоритм цифрового підпису Ель Гамаля). Ідея EGSA заснована на тому, що для обґрунтування практичної неможливості фальсифікації цифрового підпису може бути використана більш складна обчислювальна задача, ніж розкладання на множники великого цілого числа, задача дискретного логарифмування. Крім того, Ель Гамалю вдалося уникнути явної слабості алгоритму цифрового підпису RSA, зв'язаної з можливістю підробки цифрового підпису під деякими повідомленнями без визначення секретного ключа [9; 10].

Розглянемо докладніше алгоритм цифрового підпису Ель Гамаля. Для того, щоб генерувати пари ключів (відкритий ключ – секретний ключ), спочатку вибирають деяке велике просте ціле число *P* і велике ціле число *G*, причому $G<P$. Відправник і одержувач підписаного документа використовують при обчисленнях однакові великі цілі числа *P* (~10^{308} або ~2^{T024}) і *G* (~10^{154} або ~2^{512}), що не є секретними.

Відправник вибирає випадкове ціле число *X*, $1<X<(P-1)$, і обчислює $Y=G^X mod\ P$.

Число *Y* є відкритим ключем, використовуваним для перевірки підпису відправника. Число *X* є секретним ключем відправника для підписування документів і повинне зберігатися в секреті. Для того, щоб підписати повідомлення *M*, спочатку відправник хешує його за допомогою хэш-функції у ціле число *m*: $m = h(M)$, $1<m<(P-1)$.

Далі генеруємо випадкове ціле число *K*, $1< K< (P-1)$, таке, що *K* і $(P-1)$ є взаємно простими. Потім відправник обчислює ціле число *a*: $a=G^K\ mod\ P$ і,

застосовуючи розширений алгоритм Евкліда, обчислює за допомогою секретного ключа X ціле число b з рівняння: $m=X\cdot a+K\cdot b(mod\ (P-1))$.

Пара чисел (a, b) утворить цифровий підпис S: $S = (a,b)$, що проставляється під документом M. Трійка чисел (M, a, b) передається одержувачеві, у той час як пара чисел (X,K) тримається в секреті.

Після прийому підписаного повідомлення (M, a, b) одержувач повинний перевірити, чи відповідає підпис $S = (a,b)$ повідомленню M. Для цього одержувач спочатку обчислює по прийнятому повідомленню M число $m = h(M)$.

Потім одержувач обчислює значення: $A=Y^a a^b(mod\ P)$ і визнає повідомлення M справжнім, якщо, і тільки якщо $A = G^m(mod\ P)$.

Інакше кажучи, одержувач перевіряє справедливість співвідношення:

$Y^a a^b(mod\ P) = G^m(mod\ P)$.

Можна строго математично довести, що остання рівність буде виконуватися тоді, і тільки тоді, коли підпис $S = (a,b)$ під документом M отримана за допомогою саме того секретного ключа X, з якого був отриманий відкритий ключ Y. Таким чином, можна надійно упевнитися, що відправником повідомлення M був власник саме даного секретного ключа X, не розкриваючи при цьому сам ключ, і що відправник підписав саме цей конкретний документ M. Слід зазначити, що виконання кожного підпису по методу Ель Гамаля вимагає нового значення K, причому це значення повинне вибиратися випадковим чином.

Приклад

Виберемо числа P=11, G=2 і секретний ключ X=8. Обчислюємо значення відкритого ключа: $Y= G^x\ mod\ P = Y= 2^8\ mod\ 11 = 3$. Припустимо, що вихідне повідомлення M характеризується хеш-значенням $m=5$.

Для того, щоб обчислити цифровий підпис для повідомлення M, що має хэш-значення $m=5$, спочатку виберемо випадкове ціле число $K=9$. Переконаємося, що числа K и (P-1) є взаємно простими. Дійсно, $НСД(9, 10) =1$.

Далі обчислюємо елементи a і b підпису: $a = G^K\ mod\ P = 2^9\ mod\ 11 = 6$, елемент b визначаємо, використовуючи розширений алгоритм Евкліда:

$m = X\cdot a + K\cdot b(mod(P-1))$.

При $m = 5, a = 6, X = 8, K=9, P=11$ одержуємо $5 = (6\cdot 8 + 9\cdot b)(mod\ 10)$ або $9\cdot b = -43\ (mod\ 10)$. Розв'язання: b=3.

Цифровий підпис являє собою пару: a = 6, b = 3.

Далі відправник передає підписане повідомлення. Прийнявши підписане повідомлення і відкритий ключ Y=3, одержувач обчислює хэш-значення для повідомлення M: $m = 5$, а потім обчислює два числа:

$Y^a a^b(mod\ P) = 3^6\cdot 6^3\ (mod\ 11) =10\ (mod\ 11)=10;$

$G^m\ (mod\ P) = 2^5\ (mod\ 11) =10\ (mod\ 11)=10.$

Так як ці два цілих числа рівні, прийняте одержувачем повідомлення визнається справжнім.

Слід зазначити, що схема Ель-Гамаля являє характерний приклад підходу, що допускає пересилання повідомлення M в відкритій формі разом із приєднаним аутентифікатором (a,b). У таких випадках процедура встановлення

дійсності прийнятого повідомлення складається з перевірки відповідності аутентифікатора повідомленню.

Схема цифрового підпису Ель-Гамаля має ряд переваг у порівнянні зі схемою цифрового підпису RSA:

1. При заданому рівні стійкості алгоритму цифрового підпису цілі числа, що беруть участь в обчисленнях, мають запис на 25% коротший, що зменшує складність обчислень майже в два рази і дозволяє помітно скоротити обсяг використовуваної пам'яті.

2. При виборі модуля *P* досить перевірити, що це число є простим і що в числа (*P-1*) міститься великий простий множник (тобто всього дві досить прості умови).

3. Процедура формування підпису за схемою Ель-Гамаля не дозволяє обчислювати цифрові підписи під новими повідомленнями без знання секретного ключа (як у RSA).

Однак алгоритм цифрового підпису Ель-Гамаля має і деякі недоліки в порівнянні зі схемою підпису RSA. Зокрема, довжина цифрового підпису виходить у 1,5 рази більше, що, у свою чергу, збільшує час її обчислення.

13.3.3. Алгоритм цифрового підпису DSA

Алгоритм цифрового підпису DSA (Digital Signature Algorithm) запропонований у 1991 році у НИСТ США для використання в стандарті цифрового підпису DSS (Digital Signature Standard) Алгоритм DSA є розвитком алгоритмів цифрового підпису Ель-Гамаля і К. Шнорра і винайдений Девідом Кравіцом.

Відправник і одержувач електронного документа використовують при обчисленні великі цілі числа: p – просте числа, довжиною L біт $(512 < L < 1024)$; q – просте число довжиною 160 біт (дільник числа $(p-1)$). k – елемент з множини $\{1, p\}$ порядку q. Вибирається будь-яке $a < q$, обчислюємо $b=k^a \bmod p$ і $H(M)$ – хеш-функція (стандарт вимагає – SHA).

Отже, (p,q,k,b) – публічний ключ, a – секретний ключ.

Для формування ЕЦП повідомлення M власник приватного ключа виконує такі дії:

1) обираємо будь-яке число $r < q$;

2) обчислуємо $r' = r^{-1} \bmod q$;

3) обчислюємо $S_1 = (k^r \bmod p) \bmod q$;

4) обчислюємо $S_2 = r' (H(M) + a \cdot S_1) \bmod q$;

5) (S_1,S_2) – підпис.

Перевірка ЕЦП:

1) отримувач обчислює $S' = S_2^{-1} \bmod q$;

2) обчислює $U_1 = H(M)\, S' \bmod q$;

3) обчислює $U_2 = S' S_1 \bmod q$;

4) обчислює $t = (k^{U_1} \cdot b^{U_2} \bmod p) \bmod q$.

Якщо $t = S_1$, то підпис справжній.

Безпека DSA: основний недолік першої версії DSA – довжина p - 512 біт. Успішні атаки на DSA досі невідомі.

13.3.4. Вітчизняний стандарт цифрового підпису

Вітчизняний стандарт цифрового підпису ГОСТ Р 34.10-94 та ГОСТ Р 34.10-2001. Алгоритми, обумовлені цим стандартом, концептуально близькі до алгоритму DSA, в них використовуються наступні параметри:

– p – просте число, довжиною від 509 до 512 біт або 1020 до 1024 біт;

– q – простий співмножник числа ($p-1$), що має довжину 254...256 біт;

– a – будь-яке число, менше ($p-1$), причому таке, що $a^q \bmod p=1$;

– x – будь-яке число, менше q;

– $y = a^x \bmod p$.

Крім того, цей алгоритм використовує хэш-функцию $H(x)$, стандарт ГОСТ Р 34.10-94 визначає хеш-функцію, засновану на використанні стандартного симетричного алгоритму ГОСТ 28147-89.

Перші три параметри p,q,a є відкритими і можуть бути загальними для всіх користувачів мережі. Число x є секретним ключем. Число y є відкритим ключем.

Щоб підписати деяке повідомлення, а потім перевірити підпис, виконуються наступні кроки:

Крок 1 Створення підпису:

1) генеруємо $k < q$;

2) $r' = a^k \bmod p$ та $r = r' \bmod q$; // Якщо $r=0$, то генеруємо інше число;

3) $S = xr + k \cdot H(M) \bmod q$; // Якщо $r=0$, то генеруємо інше число,

S – підпис;

4) Відправляють (M,S).

Крок 2 Перевірка підпису :

$r = (a^{S \cdot H(M)^{-1}} \cdot y^{-r \cdot H(M)^{-1}} \bmod p) \bmod q$.

Є новий стандарт з 1 липня 2002 р. ГОСТ – 2001. У цьому стандарті використовуються операції з групами точок еліптичної кривої над скінченим полем.

13.3.5. Цифрові підписи з додатковими функціональними властивостями

Розглянуті в цьому розділі цифрові підписи володіють додатковими функціональними можливостями, крім звичайних властивостей аутентифікації повідомлення і неможливості відмовлення особи, що підписала, від зобов'язань, зв'язаних з підписаним текстом. У більшості випадків вони поєднують базову схему цифрового підпису, наприклад на основі алгоритму RSA, зі спеціальним протоколом, що забезпечує досягнення тих додаткових властивостей, якими базова схема цифрового підпису не володіє.

Схеми сліпого підпису

На відміну від звичайних схем цифрового підпису, схеми сліпого підпису (іноді називані схемами підпису наосліп) є двосторонніми протоколами між відправником A і стороною B, що підписує документ.

Основна ідея цих схем полягає в наступному. Відправник A посилає порцію інформації стороні B, що B підписує і повертає A. Використовуючи отриманий підпис, сторона A може обчислити підпис сторони В на більш

важливому для себе повідомленні *T*. По завершенні цього протоколу сторона *B* нічого не знає ні про повідомлення *T*, ні про підпис під цим повідомленням. Ціль сліпого підпису полягає в тому, щоб перешкодити особі, що підписує *B*, ознайомитися з повідомленням сторони *A*, що він підписує, і з відповідним підписом під цим повідомленням. Тому надалі підписане повідомлення неможливо зв'язати зі стороною *A*.

Наведемо приклад застосування сліпого підпису. Схема сліпого підпису може знайти застосування в тих випадках, коли відправник *A* (клієнт банку) не хоче, щоб сторона, що підписує *B* (банк), мала можливість надалі зв'язати повідомлення *m* і підпис $S_B(m)$ з визначеним кроком виконаного раніше протоколу. Зокрема, це може бути важливо при організації анонімних безготівкових розрахунків, коли повідомлення *m* могло б представляти грошову суму, що *A* хоче витратити. Коли повідомлення *m* з підписом $s_B(m)$ пред'являється банкові *B* для оплати, банк *B* не може простежити, хто саме з клієнтів пред'являє підписаний документ. Це дозволяє користувачеві *A* залишитися анонімним.

Схеми незаперечного підпису

Незаперечний підпис, як і звичайний цифровий підпис, залежить від підписаного документа і секретного ключа. Однак на відміну від звичайних цифрових підписів незаперечний підпис не може бути верифікований без участі особи, що поставили цей підпис. Можливо, більш підходящою назвою для цих підписів були б «підписи, що не допускають підробки».

Розглянемо два можливих сценарії застосування незаперечного підпису.

Сценарій 1. Сторона *A* (клієнт) хоче одержати доступ у захищену зону, контрольовану стороною *B* (банком). Цією захищеною зоною може бути, наприклад, депозитарій (сховище цінностей клієнтів). Сторона *B* жадає від *A* поставити до надання клієнтові доступу на заявці про допуск у захищену зону підпис, час і дату. Якщо *A* застосує незаперечний підпис, тоді сторона *B* не зможе згодом довести кому-небудь, що *A* одержав допуск без особистої участі *A* в процесі верифікації підпису.

Сценарій 2. Припустимо, що відома корпорація *A* розробила пакет програмного забезпечення. Щоб гарантувати дійсність пакета і відсутність у ньому вірусів, сторона *A* підписує цей пакет незаперечним підписом і продає його стороні *B*. Сторона *B* вирішує зробити копії цього пакета програмного забезпечення і перепродати його третій стороні *C*. При використанні стороною *A* незаперечного підпису сторона *C* не зможе переконатися в дійсності цього пакета програмного забезпечення і відсутності в ньому вірусів без участі сторони *A*. Звичайно, цей сценарій не перешкоджає стороні *B* поставити на пакеті свій підпис, але тоді для сторони *B* будуть утрачені всі маркетингові переваги, зв’язані з використанням торговельної марки корпорації *A*. Крім того, буде легше розкрити шахрайську діяльність сторони *B*.

14. УПРАВЛІННЯ КЛЮЧАМИ

Під час вибору криптографічної системи найважливішою проблемою є керування ключами. Будь-яка, навіть найнадійніша, криптосистема базується на використанні ключів. Ефективність криптографічного захисту інформації в комп'ютерних системах і мережах визначається стійкістю алгоритмів криптографічних перетворень, які в ній використовуються, й надійністю протоколів керування ключами.

Управління ключами – інформаційний процес, що включає в себе три елементи: генерація ключів; накопичення ключів; розподіл ключів.

Правильне розв’язання названих задач має величезне значення, адже в більшості випадків зловмиснику легше атакувати саме ключову підсистему, а не алгоритм криптографічного захисту. Використання стійкого алгоритму шифрування є необхідною, але далеко не достатньою умовою побудови надійної системи криптографічного захисту інформації. Ключі, які використовуються в процесі інформаційного обміну, потребують однакового захисту на всіх стадіях життєвого циклу. Крім вибору потрібної для конкретної апаратної системи криптографічної системи, ще є одна важлива проблема – політика управління ключами [9].

14.1. Генерування ключів

Безпека будь-якого криптографічного алгоритму визначається використовуваним криптографічним ключем. Цей ключ повинен бути досить довгим і мати випадкові значення бітів.

Як приклад розглянемо метод генерування сеансового ключа для симетричних криптосистем, який передбачає використання криптографічного алгоритму DES. Схема генерації випадкового сенсового ключа R_i показана на рис.1.

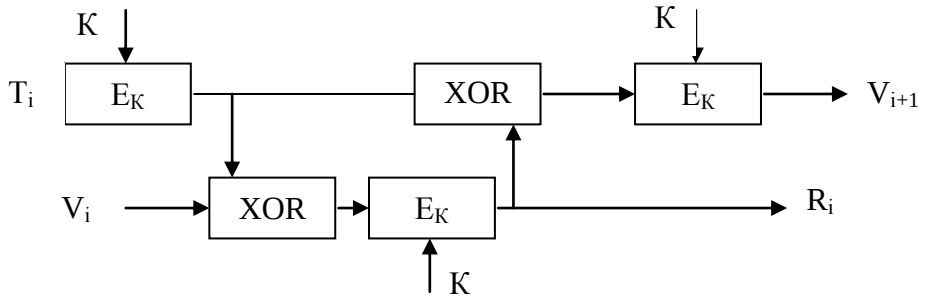

Рисунок 14.1 – Схема генерації випадкового ключа

Випадковий ключ *R* генерують, обчислюючи значення:

$R_i=E_K\ (E_K(T_i))\ \oplus V_i)$, де $V_{i=0}$ – секретне 64-бітне початкове число; T_i – часова відмітка; *K* – ключ, зарезервований для генерації секретних ключів; E_K – функція шифрування DES.

Наступне значення V_{i+1} обчислюють за формулою: $V_{i+1} = E_K(E_K(T_i) \oplus R_i)$.

У разі необхідності 128-бітного ключа генерують пару ключів R_i, R_{i+1} і об’єднують їх.

Для генерування випадкових значень ключів застосовують різноманітні програмні й апаратні засоби. Оскільки ступінь випадковості генерування чисел повинен бути досить високим, то використовуються пристрої на основі «натуральних» випадкових процесів, наприклад на основі білого радіошуму тощо. Ще одним прикладом може бути фізичний датчик випадкових чисел, вбудований в ядро процесора Pentium. Іншим випадковим математичним об'єктом є десяткові знаки трансцендентних чисел, наприклад π чи e, які обчислюються за допомогою стандартних математичних методів.

Ієрархія ключів

Є такі категорії ключів, що використовуються з різною метою:

– **Ключі шифрування даних**. Ці ключі змінюються досить часто, щоб запобігти накопиченню великої кількості криптограм, зашифрованих на одному ключі. Це полегшує роботу криптоаналітиків супротивника.

– **Ключі шифрування ключів**. Ці ключі змінюють не так часто, як перші, й використовують лише для шифрування інформації, невеликої за об'ємом: криптографічні ключі, вектори ініціалізації тощо.

– **Головний ключ**. Цей ключ змінюється найрідше з усіх, тому що використовується лише для генерування інших криптографічних ключів. Виходячи з використання, засоби безпеки при зберіганні цих ключів повинні бути найвищими, оскільки компрометація головного ключа призводить до повного краху всієї криптографічної системи. Для більшої безпеки при зберіганні такого ключа часто використовують принцип «розділення секрету», коли ключ ділять на кілька частин, які або запам'ятовують різні люди, або частини ключа записують у різні пристрої зберігання. Отже, ключ ніколи не використовується з іншою метою; ніколи не зчитується інакше, як у криптографічний пристрій.

Існує також поняття **сесійного ключа**. Рекомендовано використовувати окремі ключі для шифрування зв'язку, який є в кожній окремій сесії (сесія може охоплювати, наприклад, пересилання масивів із віддаленого сервера або операцію користувача, яка виконується в рамках персональної реєстрації в системі з можливістю доступу багатьох користувачів). У цьому випадку визначення ключа для кожної сесії відбувається з використанням ключів шифрування ключів, а не поточного сесійного ключа. Застосування сесійних ключів потрібне з огляду на таке:

– це обмежує кількість криптограм, які генерують за допомогою одного ключа, чим обмежено кількість криптограм, які зловмисник може аналізувати;

– обмежено час, необхідний для відгадування ключа зловмисником. Після закінчення кожної сесії зловмисник не має змоги виконати активну атаку.

14.2. Накопичення ключів

Під накопичення ключів розуміють організацію їх збереження, обміну, видалення. Цим питанням необхідно приділяти особливу увагу, секретні ключі ніколи не повинні записуватися в явному виді на носії, який може бути прочитаним, скопійованим.

У досить складній системі один користувач може працювати з великим об’ємом ключової інформації, інколи навіть виникає необхідність в організації мінібаз даних із ключовою інформацією. Такі бази даних відповідають за приймання, збереження, облік, видалення використовуваних ключів. Кожна інформація про використовувані ключі повинна зберігатися в зареєстрованому виді. Ключі, які зашифровують ключову інформацію, називаються мастер ключами. Бажано, щоб мастер ключі кожен користувач знав на пам'ять, і не зберігав їх взагалі на яких-небудь магнітних носіях. Дуже важливою умовою безпеки інформації – періодичне оновлення ключової інформації в АС. При цьому оновлюватися повинні як звичайні ключі, так і мастер-ключі.

14.3. Розподіл ключів

Найвідповідальніший момент в процесі управління ключами, до нього відносять ряд вимог: оперативність і точність розподілу; прихованість розподілу.

Розподіл ключів між користувачами реалізуються двома різними підходами:

– Шляхом створення одного або декількох центрів ключів. Недолік такого підходу полягає в тому, що в центрі розподілу відома кому і які ключі призначені. Це дозволяє читати всі повідомлення, що циркулюють в АС.

– Прямий обмін ключами між користувачами інформаційної системи. Проблема полягає в тому, щоб надійно визначити дійсність суб’єктів.

В обох випадках повинна бути гарантована дійсність сеансу зв’язку. Це можна зробити двома способами:

– Механізм запиту-відповіді. Він полягає в наступному: користувач А хоче бути впевненим, що повідомлення, які він отримує від В не є хибними. Він включає посилаючи до В повідомлення непередбачуваний елемент (запит). При відповіді В повинен виконати деяку операцію над цими елементами (+1). Це не можливо зробити наперед, так як не відомо, яке випадкове число прийде в запиті. Після отримання відповіді з результатом, А може бути впевнений, що сеанс істинний.

Недолік: Можливість встановлення складної закономірності між запитом і відповіддю.

– Механізм відліку часу («часовий штемпель»). Фіксація часу для кожного повідомлення. Для цього випадку кожен користувач АС може знати наскільки «старим» є надіслане повідомлення.

В обох механізмах необхідно використовувати шифрування, щоб бути впевненим, що відповідь відправлена не зловмисником, і тимчасовий штемпель не змінений. При використанні відміток часу виникає проблема допустимого часового інтервалу затримки для підтвердження дійсності сеансу. Повідомлення з тимчасовим штемпелем в принципі не може бути передане зразу, крім цього комп’ютерний годинник відправника і отримувача не можуть бути абсолютно синхронізовані. Яке запізнення «штемпеля» рахувати підозрілим? У реальних АС, наприклад системи оплати кредитних карток,

використовують другий механізм встановлення і захисту від підробок. Тимчасовий інтервал складає кілька хвилин. Велика кількість відомих способів крадіжки електронних грошей засновані на «вклиненні» в цей проміжок часу з підставним запитом на зняття грошей.

В реальних інформаційних системах, наприклад, в системах оплати кредитними картками, використовується саме механізм додавання часової відмітки, причому часовий інтервал може становити від одної до декількох хвилин. Таким чином, задача управління ключами зводиться до пошуку такого протоколу розподілу ключової інформації, який забезпечує:

- можливість відмови від центру розподілу ключів;
- взаємне підтвердження дійсності учасників сеансу;
- підтвердження дійсності сеансу механізмом запит-відповідь, використання для цього програмних і апаратних засобів;
- використання при обміні ключами мінімального числа повідомлень.

14.4. Сертифікати ключів

У криптосистемах із відкритим ключем використовуються два різні ключі – публічний та приватний.

Сертифікатом відкритого ключа називається повідомлення центру розподілу ключів (ЦРК), яке засвідчує цілісність деякого відкритого ключа об'єкта. Наприклад, сертифікат відкритого ключа користувача $A(C_A)$ містить відмітку часу t, ідентифікатор ID_A, відкритий ключ K_A, зашифровані секретним ключем ЦРК $k_{ЦРК}$, тобто $C_A=E_{kЦРК}(t, Id_A, K_A)$.

Секретний ключ $k_{ЦРК}$ відомий тільки менеджеру ЦРК. Відкритий ключ $K_{ЦРК}$ відомий учасникам A і B.

Об'єкт учасника A викликається й ініціює стадію встановлення ключа, після чого запитує у ЦРК сертифікат свого відкритого ключа і відкритого ключа учасника B, тобто A відсилає ЦРК ID_A і ID_B (унікальні ідентифікатори відповідно учасників A і B). Менеджер ЦРК відповідає таким повідомленням: $E_{kЦРК}(t, ID_A, K_A)$, $E_{kЦРК}(t, ID_B, K_B)$.

Учасник A, використовуючи відкритий ключ ЦРК $K_{ЦРК}$, розшифровує відповідь ЦРК і перевіряє обидва сертифікати. Ідентифікатор ID_B переконує A, що особистість правильного учасника зафіксована в ЦРК і K_B – дійсно відкритий ключ учасника B, оскільки обидва зашифровані ключем $k_{ЦРК}$.

Наступний крок протоколу полягає у встановленні зв'язку A з B: C_A, $E_{kA}(t)$, $E_{KB}(r_1)$.

Тут C_A – сертифікат відкритого ключа користувача A; $E_{кA}(t)$ – відмітка часу, зашифрована секретним ключем учасника A, і є підписом учасника A, оскільки ніхто інший не може створити такий підпис; r_1 – випадкове число, яке генерує учасник A і використовує для обміну з B у ході процедури перевірки достовірності.

Якщо сертифікат C_A і підпис A правильні, то учасник B упевнений, що повідомлення прийшло від A. Частину повідомлення E_{KB} (r_1) може розшифрувати тільки B, бо більше ніхто не знає секретного ключа k_B, який

відповідає відкритому ключу K_B. Користувач B розшифровує значення числа r_1, і, щоб підтвердити свою достовірність, відправляє учаснику A повідомлення: $E_{KA}(r_1)$.

Учасник A відновлює значення r_1, розшифровуючи це повідомлення з використанням свого секретного ключа k_A. Якщо це очікуване значення r_1, то A отримує підтвердження, що визнаний учасник дійсно B.

Якщо множина користувачів відкритого ключа невелика, то такий ключ можна поширити ручним способом, записуючи його на будь-який носій інформації. Далі варто захистити його від модифікації, застосовуючи, наприклад, організаційні заходи захисту. Проте такий спосіб поширення відкритих ключів неприйнятний для великих систем, особливо для розподілених у просторі. На початку 80-х років XX століття було запропоновано поняття «сертифікат відкритого ключа», використане потім у нормативних документах з інформаційних технологій, першим з яких був ITU X.509, що стосується забезпечення автентичності в електронних довідниках. Сертифікат є документом (у цьому випадку електронним), де розміщується інформація, автентичність якої гарантується органом, що видав цей сертифікат. Сертифікат містить у собі такі поля:

– номер версії;

– порядковий номер сертифіката;

– ідентифікатор алгоритму, використовуваного для цифрового підпису сертифіката;

– найменування органу, що видав сертифікат;

– термін придатності сертифіката (дати початку і кінця дії);

– ім’я власника сертифіката;

– відкритий ключ;

– цифровий підпис органу сертифікації.

Сертифікат має такі властивості:

– кожен користувач, що має доступ до публічного ключа центру сертифікації, який видав той чи інший сертифікат, може отримати з цього сертифіката публічний ключ користувача та перевірити його аутентичність;

– жодна зі сторін (окрім центру сертифікації) не може непомітно змінити сертифікат (сертифікат неможливо підробити).

Створення сертифіката починається зі створення пари ключів (приватний-публічний). Це може робити або центр сертифікації, або власне користувач. В останньому випадку публічний ключ надається центру сертифікації. Після цього центр сертифікації генерує сертифікат, який розміщується в Інтернет для того, щоб всі учасники інформаційного обміну могли ним користуватися.

Розглянемо ієрархічну модель сертифікації.

Глобальними, або, як їх ще називають, кореневими центрами сертифікації, виступають спеціально створені структури, яким довіряють усі без винятку підлеглі центри. Такі структури мають найпотужніші можливості генерування, зберігання ключів і сертифікації. Прикладом може служити спеціально створена компанія Verisign. Кореневі центри не працюють

безпосередньо з користувачами, а лише з підпорядкованими їм центрами сертифікації та їх сертифікатами [9]

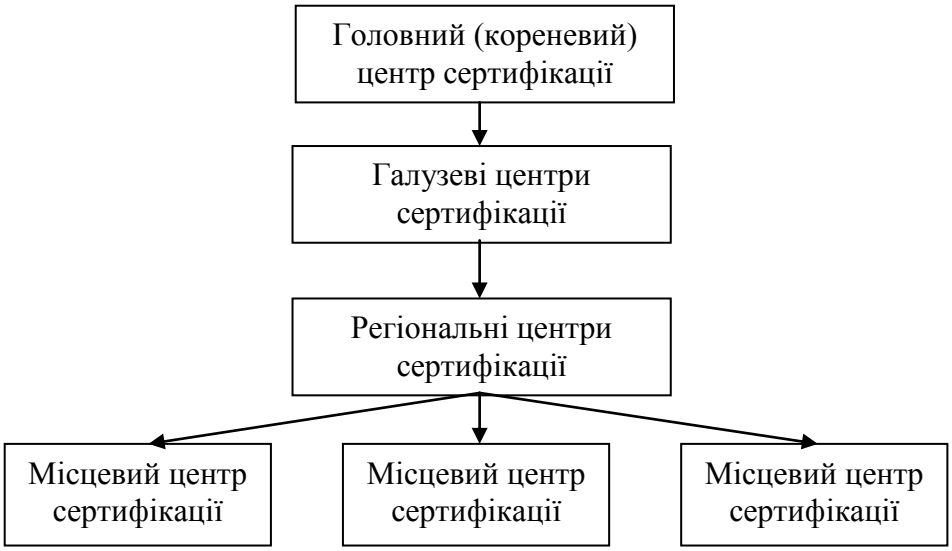

Рисунок 14.2 – Ієрархічна модель інфраструктури публічних ключів

Галузеві центри сертифікації (або Brand Certification Authority – галузеві сертифікаційні авторитети), наприклад фірма Microsoft, сертифікує підпорядковані центри та окремих крупних клієнтів. На регіональному рівні галузеві центри сертифікації делегують свої права **регіональним центрам**, які в свою чергу, організовують цілу мережу місцевих **центрів сертифікації.** Останні вже безпосередньо працюють із місцевими користувачами. Вони генерують пари ключів, забезпечують їх розповсюдження та підтримку сертифікатами. Сертифікати публічних ключів розміщуються на серверах місцевих або, найчастіше, регіональних центрів і забезпечуються захистом від підміни.

Сукупність центрів сертифікації утворює так звану **інфраструктуру публічних ключів** (Public Key Infrastructures – PKI), що забезпечує створення та функціонування в системах сертифікатів публічних ключів. Зокрема, нею визначається статус сертифікатів, використання яких може бути як заблоковане, так і анульоване. Необхідно забезпечити постійний доступ до інформації, що стосується сертифікатів публічних ключів, а також можливість відшкодування збитку, завданого внаслідок виявлених вад у системі сертифікації. Значне поширення інформаційних технологій зумовило потребу в наданні електронним документам юридичної чинності. Раніше було згадано про роль цифрового підпису в цьому явищі. У прийнятих на сьогодні відповідних законах велику увагу приділено правовим питанням діяльності органів сертифікації публічних ключів.

Із технічної (і правової) точки зору необхідно звернути увагу на такий аспект проблеми поширення публічних ключів. Через постійне збільшення кількості користувачів асиметричних криптосистем не видається можливим функціонування єдиного для всього населення земної кулі органу сертифікації

відкритих ключів. Отже, PKI та відповідні закони призначені для організації взаємодії різних органів і систем сертифікації відкритих ключів.

Таким чином, ми можемо визначити основні завдання інфраструктури публічних ключів:

– підтримка життєвого циклу ключів і сертифікатів (тобто генерування ключів, створення та підписування сертифікатів, їх розповсюдження, захист тощо);

– реєстрування фактів компрометації ключів і публікування «чорних» списків відізваних сертифікатів;

– підтримка процесів ідентифікації та аутентифікації користувачів за допомогою сертифікатів;

– реалізація механізму інтеграції існуючого програмного забезпечення безпеки на основі сервісів PKI.

Протокол прямого обміну ключами

Два користувачі, які бажають обмінятися криптографічно захищеною інформацією, при використанні для інформаційного обміну криптосистеми з симетричним секретним ключем повинні мати загальний секретний ключ і обмінятися ним по каналу зв'язку в безпечний спосіб. Якщо ключ змінюється досить часто, то його доставка перетворюється в серйозну проблему.

Є два способи розв'язання цієї проблеми:

– використання асиметричної криптосистеми з відкритим ключем для шифрування і передачі секретного ключа симетричної криптосистеми;

– використання системи відкритого розподілу ключів Діффі-Хелмана.

Реалізація першого способу, який ще іноді називають способом електронного цифрового «конверта», здійснюється в рамках комбінованої криптосистеми з симетричними й асиметричними ключами. При такому підході симетрична криптосистема застосовується для шифрування й передачі відкритого тексту, а асиметрична криптосистема з відкритим ключем - для шифрування, передачі й розшифрування тільки секретного ключа симетричної криптосистеми.

Як приклад реалізації способу електронного цифрового «конверта» розглянемо порядок роботи комбінованої критпосистеми із симетричними й асиметричними ключами із застосуванням електронного цифрового підпису і сертифікатів відкритих ключів:

1. Безпечно створюються і поширюються асиметричні публічні й приватні ключі. Закритий асиметричний ключ передається його власнику. Відкритий асиметричний ключ зберігається в базі даних відкритих ключів і адмініструється центром видачі сертифікатів ЦРК. Користувачі повинні довіряти такій системі, де проводиться безпечне створення, розподіл і адміністрування ключів.

2. Створюється електронний цифровий підпис тексту з допомогою обчислення його хеш-функції. Отримане значення шифрується з використанням асиметричного закритого ключа відправника, а потім отриманий рядок символів додається до тексту, який передається.

3. Створюється секретний симетричний ключ, який буде використовуватися для шифрування тільки даного повідомлення чи сеансу взаємодії (сеансовий ключ); потім за допомогою цього ключа й симетричного алгоритму шифрування шифрується вихідний текст разом із доданим до нього цифровим підписом. У такий спосіб отримується зашифрований текст.

4. Далі необхідно розв'язати проблему з передачею сеансового ключа отримувачу.

5. Відправник повинен мати асиметричний відкритий ключ центру видачі сертифікатів $K_{ЦРК}$. Перехоплення незашифрованих запитів на отримання цього відкритого ключа – це найпоширеніша форма атаки. Може існувати ціла система сертифікатів, які підтверджують достовірність відкритого ключа $K_{ЦРК}$.

6. Відправник запитує у ЦРК асиметричний відкритий ключ отримувача повідомлень. Цей процес вразливий до атаки, під час якої атакуючий суб'єкт втручається у взаємодію між відправником та отримувачем і може модифікувати трафік, який передається між ними. Тому відкритий асиметричний ключ отримувача «підписується» ЦРК. Це означає, що $k_{ЦРК}$ використав свій асиметричний секретний ключ для шифрування асиметричного ключа отримувача. Тільки ЦРК знає асиметричний секретний ключ $k_{ЦРК}$. Це гарантія того, що публічний асиметричний ключ отримувача одержаний саме від ЦРК.

7. Після отримання асиметричного відкритого ключа отримувача K_B розшифровує за допомогою асиметричного відкритого ключа $K_{ЦРК}$ і алгоритму асиметричного шифрування/розшифрування.

8. Відправник шифрує сеансовий ключ з використанням асиметричного ключа отримувача. Цей ключ отримується від ЦРК і розшифровується.

9. Зашифрований сеансовий ключ приєднується до зашифрованого тексту, який містить раніше доданий електронний підпис.

10. Сформований електронний цифровий «конверт» відправляється отримувачу.

11. Отримувач виділяє зашифрований сеансовий ключ із присланого цифрового «конверта» і розв'язує проблему з розшифровкою сеансового ключа.

12. Використовуючи свій секретний асиметричний ключ і такий самий асиметричний алгоритм шифрування, отримувач розшифровує сеансовий ключ.

13. Отримувач застосовує до одержаного зашифрованого тексту розшифрований симетричний (сеансовий) ключ і такий самий симетричний алгоритм шифрування/розшифрування й отримує вихідний текст разом з електронним підписом.

14. Отримувач відділяє електронний підпис від вихідного тексту і запитує у ЦРК асиметричний відкритий ключ відправника.

15. Після отримання ключа отримувач розшифровує його з допомогою відкритого $K_{ЦРК}$ і відповідного асиметричного алгоритму шифрування/ розшифрування.

16. Потім виділяється хеш-функція тексту з використанням відкритого ключа відправника й асиметричного алгоритму шифрування/розшифрування.

Повторно обчислюється хеш-функція отриманого тексту. Дві ці хеш-функції порівнюються, щоб переконатися, що вихідний текст не було змінено.

Інший спосіб безпечного поширення секретних ключів базується на застосуванні алгоритму відкритого розподілу ключів Діффі-Хелмана. Цей алгоритм дозволяє користувачам обмінюватися ключами незахищеними каналами зв'язку.

14.5. Відкритий розподіл ключів Діффі-Хеллмана

Алгоритм відкритого розподілу ключів, винайдений В. Діффі та М. Хеллманом, дозволяє користувачам обмінюватися ключами по незахищених каналах зв'язку. Його безпека зумовлена важкістю обчислення дискретних логарифмів у скінченому полі, на відміну від легкості розв'язання прямої задачі дискретного піднесення до степеня в тому ж скінченому полі [9].

Припустимо, що A і B хочуть побудувати спільний ключ.

1) Користувачі домовляються про використання відкритих даних *a, p.*

2) Незалежно один від одного обирають навчання секретні числа k_A і k_B $(1 \le k_{A,B} \le p-1)$.

3) A і B незалежно обчислюють $K_A = a^{k_A} \bmod p$ та $K_B = a^{k_B} \bmod p$ та обмінюються K_A та K_B .

4) Користувач A, отримавши K_B обчислює: $K_B^{k_A} \bmod p \equiv (a^{k_B})^{k_A} \bmod p$.

5) Користувач B, отримавши K_A обчислює: $K_A^{k_B} \bmod p \equiv (a^{k_A})^{k_B} \bmod p$

6) $K = a^{k_A k_B} \bmod p$ – спільний секретний ресурс.

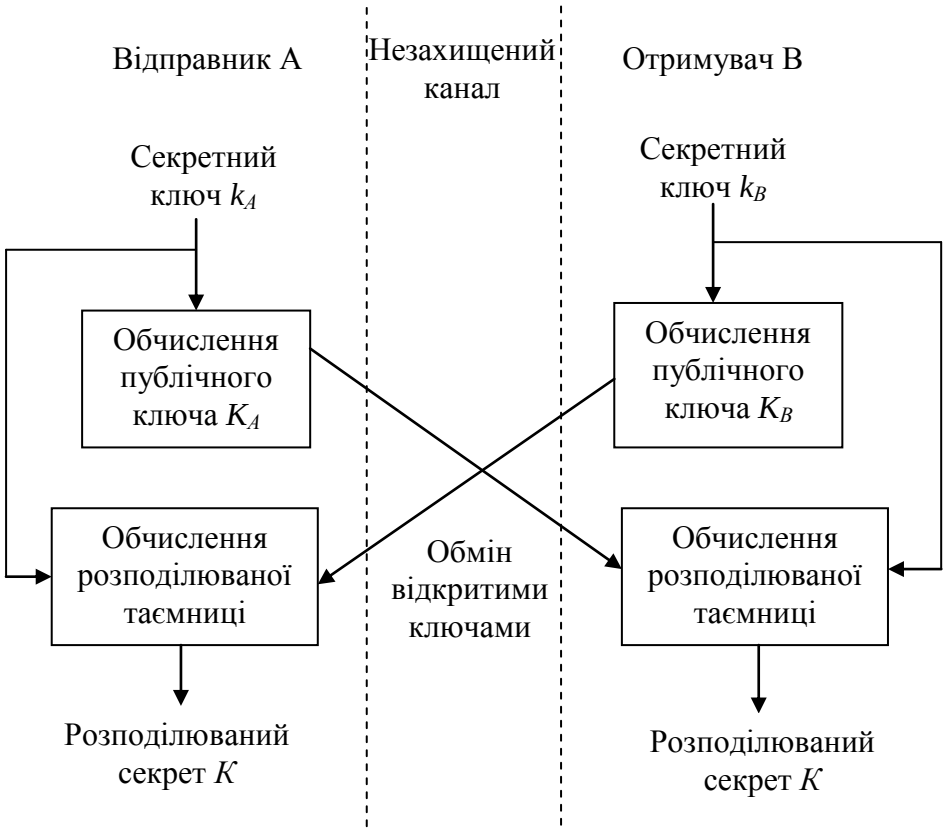

Рисунок 14.3 – Схема відкритого розподілу ключів Діффі-Хелмана

Зауваження

Супротивник з відкритого каналу лише може отримати інформацію про a,p,K_a,K_b. Для визначення k_A і k_B йому необхідно виконати задачу дискретного логарифмування, простого алгоритму розв'язку якої немає. Так при 1000-бітних простих числах потрібно більше 2000 операцій, а для оберненого перетворення – 10^{30} операцій.

Приклад

Згенерувати спільний ключ для двох користувачів.

Нехай a=7, p=13.

Відправник A	Отримувач B
Секретний ключ – k_A=5	Секретний ключ – k_B=11
Відкритий ключ – $K_A = 7^5 \bmod 13 = 11$	Відкритий ключ – $K_B = 7^{11} \bmod 13 = 2$
Отримує K_B=2	Отримує K_A=11
$K = 2^5 \bmod 13 = 6$ – секретний спільний ключ	$K = 11^{11} \bmod 13 = 6$ – секретний спільний ключ

Унікальність алгоритму Діффі-Хеллмана полягає в тому, що пара абонентів має можливість отримати відоме тільки їм секретне число, передаючи відкритою мережею публічні ключі. Після цього абоненти можуть приступити до захисту передавальної інформації вже відомим перевіреним способом – застосовуючи симетричне шифрування з використанням отриманої розділюваної таємниці.

Схема Діффі-Хеллмана дає можливість шифрувати дані при кожному сеансі зв'язку на нових ключах. Це дозволяє не зберігати таємниці на дискетах чи інших носіях. Не варто забувати, що подібне зберігання таємниць підвищує ймовірність їх попадання в руки конкурентів чи зловмисників. Протокол Діффі-Хеллмана дозволяє також реалізувати метод комплексного захисту конфіденційності й автентичності даних, які передаються. Алгоритм дає користувачу можливість сформувати та використати один ключ для виконання цифрового підпису і симетричного шифрування даних.

Протокол Діффі-Геллмана не обмежується можливістю узгодження ключа між двома учасниками. Будь-яка кількість учасників може узяти участь в узгоджені ключа через ітеративне виконання протоколу узгодження і обмін проміжними даними (які не мають бути засекреченими). Наприклад, А, Б і В можуть узяти участь в узгоджені так (всі дії відбуваються за модулем p):

1. Учасники домовляються про параметри алгоритму p і g.
2. Учасники генерують власні ключі, названі a, b і c.
3. А обчислює g^a і відправляє Б.
4. Б обчислює $(g^a)^b = g^{ab}$ і відправляє В.
5. В обчислює $(g^{ab})^c = g^{abc}$ і використовує як свій секрет.
6. Б обчислює g^b і відправляє В.
7. В обчислює $(g^b)^c = g^{bc}$ і відправляє А.
8. А обчислює $(g^{bc})^a = g^{abc}$ і використовує як свій секрет.

9. В обчислює g^c і відправляє А.
10. А обчислює $(g^c)^a = g^{ca}$ і відправляє Б.
11. Б обчислює $(g^{ca})^b = g^{abc}$ і використовує як свій секрет.

Підслуховувач може бачити $g^a, g^b, g^c, g^{ab}, g^{bc}, g^{ca}$, але не здатен використати яку-небудь з комбінацій для відтворення g^{abc}.

Для розширення механізму на більші групи, треба дотримуватись двох засад:

- Починаючи з простого ключа g, секрет утворюється піднесенням поточного значення до приватного показника один раз, в будь-якому порядку (перше таке піднесення до степеня дає нам відкритий ключ учасника).
- Будь-яке проміжне значення (яке має до N-1 виконаних піднесень до приватних показників, де N є число учасників у групі) можна показувати, але кінцеве значення (із застосованими всіма N показниками) містить спільний секрет і не може бути оприлюднене. Отже, кожен користувач повинен отримати свою копію спільного секрету застосуванням власного приватного ключа останнім.

Ці принципи залишають відкритими варіанти обрання порядку в якому учасники подають ключі. Найпростіший і найочевидніший розв'язок полягає у впорядкуванні N учасників по колу і обійти коло для кожного з них, доки кожен з учасників не зробить внеску в кожен з N ключів (з його останнім

Для одночасного захисту цілісності та конфіденційності даних варто застосовувати шифрування й електронний підпис у комплексі. Проміжні результати роботи алгоритму Діффі-Хеллмана можуть бути використані як вихідні дані для реалізації методу комплексного захисту цілісності й конфіденційності передавальних даних.

Метод комплексного захисту конфіденційності й аутентичності переданих даних працює за такою схемою:

– абонент *A* підписує повідомлення *M* за допомогою свого секретного ключа, використовуючи стандартний алгоритм цифорового підпису;

– абонент *A* обчислює сумісно розділюваний ключ за алгоритмом Діффі-Хеллмана зі свого секретного ключа і відкритого ключа абонента *B*;

– абонент *A* зашифровує повідомлення *M* на отриманому сумісно розділюваному секретному ключі , використовуючи узгоджений із партнером по обміну алгоритм симетричного шифрування;

– абонент *B* при отриманні зашифрованого повідомлення *M* обчислює за алгоритмом Діффі-Хеллмана сумісно розділюваний секретний ключ зі свого секретного ключа і публічного ключа абонента *A*;

– абонент *B* розшифровує отримане повідомлення *M* на спільному секретному ключі *K*;

– абонент *B* перевіряє підпис розшифрованого повідомлення *M* із допомогою публічного ключа абонента *A*.

15. ОСНОВИ КРИПТОАНАЛІЗУ

Криптоаналіз – це наука про розкриття вихідного тексту зашифрованого повідомлення без доступу до ключа. Успішний аналіз може розкрити вихідний текст або ключ. Він дозволяє також знайти слабкі місця в криптосистемі, що, у кінцевому рахунку, веде до тих же результатів.

Фундаментальне правило криптоаналізу, уперше сформульоване голландцем А. Керкхоффом ще в XIX столітті полягає в тому, що стійкість криптосистеми повинна визначатися тільки таємністю ключа. Іншими словами, правило Керкхоффа полягає в тому, що весь алгоритм шифрування, крім значення секретного ключа, відомий для супротивника. Це обумовлено тим, що криптосистема, що реалізує сімейство криптографічних перетворень, звичайно розглядається як відкрита система. Такий підхід відбиває дуже важливий принцип технології захисту інформації: захищеність системи не повинна залежати від таємності чого-небудь такого, що неможливо швидко змінити у випадку витоку секретної інформації. Звичайно криптосистема являє собою сукупність апаратних і програмних засобів, яку можна змінити тільки при значних витратах часу і засобів, тоді як ключ є легко змінюваним об'єктом. Саме тому стійкість криптосистеми визначається тільки таємністю ключа.

Інше майже загальноприйняте допущення в криптоаналізі полягає в тому, що криптоаналітик має у своєму розпорядженні шифротексти повідомлень.

Існує чотири основних типи криптографічних атак. Звичайно, усі вони формулюються в припущенні, що криптоаналітику відомо застосовуваний алгоритм шифрування і шифротексти повідомлень. Перерахуємо ці криптоаналітичні атаки.

1) Криптоаналітична атака при наявності тільки відомого шифртексту. Криптоаналітик має тільки шифртексти $C_1, C_2, \ldots C_m$ декількох повідомлень, причому усі вони зашифровані з використанням того самого алгоритму шифрування E_K. Робота криптоаналітика полягає в тому, щоб розкрити вихідні тексти $M_1, M_2, \ldots, M_m$, або, ще краще, обчислити ключ K, використаний для шифрування цих повідомлень, для того, щоб розшифрувати й інші повідомлення, зашифровані цим ключем.

2) Криптоаналітична атака при наявності відомого відкритого тексту. Криптоаналітик має доступ не тільки до шифротекстів $C_1, C_2, \ldots C_m$ декількох повідомлень, але також до відкритих текстів $M_1, M_2, \ldots, M_m$ цих повідомлень. Його робота полягає в пошуку ключа, використовуваного при шифруванні цих повідомлень, або алгоритму розшифрування D_K будь-яких нових повідомлень, зашифрованих тим же самим ключем.

3) Криптоаналітична атака при можливості вибору відкритого тексту. Криптоаналітик не тільки має доступ до шифрів-текстів $C_1, C_2, \ldots C_m$ і зв'язаним з ними відкритим текстам $M_1, M_2, \ldots, M_m$ декількох повідомлень, але і може за бажанням обирати відкриті тексти, що потім одержує в зашифрованому виді. Такий криптоаналіз виходить більш могутнім у порівнянні з криптоаналітиком за відомим відкритим текстом, тому що криптоаналітик може вибрати для шифрування такі блоки відкритого тексту, що дадуть більше інформації про

ключ. Робота криптоаналітика складається в пошуку ключа, використаного для шифрування повідомлень, або алгоритму расшифрування D_K нових повідомлень, зашифрованих тим же ключем.

4) Криптоаналітична атака з адаптивним вибором відкритого тексту. Це особливий варіант атаки з вибором відкритого тексту. Криптоаналітик може не тільки обирати відкритий текст, що потім шифрується, але і змінювати свій вибір у залежності від результатів попереднього шифрування. При криптоаналізі з простим вибором відкритого тексту криптоаналітик звичайно може вибирати кілька великих блоків відкритого тексту для їхнього шифрування; при криптоаналізі з адаптивним вибором відкритого тексту він має можливість вибрати спочатку більш дрібний спробний блок відкритого тексту, потім вибрати наступний блок у залежності від результатів першого вибору, і т.д. Ця атака надає йому ще більше можливостей, чим попередні типи атак.

Крім перерахованих основних типів криптоаналітичних атак, можна відзначити, принаймні, ще два типи.

5) Криптоаналітична атака з використанням обраного шифротекста. Криптоаналітик може обирати для розшифрування різні шифротексти $C_1, C_2, \ldots C_m$ і має доступ до розшифрованих відкритих текстів $M_1, M_2, \ldots, M_m$. Наприклад, він одержав доступ до захищеного від несанкціонованого розкриття блоку, що виконує автоматичне розшифрування. Робота аналітика полягає в відшуканні ключа. Цей тип криптоаналізу становить особливий інтерес для розкриття алгоритмів з відкритим ключем.

6) Криптоаналітична атака методом повного перебору всіх можливих ключів. Ця атака припускає використання криптоаналітиком відомого шифротексту і здійснюється за допомогою повного перебору всіх можливих ключів з перевіркою, чи є осмисленим відкритий текст, що виходить. Такий підхід вимагає залучення великих обчислювальних ресурсів і іноді називається силовою атакою. Існують і інші, менш розповсюджені, криптоаналітичні атаки.

15.1 Основні принципи криптоаналізу

1. **Принцип Керкхоффа.** Тільки супротивник може судити про криптостійкість системи.
2. **Принцип Керкхоффа-Шеннона.** Супротивник знає використовувану систему з точністю до ключової інформації.
3. **Принцип Жеверже.** Поверхневі ускладнення системи можуть бути ілюзійними, так як породжують хибні оцінки її стійкості.
4. При оцінці криптостійкості необхідно враховувати можливі криптографічні помилки та інші порушення безпеки.

Криптограф Ларс Кнудсен пропонує наступну класифікацію успішних результатів криптоаналізу блочних шифрів в залежності від об’єму і якості секретної інформації, яку вдалося отримати:

- повний злом – криптоаналітик обчислює ключ;

− глобальна дедукція – криптоаналітик розробляє функціональний еквівалент досліджуваного алгоритму, що дозволяє зашифровувати і розшифровувати інформацію без знання ключа;

− часткова дедукція – криптоаналітику вдається розшифрувати деякі повідомлення;

− інформаційна дедукція – крипто аналітик отримує деяку інформацію про відкритий текст чи ключ.

Як правило, криптоаналіз розпочинається із спроб злому (атаки) спрощеної версії алгоритму, після чого результати розповсюджуються на повну версію.

Властивість системи протистояти будь-яким атакам називається її **стійкістю**. Кількісно стійкість визначається складністю криптоалгоритму. **Універсальний метод прямого перебору** всієї множини ключів дозволяє зробити оцінку зверху для стійкості алгоритмів.

Розрізняють стійкість ключа (складність розкрити найкращим методом); стійкість безключового читання (складність нав'язування неправдивої інформації найкращим алгоритмом).

Аналогічно розглядають стійкість криптоалгоритму, протоколу, алгоритму генерування і розподілу ключів. У залежності від складності злому алгоритми забезпечують різні степені захисту**. Існують безумовно стійкі (теоретично стійкі), доведено стійкі і припустимо стійкі.**

Теоретично стійкий криптоалгоритм – такий алгоритм, коли ніякий метод криптоаналізу не дозволяє не тільки визначити ключ і відкритий текст, але й навіть отримати деяку інформацію про них. Такі системи на практиці незручні (симетричні системи із одноразовим використанням ключа потребують більшої захищеної пам'яті для збереження ключів, системи квантової криптографії потребують волоконно-оптичних каналів зв'язку, які є дорогими). Крім того доведення їх стійкості виходить за межі області математики і фізики. Тому абсолютно стійкі шифри використовуються тільки в межах зв'язку з невеликим об'ємом інформації, що передається. Зазвичай, ці мережі використовуються для передачі особливо важливої державної інформації.

Стійкість **доведено стійких криптоалгоритмів** визначається складністю розв'язання відомої математичної задачі. Приклад таких криптосистем: Діффі-Хеллмана – складність дискретного логарифмування, RSA – складність розкладу великого числа на множники та інші.

Припустимо стійкі криптоалгоритми засновані на складності розв'язання частинної математичної задачі. Приклади: FEAL, AES та інші. Ці шифри характеризує порівняно мала вивченість математичних задач, на яких базується їх стійкість. Однак, такі шифри мають велику гнучкість, що дозволяє при визначенні слабих місць не відмовлятися від алгоритму, а провести його доробку.

15.2. Універсальні методи криптоаналізу

Метод повного перебору ключа.

Часто криптоаналітики розкривають шифри на ЕОМ методом перебору ключів. У процесі криптоаналізу доводиться перебирати більше міліарду ключів зі швидкістю тисяча ключів у секунду.

Якщо впорядкувати множину ключів K={k_1, k_2,…,k_n} то повний перебір потребує $|K|$ операцій, де перевірка одного ключа одна операція. Ймовірність правильного вибору ключа – $\frac{1}{|K|}$.

Алгоритми повного перебору допускають розпаралелення, що дозволяє значно прискорити процес знаходження ключа. Є різні підходи до реалізації розпаралелення: метод конвеєру, поділ задачі, «китайська лотерея», «криптографічні водорості».

Атака на ключі.

Однією із вразливостей криптосистем є використання слабих ключів, які не забезпечують достатнього рівня захисту. Відповідно до вже згаданого принципу Кіркхгофа, стійкість криптоалгоритма визначається секретністю ключа. Слабкий ключ – це ключ, не забезпечує достатнього рівня захисту, чи використовує в шифруванні закономірності, які можуть бути зламані. Зазвичай вважається, що алгоритм симетричного шифрування повинен по можливості не мати слабких ключів. Якщо це неможливо, то кількість слабких ключів має бути мінімальною. Тим не менш, всі слабкі ключі, якщо їх не можливо уникнути, мають бути відомими (як у алгоритмі DES).

Генератор випадкових чисел – ще одне слабе місце криптосистем. Це означає, що, якщо для генерування ключів використовується криптографічний слабкий алгоритм, незалежно від використовуваного шифру вся система буде нестійкою. Якісний ключ, призначений для використання в рамках симетричної криптосистеми, являє собою випадковий двійковий набір. Якщо потрібно ключ розрядністю n , в процесі його генерування з однаковою ймовірністю повинен виходити з будь-яких 2^n можливих варіантів. Генерування ключів для асиметричних криптосистем – процедура складна, так як ключі, вживані в таких системах, повинні володіти певними математичними властивостями. Наприклад, у випадку системи RSA модуль шифрування являє собою добуток двох великих простих чисел.

За допомогою генераторів псевдовипадкових можна будувати стійкі криптосистеми. Хороші генератори випадкових чисел складні в розробці, так як їх надійність часто залежить від особливостей апаратного та програмного забезпечення.

Частотний аналіз.

Визначення окремих символів і їх сполучень. Цей аналіз використовує статистичні і лінгвістичні методи для отримання додаткової інформації про ключ.

Лінійний аналіз.

Заміна нелінійної функції криптографічного алгоритму деяким лінійним аналогом. Найчастіше застосовується для симетричних алгоритмів.

Різницевий аналіз.

Різницевий або диференціальний аналіз використовує аналіз пари відкритих текстів, що мають визначені відмінності, після їх шифрування. Використовується для блочних систем і загальним методом.

Методи безключового читання.

Метод читання в колонках реалізує спосіб відновлення тексту, зашифрованого нерівноймовірною гамою.

Метод Полларда. Метод колізій для хеш-функцій. Метод «зустрічі посередині» та інші.

15.3. Первинний аналіз шифровки

Опишемо деякі найбільш прості та ефективні методики розкриття зашифрованого тексту. Ці методики, звичайно, не дають гарантії розкриття будь-якого шифру, бо криптоаналіз це скоріше мистецтво, засноване на інтуїції і досвіді, ніж точна наука, заснована на законах і формулах. Однак ці методики слугують інструментом злому, свого роду набором відмичок, які можуть принести результат лише в руках досвідченого майстра. Деякі методики застосовуються для розкриття ручних шифрів, а деякі для розкриття машинних. В обох випадках злом зазвичай відбувається з використанням обчислювальної техніки, що дозволяє позбавитися від рутинного перебору ключів за ручним методом. При цьому вдалий лозунг «працювати повинна машина, а думати повинна людина». При традиційному криптоаналізі, результат роботи часто визначається кваліфікацією зломщика, а при машинному − швидкістю обчислювальної системи. Оптимальне співвідношення цих двох факторів є необхідною умовою успіху крипто аналізу [5].

Завжди процес криптоаналізу починається зі збору різноманітних відомостей про зашифрований документ:

– на якій мові написаний оригінал;

– які лінгвістичні особливості даної мови (вживання артиклів і займенників, узгодження відмінків, правила утворення суфіксів і префіксів, порядок слідування слів у реченні, ...);

– яка мінімальна змістовна одиниця інформації (біт, символ, слово,...);

– які імовірно слова і фрази можуть знаходитися в тексті і в якій послідовності;

– яка приблизно довжина вихідного тексту;

– які ймовірно методи шифрування могли бути застосовані;

– яка службова інформація може знаходитися в тексті (контрольна сума пакета, адресна інформація, дата передачі);

– на який апаратурі шифрувався текст.

Такі і подібні дані збираються агентурними або аналітичними шляхами, що часто значно полегшує поставлене перед зломщиком задачу.

Первинний аналіз шифровки – дія, яка проводиться завжди при зломі будь-якого шифру. Так як будь-яка перехоплена шифровка (папірус з ієрогліфами, клаптик паперу з колонками цифр, повідомлення на морзянці, закодований файл або протокол активності комп'ютерної мережі) в кінцевому підсумку набрана за допомогою певного алфавіту (ієрогліфічного, цифрового, літерного, байтового), то можна, по-перше, оцінити інформативність всього повідомлення, по-друге, проаналізувати діаграму розподілу ймовірностей появи окремих символів алфавіту в тексті.

Інформативність повідомлення в тому сенсі, який запропонував Шеннон, визначається за формулі $H=\left|\sum P_i \cdot \log_2(P_i)\right|$, де P_i - частота появи і-го символу в тексті. Простіше кажучи, чим вище інформативність, тим більше щільність корисної інформації в обсязі повідомлення.

Другим пунктом після визначення інформативності шифровки є побудова **діаграми розподілу ймовірностей прояву окремих символів в тексті шифровки**. Переважно діаграму представляють у вигляді лінійного графіка, де по одній осі відкладені порядкові номери символів в алфавіті, а по інший нормалізовані ймовірності прояву цих символів у тексті шифровки.

Ймовірності появи окремих символів англійської, української, російської мов стандартного літературного тексту є у вільному доступі. Ймовірності наведені в процентному поданні та округлені до цілого.

Твердження про те, що ймовірність прояву чергового символу в середньостатистичному тексті є величина незалежна неправильно, так як деяких поєднань символів в мові взагалі може не бути. Правильніше сказати, що ймовірність появи певного символу в черговий позиції тексту залежить від попереднього символу. Якщо припустити, що алфавіт повідомлення складається з символів української мови і пробілу, то неважко вивести таблицю ймовірностей появи конкретних біграм в довільному українському тексті. Аналогічно з українськими біграмами, можна скласти таблицю ймовірностей прояву англійських біграм в англомовному тексті та інше.

Крім біграм, можна побудувати аналогічні таблиці для триграм, тетраграмма і т.д.

Отже, після всього сказаного, можна зробити висновок, що діаграма розподілу ймовірностей прояву символів і інформативність можуть дати важливі непрямі відомості про мову шифровки або хоча б про можливий тип шифрування [5].

15.4. Конкретні приклади криптоаналізу

Якщо за результатами первинного аналізу шифровки був успішно визначений метод шифрування, то можна вважати, що ми зробили близько 20% всієї роботи. Якщо методами шифрування виявилися різні види підстановок, перестановок шифрування біграм або шифрування по системі Віжінера, а довжина шифровки досить велика, то можна сказати, що дешифрування

вихідного тексту діло техніки. Якщо найбільш імовірним методом шифрування виявилося гамування, а ще гірше взбивка, то доведеться визнати, що робота по розкриттю шрифту тільки починається і радіти ще дуже рано.

Ми ніколи не доб'ємося розкриття шрифту, якщо в результаті первинного аналізу шифровки ми зробили неправильне припущення про метод шифрування.

15.4.1. Криптоаналіз шифрів за методом перестановок

Перед описом методу злому шифрування за перестановкою перерахуємо кілька властивостей даного методу шифрування, якими треба буде скористатися.

По-перше, якою б не була перестановка, у неї завжди є максимальний радіус. Іншими словами, символ може переміщатися з поточної позиції не більше ніж на відстань R вперед або назад.

По-друге, для розрахунку зсуву конкретного символу відносно його положення в початковому тексті зазвичай використовується проста математична формула, а не повна матриця перестановок, так як в протилежному випадку обсяг ключової інформації для шифрування буде близьким до об'єму вихідного тексту.

І по-третє, математична формула для перестановки символів повинна бути однозначною, тобто кожному символу в початковому тексті повинен відповідати тільки один символ тексту шифровки і не повинно бути накладок.

Розкриття шифру перестановки зазвичай здійснюють перебором можливих формул і аналізом одержуваного відкритого тексту. В якості аргументів у формулах фігурують значення поточної позиції **i** і максимального радіуса **R**. В якості операцій над аргументами у формулах можуть використовуватися додавання, віднімання, побітового операції, множення, зведення в ступінь, взяття залишку за модулем **R,** а також різні комбінації цих операцій.

Під час розв’язання таких задач необхідно відновити початковий порядок слідування букв тексту. Для цього використовують аналіз сумісності символів та таблиці частот біграм відповідної мови для першого рядка таблиці.

Приклад 1.

Розшифрувати текст «ИСВИЇПНКУАПЬОТІРІУВКИТПАЦ», зашифрований стовпцевою перестановкою, без ключа (ключем є розмір таблиці та порядок перестановки стовпців).

Текст містить 25 символів, тому, ймовірно, можемо розглядати таблицю розміром 5x5. Таким чином, запишемо текст по СТОВПЦЯХ у таблицю. Можемо припустити, що мова на якій написаний текст українська(буква Ї).

И	П	П	Р	И
С	Н	Ь	І	Т
В	К	О	У	П
И	У	Т	В	А
Ї	А	І	К	Ц

Розглянемо у першому рядку можливі біграми і згідно додатку А запишемо їх ймовірності:

ПР– **0,0071** ПИ-0, 0011 РИ – **0,0057**

ИР – 0,0017 РП – 0,0001 ИП – 0,0007

Із даного аналізу бачимо, що найбільш можливими є сполучення: ПР, РИ. Тому перший рядок, ймовірно, буде – ПРИПИ. Важливо визначити прядок букв, які повторюються, наприклад:

И	П	П	Р	И
5	1	4	2	3

Переставимо стовпці згідно цього припущення:

П	Р	И	П	И
Н	І	Т	Ь	С
К	У	П	О	В
У	В	А	Т	И
А	К	Ц	І	Ї

У результаті цієї перестановки ми отримали розшифрований текст: **ПРИПИНІТЬ СКУПОВУВАТИ АКЦІЇ**.

15.4.2. Криптоаналіз шифрів за методом простої заміни.

Дешифрування проводиться в 7 етапів:

1) В якості одиниці інформації приймається 1 символ для простої підстановки або 2 символи для шифрування біграм.

2) Будується діаграма розподілу примітивних елементів для шифрованого тексту.

3) Побудована діаграма порівнюється з діаграмою розподілу примітивних елементів стандартного літературного тексту.

4) Вручну або автоматично робиться припущення про відповідність примітивних елементів вихідного тексту і тексту шифровки, причому аналіз починається з елементів, які найчастіше зустрічаються (найбільш високі піки на графіку), а закінчується елементами, які найрідше зустрічаються.

5) Після зворотної заміни примітивних елементів шифрованого тексту на відповідні їм примітивні елементи вихідного тексту, виходить досить читабельний текст вихідного повідомлення.

6) Якщо вихідне повідомлення містить помилки, або зовсім не схоже на зв'язний осмислений текст, то це значить, що ми зробили помилкове припущення про відповідності елементів і слід повернутися до пункту 4.

7) Аналіз закінчується тоді, коли отриманий текст не містить помилок та є осмисленим.

Підстановки на основі простої заміни не приховують статистичні властивості тексту. Тому криптоаналіз шифру простої заміни заснований на використанні статистичних закономірностей мови (в інтернет-джерелах такі таблиці є у вільному доступі). Додатково можна використовувати аналіз k-грам (в основному біграм), враховувати властивості мови (суфікси, префікси, подвоєння, сполучники та інше).

Приклад 1.

Розшифрувати текст, зашифрований за допомогою простої заміни (російський алфавіт):

29 15 10 17 29 22 25 31 15 33 35 41 43 45 35 57 45 25 17 59 15 10 25 41 25 69, 59 78 29 82 25 78 25 17 15 10 88 90 78 25 62 25 22 10 57 73 79 35 67 78 90 88 29 45 35 29, 54 57 90 31 90 73 22 88 15 88 29 15 17 69 41 25 15, 70 17 90 57 43 59 15 78 15 62 22 25 17 57 25 69 88 15 82 17 25 88 29 45 35…

Підрахуємо частоту кожного слова у шифровці:

29	**15**	10	17	22	**25**	31	33	35	41	43	45	57	59	69	78	82	88	90	62	73	79	67	54	70
7	**10**	4	7	4	**12**	2	1	5	3	2	4	5	3	3	4	2	6	5	1	2	1	1	1	1

Із таблиці частот російської мови мови (в інтернет-джерелах такі таблиці є у вільному доступі) найчастіше зустрічається буква «о» та «е». Тому припустимо що 25=«о», 15=«е» (може бути і навпаки). Слово у третьму рядку перед другою комою може закінчуватися на «ое» або «ео». Із таблиці біграм бачимо, що найчастіше зустрічається «ое» – частота 15, а у «ео» – 7. Тому наші припущення вірні: **25=«о», 15=«е».**

Тепер оцінимо букву 29 перед першою комою у третьому рядку, швидше за все це може бути голосні буква. Так як вона зустрічається 7 разів у тесті, що є середньою величиною, то це може бути буква «а» або «и». Тоді в останньому рядку 25 88 29, 88 – приголосна буква, згідно частот, ймовірно «т» або «н». Отже, у третьому рядку 22 88 15 88 29 15 маємо: *22 (т, н) е (т, н) (а, и) е* – найочевидніше – «мнение». Тому **22=«м», 88=«н», 29=«и».**

У першому рядку 29 15 10 17 29 22 маємо *и е 10 17 и м о*. Робимо висновки, що 10 і 17 приголосні, згідно таблиці частот це можуть бути «с» та «л». Звідси слово – «если». Тому **10=«с», 17=«л».**

У другому рядку 59 78 29 82 25 – *58 78 и 82 о.* 58 і78 можливо приголосні, наприклад – «при». **58=«п», 78=«р».**

У першому рядку 45 25 17 59 15 – *45 о л п е*. 45 може бути «т», «в», «з», «б», «г», «ч». Із усіх випадків підходить «т» – *толпе*. **45=«т».**

Продовжуючи таким чином аналіз ще кількох слів отримаємо розшифрований текст:

и если можешь быть в толпе собою, при короле с народом свьязь хранить и, уважая мнение любое, главы перед молвою неклонить.

15.4.3. Криптоаналіз шифру Цезаря

Криптосистема Цезаря визначається рівнянням $y_i = (x_i + k) \bmod m$, де $i = \overline{1,n}$, y_i – порядок букви в алфавіті криптограми, x_i – порядок букви в алфавіті відкритого тексту, k – ключ шифру, який може змінюватися від *1* до m, n – довжина криптограми, m – потужність алфавіту (кількість букв).

Ключ можна знайти за допомогою декількох методів:

– Частотного аналізу, знайти букву, яка найчастіше зустрічається у шифрі і порівняти відстань від неї до букви, яка найчастіше зустрічається у алфавіті відповідної мови.

– Здійснити атаку повним перебором на множині всіх можливих ключів.

– Обчислити ключ за допомогою формули:

$k^* = \arg\max_k l(k)$, $l(k) = \sum_{j=0}^{m-1} v_{(j-k) \bmod m} \cdot \log_2 p(j)$, v_j – частота j-ої букви у криптограмі, $p(j)$ – ймовірність j-ої букви у алфавіті.

Приклад 3.

Розшифрувати криптограму зашифровану за допомогою шифру Цезаря.

«pelcgbybtlvfpelcgbtenculnaqpelcgbnanylfvf»

Перетворимо її у числову, нумерація букв починається із нуля:

«15 4 11 2 6 1 24 1 19 11 21 5 15 4 11 2 6 1 19 4 13 2 20 11 12 0 16 15 4 11 2 6 1 13 0 13 24 11 5 21 5».

Складемо таблицю ймовірнісних характеристик англійського алфавіту

буква	a	b	c	d	e	f	g	h	i	j	k	l	m	n
j	0	1	2	3	4	5	6	7	8	9	10	11	12	13
$\text{Log}_2 p(j)$	-3.6	-6	-5	-4.7	-3	-5.4	-5.7	-4.2	-3.8	-9.3	-7.2	-4.6	-5.3	-3.8
буква	o	p	q	r	s	t	u	v	w	x	y	z		
j	14	15	16	17	18	19	20	21	22	23	24	25		
$\text{Log}_2 p(j)$	-3.7	-5.6	-9.8	-4	-3.9	-3.4	-5.2	-6.7	-5.7	-9	-5.9	-10.1		

Знайдемо частотні характеристики криптограми

буква	a	b	c	d	e	f	g	h	i	j	k	l	m	n
j	0	1	2	3	4	5	6	7	8	9	10	11	12	13
v_j	2	4	4	0	4	3	3	0	0	0	0	6	0	4
буква	o	p	q	r	s	t	u	v	w	x	y	z		
j	14	15	16	17	18	19	20	21	22	23	24	25		
v_j	0	3	1	0	0	2	1	2	0	0	2	0		

Обчисливши та побудувавши графік логарифмічної функції правдоподібності $l(k)$, знайдемо ключ рівний 13.

k	0	1	2	3	4	5	6	7	8
L(k)	-203,5	-230,8	-193,4	-219,8	-234,2	-258,6	-207,3	-211,2	-235,2

k		….	…..	….	12	**13**	14	15	…..
L(k)		**....**	…..	…..	-260,2	**-185,4**	-253,5	-211,8	……

Зсунемо всі букви криптограми на 13 позицій і отримаємо вихідний текст:

cryptologyis cruptography and cryptoanalysis

Тестові завдання для розділів 7-15

1. Дайте визначення криптографії:
   1) криптографія - це наука про захист інформації від несанкціонованого доступу сторонніх осіб;
   2) криптографія - наука про захист інформації від прочитання її сторонніми особами, що досягається шляхом шифрування і робить захищені дані марними для зловмисника;
   3) криптографія - це наука про захист інформації за допомогою математичних перетворень, які є симетричними;
   4) криптографія - це наука про захист інформації від несанкціонованого копіювання інформації;
   5) інша відповідь.
2. Яка процедура розподілу ключів не потребує захищеного каналу для передачі секретного ключа адресату?
   1) алгоритм Діффі-Хеллмана;
   2) алгоритм RSA;
   3) алгоритм DES;
   4) алгоритм RC-4;
   5) алгоритм DSA.
3. Для визначення якості шифру використовується поняття ...
   1) криптостійкості;
   2) надійності;
   3) невразливості;
   4) довіреності;
   5) інша відповідь.

$4. Яку довжину має секретний ключ криптосистеми DES?
   1) 56 біт із врахуванням контрольних розрядів;
   2) 64 біт із врахуванням контрольних розрядів;
   3) 128 біт;
   4) 32 біти;
   5) інша відповідь.
5. Секретна система - це:
   1) сукупність методів і алгоритмів шифрування, які забезпечують можливість криптографічного захисту даних;
   2) деяка множина відображень одного простору (множини можливих повідомлень) в інший простір (множини можливих криптограм), де кожне конкретне відображення з цієї множини відповідає способу шифрування за допомогою конкретного ключа;
   3) деяка множина відображень одного простору (множини можливих повідомлень) в інший простір (множини можливих криптограм), де кожне конкретне відображення з цієї множини відповідає способу шифрування за допомогою декількох ключів;
   4) системи маскування даних;
   5) інша відповідь.
6. Заміна смислових конструкцій вихідної інформації (слів, речень) кодами називається:
   1) шифруванням;

2) дешифруванням;
3) стисненням;
4) кодуванням;
5) маскуванням.

7. Методи, які дозволяють приховати не тільки зміст, що зберігається або передається, а й сам факт зберігання або передачі секретної інформації відносяться до методів:
1) шифрування;
2) кодування;
3) стиснення;
4) дешифрування;
5) стеганографія.

8. Процес дешифрування закритої інформації без знання ключа і, можливо, при відсутності відомостей про алгоритм шифрування називається:
1) повторним шифруванням;
2) імовірнісним дешифруванням;
3) криптоаналізом;
4) зворотним шифруванням;
5) дешифруванням.

9. Перші спроби шифрування інформації робилися:
1) в середині XX століття;
2) в кінці XVIII століття;
3) в V-VI столітті д.н.е;
4) на початку XII століття;
5) інша відповідь.

10. Виберіть правильне розташування систем шифрування в хронологічному порядку їх появи:
1) Шифр скітала. Шифр Віжінера. Шифр Цезаря. Квадрат Полібія.
2) Шифр Цезаря. Шифр скітала. Квадрат Полібія.. Шифр Віжінера.
3) Шифр скітала. Шифр Цезаря. Квадрат Полібія. Шифр Віжінера.
4) Квадрат Полібія. Шифр скітала. Шифр Цезаря. Шифр Віжінера.
5) Шифр скітала. Квадрат Полібія. Шифр Цезаря. Шифр Віжінера.

11. За характером використання ключа всі криптосистеми можна розділити на:
1) блокові і потокові;
2) синхронні і асинхронні;
3) симетричні і асиметричні;
4) підстановки і перестановки;
5) моноалфавітні і поліалфавітні;

12. У симетричних криптосистемах:
1) для шифрування і дешифрування завжди використовується різні алгоритми;
2) для шифрування і дешифрування використовуються різні ключі;
3) як для шифрування, так і для дешифрування застосовується один і той же ключ;
4) ключ може бути доступним для всіх користувачів;
5) використовуються два ключі: відкритий і секретний.

13. Найбільш відомим представником асиметричних систем шифрування є:
1) алгоритм Діффі-Хеллмана;
2) алгоритм RSA;
3) алгоритм DES;

4) алгоритм RC-4;
5) алгоритм DSA.

14. У потокових шифрах основною операцією шифрування є:
1) матричні перетворення;
2) перетворення, засновані на обчисленнях з плаваючою точкою;
3) обчислення логарифма в скінченому полі;
4) операція додавання за модулем два;
5) операція множення за модулем два.

15. У квадратичному конгруентному генераторі виду $X_n = (aX^2_{n-1}+bX_{n-1}+c) \bmod m$ використовуються наступні значення: a = 2, b = 4, c = 21, m = 12, $X_0 = 2$. Перший член послідовності такого генератора буде дорівнює …..
1) 1;
2) 3;
3) 4;
4) 5;
5) 2.

16. Лінійний конгруентний генератор виду $X_n = (aX_{n-1}+b) \bmod m$ використовує наступні значення: a =5, b = 7, $X_0 = 4$. Перший член послідовності такого генератора вийшов рівним 7. Мінімальне значення m дорівнює:
1) 2;
2) 4;
3) 7;
4) 10;
5) 3.

17. Результатом шифрування повідомлення M=2 за алгоритмом RSA для наступних параметрів: p=7, q=11, e=13, d=37 буде C=….
1) 30;
2) 3;
3) 6;
4) 4;
5) Інша відповідь.

18. До генераторів випадкових чисел належить:
1) метод квадратів;
2) змішаний конгруентний;
3) вимірювання температури зовнішнього середовища;
4) Фібоначчі;
5) BBS.

19. DES-алгоритм має ….. раунди (цикли) роботи:
1) 28;
2) 16;
3) 8;
4) 32;
5) інша відповідь.
3) протоколом аутентифікації й ідентифікації;
4) криптографічним протоколом;
5) інша відповідь.

20. Найбільш відомим представником систем потокового шифрування є:
1) алгоритм Діффі-Хеллмана;

2) алгоритм RSA;
3) алгоритм DSA;
4) алгоритм DES;
5) алгоритм RC4.

21. Результатом розшифрування повідомлення C=2, яке зашифроване за алгоритмом RSA із наступними параметрами: p=3, q=11, e=7, d=3 буде M=….
1) 12;
2) 4;
3) 8;
4) 15;.
5) інша відповідь.

22 .Хеш-функцією називається …
1) шифрування даних;
2) код аутентифікації;
3) цифровий електронний підпис;
4) криптографічні перетворення послідовності довільної довжини у послідовність фіксованої довжини;
5) дешифрування даних.

23. Вихідний хеш-образ при використанні алгоритму SHA-1 має довжину….
1) 64 біт;
2) 160 біт;
3) 128 біт;
4) 56 біт;
5) 256 біт.

24. Який алгоритм із нижче наведених використовується винятково для генерації цифрового електронного підпису?
1) RSA;.
2) DES;
3) DSA;
4) MD;
5) SHA.

25. Який із нижче вказаних алгоритмів не використовується для шифрування повідомлень?
1) SHA;
2) Діффі-Хеллмана;
3) DES;
4) MD;
5) RSA.

26. Розв’язком рівняння $9x = 7(\mathrm{mod}10)$ є число:
1) 3;
2) 2;
3) 9;
4) 5
5) 7.

27. Алгоритм Ель-Гамаля використовується для:
1) генерування ключів;
2) кодування інформації;
3) розподілу ключів;

4) хеш-перетворень;
5) шифрування інформації.

28. Результатом шифрування повідомлення «підпис» за алгоритмом Цезаря із зсувом на 3 позиції вправо при використанні українського алфавіту без букви ґ буде:
1) фтжклр;
2) лепові;
3) ищлтів;
4) ппаідл;
5) ткжтйф.

29. начення функції Ейлера $\varphi(17)$ буде:
1) 21;
2) 17;
3) 18;
4) 289;
5) 16.

30. Значення функції Ейлера $\varphi(8)$ буде:
1) 5;
2) 9;
3) 7;
4) 4;
5) 3;

ЛІТЕРАТУРА

1. Безбогов А. А. Методы и средства защиты компьютерной информации: учебное пособие / А. А. Безбогов, А. В. Яковлев, В. Н. Шамкин. – Тамбов : Изд-во Тамб. гос. техн. ун-та, 2006. – 196 с.

2. Гатчин Ю. А. Основы криптографических алгоритмов. Учебное пособие. / Ю. А. Гатчин, А. Г. Коробейников. – ГИТМО(ТУ), 2002. – 29 с.

3. Грайворонський М. В. Безпека інформаційно-комунікаційних систем / М. В. Грайворонський, О. М. Новіков – К.: Видавнича група BHV, 2009. – 608 с.

4. Гребенніков В.В. Історія криптології & секретного зв'язку.- Офіційна веб-сторінка книги: www.cryptohistory.ru.

5. Гундарь К. Ю. Защита информации в компьютерных системах / К. Ю. Гундарь, А. Ю. Гундарь, Д. А. Янишевський. – К.: «Корнейчук», С. 200.– 152.

6. Єфіменко А. А. Порівняльний аналіз алгоритму симетричного блокового перетворення «Калина» (ДСТУ 7624:2014) з іншими міжнародними стандартами шифрування даних / А.А. Єфіменко, Є. М. Байлюк, О. А. Покотило //Збірник наукових праць ЖВІ. Випуск 15, С. 156-162.

7. Иванов М. А. Теория, применение и оценка качества генераторов псевдослучайных последовательностей / М. А. Иванов, И. В. Чугунков. – М.: Изд-во «КУДИЦ-ОБРАЗ», 2003. – 240 с.

8. Коркішко Т. Алгоритми та процесори симетричного блокового шифрування / Т. Коркішко, А. Мельник, В. Мельник. – Львів. БаК, 2003. – 168 с.

9. Остапов С. Е. Основи криптографії: навчальний посібник / С. Е. Остапов, Л. О. Валь. – Чернівці: Книги–XXI, 2008. – 188 с.

10. Романец Ю. В. Защита информации в компьютерных сетях и системах / Ю. В. Романец, П. А. Тимофеев, В. Ф. Шаньгин. – М.: Радио и вязь, 2001. – 376 с.

11. Харин Ю. С. Математические и компьютерные основы криптологии: учеб. пособие / Ю. С. Харин, В. И. Берник, Г. В. Матвеев. – Минск: Новое знание, 1999. – 319 с.

12. Шнайер Б. Прикладная криптография: Протоколы, алгоритмы и исходные тексты на языке С / Б. Шнайер.− М. : Триумф, 2002. – 816 с.

13. Яковлев А. В. Криптографическая защита информации : учебное пособие/ А. В. Яковлев, А. А. Безбогов, В. В. Родин, В. Н. Шамкин. – Тамбов : Изд-во Тамб. гос. техн. ун-та, 2006. – 140 с.

14. Гапак О. М. Визначення довжини періоду генераторів псевдовипадкових послідовностей на основі регістрів зсуву зі зворотнім зв'язком та перенесення / О.М. Гапак // Моделювання та інформаційні технології. – 2014. – Випуск 73. – С. 92 – 97.

15. NIST SP 800-22rev1a. A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications//National Institute of Standards and Technology Special Publication 800-22rev1a, 2010, - 131 p.