ЗМІСТ

# ВСТУП

На сучасному етапі розвитку індустріально-інформаційного суспільства все більшого значення набувають автоматизовані системи передачі, зберігання і обробки інформації, основані на використанні комп'ютерних технологій.

Такі системи вимагають постійного вдосконалення, а їх поширення в глобальних масштабах приводить до необхідності уніфікації принципів побудови і вживання відповідних технічних засобів.

З розвитком телекомунікацій стали доступними зручні засоби, які реалізовують обмін електронними даними, що привело до формування сфери електронного документообігу.

При побудові систем електронного документообігу на основі каналів зв'язку загального призначення виникають специфічні вимоги, пов'язані з необхідністю захисту інформації, такі як забезпечення конфіденційності, цілісності, авторства даних і так далі. Ці вимоги ускладнюються за умови недовіри учасників інформаційного обміну один до одного.

Крім того, захист інформації необхідний для забезпечення працездатності самих автоматизованих систем.

Не випадково необхідність впровадження засобів криптографічного захисту інформації (КЗІ) є одним з положень Закону України «Про електронні документи і електронний документообіг».

Слід зазначити, що стандарти, які на сьогодні діють в Україні, в області криптографічного захисту інформації не покривають весь спектр необхідних криптоалгоритмів і криптопротоколів, наприклад, відсутні стандарти на асиметричну криптосистему і систему узгодження симетричних ключів.

З іншого боку, необхідні криптоалгоритми і криптопротоколи можна побудувати на основі окремих положень вітчизняних і міжнародних стандартів, з подальшою процедурою сертифікації.

Це зумовлює до необхідність аналізу вимог і обмежень, встановлених нормативно-правовими документами, і вимагає об'єктивної оцінки можливостей організації відносно реалізації відповідних технологій.

У запропонованому навчальному посібнику розглядаються питання організації і функціонування надійних систем криптографічного захисту інформації. Викладені принципи побудови криптосистем різного типу, наведена методика генерації і оцінки якості псевдовипадкових послідовностей, а також методи генерації псевдовипадкових простих чисел.

Особлива увага приділена генерації якісних ключів на основі криптографічних генераторів псевдовипадкових послідовностей.

Наводяться описи і характеристики стійкості блокових шифрів ГОСТ 28147-89, Rijndael і асиметричних криптоалгоритмів, розглянуті принципи організації, функціонування і забезпечення надійності інфраструктури відкритих ключів.

Метою посібника є виклад матеріалу, використання якого передбачається на етапі розробки концепції системи КЗІ електронного документообігу організації.

Посібник призначений для студентів старших курсів вищих навчальних закладів і аспірантів, знайомих з елементарною теорією чисел (включаючи теорію символу Якобі), а також фахівців, що займаються впровадженням засобів КЗІ.

Автори щиро вдячні докторові технічних наук, професорові Кузнєцову Георгію Віталійовичу, докторові технічних наук, професорові Скрипникові Леоніду Васильовичу і докторові технічних наук, професорові Шелесту Михайлу Євгеновичу за доброзичливе і уважне ставлення до запропонованої книги, а також за зауваження і рекомендації, що сприяли значному покращенню матеріалу.

Ми вдячні НВФ «КРИПТОН» і ТОВ «ТРИТЕЛ» за надану можливість використання в книзі матеріалів за виробами цих фірм.

Ми виражаємо також особливу подяку всім авторам, роботи яких були використані при підготовці даного посібника і допомогли розширити тематику, порушену в книзі.

# ГЛАВА 1 ОСНОВНІ ПОНЯТТЯ І ЗАДАЧІ КРИПТОЛОГІЇ

## 1.1 Предмет криптології, криптографія і криптоаналіз

Історія криптографії – ровесниця історії писемності, з її широким розповсюдженням криптографія сформувалася як мистецтво тайнопису. Згадки про перші шифри зустрічаються вже на початку нашої ери. Так, римський імператор Гай Юлій Цезар використовував в своєму листуванні шифр, що одержав згодом його ім'я.

Постійними замовниками криптографічних систем в усі часи були дипломати і військові. Саме збільшення об'ємів інформації, що передається, і скорочення строків на її обробку у відповідних відомствах сприяло швидкому розвитку шифрувальної техніки і її впровадженню замість ручних шифрсистем.

В розвитку шифрувальної техніки можна прослідкувати декілька етапів.

З початку і до середини XX століття – використання механічних і електромеханічних шифрмашин: вироби М-94, М-138-Т4 (США), С-36 (Швеція), «Енігма» (Німеччина, рис.1.1).

Вже на цьому етапі використовувалися оригінальні і достатньо стійкі, за мірками того часу, криптоалгоритми.

Наприклад, шифратор «Енігма» реалізує оригінальний метод комутації електричних сигналів за допомогою декількох дисків, що рухаються за певним законом (рис. 1.1). Кожний диск має рівну кількість вхідних і вихідних контактів, з’єднаних попарно.

Електричний струм послідовно проходить через контактну групу натиснутої клавіші, стале з'єднання дисків, повертається через рефлектор, знов через диски і запалює індикаторну лампу, що відповідає зашифрованій букві.

В 50 - 80-х роках XX століття – застосування електронної шифртехніки, побудованої на дискретних радіоелементах і мікросхемах малого ступеня інтеграції.

Починаючи з 80-х років минулого століття – впровадження шифрувальної техніки на основі мікрокомп'ютерів і мікроконтролерів,

створення спеціалізованих мікросхем, що реалізовують криптографічні функції (зокрема, стандарти шифрування DES, RSA).

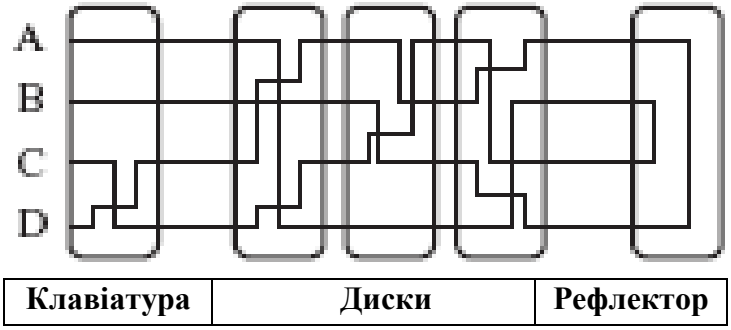



Рисунок 1.1 - Модель комутації сигналів в шифрмашині «Енігма» (символ A переходить в C, символ B – в D і навпаки)

Нові технічні можливості в 70-80 роках XX століття надали більший простір математикам для створення все більш складних для проведення криптоаналізу криптоалгоритмів. У тому числі, широке впровадження згаданих алгоритмів DES і RSA в значній мірі було зумовлено повсюдним розповсюдженням комп'ютерної техніки і впровадженням обчислювальних мереж.

З іншого боку, створення потужних комп'ютерів, технологій мережевих обчислень збільшувало ризик розкриття криптографічних систем, які ще недавно вважалися стійкими. Відзначимо, що перша в світі електронно-релейна обчислювальна машина була створена англійцями в роки другої світової війни для дешифрування німецької шифрмашини «Енігма».

Широке використання комп'ютерних мереж, зокрема, глобальної мережі Інтернет, розвиток електронних банківських технологій, збільшення об'ємів передачі інформації з обмеженим доступом державного, військового, комерційного і приватного характеру зумовили розвиток нових напрямів в криптографії, включаючи системи відкритого розподілу ключів і системи електронного цифрового підпису. На сьогодні важко знайти інформаційну (ІС) або інформаційно-телекомунікаційну систему (ІТС), в якій не застосовувалися б механізми криптографічного захисту інформації.

Сучасна шифрувальна техніка характеризується малими габаритами і вагою, високою швидкістю шифрування, надійним захистом інформації з обмеженим доступом (рис. 1.2 - 1.6).

Рисунок 1.2 - Засіб шифрування «Кокон»

Зокрема, засіб криптографічного захисту «Кокон» (рис. 1.2) використовується для захисту конфіденційної інформації в системах електронної пошти. Шифратор підключається до комп'ютера через USB-порт за допомогою спеціальної утиліти. Кожне повідомлення перед передачею по системі електронної пошти заздалегідь шифрується згідно з криптоалгоритмом, визначеним стандартом ГОСТ 28147-89.

Рисунок 1.3 - Абонентні термінали – телефонні шифратори

Показані на рис. 1.3 телефонні термінали для мереж загального користування «СЕКМОД» і «Ірпінь» забезпечують цифрову обробку

мовного повідомлення, автоматичний захищений обмін ключовою інформацією і шифрування мови з швидкістю до 9.6 Кбіт/сек.

Шифратор “Топаз-8000” (рис. 1.4) використовується для передачі даних в захищеному вигляді по телефонних мережах загального користування. Швидкість шифрування інформації до 115 Кбіт/сек. Виріб включається між комп'ютером (інтерфейс USB 1.1 або USB 2.0) і модемом (через з'єднувач DB-9 інтерфейсу RS-232).


Рисунок 1.4 - Шифратор для систем передачі даних «Топаз-8000»

Шифратор для систем передачі даних Д-300 (рис. 1.5) шифрує і розшифровує потоки даних із швидкістю 2 Мбіт/сек. Дана апаратура має розвинену систему діагностики технічного стану і тестування працездатності.


Рисунок 1.5 - Шифратор цифрових потоків Д-300

Апаратно-програмний комплекс «Пелена» (рис. 1.6) призначений для криптографічного захисту конфіденційної інформації у відомчих (корпоративних) мережах, побудованих на базі технологій IP (протокол IP

v.4). Інтерфейси комплексу відповідають рекомендаціям Ethernet IEEE 802.3-2002 100Base-TX/FX. Швидкість обробки інформації до 70 Мбіт/сек.


Рисунок 1.6 - Шифратор протоколу IP «Пелена»

Незалежно від призначення, експлуатаційних і технічних характеристик перераховані шифрувальні засоби об'єднує головна властивість: їх функціонування визначається алгоритмами захисту інформації – криптографічними перетвореннями відкритих повідомлень, які реалізовані програмним або апаратним способом. Відповідні перетворення для досягнення заданого рівня безпеки захищуваної інформації повинні відповідати деякій кількості вимог.

Криптографічні методи захисту інформації засновані на математичних перетвореннях інформації з використанням секретних параметрів – ключових даних.

Розробка подібних методів, проблеми аналізу, синтезу відповідних алгоритмів, а також оцінка їх якості складають предмет науки **криптології** (kryptos – таємний, logos - наука). Криптологія розділяється на два наукові напрями – **криптографію і криптографічний аналіз** (коротко – криптоаналіз).

Цілі цих напрямів прямо протилежні.

**Криптографія** досліджує властивості і принципи побудови алгоритмів перетворення інформації, що забезпечують її надійний захист.

Основні області застосовування криптографічних методів - це:

– забезпечення конфіденційності інформації шляхом шифрування при її передачі по відкритих каналах зв'язку (наприклад, в системі електронної пошти) або зберіганні на носіях,

– реалізація механізмів контролю цілісності і автентичності критичних даних, підтвердження авторства одержаних повідомлень шляхом формування деяких контрольних записів – цифрових сигнатур.

Предметом **криптоаналізу** є дослідження і практична реалізація інженерно-математичних методів відновлення первинного вигляду зашифрованої інформації без знання секретних параметрів перетворень (методів дешифрування).

Конкретну процедуру відновлення інформації або розкриття шифру називають криптоаналітичною атакою.

Властивість криптосистеми протистояти криптоаналітичним атакам називається **криптографічною стійкістю криптосистеми.**

Подальший виклад курсу торкається, в основному, криптографії, а методи криптоаналізу розглядаються лише в тому обсязі, який необхідний для розуміння суті механізмів захисту, а також для оцінки їх надійності.

Криптографічні перетворення застосовуються для вирішення таких основних задач захисту інформації:

1) забезпечення конфіденційності інформації, захисту від несанкціонованого ознайомлення з її змістом. В рамках даного курсу лекцій поняття «конфіденційний», якщо особливо не обумовлене інше, ототожнюється з поняттям «секретний»;

2) контроль цілісності інформації, уникнення несанкціонованих її змін шляхом вставки, видалення або заміни фрагментів;

3) аутентифікація суб'єктів і ідентифікація об'єктів інформаційного обміну, підтвердження істинності сторін, самої інформації, часу її створення і т.д;

4) підтвердження авторства, забезпечення неможливості відмови.

Об'єктами досліджень і розробок сучасної криптографії є:

1) симетричні криптографічні системи (криптосистеми);

2) криптосистеми з відкритим ключем;

3) криптографічні протоколи, включаючи системи електронного підпису і системи розповсюдження ключів;

4) хеш-функції і коди аутентифікації повідомлень.

Криптографічними процедурами, що часто використовуються, є зашифрування і розшифрування – пряме і обернене перетворення. **Зашифрування** – це обернене перетворення вихідного (інакше – **відкритого**) тексту $T$ в **шифрований** текст $S$ з використанням конфіденційних даних $K$ відповідно до деякого алгоритму $E$ (від англ. encryption – шифрування). Параметр $K$ в криптографії називається

ключем. Ключ є тією інформацією, без якої неможливе відновлення початкового повідомлення. Рівняння шифрування записують у вигляді:

$$S = E(T, K) \text{ або } S = E_K(T).$$

Останній вираз підкреслює те, що за допомогою ключа вибирається деяке єдине перетворення з параметричного сімейства перетворень.

**Розшифрування** (не плутати з дешифруванням) – обернена відносно шифрування процедура $D$ (від англ. decryption – розшифрування), в результаті виконання якої шифрований текст з використанням ключа перетвориться у вихідний:

$$T = D(S, K) \text{ або } T = D_K(S).$$

Відзначимо, що в деяких випадках зручніше користуватися однією з еквівалентних форм запису: $D(S, K) = D_K(S) = E_K^{-1}(S)$, маючи на увазі під цим звертання відповідно алгоритму або ключа.

### 1.2 Моделі відкритого тексту

Як інформація, що підлягає криптографічному перетворенню, розглядатимуться повідомлення або тексти.

Термін **текст** або **повідомлення** означає деякий впорядкований набір елементів алфавіту, непорожньої скінченної множини символів, що використовуються для кодування інформації.

На практиці найбільш часто використовуються алфавіти:

1) бінарний – $Z_2 = \{0,1\}$;
2) шістнадцятковий – $Z_{16} = \{0, 1, \ldots, A, B, C, D, E, F\}$;
3) 26 букв латинського алфавіту – $Z_{26} = \{A, B, C, \ldots A, Y, Z\}$;
4) скорочена множина букв кирилиці – $Z_{32}$={*А, Б, В,…*};
5) множина символів, що входять в стандартні коди ASCII - $Z_{256}$.

В загальному випадку, якщо не сказано інше, будемо вважати, що алфавітом повідомлень, що підлягають зашифруванню, є множина

$Z_m = \{0,1,\ldots,m-1\}$, а величину $m$ називатимемо потужністю або модулем алфавіту вихідних текстів.

Необхідно відзначити, що реальні повідомлення для будь-якої конкретної мови, характеризуються певною надмірністю. Статистика частот зустрічання елементів повідомлень свідчить про їх нерівномірний розподіл.

Зокрема, для достатньо великого обсягу літературного тексту російською мовою буквами, що часто зустрічаються, виявляються {*о, и*}, серед букв англійської мови, що часто зустрічаються, – символи {*e,t*}.

Таблиця 1.1 - Відносні частоти зустрічання букв англійської мови (вибірка 300 000 букв газетно-журнального тексту)

| | | | | | |
|---|---|---|---|---|---|
| E | 12.7% | D | 4.2% | P | 1.9% |
| T | 9.0% | L | 4.0% | B | 1.5% |
| A | 8.2% | U | 2.8% | V | 1.0% |
| O | 7.5% | C | 2.8% | K | 0.8% |
| I | 7.0% | M | 2.4% | Q | 0.1% |
| N | 6.7% | W | 2.4% | X | 0.1% |
| S | 6.3% | F | 2.2% | J | 0.1% |
| H | 6.1% | G | 2.0% | Z | 0.1% |
| R | 6.0% | Y | 2.0% | | |

Відзначені властивості мов використовуються криптоаналітиками при розкритті шифрів, тому для оцінки властивостей шифрованих текстів необхідно мати відповідну інформацію про ймовірнісні характеристики відкритих повідомлень.

З вказаною метою використовується ряд математичних моделей відкритих текстів. Наведемо приклади найпоширеніших моделей, що характеризують їх важливі ймовірнісно-статистичні властивості.

Відкритий текст $T = \{t_1, t_2, \ldots\}$ є послідовністю поліноміальних випробувань на множині виходів $\{j\} = Z_m$ з вектором зустрічності окремих знаків $\bar{p} = (p_0, p_1, \ldots, p_{m-1})$, де $p_j = P\{t_i = j\}$

Це одна з найпоширеніших моделей відкритого тексту, яка дозволяє вирішувати задачі, пов'язані з класом так званих потокових шифрів.

Відзначимо, що відповідно до схеми незалежних випробувань використовується характеристика невизначеності або міра інформації, вибираної з експерименту.

Ентропією ймовірності схеми $\xi = \begin{pmatrix} 0 & \dots & m-1 \\ p_0 & \dots & p_{m-1} \end{pmatrix}$ називається величина $H(\xi) = -\sum_{i=0}^{m-1} p_i \log_2 p_i$.

Згідно з теорією кодування, будь-який вихід імовірнісної схеми може бути закодований символами 0 і 1 так, що одержана довжина кодового слова буде скільки завгодно близька зверху до $H(\xi)$. Отже, одиницею кількості інформації логічно вважати один біт.

Неважко показати, що функція ентропії опукла вгору і досягає свого максимуму $\log_2 m$ для рівноймовірного розподілу $(1/m, 1/m, \dots, 1/m)$.

З ентропією пов'язана ще одна характеристика – надмірність мови, яка обчислюється за формулою

$$R = 1 - \frac{H}{\log_2 m}.$$

Поняття надмірності мови виникло у зв'язку з тим, що максимальна інформація, яку міг би нести кожний символ повідомлення, в рівноімовірній схемі рівна $\log_2 m$.

Для реальних текстів буква несе менше інформації, тому величина $\log_2 m - H$ характеризує невикористані можливості в передачі тексту, а відношення

$$\frac{\log_2 m - H}{\log_2 m} = 1 - \frac{H}{\log_2 m},$$

умовно кажучи, показує, яку частку символів повідомлення можна пропустити без істотної втрати значення.

Зокрема, очевидно, що викреслювання з тексту, написаного російською мовою всіх голосних букв практично не знижує його

інформативності. Більш того, в деяких мовах до 70-80 % знаків тексту можна видалити без істотної втрати значення.

Як наслідок, для *m-буквеного* алфавіту кількість смислових текстів (із заданими обмеженнями зустрічання букв) значно менша числа усіх можливих наборів з $N$ букв, рівного $m^N$.

Згідно з теоремою Мак-Міллана число відкритих текстів для реальної мови з величиною ентропії на один знак тексту $H$ для достатньо великих оцінюється величиною $\#N = 2^{NH}$ .

Інша модель – відкритий текст $T = \{t_1, t_2, \ldots, t_\nu; t_{\nu+1}, t_{\nu+2}, \ldots, t_{2\nu}; \ldots\}$ є послідовністю незалежних випробувань на множині всіх послідовностей довжини $\nu$, елементи яких належать $Z_m$. Така множина називається множиною $\nu$-грам.

При цьому, принаймні, для деяких $\nu$-грам $\{t_{i1}, t_{i2}, \ldots, t_{i\nu}\}$ має місце нерівність:

$$P\{t_{i1}, t_{i2}, \ldots, t_{i\nu}\} \neq p_{i1} p_{i2} \ldots p_{i\nu}.$$

Дана модель зазвичай використовується при аналізі шифрів блокового типу.

Ще одна модель – відкритий текст $T = \{t_1, t_2, \ldots\}$ є послідовністю випробувань, зв'язаних в простий однорідний ланцюг Маркова з початковим вектором розподілу ймовірності $\bar{p} = (p_0, p_1, \ldots, p_{m-1})$ і матрицею перехідних ймовірностей $P = \|\hat{p}_{ij}\|$.

Дана модель дозволяє враховувати наявність залежностей між знаками відкритого тексту, а тому використовується для побудови критеріїв, за допомогою яких відбраковуються помилкові варіанти ключів.

Статистичні властивості реальних відкритих текстів є початковими даними для проведення криптоаналітичних атак і оцінки практичної стійкості шифрсистем.

### 1.3 Симетричні і асиметричні криптосистеми

При розгляді цього питання, перш за все, необхідно визначити об'єкт дослідження: а саме, криптографічну систему (далі – криптосистема).

Існує декілька підходів до визначення криптосистеми, при цьому кожне з визначень, як правило, застосоване в своїй області.

Криптограф віддасть перевагу визначенню, що тяжіє до математичних понять. Для фахівця з організаційних аспектів безпеки інформаційних систем найбільш зручний нормативно-правовий підхід.

Розробник засобів криптографічного захисту інформації (КЗІ) орієнтуватиметься на визначення, засноване на принципах побудови і функціонування системи.

З математичної точки зору, криптосистема є параметричним сімейством $(E_K)$ перетворень відкритого тексту.

Члени цього сімейства індексуються, або позначаються символом *k*; параметр $k \in K$ є ключем. Простір ключів – це набір можливих значень ключа.

Зазвичай ключ є послідовністю символів деякого алфавіту. Для даного визначення ми ототожнюватимемо поняття **криптосистема, шифрсистема і шифр**.

Інший похід до визначення криптосистеми викладений в нормативних документах із захисту інформації, згідно з яким криптосистема – це сукупність засобів криптографічного захисту інформації, необхідних ключів, нормативної, експлуатаційної, а також іншої документації (у тому числі такий, що визначає заходи безпеки), використання яких забезпечує належний рівень захищеності процедури інформаційного обміну. При цьому інформаційний обмін передбачає зберігання, обробку і передачу даних.

Виходячи з технічних аспектів реалізації криптосистеми Д. Чандлер дав наведене нижче визначення.

Криптосистема працює за певною методологією (процедурою), яка включає: один і більше криптографічних алгоритмів (математичних перетворень); ключі, що використовуються цими алгоритмами; системи управління ключами; вхідні і вихідні послідовності (відкритого і зашифрованого текстів).

У відзначену методологію входять також процедури генерації, розподілу, зберігання і знищення ключів.

Щоб не створювати додаткових складнощів для розуміння основних методів і принципів криптографічного захисту інформації приймемо деякі спрощення термінології.

Зокрема, замість офіційного дуже довгого і не менш широкого поняття «засіб криптографічного захисту інформації», ми використовуватимемо більш короткі, на практиці використовуються поняття «шифратор», «генератор ключів».

Залежно від принципів побудови алгоритмів криптосистеми підрозділяють на **симетричні** і **з відкритим ключем** [1, 2, 5].

Ключі зашифрування і розшифрування в **симетричних криптосистемах** збігаються або один з іншого може бути достатньо просто обчислений.

В **системах з відкритим ключем** використовуються два ключі – **відкритий** і **секретний**, причому знаходження секретного ключа за відомим відкритим є складною щодо обчислення математичною задачею, яка, як правило, не розв'язується за допомогою звичайної електронної обчислювальної техніки.

Такі системи використовуються, наприклад, для **захищеного обміну ключами по відкритих каналах зв'язку**, при цьому вихідна інформація зашифровується за допомогою відкритого ключа, доступного всім користувачам деякої ІТС, а розшифровується тільки тим абонентом мережі, який має відповідний секретний ключ.

Інший напрям вживання асиметричних систем – створення математичного апарату **електронного цифрового підпису** (ЕЦП).

ЕЦП в криптографії – це результат криптографічного перетворення за допомогою секретного ключа функції відкритого тексту.

ЕЦП, з'єднаний з текстом, посилається іншим користувачам ІТС, які за допомогою загальнодоступного відкритого ключа і криптографічного перетворення можуть перевірити авторство і автентичність повідомлення. Докладніше механізми ЕЦП розглядаються нижче.

### 1.4 Практичні вимоги до симетричних криптосистем

Один з основоположних принципів побудови криптографічних систем, або як їх часто називають на практиці – шифрів – був сформульований в XVII столітті голландським офіцером В. Керкхофсом. Він свідчить, що стійкість шифру не повинна залежати від знання зловмисником особливостей побудови шифру і повинна визначатися

тільки секретністю ключа, знання алгоритму шифрування не повинно впливати на надійність захисту.

Згодом в одній зі своїх фундаментальних робіт американський математик К. Шеннон строго обґрунтував ще ряд важливих принципів побудови симетричних шифрів.

Найважливішою властивістю будь-якої криптосистеми є криптостійкість, що характеризує її здатність протистояти криптоаналітичним атакам, що дозволяє уникнути в деяких допустимих межах можливості дешифрування зашифрованих повідомлень без знання ключа або підробки повідомлення. Існує декілька підходів до оцінки криптостійкості, залежно від комплексу умов, що полегшують криптоаналіз. Відзначимо тільки деякі:

1) середній час, необхідний для пошуку істинного ключа шляхом повного перебору всіх можливих варіантів ключів;

2) складність найкращого алгоритму розв’язання задачі розкриття ключа за наявності шифрованого і відповідного йому відкритого тексту;

3) складність найкращого алгоритму розв’язання задачі розкриття ключа за допомогою спеціально підібраних пар шифрованих і відповідних їм відкритих текстів і ін.

Комплекс умов, відповідний останньому випадку, в криптографії часто використовується для розмежування стійких і слабких криптосистем в тому значенні, що криптосистема, яка допускає розкриття ключа при даному комплексі умов (не кажучи вже про менш небезпечні ситуації), вважається слабкою, інакше – стійкою. Зрозуміло, що при цьому розкриття ключа повинно здійснюватися в прийнятний час.

Ряд вимог до шифру обумовлений необхідністю забезпечення простоти його технічної реалізації в програмному і апаратному вигляді.

Відзначимо, що при апаратній реалізації шифрів досягаються висока швидкість обробки інформації, надійний захист від несанкціонованого доступу до конфіденційних даних, проте для неї характерна висока вартість. Програмна реалізація більш дешева, характеризується відомою гнучкістю використання, в той же час вона менш захищена від різних атак хакерів.

З урахуванням того факту, що більшість реальних каналів зв'язку в тій чи іншій мірі піддається дії шумів, висувається вимога, що спотворення шифрованого повідомлення шляхом заміни будь-якого символу не

повинно приводити до істотного розповсюдження спотворень при розшифруванні.

Якщо в результаті спотворення одного символу в шифрованому повідомленні може бути спотворений тільки один символ в розшифрованому тексті, то відповідний шифр називається таким, що не поширює спотворення.

Необхідність забезпечення високої пропускної спроможності системи зв'язку і вимога економного використання ресурсу пам'яті в обчислювальних системах висувають таку умову: шифрування не повинно істотно збільшувати довжину вихідного тексту, а додаткові біти, що вводяться в повідомлення в процесі шифрування, повинні бути повністю і надійно приховані в шифрованому тексті.

До сучасних криптографічних систем захисту інформації висуваються й інші вимоги:

1) зашифроване повідомлення повинно піддаватися прочитанню тільки за наявності ключа;

2) будь-який ключ з допустимої множини повинен забезпечувати надійний захист інформації;

3) незначна зміна ключа повинна приводити до істотної зміни виду зашифрованого повідомлення;

4) число операцій, необхідних для розшифрування інформації шляхом перебору всіляких ключів, повинно перевищувати прогнозовані на перспективу обчислювальні можливості комп'ютерів з урахуванням методів використовування мережевих обчислень;

5) не повинно бути простої і легко встановлюваної залежності між ключами, що використовуються в процесі експлуатації криптосистеми;

6) число операцій, необхідних для визначення ключа з використанням шифрованого повідомлення і відповідного йому відкритого тексту, повинно бути не менше загального числа можливих ключів;

7) структурні елементи алгоритму шифрування повинні бути незмінними.

**Питання до розділу 1**

1) Який апаратний пристрій можна запропонувати для організації шифрування даних за допомогою комп'ютера і відкритих ліній зв'язку?

2) Що являє собою предмет криптології?

3) Що являє собою предмет криптоаналізу?

4) Для яких основних задач захисту інформації застосовуються криптографічні перетворення?

5) Що таке ключ криптографічного перетворення?

6) Що таке зашифрування і розшифрування?

7) Яка буква найбільш часто зустрічається в англійській мові?

8) Чим є криптографічна система з математичної точки зору?

9) Як визначається криптосистема в нормативних документах з захисту інформації?

10) Чим відрізняється вживання ключів в симетричних і асиметричних криптосистемах?

11) Яка властивість криптосистем характеризується поняттям «криптостійкість»?

12) Які основні вимоги повинні задовольняти ключі, щоб не знизити стійкість криптосистеми?

# РОЗДІЛ 2 ЕЛЕМЕНТАРНІ ШИФРИ І ЇХ ВЛАСТИВОСТІ

## 2.1 Класифікація шифрсистем

Всю різноманітність існуючих симетричних криптосистем, що використовуються для шифрування вихідних повідомлень, можна звести до класів перетворень, наведених на рис. 2.1 [1, 2, 4, 7].

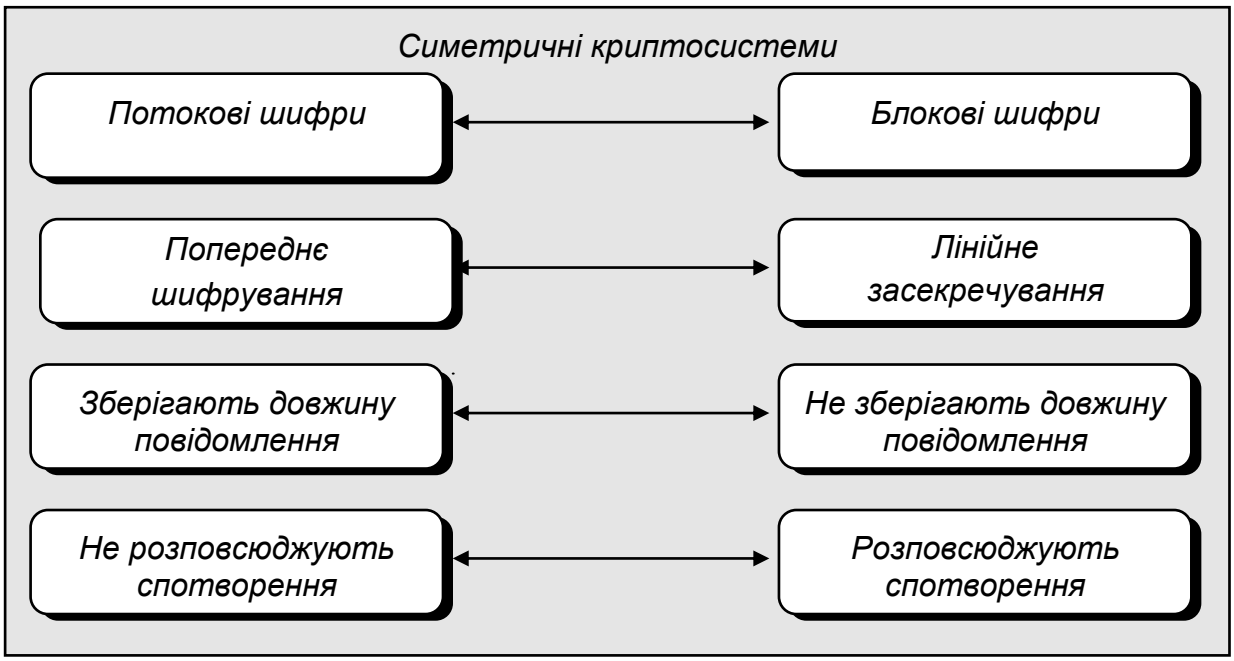



Рисунок 2.1 – Класифікація симетричних криптосистем

Розрізняють криптосистеми потокового і блокового типу [18].

Потоковим шифром називається система, що в ній на кожному такті використовується змінний алгоритм шифрування, який визначається вихідними ключовими даними і номерами тактів шифрування, аж до того, що розглядається.

Блоковим шифром називається система шифрування, яка на кожному такті використовує постійний, вибраний до початку шифрування, залежно від ключів, алгоритм.

Більшість сучасних стандартів шифрування, включаючи алгоритми ГОСТ 28147-89, DES, AES та інші, передбачає обробку блоку даних довжиною від 64 бітів.

Системи шифрування розрізняють за видами, залежно від синхронізації процедур шифрування – розшифрування і організації передачі зашифрованих даних в каналі зв'язку.

1. **Попереднє шифрування**, при якому момент передачі інформації після її криптографічного перетворення обирається передавачем повідомлення. Цей вид шифрсистеми використовується у випадку електронної пошти, коли повідомлення попередньо шифрується, а потім передається в деякій ІТС, наприклад, в мережі Інтернет;

2. **Лінійне засекречування** належить до класу синхронних потокових шифрсистем і передбачає узгоджену роботу шифрувальних пристроїв і зв'язувального обладнання. Зокрема, на цьому принципі побудована апаратура шифрування телефонних розмов.

Схема взаємодії основних елементів системи шифрування, наведена на рис. 2.2, містить такі елементи:

– генератори потокових шифрів: А, Б і відповідні вузли, що реалізують криптографічні перетворення;

– комплекс засобів зв'язку, що забезпечують роботу каналу зв'язку.

Для організації спільної роботи двох генераторів потокових шифрів вони ініціалізуються за допомогою ключової інформації і даних синхронізації (синхропосилання).

Синхронна (одночасна) робота генераторів досягається за допомогою системи синхронізації, спеціального радіоелектронного приладу, принципи побудови якого розглядаються в теорії електро- і радіозв'язку.

У випадку збою на лінії зв'язку система синхронізації забезпечує перезапуск генераторів потокового шифрування і підтримує їх узгоджену за часом роботу протягом сеансу зв'язку.

Перевагою систем лінійного засекречування перед системами попереднього шифрування є більш повне використання пропускної здатності каналу, в той же час, в системах попереднього шифрування повідомлення довгий час можуть зберігатися у зашифрованому вигляді, до тих пір, поки не виникне потреба доступу до них.

Різноманітність шифрсистем суттєво звужується, якщо вимагати виконання властивостей шифрів **зберігати довжину повідомлення і не розповсюджувати спотворень.** Як буде вказано далі, це дозволяє

повністю описати відповідний клас шифрів як комбінацію двох елементарних шифрів – перестановки і підстановки.

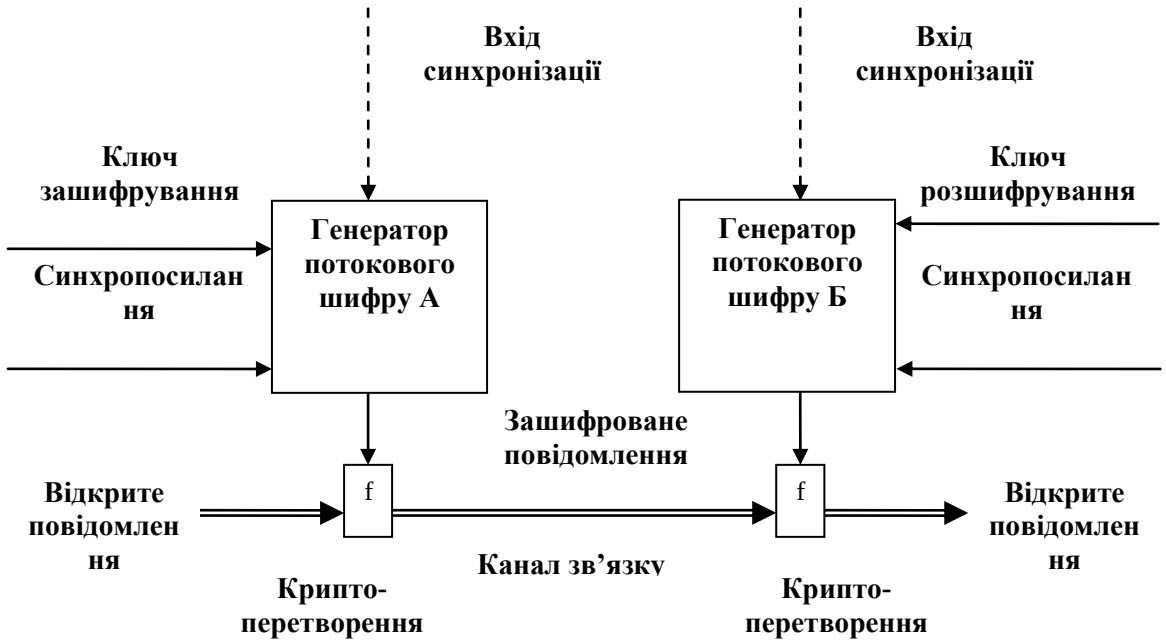



Рисунок 2.2 - Схема лінійного засекречування

Слід також відзначити, що в багатьох блокових шифрах сумарне криптографічне перетворення є послідовністю (з можливим повторенням і чередуванням) елементарних шифрів, застосованих до блоку тексту, що шифрується, або до подальших його проміжних станів. З врахуванням наведеного, вивчення елементарних шифрів викликає самостійний інтерес.

## 2.2 Властивості елементарних шифрів

**Перестановкою** $S$ множини цілих чисел $\{1,\ldots,n\}$ називається результат (образ) взаємно однозначного відображення даної множини на себе.

Застосування перестановки як системи шифрування зводиться до переупорядкування послідовності символів відкритого тексту.

Для того, щоб показати, що символ переміщений з позиції $\sigma(i)$ в позицію $i$ де $1 \le i \le n$, використовується запис $\Sigma = (\sigma(1),\ldots,\sigma(n))$.

Нагадаємо, що число різних перестановок цілих чисел $\{1,\ldots,n\}$ дорівнює $n!=1\cdot 2\ldots(n-1)n$.

На практиці дану величину для достатньо великих $n$ зручно оцінювати за допомогою формули Стірлінга:

$$n!=\sqrt{2\pi n}\cdot n^n e^{-n}\left(1+o(1)\right).$$

Наприклад, для $n=27$ загальне число ключів шифру перестановки досягає величини $10^{28}$, що при сучасному розвитку обчислювальної техніки є досить великою величиною.

Нехай маємо відкрите повідомлення $T=\{t_1,t_2,\ldots,t_n\}$ з $n$ элементов, тоді в результаті застосування відображення $\Sigma: T\to S$ отримаємо шифрований текст $t_i\to s_i=t_{\sigma(i)}$, $1\le i\le n$.

Наприклад, відкрите повідомлення КИЇВ після застосування перестановки $\Sigma=(3421)$ перетвориться в шифрований текст ЇВИК. Відновлення відкритого тексту (розшифрування) з шифрованих повідомлень виконується аналогічним чином з використанням оберненого перетворення, у нашому випадку $\Sigma^{-1}=(4312)$: {ЇВИК}$\to${КИЇВ}.

У випадку тексту $T$ довільної довжини $N$ шифр перестановки визначається як послідовне застосування перестановки $\Sigma$ до блоків (фрагментів) $T(k)$, $k=1,\ldots,N$ відкритого тексту, довжина кожного з яких дорівнює $n$.

Якщо довжина тексту $T=\{T(k)\}_{k=1}^{N}$ не кратна довжині перестановки, то останній блок доповнюється деяким набором символів алфавіту для отримання довжини, кратної $n$.

Відмітимо, що для кожного з блоків відкритого тексту може застосовуватись своя перестановка $\Sigma_k$, що збільшує загальну кількість варіантів ключів до величини $N(n!)$, де $N$ дорівнює числу блоків, які шифруємо.

Даний метод шифрування застосовується в апаратурі засекречування мовних повідомлень, так званого, мозаїчного типу (аналогія мозаїки і перестановки очевидна).

На прикладі структурної схеми, наведеної на рис. 2.2, можна говорити, що генератори потокових шифрів А і Б апаратури

засекречування мозаїчного типу створюють послідовність перестановок $\Sigma_k$ і $\Sigma_k^{-1}$, за допомогою яких здійснюється перетворення відкритого і зашифрованого повідомлення відповідно.

**Шифр перестановки**, за визначенням, руйнує стійкі сполучення у відкритому тексті. В той час, кількість однакових букв в шифрованому тексті точно відповідає їх кількості у відкритому тексті, що в термінах першої моделі відкритого тексту виражається як рівність відповідних ймовірностей зустрічі різних знаків:

$$\forall i,\ j \text{ має місце } p_j = P\{t_i = j\} = P\{s_i = j\} = q_j .$$

Ця властивість є очевидною слабкістю шифру, оскільки, використовуючи статистичні критерії (див. нижче «Методи генерації ключових даних»), при достатній довжині шифрованого повідомлення в інформаційному потоці можна легко виявити відповідні шифровки.

Потім за кожним шифрованим текстом також статистичним шляхом можна встановити довжину перестановки на основі властивості сполученості букв відкритого повідомлення або так званим методом перевірки стандартної фрази.

Таким чином, при достатньо великій кількості допустимих ключів шифр перестановки без застосування додаткових заходів не може вважатися достатньо стійким.

При аналізі ускладнень шифру перестановки використовуються властивості підстановок.

**Підстановкою** $X$ степеня $m$ на множині $A$ з $m$ елементів називається взаємно однозначне відображення множини $A$ на себе.

Підстановку записують у вигляді дворядкової таблиці:

$$X = \begin{pmatrix} 0 & 1 & \dots & m-1 \\ x_0 & x_1 & \dots & x_{m-1} \end{pmatrix} = \begin{pmatrix} i \\ x_i \end{pmatrix}.$$

Підстановки можна множити за правилом: $\begin{pmatrix} i \\ x_i \end{pmatrix}\begin{pmatrix} x_i \\ y_{x_i} \end{pmatrix} = \begin{pmatrix} i \\ y_{x_i} \end{pmatrix}$.

Множина всіх підстановок степеня $m$ є групою за множенням.

**Шифром простої заміни** або багатоалфавітною підстановкою називається криптографічне перетворення відкритого тексту в шифрований $X: T \to S$, при якому символу вихідного тексту $t_i$ відповідає символ шифрованого тексту $s_i = X(t_i)$.

Множину всіх підстановок над $Z_m$ надалі будемо позначати як $S(Z_m)$.

Нагадаємо, що група $S(Z_m)$ відносно операції добутку має такі властивості.

1. Замкнутість: добуток двох підстановок $X_1$ і $X_2$ є підстановкою: $X_1 X_2 \in S(Z_m)$. Дана властивість в термінах шифру заміни означає, що двократне застосування простої заміни еквівалентно деякій третій заміні. Тобто, стійкість шифрування від цього не збільшується.

2. Асоціативність: результат добутку $X_1 X_2 X_3$ не залежить від порядку розставляння дужок.

3. Існування одиниці: підстановка, визначена як $E = \begin{pmatrix} i \\ i \end{pmatrix}$, є одиницею $S(Z_m)$ відносно операції множення: $\forall x \in S(Z_m)\ EX = XE = X$.

4. Існування оберненого елемента: для будь-якої підстановки $X$ існує єдина обернена підстановка $X^{-1}$, яка задовольняє умову $XX^{-1} = X^{-1}X = E$.

Взаємно оберненими підстановками є:

$$X = \begin{pmatrix} 0 & 1 & \dots & m-1 \\ x_0 & x_1 & \dots & x_{m-1} \end{pmatrix} \text{ і } X^{-1} = \begin{pmatrix} x_0 & x_1 & \dots & x_{m-1} \\ 1 & 2 & \dots & m-1 \end{pmatrix}.$$

Фактично, останні вирази і є ключами зашифрування і розшифрування. Кількість різних ключів дорівнює порядку симетричної групи підстановок $S(Z_m)$ і дорівнює $m!$.

Криптографічне перетворення $\tilde{X}$, яке визначається ключем $\tilde{k} = (X_1, X_2, \dots X_n)$, де $X_i \in S(Z_m)$ і існує хоча б одна пара $i \neq j$, для якої $X_i \neq X_j$, називається **шифром колоною заміни або шифром багатоалфавітної підстановки** періоду $n$, якщо система рівнянь

шифрування послідовних $n$-грам вихідного тексту $(t_1, t_2, \ldots, t_n) \to (s_1, s_2, \ldots, s_n)$ має вигляд: $s_i = X(t_i)$, де $i = 1, 2, \ldots, n$.

Найбільш важливими властивостями для оцінки стійкості шифру простої заміни є наведені нижче.

1. Властивість збереження структури: оскільки вихідний текст шифрується посимвольно, то для шифрування $n$-грами $(t_1, t_2, \ldots, a, \ldots, a, \ldots t_n)$, у якої символи на місцях $i, j$ рівні, тобто збігаються, буде отримана $n$-грама $(s_1, s_2, \ldots, b, \ldots, b, \ldots s_n)$, у якої символи на місцях $i, j$ також рівні.

2. Нехай $\hat{X}$ – підстановкова матриця, відповідна підстановці $X$:

$\hat{X} = \|a_{i,j}\|$, где $a_{i,j} = 1$, якщо $j = X(i)$, інакше, $a_{i,j} = 0$.

Неважко переконатися, що розподіли ймовірностей знаків відкритого $\bar{p} = (p_1, p_2, \ldots, p_n)$ і шифрованого $\bar{q} = (q_1, q_2, \ldots, q_n)$ текстів пов'язані співвідношенням $\bar{q} = \hat{X}\bar{p}$ .

Останній вираз можна узагальнити на випадок $n$ різних підстановок $(X_1, X_2, \ldots X_n)$, рівноймовірно і незалежно, які обирають для зашифрування відкритого тексту: $\bar{q} = n^{-1}\left(\sum_i \hat{X}_i\right)\bar{p}$.

З останнього виразу можна зробити висновок, що при зроблених припущеннях і збільшенні числа різних підстановок, використовуваних для шифрування, імовірнісні характеристики шифрованого тексту наближаються до рівномірного розподілу. Інтуїтивно ясно, швидкість наближення залежатиме від структури підстановок.

Відзначимо один важливий окремий випадок багатоалфавітної підстановки.

Підмножина підстановок зсуву

$$C_m = \left\{ C^{(k)} \in S(Z_m) :\ C^{(k)} = \begin{pmatrix} i \\ i + k \bmod m \end{pmatrix} \right\},\ k = 0, 1, \ldots, m-1$$

називається **сімейством шифрів Цезаря**.

*Вправа*. Довести, що $C_m$ - комутативна підгрупа симетричної групи $S(Z_m)$.

*Вказівка*. Оберненою підстановкою до $C^{(k)}$ є $C^{(m-k)}$.

*Вправа*. Нехай дана підстановка Цезаря $C^{(2)}$ з верхнім рядком (алфавітом) вигляду

V=(АБВГДЕЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯ_).

Зашифрувати повідомлення (КОНТРАКТ_НА_ПОСТАВКУ) за допомогою підстановки $C^{(3)}$.

Відзначимо, що використання однієї підстановки Цезаря не забезпечує навіть мінімальної стійкості шифрування, тому що при наявності досить короткого шифрованого тексту перебором $m$ варіантів підстановки можна повністю відновити справжній ключ. У нашому прикладі маємо текст довжиною 20 символів, з них різних всього 11, на основі яких практично повністю можна відновити ключ $C^{(3)}$.

Більш складними для дешифрування криптосистемами є шифри заміни, що базуються на підстановці груп символів, наприклад, біграм (шифр Плейфера) або $m$-грам (шифр Хілла). У той же час вони складніші для реалізації і вимагають досить великого обсягу ключової інформації.

У плані підвищення стійкості шифрування більш ефективним є застосування шифру колоною заміни, який визначається базовим набором різних підстановок $\tilde{k} = (X_1, X_2, \dots X_n)$ і послідовністю індексів $\Gamma = \{\gamma_1, \gamma_2, \cdots\}$, що встановлюють порядок використання підстановок із заданого набору. У цих умовах шифр колоною заміни для повідомлення $T = t_1, t_2, \dots$ визначається таким чином:

$$E(T, (X_1, X_2, \dots X_n), \Gamma) = X_{\gamma_1}(t_1), X_{\gamma_2}(t_2), \dots$$

Послідовність індексів $\Gamma = \{\gamma_1, \gamma_2, \cdots\}$ називають **гамою шифрування** або ключовим потоком.

Окремим випадком шифру колоною заміни є **шифр гамування**, коли за базовий набір підстановок використовується сімейство підстановок Цезаря, а вихідний текст вигляду $T = t_1, t_2, \dots$ перетвориться в зашифрований текст $S = s_1, s_2, \dots$ за правилом:

$$s_i = C^{(\gamma_i)}(t_i) = (t_i + \gamma_i) \bmod m.$$

У залежності від принципу генерації гами шифру розрізняють періодичні і випадкові (неповторювані) гами.

Гама шифрування називається **періодичною,** якщо існує деяке $p > 0$ таке, що $\forall i \ \gamma_i = \gamma_{i+p}$ .

У цьому випадку періодичним називається і сам шифр.

Потужність множини ключів періодичного шифру гамування дорівнює $m^p$. При псевдовипадкових послідовностях $\Gamma = \{\gamma_1, \gamma_2, \cdots\}$ з великим періодом багатоалфавітні системи є дуже стійкими.

Досить поширеним способом отримання таких гамових послідовностей є застосування датчиків псевдовипадкових чисел.

Надалі ці методи будуть розглянуті досить докладно.

*Вправа.* Показати, що для періодичного шифру колоною заміни потужність множини ключів з $n$ підстановок степеня $m$ дорівнює $\binom{m!}{n} n!$.

Якщо довжина гами шифрування дорівнює довжині відкритого тексту, а сама гама - рівноймовірна і використовується тільки один раз, то має місце випадок системи одноразового використання або шифру Вернама, названого на честь інженера американської компанії AT&T, який запропонував у 1917 році відповідний метод шифрування.

У той час ключ записувався на паперовій стрічці. Кожна буква вихідного тексту в латинському алфавіті, розширеному деякими додатковими знаками, спочатку переводилася з використанням коду Бодо в п’ятибітовий символ. До початкового тексту в коді Бодо ключ додавався за модулем 2. Телетайп фірми AT&T зі зчитувальним пристроєм Вернама та обладнанням для шифрування використовувався в корпусі зв'язку армії США.

Оскільки спочатку як носій ключової інформації використовувалася звичайна перфострічка, а в подальшому - спеціальним чином виготовлені блокноти, то нерідко для позначення такої системи використовують також термін "одноразова стрічка" і "одноразовий блокнот".

Приклад шифру гамування з ключем (ОДНОРАЗОВАЯ_ГАММА_).

Зашифруємо текст (ВЫПОЛНИ_ТРАНЗАКЦИЮ) за допомогою цього ключа.

Шифрування зручніше виконувати шляхом додавання за модулем 33 цілих чисел - номерів літер в алфавіті, починаючи з одиниці.

Після перекодування, додавання і повторного перекодування отримаємо таке шифроване повідомлення:

| | |
|---|---|
| **ВЫПОЛНИ_ТРАНЗАКЦИЮ** | **03 28 16 15 12 14 09 33 19 17 01 14 08 01 …** |
| **ОДНОРАЗОВАЯ_ГАММА_** | **15 05 14 15 17 01 08 15 03 01 32 33 04 01 …** |
| **С_ЭЭЬОРОХС_НЛБЧВЙЮ** | **18 33 30 30 29 15 17 15 22 18 33 14 12 02 …** |

Шифр гамування забезпечує вирівнювання ймовірностей зустрічі знаків шифрованого тексту. Очевидно, якщо маємо розподіл ймовірностей зустрічності знаків гамии $(g_1,...,g_m)$, то розподіл імовіростей символів шифрованого тексту описується такою системою рівнянь:

$$
\begin{aligned}
q_1 &= p_1 g_m + p_2 g_{m-1} + ... + p_m g_1, \\
q_2 &= p_1 g_1 + p_2 g_m + ... + p_m g_2, \\
&... \\
q_m &= p_1 g_{m-1} + p_2 g_{m-2} + ... + p_m g_{m1}.
\end{aligned}
$$

*Вправа.* Показати, що у випадку незалежних, рівномірно розподілених знаків гами знаки шифрованого тексту будуть розподілені також рівномірно.

Шифр системи на основі гами одноразового використання, як буде показано далі, є досконалими щодо стійкості. Однак їх застосування для забезпечення конфіденційності передаваної інформації у багатьох випадках практично неможливе через проблеми з генерацією та розподілом великих обсягів ключової інформації.

Крім шифрів модульного гамування існують шифри, що використовують інші операції для відображення пари $(t,\gamma)$ відкритий текст-гама в знак шифрованого тексту: $c = f_\nu(t,\gamma)$.

Функція $f_\nu$ являє собою обернене табличне перетворення, що використовується на такті шифрування $\nu$, такі шифри називаються шифрами табличного гамування.

Важливою характеристикою шифрів колоною заміни є стійкість у разі повторного використання гами шифрування при невідомому базовому наборі підстановок.

У той же час потрібно відзначити, що і багатоалфавітні підстановки, в принципі, доступні криптоаналітичному дослідженню. Крипостійкість багатоалфавітних систем різко падає зі зменшенням довжини ключа.

### 2.3 Теорема Маркова

Короткий огляд основних типів шифрсистем не може дати вичерпного уявлення про все різноманіття симетричних систем, однак для фахівця в галузі менеджменту систем безпеки ІТС цього введення, на нашу думку, цілком достатньо. Тим більше, що наступна теорема дозволяє представити, в цілому, безліч практично придатних шифрсистем.

**Теорема** (Марков А. А.). Будь-який шифр, який зберігає довжини повідомлень і не розповсюджує спотворень типу заміни символу, подамо у вигляді композиції шифрів перестановки і колоною заміни.

Таким чином, найбільший інтерес з точки зору генерації ключів матимуть послідовності незалежних випадкових величин (гами шифру) і якісні підстановки.

У випадку каналу зв'язку з завадами, в результаті яких може бути пропущений один символ, практичне значення має нижчевикладена теорема.

**Теорема.** Будь-який шифр, що не поширює спотворень типу зникнення символів у шифрованому тексті, є або шифром простої заміни, або добутком шифру простої заміни та шифру перестановки, що полягає в інверсному (зворотному) порядку запису тексту.

Слід зазначити, що криптографічні методи не призначені для боротьби зі спотвореннями в каналі зв'язку.

Для вирішення відповідних проблем слід застосовувати інші методи, наприклад, завадостійке кодування.

### Питання до розділу 2

1. Що таке криптосистема потокового типу?
2. Що таке криптосистема блокового типу?
3. Що таке попереднє шифрування?
4. Що таке лінійне шифрування (лінійне засекречування)?

5. Чому необхідна синхронізація в потокових шифрах?
6. Який математичний об'єкт називається підстановкою?
7. Що таке шифр простої заміни?
8. Як працює шифр перестановки?
9. Як влаштований шифр колоною заміни?
10. Як влаштований шифр Цезаря?
11. Як відрізнити за шифртекстом шифр простої заміни і шифр перестановки за допомогою частотних характеристик?
12. Що таке період гами шифрування?
13. Які умови задовольняє гама в абсолютно стійкому шифрі Вернама?
14. Як формулюється теорема А. А. Маркова?
15. Чи є криптографічні системи засобом боротьби зі спотвореннями в каналі зв'язку?

# РОЗДІЛ 3 МОДЕЛІ ЗАГРОЗ БЕЗПЕКИ КРИПТОСИСТЕМ

## 3.1 Формальна модель загроз

Говорячи про загрози безпеки криптосистем, зауважимо, що довести стійкість криптосистем неможливо без формалізації поняття стійкості, для чого потрібно, по-перше, сформулювати припущення про можливості потенційного зловмисника, а по-друге, уточнити завдання, що стоїть перед ним. Різним можливостям зловмисника відповідають різні види атак, а завданням, що вирішуються в ході їх проведення, - загрози безпеки криптосистеми [2, 3, 7, 8].

Під зловмисником будемо розуміти аналітика, який має певні знання в області криптографії, оснащений деякими технічними та програмними засобами. Мета зловмисника - пошук слабкостей в криптосистемі для створення технології подолання її «захисних бар'єрів».

Для криптосистем розрізняють чотири рівні можливостей атакуючої сторони.

1) Нульовий: випадковий зловмисник системи безпеки, що володіє звичайною обчислювальною технікою, обладнанням, програмним забезпеченням і «озброєний» загальними відомостями з питань захисту та безпеки телекомунікацій;

2) Перший: зловмисник системи безпеки добре знає особливості побудови і функціонування комплексів захисту інформації, використовує спеціальні (хакерські) програмні засоби та найбільш потужну широкодоступну обчислювальну техніку;

3) Другий: зловмисник корпоративного типу додатково до можливостей попереднього рівня має в своєму розпорядженні фінансову підтримку потужної бізнес-структури;

4) Третій: зловмисник системи безпеки має у своєму розпорядженні наукові, технічні та фінансові можливості спеціальної служби однієї з провідних країн світу.

Стійкість криптосистеми визначається відносно можливих загроз і їх реалізацій, тобто конкретних атак на систему.

Загрози криптосистеми характеризують загальні можливості потенційного зловмисника при порушенні конфіденційності та

імітозахищеності (цілісності, достовірності повідомлень) захищеного інформаційного обміну, а не конкретні методи (алгоритми) їх здійснення.

Одним з найважливіших напрямків досліджень в теоретичній криптографії є створення систем, стійких до всіх можливих загроз (навіть таких, імовірність настання яких відносно мала).

При цьому передбачається, що зловмисник в змозі провести найбільш ефективну з усіх можливих атак.

Необхідно підкреслити, що ніяка криптосистема не може бути абсолютно стійкою в повному розумінні цього виразу, а тільки стійкою в рамках конкретної моделі загроз.

Прикладом цього може служити шифр Вернама, абсолютно стійкий проти пасивного підслуховування, але він не усуває можливість нав'язування осмисленої неправдивої інформації у випадку активного зловмисника, який знає ділянки відкритого тексту.

Іншим відомим прикладом є криптосистема простої заміни, стійка у випадку коротких шифрованих повідомлень, але нестійка проти атаки з використанням фрагментів відкритих текстів.

На практиці криптосистеми використовуються в складі обчислювальних мереж або систем спеціального зв'язку (рис. 3.1).

До складу гіпотетичної системи зв'язку входять шифратори А і Б, Центр генерації та управління ключами, мережа або лінія зв'язку, абонентські установки типу телефон, факс або автоматизоване робоче місце. Доставка ключової інформації здійснюється за допомогою спеціальної поштової служби або по захищених каналах зв'язку.

Аналіз умов функціонування моделі системи спеціального зв'язку дозволяє виявити основні загрози її безпеці, що мають особливе значення для вибору практичних варіантів побудови системи, визначення переліку організаційних і технічних заходів захисту інформації, оцінювання достатності заходів протидії виявленим загрозам.

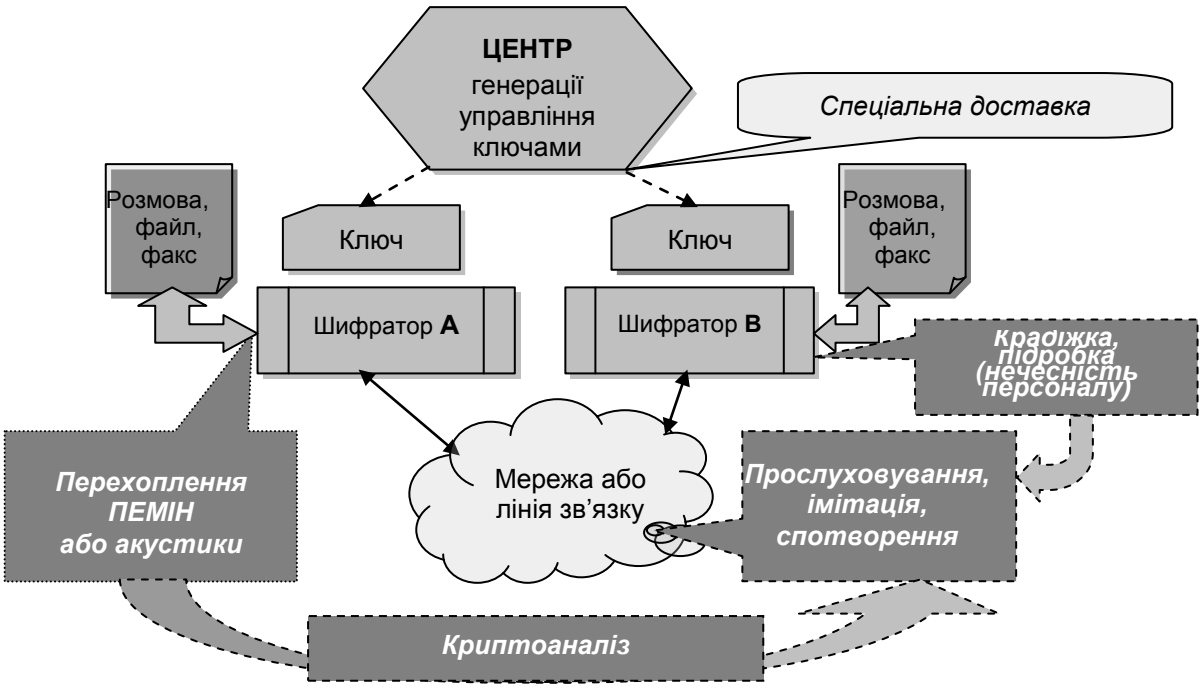



Рисунок 3.1 - Криптосистема і загрози її безпеки

До них слід віднести можливість реалізації ризиків:

- перехоплення зашифрованих повідомлень і проведення криптоаналізу з метою відновлення їх змісту або розкриття ключів;

- перехоплення критичної інформації про роботу шифраторів за рахунок технічних каналів витоку (побічні електромагнітні випромінювання, наведення, акустика) з метою спрощення задач дешифрування;

- крадіжка ключової та іншої критичної інформації внаслідок недобросовісних дій з боку обслуговуючого персоналу чи користувачів;

- підслуховування, перехоплення та маніпуляції (підробка) з даними в каналі зв'язку;

- порушення доступності та цілісності інформації в результаті зловмисного електромагнітного впливу (електромагнітний тероризм), порушень електроживлення та інших технічних факторів.

Особливо слід підкреслити, що важкість можливих наслідків для безпеки системи посилюється у випадку одночасної реалізації декількох загроз.

### 3.2 Атаки на симетричні і асиметричні шифрсистеми

У наукових виданнях досить повно описані можливі види атак на системи шифрування і ЕЦП, а види загроз - більшою мірою для систем ЕЦП і, меншою мірою, для систем шифрування.

Для усвідомлення проблеми розглянемо атаки на симетричні криптосистеми.

**При атаці на основі відомих шифртекстів** (ciphertext-only attack) передбачається, що зловмисникові відомий алгоритм шифрування і в його розпорядженні є деяка множина перехоплених повідомлень (криптограм), але невідомий секретний ключ.

За вихідними умовам дана атака є найслабшою з усіх можливих. Багато сучасних криптосистем успішно протистоять подібним атакам.

У разі атаки **на основі відомих відкритих текстів** (known-plaintext attack) додатково до умов попередньої атаки передбачається наявність у зловмисника множини пар криптограм і відповідних їм відкритих текстів.

Ефективність даної атаки така, що деякі потокові шифри, в тому числі побудовані на регістрах зсуву з лінійними зворотними зв'язками, можуть виявитися нестійкими.

Дослідження **простої атаки з вибором відкритих текстів** (chosen-plaintext attack). Тут зловмисник має можливість вибрати необхідну кількість відкритих текстів і отримати відповідні їм шифртексти. Цю атаку часто називають «опівнічною» або «обідньою» (midnight attack або coffee-break attack), що відповідає ситуації, коли оператор безконтрольно залишив засіб КЗІ в робочому стані і ним скористався зловмисник.

Хоча секретний ключ йому недоступний, зловмисник може зашифрувати підготовлені ним відкриті тексти, що дає додаткову інформацію для криптоаналізу системи.

**Адаптивна атака з вибором відкритого тексту**. В умовах попередньої атаки зловмисник має можливість вибору чергового відкритого тексту на основі знання криптограм, що відповідають попереднім відкритим текстам.

**Проста атака з вибором шифртексту** (chosen-ciphertext attack). Зловмисник має можливість вибрати необхідну кількість криптограм та отримати відповідні їм відкриті тексти. При цьому всі криптограми

повинні бути вибрані заздалегідь, тобто, до отримання першого відкритого тексту.

**Адаптивна атака з вибором шифртексту**. В умовах попередньої атаки зловмисник, вибираючи чергову криптограму, вже знає відкриті тексти, які відповідають усім попереднім.

**Атака з вибором тексту** (chosen-text attack). Зловмисник має можливість атакувати криптосистему з двох сторін, тобто, вибирати як криптограми (і розшифровувати їх), так і відкриті тексти (і зашифровувати їх). Атака з вибором тексту може бути простою, адаптивною, а також простою з одної «сторони» і адаптивною з іншої.

Атаки перераховані в порядку зростання їх сили, тобто, атака з вибором тексту є найсильнішою з перелічених.

Для асиметричних криптосистем (з відкритим ключем) класифікація атак аналогічна.

Слід мати на увазі, що в цьому випадку зловмисник завжди знає криптосистему і відкритий ключ, а адаптивна атака з вибором відкритого тексту є найслабшою з можливих атак на криптосистеми з відкритим ключем - зловмисник завжди має можливість провести таку атаку.

Крім того, існують атаки, специфічні для криптосистем з відкритим ключем. Наприклад, якщо число можливих відкритих текстів невелике, то зловмисник, знаючи відкритий ключ, може заздалегідь підготувати достатню кількість криптограм і потім, порівнюючи ці "заготовки" з перехопленими криптограмами, з високою ймовірністю отримувати відповідні відкриті тексти. Така атака називається атакою з перевіркою тексту (verifiable-text attack).

Типи загроз не мають настільки чіткої класифікації як типи атак. У літературі часто спостерігається така ситуація: дається досить точне визначення типу атаки, щодо якої розглядається стійкість криптосистеми, але нічого не говориться про те, що розуміється під розкриттям криптосистеми, тобто, в чому полягає задача зловмисника. Виділимо такі типи загроз (перераховані в порядку підсилення).

**Часткове розкриття**. Зловмисник в результаті атаки отримує часткову інформацію про секретний ключ або про відкритий текст. Хоча така загроза досить часто обговорюється в літературі, в загальному випадку далі словесних формулювань справа не йде. Причина, мабуть, в тому, що саме поняття часткової інформації досить розпливчасте і може

бути уточнено безліччю різних способів. Загроза часткового розкриття формалізована лише для абсолютно стійких в шенноновському сенсі криптосистем та криптосистем імовірнісного шифрування.

**Розкриття тексту.** У результаті проведеної атаки зловмисник повністю відновлює відкритий текст, що відповідає перехопленій криптограмі. Зазвичай передбачається, що відкритий текст вибирається навмання з деякої множини відкритих текстів, а відновлення відкритого тексту за криптограмою становить загрозу для безпеки, якщо ймовірність такого відновлення не є в певному сенсі "нехтовно малою".

**Повне розкриття**. У результаті проведеної атаки зловмисник обчислює секретний ключ криптосистеми, або знаходить алгоритм, функціонально еквівалентний алгоритму розшифрування і такий, що не вимагає знання секретного ключа.

Наведемо класифікацію типів атак на системи ЕЦП, запропоновану Голдвассером, Мікаллі і Рівестом. Атаки перераховані таким чином, що кожна наступна сильніша попередньої.

**Атака на основі відомого відкритого ключа**. Зловмисник знає тільки відкритий ключ ЕЦП. Це - найслабша з усіх можливих атак. Очевидно, що зловмисник завжди може провести таку атаку.

**Атака на основі відомих повідомлень**. Зловмиснику відомі відкритий ключ і деякий набір підписаних повідомлень. При цьому зловмисник не може вплинути на вибір цих повідомлень.

**Проста атака з вибором повідомлень**. Зловмисник має можливість вибрати необхідну кількість повідомлень і отримати підписи для них. Передбачається, що ці повідомлення вибираються незалежно від відкритого ключа, наприклад, до того як відкритий ключ стане відомий.

**Спрямована атака з вибором повідомлень**. В умовах попередньої атаки зловмисник, вибираючи повідомлення, вже знає відкритий ключ.

**Адаптивна атака з вибором повідомлень**. Додатково до попереднього випадку зловмисник, знаючи на кожному кроці відкритий ключ і підписи для всіх раніше обраних повідомлень, послідовно вибирає нові повідомлення.

Погрозами для схеми електронного підпису є розкриття схеми або підробка підпису. Голдвассер, Мікаллі і Рівестом уточнюють поняття загрози, визначаючи такі типи загроз (перераховані в порядку підсилення).

**Екзистенціальна підробка** має на меті підробку підпису хоча б для одного повідомлення, яке не було підписано під час атаки. Зловмисник не контролює вибір цього повідомлення. Воно може бути випадковим або безглуздим.

**Селективна підробка**. Підробка підпису для повідомлення, обраного зловмисником. При цьому передбачається, що це повідомлення вибирається апріорі (до початку атаки) і, якщо зловмисник проводить атаку з вибором повідомлень, то повідомлення, до якого потрібно підробити підпис, не може входити до числа обраних під час атаки.

**Універсальна підробка**. Зловмисник знаходить алгоритм, функціонально еквівалентний алгоритму обчислення підпису і такий, що не вимагає знання секретного ключа.

**Повне розкриття** передбачає знаходження секретного ключа.

Як було зазначено вище, стійкість схеми визначається відносно пари (тип атаки, тип загрози). Схема вважається нестійкою проти даної загрози, якщо існує метод її здійснення з ймовірністю, яка не може розглядатися як нехтовно мала.

Системи ЕЦП теоретично вразливі (такі, що дешифруються), оскільки їх практична стійкість базується на обчислювальній складності розв'язання деяких математичних задач (таких, як розкладання чисел на прості множники, тобто факторизація, або знаходження дискретного логарифма в кінцевих полях) і високій вартості коштів або ресурсів, необхідних для їх вирішення.

Потрібно враховувати той факт, що практична стійкість системи повинна бути порівнянна з можливими ризиками для інформації, що характеризують наслідки успішного "злому".

Безумовно, немає сенсу витрачати час і кошти на дешифрування, якщо сукупні доходи (можуть вимірюватися в різних одиницях, в залежності від області застосування криптосистеми) від успішного "злому" істотно нижче вартості витрат на криптоаналіз.

### 3.3 Теоретична стійкість, абсолютно стійкий шифр

Теоретико-інформаційний підхід до оцінювання стійкості шифрів в 1949 році запропонував американський інженер К. Шеннон [38].

На основі ймовірнісної моделі шифру він сформулював поняття абсолютно стійкого шифру і показав, що шифр Вернама на основі випадкової рівноймовірної гами задовольняє вимогу досконалої стійкості.

Хоча кінцевою метою дешифрування є відновлення відкритого тексту або ключа шифрування, та навіть можливість вгадування відкритого тексту з деякою ймовірністю може бути корисна атакуючій стороні.

Для комерційних угод сума тендерної пропозиції, «вгадана» до порядку величини, може зіграти вирішальну роль.

Наприклад, отримавши повідомлення (S={АБВБГДЕБЖЗБИКЛ-МКД…}), зашифроване простою заміною, зловмисник отримає інформацію про збіг низки символів у відкритому тексті, що робить вельми вірогідним припущення про відкритий текст T={ЗАГАЛЬНА_ВАРТІСТЬ…}. Після чого він може спробувати ототожнити наступний шифрований текст з тим чи іншим числовим виразом вартості угоди.

Сказане дозволяє зробити висновок, що для надійного шифру перехоплення шифрованого повідомлення не повинно змінювати уявлення про припустимий зміст відкритого тексту.

Шифр називається **абсолютно стійким** за Шенноном, якщо відкритий і зашифрований тексти статистично незалежні, тобто, для $\forall T, S$, у випадку $P(S) > 0$, має місце рівність умовної і безумовної імовірності: $P(T/S) = P(T)$.

Інакше кажучи, перехоплення шифрованого повідомлення (криптограми) не збільшує обсягу інформації про відкритий текст, якщо ключ шифрування невідомий. При цьому розподіли ймовірностей на множині відкритих текстів до і після перехоплення шифрованого повідомлення (апостеріорний і апріорний розподіл) збігаються.

**Теорема.** Шифр Вернама $T \to S$ за модулем $m$ для повідомлення $T$ довжини $n$: $s_i = (t_i + \gamma_i) \bmod m$, $i = 1, \ldots, n$ є абсолютно стійким шифром, якщо ключ шифрування Г вибирається випадково, рівноймовірно і незалежно з множини всіх можливих $n$-грам в алфавіті $Z_m$.

*Доведення.* Нехай $A = \{T_j\}$ - множина всіх відкритих текстів довжини $n$, $j = 1, \ldots, N$, $N = m^n$. Оскільки ключ Г вибирається випадково і рівноймовірно, то $P(\Gamma) = m^{-n}$.

Для $n$-грами відкритого тексту $T \in A$ і $n$-грами шифртексту $S$ $n$-грама $\Gamma$ визначається однозначно з рівняння шифрування. Таким чином, всі значення гами і шифртексту дозволені, $P(S) > 0$.

При фіксованому $S$ ймовірність пари $T, S$ дорівнює

$$P(T,S) = \sum_{\Gamma: T+\Gamma=S} P(T)P(\Gamma) = P(T)m^{-n}.$$

Звідки, використовуючи визначення умовної ймовірності, отримаємо:

$$P(S/T) = \frac{P(S,T)}{P(T)} = m^{-n}.$$

Апріорні ймовірності відкритих текстів, тобто, коли шифртекст невідомий, рівні між собою.

За формулою Байєса для $P(T/S)$ з останньої рівності слідує:

$$P(T/S) = \frac{P(T)P(S/T)}{P(S)} = \frac{P(T)P(S/T)}{\sum_j P(T_j)P(S/T_j)} =$$

$$= \frac{P(T)m^{-n}}{\sum_j P(T_j)P(S/T_j)} = \frac{P(T)m^{-n}}{m^{-n}\sum_j P(S/T_j)} = P(T),$$ що й потрібно було довести.

### 3.4 Поняття практичної стійкості

Шеннону належить формулювання поняття практичної стійкості, а саме: чи може вирішити завдання дешифрування криптоаналітик, що має в своєму розпорядженні обмежені обчислювальні ресурси і час, а також деякі комплекти перехоплених повідомлень?

В цьому випадку кількісною мірою надійності шифру є обчислювальна складність рішення задачі дешифрування.

Нехай $Al(E)$ - множина застосованих до шифрсистеми $E$ алгоритмів дешифрування. Позначимо $T(\alpha)$ - середня кількість елементарних операцій деякого обчислювача, необхідних для успішної реалізації алгоритму $\alpha \in Al(E)$. Тоді час на реалізацію алгоритму

оцінюється величиною $\tau_\alpha = T(\alpha)/W_{\max}$, де $W_{\max}$ - максимальна продуктивність обчислювальної техніки, що є у розпорядженні атакуючої сторони.

Зокрема, в таблиці 3.1 для деяких криптоалгоритмів наведено орієнтовний час розкриття ключа методом послідовного перебору всіх його допустимих значень для постійної обчислювальної потужності (I) комп'ютера, а також у варіанті (II) її зростання за законом Мура (подвоєння кожен рік).

Таблиця 3.1 - Середній час розкриття ключа методом повного перебору

| **Алгоритм** | **Число ключів** | **Середній час розкриття ключа (років)** | |
|---|---|---|---|
| | | **(I)** | **(II)** |
| **DES** | **$7.2 \cdot 10^{16}$** | **0.45** | **0.45** |
| **DVP** | **$2.4 \cdot 10^{21}$** | **$1.5 \cdot 10^{3}$** | **14** |
| **DVP-XL** | **$7.9 \cdot 10^{28}$** | **$4.9 \cdot 10^{11}$** | **39** |
| **IDEA** | **$2.5 \cdot 10^{38}$** | **$1.6 \cdot 10^{21}$** | **70** |
| **ГОСТ 28147-89** | **$6.3 \cdot 10^{76}$** | **$3.9 \cdot 10^{59}$** | **198** |

Таким чином, для оцінювання практичної стійкості шифрсистеми $E$ доцільно використовувати характеристику: $\tau_E = \min \tau_\alpha$.

Проблема оцінювання практичної стійкості на основі запропонованої методики полягає в тому, що криптограф, що проектує систему, може отримати її лише для відомих йому методів дешифрування.

При цьому потрібно враховувати той факт, що реалізація деяких методів вимагає певних фінансових витрат на створення спеціалізованих обчислювальних засобів.

Зокрема, такий підхід був запропонований Діффі і Хеллманом для оцінювання вартості дешифрування алгоритму DES шляхом створення спеціального обчислювача, побудованого на принципі розпаралелювання процесів перебору всіх варіантів ключів (див. розділ 5).

Важливо мати на увазі, що вартісний підхід до оцінювання стійкості шифру є відносною величиною, оскільки швидкий розвиток мікроелектроніки різко знижує вартість створення відповідних засобів. Наприклад, первинна оцінка вартості розкриття DES була зменшена на два порядки протягом десятка років.

Важливим чинником оцінювання стійкості є обсяг матеріалу, необхідного для розкриття шифрсистеми.

Вочевидь, що для коротких шифрованих повідомлень деякого потокового шифру може існувати декілька пар ключів та відкритих текстів таких, що виконується рівність: $S = T_1 \oplus K_1 = T_2 \oplus K_2 = \ldots$ .

Неважко помітити, що, наприклад, для шифру Вернама кількість таких пар дорівнює числу відкритих повідомлень довжини $N$, тобто $2^{NH}$, де H – ентропія відкритого тексту. Для кожної пари обчислюється свій ключ і лише один з них є істинним. У таких умовах відновлення істинного ключа стає проблематичним завданням.

Якщо довжина повідомлення дорівнює одному символу в коді Ascii (8 бітів), то кількість пар можливих пар досягає 256. При зростанні довжини повідомлення, через надмірність реальних мов, кількість помилкових варіантів дещо скорочується.

**Відстанню єдиності** $L_0$ шифру $E$ називається найменше натуральне число (якщо воно існує), рівне довжині шифрованого повідомлення, для якого істинний ключ визначається однозначно (тобто, очікуване число помилкових варіантів дорівнює нулю).

Для криптосистем з кінцевою множиною ключів $K$ відстань єдиності може бути отримана з формули

$$L_0 = \left\lceil \frac{\log_2 |K|}{R_A \log_2 m} \right\rceil,$$

де $R_A$ – величина надмірності мови з алфавітом, що містить $m$ знаків.

*Вправа.* Розрахувати відстань єдиності для шифру простої заміни на алфавіті $m = 33,\ R_A = 0{,}7$.

*Вказівка:* скористатися формулою Стірлінга для оцінювання числа $33!$.

На основі сказаного можна зробити висновок про те, що в разі дуже коротких повідомлень навіть шифр простої заміни може бути достатньо стійким через неоднозначність рішення задачі дешифрування.

До того моменту, як довжина шифрованого повідомлення (обсяг всіх повідомлень, що є у розпорядженні криптоаналітика) не досягне

відстані єдиності, завдання дешифрування полягає в пошуку всіх рішень, що мають найбільшу ймовірність.

Після того, як довжина шифрованого повідомлення (загальний обсяг перехоплених повідомлень) досягне величини відстані єдиності, можливе відновлення істинного ключа.

Середні трудовитрати $W(N)$, вимірювані числом елементарних операцій, необхідних для знаходження істинного ключа на основі шифрованого повідомлення довжиною $N$ символів, називають **робочою характеристикою шифру** (рис. 3.2).

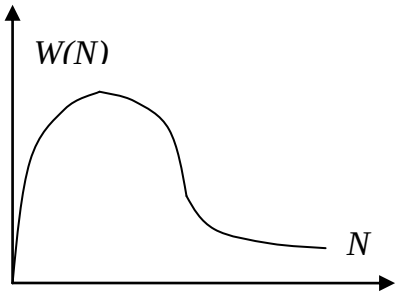



Рисунок 3.2 - Робоча характеристика шифру

Зі збільшенням обсягу перехоплення кількість необхідної роботи зменшується, наближаючись до деякого асимптотичного значення.

Певний інтерес викликає граничне значення робочої характеристики при необмеженому обсязі шифрованих повідомлень $W(\infty)$.

Практичне обчислення цієї величини є дуже складним завданням, тому, зазвичай, вона оцінюється досягнутою робочою характеристикою, яка визначається середньою трудомісткістю найкращого з відомих методів розкриття шифру.

Аналіз практичної стійкості починається з накопичення і аналізу інформації про конкретний шифр, вихідні дані для його проектування, результати пробної експлуатації і наукові публікації.

Аналіз практичної стійкості виходить з того, що сам шифр повністю відомий, включаючи особливості управління ключами, відомі не лише шифровані тексти, а також деяка кількість відкритих текстів.

Виходячи з цих допущень для оцінювання практичної стійкості необхідно:

– визначити і точно сформулювати математичні та обчислювальні завдання, які необхідно вирішити для відновлення ключа;

– проаналізувати можливість використання найкращих з числа відомих до моменту аналізу алгоритмів вирішення поставлених завдань;

– провести всебічне дослідження тенденції розвитку звичайних обчислювальних засобів і оцінити вартість створення спеціальних обчислювачів (див. «Принципи побудови і використання блокових алгоритмів шифрування»).

При оцінюванні практичної стійкості слід приділити увагу революційним новаціям в питанні підвищення швидкодії обчислювальної техніки, це стосується таких напрямів, як створення квантових обчислювачів, вживання нейронних мереж, кластерних систем, використання методів розпаралелювання алгоритмів й ін.

**Питання до розділу 3**

1. Яким чином класифікуються рівні можливостей зловмисника, що атакує криптосистему?
2. Реалізація яких ризиків для криптосистем є потенційно найбільш ймовірною?
3. Яка атака на симетричну криптосистему вважається найбільш слабкою?
4. У чому полягає проста атака з вибором відкритого тексту?
5. У чому суть так званої екзистенціальної підробки цифрового підпису?
6. Який шифр називається абсолютно стійким за Шенноном?
7. Чи існує критерій (для впізнання) відкритого тексту в разі абсолютно стійкого шифру Вернама?
8. Чи існує абсолютно стійка асиметрична криптосистема?
9. Як визначається практична стійкість шифру?
10. Що таке відстань єдиності шифру?
11. У чому суть поняття робочої характеристики шифру?
12. З чого слід починати аналіз практичної стійкості шифру?

# РОЗДІЛ 4 ПСЕВДОВИПАДКОВІ ПОСЛІДОВНОСТІ І МЕТОДИ ГЕНЕРАЦІЇ КЛЮЧОВИХ ДАНИХ

## 4.1 Дані для формування ключової інформації

Слід звернути увагу на те, що значна кількість атак на криптосистеми базується на особливостях конкретних ключових даних.

Зокрема, досконала стійкість шифру Вернама досягається за рахунок випадкового рівноймовірного розподілу знаків гамової послідовності. У той же час, нерівномірний розподіл секретних параметрів алгоритму, елементів ключа, наявність статистичних залежностей між ними створюють передумови для проведення ефективних атак.

Зокрема, стандарти цифрового підпису ГОСТ 34.310 (Росія, Україна) і DSS (США) використовують схему Ель-Гамаля, для якої обчислення цифрового підпису $R, S$ виробляється за формулами

$$R = a^k \bmod p,\ S = k^{-1}(H + xR) \bmod (p-1),$$

де $H = H(M)$ – хеш-код повідомлення $M$.

Секретними параметрами є особистий ключ $x$ і випадковий одноразовий ключ $k$, який, зазвичай, називається рандомізатором.

Якщо при формуванні величини $k$ не забезпечується рівномірний розподіл зустрічності різних значень, то може стати реальною атака на основі часткового перебору найбільш вірогідних значень $k$ в комбінації з подальшим розв’язанням приведених рівнянь відносно секретного ключа.

Таким чином, виникає три тісно пов'язаних між собою питання.

1. Що таке випадкова рівномірно розподілена послідовність, призначена для формування ключових даних?

2. Як за поліноміальний час отримати послідовність символів з деякого алфавіту, що має певний набір властивостей, достатній для того, щоб її можна було б вважати реалізацією послідовності незалежних випадкових величин з рівномірним розподілом?

3. Як визначити, що деяка фіксована послідовність задовольняє вимоги випадковості і рівномірності щодо вимог до ключових даних?

Далі розглянемо послідовності випадкових величин $x_t$ з дискретним розподілом, де для всіх $t$ випадкові величини $x_t \in Z_m$.

Відповідно до найбільш поширеного визначення [5, 9] дискретна рівномірно розподілена випадкова послідовність (РРВП) $X = \{x_t, t \geq 1\}$ – це послідовність незалежних в сукупності випадкових величин, для яких $\forall i \in Z_m \; P(x_t = i) = m^{-1}$.

РРВП має такі властивості [5]:

1) $M(x_t) = (m-1)/2$, $D(x) = (m^2 - 1)/12$;

2) Для всіх $k > 0$ будь-яка $k$-вимірна впорядкована вибірка (вектор з компонентами $x_t$) має рівномірний розподіл з вірогідністю $m^{-k}$;

3) Будь-яка підпослідовність послідовності $X$ також РРВП;

4) Сума за модулем $m$ РРВП $X$ і будь-якої незалежної від неї послідовності також є РРВП;

5) РРВП не може бути передбачена, тобто для будь-якого натурального $n$ кількість інформації за Шенноном, що міститься у відрізку $X_n = (x_1, \ldots, x_n)$, про елемент $x_{n+1}$ дорівнює нулю: $I(x_{n+1}, X_n) = 0$.

Пристрій, що реалізує РРВП, називається **генератором РРВП**.

Нехай $X = (x_1, \ldots, x_T \ldots)$ – двійкова послідовність, що має деякий період $T$. Близькі за суттю вимоги до вихідних даних для формування ключів висуваються в таких постулатах Голомба.

1. Необхідно, щоб кількість одиниць і кількість нулів відрізнялися на кожному (тобто, взятому з будь-якого місця) періоді не більше, ніж на одиницю.

2. У множині всіх серій (відрізків різної довжини, що складаються підряд з нулів, або одиниць) рівно половина серій повинна складатися з серій довжиною один, одна чверть (половина від половини, що залишилася) повинна складатися з серій довжиною два, серії довжиною три повинні займати одну восьму всієї множини серій і так далі.

3. Нехай дано дві копії послідовності періоду $T$, що мають зсув одна відносно одної на величину $0 \leq d \leq T-1$. Нехай $A_d$ і $B_d$ – відповідно, кількість збігів і незбігів бітів в цих послідовностях. Назвемо **функцією автокореляції** вираз $f(X, d) = (A_d - B_d)/T$.

Необхідно, щоб для будь-яких допустимих значень $d$ функція $f(X, d)$ набувала лише два різні значення.

Останнє правило формулює необхідну умову незалежності знаків послідовності $X$. Одночасно – це деяка міра відмінності послідовності $X$ і її копії, яка починається в іншій точці циклу.

Послідовність, яка задовольняє постулати Голомба, називається **псевдошумовою** послідовністю (ПШП).

Відмітимо, що лінійні рекурентні послідовності максимального періоду по суті псевдошумові. Це свідчить, що постулати були сформульовані, виходячи з умови збереження позитивних статистичних властивостей, характерних для лінійних рекурентних послідовностей максимального періоду.

На практиці одним з найважливіших є таке завдання. Виходячи з вищеперерахованих та інших властивостей РРВП, необхідно визначити, чи є конкретна послідовність реалізацією РРВП.

Надалі, для скороченого запису, реалізацію РРВП називатимемо просто випадковою послідовністю. Для вирішення цього завдання, перш за все, необхідно дати визначення РРВП, яке практично застосовується.

Існує декілька визначень випадкової послідовності, але з різних причин використовувати на практиці можна лише деякі з них.

Один з підходів до визначення випадкової послідовності, що базується на теорії інформації, сформульований К. Шенноном. Послідовність називається випадковою за Шенноном, якщо вміст інформації в ній максимальний.

Інакше це можна сформулювати так: у послідовності відсутня надмірність; ентропія послідовності максимальна. Відзначимо, що випадковість тут характеризується як властивість генератора.

Якщо взяти за основу це визначення, то досить часто використовувані на практиці генератори випадкових чисел, побудовані на основі будь-якого закону алгебри перетворення деякого початкового випадкового числа (генератори псевдовипадкових чисел – ГПВЧ), є неякісними. Дійсно, оскільки вони отримані з відносно короткого вектора – початкового заповнення, то вони обов'язково надлишкові, а ентропія всієї послідовності дорівнює ентропії початкового вектора.

Інший підхід запропонований А. М. Колмогоровим, він базується на алгоритмічній складності обчислень. Послідовність довжини $n$ називається випадковою за Колмогоровим, а її колмогорівська складність рівна $n$, якщо її не можна отримати в результаті роботи якого-небудь

алгоритму, двійковий запис найкоротшої програми для реалізації якого містить менше $n$ бітів.

Наприклад, колмогорівська складність послідовності з $n$ одиниць рівна $\log_2 n + O(1)$, оскільки її можна задати за допомогою алгоритму "надрукувати $n$ одиниць", довжина входу якого порядку $\log_2 n$, тому що стільки займає двійковий запис числа $n$ (лічильник циклу друкування символу).

Сама програма друкування символу має фіксовану довжину, незалежно від $n$.

Подібно шеннонівському визначенню, випадкова за Колмогоровим послідовність не може бути згенерована поліноміальним алгоритмом з короткого початкового стану.

Ще один підхід до визначення випадкової послідовності запропонували Блюм, Голдвассер, Мікаллі і Яо.

У відповідному визначенні послідовність називається випадковою, якщо не існує поліноміального (імовірнісного) алгоритму, який зможе відрізнити її від суто випадкової.

Така послідовність називається **поліноміально невідрізненною від випадкової або псевдовипадковою.**

Даний підхід дозволяє використовувати для формування псевдовипадкових послідовностей (ПВП) **детерміновані** алгоритми, що реалізовуються кінцевими автоматами. Хоча з математичної точки зору такі послідовності невипадкові, оскільки вони повністю визначаються початковим заповненням, проте їх практичне використання не дає жодних переваг криптоаналітику завдяки «нерозрізненості» від випадкових. Оскільки цей підхід вважається конструктивнішим, розглянемо його детальніше.

Випадкові послідовності в розумінні останнього визначення також називають «випадковими для всіх практичних вживань».

Генератори таких послідовностей називають криптографічно сильними, криптографічно стійкими (cryptographically strong) або криптографічно надійними (cryptographically secure). Надійність, в даному випадку, є не лише властивістю послідовності (або генератора), але і властивістю спостерігача, а точніше, його обчислювальних можливостей.

Для ПВП доведено два важливі твердження.

1. Послідовність є псевдовипадковою тоді і лише тоді, коли вона **непередбачувана, тобто витримує тестування черговим бітом.**

Це означає, що якщо навіть відома частина послідовності будь-якої довжини, то при невідомих початковому заповненні генератора і параметрах алгоритму генерації для здобуття чергового біта не можна запропонувати алгоритм, істотно кращий простого вгадування або підкидання монети.

2. Криптографічно сильні генератори існують в тому і лише в тому випадку, якщо існують легко обчислювані функції, але обчислювально складно обернені (однобічні функції – one-way functions). В цьому випадку кожному генератору ПВП можна поставити у взаємно однозначну відповідність деяку однобічну функцію, яка залежить від певних параметрів.

### 4.2 Статистичне тестування ПВП

З врахуванням викладеного, вважатимемо послідовність символів деякого алфавіту псевдовипадковою у вузькому сенсі, якщо жоден алгоритм з деякого набору поліноміальних алгоритмів не може її відрізнити від випадкової.

Алгоритми цього набору називатимемо **тестами** (критеріями) на випадковість, а процес перевірки послідовності – **тестуванням.**

Якщо тест не зміг відрізнити послідовність від випадкової, то говоритимемо, що послідовність проходить або витримує даний тест. Інакше говоритимемо, що тест відхиляє послідовність.

Отже, питання про перевірку випадковості послідовності зводиться до питання побудови відповідного набору тестів. Для цього спочатку необхідно визначити принцип їх роботи, а саме: яким чином тест повинен приймати одні послідовності і відхиляти інші?

Однією з відповідей на це питання є перевірка наявності у заданої послідовності деякої властивості, характерної для будь-яких випадкових послідовностей.

Наприклад, хай елементи послідовності вибираються незалежно і рівноймовірно з множини $Z_2$. В цьому випадку послідовність з $n$ одиниць погано підходить на роль випадкової для практичних вживань, хоча

вірогідність її появи рівна $2^{-n}$, як і будь-якої іншої реалізації з множини $Z_2^n$.

Ця послідовність не буде **типовою** з точки зору появи в РРВП досить великої довжини істотного дисбалансу між числом одиниць і нулів. У той же час, краще вважати типовою послідовність, в якої кількість одиниць і нулів неістотно відрізняються, оскільки такі послідовності переважають в множині $Z_2^n$.

Надалі тести, побудовані на властивостях зустрічності елементів послідовності, називатимемо **частотними,** оскільки вони перевіряють розподіл частот зустрічності і виявляють його відхилення від рівноймовірного розподілу (у т. ч.: критерії $\chi^2$, максимальної правдоподібності і максимальної частоти).

Повертаючись до роботи тесту, відмітимо, що його результатом завжди буде прийняття або відхилення гіпотези $H_0$, яка передбачає, що конкретна послідовність є послідовністю незалежних, рівномірно розподілених випадкових величин з відомим розподілом.

Її альтернативою вважатимемо гіпотезу $H_1$, яка містить як всі можливі випадки нерівномірних розподілів елементів послідовності, так і всі можливі види залежностей між ними.

Отримавши результати тестування деякої послідовності, ми робимо висновок про її походження з чималою (але меншої одиниці) ймовірністю, яка залежить від параметрів тесту. З кожним тестом пов'язано два параметри.

1. $\alpha$ – ймовірність помилки 1-го роду – ймовірність марно відхиляти істинно випадкову послідовність, тобто прийняти гіпотезу $H_1$, тоді як правильна гіпотеза $H_0$ (можна сказати, марно прийняти $H_1$). Величина α також називається рівнем значущості тесту.

2. $\beta$ – ймовірність помилки 2-го роду – ймовірність марно прийняти невипадкову послідовність за випадкову, тобто прийняти гіпотезу $H_0$ за умови, що істинна гіпотеза $H_1$.

Рівень значущості задається до початку роботи тесту, при цьому він визначає відносну частину суто випадкових послідовностей, що не мають властивостей, наявність яких перевіряє тест. Зазвичай $\alpha$ вибирається з

практичних міркувань. Якщо ймовірність гіпотези, що повіряється, досить висока, то рівень значущості вибирається маленьким: $10^{-2} - 10^{-3}$.

Параметр $\beta$ залежить від $\alpha$ і довжини послідовності $n$ так, що при фіксованому $\alpha$ із зростанням $n$ величина $\beta$ зменшується. Оскільки гіпотеза $H_1$ складна, то з множини тестів не можна вибрати єдиний тест, який відповідає рівномірно найбільш потужному критерію, тобто такий, який при заданому рівні значущості $\alpha$ мінімізує значення $\beta$.

В ході роботи тесту, виходячи з тестованої послідовності, обчислюється деяка величина $U$, звана **статистикою**. Часто тест побудований так, що він приймає послідовність, тобто, не відкидає $H_0$, якщо $U \le U_\alpha$, де $U_\alpha$ називається пороговим значенням статистики. Тест називається двостороннім, якщо при його перевірці використовується нерівність вигляду $|U| \le U_\alpha$.

Значення $U_\alpha$ розраховується за умови істинності гіпотези $H_0$. Якщо $U > U_\alpha$, то послідовність відхиляється. Ймовірність в цьому випадку марно відкинути, послідовність повинна дорівнювати $\alpha$.

Тому величина $U_\alpha$ визначається із співвідношення $\Phi(U_\alpha) = 1 - \alpha$, де $\Phi(U)$ – функція розподілу ймовірності, $\alpha$ – рівень значущості тесту.

В цьому випадку графік функції $\Phi(U)$ розбивається абсцисою $U_\alpha$ на дві частини так, що при істинній гіпотезі $H_0$ ймовірність появи значень статистики, що не перевищують $U_\alpha$, рівна $1 - \alpha$.

Відповідно, ймовірність появи значень статистики, що перевищують $U_\alpha$, рівна $\alpha$.

Дуже часто функція $\Phi$ – монотонно зростаюча, тому для перевірки нерівності $U > U_\alpha$ можна порівнювати $\Phi(U)$ і $\Phi(U_\alpha)$.

В цьому випадку $1 > \Phi(U) > \Phi(U_\alpha) > 0$, звідки $1 - \Phi(U) < 1 - \Phi(U_\alpha) = \alpha$.

Зручно ввести значення $P_U = 1 - \Phi(U)$, яке називається $P$-величиною. Оскільки $\Phi(U)$ – ймовірність появи значень статистики, що не перевищує $U$, то $P_U$ – ймовірність появи значень статистики більших, ніж $U$.

Отже, на підставі результатів тесту послідовність вважається випадковою, якщо $P_U < \alpha$.

Наприклад, для перевірки виду розподілу тестованої послідовності можна скористатися статистикою $\chi^2$: $\chi^2 = \sum_{i=0}^{m-1} \frac{(\nu_i - Np_i)^2}{Np_i}$.

Тут $\nu_i$ – частота зустрічності символу $i \in Z_m$ в послідовності знаків шифрованого тексту, $N = \nu_0 + \ldots + \nu_{m-1}$ – довжина тестованої послідовності, $(p_0, \ldots, p_{m-1})$ – вектор розподілу ймовірності символів, відповідний гіпотезі $H_0$. Статистика $\chi^2$ при $N \to \infty$ має так званий розподіл $\chi^2$ Пірсона з $k$ ступенями свободи, де $k = m-1$. Цей розподіл табульовано.

При $k \geq 30$ замість $\chi^2_k$ зазвичай використовуються різні апроксимації за допомогою нормального розподілу.

В разі РРВП має місце $(p_0, \ldots, p_{m-1}) = (1/m, \ldots, 1/m)$. Нехай величина $\chi^2_{m-1,\alpha}$ – квантиль рівня $1-\alpha$ розподілу $\chi^2$ з $m-1$ ступенем свободи, відповідна рівню значущості $\alpha$, тобто аналог $U_\alpha$: $P(\chi^2 \geq \chi^2_{m-1,\alpha}) = \alpha$.

Якщо для вибірки виконується нерівність: $\chi^2 < \chi^2_{m-1,\alpha}$, то приймається гіпотеза $H_0$ про випадковий характер послідовності, інакше – гіпотеза $H_1$ відкидається.

Відзначимо, що відповідно до експериментальних даних з гіпотезою про наявність у послідовності певної властивості не свідчить про протиріччя цих же даних з іншою гіпотезою.

На основі спостереження реалізації послідовності за допомогою статистичних критеріїв не можна довести справедливість тієї або іншої гіпотези, а можна лише стверджувати, що результати спостережень не суперечать даній гіпотезі.

Розглянемо основні вимоги до **набору тестів.** Зрозуміло, що набір повинен містити деяку мінімально необхідну множину тестів, яка визначається майбутнім застосуванням тестованих послідовностей, і може бути різним у кожному конкретному випадку.

Наприклад, набір тестів, рекомендований Національним інститутом стандартів США NIST для тестування двійкових послідовностей [39], включає 16 тестів (в т. ч. тести: частотний монобітовий; частотний блоковий; тести серій і довгих серій одиниць; рангу випадкової {0,1}-матриці і ін.).

Більшість тестів з набору NIST призначена для тестування двійкових послідовностей довжиною від $10^3 - 10^6$, проте деякі з них можуть використовуватися для тестування послідовностей довжиною від $10^2$ елементів (монобітовий і блоковий частотні тести, тест серій, тест серій максимальної довжини, тест частот $\nu$-грам з перекриттям й ін.). Можна створити аналоги тестів цього набору для тестування послідовностей елементів з довільного алфавіту.

Кількість тестів в деякому їх наборі визначається, виходячи з так званої загальної помилки 1-го роду, тобто тої частини випадкових послідовностей, які не пройдуть хоч би один з тестів.

Загальна помилка 1-го роду $A(n,\alpha)$ знаходиться для заданих помилок 1-го роду кожного з тестів за умови їх незалежності. Якщо для всіх тестів задане одне значення $\alpha$, то ймовірність того, що випадкова послідовність не пройде, принаймні, один тест з набору, рівна

$$A(n,\alpha) = 1 - (1-\alpha)^n = 1 - \left(1 + \frac{1}{n/(-n\alpha)}\right)^{\frac{n}{-n\alpha}(-n\alpha)} \approx 1 - e^{-n\alpha},$$

оскільки $1/\alpha$ швидко зростає зі зменшенням $\alpha$.

Для вибраної величини $A$ кількість тестів визначається як $n = -\ln(1 - A(n,\alpha))/\alpha$ із співвідношення $A \approx 1 - e^{-n\alpha}$.

Відмітимо, що зі зростанням $n$ величина $A$ прямує до 1, тому кількість тестів не має бути надто великою.

*Приклад.* Якщо в кожному з 16 тестів з набору, рекомендованого NIST, помилка 1-го роду дорівнює 0.01, то $A(n,\alpha) = 0.15$, тобто близько 15% суто випадкових послідовностей не пройде хоч би один з тестів набору.

Важливим чинником «об'єктивного» тестування є незалежність тестів. Для перевірки їх незалежності використовується $k$ “еталонних” послідовностей $S_1,\ldots,S_k$, отриманих, наприклад, з використанням фізичних датчиків випадкових чисел. Величина $k$ має бути достатньою ($k \geq 500$) для застосування критерію незалежності $\chi^2$ виду

$$\chi^2 = N\sum_{i,j}\frac{\nu_{i,j}}{\nu_{i\bullet}\nu_{\bullet j}}, \text{ де } \nu_{i\bullet} = \sum_j n_{ij}, \text{ а } \nu_{\bullet,j} = \sum_i n_{ij}.$$

Запропонована така емпірична методика перевірки незалежності тестів. Позначимо $T_1,\ldots,T_n$ – події, відповідні фактам проходження випадковою послідовністю сукупності тестів $\hat{T}_1,...,\hat{T}_n$ відповідно.

Необхідно перевірити виконання рівності $P(T_1\cdot T_2\cdots T_n)=\prod_{i=1}^{n}P(T_i)$.

За формулою множення ймовірностей:

$$P(T_1\cdot T_2\cdots T_n)= P(T_1/T_2\cdot T_3\cdots T_n)\cdot P(T_2/T_3\cdot T_4\cdots T_n)\cdots P(T_{n-1}/T_n)P(T_n).$$

Отже, для перевірки незалежності тестів в сукупності потрібно перевірити незалежність $(n-1)$-ої пари подій вигляду $P(T_i/T_{i+1}\cdots T_n)$.

Для цього на основі «еталонних» послідовностей $S_1,\ldots,S_k$ побудуємо нові 0,1-послідовності $M_i$, $i=1,\ldots,n$ таким чином.

$$M_1=(T_1(S_1),\ldots,T_1(S_k)),\ \ldots,\ M_2=(T_n(S_1),\ldots,T_n(S_k)),$$

де $T_i(S_j)$ рівне 1, якщо послідовність $S_j$ проходить тест $T_i$, або нулю в іншому випадку.

Після цього побудуємо послідовності $L_k$:

$$L_1=M_1,\ L_2=M_1\cdot M_2,\ \ldots,L_{n-1}=M_1\cdot M_2\cdots M_{n-1},\ k=1,\ldots,n-1,$$

де операція «·» означає множення елементів послідовностей, розташованих на однакових місцях.

Далі, застосовуючи $n-1$ раз критерій незалежності $\chi^2$, перевіряємо попарну незалежність для кожної з наступних пар послідовностей: $L_1,M_2$; $L_2,M_3$ і так далі, до $L_{n-1},M_n$.

За результатами тестування приймається рішення про незалежність системи тестів.

Як вже було відмічено, тестуванню підлягають:

– криптографічні властивості конкретних послідовностей;

– властивості генератора, використовуваного для створення випадкових послідовностей.

Для статистики найбільш істотною особливістю будь-якої послідовності є її довжина. На практиці окремо розглядаються питання тестування послідовностей «великої» і «малої» довжини.

Основною причиною диференційованого підходу до цих двох випадків є той факт, що властивості випадкової послідовності стають тим більше виразними, чим більше її довжина.

Вважається, що мінімальна довжина послідовності для частотних тестів має бути в 5-10 разів більша обсягу алфавіту. В такому разі за умови рівномірного розподілу символів послідовності ймовірність того, що кожен символ зустрінеться щонайменше 1 раз, наближається до одиниці.

Дійсно, в цих умовах ймовірність того, що якийсь символ не зустрівся жодного разу, надзвичайно мала:

$$\left(1-\frac{1}{m}\right)^{N}=\left(1-\frac{1}{m}\right)^{10m}<e^{-10}\approx 5\cdot 10^{-5}.$$

Інші тести можуть висувати ще жорсткіші вимоги до довжини послідовності.

Оскільки будь-яка підпослідовність РРВП також є РРВП, разом з тестуванням всієї реалізації необхідно тестувати її різні підпослідовності.

Ці підпослідовності повинні вибиратися за допомогою деяких правил, причому так, щоб номери місць символів, які вибираються, не залежали б від значень самих символів.

Наприклад, не слід вибирати підпослідовність, яка складається лише з нулів.

Правила вибору підпослідовностей можуть бути такими:

– кожен $j$-й елемент вихідної послідовності;

– кожен елемент, перед яким зустрічається деяка фіксована комбінація символів;

– елементи, номери місць яких визначаються деякою послідовністю, незалежною від даної і так далі.

Тестування генераторів випадкових послідовностей (чисел) складається з двох етапів:

– тестування достатньої кількості ($\approx 10^3$) послідовностей за допомогою підібраного набору тестів;

– обробка отриманих результатів.

Для генератора випадкових послідовностей повинні виконуватися такі міркування:

– якщо послідовність випробовується тестом, для якого заданий рівень значущості $\alpha$, то з ймовірністю $\alpha$ послідовність не пройде цей тест (наприклад, при $\alpha = 0.01$, в середньому, не пройде тест 1 з 100 послідовностей);

– вихідні послідовності генератора повинні мати рівномірний розподіл відповідних їм значень $P_U$.

Методика NIST обробки результатів тестування генераторів передбачає такі дії.

1. Обчислення пропорції послідовностей, які пройшли тест.

Нехай було протестовано $k$ послідовностей, з них $k_1$ пройшли тест. Тоді пропорція дорівнює $k_1/k$. Якщо рівень значущості тесту рівний $\alpha$, то повинна виконуватися нерівність:

$$1-\alpha-3\sqrt{\frac{\alpha(1-\alpha)}{k}} \le \frac{k_1}{k} \le 1-\alpha+3\sqrt{\frac{\alpha(1-\alpha)}{k}}. \qquad (4.1)$$

2. Перевірка рівномірності розподілу $P$-величин.

Підраховуються числа: $F_i$ – кількість $P$-величин, що належать інтервалам $[\frac{(i-1)}{10}, \frac{i}{10}]$, де $i = 1,\ldots,10$, після чого обчислюється статистика

$$\chi^2 = \sum_{i=1}^{10} \frac{(F_i - 0.1k)^2}{0.1k},$$

визначається значення $P_T = igams\ (4.5,\ 0{,}5\,\chi^2)$,

де *igams* – неповна гама-функція, обчислювана за формулою:

$$\Theta(a,x)=\frac{1}{\Gamma(a)}\int_x^{\infty}t^{a-1}e^{-t}dt,$$

де $\Gamma(z)=\int_0^{\infty}t^{z-1}e^{-t}dt$.

3. Якщо виконується нерівність $P_T \geq 0.0001$, то $P$-величини вважаються рівномірно розподіленими, а генератори з вказаними властивостями, за термінологією NIST, називаються **санкціонованими** (approved).

Санкціонований генератор, залежно від набору тестів, може бути досить хороший для формування гами однократного використання та інших криптографічних застосувань.

### 4.3 Генератори псевдовипадкових послідовностей

Відзначимо, що для створення РРВП використовуються три типи генераторів: табличні або фізичні генератори, а також генератори ПВП.

Прикладом табличного генератора може слугувати опублікована в 1955 році компанією Rand Corporation таблиця обсягом $10^6$ випадкових цифр.

Фізичні генератори набули широкого поширення після створення мікропроцесорів, що мають невисоку вартість за умови достатньої продуктивності. На рис. 4.1 показано фізичний генератор випадкових даних ORB, реалізований компанією APA Consulting на мікроконтролері сімейства PIC12C67X (8-ми контактний корпус SOIC розміром 5.3(8.1мм)).

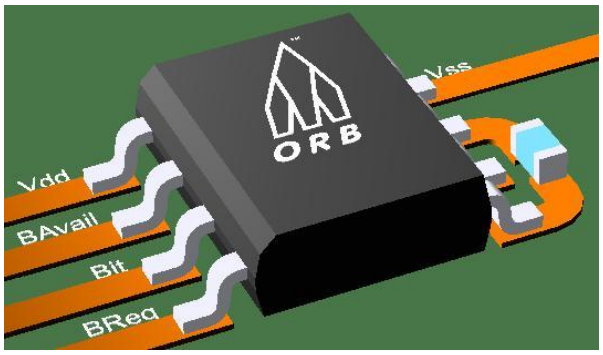



Рисунок 4.1 - Генератор випадкових чисел ORB

В основу роботи даного генератора покладений принцип вимірювання напруги на конденсаторі, який заряджається і розряджається відповідно до деякого потоку бітів.

Табличні та фізичні генератори разом з гарними статистичними властивостями мають ряд недоліків, до головних з яких можна віднести складність технічної реалізації, відносно невисоку швидкодію і високу вартість.

Через названі причини при побудові програмних і програмно-апаратних засобів криптографічного захисту інформації широкого поширення набули генератори ПВП.

Найбільш простим програмним датчиком псевдовипадкових чисел є **лінійний конгруентний генератор** (ЛКГ), який описується рекурентним рівнянням виду $X_n = (aX_{n-1} + b) \bmod N$, де $X_0$ – випадкове початкове значення, $a$ – множник, $b$ – приріст, $N$ – модуль.

Період вихідної послідовності такого генератора не перевищує $N$, максимальне значення досягається при правильному виборі параметрів $a, b, N$, а саме, коли:

– числа $N$ і $b$ взаємно прості: НСД $(N, b) = 1$;

– $a - 1$ кратно будь-якому простому $p$, що ділить $N$;

– $a - 1$ кратно 4, якщо $N$ кратно 4.

У [10] наведений список констант для ЛКГ, що забезпечують максимальний період послідовності і, що не менше важливо, відповідні послідовності проходять статистичні тести.

Для реалізації ЛКГ на персональних комп'ютерах з урахуванням їх розрядної сітки нерідко використовується модуль $N = 2^{31} - 1 \approx 2.14 \cdot 10^9$. При цьому найбільш якісні статистичні властивості ПВП досягаються для константи $a = 397204094$.

У порівнянні з іншими видами генераторів ПВП даний вид забезпечує високу продуктивність за рахунок малого числа операцій для створення одного псевдовипадкового біта.

Недоліком ЛКГ в плані їх використання для створення потокових шифрів є передбачуваність вихідних послідовностей.

Ефективні атаки на ЛКГ були запропоновані Джоан Бояр.

Їй належать методи атак на квадратичні і кубічні генератори: $X_n = (aX_{n-1}^2 + bX_{n-1} + c) \bmod N$ і $X_n = (aX_{n-1}^3 + bX_{n-1}^2 + cX_{n-1} + d) \bmod N$.

Інші дослідники узагальнили результати робіт Бояр на випадок загального поліноміального конгруентного генератора. Стерн і Бояр показали, як зламати ЛКГ, навіть якщо відома не вся послідовність.

Вішман і Хілл, а пізніше П'єр Ель-Ек'югер вивчили комбінації ЛКГ. Ускладнення не є стійкішими криптографічно, але мають великі періоди і краще поводять себе на деяких критеріях випадковості.

**Регістри зсуву з лінійним зворотним зв'язком** (Linear Feedback Shift Registers – LFSR) включають власне регістр зсуву і схему обчислення функції зворотного зв'язку (tap sequence) (рис. 4.2).

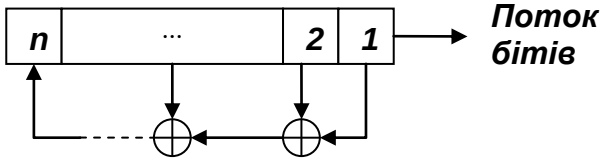



Рисунок 4.2 - Регістр зсуву з лінійним зворотним зв'язком (LFSR)

На схемі вміст регістра – послідовність бітів – зсувається з приходом тактового імпульсу (clock pulse) на один розряд вправо. Біт наймолодшого розряду вважається виходом LFSR в даному такті роботи. Значення найстаршого розряду при цьому є результатом додавання за модулем 2 (функція XOR) розрядів (точок знімання) зворотного зв'язку. Послідовність, що генерується, називається лінійною рекурентою.

Теоретично, $n$-бітовий LFSR може згенерувати псевдовипадкову послідовність з періодом $2^n - 1$ бітів. Такі LFSR називаються регістрами максимального періоду [1].

Для цього регістр зсуву повинен побувати в усіх $2^n - 1$ ненульових внутрішніх станах.

Одна і та ж рекурента може бути згенерована регістрами різної довжини. Припустимо, що серед подібних регістрів наш $n$-бітовий LFSR має мінімальну довжину.

Функції зворотного зв'язку регістра можна поставити у відповідність поліном $m(x)$ степеня не більше $n$ з коефіцієнтами з поля лишків за модулем два, що складається з одночленів вигляду $x^{n_i - 1}$, де $\{n_i\}$ - множина номерів точок знімання зворотного зв'язку.

Поліном $m(x)$ називається **мінімальним поліномом** відповідної рекурентної послідовності.

Для кожної кінцевої (або періодичної) послідовності $S$ можна вказати LFSR, який, при деякому початковому заповненні, породжує цю послідовність.

Серед всіх таких регістрів існує регістр мінімальної довжини $L$.

Величина $L$ називається **лінійною складністю** послідовності $S$.

Нагадаємо, що поліном називається незвідним, якщо він не може бути виражений як добуток двох поліномів меншого степеня, відмінних від констант.

**Примітивний поліном** степеня $n$ над полем лишків за модулем два – це незвідний поліном, який ділить $x^{2^n-1}-1$, але не ділить $x^d-1$ для будь-яких $d: d\left|2^{n-1}\right.$.

**Теорема.** Для того, щоб послідовність, породжена LFSR, мала максимальний період, необхідно і достатньо, щоб її мінімальний поліном був примітивним поліномом за модулем 2.

Список практично застосовуваних примітивних поліномів наведений в [1, 7]. Наприклад, примітивним поліномом є $x^{37}+x^7+x^5+x^3+x^2+x+1.$

Набір показників (32, 7, 5, 3, 2, 1, 0) означає, що, взявши регістр зсуву довжиною 32 і генеруючи біт зворотного зв'язку шляхом додавання 7-го, 5-го, 3-го, 2-го і 1-го бітів за модулем 2, ми отримаємо LFSR максимальної довжини (із $2^{32}-1$ станами).

Наведемо програму мовою С для послідовності, що генерується даним LFSR:

```
Int LFSR() {
Static unsigned long ShiftRegister=1; //будь-яке ненульове початкове заповнення
ShiftRegister = (((( ShiftRegister>>31)
^( ShiftRegister>>6)
^( ShiftRegister>>4)
^( ShiftRegister>>2)
^( ShiftRegister>>1)
^( ShiftRegister))
&0x00000001)
<<31
| ShiftRegister>>1);
return ShiftRegister & 0x00000001; }
```

Відмітимо, якщо $p(x)$ – примітивний поліном, то $x^n p(1/x)$ – також примітивний. Крім того, якщо поліном $(a,b,0)$ примітивний, то *(a, a – b, 0)* – примітивний. Якщо поліном $(a,b,c,d,0)$ примітивний, то *(a, a – d, a – c, a – b, 0)* – примітивний і тому подібне.

Примітивні тричлени особливо зручні, оскільки додаються лише 2 біти регістра зсуву, але при цьому вони і більш уразливі до атак.

Взагалі кажучи, LFSR зручні для технічної реалізації, але з точки зору криптографічної стійкості мають слабкості.

Послідовні біти лінійної рекуренти лінійно залежні, що робить їх непотрібними для шифрування.

Досить $2n$ послідовних бітів рекуренти, щоб визначити множину номерів точок знімання зворотного зв'язку.

Великі випадкові числа, згенеровані з послідовних бітів LFSR, сильно корельовані. Проте LFSR досить часто використовуються як елементи складних алгоритмів формування шифрувальної ключової послідовності.

Існує ще ряд генераторів ПВП (в т. ч. генератори Галуа), які через ряд причин не знайшли широкого застосування в криптографічних системах. Найбільш ефективні рішення були отримані на основі складених генераторів [11].

Ідея побудови складеного генератора базується на тому факті, що комбінація двох і більше простих генераторів ПВП, в разі правильного вибору об'єднувальної функції (в т. ч. додавання за модулем, $2^{32}-1$ й ін.), дає генератор з покращеними властивостями випадковості і, як наслідок, з підвищеною криптографічною стійкістю.

В разі створення криптографічно стійкого генератора ПВП легко вирішується питання створення потокових шифрів. Вихід таких ПВП не відрізняється (точніше, має не відрізнятися) від РРВП. Два генератори завжди можуть бути синхронно запущені з одного вектора початкового стану, який набагато коротший передаваного повідомлення, що вигідно відрізняє цю схему від шифру Вернама.

Відомо 4 підходи до конструювання відповідних генераторів:

1) системно-теоретичний;

2) складнісно-теоретичний;

3) інформаційно-теоретичний;

4) рандомізований.

Ці підходи розрізняються своїми припущеннями про можливості криптоаналітика, визначення криптографічного успіху і поняття надійності.

В разі системно-теоретичного підходу криптограф створює генератор ключового потоку, який має властивості, що піддаються перевірці, включаючи довжину періоду вихідної послідовності, статистичний розподіл потоку бітів, лінійну складність перетворення і так далі.

З врахуванням відомих методів криптоаналізу криптограф оптимізує генератор проти цих атак.

На основі такого підходу Рюппелем сформульований такий набір критеріїв для потокових шифрів.

1. Великий період вихідної послідовності, відсутність повторень.

2. Висока лінійна складність, як характеристика нашого генератора через регістр LFSR мінімальної довжини, який може згенерувати такий же вихід.

3. Неможливість відрізнити від РРВП за допомогою статистичних критеріїв.

4. Перемішування: будь-який біт ключового потоку має бути складним перетворенням всіх або більшості бітів початкового стану (ключа).

5. Розсіювання: надмірність у всіх підструктурах алгоритму роботи генератора повинна розсіюватися.

6. Критерії нелінійності перетворень: відповідно до деякої метрики відстань до лінійних функцій має бути достатньо великою; потрібне лавиноподібне поширення помилок в разі зміни одного біта аргументу й ін.

Практика підтверджує доцільність застосування вказаних критеріїв не лише для аналізу і оцінювання потокових шифрів, створених в рамках системно-теоретичного підходу, але і для будь-яких потокових і блокових шифрів.

Основна проблема подібних криптосистем полягає в тому, що для них важко довести які-небудь факти про їх криптостійкість, оскільки для всіх цих критеріїв не була доведена їх необхідність або достатність.

Потоковий шифр може задовольняти всі ці принципи і все-таки виявитися нестійким, оскільки стійкість відносно заданого набору криптоаналітичних атак нічого не гарантує.

Прикладом вдалої побудови складеного генератора з точки зору підвищення лінійної складності є каскад Голмана (рис. 4.3). Каскад Голмана містить декілька регістрів зсуву LFSR. Перший регістр рухається рівномірно з кроком 1. Зсув кожного подальшого регістра управляється попереднім так, що зміна стану подальшого регістра в такті $t$ відбувається, якщо в такті $t-1$ з попереднього регістра знімається 1. Інакше стан подальшого регістра не змінюється.

Якщо всі LFSR – довжини $l$, то лінійна складність системи з $n$ регістрами дорівнює $l\left(2^l-1\right)2^{n-1}$.

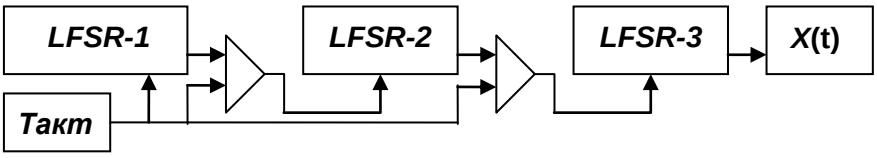



Рисунок 4.3 - Каскад Голлмана

Типовим прикладом комбінування регістрів зсуву є схема «старт-стоп» генератора, що чергується (Alternating Stop-and-Go Generator).

У цього генератора великий період і велика лінійна складність.

У «старт-стоп» генераторі (рис. 4.4) використовуються три лінійні регістри зсуву різної довжини. LFSR-2 змінює стан, якщо вихід LFSR-1 дорівнює 1; LFSR-3 змінює стан в протилежному випадку. Результат генератора є додавання за модулем 2 виходів регістрів LFSR-2, LFSR-3.

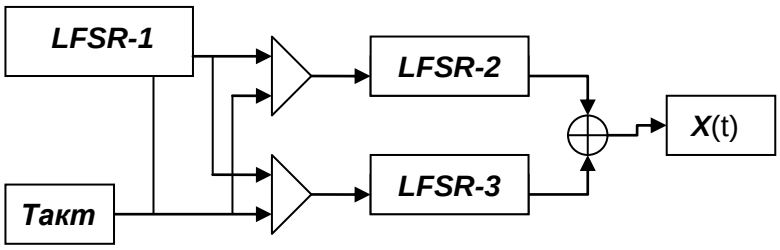



Рисунок 4.4 - Старт-стопний генератор, що чергується

Застосовуючи **складнісно-теоретичний** підхід, криптограф намагається довести стійкість генератора, використовуючи теорію складності.

Основу рішень при цьому підході складають генератори, що базуються на понятті **однонаправленої функції.**

Значення однонаправленої функції $y = f(x): X \to Y$ легко обчислюване, але майже для всіх значень $y$ практично неможливо визначити відповідне значення $x$. Інакше, якщо $V$ – обчислювальна складність отримання $f(x)$, а $V^T$ – обчислювальна складність знаходження $f^{-1}(x)$, то $V^1 >> V$.

На загальну думку, одним з кандидатів на однонаправлену функцію може бути степенева функція в деякому кінцевому полі $f(x) = a^x$, де $a, x \in GF(q)$.

Легко побачити, що піднесення до степеня можна прискорити за рахунок властивостей асоціативності. Наприклад, $a^9 = a\left(\left(a^2\right)^2\right)^2$, що дозволяє обчислити степінь за чотири кроки, замість восьми.

Обернена операція – задача знаходження показника степеня за значенням степеневої функції (дискретний логарифм), в загальному випадку поки не може бути вирішена краще, ніж за допомогою оптимізованих методів перебору.

При відповідно вибраній характеристиці $p > 0$ і степеня розширення поля $n \approx 1024$ це завдання при сучасному розвитку комп'ютерної техніки обчислювально нерозв'язне.

Прикладом генератора на основі однонаправленої функції може служити генератор (ідея криптоалгоритму RSA з параметрами $N, e, x_0$ - див. далі) вигляду $x_{i+1} = x_i^e \bmod N \bmod 2$. Тут $N = pq$, де $p, q$ – секретні великі, нерівні прості числа, $e$ – показник степеневої функції, НСД $(e, (p-1)(q-1)) = 1$, $e \approx 0.5(p-1)(q-1)$.

Результат роботи одного такту генератора – молодший біт $x_{i+1}$. Стійкість цього генератора не нижча за стійкість алгоритму RSA. Якщо $N$ достатньо велике, то генератор забезпечує практичну стійкість.

BBS – інший приклад генератора, побудованого на складнісному підході (запропонований Blum, Blum і Shub).

Це один з простих і ефективних алгоритмів. Математична теорія цього генератора – квадратичні лишки за складеним модулем $n$.

Параметри генератора: секретні великі, нерівні прості числа, $p, q$, такі, що $p = q = 3 (\mathrm{mod}\, 4)$; число $N = pq$; $x$ – випадкове секретне обмеження за модулем $N$.

Першим кроком обчислюється початковий стан $x_0 = x^2 (\mathrm{mod}\, N)$.

В основному циклі елемент ПВП з номером $i > 1$ рівний $x_i = \left(x_{i-1}^2 (\mathrm{mod}\, N)\right) (\mathrm{mod}\, 2)$, тобто $i$-им псевдовипадковим числом є молодший біт числа $x_i = x_{i-1}^2 \bmod N$.

Відмітимо, що алгоритм можна використовувати для шифрування файлів з довільним доступом, якщо, окрім $x$, ввести секретний параметр $\varphi(n) = (p-1)(q-1)$, оскільки тоді $x_t$ можна обчислювати через $x_0$, тому що $x_t = \left(x_0^{h(t)} (\mathrm{mod}\, N)\right) (\mathrm{mod}\, 2)$, де $h(t) = 2^t (\mathrm{mod}\, \varphi(N))$.

Ця властивість дозволяє використовувати BBS-генератор для роботи з файлами довільного доступу (random-access).

Число $n$ можна поширювати вільно для того, щоб кожен абонент мережі зміг самостійно згенерувати необхідні біти. При цьому, якщо криптоаналітик не зможе розкласти на прості множники число $n$, він не зможе передбачити наступний біт, навіть в імовірнісному сенсі, наприклад, «з ймовірністю 51% наступний біт дорівнює 1».

Відзначимо, що подібні генератори дуже повільні, для їх практичної реалізації необхідні спеціальні процесори.

Наступні два підходи, **інформаційно-теоретичний** і **рандомізований**, не знайшли широкого практичного вживання.

З точки зору **інформаційно-теоретичного** підходу найкращим засобом в боротьбі з криптоаналітиком, що має нескінченні обчислювальні ресурси і час, є одноразова стрічка або одноразовий блокнот.

У випадку **рандомізованого** підходу завдання полягає в тому, щоб збільшити число бітів, з якими необхідно працювати криптоаналітику (не збільшуючи при цьому ключ). Цього можна досягти шляхом використання великих випадкових загальнодоступних рядків.

Ключ позначатиме, які частини (або біти) цих рядків необхідно використовувати для зашифрування і розшифрування. Тоді

криптоаналітику доведеться використовувати метод тотального перебору варіантів (грубої сили) на випадкових рядках.

Стійкість цього методу може бути виражена в термінах середнього числа бітів, які доведеться вивчити криптоаналітику, перш ніж шанси визначити ключ стануть вищі за просте вгадування.

Улі Маурер описав таку схему. Ймовірність розкриття такої криптосистеми залежить від об'єму пам'яті, доступного криптоаналітику (але не залежить від його обчислювальних ресурсів).

Щоб ця схема набула практичного вигляду, потрібно близько 100 бітових послідовностей по $10^{20}$ бітів кожна. Оцифрування поверхні Місяця – один із способів здобуття такої кількості бітів.

На закінчення відзначимо, що для побудови генератора ПВП необхідно отримати **декілька випадкових бітів**. Найбільш простий спосіб: використовувати найменший значимий біт таймера комп'ютера.

За допомогою такого способу не можна отримувати багато бітів, оскільки кожен виклик процедури генерації біта може займати парне число кроків таймера, що обов'язково позначиться на властивостях послідовності.

Найкращий спосіб отримати випадкове число – це звернутися до природної випадковості реального світу – шуми в результаті перехідних процесів в напівпровідникових діодах, теплові шуми високоомних резисторів, радіоактивний розпад і так далі.

В принципі, елемент випадковості є і в комп'ютерах:

– час дня;

– завантаженість процесора;

– час прибуття мережевих пакетів і тому подібне.

Проблема не в тому, щоб знайти джерела випадковості, а в тому, щоб зберегти випадковість при вимірюваннях.

Наприклад, це можна робити так: знайдемо подію, що трапляється регулярно, але випадково (шум перевищує деякий поріг).

Виміряємо час $t_{1,2}$ між першою подією і другою, потім – час $t_{2,3}$ між другою подією і третьою.

Якщо $t_{1,2} > t_{2,3}$, то вважаємо вихід генератора рівним 1; якщо $t_{2,3} \le t_{1,2}$, то вихід дорівнює 0. При необхідності процес продовжимо далі.

Суттєвою проблемою систем генерації випадкових даних є наявність відхилень і кореляцій в згенерованій послідовності. Самі процеси можуть бути випадковими, але проблеми можуть виникнути в процесі вимірювання. Як з цим боротися?

Нехай ймовірність появи нуля зміщена на $\varepsilon$, тобто може бути записана як $P(0) = 0.5 + \varepsilon$.

Додавання за mod 2 двох однаково розподілених незалежних бітів дає вираз для ймовірності: $P(0) = (0.5+\varepsilon)^2 + (0.5-\varepsilon)^2 = 0.5 + 2\varepsilon^2$. При додаванні чотирьох бітів отримаємо: $P(0) = 0.5 + 8\varepsilon^4$. Процес поступово сходиться до рівноймовірного розподілу бітів.

Інший підхід. Нехай розподіл одиниць і нулів в послідовності є величини $p$ і $q$ відповідно.

Перетворимо послідовні пари бітів:

– якщо це однакові біти, то відкинемо їх і розглянемо наступну пару;

– якщо біти різні, то як вихідне значення візьмемо перший біт.

Даний метод дозволяє вирішити проблему зсуву, зберігши властивості випадковості джерела (з деякою втратою в об'ємі даних).

Потенційна проблема обох методів в тому, що за наявності кореляції між сусідніми бітами дані методи збільшують зсув. Один із способів уникнути цього – використовувати різні джерела випадкових чисел і підсумовувати біти підписаних одна під одною послідовностей по вертикалі.

Факт наявності зсуву в генераторі випадкових чисел, загалом, не завжди означає його непридатність.

Наприклад, припустимо, що для генерації 112-бітового ключа для алгоритму «потрійний» DES (Triple DES, див. далі) використовується генератор із зсувом до нуля: $P(x_t = 0) = 0.55$, $P(x_t = 1) = 0.45$ (ентропія $H = 0.99277$ на один біт ключа в порівнянні з 1 для ідеального генератора).

В цьому випадку зловмисник може оптимізувати процедуру тотального перебору ключів за рахунок пошуку ключа, починаючи з найбільш ймовірного значення $(00\ldots0)$ і закінчуючи найменш ймовірним $(11\ldots1)$. Унаслідок наявності зсуву можна чекати знаходження ключа в середньому за $2^{109}$ спроб. Якби зсуву не було, то було б потрібно $2^{111}$ спроб.

**Питання до розділу 4**

1. У чому полягає властивість непередбачуваності рівномірно розподіленої випадкової послідовності?
2. Яким чином визначається функція автокореляції періодичної послідовності?
3. Яка послідовність називається випадковою за Шенноном?
4. Яка послідовність називається випадковою за Колмогоровим?
5. Яка послідовність називається такою, що поліноміально не відрізняється від випадкової?
6. Яка послідовність називається псевдовипадковою у вузькому сенсі?
7. У чому полягає гіпотеза $H_0$ при перевірці випадковості послідовності на основі частотних тестів?
8. Яким чином визначаються помилки 1-го і 2-го роду для статистичних тестів?
9. Яка величина називається пороговим значенням статистики для даного тесту?
10. Що таке загальна помилка першого роду для набору тестів?
11. Чому для набору тестів необхідна перевірка їх незалежності?
12. Як влаштований лінійний конгруентний генератор РРВП?
13. Як функціонує регістр зсуву з лінійним зворотним зв'язком?
14. Що таке лінійна складність послідовності?
15. Які критерії якості для вихідних послідовностей криптографічних генераторів висунуті Рюппелем?
16. В чому полягає ідея конструювання генераторів РРВП на основі регістрів зсуву з рухом, яким можна керувати?
17. Які однонаправлені функції використовуються в генераторах РРВП BBS і RSA?

# РОЗДІЛ 5 ПРИНЦИПИ ПОБУДОВИ БЛОКОВИХ ШИФРІВ НА ПРИКЛАДІ АЛГОРИТМУ DES

## 5.1 Криптосхема алгоритму DES

У 1972 році Національним бюро стандартів США (нині підрозділ Департаменту торгівлі США NIST - National Institute of Standards and Technology), з урахуванням зростаючих потреб забезпечення безпеки інформації, була розроблена програма захисту даних.

Як складову частину програма передбачала створення єдиного алгоритму для безпечної передачі та зберігання інформації.

Уніфікація алгоритму мала на меті економію витрат на створення надійних систем захисту. Остаточна версія алгоритму DES була прийнята в листопаді 1976.

Вивчення цього алгоритму цікаве з точки зору з'ясування базисних понять методології підвищення безпеки інформації, оцінювання її вразливості і ефективності застосування різних криптоаналітичних атак.

До алгоритму шифрування висувалися такі основні вимоги:

- високий рівень безпеки;
- наявність повної специфікації алгоритму і простота розуміння;
- безпека алгоритму повинна базуватися на секретності ключа;
- широка доступність алгоритму;
- простота адаптації алгоритму для використання в різних прикладних програмах;
- економічність у реалізації алгоритму;
- ефективність алгоритму;
- можливість верифікації алгоритму;
- відсутність експортних обмежень.

Алгоритм DES [1,11] відносять до класу блокових шифрів, для яких вихідні дані та результати шифрування подаються у вигляді 64-бітових рядків - блоків.

Шифрування і розшифрування в алгоритмі здійснюються за однаковими схемами за винятком процедури формування ключів, що використовуються в циклах його роботи.

Ключем шифрування є випадкове число довжиною 56 бітів, що доповнюється для підвищення надійності зберігання та транспортування вісьмома бітами перевірки парності, так що загальна довжина ключа дорівнює 64 бітам.

Алгоритм реалізує так звану схему Файстеля, що передбачає ітераційну обробку двох частин блоку вихідних даних.

При цьому в кожному циклі (раунді - в оригінальній документації) роботи алгоритму з використанням підстановок і перестановки реалізуються функції розсіювання (дифузії) і перемішування. Всього в алгоритмі виконується 16 раундів (рис. 5.1).

Перед шифруванням вихідний 64-бітовий ключ звужується до 56 бітів (забираються біти парності) і розбивається на два півблоки по 28 бітів кожен.

Після початкової перестановки (initial permutation - $IP$) блок вихідних даних розбивається на ліву $L_0$ і праву $R_0$ половини (півблоки) по 32 біти кожна.

Потім відбувається 16 раундів ідентичних операцій, входом і виходом яких є пари півблоків $L_j$, $R_j$, $j = 0,1,\ldots,15$.

На кожній ітерації півблоки комбінуються з ключем $K_{j+1}$ відповідного раунду (нумерація ключів, прийнята в стандарті Fips 46 для раундів і ключів, починається з одиниці, а півблоків - з нуля).

Після останнього раунду правий і лівий півблоки об'єднуються і фінальна перестановка $IP^{-1}$, що є інверсією початкової перестановки, завершує алгоритм.

Перестановки $IP$ і $IP^{-1}$ не впливають на стійкість DES, тому деякі, реалізовані у вигляді програм його версії, не виконують через повільне виконання бітових операцій.

Для отримання чергового раундового ключа половини попереднього ключа незалежно одна від одної циклічно зсуваються. Величина і напрямок зсуву для кожного раунду фіксовані (задані в описі криптоалгоритму).

Потім зсунуті півблоки ключа об'єднуються в 56-бітову послідовність, з якої вибираються 48 бітів за допомогою постійної перестановки стискання (compression permutation або permuted choice).

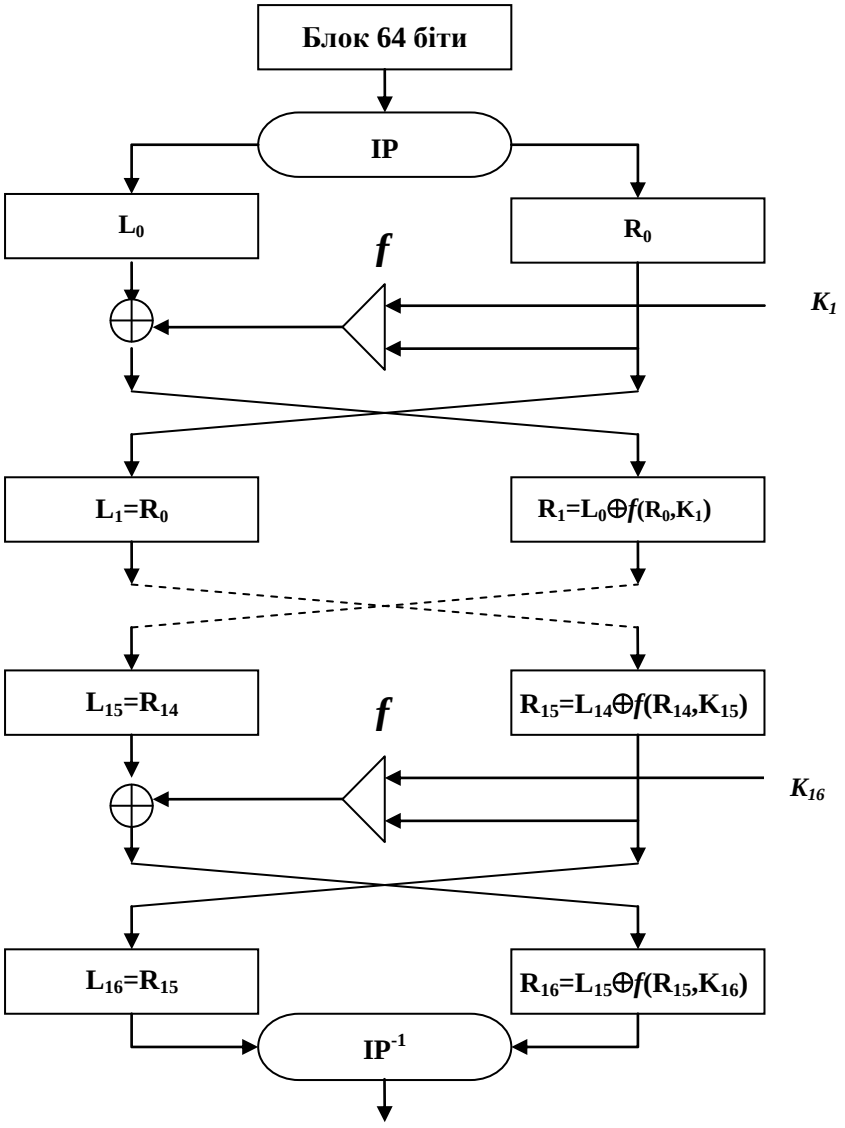



Рисунок 5.1 - Загальна схема алгоритму DES

У кожному раунді правий півблок даних $R_{j-1}$, $j=\overline{1,15}$ розширюється до 48 бітів з використанням перестановки розширення $E$ (за рахунок повторень бітів), потім порозрядно додається за модулем 2 з 48 бітами раундового ключа (рис. 5.2).

Отримані 28 бітів замінюються на блок з 32 бітів за допомогою восьми $S$-блоків (таблиць заміни, що перетворюють 6 бітів в чотири).

Отримані 32 біти переставляються за фіксованою підстановкою, в результаті чого виходить правий півблок $R_j$ нової пари півблоків.

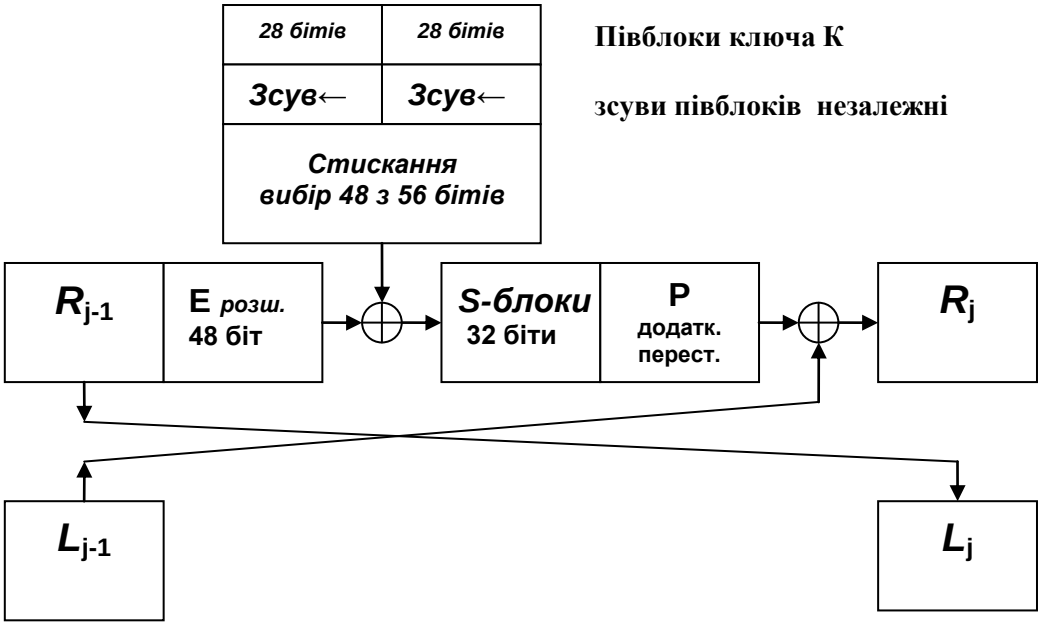
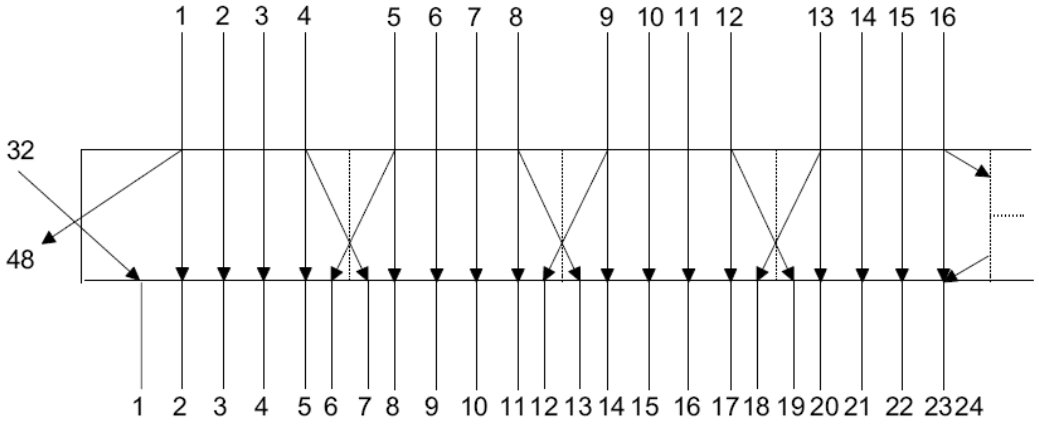



Рисунок 5.2 - Схема перетворень одного раунду

За $L_j$ приймається блок $R_{j-1}$. Таким чином, перетворення півблоків має вигляд $L_j = R_{j-1},\ R_j = L_{j-1} \oplus f\left(R_{j-1}, K_j\right)$.

Відзначимо особливу роль перестановки розширення $E$ (expansion permutation), з застосуванням якої правий півблок $R_j$ розширюється з 32 до 48 бітів одночасно з їх перестановкою (рис. 5.3).




| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **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** |

Рисунок 5.3 - Перестановка розширення $E$

Не менш важлива роль в алгоритмі відведена **восьми $S$-блокам** ($S$-box substitution), на вхід яких подається 48 бітів результату додавання раундового ключа і розширеного півблоку тексту (рис. 5.4), а на виході отримуємо 32-бітовий рядок.

Кожна підстановка наведена таблицею з 4 рядків і 16 колонок. Блок у 48 бітів ділиться на шестибітові комбінації $(b_1, b_2, \ldots, b_6)$ так, що перша комбінація замінюється при 1-ій підстановці і т.д., восьма комбінація - при 8-ій підстановці.

У кожній комбінації двобітове число $(b_1, b_2)$ позначає рядок таблиці заміни (від 0 до 3). Аналогічно, чотири біти, що залишилися, визначають номер стовпця. За рядком і стовпцем знаходиться тетрада бітів, яка замінює вихідну комбінацію, в результаті отримаємо вісім тетрад, об'єднаних в 32-бітовий блок. $S$-блоки (вузли заміни) забезпечують нелінійність і стійкість перетворення (інші операції - лінійні і легко піддаються криптоаналізу).

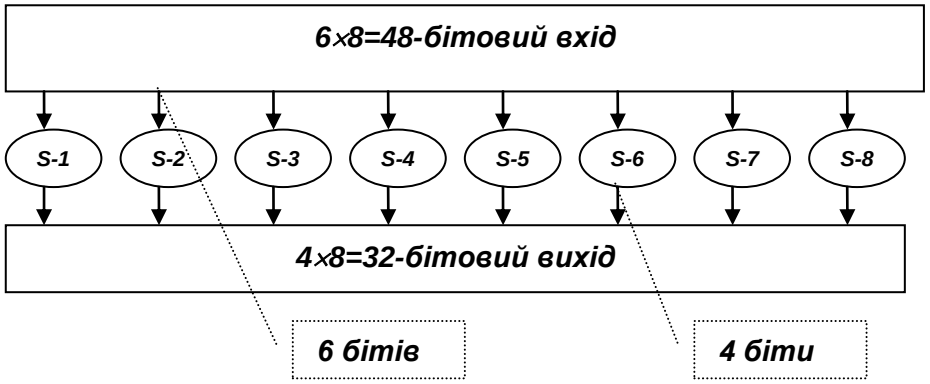



Рисунок 5.4 - Схема реалізації замін (S-блоків)

Результат застосування підстановок знову перемішується (переставляється) за допомогою постійної перестановки $P$. Операції, починаючи з $E$ і закінчуючи $P$, утворюють раундову (циклову) функцію $f$ (див. рис. 5.1, 5.2).

Розшифрування повідомлень здійснюється аналогічно викладеній вище схемі з невеликою зміною. Єдина різниця в тому, що ключі треба використовувати у зворотному порядку: спочатку $K_{16}$, потім $K_{15}$ і т. д. Алгоритм генерації ключів для раунду також обернений.

Спочатку, коли були невідомі принципи формування вузлів заміни, Агентство Національної Безпеки (АНБ) США звинувачували у вбудовуванні прихованих лазівок в їх алгоритм.

У подальшому, на 2-му семінарі, присвяченому DES, АНБ розкрило кілька критеріїв побудови блоків заміни.

1. Жоден $S$-блок не є лінійною (афінною) функцією вхідних даних, тобто не існує системи лінійних рівнянь, якими можна виразити 4 вихідних біти в термінах 6 вхідних.

2. Зміна одного біта на вході змінює, як мінімум, два біти на виході, отже, досягається максимізація дифузії.

3. S-блоки повинні мінімізувати різницю між числом нулів та одиниць.

До числа цікавих результатів дослідження алгоритму слід віднести той факт, що два різних спеціально підібраних вхідних блоки можуть давати однаковий вихід. Можна також досягти рівності входу і виходу зміною тільки в трьох сусідніх S-блоках по одному біту. Тому ще один критерій АНБ полягав у стійкості до методів так званого диференціального криптоаналізу.

## 5.2 Криптографічні властивості алгоритму DES

Найбільш слабкою ланкою алгоритму є невелика довжина ключа.

Відзначимо, що в алгоритмі Lucifer - прототипі DES, який був розроблений компанією IBM, довжина ключа становила 128 бітів.

Короткий ключ збільшує ймовірність реалізації ризику атаки методом повного перебору ключів за допомогою швидкісних обчислювальних засобів.

Ще в 1979 році У. Діффі і Д. Хеллман (W. Diffe, D. Hellman) дали оцінку вартості спеціального обчислювача для розкриття ключа за одну добу: $\approx$ 20 мільйонів доларів. У 1981 році цей результат був уточнений: вартість обчислювача, що вирішує завдання за два дні, не перевищує $ 50 млн. Для сучасних спеціальних служб перебір ключів DES цілком реальний.

Критичним для алгоритму є кількість раундів його роботи. Чому використовується саме 16 раундів, а не 8 чи 32? А. Конхейм (A. Konheim)

показав, що після 8 ітерацій шифрований блок є практично випадковою функцією будь-якого біта тексту і будь-якого біта ключа. Тим не менше, обмежитися 8 раундами не є можливим, виходячи з ряду практичних і теоретичних результатів. Зокрема, в 1982 році були отримані результати криптоаналізу 4-раундового DES; через шість років був дешифрований 6-раундовий DES.

Розроблені в 1990 році Е. Біхамом (E. Biham) і А. Шаміром (A. Shamir) методи диференціального криптоаналізу показали, що DES з меншою ніж 16 кількістю раундів може бути розкритий за допомогою атаки на основі відомих відкритих текстів (known-plaintext attack ) більш ефективно, ніж на основі повного перебору ключів.

Стійкість алгоритму DES істотно залежить від використовуваних ключів [11,12]. Схема формування ключів для раундів допускає деякі ключі, які називаються слабкими.

Як вже говорилося, ключ ділиться на дві половини і кожна з них зсувається незалежно. У зв'язку з цим може вийти так, що кожен раундовий ключ буде однаковим.

Це може статися, якщо ключ складається тільки з 0, тільки з 1 або, якщо одна половина з 0, а інша з 1 (таблиця 5.1).

Таблиця 5.1 - Значення слабких ключів алгоритму DES (у 16 с / с)

| **Ключ 56 бітів** | |
|---|---|
| **28 бітів(лів.)** | **28 бітів(прав.)** |
| **0000000** | **0000000** |
| **0000000** | **FFFFFFF** |
| **FFFFFFF** | **0000000** |
| **FFFFFFF** | **FFFFFFF** |

Крім того, існують пари ключів, що дають однаковий результат для шифрування відкритих текстів (еквівалентні ключі). Це теж пов'язано з генерацією підключів: замість 16, насправді, формуються тільки 2 різних, використаних за 8 раз в 16-ти раундах. Ці ключі називаються напівслабкими. Є також ключі, які виробляють тільки 4 підключі, кожен з яких використовується по 4 рази на алгоритмі шифрування.

Загалом, 64 слабких ключі із 72 057 594 037 927 936 варіантів можливих ключів - це небагато. Шанси отримати такий ключ мізерно малі. Інших слабких ключів не знайдено.

Певні слабкості DES пов'язані з властивостями так званих доповнювальних (інверсних) ключів.

Нехай блок тексту $T$ переходить в блок шифртексту $X$ при ключі $K$.

Позначимо через $\tilde{K}$ побітове доповнення (інверсію) ключа $K$: $K \oplus \tilde{K} = (11\ldots1)$. Виявляється, що результат зашифрування блоку $\tilde{T}$ на ключі $\tilde{K}$ рівний $\tilde{X}$.

Це означає, що застосування проти DES атаки із заздалегідь обраним відкритим текстом (chosen-plaintext attack) дозволяє скоротити вдвічі перебір варіантів ключів, тобто потрібно перевірити $2^{55}$ варіантів замість $2^{56}$.

Крім того, Е. Біхам і А. Шамір показали також, що існує атака на основі відомих відкритих текстів (known-plaintext attack) такої ж складності за наявності як мінімум $2^{33}$ відомих пар відкритих і шифрованих текстів.

Вказана слабкість є скоріше теоретичною, тому що ймовірність появи у відкритого повідомлення доповнення у вигляді відкритого тексту дуже низька.

Аж до 1992 року залишалося дискусійним питання: чи є множина відображень, породжена алгоритмом DES, групою? Суть проблеми полягає ось в чому.

Кількість всіх взаємно однозначних відображень 64-бітових блоків на себе складає $2^{64}!$ DES з 56-бітовим ключем дає не більше $2^{56} \approx 10^{16}$ шифрувальних відображень.

Кожне таке відображення визначається ключем, є підстановкою на множині 64-бітових блоків і визначає операції зашифрування-розшифрування.

Якщо множина $\Omega$ всіх операцій DES (тобто, операцій зашифрування і розшифрування) не замкнута відносно їх добутку, то, комбінуючи ці операції, можна одержувати велику кількість нових шифрперетворень зі збільшеною довжиною ключа (рис. 5.5).

Це справедливо лише в тому випадку, якщо множина $\Omega$ не є групою (підгрупою групи підстановок).

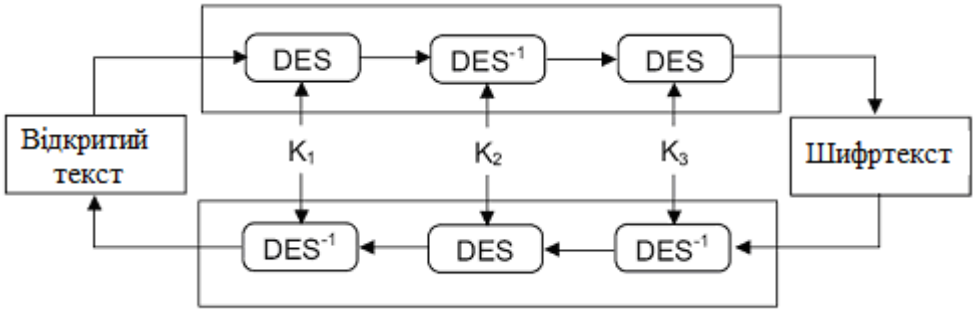



Рисунок 5.5 - Потрійне шифрування за допомогою DES

Якщо ж $\Omega$ є групою, то криптоаналіз алгоритму спрощується, бо тоді для будь-яких ключів $K_1$, $K_2$ існує ключ $K_3$ такий, що $E_{K_2}\left(E_{K_1}(T)\right)=E_{K_3}(T)$, а це дозволяє скористатися атакою типу «зустріч посередині» (meet-in-the-middle) зі складністю $2^{28}$ варіантів замість $2^{56}$ у разі повного перебору ключів.

Тільки в 1992 році було доведено, що множина перетворень, які породжуються DES, не є групою.

Як вже було зазначено вище, дослідження блокових алгоритмів, включаючи DES, стало поштовхом для розробки методів диференціального криптоаналізу.

Ці методи засновані на аналізі пар шифрованих текстів, відповідні пари відкритих текстів яких відрізняються однаковим, заздалегідь встановленим чином.

У цьому випадку прийнято говорити, що для аналізу вибираються пари відкритих текстів з однаковими різницями.

Вихідні тексти не обов'язково повинні бути осмисленими, вони можуть вибиратися випадково, аби вони відповідали певним умовам.

Деякі відмінності у відкритих текстах з досить високою ймовірністю можуть (можливо, побічно) проявитися і в шифрованих блоках. Такі відмінності називаються характеристиками.

Для алгоритму DES, наприклад, якщо різниця між блоками відкритих текстів (щодо операції $XOR$) дорівнює $0080\ 8200\ 6000\ 0000_{16}$, то існує шанс, приблизно в 5%, що різниця між шифрованими блоками така ж.

Диференціальний криптоаналіз використовує подібні характеристики (їх багато) для переоцінки ймовірностей окремих ключів і,

згодом, для пошуку найбільш ймовірних значень розрядів істинного ключа.

Відповідна атака може бути проведена як проти DES, так і інших блокових алгоритмів, що мають постійні блоки заміни. Ефективність атаки істотно залежить від структури S-блоків.

У стандарті алгоритму DES блоки заміни оптимізовані проти таких атак. Зокрема, при заданих блоках, для 16 раундів оцінка складності складає близько $2^{42}$. Для алгоритму з 17-18 раундами час розкриття ключа виявляється приблизно рівним часу повного перебору. При 19 раундах перебір стає вигіднішим.

У цілому, ризик успішного проведення відповідних атак, з огляду на колосальні ресурси, необхідні для підготовки атаки, мінімальний.

Для проведення атаки chosen plaintexts attack для отримання необхідної інформації потрібне шифрування потоку 1.5 Мбіт/сек. протягом 3 років.

У більшості раундів вихідний ключ DES зсувається на 2 біти, крім 1, 2, 9 і 16 раундів, коли зсув дорівнює 1 біту. Причина - поява регулярності в перетворенні, яка може бути використана для проведення криптоаналітичних атак.

Модифікація DES, шляхом запровадження постійного зсуву ключа на 2 біти після будь-якого раунду, призводить до зниження його стійкості.

Е. Біхаму належить ідея відповідної ефективної **атаки на основі зв'язаних ключів** (related-key criptanalysis). Ця атака реалізується у випадку постійного зсуву ключа і при наявності $2^{17}$ відкритих текстів з вибраними ключами (chosen-key plaintexts) або у випадку відомих пар відкритих і шифрованих текстів (known-plaintexts).

Для проведення атаки проводиться аналіз різниць між парами ключів і парами відкритих і шифрованих текстів.

Атака не залежить від числа раундів, проте її слід визнати малореальною загрозою, оскільки для її реалізації криптоаналітик повинен знати властивості датчика випадкових чисел, який використовується для створення ключів DES, а також мати доступ до шифратори для отримання величезної кількості пар відкритих і шифрованих текстів. Можна вважати DES оптимізованим і проти цієї атаки.

## 5.3 Режими роботи блокових алгоритмів

Існує кілька режимів роботи блокових алгоритмів, які використовуються в залежності від конкретного застосування [11,12].

Режим електронної кодової книги ECB (Electronic Codebook Mode) - найбільш очевидний спосіб використання блокового криптоалгоритму.

Суть режиму полягає в послідовній заміні блоків вихідного тексту на відповідні шифровані блоки. Однакові блоки завжди зашифровуються однаково, що аналогічно шифруванню за допомогою звичайної кодової книги, звідси - назва електронна кодова книга (Electronic Codebook).

Зазвичай, розмір блоку дорівнює 64 бітам, отже, в «кодовій книзі» досить багато записів - $2^{64}$. Структура кожної «кодової книги» визначається своїм ключем.

Перевагою ECB є незалежне шифрування блоків. Це важливо для шифрування сукупності файлів з довільним (випадковим) доступом, наприклад, бази даних. Шифровані записи можна видаляти і додавати в будь-яку довільну ділянку БД, якщо вважати, що запис складається з цілого числа блоків.

Проблемою ECB є те, що криптоаналітик може почати збирати кодову книгу без знання ключа за наявності пар відкритих і шифрованих текстів. Практично в реальних повідомленнях зустрічаються повторювані структури: початок та завершення листів електронної пошти, заголовки, звернення, підписи, інформація про відправника, дата листа і т. д. Для руйнування цих закономірностей використовують інші режими.

Не менш, а потенційно може й більш, небезпечний ризик полягає в можливості імітації або відтворення блоків (block replay) без знання ключа і навіть деталей алгоритму. По суті, зловмисник може змінювати зашифровані повідомлення так, щоб обдурити приймача.

Зокрема, розглянемо систему електронних банківських розрахунків, у якій повідомлення шифруються блоковим алгоритмом в режимі ECB. Стандартний формат повідомлень може мати вигляд 13 блоків по 64 біти.

Нехай зловмисник контролює лінію зв'язку між двома банками. Він посилає певну суму (наприклад, $ 100) з одного банку в інший, потім ще раз. Далі, переглянувши всі повідомлення, він знаходить пару повідомлень, які збігаються. Якщо таких багато, то операцію потрібно повторити для виявлення «свого» переказу.

| 1 | 2 | 3 | 4 | 5-10 | 11 | 12 | 13 |
|---|---|---|---|---|---|---|---|
| **Штамп часу** | **Банк - відправник** | | **Банк - одержувач** | **Ім'я вкладника** | **№ рахунку** | | **Сума** |
| **8 байт** | **12 байтів** | | **12 байтів** | **48 байтів** | **16 байтів** | | **8 байтів** |

Рисунок 5.6 - Формат повідомлення про транзакції

Відіславши виявлене повідомлення ще раз, можна отримати на рахунок доходу + $ 100. Якщо зробити це з сумами більше $ 100 і кілька разів, то можна отримати достатній куш.

Ця атака носить назву «повторення блоку» (block replay).

На перший погляд, повинна допомогти виявити подібну атаку відмітка часу передачі повідомлення (timestamp), проте це не завжди допомагає.

Банки можуть зменшити ризик, частіше змінюючи ключі, але це означає тільки те, що зловмисникові необхідно працювати швидше.

Адекватною відповіддю системи захисту на атаки подібного роду є перетворення, зване зчепленням (chaining), якому відповідає так званий режим CBC (Cipher Block Chaining Mode).

### 5.3.1 Режим зчеплення шифрованих блоків (CBC)

Зчеплення забезпечується шляхом застосування результату шифрування попереднього блоку для шифрування поточного блоку. Таким чином, будь-який блок шифртексту залежить від усіх попередніх блоків вихідного тексту.

У цьому режимі блок тексту, перед шифруванням за допомогою алгоритму $E(\cdot)$, додається за модулем 2 з попереднім шифрованим блоком чи вектором ініціалізації $IV$ (рис. 5.7).

Розшифрування виконується аналогічно. Математичний запис рівнянь зашифрування і розшифрування виглядає так:

$$S_0 = IV,\ S_i = E_k(T_i + S_{i-1}), T_i = S_{i-1} \oplus D_k(S_i).$$

Вектор ініціалізації $IV = S_0$ повинен бути випадковим, щоб кожне повідомлення було унікальним з метою уникнення можливості атаки на

стандартний заголовок повідомлення. Вектор IV не обов'язково зберігати в секреті, його можна передавати разом з повідомленням.

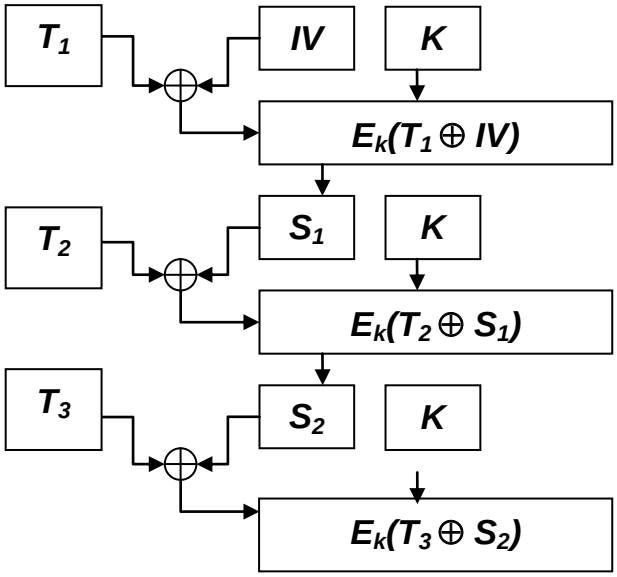



Рисунок 5.7 - Режим шифрування CBC для трьох символів тексту

Більшість повідомлень не ділиться без остачі на 64-бітові блоки, зазвичай залишається короткий блок в кінці тексту. Можна по-різному боротися з цим.

Найпростіший метод - це доповнення до повного блоку (padding) несмисловим набором символів. Для видалення «сміття» при відновленні повідомлення достатньо в останньому байті вказати кількість зайвих символів. Також з цією метою можна використовувати як останній байт тексту символ кінця файлу, але іноді це неможливо реалізувати, наприклад, у випадку, коли розшифрований блок негайно відправляється в пам’ять.

Тоді застосовується така схема шифрування: припустимо, останній блок складається з $j$ бітів. Після шифрування останнього цілого блоку, зашифруємо шифрований блок ще раз, виберемо $j$ початкових бітів отриманого тексту і додамо за mod 2 з останнім блоком вихідного тексту.

Це і є шифр для неповного блоку (рис. 5.8).

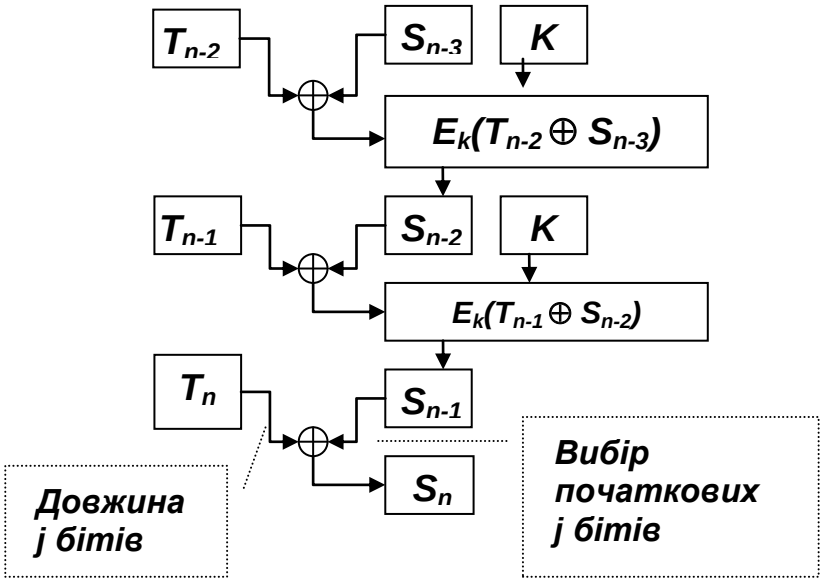



Рисунок 5.8 - Шифрування останнього короткого блоку

### 5.3.2 Відновлення після помилок у CBC

Режим CBC може бути охарактеризований як процедура зі зворотним зв'язком за шифртекстом (feedback) при шифруванні і з попереджувальним зв'язком (feedforward) за шифртекстом при розшифруванні. Це означає, що одна помилка у вихідному тексті вплине на шифрований блок і всі наступні шифровані блоки. Але це неважливо, оскільки розшифрування оберне цей ефект і у відновленому тексті знову буде тільки одна помилка.

Помилки в зашифрованому тексті досить поширені через низьку якість ліній зв'язку або помилок зчитувань з дискети чи жорсткого диска.

У режимі CBC помилка одного біта в шифрованому тексті повністю пошкоджує цей блок і ще один біт наступного блоку (у тій же позиції, що і попередній). Наступні блоки не пошкоджуються, тобто CBC є режимом, що самовідновлюється (self-recovering). Незважаючи на те, що режим CBC відновлюється за наявності спотворень, він зовсім не захищений від помилок синхронізації. Якщо додати або втратити біт, то система буде формувати безглузді повідомлення.

Тому будь-яка система, що використовує режим CBC, повинна забезпечити збереження структури вихідного тексту (framing).

Властивість перетворення маленької помилки в шифрі у велику помилку тексту називається розповсюдженням помилки (error extention).

Для боротьби з таким явищем необхідне застосування механізму виявлення та виправлення помилок (error detection & recovery). Причому, така процедура повинна виконуватися як після зашифрування, так і до розшифрування. Повна схема механізму виявлення та виправлення помилок (error control) наведена на рис. 5.9.

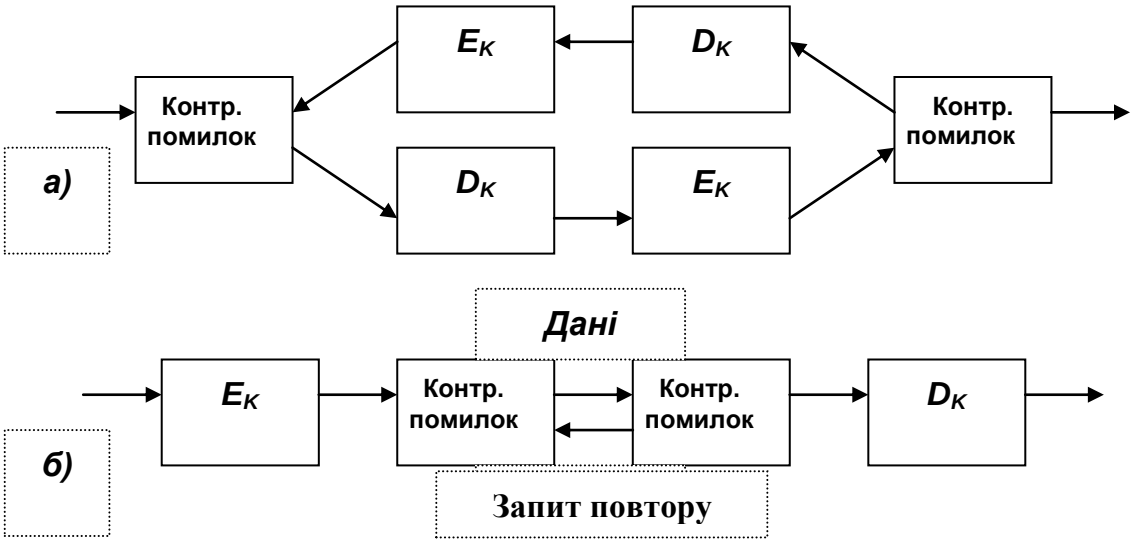



Рисунок 5.9 - Проста (а) і покращена (б) схеми контролю помилок

### 5.3.3 Режим зворотного зв'язку з шифром (CFB)

Однією з проблем використання режиму CBC є неможливість почати шифрування до отримання цілого блоку даних. Ця обставина може ускладнити роботу додатків з віддаленим терміналом.

Режим CFB (Cipher Feedback Mode) передбачає можливість шифрування даних записами, меншими, ніж цілий блок.

Ми розглянемо варіант шифрування одного символу в коді ASCII (так званий 8-бітовий CFB), але його можна узагальнити і на випадок одного біта.

На рис. 5.10 показаний 8-бітовий режим CFB у випадку 64-бітового блокового алгоритму. У цьому варіанті блоковий алгоритм працює з чергою (стеком) з розміром, що дорівнює довжині вхідного блоку - 64 біти.

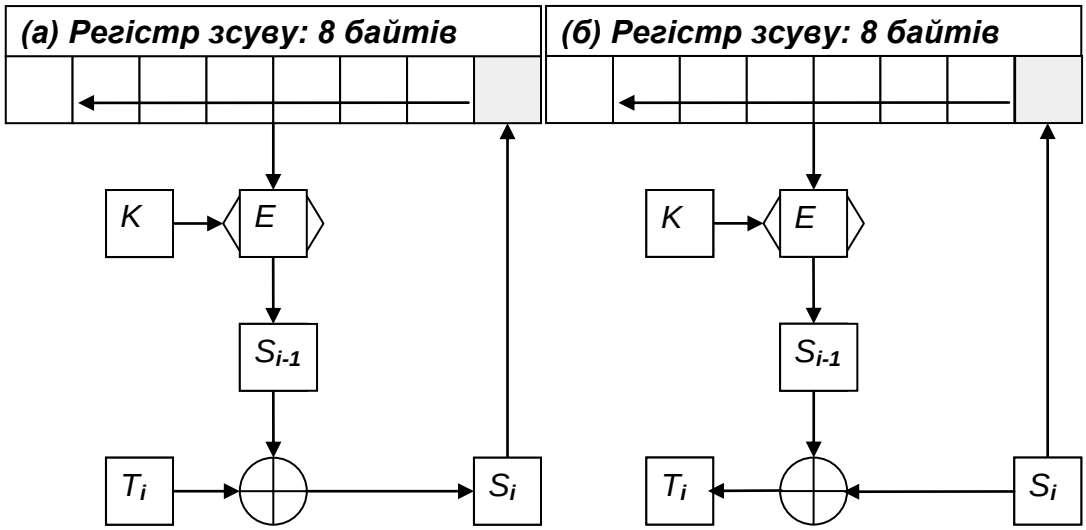



Рисунок 5.10 - Режим 8-бітовий CFB-mode, (а) зашифрування, (б) розшифрування

Перед початком шифрування стек, як і в режимі CBC, заповнюється вектором ініціалізації IV.

У кожному такті шифрується вміст стека, а крайні ліві 8 бітів результату складаються за модулем два з першим 8-бітовим символом тексту, в результаті отримуємо 8 бітів шифрованого тексту, готових для передачі.

Потім вміст стека зсувається вліво на 8 бітів, крайні ліві 8 бітів при цьому відкидаються. Раніше отриманий символ шифрованого тексту заміщає крайні праві 8 бітів стека. Потім процес повторюється. Таким чином, по суті, реалізується генератор потокового шифру.

Розшифрування здійснюється в зворотному порядку. Відзначимо, що на обох сторонах блоковий алгоритм працює в режимі зашифрування.

Як і в CBC, в режимі CFB зашифрований текст залежить від усього попереднього тексту. **У CFB, як і в CBC, помилка у вихідному тексті обертається при розшифруванні.**

Помилка в зашифрованому тексті більш цікава. Перший ефект однобітової помилки - помилка в початковому тексті.

При розшифруванні помилковий біт потрапляє на регістр зсуву і псує результат розшифрування до тих пір, поки не вийде з іншого кінця регістра.

Таким чином, у 8-бітовому CFB помилка одного біта в шифртексті призводить до спотворення при розшифрування дев'яти байтів, хоча подальший шифртекст розшифровується коректно.

Нюанс: якщо атакуюча сторона знає текст повідомлення, то, можливо, навмисне спотворення бітів в даному блоці таке, що блок розшифрується в усе, що завгодно. Наступний блок буде розшифровано у сміття, але шкода вже може бути заподіяна.

Таким чином, **режим CFB є таким, що самовідновлюється** (self-recovering) і щодо помилок синхронізації. Помилка потрапляє на регістр, псує 8 байтів і випадає з іншого кінця (оскільки синхронізація завжди починається від останнього зашифрованого байта).

### 5.3.4 Режим зворотного зв'язку за виходом (OFB)

Режим OFB (Output Feedback Mode, рис. 5.11) схожий на CFB за винятком того, що в регістр зсуву поміщається $S_{i-1}$, а не $S_i$.

Тому цей режим ще називається внутрішнім зворотним зв'язком (internal feedback), оскільки механізм зворотного зв'язку не залежить від початкового тексту та зашифрованого тексту.

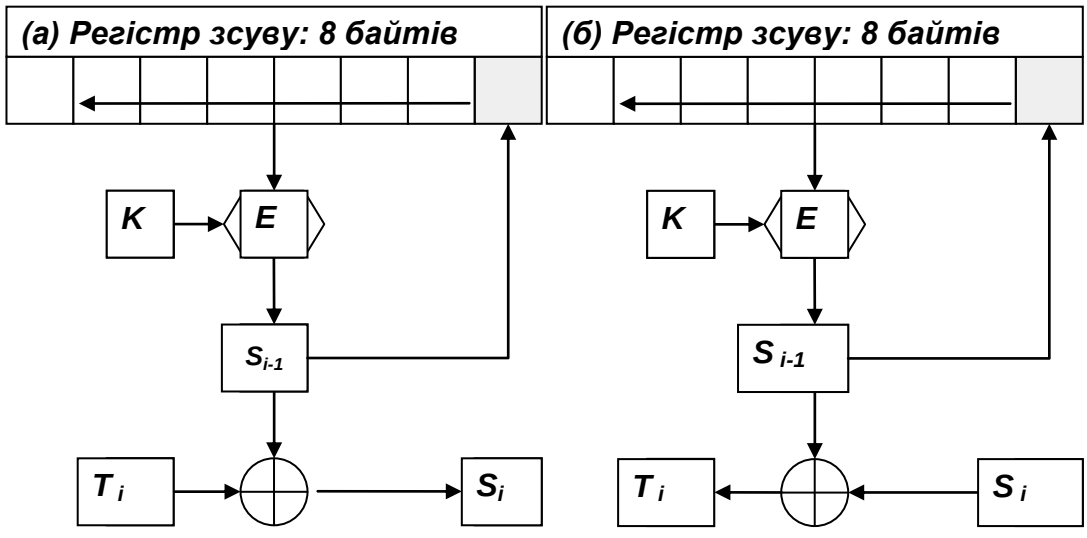



Рисунок 5.11 - Режим OFB, (а) зашифрування, (б) розшифрування

**Головною перевагою режиму OFB є нерозповсюдження помилок** або відсутність явища error extension. Помилка одного біта в шифрованому блоці викликає однобітову помилку при відновленні тексту.

Це важлива якість для передачі оцифрованих відео- або аудіофайлів, де припустимі поодинокі помилки, а ефект поширення помилок - ні.

З іншого боку, втрата синхронізації буде мати неусувні наслідки. Тому шифрсистема в режимі OFB повинна мати механізм виявлення втрати синхронізації і механізм завантаження регістрів зсуву новим вектором ініціалізації $IV$ для відновлення синхронізації.

Проведені в останні роки дослідження режиму OFB довели, що цей режим повинен бути використаний тільки в тому випадку, коли розмір блоку рівний розміру заміщуваного елемента за зворотним зв'язком (feedback), тобто 64-бітовий блоковий алгоритм повинен бути використаний з 64-бітовим OFB. Вибір відповідного режиму використання блокового алгоритму слід робити виходячи з таких міркувань.

ECB - найпростіший і найслабший. CBC трохи складніший, але набагато надійніший. CFB і OFB сильніші, але повільніші; CFB зазвичай використовується для віддалених терміналів, OFB - для високошвидкісних синхронних систем передачі. DES дозволено використовувати з будь-яким з цих чотирьох режимів. Інші режими застосовувати не рекомендується.

**Питання до розділу 5**

1. Який розмір блоку, довжина ключа і кількість раундів в DES?
2. Який порядок формування підключів в DES?
3. Яким чином перетворюються дані на одному раунді DES?
4. Яка роль і які параметри перестановки розширення?
5. Яка структура вузлів заміни і як з їх допомогою здійснюється перетворення даних?
6. Яка найслабша ланка алгоритму DES?
7. У чому полягає властивість незамкнутості множини відображень, що породжуються DES?
8. Яка середня оцінка кількості пар вхідних блоків, необхідних для дешифрування DES методом диференціального криптоаналізу?
9. У чому полягає режим ECB блокового шифру?
10. У чому полягає режим CBC блокового шифру?
11. У чому полягає режим CFB блокового шифру?
12. У яких випадках рекомендується використовувати режими CFB і OFB?

# РОЗДІЛ 6 БЛОКОВІ ШИФРИ ГОСТ 28147-89 І RIJNDAEL

## 6.1 Схеми шифрування ГОСТ 28147-89 і Rijndael

Характерні для початку 70-х років XX століття принципи побудови блокових шифрів були обумовлені, з одного боку, рівнем розвитку математичних наук, з іншого – компромісом між необхідною швидкістю шифрування і складністю технічної реалізації алгоритму, що забезпечує необхідну криптографічну стійкість. Оскільки електронна промисловість у той час масово освоїла тільки випуск мікросхем малого і середнього рівня інтеграції, то вибір того або іншого алгоритмічного рішення в значній мірі залежав від існуючої мікроелектронної бази і характеристик обчислювальної техніки, перш за все, від її швидкодії і обсягу оперативної пам'яті.

Створений в ці роки і затверджений як стандарт США в 1977 році криптоалгоритм DES реалізує архітектуру, що одержала назву збалансована мережа Файстеля (balanced Feistel network). Відповідно до цієї архітектури весь процес шифрування складається з серії однотипних циклів (раундів).

Даний принцип побудови займав домінуюче місце в криптографії блокових шифрів практично до наших днів. Зокрема, він лежить в основі ряду шифрів, що широко використовуються, включаючи криптоалгоритм, визначений стандартом ГОСТ 28147-89 (далі – ГОСТ), а також алгоритм IDEA, реалізований в програмному продукті PGP [8,30].

Зростання швидкостей обробки інформації ІТС до десятків гігабітів в секунду висунуло підвищені вимоги до часу виконання криптографічних перетворень. З іншого боку, розвиток науки і техніки відкрив нові можливості для проведення ефективного криптоаналізу і атак на шифри, що поставило вимоги до підвищення їх криптографічної стійкості. В той же час, поява великих інтегральних схем і збільшення обсягів оперативної пам'яті створили передумови для розробки і впровадження більш досконалих шифрів.

Перераховані чинники зумовили розробку нових підходів до побудови блокових симетричних криптоалгоритмів.

В кінці 90-х років Національний інститут стандартів і технологій (NIST) ініціював конкурс на створення нового стандарту шифрування

замість алгоритму DES. Всього було запропоновано 15 криптографічних алгоритмів, що задовольняли основні конкурсні вимоги. Після підведення проміжних підсумків залишилося п'ять фіналістів, з яких в жовтні 2000 року для затвердження як стандарт AES (advanced encryption standard) був вибраний криптографічний алгоритм Rijndael (Рендел) [12, 15, 40].

На відміну від ГОСТ, довжину ключа і розмір шифрованого блоку алгоритму Rijndael можна змінювати, що досягається завдяки застосованій схемі шифрування. При цьому кількість раундів (циклів роботи) алгоритму визначається довжиною ключа і розміром шифрованого блоку за таким правилом: якщо максимальна з цих двох величин рівна 128 бітам, то встановлюється 10 раундів, для 192 бітів – 12 раундів і для 256 бітів – 14 раундів (таблиця 6.1).

Це дозволяє в залежності від вихідних вимог до криптосистеми варіювати в достатньо широких межах стійкістю (від $10^{36}$ до $10^{77}$), і в не дуже широких – швидкодією алгоритму (не більше ніж в 1.4 раза). Відзначимо, що в стандарті AES вибраний розмір блоку, рівний 128 бітам.

Таблиця 6.1 - Порівняльні характеристики ГОСТ 28147-89 і Rijndael

| Характеристика | Криптоалгоритм | |
|---|---|---|
| | ГОСТ 28147-89 | Rijndael |
| Розмір блоку в бітах | 64 | 128, 192, 256 |
| Довжина ключа в бітах | 256 (+512) | 128, 192, 256 |
| Архітектура | Однорідна збалансована мережа Файстеля | «квадрат» (Square) |
| Число циклів (раундів) | 32 | 10, 12, 14 |
| Частина блоку, шифрована за один цикл (раунд), бітів | 32 (половина блоку) | 128, 192, 256 (повний блок) |
| Довжина раундового ключового елемента, бітів | 32 (половина блоку) | 128, 192, 256 (рівна розміру блоку) |
| Операції, що використовуються | Додавання за mod 2 і mod $2^{32}$, операції потетрадної заміни, цикл. зсув вліво на 11 бітів | Різні операції над кінцевими полями і кільцями |

В алгоритмі ГОСТ, як і в DES, весь процес шифрування розбитий на серію однотипних циклів (раундів). Особливості побудови алгоритмів

ГОСТ і Rijndael, а також їх властивості зручно вивчати на основі їх порівняльного аналізу, запропонованого в [13].

В кожному циклі алгоритму ГОСТ блок $T$, який шифрують, являє собою конкатенацію – об'єднання лівої $L_i$ і правої $R_i$ частин, одна з яких модифікується шляхом побітового додавання за mod 2 із значенням, що виробляється з іншої частини і ключового елемента циклу за допомогою функції шифрування. Між циклами частини блоку міняються місцями, і, таким чином, в наступному циклі змінений блок не модифікується та навпаки.

Схема шифрування за ГОСТ наведена на рис. 6.1, а. Подібна архітектура дозволяє легко одержати обернене криптографічне перетворення зі складної і, можливо, необерненої функції шифрування. Особливістю цього підходу є те, що в одному циклі шифрується рівно половина блоку.

Шифр Rijndael (рис. 6.1, б) має принципово іншу архітектуру, що одержала назву «квадрат» (від авторської назви першої версії шифру – Square). Ця архітектура базується на прямих перетвореннях блоку, що шифрується, який подається у формі матриці байтів.

Зашифрування складається з серії однотипних кроків (раундів), проте в кожному раунді блок перетворюється як єдине ціле і не залишається незмінних частин блоку. Оскільки за один раунд шифрується повний блок, то для досягнення зіставної з мережею Файстеля складності і нелінійності перетворення таких кроків потребується вдвічі менше, ніж в ГОСТі.

Кожний раунд полягає в побітовому додаванні за mod 2 поточного стану шифрованого блоку і ключового елемента раунду, за яким слідує складне нелінійне перетворення блоку, сконструйоване з трьох більш простих перетворень, розглянутих далі.

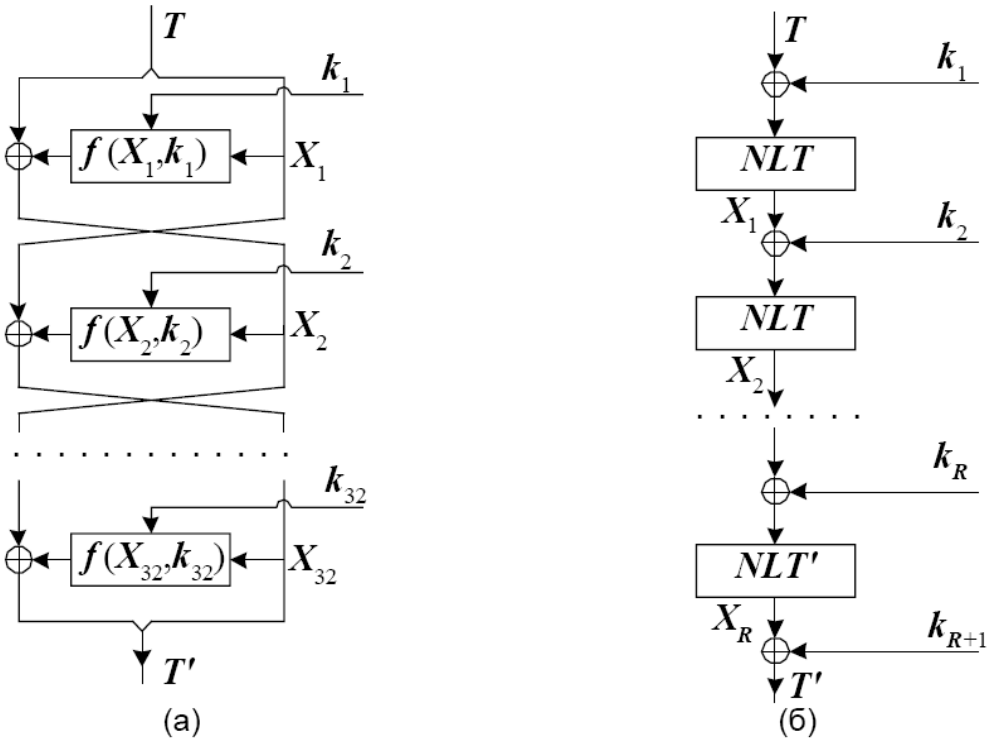



Рисунок 6.1 - Шифрування за алгоритмами ГОСТ 28147-89 (а) і Rijndael (б)

Позначення, прийняті на рис. 6.1, мають такий зміст.

$T, T'$ –вихідний і зашифрований блоки, відповідно;

$k_i$ – ключовий елемент циклу (раунду);

$X_i$ – зашифрована після $i$-го циклу частина блоку;

$f(X,k)$– функція шифрування алгоритму ГОСТ;

$NLT$, $NLT'$– регулярне нелінійне перетворення і нелінійне перетворення останнього раунду алгоритму Rijndael відповідно;

$R$ – число раундів в алгоритмі Rijndael (10, 12 або 14).

## 6.2 Порівняння раундів шифрування ГОСТ 28147-89 і Rijndael

Алгоритм ГОСТ реалізує відносно нескладну функцію шифрування, яка складається з операції додавання за модулем $2^{32}$ вхідного півблоку з ключовим елементом раунду, восьми підстановок, що виконуються незалежно у восьми тетрадах, і бітової перестановки – обертання на 11 бітів у бік старших розрядів.

Схема раунду шифрування за ГОСТ зображена на рис. 6.1, а.

В Rijndael блок, який шифрують і його проміжні стани в ході перетворення подаються у вигляді матриці байтів $4n$, де $n$ = 4, 6 , 8, залежно від розміру блоку.

Аналог циклової функції шифрування в алгоритмі Rijndael складається з трьох наведених нижче елементарних перетворень, що виконуються послідовно.

1. Байтова підстановка: кожний байт блоку замінюється новим значенням із загального для всіх байтів матриці вектора заміни.

2. Побайтовий циклічний зсув рядків матриці: перший рядок залишається незмінним, другий рядок зсувається вліво на один байт, третій і четвертий рядок для $n$ = 4, 6 зсуваються вліво на 2 і 3 байти відповідно, а для $n$ = 8 на 3 і 4 байти.

3. Множення матриць (змішування колонок): одержана на попередньому кроці матриця множиться зліва на матрицю-циркулянт розміру 4×4, де

$$M = \begin{pmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{pmatrix}.$$

Операції з елементами матриць (додання і множення) виконуються в скінченному полі $GF(2^8)$, породженому незвідним над $GF(2)$ поліномом $m(x) = x^8 \oplus x^4 \oplus x^3 \oplus x \oplus 1$.

В цьому полі додавання байтів виконується як побітове підсумовування за mod 2. При множенні двох байтів, кожний з них перетвориться на многочлен, вектор коефіцієнтів якого є послідовністю бітів, які створюють відповідний байт. Ці многочлени перемножуються над $GF(2)$, а потім обчислюється лишок добутку за модулем $m(x)$. Коефіцієнти лишку складають байт, що є результатом операції.

Схема раунду алгоритму Rijndael зображена на 6.2, б.

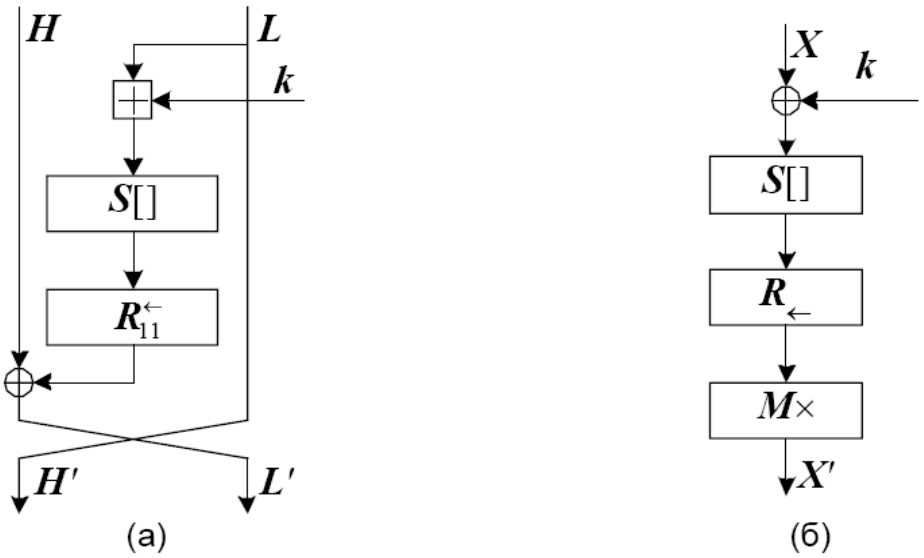



Рисунок 6.2 - Схема перетворення даних для 1-го раунду шифрування за алгоритмами ГОСТ 28147-89 (а) і Rijndael (б)

Позначення, прийняті на рис. 6.2, означають:

$X, X'$ – блок, що перетворювався алгоритмом Rijndael;

$H, L$ – старший і молодший півблоки на вході раунду ГОСТ відповідно;

$H', L'$ – до старший і молодший півблоки на виході раунду ГОСТ відповідно;

$k$ – ключовий елемент раунду;

$S[\ ]$ – функція підстановки – групами по 4 біти для ГОСТ і байтами для алгоритму Rijndael;

$R_{11}^{\leftarrow}$ – операція циклічного зсуву 32-бітового слова на 11 бітів у бік старших розрядів;

$R_{\leftarrow}$ – операція порядкового обернення матриці в алгоритмі Rijndael;

$M\times$ – множення в алгоритмі Rijndael матриці проміжних даних зліва на матрицю $M$.

Якщо в алгоритмі ГОСТ перестановку півблоків віднести до раунду шифрування, як це показано на рисунку 6.2, а, то можна помітити, що в обох алгоритмах всі раунди шифрування ідентичні один одному, за винятком останнього раунду, в якому відсутня частина операцій.

Такий підхід дозволяє одержати найкомпактнішу реалізацію, як апаратну, так і програмну.

В останньому раунді ГОСТу відсутня операція перестановки півблоків шифрованого блоку, в останньому раунді алгоритму Rijndael – множення зліва на матрицю $M$.

В обох обговорюваних алгоритмах це зроблено для того, щоб забезпечити еквівалентність структури прямого і оберненого шифрувальних перетворень, про що буде далі йти мова.

В алгоритмі ГОСТ еквівалентність структури прямого і оберненого криптографічного перетворення не забезпечується спеціально, а є простим наслідком використаного архітектурного рішення.

В будь-якій однорідній збалансованій мережі Файстеля обидва ці перетворення алгоритмічно ідентичні і розрізняються тільки порядком використання ключових елементів: при розшифруванні елементи використовуються в порядку, оберненому тому, в якому вони використовуються при зашифруванні.

Шифр Rijndael побудований на базі прямих перетворень. Як і для всіх подібних алгоритмів, обернене перетворення будується із обернень кроків прямого перетворення, вживаних в зворотному порядку.

В зв’язку з вищесказаним, забезпечити такий же ступінь ідентичності прямого і оберненого перетворень, який досягається в мережах Файстеля, не є можливим.

Проте за рахунок алгоритмічних рішень можна досягти ідентичності прямого і оберненого перетворень з точністю до констант, що використовуються в перетвореннях.

В таблиці 6.2 наведені послідовності операцій двох останніх раундів алгоритму Rijndael, їх формальне обернення, а також модифіковане обернене перетворення, яке побудоване з урахуванням співвідношень між параметрами і алгоритмічно подібне прямому.

Через $R_{\rightarrow}$ позначена обернена до $R_{\leftarrow}$ операція циклічного зсуву рядків матриці. Як видно з перших двох стовпців таблиці 6.2, алгоритмічна структура прямого і оберненого перетворень істотно розрізняються. Проте шляхом тотожних перетворень можна добитися більшої відповідності між ними.

Таблиця 6.2 - Два останні раунди алгоритму Rijndael і їх обернення

| Пряме перетворення | Обернене перетворення | Обернене перетворення (модифіковане) |
|---|---|---|
| $X=X\oplus k_{R-1}$ | $X=X\oplus k_{R+1}$ | $X=X\oplus k_{R+1}$ |
| $X=S(X)$ | $X=R_{\rightarrow}(X)$ | $X=S^{-1}(X)$ |
| $X=R_{\leftarrow}(X)$ | $X=S^{-1}(X)$ | $X=R_{\rightarrow}(X)$ |
| $X=M\times X$ | $X=X\oplus k_R$ | $X=M^{-1}\times X$ |
| $X=X\oplus k_R$ | $X=M^{-1}\times X$ | $X=X\oplus( M^{-1}\times k_R)$ |
| $X=S(X)$ | $X=R_{\rightarrow}(X)$ | $X=S^{-1}(X)$ |
| $X=R_{\leftarrow}(X)$ | $X=S^{-1}(X)$ | $X=R_{\rightarrow}(X)$ |
| $X=X\oplus k_{R+1}$ | $X=X\oplus k_{R-1}$ | $X=X\oplus k_{R-1}$ |

Перш за все відзначимо, що операція побайтової заміни *S* комутативна з процедурою побайтового зсуву рядків матриці:

$$S^{-1}\left(R_{\rightarrow}(X)\right)=R_{\rightarrow}\left(S^{-1}(X)\right).$$

Крім того, згідно з правилами матричної алгебри за законом асоціативності можна також поміняти порядок побітового додавання ключа за модулем два і множення на матрицю:

$$M^{-1}\times\left(X\oplus k_R\right)=\left(M^{-1}\times X\right)\oplus\left(M^{-1}\times k_R\right).$$

Застосовуємо вказані зміни до другого стовпця таблиці 6.2, одержуємо в третьому стовпці модифіковану послідовність операцій при двох раундах оберненого перетворення.

Порівняння 1-го і 3-го стовпців таблиці свідчить про те, що алгоритмічна структура прямого і оберненого перетворень ідентична. Результат легко узагальнюється на довільне число раундів.

Таким чином, в алгоритмі Rijndael процедури зашифрування і розшифрування алгоритмічно ідентичні і розрізняються тільки в деталях:

a. в оберненому перетворенні використовується вектор замін, обернений в операційному значенні вектору замін прямого перетворення;

b. в оберненому перетворенні число байтів, на які зсувається кожний рядок матриці даних в операції порядкового байтового зсуву, інше;

c. в оберненому перетворенні в кроці матричного множення блок даних множиться зліва на матрицю

$$\circ \quad M^{-1} = \begin{pmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{pmatrix},$$

обернену тій, що використовується при прямому перетворенні;

d. в оберненому перетворенні ключові елементи використовуються в оберненому порядку, і, крім того, всі елементи, за винятком першого і останнього, повинні бути помножені зліва на матрицю $M^{-1}$.

Таким чином, аналогічно ГОСТу, в алгоритмі Rijndael можна сумістити реалізацію процедур шифрування як при апаратній, так і при програмній реалізації.

### 6.3 Формування ключових елементів

У стандарті шифрування ГОСТ для вироблення тридцяти двох 32-бітових ключових елементів з 256-бітового ключа застосований дуже простий підхід. Ключ інтерпретується як масив, що складається з восьми подключів: $K = (k_1, k_2, \ldots, k_8)$.

Таблиця 6.3 - Порядок вибору подключів в алгоритмі ГОСТ 28147-89

| № раунду | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Ключ. елемент | $k_1$ | $k_2$ | $k_3$ | $k_4$ | $k_5$ | $k_6$ | $k_7$ | $k_8$ | $k_1$ | $k_2$ | $k_3$ | $k_4$ | $k_5$ | $k_6$ | $k_7$ | $k_8$ |
| № раунду | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 | 32 |
| Ключ. елемент | $k_1$ | $k_2$ | $k_3$ | $k_4$ | $k_5$ | $k_6$ | $k_7$ | $k_8$ | $k_8$ | $k_7$ | $k_6$ | $k_5$ | $k_4$ | $k_3$ | $k_2$ | $k_1$ |

Ключові елементи використовуються на раундах шифрування – послідовність подключів «проглядається» три рази в прямому порядку і один раз в оберненому, в результаті кожен ключовий елемент використовується рівно чотири рази. У таблиці 6.3 наведена відповідність номера раунду і використовуваного на раунді ключового елемента.

У шифрі Rijndael використовується трохи складніша схема, що враховує можливу відмінність в розмірах блоку алгоритму і ключа.

Існують два алгоритми генерації послідовності ключових елементів – для ключа розміром 128/192 біти і для ключа розміром 256 біт, які розрізняються досить незначно.

Ключ і ключова послідовність подаються у вигляді векторів з компонентами з чотирибайтових слів. Початкова ділянка послідовності заповнюється словами з ключа – точно так, як і в ГОСТі.

Подальші слова ключової послідовності виробляються за рекурентним співвідношенням групами, кратними розміру ключа.

Перше 4-байтове слово такої групи виробляється з використанням достатнього складного нелінійного перетворення, всі інші – за простим лінійним співвідношенням:

$$w_i = \begin{cases} w_{i-N_K} \oplus G(w_{i-1}), & i \bmod N_K = 0 \\ w_{i-N_K} \oplus w_{i-1}, & i \bmod N_K \neq 0 \end{cases}.$$

Тут $N_K$ – число 32-бітових слів в ключі (4 або 6, для 8-ми використовується інша схема); $G(\cdot)$ – нелінійне перетворення 32-бітових слів. Це перетворення має вигляд: $G(x) = S(R_S^{\leftarrow}(x)) \oplus P(i/N_K)$ і включає байтовий зсув, побайтову підстановку і побітове додавання за mod 2 з вектором, залежним від номера групи елементів, що виробляється.

$S$ – це описана вище функція побайтової заміни, $R_S^{\leftarrow}$ – операція циклічного зсуву аргументу на вісім бітів вліво, а $P(i/N_K)$ – 4-байтове слово, що конструюється особливим чином і не залежне від ключа.

Перетворення $G(x)$ вносить складність і нелінійність в схему вироблення ключових елементів, ускладнюючи тим самим криптоаналіз шифру.

Отримані з описаного вище потоку 4-байтові слова групуються в послідовності ключових елементів довжини, рівної розміру шифрованого блоку, для використання на раундах шифрування як описано вище.

З вищесказаного витікає, що алгоритм вироблення ключової послідовності в шифрі Rijndael складніший, ніж в ГОСТі. Проте він достатньо ефективний і не вносить скільки-небудь істотного внеску до загального об'єму обчислювальних витрат на шифрування. Швидкість шифрування з обчисленням ключових елементів «на льоту» незначно

менше швидкості шифрування з використанням заздалегідь підготовлених ключових елементів.

### 6.4 Вибір вузлів замін і констант, дифузійні характеристики

Найважливішими константами ГОСТу є довготривалі ключові елементи – вузли замін. Вони не зафіксовані в стандарті, а постачаються спеціалізованими організаціями, що забезпечують користувачів шифру ключовою інформацією. Через це будь-які відомості про критерії проектування вузлів замін відсутні.

Із загальних міркувань можна помітити, що, швидше за все, вузли замін виробляються з використанням однієї з методик конструювання вузлів (наприклад, з використанням так званих бент-функцій [8, 14]), а потім оцінюються за декількома критеріями, і вузли, що мають недостатню якість, бракуються. Серед критеріїв оцінки, ймовірно, присутні такі:

– складність і нелінійність булевих функцій, що описують вузли;

– диференціальна характеристика вузлів замін;

– лінійна характеристика вузлів замін.

На відміну від ГОСТу, в шифрі Rijndael відомі критерії вибору вектора замін. При його конструюванні повинні бути взяті до уваги такі міркування:

– мінімізація найбільшої за величиною характеристики кореляції між лінійними комбінаціями вхідних і вихідних бітів (визначає стійкість до лінійного криптоаналізу);

– мінімізація найбільшого нетривіального значення в таблиці частот пар різниць (визначає стійкість до диференціального криптоаналізу);

– складність алгебраїчного виразу, що описує вузол, в полі $GF\left(2^8\right)$.

Автори алгоритму Rijndael відзначили, що якщо даний вузол замін викликає сумніви в його якості і у відсутності «потайного ходу», то він може бути замінений на іншій.

Причому, структура шифру і кількість циклів вибрані так, щоб навіть для випадково вибраного вектора замін шифр був би стійкий щодо методів диференціального і лінійного криптоаналізу.

Операція байтової заміни в алгоритмі Rijndael описується рівнянням:

$$S(X)=(x^7+x^6+x^2+x)+X^{-1}(x^7+x^6+x^5+x^4+1)\bmod(x^8+1).$$

Це перетворення включає мультиплікативну інверсію байта $X$ в полі $GF(2^8)$ (значення 00 при цьому міняється саме на себе) і афінне перетворення результату.

Його поліноми вибрані так, щоб у результаті були відсутні точки нерухомості: $S(X)=X$ і «антинерухомості»: $S(X)=\sim X$, де знак «~» позначає операцію побітового інвертування.

З інших важливих констант алгоритму Rijndael необхідно відзначити матрицю $M$ раундового перетворення (змішування колонок).

Метою цього кроку є дифузія зміни в одному байті на весь стовпець матриці.

При виборі матриці, крім звичайних вимог оберненості і простоти опису, були взяті до уваги такі бажані характеристики перетворення.

1. Лінійність в полі GF(2).

2. Достатній рівень дифузії.

3. Висока швидкість реалізації на 8-бітових процесорах.

Даний крок перетворення може бути також представлений як множення стовпців перетворюваної матриці даних, інтерпретованих як поліноми третього степеня з коефіцієнтами із поля $GF(2^8)$, на інший поліном $c(x)$ з коефіцієнтами із цього ж поля, за модулем полінома $x^4+1$.

Згідно з третьою з перерахованих вище вимог, коефіцієнти множника повинні бути якомога менші. Автори шифру вибрали поліном, взаємно простий з (наведеним) поліномом $x^4+1$:

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

Виявляється, та ж процедура, записана в матричній формі, еквівалентна множенню зліва на матрицю $M$ – циркулянт полінома $c(x)$.

Таким чином, на відміну від ГОСТу, раунд якого побудований на традиційних для багатьох блокових алгоритмів операціях, таких як підстановки в малих бітових групах, бітові перестановки – зсуви, адитивні операції з елементами даних, шифр Rijndael базується на двох основних перетвореннях: байтовій заміні і операції перемішування стовпців матриці даних за допомогою її множення зліва на матрицю $M$.

Аналіз властивостей стійкості алгоритмів почнемо з дослідження характеристик дифузії.

Розглянемо вплив змін в початкових даних на вихідне значення функції шифрування алгоритму ГОСТ.

Нехай у вхідному 32-бітовому півблоці змінений один біт. Першою операцією функції шифрування є додавання за модулем $2^{32}$, внаслідок чого можливе перенесення з молодших розрядів в старші.

Теоретично, зміна наймолодшого біта операнда може привести до зміни всіх бітів суми. Проте ймовірність цієї події надзвичайно мала.

Можна показати, що за умови рівноймовірного і незалежного розподілу бітів операндів на множині {0,1} ймовірність події, що вплив одного біта операнда розповсюджується вліво рівно на $n$ бітів результату, рівна приблизно $2^{-n}$.

Практично можна констатувати, що при додаванні за модулем $2^{32}$, зміна одного біта операнда зробить вплив не більше, ніж на дві сусідні тетради результату.

Припустимо, що в аргументі циклової функції шифрування змінено значення одного з бітів. В результаті підсумовування з ключовим елементом ця зміна з ймовірністю 1/32 розповсюдиться на групу з 5 бітів суми.

Наступною операцією у функції шифрування є заміна в тетрадах. П'ять суміжних бітів суми покривається двома такими групами. Отже, після виконання заміни зміна розповсюдиться на 8 біт, що входять до цих двох груп.

Подальший циклічний зсув вліво на 11 бітів не змінює числа бітів, що були зачеплені, проте змінює їх положення в слові: ці 8 бітів будуть розподілені по трьох суміжних тетрадах, як показано на рис. 6.3

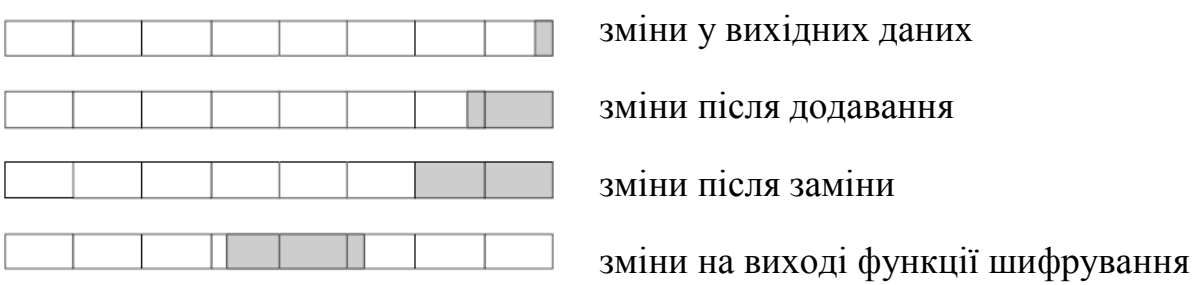



Рисунок 6.3 - Дифузія спотворень на раунді ГОСТ 28147-89

На рисунку кожний прямокутник відповідає одній тетраді, де біти, що їх торкнулися зміни, виділені сірим кольором.

Тепер прослідкуємо зміну у вихідних даних через декілька раундів шифрування припускаючи, що інвертовано біт в молодшій половині блоку. Відповідні діаграми наведено на рис. 6.4.

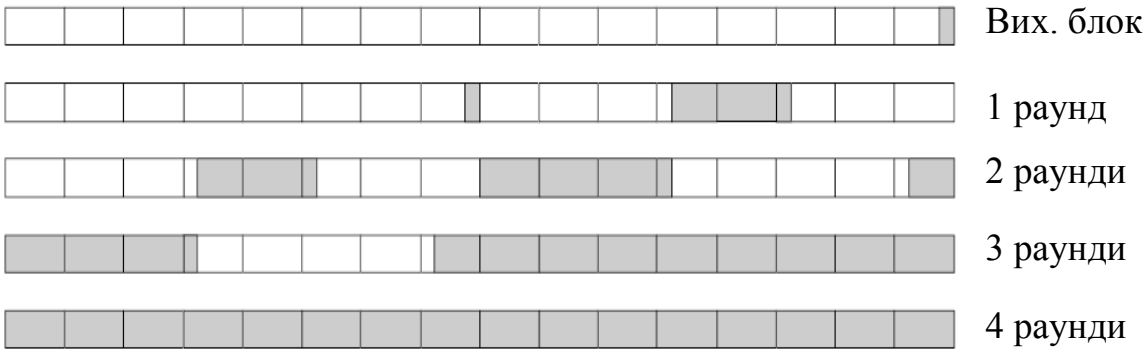



Рисунок 6.4 - Дифузія спотворень після декількох раундів ГОСТ 28147-89

Як видно на рисунку, зміна одного біта в молодшій половині вихідного блоку після чотирьох раундів шифрування розповсюджується на весь шифрований блок.

Якщо ж буде інвертований біт із старшої половини, то це число збільшиться на 1 і стане рівним 5. Таким чином, за 32 раунди шифрування, передбачені ГОСТом, дані встигають повністю перемішатися 32:5≈6 раз.

Тепер розглянемо дифузійні характеристики алгоритму Rijndael. Перша операція раунду шифрування цього алгоритму – побітове підсумовування з ключем за модулем 2 – не приводить до виходу зміни за межі одного біта. Наступна операція – заміна по таблиці – поширює зміну в одному біті на весь байт.

Наступний за нею порядковий байтовий зсув не змінює нічого в цій картині. І, нарешті, завершальна операція раунду – перемішування байтів в стовпцях матриці – приводить до дифузії зміни на весь стовпець. Таким чином, за один раунд шифрування зміна в одному біті вхідних даних зробить вплив на один стовпець матриці даних.

На наступному раунді шифрування ці байти в ході операції порядкового байтового зсуву будуть «розведені» по різних стовпцях, і в

результаті подальшої операції перемішування байтів в стовпцях вихідна зміна розповсюдиться на 4 стовпці.

Тобто на всю матрицю даних при 128-бітовому блоці даних, на 2/3 матриці при 192-бітовому блоці і на її половину при 256-бітовому блоці даних. Діаграма дифузії в алгоритмі Rijndael наведена на рис. 6.5.

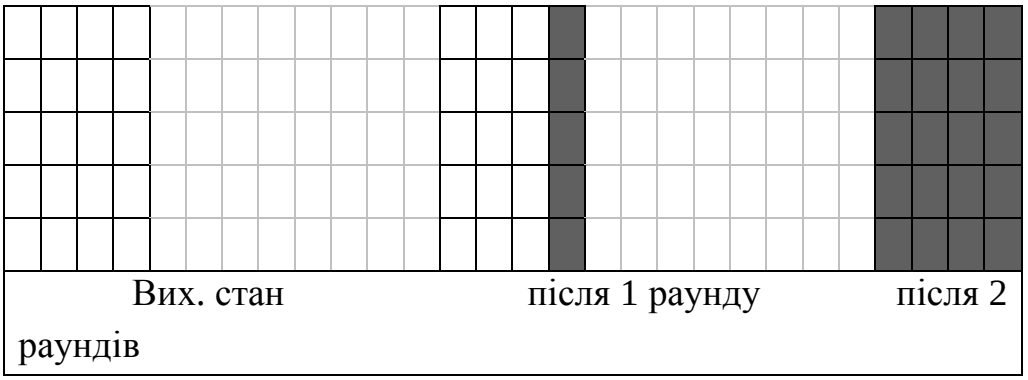



Рисунок 6.5 - Дифузія спотворень після декількох раундів Rijndael

Таким чином, при шифруванні 128-бітових блоків зміна в одному біті вхідних даних розповсюджується на весь блок рівно за два раунди, при більшому розмірі блоку – за три. В результаті за 10-14 раундів алгоритму Rijndael дані встигають цілком перемішатися 5-7 разів.

Як видно з вищевикладеного, за показниками дифузії алгоритми ГОСТ і Rijndael несуттєво відрізняються один від одного.

## 6.5 Показники стійкості, продуктивності і зручність реалізації алгоритмів

Розглянемо стійкість обох алгоритмів до відомих видів криптоаналізу.

Найбільш універсальними та ефективними для алгоритмів широкого класу є диференціальний і лінійний види криптоаналізу.

Дати оцінку стійкості алгоритму ГОСТ до конкретних видів криптоаналізу неможливо без специфікації вузлів замін, оскільки стійкість шифру залежить від властивостей використаних підстановок. В той же час, досвід дослідження близьких за архітектурою шифрів з відомими таблицями замін (включаючи DES) свідчить про те, що криптоаналіз шифру до 16 раундів має сенс тільки за наявності великого обсягу

вихідних даних, а при 20-24 раундах стає практично даремним. ГОСТ реалізує 32 раунди шифрування, що забезпечує достатній запас, щоб успішно протистояти вказаним видам криптоаналізу.

У відкритому друці відсутні повідомлення про успішне розкриття ГОСТу з якими-небудь конкретними вузлами замін як з тестовими, наведеними в стандарті ГОСТ Р34.11-94, так і з тими, що постачалися в комерційні організації з конкретними реалізаціями ГОСТу.

За оцінкою розробників, вже на 4-х раундах шифрування алгоритм Rijndael набуває достатню стійкість до вказаних видів атак.

Залежно від розміру блоку проведення лінійного і диференціального криптоаналізу алгоритму втрачає сенс після 6-8 циклів, тоді як його специфікацією передбачено 10-14 циклів. Отже, шифр Rijndael також стійкий до вказаних видів атак з деяким запасом.

Таким чином, обидва шифри мають достатню стійкість до відомих видів криптоаналізу. В наукових публікаціях немає відомостей про атаки на вказані шифри, які теоретично дозволили б дешифрувати повідомлення з меншими обчислювальними затратами, ніж повний перебір всіх ключів.

Для оцінки досягнутої ефективності апаратної реалізації шифрів головним критерієм є кількість і складність елементарних операцій, які необхідно виконати в циклі шифрування, а також можливість їх паралельного виконання.

Оцінка ефективності програмних реалізацій шифрів повинна орієнтуватися, головним чином, на 32-бітові платформи, оскільки такі комп'ютери нині складають велику частину світового парку.

В той же час інтерес викликають реалізації шифрів на 8-бітових мікроконтролерах, що є основою технології інтелектуальних карт, яка використовується різними системами безготівкових розрахунків. Число користувачів таких систем останнім часом зростає швидкими темпами.

Стандарт шифрування ГОСТ зручний як для апаратної, так і для програмної реалізації. При довжині блоку даних 64 біти основна робота ведеться з половинками цього блоку – 32-бітовими словами, що дозволяє ефективно реалізувати стандарт на більшості сучасних комп'ютерів.

Найбільш трудомісткою операцією при реалізації ГОСТу на 32-бітових ПК є заміна. Тому при його програмній реалізації підстановки в тетрадах можна попарно об'єднати і виконувати заміну на байтах, що істотно ефективніше. Належне подання підстановок заміни дозволить

також уникнути виконання циклічного зсуву слова на виході функції шифрування, якщо вузли заміни зберігати як масиви 4-байтових слів, в яких вже виконані необхідні зсуви.

Така «роздута» таблиця замін вимагатиме для свого зберігання $4 \times 2^8 \times 4 = 2^{12}$ байтів або 4K оперативної пам'яті. Одна заміна реалізується за три однотактові машинні команди: завантаження байта в індексний регістр, завантаження замінювального значення в регістр, використання завантаженого значення в операції побітового підсумовування.

У результаті запропоновані методи оптимізації дозволяють реалізувати раунд шифрування за ГОСТом за 15 однотактових команд ПК. З урахуванням можливості паралельного виконання команд процесорами Intel Pentium раунд ГОСТу може бути реалізований за 8 тактів роботи процесора, а весь процес шифрування – за $32 \times 8 = 256$ тактів.

На процесорі Intel Pentium 200 це може дозволити досягти швидкості шифрування до 6.0 Мбайт/сек.

Оскільки більшість елементарних операцій, що використовуються ГОСТом, входить в систему команд поширених 8-бітових мікроконтролерів, його реалізація в цьому випадку не викликає ускладнень. Виняток – складання за модулем $2^{32}$ потрібно виконати як одну операцію додавання без переносу і три операції додавання із переносом, що виконувалися каскадно.

Апаратна реалізація одного циклу ГОСТу складається з послідовного виконання 3-х операцій над 32-бітовими аргументами: підсумовування, заміна одночасно у всіх восьми тетрадах і побітове додавання за модулем 2. Циклічний зсув не вимагає окремої операції, оскільки досягається простою комутацією провідників. Таким чином, апаратне виконання ГОСТу вимагає виконання 106 елементарних операцій на циклі шифрування, і ця робота не може бути розпаралелена.

Тепер розглянемо особливості реалізації алгоритму Rijndael. Цей алгоритм є байт-орієнтованим, тобто цілком може бути сформульований в термінах операцій з байтами. В алгоритмі використовуються алгебраїчні операції в кінцевих полях, найскладнішою з яких є множення в $GF\left(2^8\right)$.

Безпосереднє виконання цих операцій привело б до надто неефективної реалізації алгоритму. Однак байтова структура шифру відкриває широкі можливості з оптимізації програмної реалізації. Заміна

байта у таблиці із подальшим множенням на константу в кінцевому полі $GF(2^8)$ може бути подана як одна заміна у таблиці.

В прямому шифрі використовуються три константи (01, 02, 03) і, отже, знадобляться три такі таблиці. В зворотному – чотири (0E, 0D, 0B, 09).

При належній організації процесу шифрування порядковий байтовий зсув матриці даних можна не виконувати.

При реалізації на 32-бітових платформах можна реалізувати байтову заміну і множення елемента матриці даних на стовпець матриці $M$ як одну заміну 8 бітів на 32 біти. Таким чином, перетворення одного 32-бітового слова даних включає чотири байтові заміни, кожна з яких, як було відзначено вище, вимагає трьох однотактових машинних команд.

У результаті, частина раунду для одного 32-бітового слова може бути реалізована на процесорах Intel Pentium за 14 команд або за 7 тактів, що при 14 раундах шифрування дозволяє на процесорах Intel Pentium 200 досягти теоретичної межі швидкодії приблизно 7.8 Мбайт/сек, незалежно від розміру блоку даних і ключа. Для меншого числа раундів швидкість збільшується.

Відповідна оптимізація приведе до певних витрат оперативної пам'яті. Для кожного стовпця матриці $M$ будується свій вектор заміни одного байта на 4-байтове слово, отримуємо таблицю заміни, розміром $4 \times 2^8 \times 4 = 2^{12}$ байтів або 4K.

Далі, таблиці, що використовуються при зашифруванні і розшифруванні, різні – це подвоює вимоги до обсягу оперативної пам'яті. Крім того, для виконання останнього циклу розшифрування потрібен окремий вузол замін, його розмір рівний 256 байтам або 0.25 Кбайта. У результаті отримуємо, що для 32-бітових програмних реалізацій шифру Rijndael необхідні 8.25 Кбайта оперативної пам'яті для зберігання вузлів замін. Для сучасних комп'ютерів ця вимога не виглядає надмірною.

Байт-орієнтована архітектура алгоритму Rijndael дозволяє надзвичайно ефективно реалізувати його на 8-бітових мікроконтролерах, використовуючи тільки операції завантаження-вивантаження регістрів, індексованого витягування байта з пам'яті і побітового підсумовування за модулем два.

Також вказана особливість дозволяє виконати ефективну програмну реалізацію алгоритму. Раунд шифрування вимагає виконання 16 байтових

замін плюс чотири операції побітового виключного або над 128-бітовими блоками, які можуть бути виконані в три етапи.

У результаті отримуєм 4 операції на цикл або 57 операцій на 14-раундовий цикл шифрування з урахуванням «зайвої» операції побітового додавання ключа за модулем два – це приблизно вдвічі менше, ніж в ГОСТі.

Оскільки Rijndael має вдвічі більший розмір блоку, це веде до приблизно чотирикратної переваги в швидкості за умови апаратної реалізації на базі однієї і тієї ж технології. Необхідно відмітити, що вказана вище оцінка є досить грубою.

Авторами роботи [13] була проведена оцінка практичних характеристик швидкодії програмних реалізацій порівнюваних шифрів на платформі Intel Pentium, для шифру Rijndael розглядався варіант із 14 раундами.

Для кожного з алгоритмів мовою Сі був написаний еквівалент функції зашифрування одного блоку, в якому послідовність раундів була розгорнена в лінійний код, це дозволяє досягти максимальної швидкодії.

В еквівалентах використовувалася випадкова ключова інформація і випадкові вузли замін, однак на швидкодію реалізації це не впливає, оскільки швидкість виконання використаних команд процесора не залежить від їх операндів.

Функції, що виконують зашифрування, викликалися по $10^7$ раз, і вимірювався час їх виконання, який потім перераховувався в показник швидкодії. Для компіляції і побудови виконуваного модуля використовувався компілятор Intel C++ v4.5, оскільки він дозволяє одержати код із максимальною швидкодією. Були випробувані також компілятори MS Visual C++ 6.0, Borland C++ v5.5 і GNU C++ v2.95.2, однак отриманий із їх використанням код був помітно повільніший.

Код оптимізувався для процесорів Intel Pentium і Intel Pentium Pro/II/III. За допомогою одержаних тестових задач була заміряна швидкодія реалізацій шифрів на процесорах Intel Pentium 166 МГц і Intel Pentium III 500 МГц. Результати вимірювань наведені в таблиці 6.4.

Очевидно, відношення величин швидкодії двох шифрів від типу процесора, практично, не залежить, тому результати даних експериментів актуальні і в даний час.

Таблиця 6.4 - Показники швидкодії реалізацій алгоритмів мовою С

| | ГОСТ 28147-89 | Rijndael, 14 раундів |
|---|---|---|
| Pentium 166 | 2.04 Мбайт/сек | 2.46 Мбайт/сек |
| Pentium III 500 | 8.30 Мбайт/сек | 9.36 Мбайт/сек |

Таким чином, при реалізації на 32-бітових платформах алгоритми мають порівнянні характеристики швидкодії. На 8-бітових комп'ютерах ситуація, ймовірно, схожа.

Що стосується апаратної реалізації, то, на відміну від ГОСТу, Rijndael дозволяє досягти високого рівня паралелізму при виконанні шифрування, містить менше число раундів, через що його апаратне втілення може виявитися істотно більш швидким. Якщо судити за довжиною найбільшого шляху в мережному поданні обох алгоритмів, його перевага приблизно чотирикратна.

Порівняння параметрів алгоритмів шифрування ГОСТ 28147-89 і Rijndael показує, що, не дивлячись на істотну відмінність в принципах побудови цих шифрів, їх основні характеристики порівнянні.

Винятком є те, що Rijndael має значну перевагу в швидкодії перед ГОСТом при апаратній реалізації на базі однієї і тієї ж технології.

За ключовими для алгоритмів такого роду параметрами криптостійкості жоден з алгоритмів не має істотної переваги, також приблизно однакові швидкості оптимальної програмної реалізації для процесорів Intel Pentium, що можна екстраполювати на всі сучасні 32-розрядні процесори.

Із сказаного можна зробити висновок, що стандарт шифрування ГОСТ відповідає вимогам, що висуваються до сучасних шифрів, і може залишатися стандартом ще достатньо довгий час. Очевидним кроком в його вдосконаленні може бути перехід від замін в тетрадах до байтових замін, що повинно ще більше підвищити стійкість алгоритму до відомих видів криптоаналізу.

## Питання до розділу 6

1. Яким чином подаються блоки даних в ході роботи алгоритму Rijndael?
2. На яких елементарних перетвореннях побудований Rijndael?

3. Які довжини блоків допускаються при шифруванні алгоритмами ГОСТ 28147-89 і Rijndael?

4. Які операції використовуються в циклі (раунді) алгоритму ГОСТ 28147-89?

5. Яким алгебраїчним системам належать операції, що використовуються в алгоритмі Rijndael?

6. На яких принципах реалізовано формування ключів в Rijndael?

7. Яка характеристика дифузії при зміні одного біта вхідного блоку для ГОСТ 28147-89?

8. Яка характеристика дифузії при зміні одного біта вхідних блоків різної довжини для Rijndael?

9. Яка кількість раундів ГОСТ 28147-89 достатня для його стійкості проти диференціального аналізу?

10. Яка кількість раундів Rijndael достатня для його стійкості проти диференціального аналізу?

# РОЗДІЛ 7 КРИПТОСИСТЕМИ З ВІДКРИТИМИ КЛЮЧАМИ

## 7.1 Односторонні функції з секретом і асиметричні системи

Найслабшою ланкою при реалізації симетричних криптосистем в системах захищеного електронного документообігу, електронних банківських платежів і, особливо, електронної торгівлі є питання розподілу ключів.

Для забезпечення обміну конфіденційною інформацією між двома абонентами телекомунікаційної мережі повинен бути згенерований ключ (можливо, одним з абонентів), а потім по деякому захищеному каналу переданий іншим користувачам (іншому абоненту). Як засіб створення захищеного каналу знову може бути використана деяка криптосистема.

Найбільшу гостроту питання розподілу і доставки ключів набуває у разі неможливості наперед описати склад інформаційно-телекомуні-каційної мережі.

Для вирішення цієї проблеми на основі результатів, одержаних класичною і сучасною алгеброю, були запропоновані системи з відкритим ключем (СВК). В СВК для зашифрування не використовуються секретні ключі – вони необхідні тільки при розшифрування (рис. 7.1).

Головним поняттям СВК є однонаправлена функція з секретом (one-way trapdoor function). Однонаправлену функцію з секретом $f_t(x)$: $D \to R$ легко обчислити для всіх $x \in D$, але дуже важко обернути майже для всіх значень із $R$. Однак, якщо використовується деяка секретна інформація $t$, то для всіх значень $y \in R$ легко обчислити величину $x \in D$, що задовольняє умову $y = f_t(x)$.

Криптосистеми з відкритим ключем, що використовують однонаправлені функції з секретом, ще називаються асиметричними (asymmetric cryptosystems).

В 1975 році Діффі і Хеллман в роботі, присвяченій криптографії з відкритим ключем, запропонували декілька варіантів побудови однонаправлених функцій з секретом. Проте ці функції не були точно асиметричними, тому незабаром Діффі і Хеллман запропонували більш вдалий варіант: показникову функцію за простим модулем.

Ця функція була використана в широко розповсюдженому криптографічному протоколі − протоколі обміну ключами Діффі-Хеллмана (Diffie-Hellman key exchange protocol).

Раніше, в 1974 році, Меркл (Merkle) винайшов механізм узгодження криптографічного ключа шляхом очевидних асиметричних обчислень, що одержали назву головоломка Меркла.

Асиметричність головоломки Меркла полягає в тому, що її обчислювальна складність для законних учасників протокола узгодження ключа і для перехоплювачів зовсім різна: легальні участники легко виконують обчислення, а нелегальні — ні. Головоломка Меркла – це перша ефективна реалізація однонаправленої функції з секретом.

Не дивлячись на те, що головоломка Меркла не підходить для застосування в сучасних криптографічних додатках, її вплив на криптосистеми з відкритим ключем неможливо переоцінити.

Останнім часом стало відомо, що першу криптосистему з відкритим ключем розробив британський математик Кокс (Cocks) ще в 1973 році. Алгоритм Кокса, що одержав назву алгоритму з несекретним ключем шифрування, використовував складність розкладання цілого числа на прості множники і, по суті, збігався з криптосистемою RSA.

Спочатку алгоритм Кокса був засекречений і лише в грудні 1997 року Група електронного захисту засобів зв'язку (Communications Services Electronic Security Group – CESG) його розсекретила.

Не дивлячись на те, що спочатку криптосистеми з відкритим ключем були надбанням вузького кола осіб, саме завдяки відкритим дослідженням вони знайшли два найважливіші застосування для створення алгоритмів цифрового підпису і механізмів обміну секретними ключами через відкриті канали зв'язку.

В наш час алгоритми шифрування з відкритим ключем отримали широке розповсюдження в інформаційних системах.

Так, алгоритм RSA [1, 16, 18] фактично став стандартом для відкритих систем і рекомендований міжнародною організацією зі стандартизації ISO, відповідні додатки лежать в основі електронної комерції, здійснюваної через Internet.

Суть їх полягає в тому, що кожним абонентом мережі генеруються два ключі, зв'язані між собою за певним математичним законом. Один ключ оголошується відкритим, а інший закритим (секретним).

Відкритий ключ опубліковується і доступний будь-кому, хто бажає послати повідомлення власнику секретного ключа. Секретний ключ зберігається в таємниці.

Вихідний текст шифрується відкритим ключем Одержувача і передається йому. Зашифрований текст в принципі не може бути розшифрований тим же відкритим ключем. Розшифрувати повідомлення можна тільки з використанням секретного ключа, який відомий лише самому адресату.

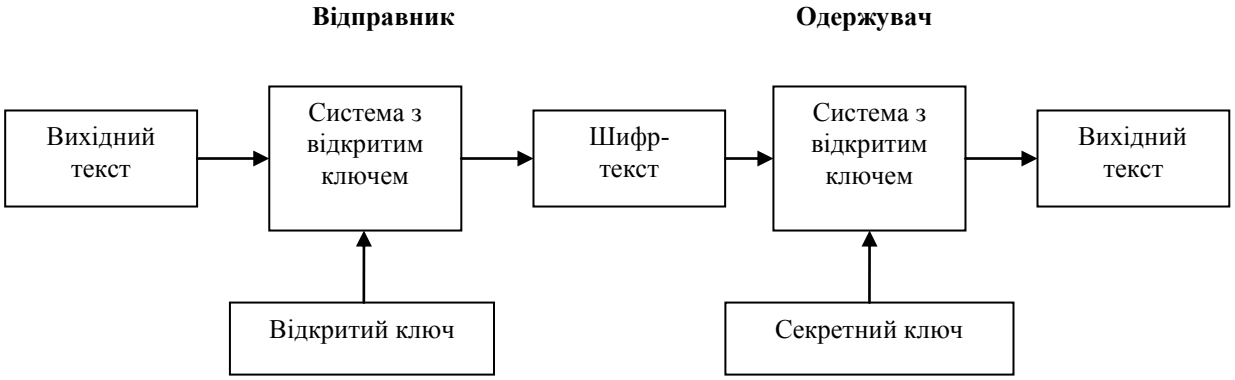



Рисунок 7.1 - Криптосистема з відкритим ключем

Таким чином, одностороння функція з секретом, що використовується в асиметричній системі, є взаємооднозначною, але разом з тим має властивості необерненості.

Необерненість, звичайно, розуміється не в загальноприйнятому значенні, а як практична неможливість обчислити обернену функцію, використовуючи сучасні обчислювальні засоби за досяжний інтервал часу.

Тому, щоб гарантувати надійний захист інформації, до СВК висуваються дві важливі і очевидні вимоги.

1. Перетворення вихідного тексту повинно бути обчислювано нереалізованим (необернениим) і уникати його відновлення на основі відкритого ключа.

2. Визначення закритого ключа на основі відкритого також повинно бути неможливим на сучасному технологічному рівні. При цьому бажана точна нижня оцінка складності (кількості операцій) розкриття шифру.

В основі сучасних криптосистем з відкритим ключем обчислювально необернені перетворення частіше за все будуються на основі таких алгоритмічних проблем:

− розкладання великих чисел на прості множники;

− обчислення логарифма в скінченному полі;

− знаходження кратності точки (дискретний логарифм) на еліптичній кривій;

− обчислення коренів алгебраїчних рівнянь над кільцями і полями.

Відзначимо області вживання СВК [4, 6, 7, 12, 20].

1. Засоби шифрування даних, що передаються і зберігаються.

2. Засоби аутентифікації користувачів і перевірки цілісності даних, формування електронного цифрового підпису.

3. Механізми розподілу ключів.

Алгоритми СВК працюють повільніше, ніж алгоритми симетричних шифрсистем. Тому на практиці переважним є комбіноване використовування СВК і симетричних систем. При цьому СВК використовується для створення механізмів розподілу ключів, об'єм яких незначний, а за допомогою симетричних алгоритмів здійснюється шифрування великих інформаційних потоків.

Найбільше розповсюдження отримали системи з відкритим ключем на основі алгоритму RSA, криптосистема Ель-Гамаля і криптосистеми на основі еліптичних кривих.

### 7.2 Криптосистема RSA

Криптосистема RSA, розроблена в 1977 році, одержала свою назву на честь її авторів: Рона Рівеста, Аді Шаміра і Леонарда Ейдельмана.

Розробники скористалися тим фактом, що знаходження великих простих чисел в обчислювальному відношенні здійснюється достатньо просто, але не відомий алгоритм, що виконує за поліноміальний час розкладання на прості множники великих чисел.

Доведено (теорема Рабіна), що розкриття шифру RSA еквівалентне знаходженню такого розкладу. Тому для будь-якої довжини ключа можна дати (сучасну практичну) нижню оцінку числа операцій для розкриття шифру, а з урахуванням продуктивності сучасних комп'ютерів оцінити і необхідний на це час.

Можливість реально оцінити захищеність алгоритму RSA стала однією з причин популярності цієї СВК на фоні десятків інших схем. Тому алгоритм RSA використовується в банківських комп'ютерних мережах,

особливо для роботи з віддаленими клієнтами (обслуговування кредитних карток).

В наш час алгоритм RSA використовується в багатьох стандартах, серед яких SSL, S-HTTP, S-MIME, S/WAN, STT і PCT. Крім того, алгоритм RSA реалізується як у вигляді самостійних криптографічних продуктів (PGP), так і як вбудовані засоби в деяких додатках (Інтернет- браузери від Microsoft і Netscape).

Нагадаємо ряд положень елементарної теорії чисел, що лежать в основі цього алгоритму.

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

Важливим фактом є те, що НСД$(a,b)$ можна виразити за допомогою рішень діофантового рівняння.

Нехай $a>b$ і $d=(a,b)$. Тоді існують цілі числа $x$, $y$, що є розв'язками рівняння $xa+yb=d$.

Розв'язки $x$, $y$, $d$ рівняння $xa+yb=d$, за умов $a>b$ і $d=(a,b)$, можна знайти з допомогою так званого розширеного алгоритму Евкліда. Очевидно, достатньо розв'язати рівняння при додатних $a$ і $b$.

Отже, для взаємно простих чисел $m$ і $b$, $m>b$ можна знайти числа $x$, $y$ такі, що $xm+yb=d=1$. Приведемо останню рівність за модулем $m$, отримаємо $yb\equiv 1 \bmod m$.

Побудоване число $y$ називається числом, оберненим до $b$ за модулем $m$ і позначається через $y\equiv b^{-1} \bmod m$. Таке число в діапазоні $(1,\ldots,m-1)$ є єдиним.

Відмітимо, що з цієї причини множина $H$ всіх найменших невід'ємних лишків, взаємно простих з $m$, при множенні на будь-який свій елемент $h$, піддається перестановці: $h\cdot H=H$.

Розглянемо степені числа $a$ за модулем $m$, де $a$ і $m$ взаємно прості.

Нехай $m=11$. Степені 1, 2, …10 числа 2 такі: 2, 4, 8, 5, 10, 9, 7, 3, 6, 1.

Аналогічно, степені числа 3 дорівнюють 3, 9, 5, 4, 1, 3, 9, 5, 4. В кожному випадку є періодичність. Найменша довжина періоду для числа $a$ за модулем $m$ називається порядком (показником, періодом) числа $a$ за модулем $m$.

Порядок числа $a$ за модулем $m$ позначається $ord_m a$.

Порядки чисел за модулем $m$ різні. Існують числа, що є порядком одночасно для всіх чисел, взаємно простих з $m$. Одне з них рівне значенню так званої функції Ейлера $\varphi(m)$, яка визначається як кількість чисел в послідовності *1, …, m*, взаємно простих з $m$.

З означення слідує, що $\varphi(p) = p - 1$ для простого числа $p$ і, крім того, $\varphi(1) = 1$.

Функція Ейлера є мультиплікативною: якщо $(a,b) = 1$, то $\varphi(ab) = \varphi(a)\varphi(b)$.

**Теорема Ейлера.** Якщо $(a, m) = 1$, то $a^{\varphi(m)} \equiv 1 \bmod m$.

*Доведення*. При $m = 2$ теорема справедлива. Нехай $m > 2$ і $a_1, \ldots, a_k$ – всі лишки за модулем $m$, взаємно прості з модулем (за означенням, $k = \varphi(m)$).

Нехай $a > 1$ – один з таких лишків. Тоді множини $\{a_1, \ldots, a_k\}$ і $\{aa_1 \bmod m, \ldots, aa_k \bmod m\}$ збігаються.

Тому $a^{\varphi(m)} a_1 \cdots a_k = a_1 \cdots a_k \equiv b \bmod m$. Помноживши обидві частини порівняння на $b^{-1} \bmod m$, одержимо $a^{\varphi(m)} \equiv 1 \bmod m$

З теореми Ейлера слідує мала теорема Ферма: $a^{p-1} \equiv 1 \bmod p$, де $p$ – просте, $a \neq 0 \bmod p$.

Випадок $a \equiv 0 \bmod p$ можна врахувати у виразі $a^p \equiv a \bmod p$.

Існує загальна формула для $\varphi(m)$ при відомому канонічному розкладі $m$ на степені простих чисел.

Нехай $m = p_1^a p_2^b \ldots p_s^t$, тоді $\varphi(m) = p_1^{a-1} p_2^{b-1} \ldots p_s^{t-1}(p_1 - 1)\ldots(p_s - 1)$.

Таким чином, якщо $n = pq$, де $p, q$ нерівні прості числа, то $\varphi(n) = (p-1)(q-1)$.

Для модулів $n$ вказаного вигляду можна показати, що якщо $e$ число, взаємно просте з $\varphi(n)$, то відображення $E_{e,n}$: $x \to x^e \bmod n$ є

взаємно однозначним на множині лишків за модулем $n$. При цьому, $ed \equiv 1 \bmod \varphi(n)$, а оберненим відображенням є $E_{d,n}: \left(x^e\right)^d \to x \bmod n$.

В криптосистемі RSA зашифрування блоку повідомлення $m$ проводиться за формулою $c = E_{e,n}(m)$, а для розшифрування застосовується операція $m = E_{d,n}(c)$. Таким чином, відкритим і секретним ключем криптосистеми є, відповідно, $(e,n)$ і $d$.

Побудова ключа $d$ при відомих $e$, $d$, $p$, $q$ легко здійсненна. За наявності відповідних параметрів функції $E_{e,n}$ і $E_{d,n}$ також легко обчислюються.

Якщо відомі $e$ і $n$, але $p$ і $q$ невідомі, то $E_{e,n}$ являє собою односторонню функцією з секретом.

Побудова $E_{d,n}$ за заданими $e$ і $n$ рівносильна розкладанню числа $n$ на співмножники. У разі, коли $p$ і $q$ − достатньо великі прості числа, то розкладання $n$ практично неможливе. Це і є причиною стійкості криптосистеми RSA.

Розглянемо принципи організації інформаційного обміну з використанням системи RSA. Спочатку абонент $i$ вибирає пару різних простих чисел $p_i$ і $q_i$. Потім він обчислює $n_i = p_i q_i$, вибирає випадковий лишок $e_i$, взаємно простий з $\varphi(n_i)$, і знаходить $d_i \equiv e_i^{-1} \bmod \varphi(n_i)$.

На загальнодоступному сервері (сайті) розміщується довідкова таблиця, яка містить відкриті ключі абонентів $(e_i, n_i)$.

Для передачі криптограми від абонента $i$ до абонента $j$ абонент $i$ розбиває відкритий текст на блоки і послідовно зашифровує їх за допомогою перетворення $E_{e_j,n_j}$. Абонент $j$ проводить розшифрування поблочно, застосовуючи відображення $E_{d_j,n_j}$.

Очевидно, для того, щоб знайти $d_i$, достатньо знання співмножників $p_i, q_i$. Для сучасних технологічних можливостей час виконання найкращих з відомих алгоритмів розкладання для значень $n \approx 2^{1024}$ дуже великий.

Розглянемо навчальний приклад, що ілюструє застосування алгоритму RSA. Зашифруємо повідомлення $m = 3, 1, 2$, що складається з трьох блоків.

1. Виберемо прості числа: $p = 3$ $q = 11$.

2. Обчислимо $n = pq = 33$ і $\phi(n) = (p-1)(q-1) = 20$.

3. Виберемо випадкове значення $e = 7$, $(e, \phi(n)) = 1$.

5. Обчислимо секретний ключ $d = 3$, $ed \equiv 1 \bmod \phi(n)$

Відкритий ключ – $(7,33)$.

Зашифруємо повідомлення: $m = 3,1,2$.

$RSA(3) = 3^7 = 2187 \equiv 9 \bmod 33$, $RSA(1) = 1^7 \equiv 1 \bmod 33$,

$RSA(2) = 2^7 = 128 \equiv 29 \bmod 33$.

Для розшифрування піднесемо кожний блок до степеня $d = 3$ за модулем 33: $9^3 = 729 \equiv 3 \bmod 33$ , $1^3 \equiv 1 \bmod 33$, $29^3 = 24389 \equiv 2 \bmod 33$.

Секретний ключ для навчальної системи легко знайти перебором. На практиці це неможливо, оскільки реальний розмір модуля (довжина бітового подання) $size(n)$ знаходиться в діапазоні від 512 до 4096 бітів.

Основні зусилля в ході практичної реалізації RSA припадають на генерацію випадкових великих простих чисел.

Є очевидний шлях рішення задачі: випадково вибрати велике непарне число $n$ і перевірити його подільність на множники в діапазоні $3, \ldots, \sqrt{n}$.

У разі невдачі вибираємо число $n + 2$ і так далі. Проте при великих $n$ такий підхід нездійсненний.

Помітимо, що з теорії чисел відомо, що ймовірність того, що навмання вибране число порядку $n$ буде простим, оцінюється як $1/\ln(n)$.

В принципі за $p$ і $q$ можна використовувати «майже» прості числа, тобто такі числа, для яких ймовірність того, що вони прості, наближається до 1. Але у випадку, якщо використано складене число, а не просте, криптостійкість RSA падає. Є непогані алгоритми, які дозволяють генерувати «майже» прості числа з досить малою ймовірністю помилки.

Інша проблема – якої довжини слід використовувати ключі?

Корисно навести довжини параметрів RSA в бітах [41], встановлені і рекомендовані французькими фахівцям для практичних додатків.

1. Співмножники RSA-модуля $n = pq$ повинні вибиратися випадково і бути однакової довжини.

2. Довжина секретного ключа $d$ повинна бути порівнянна з розміром модуля $n$.

3. Довжина відкритого ключа $e$ повинна бути строго більше 16 бітів.

4. До 2010 року дозволялося використовувати значення модуля довжиною не менше 1536 бітів, проте рекомендується – не менше 2048 бітів.

5. З 2010 року по 2020 рік дозволяється використовувати значення модуля довжиною не менше 2048 бітів.

6. Після 2020 року передбачається використовувати значення модуля довжиною не менше 4096 бітів.

Наступний важливий аспект реалізації RSA − обчислювальний. Адже доводиться використовувати апарат арифметики великих чисел.

Слід враховувати, що, в порівнянні з тим же алгоритмом DES, для RSA потрібен в тисячі і десятки тисяч раз більший час.

### 7.3 Криптосистема Ель-Гамаля

На відміну від RSA асиметрична криптосистема Ель-Гамаля заснована на проблемі дискретного логарифма [11, 16, 17].

Відповідна одностороння функція є показниковою функцією за простим модулем $p$: секретні параметри входять в показники степенів.

Піднесення числа до степеня в скінченному полі виконується легко, тоді як відновлення показника степеня за значенням функції (тобто знаходження дискретного логарифма) при великих $p$ є складною обчислювальною задачею.

Особливістю цієї криптосистеми є те, що дискретний логарифм не є односторонньою функцією з секретом. Тому для її обернення відправник повідомлення формує додаткову інформацію на основі разового ключа, яку одержувач може використовувати для читання повідомлення захищеним від просочування інформації способом.

В криптосистемі Ель-Гамаля для побудови пари асиметричних ключів вибирається велике просте число $p$ і два псевдовипадкові числа, менші $p-1$.

Одне з них, $g$, повинно бути елементом великого порядку за модулем $p$, скажімо, первісним коренем. Друге число, $x$, вибирається як секретний ключ. Вважається, що повідомлення – лишки за модулем $p$.

Відкритим ключем є трійка чисел $p, g, y = g^x \bmod p$.

Для кожного повідомлення формуються додаткові дані, що грають роль лазівки для конкретного сеансу шифрування.

Для зашифрування повідомлення $m$ вибирається псевдовипадкове число $k$ (рандомізатор, разовий ключ) з умовою $\text{НСД}(k, p-1)=1$. Рандомізатори не повинні повторюватися і повинні триматися в секреті.

Потім обчислюються числа $a = g^k \bmod p$ – лазівка і $b = y^x m \bmod p$ – шифртекст. Криптограмою є пара блоків даних $a, b$.

Для розшифрування достатньо одержати співмножник $y^k$, що можна зробити за допомогою секретного ключа, обчисливши значення $a^x = g^{kx} \bmod p$.

Дійсно, $y^k = g^{xk} \bmod p$, тому $m = a^{-x} b \bmod p$.

Таким чином, забезпечується захищений інформаційний обмін без попереднього розсилання секретних ключів.

### 7.4 Криптосистеми на основі еліптичних кривих

Для побудови СВК можуть бути використані еліптичні криві − математичні об'єкти, визначені над скінченними полями [19].

Для випадків характеристик поля $p = 2$, $p = 3$, $p > 3$ алгоритми обчислення значень, пов'язаних з реалізацією подібних СВК, істотно різні. Разом з тим, основні принципи, на основі яких побудовані СВК на еліптичних кривих, в ідейному значенні однакові.

Цікаво відзначити, що, на відміну від розглянутих раніше криптосистем, побудова відкритих параметрів для систем на еліптичних кривих є складною алгоритмічною проблемою. В даному випадку ефект від стандартизації криптоалгоритмів особливо відчутний.

Для спрощення викладу розглянемо еліптичні криві над простим полем характеристики $p > 3$.

Еліптична крива (ЕК) над полем лишків за модулем $p > 3$ безпосередньо пов'язана з розв'язками порівняння $y^2 = x^3 + ax + b \bmod p$ за так званої умови невиродженості кривої $4a^3 + 27b^2 \neq 0$.

Вказане порівняння називається рівнянням кривої в афінних координатах.

Кожний розв'язок $(x, y)$ порівняння $y^2 = x^3 + ax + b \bmod p$ називається точкою кривої $P(x, y)$.

Множину розв'язків порівняння $y^2 = x^3 + ax + b \bmod p$ можна розширити таким чином, що розширена множина стане комутативною групою $E$. Ця група називається групою точок на еліптичній кривій.

Груповий закон в групі $E$ називається додаванням. Основною причиною, що дозволяє побудувати групу $E$, є можливість побудови нових рішень рівняння кривої, виходячи з вже відомих. Виявляється, якщо дані розв'язки $P_1(x_1, y_1)$ і $P_2(x_2, y_2)$, то «практично завжди» можна знайти третій розв'язок, використовуючи знання координат перших двох.

Операція, що зіставляє двом (можливо, однаковим) точкам їх «суму», в афінних координатах записується у вигляді дробових виразів, тому при обчисленнях може виникнути особливість, якщо у відповідному знаменнику з'явиться нульове значення (за модулем $p$).

Очевидно, це єдина ситуація, коли виникає особливість.

Отже, їй можна зіставити деяке позначення ($O$) і розширити множину розв'язків рівняння кривої, додавши символ $O$, імітуючи тим самим існування додаткового елемента, званого нескінченно віддаленою точкою.

Можна показати, що якщо для операції «+» вважати $O$ нейтральним елементом, то розширена множина точок кривої перетворюється на комутативну групу, а сама операція – в груповий закон.

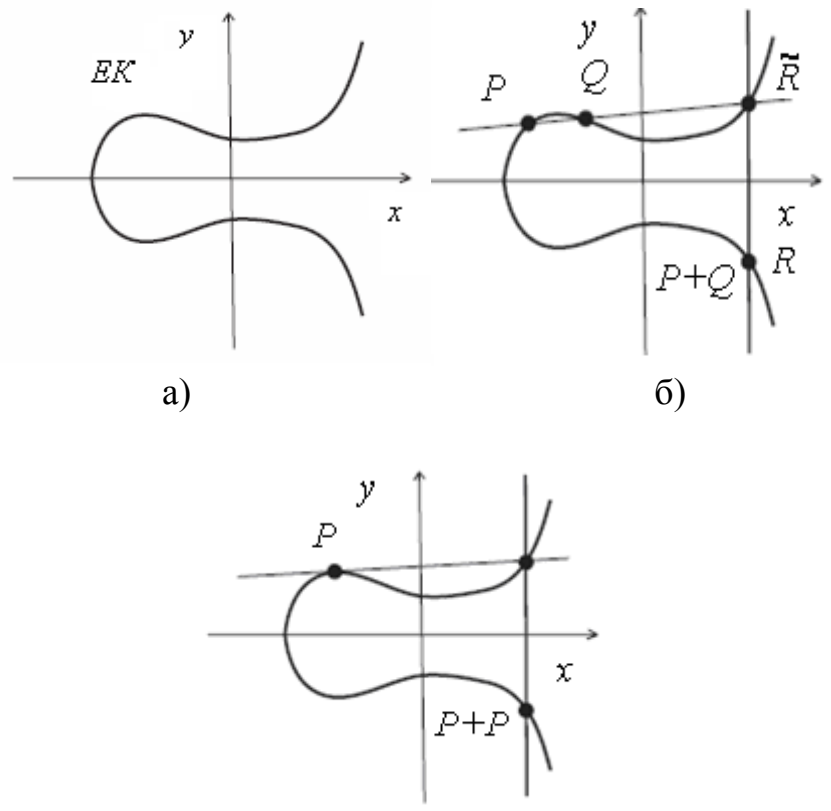
Насправді, для аналітичного опису розширеної множини точок кривої слід використовувати так звані проективні координати, в яких точки кривої будуть задаватися не парою, а трійкою чисел.

Виявляється, в цьому випадку всі точки кривої будуть рівноправними і будуть підкорятися більш загальному груповому закону.

Груповий закон в афінних координатах був відкритий при дослідженнях еліптичних кривих на звичайному полі дійсних чисел, що «підказало» вид відповідних перетворень над скінченними полями.

Додавання двох точок ЕК над полем дійсних чисел можна сформулювати у вигляді геометричної побудови, що зв'язує координати точок-доданків з координатами точки-суми (рис. 7.2).

Для цього слід враховувати, що якщо пряма перетинає ЕК в двох точках розширеної кривої, то вона перетинає криву в третій точці (точка дотику вважається подвійною точкою).




а) б)

в)

Рисунок 7.2 - Графік еліптичної кривої $y^2 = x^3 - 3x + 4$ (а), додавання двох точок (б), кратна точка (в)

Для складання точок $P$ і $Q$ кривої (рис. 7.2, б) проводимо через $P$ і $Q$ пряму. Вона перетне ЕК в деякій третій точці $\tilde{R}$. Проведемо через $\tilde{R}$ пряму, паралельну осі ординат, яка перетне ЕК в деякій точці $R$, яку приймемо за суму точок кривої: $R = P + Q$.

Якщо $P = Q$, то точка $\tilde{R}$ є перетином кривої і дотичної до кривої в точці $P$. У відповідних випадках вважаємо, що прямі, паралельні осі ординат, проходять через нескінченно віддалену точку $O$.

Вказана геометрична побудова називається методом січних і дотичних.

Нехай $P = (u, v)$. Позначимо $\tilde{P} = (u, -v)$. Очевидно, що при нашій побудові $R = (x, y)$, а $\tilde{R} = (x, -y)$. Розглянемо точку $R + \tilde{P}$. Користуючись геометричною побудовою, легко прослідкувати, що $R + \tilde{P} = Q$. Іншими словами, додавання до точки $R = P + Q$ точки $\tilde{P}$ еквівалентно «відніманню» точки $P$ з точки $R$. Таким чином, для визначення операції віднімання, оберненої додаванню, приймемо $(-P(u, v)) = P(u, -v)$.

Нехай $G_1 = G$, $G_2 = G + G$ і т. д. Виходячи з точки $G$, можна побудувати послідовність точок $G_1, \ldots, G_{n-1}, G_n = O$ деякої довжини $n$.

Якщо записувати подібне $k$-кратне додавання на кривій у вигляді $kG$, прийнявши $0G = O$, то, очевидно, коефіцієнт $k$ можна приводити за модулем $n$ і розглядати вирази вигляду $uP + vQ$ і $u(vP)$. Операція $kG$ називається скалярним множенням точки на $k$. Найменше ціле $n$, для якого $nG = O$, називається порядком точки $G$.

Використовуючи аналітичну геометрію, можна показати, що груповий закон $P_1(x_1, y_1) + P_2(x_2, y_2) = P_3(x_3, y_3)$ відповідає деяким правилам, які, у разі скінченного поля характеристики $p > 3$, зводяться до нижчевикладеного.

1. $x_3 = \lambda^2 - x_1 - x_2 \bmod p$, $y_3 = \lambda(x_1 - x_3) \bmod p$, де

$\lambda = \dfrac{y_2 - y_1}{x_2 - x_1} \bmod p$, якщо $P_1 \neq P_2$ і $\lambda = \dfrac{3x_1^2 + a}{2y_1} \bmod p$, якщо $P_1 = P_2$.

2. Якщо знаменник $\lambda$ перетворюється в нуль, то $P_1(x_1, y_1) + P_2(x_2, y_2) = O$.

3. Операція, обернена додаванню: $-P(x, y) = P(x, -y)$

4. Для будь-якої точки $P$ розширеної кривої: $O \pm P = P$.

Як правило, при побудові СВК на еліптичній кривій використовується точка $G$ великого простого порядку $n$. Всі операції в СВК здійснюються над точками вигляду $kG$, які утворюють циклічну підгрупу групи $E$.

Покажемо, як можна побудувати аналог алгоритму Діффі-Хеллмана на еліптичній кривій. Вихідними параметрами є сама крива $E$ над скінченним полем і точка $G$ великого простого порядку $n$.

Користувачі криптосистеми незалежно генерують секретні великі числа $m_A$ і $m_B$, які є параметрами для формування спільного секретного ключа. На їх основі обчислюються відкриті ключі $Q_A = m_A G$ і $Q_B = m_B G$.

Провівши обмін відкритими ключами, кожний з абонентів може сформувати загальний ключ $K_{AB} = m_A m_B G = m_A Q_B = m_B Q_A$.

Стійкість СВК заснована на складності задачі дискретного логарифмування на кривій. Ця задача полягає в знаходженні скалярного множника із співвідношення $P = kG$ і, в загальному випадку, є обчислювано недоступною.

Найбільш трудомістким етапом обчислення відкритих параметрів еліптичної кривої є визначення простого числа $n$, яке є дільником кількості $\#E$ елементів в групі $E$.

Як правило, обчислення $n$ зводиться до обчислення $\#E$ з його подальшою факторизацією. Для обчислення $\#E$, взагалі кажучи, використовуються складні алгоритми. Відомі прості обмеження на коефіцієнти рівняння ЕК, при яких або $\#E = n$, або факторизація $\#E$ здійсненна.

Теоретично група $E$ складається з не більш, ніж двох циклічних підгруп. Зазвичай параметри кривих такі, що порядок $n_2$ однієї з підгуп дуже малий і визначається порівняно легко. Після чого $n$ обчислюється у вигляді $n = \#E : n_1$.

Спільні алгоритми обчислення $\#E$ існують, але є складними і трудомісткими.

Для числа $\#E$ є оцінки. Наприклад, нерівність Хассе: $q + 1 - 2\sqrt{q} \le \#E \le q + 1 + 2\sqrt{q}$, де $q = p^r$ – кількість елементів в полі характеристики $p$. Таким чином, $\#E = q + 1 - t$, де $|t| \le 2\sqrt{q}$.

Для деяких кривих число $\#E$ можна обчислити теоретично.

Криві, для яких $\#E = p$, називаються аномальними, а криві, для яких $t = 0(p)$, – суперсингулярними.

Для кривих цих типів відомі ефективні методи дискретного логарифмування.

Проте існує клас криптопротоколів, в яких істотно використовуються властивості саме суперсингулярних еліптичних кривих.

Цікаво відзначити, що багато криптографічних систем, розроблених на основі системи RSA, породжують аналоги на еліптичних кривих над кільцем лишків $Z/nZ$ для складеного числа, де $p,q$ — різні прості числа.

Еліптична крива $E_n$ над $Z/nZ$ задається тими ж алгебраїчними рівняннями, що і у разі простого модуля. Наприклад, крива задається як множина розв'язків основного порівняння $y^2 = x^3 + ax + b \bmod n$, при НСД $\left(4a^3 + 27b^2, n\right) = 1$ з нескінченно віддаленою точкою $O$.

Операції на $E_n$ виконуються за формулами, відповідними методу січних і дотичних. На жаль, такий об'єкт не є групою.

Тому замість $E_n$ розглядається так звана пряма сума кривих $\tilde{E}_n = E_p + E_q$: інший об'єкт, який «мало» відрізняється від $E_n$.

Кожний елемент $\tilde{E}_n$ – чотиривимірний вектор, що складається з двох точок кривих $E_p$ і $E_q$.

Якщо пара $(x, y)$ задовольняє основне рівняння для $E_n$, то пари лишків $(x \bmod p, y \bmod p)$ і $(x \bmod q, y \bmod q)$ автоматично задовольняють те ж рівняння за модулями $p$ і $q$ відповідно.

Нагадаємо, що Китайською теоремою про лишки називається наведене нижче твердження.

Нехай числа $m_1, m_2, \ldots, m_k$ попарно взаємно прості і $M = m_1 m_2 \cdots m_k$.

Тоді єдиний за модулем $M$ розв'язок системи порівнянь $x = c_i \bmod m_i$, $i = 1, \ldots, k$ має вигляд: $x = \sum_{i=1}^{k} c_i M_i N_i \bmod M$, де $M_i = m_1 m_2 \cdots m_{i-1} m_{i+1} \cdots m_k$, $N_i = M_i^{-1} \bmod m_i$.

Дійсно, у вказаному виразі для $x$ доданок $c_i M_i N_i$ порівнянний з $c_i$ за модулем $m_i$, а всі інші порівнянні за цим модулем з нулем.

Враховуючи Китайську теорему про лишки, $x$ і $y$ однозначно відновлюються за значеннями $(x \bmod p, x \bmod q)$ і $(y \bmod p, y \bmod q)$.

Проте, разом з нескінченно віддаленою точкою $O$, група $\tilde{E}_n$ містить сукупність точок виду $\{(O_p, y), (x, O_q)\}$. Тому $\tilde{E}_n$ складається з точок кривої $E_n$ і деякої множини особливих точок.

Наведемо основні властивості $\tilde{E}_n$, близькі до властивостей $E_n$, що дозволяє будувати RSA-подібні криптосистеми.

1. Метод січних і дотичних у разі, коли він визначений для $E_n$, збігається з груповою операцією в $\tilde{E}_n$.

Таким чином, якщо виконується додавання конкретних точок в $E_n$, то воно виконується без факторизації $n$.

2. При великих $p$ і $q$ з ймовірністю, близькою до одиниці, множення точки на скаляр в $\tilde{E}_n$ поводиться як аналогічна операція в $E_n$.

Суть цього твердження така.

Приймемо $N = \text{НСД}(\#E_p, \#E_q)$. Тоді $(kN+1)P = P$ для переважного числа точок і будь-яких цілих $k$.

Розглянемо аналог системи RSA з використанням еліптичної кривої. Відкритим ключем є пара $N, e$, де $N = \text{НСД}(\#E_p, \#E_q)$, $(N, e) = 1$, секретним – число $d$ таке, що $ed = 1 \bmod N$.

Шифртекст повідомлення $m$, яке зіставляється наперед вибраним способом деякій точці $M$ кривої $E_n$, Відправник одержує з допомогою скалярного множення у вигляді $C = eM$. Одержувач, знаючи число $d$, відновлює $M = deM = dC$.

Виявляється, однак, що зіставлення $m \Leftrightarrow M$ може вимагати факторизацію $n$, тобто вказана схема шифрування, взагалі кажучи, не є коректною. Проте існує ряд більш досконалих RSA-подібних криптосистем, наприклад, криптосистема Демітко (для кривих загального вигляду), або криптосистема Кувакадо-Кояма для кривих вигляду $y^2 = x^3 + b \bmod n$.

**Питання до розділу 7**

1. Що таке однонаправлена функція з секретом (лазівкою)?
2. З якою метою використовується відкритий ключ в асиметричній криптосистемі?
3. Які основні вимоги до відкритого ключа і шифрперетворення в асиметричній криптосистемі?
4. Які функції, що використовуються в криптографії, мають властивості односторонніх?

5. Які основні властивості функції Ейлера?

6. Яким чином секретний ключ обчислюється через відкритий при формуванні криптосистеми RSA?

7. Що таке порядок числа за заданим модулем?

8. Для чого використовується рандомізатор в криптосистемі Ель-Гамаля?

9. Які довжини параметрів криптосистеми RSA вважаються безпечними в наш час?

10. Яка множина точок еліптичних кривих над скінченним полем використовується для криптографічних перетворень?

11. Що розуміється під розширенням множини точок еліптичної кривої (ЕК) до абелевої групи?

12. В чому полягає операція скалярного множення точки ЕК на число?

13. Що означає твердження, що точка ЕК має порядок $n$?

14. В чому полягає задача дискретного логарифмування на ЕК?

15. Який етап є найбільш трудомістким при обчисленні відкритих параметрів еліптичної кривої?

16. Чому точки еліптичної кривої над кільцем лишків по складеним модулем не утворюють групу відносностандартної операції додавання точок методом січних і дотичних?

17. Чи можна побудувати RSA-подібні криптосистеми на еліптичних кривих над кільцем за модулем $N = pq$?

# РОЗДІЛ 8 ТЕСТУВАННЯ ЧИСЕЛ НА ПРОСТОТУ І ВИБІР ПАРАМЕТРІВ RSA

При побудові асиметричних криптосистем, а також модифікації з параметрів в ході експлуатації виникає необхідність побудови надвеликих псевдовипадкових простих чисел, що мають ті або інші специфічні властивості.

У багатьох випадках, наприклад, у випадку RSA, великі прості числа є ключовими параметрами.

Відповідні обчислювальні процедури включають в себе алгоритми, що реалізують етап перевірки чисел на простоту. В літературі і криптографічній практиці подібні алгоритми носять назву тестів.

В основі тестів лежать так звані критерії простоти [16, 17, 18]. Існує два типи критеріїв простоти: детерміновані і ймовірнісні.

Детерміновані тести дозволяють довести, що число, яке тестується, – просте. Практично застосовувані детерміновані тести здатні дати позитивну відповідь не для кожного простого числа, оскільки використовують лише достатні умови простоти.

Детерміновані тести більш корисні, коли необхідно побудувати випадкове велике просте число, а не перевірити простоту, скажімо, деякого єдиного числа.

Детермінований тест використовується, наприклад, в процедурах обчислення несекретних параметрів цифрового підпису типу Ель-Гамаля, встановлених ГОСТ 34.310. Цей тест заснований на викладеній нижче теоремі.

**Теорема. (Демітко).** Нехай $n = qR + 1$, де $q$ – просте, $R$ – парне і $R < 4(q+1)$.

Якщо існує $a$ таке, що $a^{n-1} \equiv 1 \bmod n$ і $a^{(n-1)/q} \neq 1 \bmod n$, то число $n$ – просте.

На відміну від детермінованих, імовірнісні тести можна ефективно використовувати для тестування окремих чисел, проте їх результати, з деякою ймовірністю, можуть бути недостовірними.

На щастя, ціною кількості повторень тесту з модифікованими вихідними даними ймовірність помилки можна зробити як завгодно малою.

### 8.1 Тест на основі малої теореми Ферма

При побудові ймовірнісних критеріїв простоти виникає ряд типових питань, які зручно розглянути на прикладі, за який виберемо тест на основі малої теореми Ферма. Як відомо, ця теорема стверджує, що якщо $n$ – просте, то для всіх $a$, взаємно простих з $n$, виконується умова (порівняння Ферма): $a^{n-1} \equiv 1 \bmod n$.

Таким чином, якщо порівняння Ферма не виконано хоча б для одного числа з множини $\{2,\ldots,n-1\}$, то $a$ – складене.

Тест на основі малої теореми Ферма пояснюється нижче.

Псевдовипадково вибираємо лишок $a \in \{2,\ldots,n-1\}$ і перевіряємо умову $(a,n)=1$. Якщо ця умова не виконана, значить, $n$ – складене. Перевіряємо порівняння Ферма. Якщо воно не виконується, то число $n$ – складене. Інакше, повторюємо тест для іншого значення $a$.

Очевидно, основне питання полягає в тому, щоб оцінити, якою мірою тест є ефективним. Перш за все, слід з'ясувати, чи існують складені числа $n$, для яких при деякому $a$ виконується умова малої теореми Ферма. Виявляється, це так, оскільки існують контрприклади: $2^{340} = \left(2^{10}\right)^{34} \equiv 1 \bmod 341$, $7^{24} \equiv 1 \bmod 25$.

#### 8.1.1 Основні властивості псевдопростих чисел

Назвемо складене непарне число $n$ псевдопростим за основою $a$, якщо пара чисел задовольняє порівняння Ферма $a^{n-1} \equiv 1 \bmod n$.

Виявляється, можна показати, що для будь-кого $a$ існує нескінченно багато псевдопростих чисел за основою $a$.

Основні властивості псевдопростих чисел такі.

**Теорема** [17]. Нехай $n$ – непарне складене число. Тоді:

1) $n$ - псевдопросте за основою $a$ в тому і лише тому випадку, коли $(a,n)=1$ і $ord_n a \mid n-1$;

2) якщо $n$ – псевдопросте за основою $a$ і $b$, то $n$ – псевдопросте за основами $ab$ і $ab^{-1} \bmod n$;

3) множина $F_n = \{a \in Z_n : a^{n-1} \equiv 1 \bmod n\}$ утворює мультиплікативну підгрупу в мультиплікативній групі $Z_n^*$ обернених елементів кільця лишків за модулем $n$;

4) якщо $n$ не є псевдопростим за основою $a$ хоча б для одного числа $a$, то $|F_n| \le (1/2)|Z_n^*|$ (тут через $|F|$ позначено кількість елементів, що належать множині $F$).

Помітимо, що $|Z_n^*| = \varphi(n) \le n-1$. Таким чином, якщо тест Ферма виявляє складене $n$ при одній основі $a$, то існує не менше $(n-1)/2$ основ з аналогічною властивістю.

Дійсно, властивість 1) виходить з визначення $ord_n a$. Властивості 2) і 3) виходять з правила почленного перемножування частин порівнянь і перевірки порівняння Ферма для основи $b^{-1}$.

Доведемо властивість 4). Нехай $F_n = \{a_1, \ldots, a_s\}$ і $a$ – основа, за якою $n$ не є псевдопростим. Тоді для будь-якого $i$ пари чисел $aa_i, n$ не задовольняють порівняння Ферма. Тому кількість основ, для яких $n$ не є псевдопростим, не менше, ніж кількість елементів в $F_n$.

Отже, якщо $|F_n| > (1/2)|Z_n^*|$, то спільне число елементів у $Z_n^*$ виявиться більше $n$, що неможливо.

Таким чином, можна сказати, що якщо існує (навіть не відома нам) основа, за якою $n$ не є псевдопростим, то, при повторенні тесту Ферма $k$ раз, ймовірність $k$-кратного вибору основ з множини $F$ не перевищує $(1/2)^k$. В цьому випадку ймовірність помилки тесту наближається до нуля зі збільшенням $k$.

Проблема може виникнути лише в тому випадку, якщо $n$ є псевдопростим для всіх (ненульових) основ. Виявляється, такі числа існують. Вони називаються числами Кармайкла. Наприклад, числом Кармайкла є число $n = 561 = 3 \cdot 11 \cdot 17$.

### 8.1.2 Властивості чисел Кармайкла

Властивості чисел Кармайкла описуються такою теоремою.

**Теорема (Кармайкл)**. Нехай $n$ – непарне складене число. Тоді якщо:

1) $p^2 | n$, $p > 1$, то $n$ не є числом Кармайкла;

2) $n = p_1 p_2 \cdots p_k$, $p_i \neq p_j$ для $(i \neq j)$, то $n$ – число Кармайкла в тому і лише тому випадку, коли $\forall i \ (p_i - 1) | (n-1)$;

3) $n = p_1 p_2 \cdots p_k$, $p_i \neq p_j$ для $(i \neq j)$ – число Кармайкла, то $k \geq 3$.

Числа Кармайкла є достатньо рідкісними. В межах до 100000 існує лише 16 чисел Кармайкла: 561, 1105, 1729, 2465, 2821, 6601, 8911, 10585, 15841, 29341, 41041, 46657, 52633, 62745, 63973, 75361.

### 8.2 Тест Соловея-Штрассена і Ейлерові псевдопрості числа

З попереднього випливає, що при тестуванні чисел на простоту за допомогою імовірнісного тесту, заснованого на малій теоремі Ферма, може виникнути ситуація, коли ймовірність помилки не знижується з кількістю повторень тесту.

В подібному випадку згадана ймовірність рівна одиниці, і в результаті тестування може бути прийнято неправильне рішення.

В зв'язку з цим розроблені і застосовуються на практиці імовірнісні тести, вільні від вказаного недоліку.

Прикладами таких тестів є тест Соловея-Штрассена і тест Рабіна-Міллера [18] перевірки чисел на простоту.

В тесті Соловея-Штрассена використовуються властивості так званих символів Лежандра і Якобі, пов'язаних з розв’язністю двочленних квадратичних порівнянь. Нагадаємо коротко їх властивості.

Двочленним квадратичним порівнянням називається порівняння виду $x^2 = a \bmod n$, де $x$ - невідомий лишок.

Ціле число $a$ називається квадратичним лишком за модулем $n$, якщо порівняння $x^2 = a \bmod n$ має розв’язки. Якщо порівняння має розв’язки, то для складеного непарного модуля кількість розв’язків, як правило, більша двох.

В загальному випадку, не тільки розв’язання квадратичних порівнянь, але навіть питання про розв’язність двочленного квадратичного порівняння за складеним модулем, факторизація якого невідома, є алгоритмічною проблемою, на складності якої заснований ряд криптопротоколів і криптосистем.

В той же час для модулів, що є простими числами, задача легко піддається аналізу.

Існує ефективний алгоритм для визначення є дане число квадратичним лишком за простим непарним модулем $p$, чи ні. Цей алгоритм дозволяє обчислити, практично вручну, так званий символ Лежандра $\left(\frac{a}{p}\right)$, значення якого при $a \neq 0$ рівно 1, якщо $a$ - квадратичний лишок, і (-1) – в іншому випадку. Для $a \equiv 0 \bmod p$ вважається $\left(\frac{a}{p}\right) = 0$.

З появою комп'ютерів значення $\left(\frac{a}{p}\right)$ звичайно обчислюється, виходячи із співвідношення: $\left(\frac{a}{p}\right) = a^{(p-1)/2} \bmod p$, яке є важливою властивістю символу Лежандра.

Якщо непарне число $n$ факторизовано: $n = \prod_{i=1}^{k} p_i^{a_i}$, то розв'язання порівняння $x^2 = a \bmod n$ еквівалентне розв'язанню всіх порівнянь виду $x^2 = a \bmod p_i^{a_i}$.

Помітимо, що, у свою чергу, порівняння $x^2 = a \bmod p_i^{a_i}$ має корені тоді і тільки тоді, коли $\left(\frac{a}{p_i}\right) = 1$, а також, добуток двох квадратичних нелишків є квадратичним лишком за простим модулем.

Тому значення $\left(\frac{a}{p}\right)$ можна обчислити через величини $\left(\frac{a}{p_i}\right)$.

З символом Лежандра тісно пов'язаний символ Якобі $\left(\frac{x}{n}\right)$ числа $x$ за модулем $n$. Символ Якобі визначається для непарних, взаємно простих чисел як добуток значень символів Лежандра: $\left(\frac{x}{n}\right) = \left(\frac{x}{p_1}\right)^{a_1} \cdots \left(\frac{x}{p_k}\right)^{a_k}$.

Він має практично всі ті ж властивості, що і символ Лежандра, але за значенням символу Якобі, рівним одиниці, не можна стверджувати, що відповідний лишок $x$ – квадратичний. Для квадратичного лишку, проте,

символ Якобі рівний одиниці. Отже, якщо $\left(\frac{x}{n}\right)=-1$, то $x$ - квадратичний нелишок за модулем $n$.

Ця особливість пов'язана з тим, що співвідношення $\left(\frac{a}{p}\right)=a^{(p-1)/2} \bmod p$ не обов'язково виконується для символу Якобі, тобто, коли число $p$ (модуль) не є простим. Для нас символ Якобі важливий тому, що його можна обчислити без факторизації модуля.

На основі нижченаведеної теореми, в тесті Соловея-Штрассена використовується критерій Ейлера для визначення значення символу Лежандра (квадратичного характеру числа за простим модулем). В самому тесті при тестуванні числа $n$ обчислюється символ Якобі $\left(\frac{a}{n}\right)$.

**Теорема** [17]. Непарне ціле число $n>1$ є простим тоді і тільки тоді, коли для всіх чисел $a$: $1\le a\le n-1$ виконується співвідношення Ейлера: $a^{\frac{n-1}{2}}=\left(\frac{a}{n}\right)\bmod n$.

Складене число $n$, що задовольняє співвідношення Ейлера, називається ейлеровим псевдопростим за основою $a$.

З вказаної теореми виходить, що складених чисел, які були б ейлеровими псевдопростими за будь-якою основою, не існує.

Отже, ми можемо запропонувати такий тест, аналогічний тесту Ферма.

Псевдовипадково вибираємо лишок $a\in\{2,\ldots,n-1\}$ і перевіряємо умову $(a,n)=1$. Якщо умова не виконана, значить, $n$ – складене.

Перевіряємо співвідношення Ейлера. Якщо воно не виконується, то число $n$ – складене. Інакше, повторюємо тест для іншого значення $a$.

Якщо ми могли б перевірити співвідношення Ейлера для всіх значень $a$, то ми змогли б точно визначити, чи є число $n$ простим.

Але для великих $n$ це неможливо. Тому необхідно оцінити, як веде себе ймовірність помилки при збільшенні числа $k$ повторень тесту.

Це можна зробити, виходячи з твердження, аналогічного тому, яке ми розглядали при аналізі властивостей псевдопростих чисел.

Цікаво, що ейлерові псевдопрості є псевдопростими числами.

**Теорема** [17]. Нехай $n$ – непарне складене число. Тоді:

а) якщо $n$ – ейлерове псевдопросте за основою $a$, то воно – псевдопросте за основою $a$;

б) якщо $n$ – ейлерове псевдопросте за основами $a$ і $b$, то $n$ – ейлерове псевдопросте за основами $ab$ і $ab^{-1} \bmod n$;

в) множина $E_n = \left\{ a \in Z_n : a^{(n-1)/2} = \left( \frac{a}{n} \right) \bmod n \right\}$ є підгрупою групи $F_n = \left\{ a \in Z_n : a^{n-1} = 1 \bmod n \right\}$;

г) якщо $n$ не є ейлеровим псевдопростим за основою $a$ хоча б для одного числа $a$, то $|E_n| \le (1/2)|Z_n^*|$.

Таким чином, при повторенні тесту Соловея-Штрассена $k$ раз ймовірність невідбракування складеного числа не перевершує $(1/2)^k$.

### 8.3 Тест Рабіна-Міллера і сильні псевдопрості числа

Ще більш ефективним ймовірнісним тестом є тест Рабіна-Міллера, в якому використовується критерій, в кінцевому рахунку, оснований на факті, що для простого модуля квадратними коренями з одиниці є лише числа $\pm 1$, а для складеного непарного модуля $n = uv$, $(u,v) = 1$, число таких коренів більше двох.

Нехай $n$ – непарне натуральне число. Тоді можна записати $n - 1 = 2^s t$, де $t$ – непарне і $s \ge 1$.

Якщо число $n$ – просте, то $a^{n-1} = 1 \bmod n$, при $(a,n) = 1$. Тому квадратні корені з одиниці мають вигляд: $a^{(n-1)/2} = \pm 1 \bmod n$, де показник рівний $2^{s-1} t$.

Це означає, що в послідовності $a^t, a^{2t}, \ldots, a^{2^{s-1} t}$ лишків за простим модулем, які є послідовними квадратами числа $a^t$, або з'явиться $(-1) \bmod n$, або всі ці лишки порівнянні з одиницею, тобто $a^t = 1 \bmod n$. Помітимо, що при простому $n$ лівіше $(-1) \bmod n$ можуть розташовуватися лише лишки не рівні $(\pm 1) \bmod n$.

Якщо $n$ – складене, то можливі й інші варіанти, оскільки в цьому випадку крім $\pm 1$ існують інші корені з одиниці за модулем $n$.

Заснований на даному зауваженні тест Рабіна-Міллера полягає в тому, що:

1) псевдовипадково вибираємо лишок $a \in \{2,\ldots,n-1\}$ і перевіряємо умову $(a,n)=1$. Якщо умова не виконана, значить, $n$ – складене і робота закінчена;

2) обчислюємо $a^t \bmod n$. Якщо $a^t = \pm 1 \bmod n$, то не виключено, що число $n$ – просте і необхідно перейти на початок, щоб повторити тест для іншої основи;

3) обчислюємо послідовно лишки чисел $a^{2t}$, $\ldots a^{2^{s-1}t}$ за модулем $n$, поки не з'явиться $(-1)$, або не вичерпається список;

4) якщо $(-1)$ знайдено в списку, то не виключено, що число $n$ – просте і необхідно перейти на початок, щоб повторити тест для іншої основи;

5) якщо жодне число із списку не порівнянно з $(-1)$, то число $n$ – складене і необхідно закінчити роботу.

Як і для інших імовірнісних тестів, існують складені числа $n$, які, для відповідних основ $a$, проходять даний тест.

Назвемо число $n = 2^s t + 1$, де $s \geq 1$, $t$ – непарне, сильним псевдопростим за основою $a \neq 1 \bmod n$, $(a,n)=1$, якщо виконується одна з двох умов: $a^t = \pm 1 \bmod n$, або в послідовності $a^{2t}, \ldots a^{2^{s-1}t}$ існує число, порівнянне з $-1$ за модулем $n$.

Виявляється, можна показати, що для будь-якого $a$, $(a,n)=1$, існує нескінченно багато сильних псевдопростих чисел $n$ за основою $a$.

Приклади: $a=7,\ n=25$; $a=5,\ n=781$.

Можна довести такі основні властивості сильних псевдопростих чисел [16]:

1) число $n$, сильне псевдопросте за основою $a$, є ейлеровим псевдопростим за тією ж основою;

2) якщо непарне складене число $n$ є сильним псевдопростим за основою $a$, то загальна кількість основ, за якою це число є сильним псевдопростим, не перевищує $(n-1)/4$.

Тому можна стверджувати, що при повторенні випробувань тесту Рабіна-Міллера $k$ раз ймовірність невідбракування складеного числа $\leq (1/4)^k$.

Крім того, виявляється, кількість повторень тесту, достатню для практичних додатків, можна обмежити величиною $2\log_2^2 n$.

Цікаво відзначити, що простоту невеликих простих чисел можна довести, використовуючи декілька раніше вказаних основ.

Приклади [17]: якщо $n < 1373653$ – сильне псевдопросте за основами 2 і 3, то $n$ – просте; якщо $n < 341550071728321$ – сильне псевдопросте за основами, 3, 5, 7, 11, 13, 17, то $n$ – просте.

### 8.4 Загальні вимоги до вибору параметрів RSA

Коректність параметрів RSA пов'язана з оцінюванням стійкості системи і може бути визначена лише з погляду практичної стійкості.

Отже, коректні параметри повинні бути побудовані так, щоб мінімізувати шкоду від відомих підходів до ослаблення криптосистеми.

Виходячи з цих міркувань, розглянемо найбільш загальні вимоги до вибору параметрів $p, q, e, d$ криптосистеми RSA [17,18].

Перш за все, слід враховувати, що слабкість одного з параметрів практично не компенсується посиленням властивостей інших параметрів.

Очевидно, число $n = pq$ повинно бути великим. Числа $p$ і $q$ не повинні міститися в списках відомих великих простих чисел, не повинні бути дуже близькі один до одного, або істотно розрізнятися за величиною.

Вони не повинні бути побудованими за детермінованими алгоритмами з невеликим числом відомих варіантів початкових параметрів або містити закономірності в двійковому записі.

Загалом, $p$ і $q$ не повинні відрізнятися від типових представників випадкових простих чисел. Аналогічні властивості повинні мати параметри $e$ і $d$.

Наприклад, якщо секретний ключ $d$ містить в двійковому записі невелику кількість одиниць, то номери місць цих одиниць легко визначити перебором.

Можна довести, що при відомому $d$ існує можливість факторизації модуля.

Відомо, що для читання повідомлень, зашифрованих криптосистемою RSA, достатньо знання деякого кратного функції Ейлера

від модуля, оскільки в цьому випадку можна обчислити ключ, криптоеквівалентний ключу $d$.

Помітимо також, що за наявності $ord_n a$ легко одержати $a$ з порівняння $a^e = c \bmod n$. Достатньо піднести $c$ до степеня $h$, що задовольняє співвідношення $eh = 1 \bmod ord_n a$.

Далі. Для будь-якого $a$, взаємно простого з $n = pq$, $ord_p a$ ділить $p-1$, а $ord_q a$ ділить $q-1$. Тому $ord_n a$ ділить $G = \text{НСД}((p-1),(q-1))$.

Отже, для побудови криптосистеми, замість визначення $d$ з порівняння $ed = 1 \bmod \varphi(n)$, можна скористатися розв'язком порівняння $ed_1 = 1 \bmod G$.

Нехай $g = \text{НСД}((p-1),(q-1))$. Тоді $gG = \varphi(n)$.

Очевидно, із співвідношення $ed = 1 \bmod \varphi(n)$ виходить $ed = 1 \bmod G$, тому $d = d_1 (\bmod G)$ і $d \neq d_1 (\bmod \varphi(n))$.

Ці умови задовольняють ключі $d_1$, $d_1 + G$, $d_1 + 2G$, …, $d_1 + (g-1)G$, криптоеквівалентні, таким чином, ключу $d$. Отже, чим більше $\text{НСД}((p-1),(q-1))$, тим більше криптоеквівалентних ключів, тим гірше для криптосистеми.

Очевидно, в найкращому можливому випадку, $\text{НСД}((p-1),(q-1)) = 2$, при цьому, $p = 2s+1$, $q = 2t+1$, де $(s,t) = 1$, скажімо, $s$ і $t$ – прості.

Помітимо, до речі, що невідомо, чи є множина простих чисел виду $s = (p-1)/2$ нескінченною.

Виявляється, щоб уникнути можливості застосування більшості часткових методів факторизації для дешифрування криптосистеми RSA [18], достатньо вимагати, щоб числа $p_1 = (p-1)/2$, $p_2 = (p+1)/2$, $q_1 = (p-1)/2$ і $q_2 = (p+1)/2$ не розкладалися в добуток степенів невеликих простих чисел, тобто щоб вони містили в розкладі велике просте число.

Відповідні вимоги, в найбільш сильній формі, полягають в тому, щоб числа $p_1$, $p_2$, $q_1$, $q_2$ були простими, більш того, вимагається, щоб в розкладі як $p_1 - 1$, так і $q_1 - 1$ було велике просте число.

На практиці, при побудові відповідного цим вимогам простого числа, скажімо $p$, достатньо, щоб існував достатньо великий простий дільник числа $p-1$. Очевидно, такий дільник має вигляд $r=(p-1)/2j$.

Таким чином, необхідно виділити деякий специфічний клас простих чисел.

*Визначення*. Просте число $p$ називається сильним простим, якщо виконуються умови:

$$p \equiv 1 \bmod r,\ p \equiv -1 \bmod s,\ r \equiv 1 \bmod t,$$

де $r, s, t$ – великі прості числа.

Оскільки числа $p, r, s, t$ – непарні, то вони подаються у вигляді $p=1+2jr$, $p=-1+2ks$, $r=1+2lt$. Крім того, для наших цілей чим менші числа $j, k, l$, тим краще.

Для побудови сильних простих чисел застосовується так званий метод Гордона.

В цьому методі здійснюється перегляд околів деяких псевдовипадкових чисел з метою виявлення простих чисел, що задовольняють специфічні умови.

При цьому неодноразово використовуються імовірнісні процедури побудови проміжних даних, а також застосовується тестування чисел на простоту за допомогою тесту Рабіна-Міллера.

### 8.5 Метод Гордона побудови сильних простих чисел

Суть методу Гордона побудови сильного простого числа $p$ така.

1. Будуємо випадкове просте число $s$, виходячи з заздалегідь вибраної для нього розрядності $h$. Для цього вибираємо псевдовипадково число $x$ розрядності $h$ і за допомогою методу пробних ділень залишаємо в проміжку $x, x+\log_2 x$ числа, що не мають малих дільників. Серед чисел, що залишилися, за допомогою тестів на простоту, визначаємо просте число $s$.

2. Будуємо просте число $t \neq s$ аналогічно побудові числа $s$.

3. За допомогою методу пробних ділень і тестів на простоту аналогічно пункту 1 будуємо просте число $r=1+2lt$, перебираючи $l$ в проміжку $[1, \log_2 t]$.

4. Обчислюємо $u = u(r,s) = \left(s^{r-1} - r^{s-1}\right) \bmod rs$. Щоб не підносити до степеня, це зручно зробити за допомогою китайської теореми про лишки, оскільки $u \equiv 1 \bmod r$ і $u \equiv -1 \bmod s$.

Вимагатимемо, щоб шукане число $p$ задовольняло ті ж умови: $p = u \equiv 1 \bmod r$ і $p = u \equiv -1 \bmod s$.

5. Якщо число $u$ – непарне, то присвоюємо $p_0 = u$, інакше вважаємо $p_0 = u + rs$.

6. Будуємо $p$ – найближче просте число порівнянне з непарним числом $p_0$ за модулем $rs$, тобто тестуємо на простоту числа вигляду $p = p_0 + 2krs$, $k = 0, 1, \ldots$, поки не знайдеться просте число (або спрацюють обмеження реалізації).

Алгоритм заснований на такій теоремі.

**Теорема (Гордон).** Якщо $r, s$ – непарні прості числа, то просте число $p$ задовольняє умови $p = u \equiv 1 \bmod r$ і $p = u \equiv -1 \bmod s$ тоді і тільки тоді, коли воно подано у вигляді $p = p_0 + 2krs$, де $p_0$ – непарне число з пари $u, u + rs$.

### 8.5.1 Приклад побудови сильного простого числа

1. Будуємо вихідне псевдовипадкове просте число $s$, розміром, скажімо, в 6 бітів. Вибираємо псевдовипадково шестибітове число: $x = 46$. В проміжку $[46, 46 + 5]$ визначаємо просте число $s = 47$.

2. Будуємо випадкове просте число $t$ аналогічно побудові числа $s$. Нехай $x = 25$. В проміжку $[25, 25 + 4]$ визначаємо просте число $t = 29$.

3. Будуємо просте число $r = 1 + 2lt$, перебираючи $l$ в проміжку $[1, 4]$. Отримуємо $l = 59$.

4. Обчислюємо $u = u(r,s) \bmod rs$, розв'язуючи за допомогою китайської теореми про лишки систему: $u \equiv 1 \bmod r$, $u = -1 \bmod s$.

Використовуючи розширений алгоритм Евкліда, отримаємо співвідношення

$4 \cdot 59 - 5 \cdot 47 = 1$, звідки: $47^{-1} \bmod 59 = 4 = -5$.

Отже,

$$u(r,s) = 1 \cdot 47 \cdot \left(47^{-1} \bmod 59\right) + (-1)59 \cdot \left(59^{-1} \bmod 47\right) = -471 = 2302(2773).$$

5. Число $u(r,s)$ – парне, тому вважаємо $p_0 = 2302 + 59 \cdot 47 = 5075$.

6. Будуємо просте число порівнянне з $p_0$ за модулем 2773, тестуючи на простоту числа вигляду $p = p_0 + 2 \cdot 2773k$. При $k = 0, 1, 2, 3$, отримуємо, відповідно: 5075, 10621, 11167, 21713.

Лише останнє число є простим. Його розрядність значно перевищує розрядність вихідного числа $s$, що небажано, оскільки, наприклад, для криптосистеми RSA потрібна побудова сильних простих чисел заданої розрядності.

При великій розрядності чисел, що використовуються в обчисленнях, кількість сильних простих велика, тому вказаний вище недолік проявляється значно менше.

Проте завжди необхідно передбачати відповідні дії у випадку, якщо проміжні дані на відповідному кроці побудувати не вдалося. Крім того, слід завжди контролювати розрядність проміжних даних і результатів.

**Питання до розділу 8**

1. Чим відрізняються детермінований і ймовірнісний тести на простоту?
2. Яка властивість імовірнісних тестів дозволяє використовувати їх для практичних додатків?
3. Тести яких типів, як правило, використовуються для вибору параметрів криптосистеми RSA?
4. Тести яких типів, як правило, використовуються для вибору параметрів криптосистеми Ель - Гамаля?
5. Яке порівняння називається порівнянням Ферма?
6. Які числа називаються псевдопростими?
7. Які числа називаються числами Кармайкла?
8. В чому полягає тест Ферма на простоту і який його недолік?
9. Яке порівняння називається співвідношенням Ейлера?
10. Які числа називаються ейлеровими псевдопростими?
11. Чи існують числа, що є ейлеровими псевдопростими за будь-якою основою?
12. В чому полягає тест Соловея-Штрассена на простоту і яка ймовірність помилки при тестуванні числа за різними основами?

13. Які числа складають множину квадратних коренів з одиниці за простим модулем?

14. Чи можна стверджувати, що кількість множин квадратних коренів з одиниці за складеним модулем не менша трьох?

15. В чому полягає тест Рабіна-Міллера на простоту і яка ймовірність помилки при тестуванні числа за різними основами?

16. Яке число називається сильним псевдопростим за заданою основою і для яких цілей застосовується метод Гордона?

17. Які загальні вимоги висуваються до параметрів криптосистеми RSA?

18. Яке визначення сильних простих чисел?

# РОЗДІЛ 9 ЕЛЕКТРОННИЙ ЦИФРОВИЙ ПІДПИС

## 9.1 Забезпечення цілісності і авторства в електронному документообігу

Підтвердженням автентичності документа є підпис уповноваженої особи.

Поставлення підпису переслідує дві цілі: по-перше, необхідно надати одержувачеві можливість переконатися в істинності повідомлення, шляхом звірення підпису з деяким зразком.

По-друге, особистим підписом забезпечується юридично значуща гарантія авторства документа (така, що може бути представлена як доказ в суді). Останній аспект особливо важливий при укладанні договорів, складанні довіреностей, зобов'язань і т. д.

Якщо підробити підпис людини на папері дуже непросто, а встановити авторство підпису сучасними криміналістськими методами – технічна деталь, то з підписом електронного документа справа інша.

Використовуючи можливості програм – редакторів текстів, будь-який користувач комп'ютера може модифікувати послідовність бітів, які представляють вихідний документ.

Значне поширення в діловому світі електронних форм подання документів, електронних банківських операцій і електронної торгівлі зумовило актуальність проблеми встановлення автентичності і авторства документів в безпаперових технологіях.

Важливо відзначити, що ця проблема виникає не тільки внаслідок дій потенційних зловмисників.

Сучасні системи електронного документообігу характеризуються масовістю інформаційного обміну, що призводить до виникнення ненавмисних помилок при роботі операторів.

Виявлення подібних випадків є гарантією правильного функціонування організацій, що використовують електронний документообіг.

Підсистема безпеки системи електронного документообігу повинна захищати її легальних користувачів від таких зловмисних дій (рис. 9.1).

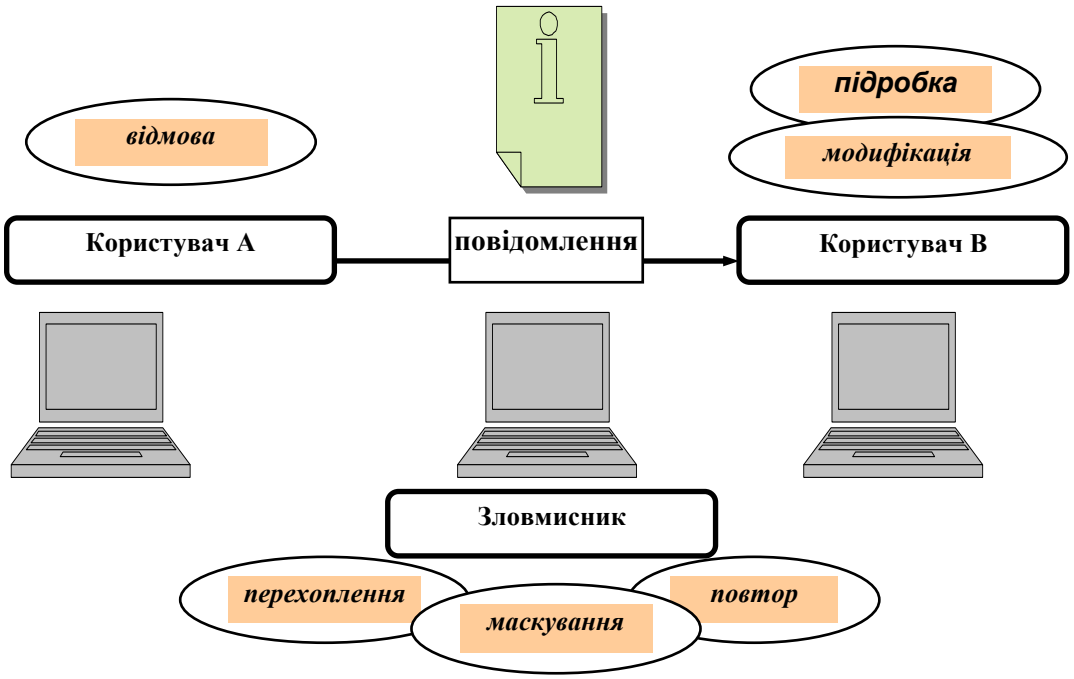



Рисунок 9.1 - Загрози безпеки повідомлень

1. Відмова від відправленого повідомлення.

2. Модифікація прийнятого повідомлення і спроба стверджувати, що воно прийнято саме у такому вигляді від будь-кого.

3. Підробка – створення фіктивного повідомлення і спроба стверджувати, що воно прислане конкретною особою.

4. Активне перехоплення – втручання в роботу системи зв'язку з метою прихованої модифікації повідомлень, що передаються.

5. Маскування (імітація) - відправлення повідомлень від імені абонента мережі, що не відправляв їх.

Для захисту від модифікації, підробки і маскування використовуються цифрові сигнатури – спеціально сформовані за допомогою криптографічних перетворень послідовності символів, залежні від секретного ключа і вихідних повідомлень.

6. Повтор повідомлень, що раніше передавались. Не дивлячись на те, що вживаються всі можливі заходи захисту від повторів, саме на цей метод припадає більшість випадків незаконного зняття і витрати грошей в системах електронних платежів. Дійовим методом захисту від повтору є використання імітовставок і облік вхідних повідомлень.

Найбільш вдалим методом аутентифікації повідомлень, джерел повідомлень, цілісності даних, а також інших загроз є протоколи, які називаються цифровими підписами.

Ідеї, на яких засновані цифрові підписи, використовуються також для побудови протоколів інших типів, наприклад, протоколів поширення ключів, протоколів фінансової криптографії і т. п. [5, 6, 7, 11, 12].

Цифровим підписом (ЦП) називається результат спеціального криптографічного перетворення, здійсненого над електронним документом його власником.

Мета перетворення – довести незаперечність тексту документа і факту перетворення даних конкретною особою.

Основний метод – перевірка перевіряючим факту використовування секретного параметра (ключа) підпису без його знання.

Як правило, підпис являє собою один або два блоки даних.

Підписане повідомлення – це вихідне повідомлення, що передається разом з ЦП.

Відомо багато схем цифрового підпису, найбільш просто принцип його побудови ілюструється на прикладі алгоритму RSA.

Припустимо, що задана криптосистема RSA з параметрами $e, d, n$.

Нехай абонент А повинен передати абоненту Б повідомлення $m$.

Для цього абонент А, використовуючи свій секретний ключ $d_A$ і відкритий ключ $e_B$ абонента Б, «підписує» повідомлення за допомогою перетворення: $E_{e_B,n_B}\left(E_{d_A,n_A}(m)\right)$.

Для перевірки підпису абонент Б спочатку розшифровує підписане повідомлення за допомогою свого секретного ключа $d_B$ і одержує: $E_{d_A,n_A}(m)$. Потім, використовуючи відкритий ключ абонента А він отримає $m$, виходячи з $E_{d_A,n_A}(m)$.

Істинність повідомлення, а значить і підпису, забезпечується використанням секретного ключа $d_A$. Проте не все так просто.

Для даного протоколу можлива підробка випадкових, тобто не смислових, даних. Для асиметричної криптосистеми це явний недолік, оскільки початкове її призначення – поширення ключів.

Спробуємо нав'язати ключ $m$ абоненту Б від імені абонента А, для чого, відповідно до протоколу, відішлемо абоненту Б повідомлення $m$ виду $m = k^{e_A} \bmod n_A$, де $k$ – довільний великий лишок за модулем $n$.

Для цього потрібно зашифрувати повідомлення секретним ключем $d_A$ і потім перешифрувати результат відкритим ключем $e_B$ абонента Б.

Проте ми знаємо результат зашифрування секретним ключем $d_A$: $m^{d_A} = k^{e_A d_A} = k \bmod n_A$, і, отже, можемо перешифрувати результат ключем $e_B$.

Таким чином, абонент Б отримає повідомлення $k = m^{d_A} \bmod n_A$, потім застосує відкритий ключ $e_A$ абонента А і отримає $k^{e_A} = m \bmod n_A$.

Щоб захиститися від подібної атаки, достатньо шифрувати повідомлення $m$ відкритим ключем одержувача (якщо в цьому є необхідність), а секретний ключ відправника застосовувати для шифрування перетвореного повідомлення виду $h = h(m)$.

Перетворення $h = h(m)$ повинно бути складним і необернениим, а також, в нашому випадку, несекретним, хоча, при необхідності, таке перетворення може залежати і від секретних параметрів: $h = h(m, K)$.

В цифровому підписі, заснованому на RSA, як підписане повідомлення використовується пара $(m, c) = \left(m, h(m)^{d_A} \bmod n_A\right)$, де $h(m)$ – так звана хеш-функція повідомлення $m$: несекретне, криптографічно стійке перетворення з особливими властивостями.

Значення $z = h(m)$ називається хеш-кодом повідомлення $m$. Для заданої хеш-функції бітова довжина хеш-коду постійна, тобто від $m$ не залежить.

Хеш-код також називають «дайджестом» повідомлення.

Перевірка підпису $(m, c)$ починається з переобчислення $h(m)$, а потім результат порівнюється з $c^e \left(\bmod n_A\right)$.

Очевидно, що подібна схема дозволяє захиститися від декількох видів порушень.

Абонент А не може відмовитися від свого повідомлення, якщо він визнає, що секретний ключ відомий тільки йому. Зловмисник без знання секретного ключа не може ні сформувати, ні зробити зміну повідомлення, що передається по лінії зв'язку.

Дана схема дозволяє при розв'язанні багатьох конфліктних ситуацій обійтися без посередників.

Іноді немає необхідності зашифровувати повідомлення, що передається, але його потрібно завірити електронним цифровим підписом.

В цьому випадку використовується схема, зображена на рис. 9.2.

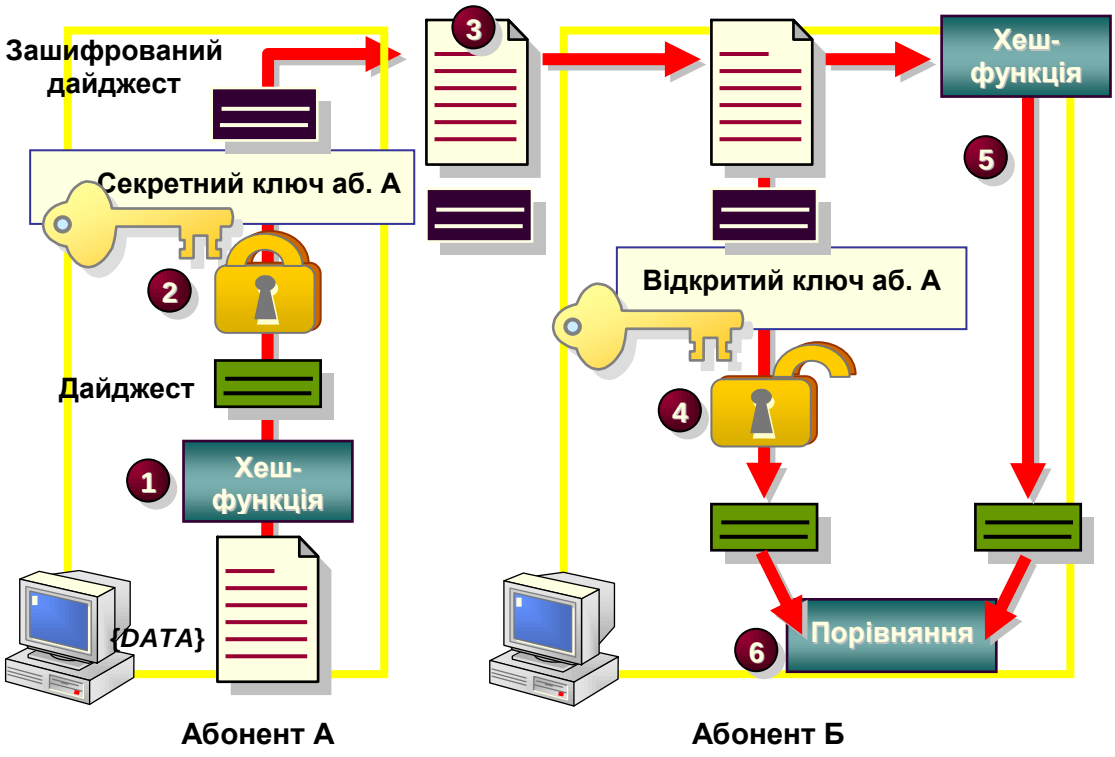



Рисунок 9.2 - Схема ЕЦП без шифрування повідомлення

Першим кроком обчислюється дайджест повідомлення.

Для цього використовується деяка спеціальна функція - хеш-функція. Далі (крок 2) дайджест шифрується секретним ключем відправника. Одержана послідовність символів – електронний цифровий підпис – об'єднується з вихідним документом і прямує до абонента Б (крок 3). Одержувач за допомогою відкритого ключа відправника розшифровує дайджест (крок 4) і обчислює заново хеш-код прийнятого повідомлення (крок 5). Набуті значення порівнюються (крок 6).

У разі збігу вважається підтвердженим авторство абонента А і відсутність спотворень (цілісність) повідомлення.

## 9.2 Функції хешування

Хеш-функції є одним з найважливіших елементів криптосистем, що реалізуються ЕЦП.

Реальні хеш-функції є складними алгоритмами, рекомендованими у відповідних стандартах. Методика їх обґрунтовування виходить за рамки даної книги.

Означення хеш-функції. Хеш-функція $z = h(m)$ – перетворення бітового рядка довільної довжини в бітовий рядок (блок) фіксованої довжини (як правило, 128-512 бітів), що має такі властивості.

1. Відновлення $m$ за $z$, виходячи із співвідношення $z = h(m)$, обчислювально нереалізовне.

2. Виходячи з $m$ і $z$, обчислювально нереалізоване визначення другого прообразу для $z$, тобто такого повідомлення $m_1 \neq m$, що $z = h(m) = h(m_1)$.

В криптографічній практиці, як правило, використовуються так звані вільні від колізій хеш-функції, що задовольняють наведену нижче, більш жорстку, ніж попередня, умову 3.

3. Вимагається, щоб знаходження довільної колізії, тобто пари повідомлень $x, y$ $(x \neq y)$ таких, що $h(x) = h(y)$, було обчислювально нереалізоване.

В криптографії застосовуються також перетворення за властивостями, близькими до хеш-функцій, які використовуються, наприклад, для аутентифікації повідомлень в симетричних криптосистемах.

Їх називають кодами аутентифікації повідомлень – Message Authentication Code (MAC). Коди MAC можуть бути побудовані як хеш-функції з секретним параметром $K$ у вигляді $h = h(m, K)$.

Обчислення MAC для формованих в сеансі зв'язку випадкових текстів дозволяє абонентам продемонструвати один одному володіння загальним симетричним ключем, не дозволяючи зловмиснику одержати пару відкритий-шифрований текст.

Для аутентифікації повідомлень в симетричних криптосистемах MAC передається разом з повідомленням. Абонент, якому відомий

симетричний ключ, переобчислює MAC і порівнює його з прийнятим значенням.

В комп'ютерних системах і мережах широко використовується типовий алгоритм HMAC обчислення коду MAC, з використанням довільної вбудованої хеш-функції $h$.

Це пов'язано з тим, що стандартні хеш-функції безпосередньо не передбачають використовування секретних параметрів.

Проте бажано мати нагоду застосовувати їх для вироблення кодів аутентифікації повідомлень, оскільки криптографічна стійкість хеш-функцій висока, а час їх обчислення менший, ніж час шифрування даних, скажімо, блоковими шифрами при побудові кодів аутентифікації на їх основі.

Хеш-функції також стандартизовані. Крім того, існує загальна методика побудови хеш-функцій на основі блокових шифрів.

Характеристики деяких стандартизованих хеш-функцій наведені в таблиці 9.1.

Найвідоміші з хеш-функцій: MD2, MD4, MD5 і SHA-1.

Три алгоритми серії MD розроблено Рівестом в 1989-му, 90-му і 91-му роках відповідно. Всі вони перетворюють бітові рядки довільної довжини в 128-бітовий хеш-код (дайджест).

Таблиця 9.1 - Характеристики хеш-функцій

| Назва | Характеристика функції |
|---|---|
| **MD2** | **Довжина хеш-коду - 128 бітів. Найбільш повільна, розрахована для 8-бітових машин. Використовується відносно рідко.** |
| **MD4** | **Довжина хеш-коду - 128 бітів. Швидка, оптимізована для 32-бітових машин. Непопулярна, оскільки є повідомлення про її дешифрування** |
| **MD5** | **Довжина хеш-коду - 128 бітів. Найбільш розповсюджене використання – спільно з RSA. Схожа на MD4. Більш надійна, але засоби підвищення безпеки роблять її на третину повільнішою, ніж MD4.** |
| **SHA-1** | **Довжина хеш-коду-160 бітів. Призначена для використання в стандарті DSS. Використовує принципи, закладені в MD4, MD5.** |
| **ГОСТ 34.311** | **Довжина хеш-коду - 256 бітів. Побудована на основі оригінальних перетворень. Використовує, у тому числі алгоритм шифрування ГОСТ 28147-89. Застосовується спільно з Національним стандартом ДСТУ 4145-2002.** |

В алгоритмі MD4 досить швидко були знайдені слабкі місця, тому він був замінений алгоритмом MD5, в якому кожний блок бере участь не в трьох, а в чотирьох різних циклах.

Алгоритм SHA (Secure Hash Algorithm), розроблений NIST, повторює ідеї серії MD. В SHA використовуються тексти більші $2^{64}$ бітів. Даний алгоритм спочатку призначався для використання в рамках державної програми США – «Capstone», яка припускала централізоване зберігання всіх ключів, що використовуються організаціями і приватними особами.

### 9.3 Алгоритм SHA-1

Алгоритм SHA (Secure Hash Algorithm) є частиною стандарту SHS, прийнятого АНБ і Національним інститутом стандартів і технологій США в 1993 році. Для версії SHA-1 довжина хеш-коду $H = H(M)$ складає 160 бітів.

У кінець повідомлення $M$, яке подано у вигляді рядка бітів, приписується біт, рівний 1. Потім, при необхідності, дописуються нулі так, щоб довжина отриманого повідомлення, збільшена на 64, була кратна 512.

Далі формується повідомлення, що складається з $N$ блоків довжиною 512 бітів, в останньому блоці якого зарезервовано поле довжиною в 64 біти.

У цьому полі розміщується число, рівне вихідній довжині повідомлення $M$ (без допису).

Розширене повідомлення $M$ обробляється блоками по 512 бітів.

При цьому окремий блок розглядається як масив, що складається з 16 слів, кожне довжиною в 32 біти.

Кожен блок обробляється за одну ітерацію (цикл). Обчислення хеш-коду здійснюється як послідовність ітерацій з кроковою функцією стискання $H_i = f(M_i, H_{i-1}), H_0 = V$, $i = 1, \ldots, N$.

Блок $V$ є конкатенацією п'яти зарезервованих початкових значень $V = A, B, C, D, E$, кожне дожиною в 32 біти, які в шістнадцятковій системі числення мають такий вигляд:

$A = 67452301$, $B = EFCDAB89$, $C = 98BADCFE$, $D = 10325476$, $E = C3D2E1F0$.

Після обробки чергового блоку $M_i$ повідомлення $M$ значення змінних $A, B, C, D, E$ модифікуються.

Результатом роботи кожного циклу $H_i = f(M_i, H_{i-1})$ є конкатенація модифікованих значень змінних $A, B, C, D, E$.

Результат роботи останнього циклу дає значення хеш-коду $H = H(M)$.

Кожен цикл складається з 80 кроків. На кроці $j$ циклу з номером $i$ обробляється слово $M_i(j)$ довжиною в 32 біти.

На початку кожного циклу створюються копії змінних $A, B, C, D, E$: $A1 = A, B1 = B, C1 = C, D1 = D, E1 = E$.

Слова, що обробляються в циклі $i$, утворюються з блоку $M_i$ таким чином.

1. Блок $M_i$ розглядається як послідовність 16 слів $M_i(j)$, $j = 1,\ldots,16$, виходячи з яких, за рекурентним законом формуються останні 64 слова. Рекурентне співвідношення має вигляд:

$$M_i(j) = Sft[1](M_i(j-3) \oplus M_i(j-8) \oplus M_i(j-14) \oplus M_i(j-16)),\ j = 17,\ldots,80.$$

Тут функція $Sft[t]$ означає циклічний зсув слова на $t$ розрядів вліво, а операція $\oplus$ є порозрядною сумою слів за модулем два.

2. Обробка вісімдесяти слів $M_i(j)$ виконується послідовно, групами, по 20 слів в групі.

Кожній з чотирьох груп відповідають 20 кроків циклу (один раунд): $j = 1,\ldots,20$; $j = 21,\ldots,40$; $j = 41,\ldots,60$; $j = 61,\ldots,80$.

Крім того, з кожним раундом $R(k)$, $k = 1,\ldots,4$ пов'язана своя константа $c(k)$ довжиною в 32 біти і функція $f_k(x, y, z)$ від трьох змінних, кожна з яких є словом тієї ж довжини.

При обробці слова $M_i(j)$ використовується константа і функція, пов'язані з відповідним раундом $R(k)$.

3. Обробка чергового слова приводить до зміни слів $A, B, C, D, E$, що виглядає так:

$T = Sft[5]A + f_k(B, C, D) + E + M_i(j) + c_k$, $E = D$, $D = C$, $C = Sft[30]B$, $A = T$.

Тут $T$ – допоміжна змінна, а додавання відбувається за модулем $2^{32}$.

4. Після 80 кроків обробки блоку повідомлення цикл завершується модифікацією змінних $A,B,C,D,E$: $A = A + A1, B = B + B1$, $C = C + C1$, $D = D + D1$, $E = E + E1$ (додавання за $\bmod 2^{32}$).

Кінець циклу.

Для повноти наведемо $c(k)$ і $f_k(x,y,z)$, які використовуються в циклах $R(k)\ k = 1,\ldots,4$.

$c(1) = 5A827999$, $f_1(x,y,z) = (x \wedge y) \vee ((\neg x) \wedge z)$;

$c(2) = 6ED9EBA1$, $f_2(x,y,z) = x \oplus y \oplus z$;

$c(3) = 8F1BBCDC$, $f_3(x,y,z) = (x \wedge y) \vee (x \wedge z) \vee (y \wedge z)$;

$c(4) = CA62C1D6$, $f_4(x,y,z) = x \oplus y \oplus z$.

## 9.4 Стандарти алгоритмів формування і перевірки ЕЦП

Один з перших стандартів ЕЦП DSS був запропонований NIST в 1991 році для урядових систем (використовується спільно з функцією хешування повідомлень SHA-1 -Secure Hash Algorithm).

У 1994 році ФАПСІ РФ був розроблений стандарт ГОСТ Р34.10 цифрового підпису спільно з функцією хешування – ГОСТ Р34.11, які як міждержавні були прийняті в Україні (ГОСТ 34.310 і ГОСТ 34.311 відповідно).

Механізми цифрового підпису, рекомендовані DSS і ГОСТ Р34.10, засновані на складності розв'язання задачі дискретного логарифмування. Розмір ключа – 512 або 1024 біти.

Основою вказаних механізмів ЦП, як і більшості інших, є схема цифрового підпису Ель Гамаля, яка побудована так.

Нехай $p$ – просте число і $g$ – елемент великого порядку за модулем $p$. Скажімо, для простоти, $g$ – примітивний елемент поля лишків за модулем $p$. Пояснимо суть справи докладніше.

Порядки елементів поля ділять на $p-1$. Порядок примітивного елемента рівний $p-1$. Оскільки пошук примітивного елемента, в загальному випадку, вимагає факторизації числа $p-1$, на практиці застосовуються процедури побудови простих чисел $p$ з відомим простим

дільником $q$ числа $p-1$. При цьому розмір $q$ складає не менше половини розміру $p$.

Тому ймовірність того, що не рівний одиниці, вибраний навмання лишок $a$ задовольняє співвідношення $a^q = 1(\mathrm{mod}\, p)$, достатньо велика, щоб легко знайти його випадковим пошуком.

Взагалі кажучи, при виконанні вказаного співвідношення можна стверджувати, що порядок числа $a$ ділить $q$, але з умови, що $q$ просте і $a \neq 1$, слідує, що порядок числа $a$ рівний $q$.

Отже, нехай $p$ і $g$ вибрані. Далі виберемо випадкове секретне число $x$ з множини $\{2,\ldots, p-2\}$ і обчислимо значення $y = g^x(p)$. Число $x$ називається секретним ключем, а набір $(p, g, y)$ – відкритим ключем цифрового підпису.

Підпис для повідомлення $T$ подається парою блоків $(r, s)$ і обчислюється за допомогою такого алгоритму.

1. Згенерувати випадкове секретне число $k$ в інтервалі $\{2,\ldots, p-2\}$, взаємно просте з $p-1$.

2. Обчислити $r = g^k \bmod p$.

3. Для повідомлення $T$ обчислити значення хеш-функції $h = h(T)$ (хеш-код).

4. Присвоїти значення формальним параметрам $A, B, C$ порівняння $Ak + Bx + C = 0 \bmod(p-1)$ у вигляді $(A, B, C) = (s, r, -h)$, що дає порівняння $h = xr + ks \bmod(p-1)$.

5. Вирішити отримане порівняння відносно $s$: $s = k^{-1}(h - xr) \bmod(p-1)$.

6. Як підпис для повідомлення $T$ приймаємо пару $(r, s)$.

Перевірка підпису полягає в перевірці співвідношення $g^h = y^r r^s \bmod p$, оскільки при дійсних значеннях параметрів $h = xr + ks \bmod(p-1) \Rightarrow g^h = g^{xr} g^{ks} \bmod p \Rightarrow g^h = y^r r^s \bmod p$.

Відзначимо, що число $k$ (рандомізатор) повинне знищуватися відразу ж після формування підпису, оскільки його компрометація неминуче приводить до компрометації секретного ключа $x$. Це очевидним чином виходить з формули для обчислення $s$.

*Примітка*. Особливі випадки, як правило, обговорюються окремо і враховуються при перевірці ЦП. Наприклад, при $r = 0$ або $s = 0$ перейти на вибір нового рандомізатора; при $h(T) = 0$ прийняти $h = 1$.

Схема цифрового підпису Ель-Гамаля [11] стала основою для побудови цілого сімейства схем, залежно від значень, привласнених вектору параметрів $(A, B, C)$ з множини векторів-перестановок чисел $(s, \pm r, \pm h)$.

Зокрема, в алгоритмі DSA $r = g^k \bmod p$, $(A, B, C) = (s, -r, -h)$.

Тому $sk - rx - h = 0 \bmod(p-1)$ і $s = k^{-1}(rx + h) \bmod(p-1)$.

Перевірне співвідношення для DSA задається у вигляді $r = y^m g^n$, де $m = s^{-1}r$, $n = s^{-1}h$.

У деяких схемах ЦП параметрам $(A, B, C)$ присвоюються декілька більш складних значень, а також вводиться додатковий параметр $D = \pm 1$, який служить для формування відкритого ключа у вигляді $y = g^{x^D} (\bmod p)$.

Подібні ускладнення дозволяють побудувати оригінальні схеми ЦП, а також допомагають оптимізувати обчислення.

Очевидною атакою на ЦП, а також на асиметричні криптосистеми взагалі, є оптимізований перебір варіантів ключів. Як наслідок, для еквівалентного рівня захисту інформації необхідне використання довших ключів, ніж ті, що застосовуються в симетричних криптосистемах. Це відразу ж позначається на обчислювальних ресурсах, потрібних для їх реалізації.

Останнім часом активно розвивається напрям, пов'язаний з побудовою криптографічних алгоритмів, що використовують апарат еліптичних кривих (ECC). Саме з метою вирішення проблеми зростання довжин ключів в 2000-2002 р.р. у США, Росії і Україні (ДСТУ 4145-2002) були введені в дію стандарти ЦП, що базуються на застосуванні еліптичних кривих.

В ряді видань наводяться дані про еквівалентні довжини ключів. Деякі умовно еквівалентні довжини ключів зведені в таблицю 9.2.

Таблиця 9.2 - Еквівалентні довжини ключів

| **Довжина ключа в бітах** | | | **Число варіантів ключів** |
|---|---|---|---|
| **Симетричні системи** | **RSA системи** | **ECC системи** | |
| **56-64** | **384-512** | **112** | $\mathbf{10^{17}\text{-}10^{19}}$ |
| **80** | **768** | **132** | $\mathbf{10^{24}}$ |
| **112** | **1792** | **160** | $\mathbf{10^{34}}$ |
| **128** | **2304** | **240** | $\mathbf{10^{38}}$ |

## 9.5 Протоколи взаємодії і сертифікати в стандарті X.509

Криптосистеми, що реалізують відкритий розподіл ключів і ЕЦП, привабливі для користувачів через можливість забезпечення високого рівня безпеки в умовах обміну по каналу зв'язку тільки несекретною інформацією.

Проте, в цьому випадку, серйозним ризиком для безпеки системи є можливість підміни відкритих ключів. Як варіант, зловмисник може зайняти місце легального користувача або підмінити його відкритий ключ (а разом з ним і секретний) своїм власним. В цьому випадку він зможе читати не призначені для нього повідомлення.

Для захисту від цієї загрози відкриті ключі можуть розміщуватися в підписаних (завірених) сертифікатах, використовуваних для перевірки дійсності відкритих ключів.

Перед використанням відкритого ключа перевіряється його сертифікат (зокрема, шляхом перевірки його підпису). Сертифікат зазвичай містить ідентифікатор користувача, відкритий ключ і відмітку дати-часу.

Сертифікат підписується центром сертифікації ключів, чиї власні ключі можуть бути завірені повноваженнями органу сертифікації вищого рівня. Сертифікати передаються від центру сертифікації електронним шляхом.

Для деякої компенсації низької швидкості алгоритмів асиметричного шифрування генерується тимчасовий сеансовий симетричний ключ для кожного повідомлення.

Само повідомлення шифрується з використанням цього тимчасового сеансового ключа і відповідного алгоритму симетричного шифрування.

Сеансовий ключ шифрується за допомогою відкритого асиметричного ключа адресата і асиметричного алгоритму шифрування. Після цього зашифрований сеансовий ключ разом із зашифрованим повідомленням передається адресатові.

Адресат, використовуючи той же самий асиметричний алгоритм шифрування і свій секретний ключ, розшифровує сеансовий ключ, а з його допомогою расшифровується власне повідомлення.

Подібні криптосистеми називаються комбінованими (змішаними, гібридними).

В асиметричних криптосистемах важливо, щоб сеансові і асиметричні ключі були зіставні відносно рівня безпеки, який вони забезпечують.

Якщо використовується короткий сеансовий ключ (наприклад, 40-бітовий DES), то немає значення, наскільки великі асиметричні ключі. Атаці будуть піддані не вони, а сеансові ключі алгоритму DES.

Потрібно мати на увазі, що якщо в комбінованій системі атакуючій стороні стане відомим секретний ключ асиметричного алгоритму, то буде скомпрометовано не тільки поточне, але і всі подальші повідомлення, відправлені адресатові.

Для організації захищеного інформаційного обміну в системі з відкритим розподілом ключів повинен бути забезпечений певний порядок взаємодії підсистем, що визначається так.

Безпечно створюються і розповсюджуються відкриті і секретні ключі асиметричних алгоритмів.

Секретний ключ передається його власникові. Відкритий ключ зберігається в базі даних, відповідній стандарту X.500, і адмініструється центром сертифікації ключів (Certification Authority або ЦСК).

Передбачається, що користувачі довіряють адміністрації системи в питаннях забезпечення безпеки при створенні, розподілі і адмініструванні ключів.

Більш того, якщо центр генерації ключів і особа (орган), що адмініструють їх, не одне і те ж, то кінцевий користувач повинен вірити, що створювач ключів дійсно знищив всі їх копії.

Для кожного повідомлення обчислюється його хеш-функція. Отримане значення за допомогою алгоритму ЕЦП і секретного ключа відправника перетворюється в ЕЦП, а потім отриманий рядок символів додається до передаваного повідомлення (тільки відправник може створити ЕЦП).

Далі генерується секретний ключ симетричного криптоалгоритму, який використовуватиметься для шифрування тільки цього повідомлення або сеансу взаємодії (сеансовий ключ). Після чого вихідний текст шифрується разом з доданим до нього електронним підписом за допомогою симетричного криптоалгоритму і разового (сеансового) ключа – виходить шифртекст.

Тепер потрібно вирішити проблему з передачею сеансового ключа одержувачеві повідомлення.

Відправник повинен отримати для асиметричного криптоалгоритму відкритий ключ одержувача від центру сертифікації ключів ЦСК.

При цьому потрібно враховувати, що перехоплення незашифрованих запитів на отримання відкритого ключа є поширеною формою атаки.

Може існувати ціла система сертифікатів, які підтверджують достовірність відкритого ключа ЦСК.

Стандарт X.509 описує ряд методів для отримання користувачами відкритих ключів від ЦСК, проте практично жоден з них не може з 100% гарантією захистити від підміни відкритого ключа, що є певним обмеженням для застосування систем з відкритим розподілом ключів (Д. Чандлер однозначно стверджує, що немає такої системи, в якій можна було б гарантувати достовірність відкритого ключа ЦСК).

Отже, відправник запитує у ЦСК відкритий ключ одержувача повідомлення. Цей процес уразливий до атаки, в ході якої той, хто атакує втручається у взаємодію між відправником і одержувачем і може модифікувати трафік, що передається між ними. Тому відкритий ключ одержувача «підписується» ЦСК.

Це означає, що ЦСК використовував свій секретний ключ для шифрування відкритого ключа одержувача.

Оскільки тільки ЦСК знає свій секретний ключ, є певна гарантія того, що відкритий ключ адресата отриманий саме від ЦСК.

Після отримання відкритий ключ одержувача перевіряється за допомогою відкритого ключа ЦСК і алгоритму ЕЦП. При цьому, природно, передбачається, що центр не був скомпрометований. Якщо ж він виявляється скомпрометованим, то може бути виведена з ладу вся мережа користувачів.

Потім сеансовий ключ шифрується з використанням асиметричного криптоалгоритму (наприклад, алгоритму Ель-Гамаля) і відкритого ключа одержувача (отриманого від ЦСК і перевіреного).

Зашифрований сеансовий ключ об'єднується з шифртекстом, який включає доданий раніше ЕЦП.

Весь отриманий пакет даних (зашифрований сеансовий ключ і шифртекст, в який, окрім початкового тексту, входить його ЕЦП) передається одержувачеві.

Оскільки зашифрований сеансовий ключ передається по незахищеній мережі, він є очевидним об'єктом різних атак.

Одержувач виділяє з отриманого пакета зашифрований сеансовий ключ, який розшифровує, використовуючи свій секретний ключ і той же самий асиметричний криптоалгоритм.

Потім одержувач, застосовуючи той же самий симетричний криптоалгоритм і розшифрований сеансовий ключ, відновлює з шифртексту початковий текст разом з ЕЦП. Електронний підпис відділяється від початкового тексту.

Далі одержувач запитує у ЦСК відкритий ключ ЕЦП відправника.

Після отримання цього ключа він перевіряє його за допомогою відкритого ключа ЦСК і відповідного асиметричного криптоалгоритму.

Потім за допомогою алгоритму ЕЦП і відкритого ключа адресанта перевіряється достовірність повідомлення.

### 9.6 Структура сертифіката відкритих ключів

Для захисту від загрози підміни відкритих ключів, вони можуть розміщуватися в підписаних (завірених) сертифікатах, використовуваних для перевірки дійсності ключів.

Сертифікат відкритого ключа (public key certificate – PKC) являє собою деяку сукупністю даних, правильне використання яких мінімізує

ризики атак на інфраструктуру відкритих ключів (public key infrastructure - PKI).

Сертифікат відкритого ключа пов'язаний з його власником, при цьому юридична відповідальність за підтвердження зв'язку сертифіката і деякого користувача системи ЕЦП покладається на центри сертифікації ключів або засвідчувальні центри.

Залежно від конкретного застосування використовуються різні види сертифікатів. Серед них можна відзначити такі, як сертифікати PGP (Pretty Good Privacy) і сертифікати IPSec (Internet Protocol Security), використовувані відповідними засобами.

Найбільшого поширення набув формат сертифіката, визначеного стандартом X.509v3 Міжнародного союзу з телекомунікацій. Зразкова структура сертифіката наведена на рис. 9.3.

Сертифікат містить ідентифікатор користувача, відкритий ключ і термін його дії, відмітку синхронізованого часу у форматі – «ррммддггххсс» або узагальненої дати-часу «рррррммддггххсс».

| **Версія** |
|---|
| **Реєстраційний номер сертифіката** |
| **Ідентифікатор алгоритму підпису** |
| **Назва засвідчувального центру** |
| **Термін дії (з . по .)** |
| **Ім'я власника відкритого ключа** |
| **Значення відкритого ключа** |
| **Унікальний ідентифікатор центру сертифікації ключів** |
| **Унікальний ідентифікатор власника відкритого ключа** |
| **Розширення** |
| **Підпис** |

Рисунок 9.3 - Формат сертифіката відкритого ключа

У поле «Версія» заноситься інформація про використовувану версію стандарту X.509: версія 1, 2 або 3.

Сертифікат підписується центром сертифікації ключів, чиї власні ключі можуть бути завірені органом сертифікації вищого рівня. Перед використанням відкритого ключа його сертифікат перевіряється, зокрема, за допомогою перевірки цифрового підпису самого сертифіката.

Передача сертифіката від центру сертифікації може здійснюватися електронним шляхом або особисто користувачеві при первинній реєстрації в системі.

### 9.6.1 Приклад сертифіката в технології Fortezza

Технологія Fortezza була розроблена для створення повнозв'язних корпоративних обчислювальних мереж, призначених для захищеного обміну документальною інформацією органів федерального управління США. Аналогічні системи розроблені в Україні (засіб КЗІ «Мімоза» й ін.) і Росії (засіб КЗІ «Верба»).

Оскільки в технології Fortezza використовується відкритий розподіл ключів, то повинен існувати протокол, що регламентує видачу і розповсюдження відкритих ключів користувачів.

У технології Fortezza [43] відкриті ключі асоціюються з їх власниками за допомогою сертифікатів, що являють собою деяку структуру даних, яка включає ідентифікатор користувача, відкриті ключі, призначені для алгоритмів KEA (обмін ключами) і DSA (цифровий підпис), а також інформацію про особу, що видала сертифікат.

З метою захисту сертифіката від підробки він підписується цифровим підписом органу, що видав сертифікат. Сертифікати і пари секретних/відкритих ключів утворюють основу системи управління ключами технології Fortezza.

За основу системи аутентифікації Fortezza використовує схему аутентифікації сертифікатів X.509 і угоди про найменування об'єктів X.500. Загальна довжина сертифіката 2820 байтів.

Технологія Fortezza розрізняє дві структури сертифікатів.

Під «сертифікатом Fortezza» розуміється внутрішня структура даних технології Fortezza, під «сертифікатом X.509» - блок даних стандарту X.509, що міститься в сертифікаті Fortezza (рис. 9.4).

Кожний сертифікат Fortezza складається з двох пар відкритих/секретних ключів (одна з них призначена для використання в KEA, інша – в DSA) і відповідних їм значень параметрів P, Q і G.

Сертифікат X.509 містить відкриті компоненти цих ключів. Відкриті ключі завжди доступні користувачеві криптокарти.

| Мітка | Прапорці | Розмір даних | DSA Private (X) | KEA Private (X) | DSA | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | Розмір P,G | Розмір Q | P | Q | G |
| 32 | 16 | 4 | 48 | 48 | 4 | 4 | 128 | 48 | 128 |

| KEA | | | | | Сертифікат X.509 |
|---|---|---|---|---|---|
| Розмір P,G | Розмір Q | P | Q | G | |
| 4 | 4 | 128 | 48 | 128 | 2048 байт |

Рисунок 9.4 - Структура сертифіката в технології Fortezza

Ключі зберігаються в закодованому вигляді: секретні – за допомогою локальних ключів користувача $K_s$ (ключ $K_s$ має розмір 80 бітів, знаходиться в спеціальному регістрі криптокарти і стає доступним після успішного введення PIN користувача), відкриті – за допомогою кодування ASN.1.

Поле даних розміром 2048 байтів, зарезервоване для сертифіката X.509, може використовуватися для зберігання будь-якої інформації (біометричних даних, фотозображень). Додатки можуть завантажувати ці дані в енергонезалежну пам'ять карти і зберігати їх там тривалий час.

Сертифікати X.509 можуть бути розміщені в базу даних спеціалізованого «сертифікаційного» сервера (декількох серверів) або розподілені по мережі і збережені локально в криптокартах всіх учасників інформаційного обміну. Єдиною умовою є доступність сертифіката для криптографічних функцій додатків Fortezza.

Деякі додатки дозволяють включати сертифікат відправника в заголовок повідомлення, надаючи адресатові можливість динамічно створювати локальну базу сертифікатів абонентів.

Така локальна база може служити свого роду кешем сертифікатів, що робить можливим відправлення повідомлень без звернення до

сервера.Проте довге використання локальної бази може привести до застаріння інформації, що міститься в ній.

**Питання до розділу 9**

1. Які зловмисні дії є найбільш небезпечними для електронного документообігу?
2. Що називається цифровим підписом електронного документа і яка його мета?
3. На чому заснований метод перевірки цифрового підпису?
4. Яким чином реалізується механізм цифрового підпису на основі RSA з використанням хеш-функції?
5. Яке визначення хеш-функції, стійкої проти колізій?
6. У чому основна відмінність параметрів коду аутентифікації повідомлення від параметрів цифрового підпису?
7. З якою метою застосовується алгоритм HMAC в комп'ютерних системах і мережах?
8. Які довжини хеш-кодів відповідають хеш-функціям MD5, SHA-1, ГОСТ 34.311-95?
9. На чому заснована стійкість цифрового підпису Ель-Гамаля і як виглядає підписане повідомлення?
10. Які основні вимоги до відкритих параметрів ЦП Ель-Гамаля?
11. Яке перевіряюче співвідношення для ЦП Ель-Гамаля?
12. Чому рандомізатори в ЦП Ель-Гамаля повинні бути взаємно простими з функцією Ейлера від характеристики поля?
13. Чому можливість підміни відкритих ключів є серйозним ризиком для безпеки інформаційної системи?
14. У чому полягає механізм блокування можливості підміни відкритих ключів, що використовує цифрові сертифікати?
15. Яким чином засвідчується достовірність цифрового сертифіката?
16. Які криптосистеми називаються комбінованими (змішаними, гібридними) і чому для них необхідно використовувати сертифікати відкритих ключів?

17. На кого покладається юридична відповідальність за підтвердження належності сертифіката користувачеві інформаційної системи?

18. Яке призначення основних даних, передбачених форматом сертифіката X.509 версії 3?

19. На чому заснована система аутентифікації в технології Fortezza?

20. Чи дозволяє сертифікат Fortezza оперувати з біометричними даними?

# РОЗДІЛ 10 КРИПТОГРАФІЧНІ ПРОТОКОЛИ

## 10.1 Поняття криптографічного протоколу

Проблеми забезпечення безпеки електронного документообігу і електронної комерції складні і багатогранні.

Забезпечення безпеки інформації не полягає лише у створенні умов для збереження службової або комерційної таємниці, а вимагає створення ефективних механізмів протидії різним атакам, проведення надійної аутентифікації легальних суб'єктів і об'єктів обчислювальних мереж, безпечного розподілу між ними деякої критичної інформації.

Для вирішення таких завдань використовуються криптографічні протоколи, які, по суті, є послідовністю дій, що виконуються одною, двома або більше сторонами – користувачами ІТС із застосуванням криптографічних перетворень інформації або таких, що є їх невід'ємною частиною [11, 12].

До криптографічних протоколів висуваються такі вимоги:

– протокол має бути чітко і однозначно визначений;

– протокол має бути повним, дії його учасників мають бути визначені для будь-яких можливих ситуацій;

– кожен учасник повинен заздалегідь знати весь протокол, а також послідовність дій при виявленні його порушень;

– кожен учасник протоколу зобов'язаний його дотримуватися;

– порушення протоколу учасниками або втручання зловмисника не повинні знижувати стійкість криптосистеми в цілому, якщо ключі не розкриваються їх власниками.

Термін «криптографічні протоколи» залежно від контексту слід сприймати у вузькому або широкому розумінні.

У вузькому розумінні криптографічний протокол є криптографічним алгоритмом або послідовністю деяких операцій з вихідними даними, включаючи їх пересилання, зберігання, шифрування, формування і перевірку ЕЦП, генерацію випадкових даних і так далі.

В цьому випадку в результаті виконання відповідного алгоритму виконується, як правило, одна, рідше дві функції.

Прикладом може слугувати алгоритм Діффі-Хеллмана, використовуваний для формування єдиного (сеансового) симетричного ключа в незахищеному середовищі.

Широке тлумачення поняття криптографічного протоколу можна проілюструвати на прикладі Інтернет-стандартів.

В цьому випадку криптографічні протоколи є складовою частиною протоколів взаємодії в моделі відкритих систем OSI (рис. 10.1).

<table>
<tr><td colspan="2">Користувач А</td><td rowspan="2"></td><td colspan="2">Користувач Б</td></tr>
<tr><td rowspan="3">Рівень додатків HTTP,DNS, SMTP,FTP, SNMP,POP3, Telnet</td><td>7</td><td>7</td><td rowspan="3">Рівень додатків HTTP,DNS, SMTP,FTP, SNMP,POP3, Telnet</td></tr>
<tr><td>6</td><td rowspan="2">S/MIME, PGP, SET, SSH,<br>⟷<br>Kerberos, PEM, S-HTTP</td><td>6</td></tr>
<tr><td>5</td><td>5</td></tr>
<tr><td>Транспортний рівень UDP/TCP</td><td>4</td><td>SSL,TLS<br>⟷</td><td>4</td><td>Транспортний рівень UDP/TCP</td></tr>
<tr><td>Рівень з’єднань IP</td><td>3</td><td>IPsec<br>⟷</td><td>3</td><td>Рівень з’єднань IP</td></tr>
<tr><td rowspan="2">Мережевий рівень Ethernet,FDDI, Token Ring,PPP</td><td>2</td><td rowspan="2">ECP, CHAP<br>⟷</td><td>2</td><td rowspan="2">Мережевий рівень Ethernet,FDDI, Token Ring,PPP</td></tr>
<tr><td>1</td><td>1</td></tr>
<tr><td colspan="5">Фізичне транспортне середовище</td></tr>
</table>

Рисунок 10.1 - Схема протоколів взаємодії в моделі OSI

Під відкритими системами розуміють впорядковані сукупності різного обчислювального і телекомунікаційного устаткування, програмних систем (можливо, від різних виробників), спільне функціонування яких забезпечується відповідністю вимогам стандартів міжнародних і національних організацій з телекомунікацій [21, 22, 23].

Серед таких організацій потрібно відзначити Міжнародний союз з електрозв'язку (International Telecommunication Union, ITU), Американський національний інститут стандартів (American National Standards Institute, ANSI) і Інститут інженерів з електроніки і електромеханіки (Institute of Electrical and Electronic Engineers, IEEE).

Такий підхід дозволяє пов’язати деяким оптимальним чином в єдиній системі різноманітні технічні і програмні засоби, використовувані ІС.

При цьому мається на увазі, що будь-яка ІС, яка відповідає основним стандартам, буде відкрита для взаємозв'язку з будь-якою іншою системою, яка сертифікована на відповідність тим же стандартам.

Це ж стосується і механізмів криптографічного захисту інформації або захисту від несанкціонованого доступу до інформації.

### 10.2 Розподіл ключів і аутентифікація

Основою криптографічного протоколу (у математичному розумінні цього терміна) є деякий криптоалгоритм. На відміну від шифрування, окрім наявності зловмисника, в криптопротоколах передбачається недовіра учасників один до одного.

Можна виділити такі основні класи прикладних криптографічних протоколів:

– генерація ключів: пересилання ключів (перешифрування), узгодження ключів (протоколи типу Діффі-Хеллмана);

– аутентифікація (підтвердження достовірності джерела, даних, ключів);

– підпис (звичайний, затемнений, конфіденційний і т. д.);

– електронні гроші (система протоколів);

– протоколи голосування;

– конфіденційні обчислення.

У попередніх розділах були в тій чи іншій мірі обговорені питання, пов'язані з протоколами перших трьох класів.

Відзначимо декілька аспектів, пов'язаних з останніми протоколами.

Протоколи, що забезпечують реалізацію функції електронні гроші, повинні забезпечувати дві основні властивості: неможливість їх підробки (вочевидь) і невідстежуваність.

Невідстежуваність досягається таким чином. Клієнт звертається в банк, ідентифікує себе. Банк перевіряє, що у клієнта дійсно є рахунок і йому можна видати запитані купюри.

Після цього клієнт передає банку заготівки купюр з номерами і отримує так званий затемнений (сліпий) підпис. З підписаного матеріалу клієнт поступово «витягує» купюри, якими він може розплатитися.

При цьому банк, отримавши ці купюри, не дізнається, в якому саме сеансі він їх підписав. Ось що гарантує протокол зняття з рахунку. Іншими словами: банк завжди впізнає купюру, яку він підписав; але він не зможе (з ймовірністю вищою за просте вгадування) вказати, кому і в якому сеансі він цю купюру видав.

Потрібно відзначити, що всі розрахунки за допомогою купюр, підписаних даним банком-емітентом, використовуються лише в розрахунках клієнтів цього банку між собою (і, звичайно, з самим банком).

Протоколи голосування застосовуються для проведення різних виборів. Зокрема, з їх допомогою гарантується перевірка правильності підрахунку голосів.

Завдання забезпечення конфіденційності обчислень формулюється таким чином. Є $N$ процесорів і $N$ аргументів. Кожен процесор «знає» свій аргумент.

Завдання полягає в тому, що вони повинні спільно обчислити задану функцію від цих аргументів, але так, щоб жоден процесор в результаті не «дізнався» нічого про чужі аргументи (не враховуючи те, що виходить з його власного аргументу і знайденого значення функції). Більше того, деякі учасники цього обчислення можуть бути супротивниками, тобто вони прагнутимуть перешкодити правильному обчисленню.

Виявляється, що будь-яку функцію можна обчислити конфіденційно, якщо лише частина супротивників не перевищує деякого порогу.

Такі задачі можуть виникати, коли реалізується деяка програма для порівняно простих, але дуже секретних обчислень. З міркувань секретності розробка частин цієї програми замовляється різним програмістам. Серед них можуть бути «розвідники», які вставлять свої закладки, а також «халтурники», які нароблять помилок.

Криптопротоколи прийнято підрозділяти на такі види:

– арбітражні (arbitrated);

– з суддівством (adjudicated);

– самодостатні (self-enforcing).

У протоколі з арбітражем: арбітр — учасник протоколу, якому інші учасники повністю довіряють, роблячи відповідні дії для завершення чергового кроку протоколу. Це означає, що у арбітра немає особистої

зацікавленості в досягненні тих або інших цілей, що їх переслідують учасники протоколу, і він не може виступити на стороні будь-якого з них. Учасники протоколу беззаперечно слідують всім його рекомендаціям.

Протокол з суддівством для зниження витрат на арбітраж функціонує без арбітражу до тих пір, поки не виникають розбіжності між учасниками. Для вирішення конфліктів між ними використовується особливий арбітр – суддя.

У комп'ютерних протоколах з суддівством передбачається наявність даних, перевіривши які довірена третя особа може вирішити, чи не зшахраював хто-небудь з учасників цього протоколу. Хороший протокол з суддівством також дозволяє з'ясувати, хто саме поводиться нечесно.

Самодостатній протокол не вимагає наявності арбітра для завершення кожного кроку протоколу. Він також не передбачає участі судді для розв’язання конфліктних ситуацій. Цей протокол влаштований так, що, якщо один з його учасників шахраює, інші можуть моментально розпізнати нечесність і припинити виконання подальших кроків протоколу.

Бажано мати універсальний самодостатній протокол для всіх застосувань. Проте на практиці у кожному конкретному випадку доводиться конструювати свій спеціальний самодостатній протокол.

Розрізняють такі види атак на протоколи:

– пасивні, метою яких є аналіз або підслуховування протоколу для витягання необхідної інформації. Відповідає випадку атаки на криптосистеми ciphertext-only attack;

– активні, орієнтовані на зміну протоколу на користь зловмисника, включаючи вставку, видалення або зміну вихідного повідомлення;

– пасивне шахрайство (cheaters) з боку учасників протоколу, що має на меті в рамках протоколу здобуття більшого обсягу інформації;

– активне шахрайство, якщо для досягнення мети попередньої атаки можлива зміна протоколу.

Захист від пасивного шахрайства повинен надавати будь-який протокол незалежно від умов, в які поставлені його учасники.

Захист протоколу від дій декількох активних шахраїв є досить складним завданням. Проте, в деяких умовах його вдається вирішити, надавши учасникам протоколу можливість вчасно розпізнавати ознаки активного шахрайства.

Для успішного проведення сеансу захищеного зв'язку необхідно сформувати разовий або сеансовий ключ. Розглянемо ситуацію, при якій відправлення ключів за допомогою служби спеціального зв'язку утруднено тими чи іншими причинами (відсутність постійного зв'язку між центром генерації ключів і абонентами мережі, великі накладні витрати на пересилання та ін.).

В цьому випадку необхідно «економно» витрачати наявні секретні ключі або навчитися обмінюватися ними по незахищених каналах зв'язку.

Нехай у розпорядженні абонентів захищеної ІС є деякий блоковий алгоритм шифрування, а також в кожного з них є якісний генератор випадкових чисел. У цих умовах можна розглянути такий (навчальний) варіант взаємодії абонентів системи.

1. Абоненти $A$ і $B$ домовляються про криптосистему $A\text{lg}$ і встановлюють деякий довготривалий ключ $TK$ (у літературі він часто називається транспортним ключем або майстер ключа).

2. Абонент $A$ генерує деяке випадкове число $RndA$ (рівне довжині ключа шифрування), разом з особистим ідентифікатором $IA$ шифрує в режимі CBC і направляє абонентові $B$: $A\text{lg}(RndA, IA, TK) \to B$.

3. Абонент $B$ генерує своє випадкове число $RndB$ (за довжиною $RndA$), зашифровує в режимі CBC за допомогою транспортного ключа обидва випадкових числа, свій ідентифікатор $IB$ і посилає їх абонентові $A$: $A\text{lg}(RndB, IB, RndA, TK) \to A$.

4. Абонент $A$ розшифровує отримане повідомлення і переконується, що випадкове число, послане $B$, збігається з результатом.

5. Абонент $A$ з першої половини випадкового числа $RndA$ і другої половини випадкового числа $RndB$ складає свій сеансовий ключ зашифрування. Абонент $B$ скористається, відповідно, другою половиною випадкового числа $RndA$ і першою половиною випадкового числа $RndB$ і складе свій ключ зашифрування.

Перевагою запропонованого протоколу слід вважати високу швидкість його реалізації, за рахунок високої швидкості шифрування за допомогою симетричних алгоритмів, і гарантовану стійкість протоколу в разі нескомпрометованого транспортного ключа.

Недоліків у протоколах подібного типу декілька. Перш за все, стійкість такого протоколу базується на одному ключі і потрібно враховувати можливість його компрометації.

У протоколі присутній пункт секретного поширення ключа, що накладає відомі обмеження на його застосування. Як недолік можна відзначити швидке зростання кількості необхідних ключів залежно від числа абонентів в ІС (практично пропорційно квадрату числа абонентів).

Формальний опис простого протоколу на основі СВК можна подати у вигляді такої послідовності дій.

1. Абоненти $A$ і $B$ домовляються про використовувану криптосистему.

2. Абонент $B$ посилає абонентові $A$ свій відкритий ключ.

3. Абонент $A$ зашифровує повідомлення з використанням отриманого відкритого ключа і посилає його абонентові $B$.

4. Абонент $B$ розшифровує отримане від $A$ повідомлення, використовуючи власний секретний ключ.

Протокол захищеного за допомогою СВК обміну сеансовими ключами носить назву алгоритму рукостискання. Одним з найбільш поширених варіантів його реалізації є алгоритм Діффі-Хеллмана, побудований на базі складності дискретного логарифмування.

Отже, завдання полягає в тому, щоб сформувати загальний сеансовий ключ для абонентів деякої мережі. Виберемо велике просте число $p$ і лишок $a$ великого порядку за модулем $p$.

Практична вимога до числа $p$: розкладання числа $p-1$ на множники повинне містити великий простий множник розміру, приблизно в половину розміру $p$, а розмір повинен перевищувати 512 бітів.

1. Абоненти $A$ і $B$ домовляються про значення $a, p$ (несекретні параметри) і обмінюються ними у відкритому вигляді.

2. Абонент $A$ генерує велике випадкове ціле $x_A$ (секретний ключ) і обчислює свій відкритий ключ $y_A = a^{x_A} (\mathrm{mod}\, p)$. У свою чергу $B$ генерує $x_B$ і обчислює $y_B = a^{x_B} (\mathrm{mod}\, p)$.

3. Абоненти $A$ і $B$ обмінюються відкритими ключами $y_A, y_B$.

4. Абоненти $A$ і $B$ обчислюють загальний ключ $k = y_A^{x_A} = y_B^{x_B} (\mathrm{mod}\, p)$.

Безперечною перевагою даного протоколу слід вважати незалежність системи розподілу ключів від конфігурації і розміру ІС.

До недоліків можна віднести невисоку швидкість роботи криптопротоколу і небезпеку проведення криптоаналітичних атак типу «агент посередині» (Man-in-the-Middle Attack). Один з варіантів даної атаки може бути описаний таким чином.

1. Зловмисник $H$ має можливість «розірвати лінію зв'язку», тобто впровадитися між $A$ і $B$ таким чином, що потік інформації йде за схемою $A \to H \to B$ і назад.

2. Абонент $A$ посилає $B$ свій відкритий ключ. Зловмисник $H$ посилає $B$ свій відкритий ключ від імені $A$.

3. Абонент $B$ посилає $A$ свій відкритий ключ, який також перехоплюється і підміняється власним ключем зловмисника $H$, який діє від імені $B$.

4. Абонент $A$ посилає зашифроване за допомогою відкритого ключа $B$ повідомлення абоненту $B$. Зловмисник $H$, перехопивши повідомлення, спочатку розшифровує його, читає, а потім зашифровує за допомогою відкритого ключа $B$ і посилає $B$.

5. При передачі у зворотному напрямі ситуація повністю повторюється.

Ця атака працює ефективно, якщо у абонентів $A$ і $B$ немає засобу для аутентифікації (перевірки достовірності і легальності) учасників інформаційного обміну.

Простий спосіб боротьби з атакою типу «агент посередині» – це передати телефоном (або іншим каналом, де забезпечується аудіовізуальна впізнанність) результат хешування свого ключа, але дана процедура стає нереальною в разі проведення тисяч з'єднань за хвилину.

Сказане підкреслює актуальність і важливість реалізації автоматичних процедур аутентифікації в мережах передачі даних, коли за умовами функціонування ІС не можна покладатися на особисте знання учасників інформаційного обміну.

Проблема визначення достовірності особи, що отримує доступ до комп'ютера, банкомата, робочого місця автоматизованої системи електронного документообігу традиційно вирішується за допомогою паролів: людина і комп'ютер знають пароль доступу і комп'ютер регулярно запитує його при кожному входженні в систему.

Практично в пам'яті комп'ютера не обов'язково повинен зберігатися істинний пароль. Комп'ютеру досить уміти розрізняти істинні паролі від хибних.

Таку перевірку досить просто реалізувати за допомогою хеш-функції. Для цього в пам'яті комп'ютера замість паролів зберігається результат хешування пароля – його дайджест.

Для кожного прийнятого пароля обчислюється значення хеш-функції, яке порівнюється із значенням, що зберігається в пам'яті. Така процедура практично унеможливлює компрометацію пароля в разі доступу зловмисника до пам'яті комп'ютера.

### 10.3 Розподіл секрету

Протоколи розділення секрету (secret splitting) використовуються для зниження ризику компрометації деяких критичних даних.

Припустимо, необхідно захистити деяку банківську таємницю (можливо, секрет – код доступу до зашифрованих рахунків). З деякою ймовірністю в банку працює «агент конкуруючої фірми».

Для забезпечення безпеки секрету назначається декілька співробітників, які отримують свої частини секрету. Припустимо, що секрет ділиться між чотирма співробітниками і використовуємо такий протокол.

1. Адміністратор генерує три випадкові рядки $S_1$, $S_2$, $S_3$ такої ж довжини, що і сам секрет $S$.

2. Далі обчислюється $S_4 = S \oplus S_1 \oplus S_2 \oplus S_3$.

3. Адміністратор доручає зберігання $S_1$ першому співробітникові, $S_2$ – другому співробітникові і так далі до $S_4$.

4. При необхідності відновлення секрету: $S = S_4 \oplus S_1 \oplus S_2 \oplus S_3$.

У цьому протоколі адміністратор – головна особа, його шахрайство вже ніяк не контролюється аж до моменту відновлення секрету. Недоліком схеми є те, що втрата одним з учасників протоколу своєї частини приведе до неможливості відновлення секрету.

Ефективнішим на практиці є протокол розподілу секрету (secret sharing), запропонований Шаміром. У цій схемі деяка інформація (секрет) ділиться на $n$ частин, званих тінями (shadows) або частинами секрету, так,

щоб будь-яких $t$ частин було достатньо для відновлення секрету. Схема носить назву $(n,t)$ -порогової схеми (threshold scheme).

Схема Шаміра використовує так званий інтерполяційний поліном Лагранжа для системи $m$ різних пар точок $(x_i, y_i)$.

Цей поліном серед поліномів степеня не вище $m$ визначається однозначно і приймає в точках $x_i$ значення $y_i$.

Виберемо випадковим чином секретний набір лишків $(a_1,\ldots,a_{t-1})$, $a_{t-1} \neq 0$ за великим модулем $p$ і утворюємо поліном $Q(x) = a_0 + a_1 x + \ldots + a_{t-1}x^{t-1}$, де $a_0 = S$ (наш секрет).

Для кожного з $n$ учасників протоколу виберемо відповідно попарно різні несекретні лишки $(b_1,\ldots,b_n)$. Обчислимо значення $Q(b_i)$ і розподілимо між користувачами у вигляді частин секрету пари виду $(b_i, Q(b_i))$, $i = 1,\ldots,n$. Тут ми використовуємо те, що степінь полінома $(t-1)$ не обмежує кількість точок $(n)$, в яких ми бажаємо обчислити значення полінома.

Для відновлення секрету скористаємося формулою Лагранжа, згідно з якою многочлен степеня $t-1$ можна відновити за допомогою $t$ попарно різних точок $(b_i, Q(b_i))$, при цьому вільний член обчислюється за формулою:

$$S = a_0 = \sum_{i=1}^{t} Q(b_i) \cdot \prod_{j \neq i} b_j \left(b_j - b_i\right)^{-1} \bmod p\,.$$

Даний вираз дозволяє довільній групі з $t$ користувачів обчислити секрет, а групи, що складаються з меншого числа користувачів, цієї процедури виконати не можуть.

Розглянемо конструкцію протоколу розподілу секрету, що перевіряється, з роботи [42]. Конструкція заснована на складності дискретного логарифмування.

За аналогією зі схемою Шаміра дилер формує секретний випадковий поліном $Q(x) = a_0 + a_1 x + \ldots + a_t x^t$ степеня $t$, де $a_0 = S$ (наш секрет).

Потім він публікує значення $r_i = g^{a_i} \pmod p$, $i = 0,\ldots,t$, де $g$ – елемент великого порядку за модулем $p$. Для кожного $j = 1,\ldots,n$ дилер пересилає значення $S_j = Q(j)$ процесору $P_j$ по захищеному каналу.

Процесор $P_j$ може (не знаючи $Q(x)$) проконтролювати виконання рівності $S_j = Q(j)$, оскільки для $A = a_0 + a_1 j + a_2 j^2 + \ldots + a_t j^t$

$$g^{S_j} = g^{Q(j)} = g^A = r_0 \cdot r_1^j \cdot r_2^{j^2} \cdots r_t^{j^t} \pmod p.$$

Конструкцію протоколу для фази відновлення секрету розглянемо в найбільш простому випадку, коли дилер чесний.

На цій фазі кожен процесор $P_j$ пересилає по захищеному каналу кожному іншому процесору свою частину $S_j$.

Будь-який чесний учасник $P_i$, набувши деякого значення $S_j$ від $P_i$, перевіряє це значення (як описано вище) і відкидає всі частини секрету, що не пройшли перевірку.

Оскільки чесних учасників не менше $t+1$, $P_i$ отримає принаймні $t+1$ правильних частин секрету. Використовуючи процедуру відновлення секрету зі схеми Шаміра, $P_i$ відновить значення $S$.

### 10.4 Стандарти криптопротоколів в Інтернет

Даючи загальне уявлення про процеси, що відбуваються в світі інформаційних технологій, розглянемо лише особливості забезпечення безпеки інформації в мережах на основі TCP/IP протоколів передачі даних (в т. ч. Інтернет).

Інтерес до створення захищених систем документообігу і електронної торгівлі на базі Інтернету зумовлений рядом чинників.

По-перше, на відміну від локальних мереж спеціального призначення, архітектура Інтернет базується на відкритих стандартах, що дозволяє деяким чином оптимізувати закупівлю устаткування і програмного забезпечення на основі співвідношення ціна-якість за рахунок багаточисельних пропозицій від різних фірм, що пропонують свою продукцію на світовому ринку.

По-друге, Інтернет є глобальною мережею, до якої підключена величезна кількість користувачів, що надає практично необмежені можливості для розширення електронного бізнесу.

І, нарешті, Інтернет фактично є загальноприйнятим засобом представлення новітньої продукції і технологій на світовому ринку.

Існує ряд міжнародних комітетів, в основному, з організацій-добровольців, які проводять заходи зі стандартизації технологій в Інтернет.

Завдяки зусиллям цих комітетів, які складають основу Робочої групи інженерів Інтернету (Internet Engineering Task Force, IETF), в Інтернет упроваджені протоколи сімейства TCP/IP для передачі даних, протоколи SMTP (Simple Mail Transport Protocol) і POP (Post Office Protocol) для електронної пошти, а також SNMP (Simple Network Management Protocol) для управління мережею.

Тривалий час опрацювання стандартів в міжнародних організаціях приводить до того, що провідні виробники і споживачі задають стандарти де-факто, своїми замовленнями або покупками. Зокрема, популярні в Інтернет протоколи безпечної передачі даних SSL, SET, IPv6. з'явилися порівняно недавно, як наслідок необхідності захисту коштовної комерційної інформації, і одразу стали стандартами де-факто.

SSL (Secure Socket Layer) розроблений компанією Netscape Communications Corp. як протокол для криптографічного захисту даних між сервісними протоколами (такими як HTTP, NNTP, FTP і ін.) і транспортними протоколами (TCP/IP).

Протокол SSL призначений для безпечної інформаційної взаємодії на рівні клієнт-сервер, включаючи такі вимоги, що:

– при встановленні з'єднання сервер і користувач мають бути взаємно впевненими, що вони обмінюється інформацією з легальними абонентами;

– після встановлення з'єднання весь трафік між сервером і клієнтом має бути надійно захищений від несанкціонованого контролю за ним;

– при обміні інформацією має бути забезпечений дієвий контроль за наявністю випадкових або умисних спотворень.

Перед початком інформаційного обміну в протоколі SSL забезпечується аутентифікація його учасників (server-client authentication), узгоджується алгоритм шифрування і формуються спільні криптографічні ключі.

З цією метою в протоколі використовуються асиметричні криптосистеми, зокрема, RSA. Конфіденційність інформації, передаваної по захищеному з'єднанню, забезпечується шляхом шифрування потоку даних на сформованому загальному ключі з використанням симетричних криптографічних алгоритмів, наприклад, RC2 (з ключем 40 або 128 бітів), RC4 (40 або 128 бітів), DES (з ключем 40 бітів) і ін.

Контроль цілісності передаваних блоків даних виробляється за рахунок застосування кодів аутентифікації повідомлень.

Протокол забезпечує взаємодію з інфраструктурою відкритих ключів (public key infrastructure – PKI) на основі сертифікатів, що задовольняють рекомендації стандарту X.509 ITU-T.

Протокол SSL розглядається як основний захисний протокол для клієнтів і серверів (WWW Browsers and Servers) в мережі Інтернет.

Найбільш перспективним протоколом, призначеним для організації електронної торгівлі через мережу Інтернет, є стандарт **безпечних електронних транзакцій SET** (Secure Electronics Transaction), розроблений компаніями MasterCard і Visa за участю IBM і ін.

SET, на відміну від інших протоколів, дозволяє комплексно вирішувати базові завдання захисту інформації, включаючи забезпечення її доступності, конфіденційності, цілісності і юридичної значущості.

Протокол надає можливість придбання товарів з використанням пластикових карт, на основі найзахищенішого на даний час механізму виконання платежів.

Протокол базується на використанні цифрових сертифікатів в стандарті X.509 (версія 3), які асоціюють власника картки, продавця і банк продавця з рядом банківських установ платіжних систем Visa і Mastercard.

SET забезпечує крос-аутентифікацію рахунку власника картки, продавця і банку продавця для перевірки готовності оплати, цілісність і конфіденційність повідомлення шляхом шифрування критичних даних.

Тому правильніше було б називати SET стандартною технологією або системою протоколів виконання безпечних платежів в Інтернет з використанням пластикових карток.

SET за допомогою криптографічних методів, включаючи цифрові сертифікати, дозволяє покупцям і продавцям встановити достовірність всіх учасників операції, що проводиться в Інтернет.

Протокол також забезпечує виконання таких спеціальних вимог до захисту операцій електронної комерції:

– конфіденційність даних оплати та інформації замовлення, переданої разом з даними про оплату;

– цілісність даних платежів на основі застосування ЕЦП;

– аутентифікацію власника кредитної картки завдяки використанню ЕЦП і сертифіката власника картки;

– аутентифікацію продавця і його можливості приймати платежі за допомогою кредитних карток із застосуванням ЕЦП і сертифіката продавця;

– аутентифікацію того, що банк продавця є легітимною установою, яка може приймати платежі за допомогою кредитних карток через зв'язок з процесинговою картковою системою. Аутентифікація банку продавця забезпечується використанням цифрового підпису і сертифікатів банку продавця;

– готовність до проведення транзакцій (оплати) завдяки аутентифікації сертифікатів відкритих ключів всіх сторін, що беруть участь.

У 1992 році робоча група IETF виступила з ініціативою з розробки вимог до протоколів сімейства TCP/IP нового покоління. Після обговорення декількох версій в 1995 році були опубліковані основні вимоги до архітектури побудови і принципи функціонування засобів забезпечення безпеки.

Протоколам TCP/IP нового покоління (IPv6) властиві такі особливості: розширений адресний простір; покращені можливості маршрутизації; можливість управління доставкою інформації; покращені засоби забезпечення безпеки, з використанням ефективних алгоритмів аутентифікації і шифрування.

Специфікація IPsec додаткова відносно поточної версії протоколів TCP/IP і така, що входить в стандарт IPv6, розроблена робочою групою IP Security тематичної групи IETF.

IPsec включає три алгоритмо-незалежні базові специфікації, опубліковані як RFC-документи (Інтернет-стандарти). IPsec підтримує DES, MD5 і ряд інших криптографічних алгоритмів.

Протокол IPsec, передбачає наскрізне шифрування трафіку (від абонента до абонента) на третьому рівні IP-рівні з'єднань (див. рис. 10.1). По суті, IPsec забезпечує роботу понад зв'язних протоколів.

При цьому кожен пакет, який проходить по каналу, шифрується незалежно від додатка. Це дозволяє організації створювати в Інтернет віртуальні приватні мережі VPN.

Перевагами забезпечення безпеки інформації за допомогою IPsec є:

– можливість створення віртуальних захищених мереж «над» небезпечним середовищем;

– підвищена безпека від аналізу трафіку за рахунок захисту заголовка транспортного рівня від перехоплення;

– захист від атак типу «відмова в обслуговуванні»;

– підтримка відмінних від TCP протоколів.

IPsec має ще дві важливі якості - його застосування не вимагає зміни проміжних пристроїв в мережі, робочі місця і сервери необов'язково повинні підтримувати IPsec.

IPsec для забезпечення комплексної інформаційної безпеки використовує декілька криптографічних методів:

– відкритий обмін ключами через мережу на основі алгоритму “рукостискання” Діффі-Хеллмана;

– застосування цифрового підпису на основі СВК;

– шифрування даних за допомогою блокового криптоалгоритму;

– використання хеш-функцій для перевірки достовірності пакетів.

### 10.4.1 Формальний аналіз криптографічних протоколів

Криптографічні протоколи, призначені для аутентифікації і обміну ключами, відіграють дуже важливу роль в забезпеченні надійного функціонування сучасних комп'ютерних систем.

Криптологами було створено багато таких протоколів, однак з часом з'ясувалося, що не кожний з них дозволяє належним чином захистити комп'ютерну систему від несанкціонованого доступу і запобігти компрометації ключів. Причому для деяких протоколів цей часовий проміжок виявився досить значним і розтягнувся на декілька років.

Як наслідок, виникла нагальна потреба в розробці формальних методів, які дозволили б знаходити вади в криптографічних протоколах в

коротші терміни. І хоча більшість з цих методів є універсальними і можуть бути використані для аналізу довільних криптографічних протоколів, особлива увага спочатку стала приділятися саме протоколам, що дозволяють виконувати аутентифікацію користувачів і обмін ключами.

На сьогодні є чотири основні підходи до аналізу криптографічних протоколів. Перший з них застосовує верифікаційні методи, які не були спеціально розроблені для аналізу криптографічних протоколів. Деякі з цих методів подають протоколи у вигляді кінцевих автоматів, інші поширюють на них теорію обчислення предикатів першого порядку.

Другий підхід полягає у використанні експертних систем, що дозволяють визначити, чи виникають в ході виконання кроків протоколу такі небажані події, як компрометація ключа або розголошування пароля. Цей підхід дає можливість ефективніше знаходити в криптографічних протоколах конкретні вади, проте жодним чином не гарантує їх повної відсутності.

Авторами третього підходу є американські криптологи М. Берроуз, М. Абаді, Р. Нідхем (M. Burrows, M. Abadi, R. Needham). Вони розробили формальну логічну модель, названу за першими буквами їх прізвищ БАН-логікою (BAN-logic).

Скласти з її допомогою загальне уявлення про надійність криптографічного протоколу не можна. Проте основна перевага БАН-логіки над іншими підходами полягає в тому, що вона складається з набору простих логічних правил, які легко застосовуються на практиці і вельми корисні у виявленні окремих вад в аналізованих протоколах.

У четвертому підході використовується моделювання кроків протоколу за допомогою алгебраїчної системи, яка будується виходячи зі знань, що є у учасників протоколу відносно цього протоколу. Потім побудована алгебраїчна модель піддається формальному аналізу на предмет досяжності деяких з її станів.

В цілому формальний аналіз криптографічних протоколів поки що є досить новою і до кінця не сформованою областю криптології, і тому робити далекоглядні висновки про переваги якогось одного з перерахованих чотирьох підходів ще рано.

**Питання до розділу 10**

1. Які вимоги висуваються до криптографічних протоколів?
2. Що розуміється під відкритими системами і чим регламентується їх взаємодія?
3. Окрім наявності зловмисника, які ще припущення висуваються відносно учасників криптопротоколу?
4. Які основні класи прикладних криптографічних протоколів?
5. Чим протокол з арбітражем відрізняється від протоколу з суддівством?
6. Що таке самодостатній протокол?
7. Які основні види атак на криптопротоколи?
8. На чому заснована стійкість абстрактного протоколу Діффі-Хеллмана узгодження ключів?
9. Яким чином можна скомпрометувати на практиці протокол Діффі-Хеллмана за допомогою атаки типу «агент всередині»?
10. У чому полягає задача розподілу секрету?
11. У чому полягає основна властивість $(n,t)$-порогової схеми Шаміра щодо відновлення секрету?
12. Яким чином для схеми Шаміра гарантується неможливість відновлення секрету зловмисником?
13. Чому група кількістю менше $t$ легальних учасників не в змозі відновити секрет, захищений $(n,t)$-пороговою схемою Шаміра?
14. Для чого призначені протоколи типу SSL?
15. Якими криптографічними методами забезпечується конфіденційність і цілісність даних в SSL?
16. Якого типу сертифікати використовуються SSL для організації взаємодії з інфраструктурою відкритих ключів?
17. Яке призначення системи протоколів безпечних електронних транзакцій SET?
18. Які основні вимоги до захисту операцій електронної комерції забезпечує протокол SET?
19. Для чого призначений протокол IPsec і чи впливають на його роботу зв'язні протоколи, що діють в мережі?
20. Які основні переваги забезпечення безпеки інформації на основі IPsec?

# РОЗДІЛ 11 АРХІТЕКТУРА СИСТЕМИ ЕЦП

## 11.1 Архітектура системи ЕЦП

Під архітектурою системи ЕЦП ми розумітимемо принципи побудови апаратно-програмних комплексів, що забезпечують функціонування засвідчувальних центрів, за допомогою яких корпоративні і індивідуальні користувачі можуть працювати з сертифікатами відкритих ключів [25, 26]. При цьому користувачам надаються комплексні рішення в області створення інфраструктури відкритих ключів (Public Key Infrastructure – PKI), описаної робочою групою PKIX (Public Key Infrastructure Working Group).

**Основними характеристиками архітектури системи ЕЦП є модульність, модель довіри і масштабовність.** Не дивлячись на відсутність строгих визначень цих понять, користуючись ними як критеріями можна в цілому оцінити якість рішень, реалізованих в програмних продуктах провідних виробників світу.

На світовому ринку програмних продуктів, що підтримують інфраструктуру PKI, представлені такі рішення, що набули найбільш широкого поширення:

– Entrust Authority 6.0 (далі – Entrust);
– iPlanet Certificate Management System 4.7 (далі – iPlanet);
– Microsoft Certificate Server 2.0 (далі – MCS);
– RSA Keon 5.7 (далі – RSA);
– UNICERT PKI 3.5.3 (далі – UNICERT).

**Модульність.** При порівнянні систем ЕЦП за критерієм модульності слід враховувати склад вибраних рішень, який повинен забезпечити повнофункціональну роботу інфраструктури відкритих ключів.

Ділення продуктів, що реалізовують системи ЕЦП, на сервіси (послуги, модулі) досить умовний. Разом з тим, при аналізі архітектури різних систем можна виділити деякі загальні риси, у тому числі:

*Сервіс реєстрації*, що забезпечує підтримку процедур обслуговування користувачів систем ЕЦП, включаючи:

– прийняття заявок на реєстрацію;

– перевірка даних користувача, аутентифікація і контроль повноважень;

– створення заявок на сертифікацію;

– запис даних на електронні носії (смарт-карти, флеш-пам'ять і так далі);

– консультаційна підтримка.

*Сервіс сертифікації*, орієнтований на виконання процедур обслуговування сертифікатів відкритих ключів користувачів, у тому числі:

– перевірка надходження запитів на сертифікацію;

– формування сертифіката;

– відправлення сертифіката до сервісу баз даних;

– створення переліку блокованих і скасованих сертифікатів.

*Сервіс ведення баз даних*, що забезпечує зберігання і публікацію сертифікатів, а також списків блокованих або скасованих сертифікатів.

*Сервіс статусу сертифіката (on-line certificate status protocol - OCSP)*, що входить в режимі on-line до складу сервісу баз даних і забезпечує:

– створення повідомлень про статус сертифікатів;

– підтримку інфраструктури відкритих ключів PKI.

*Сервіс фіксації часу*, орієнтований на:

– створення відміток часу для документів і повідомлень;

– засвідчення існування документа або повідомлення до певного часу.

*Сервіс блокування і відміни* включає функції:

– перевірки запитів на блокування або відміну сертифіката;

– передачі вказівок на блокування або відміну в сервіс сертифікації.

У центри сертифікації ключів запити на блокування або відміну сертифікатів можуть надходити по телефону або у вигляді письмового повідомлення за допомогою електронної пошти (e-mail), а також звичайного поштового повідомлення.

Підставами для блокування або відміни можуть бути компрометація секретного ключа внаслідок його втрати або крадіжки носія інформації, завершення терміну роботи користувача в системі або зміна даних сертифіката на основі рішення клієнта.

*Служба управління ключами* забезпечує:

– управління Root-ключем (кореневий ключ);

– управління ключами для Сервісу ведення баз даних і Сервісу фіксації часу;

– генерацію відкритих і секретних ключів.

**Протоколи сервісу баз даних,** у тому числі:

а) Протокол опитування відкритих сертифікатів (LDAP);

б) Протокол відомостей про статус (OCSP):

1) протокол HTTP;

2) відомості про унікальність запитаного сертифікатом переліку;

3) в разі унікальності - перевірка на блокування або відміну;

4) підлягають перевірці за допомогою посиленого цифрового підпису.

в) Список блокування або відмін (LDAP):

1) перевірка за допомогою посиленого цифрового підпису, можливий on-line доступ.

Найпривабливішими при порівнянні за критерієм «модульність» виглядають вирішення Entrust і UNICERT, що надають модулі для всіх областей використання інфраструктури відкритих ключів.

Зокрема, наявність сервісів тимчасової мітки, підтримки «бродячих» користувачів і протоколу безпровідних додатків, призначеного для безпечних безпровідних транзакцій, робить ці рішення унікальними за даним пунктом порівняння.

**Модель довіри.** Існує декілька моделей довіри для процедур сертифікації відкритих ключів, яким властиві свої переваги і недоліки. Модель довіри – це основа, на якій будується взаємодія різних систем PKI.

Існує дві моделі довіри, схематично зображені на рис. 11.1 і на рис. 11.2:

– **ієрархічна** модель, що встановлює єдину точку довіри двом або більше Центрам сертифікації ключів (ЦСК - Certificate Authority, CA);

– **мережева модель**, що визначає прямі довірчі стосунки між двома або більше ЦСК і що реалізовує так званий метод крос-сертифікації. У цій моделі кожен з ЦСК вважається підпорядкованим центру, з яким встановлені довірчі стосунки.

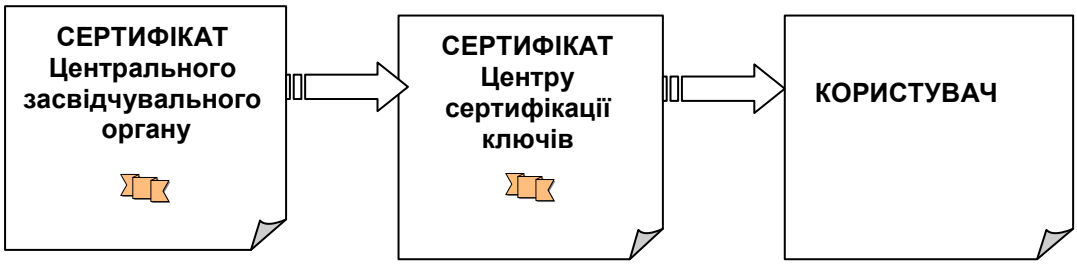



Рисунок 11.1 - Ієрархічна модель довіри

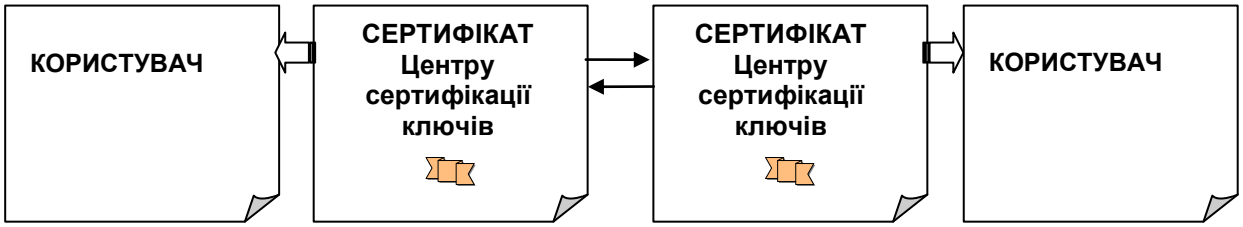



Рисунок 11.2 - Мережева модель довіри

Основні відмінності між двома моделями довіри полягають в методах побудови ланцюгів сертифікатів і методах верифікації сертифікатів, виданих різними ЦСК.

Даний критерій відображає можливість різних рішень PKI взаємодіяти між собою без додаткових витрат і найбільш критичний для корпорацій, які об'єднують в своїй інфраструктурі декілька рішень або розширяють свої підрозділи.

Краще за цими параметрами виглядають продукти Entrust, RSA і UniCert, які забезпечують гібридну модель довіри.

MCS та iPlanet підтримують лише ієрархічну модель довіри.

Законом України «Про електронний цифровий підпис» визначена як основа побудови національної системи ЕЦП найбільш поширена ієрархічна модель сертифікації відкритих ключів з єдиним коренем. Законом також накладено обмеження на глибину можливої сертифікації.

Для більшості додатків це – дворівнева схема, що включає центральний засвідчувальний орган (ЦЗО – кореневий засвідчувальний орган) і окремі центри сертифікації ключів.

При необхідності, в державних органах передбачається можливість створення трирівневої системи сертифікації: центральний засвідчувальний орган – засвідчувальний центр – центр сертифікації ключів.

Перевагою ієрархічної моделі порівняно з мережевою є простіший процес перевірки сертифіката користувача.

Недоліком цієї архітектури є те, що в разі порушення роботи кореневого центру сертифікації ключів (ЦСК) практично порушується робота всієї системи.

Це означає, що в разі компрометації секретного ключа ЦСК всі сертифікати, які були завірені ЦСК з його допомогою, вважаються компрометованими і підлягають заміні.

Відмічений недолік вимагає особливої уваги до питань забезпечення безпеки як на етапі проектування ЦСК, так і в ході його практичної експлуатації.

**Масштабовність.** Критерій масштабовності часто дуже вільно трактується виробниками тих або інших продуктів. Характеристика цього показника типу: «висока масштабовність – 1 мільйон сертифікатів», що швидше свідчить про вельми середню базу даних, ніж про якість самого рішення PKI.

Визнано, що масштабовне рішення повинне задовольняти викладені нижче умови.

1. Мати модульну структуру, що забезпечує побудову повнофункціональної інфраструктури відкритих ключів.

2. Підтримувати різні програмні платформи – операційні системи.

3. Підтримувати відкриті криптографічні стандарти для взаємодії з іншими системами.

4. При необхідності взаємодії різних реалізацій PKI підтримувати гібридну модель довіри.

5. Забезпечувати підтримку побудови довірчих відносин з центром сертифікації, що не входить в наявну ієрархію, в реальному масштабі часу. Дана умова мінімізує витрати на побудову, управління і технічну підтримку інфраструктури PKI. Таку умову задовольняють рішення Entrust, UNICERT і MCS.

6. Забезпечувати гнучку систему управління і контролю над інфраструктурою PKI. Дана вимога обумовлена тим фактором, що проблеми в роботі інфраструктури відкритих ключів можуть привести до серйозних фінансових втрат, тому дуже важливо у складі програмного продукту мати засоби реагування на аварійні ситуації, проведення аналізу стану системи і отримання необхідних звітів.

7. Підтримувати інфраструктуру управління привілеями (PMI), що обумовлено все більш інтенсивним використанням в корпоративних середовищах можливостей вживання сертифікатів визначальних ознак (Attribute certificates) для передачі сервісам авторизації інформації про призначені для користувача привілеї. Такий сервіс надають рішення Entrust, RSA і UNICERT.

В цілому, при порівнянні за критерієм масштабовності найкращими виглядають рішення Entrust і UNICERT. Рішення RSA і iPlanet трохи відстають за критеріями модульності і підтримки побудови довірчих відносин в реальному масштабі часу.

Крім того, рішення iPlanet, як і рішення MCS, не має підтримки інфраструктури управління привілеями (PMI). Рішення Microsoft можна назвати масштабовним виключно в межах платформи Windows.

### 11.2 Управління сертифікатами і ключами

Управління сертифікатами передбачає випуск, публікацію, відгук і припинення дії сертифікатів. Гнучкість підходів до управління сертифікатами сприяє оптимальному вибору рішення, відповідного потребам корпорації.

У цьому питанні найбільшого успіху досягли рішення Entrust і UNICERT, що забезпечують управління сертифікатами за допомогою Web і e-mail.

Рішення iPlanet, Microsoft і RSA надають можливість отримання сертифікатів лише за допомогою web інтерфейсу, хоча слід зазначити, що MCS на рівні підприємства може випускати сертифікати автоматично (auto-enrollment).

Видалення або припинення дії сертифікатів у рішеннях iPlanet і RSA здійснюється за допомогою Web або адміністратора. MCS пропонує видаляти сертифікати лише використовуючи адміністративну консоль.

При порівнянні методів публікації сертифікатів слід зазначити, що всі продукти підтримують публікацію в Сервісах каталогів і Active Diretory за протоколом LDAP, продукти UNICERT і Entrust так само підтримують DAP.

Користувачі інфраструктури відкритих ключів мають бути впевнені, що на момент використання сертифікат чинний.

В разі компрометації сертифіката або криптографічних ключів сертифікат має бути негайно відкликаний ЦСК і внесений до списку відкликаних сертифікатів, доступного для всіх користувачів.

Перевірка статусу сертифіката може здійснюватися за допомогою списків відкликаних сертифікатів (CRL), або змінених списків відкликаних сертифікатів (Delta CRL), а також шляхом визначення статусу сертифіката в реальному часі.

У розширеннях сертифікатів знаходиться уніфікований покажчик інформаційного ресурсу – стандартизований рядок символів, який вказує адресу документа в мережі Internet з інформацією про місце розташування списків. Покажчик інформаційного ресурсу може бути асоційований з ресурсом LDAP, HTTP, FTP або X.500. На підставі цієї інформації користувач може отримувати актуальні стани списків.

При порівнянні за можливостями публікації CRL складається майже рівна ситуація: списки відкликаних сертифікатів, за винятком MCS, підтримуються в усіх продуктах шляхом публікації вручну або періодично у файлову систему (HTTP і FTP) або Сервіс директорій (Active Directory) за допомогою протоколу LDAP. У продукті MCS можлива лише періодична публікація.

Змінений список відкликання сертифікатів містить лише сертифікати зі зміненим статусом.

Перевагами Delta CRL в порівнянні з CRL є істотно менший об'єм, ніж у базового списку відкликаних сертифікатів, частіша публікація і невелика затримка оновлення статусу відкликання у клієнта, мінімальне навантаження на мережу.

Змінені списки відкликання сертифікатів підтримують Entrust і MCS.

Протокол визначення статусу сертифіката в реальному часі (OCSP – on-line certificate status protocol) заснований на використанні служби мережевих каталогів, яка надає інформацію про статус сертифіката в режимі реального часу.

В цьому випадку сертифікати, видані ЦСК, вважаються недійсними до тих пір, поки інформація про їх статус не буде вибрана з каталогу, що підтримується центром сертифікації. Розширення сертифікатів містять значення, що надає місцерозташування сервісу OCSP.

Лише один продукт надає власний повноцінний сервіс визначення статусу сертифіката в реальному часі – iPlanet. Рішення RSA обмежилося сервісом OCSP (в термінах RSA – OCSP Responder), що підтримує лише http запити. Останні продукти використовують надбудову у вигляді програмного продукту ValiCert Global VA Service, що забезпечує сервіс OCSP.

При порівнянні за критерієм управління сертифікатами всі продукти мають рівні показники, слід лише відзначити можливості випуску сертифікатів рішень Entrust і UNICERT, а також сервер OCSP, що входить в поставку iPlanet.

### 11.2.1 Резервне зберігання пар ключів

Вимога використання різних пар криптографічних ключів для шифрування і ЕЦП в продуктах PKI характеризує підхід постачальника до питання розмежування даних. Строге ділення даних корпорацій і даних користувача сприяє підвищенню безпеки системи.

З даних рішень різні пари ключів для шифрування і підпису підтримують лише Entrust, iPlanet і RSA.

Можливість відновити зашифровані дані – природна необхідність корпорації щодо забезпечення безпеки інформації.

За цим параметром всі рішення показали себе однаково добре, відрізнялася лише реалізація, зокрема:

1) у ідеології Entrust спочатку закладено резервування пар ключів, призначених для шифрування;

2) iPlanet використовує Data Recovery Manager;

3) MCS використовує модуль Key Management Server;

4) у RSA Keon для зберігання ключів використовується Security Server;

5) у UNICERT PKI існує додатковий модуль KAS, що використовує Oracle як базу даних.

## 11.3 Управління інфраструктурою відкритих ключів (PKI)

Засоби управління PKI – це основний інструмент спілкування системи і адміністратора. Від того, наскільки цей інструмент універсальний і зручний, інколи залежить працездатність всієї системи.

Відповідно до вимог Профілю захисту PKI, в управлінні інфраструктурою ОК необхідно розділяти ролі адміністратора системи, адміністратора політики безпеки, аудитора і кінцевого користувача.

В усіх рішеннях, за винятком Microsoft, існує жорстке розділення функцій щодо управління інфраструктурою. У реалізації Microsoft лише адміністратор має право управляти PKI (за прес-релізами Microsoft в Windows.NET ця проблема вже усунена).

Управління і моніторинг інфраструктурою здійснюється у ряді продуктів таким чином:

а) Entrust:

1) за допомогою адміністративної консолі Entrust/RA;

2) з командного рядка;

3) можливе виконання деяких адміністративних операцій за допомогою AutoRA (з використанням challenge-response pages);

4) Web/VPN/WAPConnector – сімейство модулів, що забезпечують можливість випуску сертифікатів для Web оглядачів, Web серверів,VPN і WAP пристроїв;

б) iPlanet:

1) з адміністративної Java консолі;

2) з командного рядка;

3) через web-интерфейс;

в) MCS: стандартна консоль управління (mmc), розширена необхідними вставками (Snap-in);

г) RSA Keon: через web-интерфейс;

д) UniCERT:

1) за допомогою модулів CAO і RAO, кожен з яких призначений для управління і моніторингу вищестоящого модуля, як CA і RA (присутній унікальний редактор інфраструктури Policy Editor);

2) управління здійснюється центром реєстрації через Web за допомогою додаткового модуля WEBRAO.

З даної групи продуктів варто виділити: Entrust, UniCERT і iPlanet, які надали потужні і всеосяжні способи управління і контролю за системою.

### 11.3.1 Управління політиками

Політики – правила, що призначаються, які перевіряють достовірність вмісту запитів на сертифікацію і визначають вміст видаваного сертифіката. Політики також визначають правила прийому, відхилення, перенесення запитів і оновлення раніше виданих сертифікатів.

Політика ЦС – офіційний документ. Його розділи складаються з підрозділів, в яких описані положення, на основі яких забезпечується надійне функціонування ЦС і якісне обслуговування його користувачів [24].

У документі можуть бути присутні розділи, які не доводяться до відома користувачів, але користувачі мають можливість в повному обсязі познайомитися з політикою ЦС, в частині умов їх обслуговування, прав, а також передбачених санкцій.

У міжнародному стандарті X.509 політика використання сертифікатів визначається як встановлений набір правил, що характеризують можливість застосування сертифіката певним співтовариством користувачів чи деяким класом прикладних програм з обумовленими вимогами до безпеки.

Політика застосування сертифікатів, об’явлена даним ЦС, дозволяє користувачеві оцінити надійність і ефективність використання відповідних сертифікатів для конкретного прикладного об'єкта.

Гнучкість і можливість зміни існуючих політик – основні вимоги, що висуваються до політиків на корпоративному рівні. Всі продукти надають дану можливість.

### 11.3.2 Реалізація засобів аудиту і зберігання налаштувань в PKI

При проектуванні і побудові Інфраструктури Відкритих Ключів слід брати до уваги особливості реалізації зберігання параметрів і налаштувань PKI. Так само, відповідно до профілю захисту, необхідно

запобігти доступу неавторизованих користувачів до конфігураційної інформації інфраструктури.

Всю необхідну для функціонування інформацію Entrust/PKI зберігає в зашифрованій базі даних Informix/Oracle, використовуючи при цьому систему безпеки сервера бази даних.

Рішення iPlanet і RSA використовують власну базу даних, MCS – Active Directory.

Всю необхідну для функціонування інформацію UniCERT зберігає в базі даних Oracle, використовуючи при цьому систему безпеки сервера бази даних.

При порівнянні за цим параметром ми отримуємо досить схожу картину для всіх продуктів, слід лише відзначити можливість шифрування бази даних в Entrust і RSA.

Засоби аудиту і складання звітів – невід'ємна частина інфраструктури будь-якої інформаційної системи і тим більше такої складної і розгалуженої, як PKI.

При порівнянні за цим критерієм слід виділити три рішення Entrust, UNICERT і RSA, які забезпечують повнофункціональні засоби аудиту і засобів створення звітів. iPalnet і MCS не мають власних засобів створення звітів. Всі з представлених продуктів мають можливість перенаправляти потік подій в журнали подій операційних систем, на яких вони працюють, що дає можливість здійснювати контроль за станом системи і захистити їх від несанкціонованого доступу.

В цілому при порівнянні за критерієм Управління Інфраструктурою слід зазначити рішення Entrust і UniCERT, що мають зручні і різноманітні засоби управління, аудиту і подання звітів.

### 11.4 Проект системи, центри сертифікації ключів

Складність і різноманіття завдань, що вирішуються в ході створення системи ЕЦП, настійно вимагають попереднього практичного опрацювання окремих компонентів і інтеграції рішень у вигляді невеликого за об'ємом проекту. Розглянемо низку запитань, які необхідно враховувати при його розробці.

Основні положення зі створення і функціонування системи ЕЦП визначені Законом України «Про електронний цифровий підпис».

Закон визначає, що ЕЦП прирівнюється за своїм правовим статусом до власноручного підпису в тому випадку, якщо:

1) ЕЦП підтверджений використанням посиленого сертифіката з використанням надійних засобів цифрового підпису;

2) в ході перевірки використовувався посилений сертифікат ключа, що діяв у момент формування ЕЦП;

3) особистий ключ того, хто підписує, відповідає відкритому ключу, вказаному в сертифікаті.

На основі Закону розроблені і затверджені постановами КМ України повноваження і обов'язки основних елементів системи: центрального засвідчувального органу (ЦЗО), засвідчувального центру (ЗЦ), центру сертифікації ключів (в т.ч. акредитованого).

Нагадаємо, що на центри сертифікації покладаються завдання надання послуг ЕЦП і обслуговування сертифікатів ключів, реєстрації користувачів системи і генерації ключів. ЦЗО і ЗЦ виконують функції кореневого ЦС і ЦС другого рівня відповідно.

Відповідно до законодавства система ЕЦП в Україні повинна мати ієрархічну структуру, що включає кореневий центр сертифікації ключів, корпоративний центр сертифікації ключів і центр сертифікації ключів другого рівня.

На кореневий ЦС покладається завдання обслуговування виключно кореневих сертифікатів (він не обслуговує клієнтів).

Кореневий ЦС повинен забезпечити виконання таких функцій:

– генерацію власної ключової пари і формування власного самопідписувального сертифіката;

– реєстрацію центрів сертифікації ключів другого рівня;

– формування кореневих сертифікатів ключів;

– ведення основних і резервних баз даних, зокрема, реєстраційної інформації підлеглих ЦС, кореневих сертифікатів ключів (запити на кореневі сертифікати, повідомлення про відміну або блокування сертифікатів, журнали подій і так далі), резервне копіювання баз даних на зовнішні носії;

– публікацію кореневих сертифікатів на власних інформаційних ресурсах для забезпечення вільного доступу до них (HTTP, FTP, Active Directory);

– блокування або відміну сертифікатів;

– перевірку легітимності сертифікатів.

Корпоративний центр сертифікації ключів щодо ЦС нижнього рівня виконує функції кореневого, сертифікат його відкритого ключа завіряється підписом кореневого ЦС.

Центр сертифікації ключів другого рівня обслуговує користувачів системи. Сертифікат його відкритого ключа завіряється підписом кореневого ЦСК або корпоративного центру сертифікації ключів.

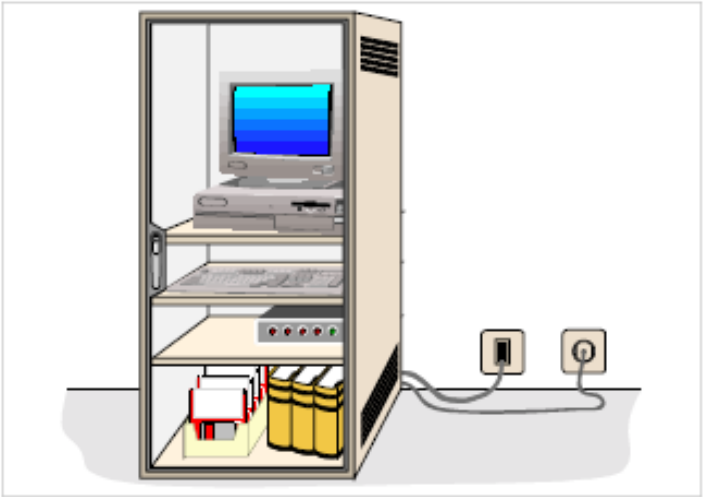


Рисунок 11.3 - Центр сертифікації ключів

ЦС другого рівня забезпечує виконання таих функцій:

– генерацію ключів;

– реєстрацію користувачів і обслуговування запитів на формування сертифікатів;

– формування сертифікатів ключів;

– запис секретних ключів;

– ведення основних і резервних баз даних, забезпечення резервного копіювання;

– блокування і відміну сертифікатів;

– публікацію чинних і відкликаних сертифікатів;

– перевірку легітимності сертифікатів;

– формування відміток часу.

Користувацькі АРМ виконують такі функції:

– генерацію пар ключів;

– перевірку дієвості сертифіката;

– формування електронного цифрового підпису, тимчасової відмітки.

Ефективність створюваної системи захисту інформації багато в чому залежить від того, наскільки плановані заходи і використовувані засоби взаємно погоджені і взаємно доповнюють один одного.

Ігнорування цього постулату може, з одного боку, привести до надмірних витрат на придбання дорогого устаткування, з іншого боку – до появи «проломів в оборонних редутах».

Одними з найбільш простих і одночасно вельми ефективних заходів забезпечення безпеки інформації є організаційні заходи або заходи обмежувального характеру. Вони передбачають, перш за все, регламентацію доступу в режимні приміщення і облік дій персоналу, що працює з телекомунікаційним устаткуванням і обчислювальними системами, в яких накопичується, обробляється і передається інформація з обмеженим доступом.

### 11.4.1 Акредитація центру сертифікації

Відповідно до Закону України «Про електронний цифровий підпис» акредитація – це процедура документального засвідчення компетентності центру сертифікації ключів здійснювати діяльність, пов'язану з обслуговуванням посилених сертифікатів ключів.

В цілому процедура акредитації має бути визначена постановою КМ України, проте вже можна говорити про основні вимоги до АЦСК, які коротко розкриваються простою формулою «знати, уміти і мати»:

– знати вимоги нормативних документів про порядок здійснення діяльності у вибраній області, принципи і методи захисту, характеристики експлуатованого устаткування;

– уміти безпечно експлуатувати засоби КЗІ;

– мати необхідні для здійснення заявленого виду діяльності **сертифіковані засоби КЗІ,** устаткування, персонал, приміщення, регламентувальні документи.

### 11.4.2 Сертифікація і допуск до експлуатації

Існуюча нормативно-правова база визначає, що для захисту конфіденційної інформації використовуються сертифіковані засоби криптографічного захисту інформації (КЗІ). Захист інформації з обмеженим доступом, що є власністю держави, здійснюється за допомогою засобів КЗІ, що мають допуск до експлуатації.

Сертифікація СКЗІ здійснюється з метою підтвердження їх відповідності стандартам ГОСТ 28147-89, ГОСТ 34.310-95, ГОСТ 34.311-95, ДСТУ 4145-2002, а також іншим нормативним документам і технічним умовам на ці засоби. Порядок проведення сертифікації засобів КЗІ регламентований нормативними документами Системи УкрСЕПРО.

Допуск до експлуатації засобів здійснюється на основі позитивних результатів експертизи у сфері КЗІ, яка проводиться згідно з Положенням про державну експертизу у сфері криптографічного захисту інформації.

Процедури допуску до експлуатації (експертизи) і сертифікації багато в чому дуже схожі, разом з тим вони мають і ряд відмінностей.

Перелік об'єктів допуску до експлуатації (експертизи) значно ширший, ніж в процедурі сертифікації. На відміну від сертифікації, об'єктами експертизи окрім засобів КЗІ можуть бути:

– засоби і методи, призначені для розробки, дослідження, виробництва і випробувань засобів КЗІ;

– звіти про наукові дослідження і розробки, інші результати наукової і науково-технічної діяльності у сфері КЗІ;

– криптографічні алгоритми і протоколи, методи генерації ключової інформації;

– криптографічні системи, засоби і устаткування КЗІ;

– положення програм і проектів державного значення в частині, що стосується КЗІ;

– системи і засоби генерації, тестування і розподілу ключів.

Таким чином, сфера застосування експертизи істотно ширша, ніж сертифікації. Зокрема, експертиза у сфері КЗІ надає можливість в тому або іншому випадку визначити можливість використання для захисту інформації конкретного криптоалгоритму і надати рекомендації відносно його застосування.

Наприклад, об'єктом експертизи може бути як криптоалгоритм, так і засіб КЗІ, які реалізують алгоритми, що не є державними (міждержавними) стандартами.

Процедури сертифікації в системі сертифікації продукції УкрСЕПРО є відкритими і загальнодоступними.

**Питання до розділу 11**

1. Що розуміється під архітектурою системи електронного цифрового підпису (ЕЦП)?
2. Які характеристики архітектури системи ЕЦП є основними?
3. Які основні послуги відносять до сервісу реєстрації системи ЕЦП?
4. Які основні процедури відносять до сервісу сертифікації?
5. Яке призначення сервісу ведення баз даних і для чого використовуються протоколи LDAP і OCSP?
6. Які функції виконує сервіс блокування і відміни?
7. Які завдання служби управління ключами?
8. У чому відмінність між ієрархічною і мережевою моделями довіри?
9. Які вимоги до масштабовної архітектури системи ЕЦП?
10. Що таке політика центру сертифікації?
11. Яким чином визначається статус ЕЦП в Законі України «Про електронний цифровий підпис»?
12. Яким чином організовується система ЕЦП відповідно до законодавства України?
13. Для чого призначений центр сертифікації ключів другого рівня?
14. У чому полягає акредитація центру сертифікації ключів?
15. У чому полягає сертифікація і допуск до експлуатації засобів КЗІ?

# ЛІТЕРАТУРА


1. Тилборг ван Х. К. А. Основы криптологии / Тилборг ван Х. К. А. – М. : Мир, 2006. - 471 с.
2. Основы криптографии / [Алферов А. П., Зубов А. Ю., Кузьмин А. С., Черемушкин А. В.]. – М. : «Гелиос АРВ», 2001. –480 с.
3. Бабаш А. В. Криптография (аспекты защиты) / А. В.Бабаш, Шанкин Г. П. – М. : СОЛОН-Р, 2002. – 512 с.
4. Ленков С. В. Методы и средства защиты информации / Ленков С. В., Перегудов Д. А., Хорошко В. А. – К. : АРИЙ, 2008, Том I – 464с., Том II – 344 с.
5. Харин Ю. С. Математические основы криптологии / Харин Ю. С. Берник В. К. Матвеев Г.В. – Минск, БГУ, 1999. – 319 с.
6. Защита информации в системах телекоммуникации: Учебное пособие для вузов / [ Банкет В. Л. и др.]. – Од., УГАС им. А.С. Попова, 1997. – 96 с.
7. Иванов М.А. Криптографические методы защиты в компьютерных системах и сетях / Иванов М.А. – М. : КУДИЦ-ОБРАЗ, 2001. – 368 с.
8. Фомичев В.М. Дискретная математика и криптология / Фомичев В.М. – М. : ДИАЛОГ-МИФИ, 2003. – 400с.
9. Гулак Г. Різні підходи до визначення випадкових послідовностей: / Г. Гулак. Л. Ковальчук // Науково-технічний збірник «Правове, нормативне та метрологічне забезпечення системи захисту інформації в Україні». – 2001. – №3 – С. 127-133.
10. Кнут Д. Искусство программирования для ЭВМ. Том 2. Получисленные алгоритмы / Кнут Д. – М. : МИР, 1977. – 235 с.
11. Б. Шнайер Прикладная криптография. Протоколы, алгоритмы, исходные тексты на языке Си / Б. Шнайер – М. : Изд-во ТРИУМФ, 2002. – 816 с.
12. Венбо Мао. Современная криптография. Теория и практика / Венбо Мао ; [пер. с англ.]. – М. : Изд. дом «Вильямс», 2005. – 786 с.
13. Винокуров А. Сравнение российского стандарта шифрования, алгоритма ГОСТ 28147-89 и алгоритма Rijndael, выбранного в качестве нового стандарта шифрования США, «Системы безопасности» / А.Винокуров. Применко Э. – М. : Изд. «Гротэк», 2001, №№1,2.

14. А.Винокуров. Алгоритм шифрования ГОСТ 28147-89, его использование и реализация для компьютеров платформы Intel x86. Работа на правах рукописи, доступна на веб-сайте http://www.enlight.ru/crypto.

15. Основные принципы проектирования, оценка стойкости и перспективы использования в Украине алгоритма шифрования AES / [Гулак Г., Горбенко И., Олейников Р. и др.] // Науково-технічний збірник «Правове, нормативне та метрологічне забезпечення системи захисту інформації в Україні» – Київ. – 2003. – №7 – С. 14-25.

16. Коблиц Н. Курс теории чисел и криптографии / Коблиц Н. – М. : Научное издательство ТВП, 2001. – 106с.

17. Черемушкин А. В. Лекции по арифметическим алгоритмам в криптографии / Черемушкин А. В.– М. : МЦНМО, 2002.

18. Мухачев В. А. Методы практической криптографии / В. А. Мухачев, В. А. Хорошко. – К. : ООО «ПолиграфКонсалтинг», 2005. –215с.

19. Бессалов А. В. Криптосистемы на эллиптических кривых / А. В. Бессалов, А. Б. Телиженко. – К. : ІВЦ Видавництво «Політехніка», 2004. – 224 с.

20. Введение в криптографию/ [Ященко В. В. и др.]; под общ. ред. В. В. Ященко. – М.: МЦНМО, «ЧеРо», 1998. – 272 с.

21. ГОСТ Р ИСО 7498-2-99 Информационная технология. Взаимосвязь открытых систем. Базовая эталонная модель. Часть 2. Архитектура защиты информации.

22. ГОСТ Р ИСО/МЭК 9594-8-98 Информационная технология. Взаимосвязь открытых систем. Справочник. Часть 8. Основы аутентификации.

23. ГОСТ Р ИСО/МЭК 9594-9-95 Информационная технология. Взаимосвязь открытых систем. Справочник. Часть 9. Дублирование.

24. Горбатов В.С. Основы технологии PKI / В. С. Горбатов, О. Ю. Полянская. – М. : Горячая линия – Телеком. 2004. С.171-224.

25. Бернет С. Криптография. Официальное руководство RSA Security / С. Бернет, С. Пейн. – М. : Бином-Пресс, 2002. С.279-292.

26. Щербаков А. Прикладная криптография. Использование и синтез криптографических интерфейсов / А. Щербаков, А. Домашев. – М.: «Русская редакция», 2003. С.28-136.

27. Вербицький О. Вступ до криптології / Вербицький О. – Львів : Видавництво науково-технічної літератури, 1998. – 247 с.

28. Горбенко І.Д. Захист інформації в інформаційно-телекомунікаційних системах: Навч. Посібник. Ч.1. Криптографічний захист інформації / І. Д. Горбенко. Т. О. Гріненко. – Харків: ХНУРЕ, 2004. – 368 с.

29. Аналіз властивостей алгоритмів блокового симетричного шифрування (за результатами міжнародного проекту NESSIE): Радіотехніка. Всеукраїнський міжвідомчий науково-технічний збірник, вип. 141 / [Горбенко І.Д., Гулак Г.М., Олійников Р.В. та інш.]. – Харків: ХНУРЕ, 2005. – С. 7-24.

30. ГОСТ 28147-89. Системы обработки информации. Защита криптографическая. Алгоритм криптографического преобразования

31. ГОСТ 34.310-95 Криптографическая защита информации. Процедура выработки и проверки цифровой подписи на базе асиметричного криптографического алгоритма

32. ГОСТ 34.311-95. Информационная технология. Криптографическая защита информации. Функция хэширования.

33. Блочный симметричный алгоритм SHACAL-2 / Г. Гулак. И. Горбенко. М. Михайленко. Ю. Гитис // Науково-технічний збірник «Правове, нормативне та метрологічне забезпечення системи захисту інформації в Україні». – 2003. – №7 – С. 86-100.

34. Гулак Г. Н. Национальная технология ЕЦП: «время собирать камни» / Г. Н. Гулак, Л. В. Скрыпник // Збірка наукових праць Інституту проблем моделювання в енергетиці ім. Г.Є. Пухова, спеціальний випуск «Моделювання та інформаційні технології» – 2005 – С. 23-28.

35. Конхейм А. Г. Основы криптографии / Конхейм А. Г. – М. : Радио и связь, 1987. – 412 с.

36. Математичні основи криптографії / [Кузнецов Г. В., Фомічов В. В., Сушко С. О., Фомічова Л. Я.]. – Дніпропетровськ : НГУ, 2004. – 389 с.

37. Положення про порядок розроблення, виготовлення та експлуатації засобів криптографічного захисту конфіденційної інформації. Затверджено наказом ДСТСЗІ СБ України від 30.11.99 №53, зареєстровано в Міністерстві юстиції України 15.12.99 за № 868/4161, із змінами, згідно з наказом ДСТСЗІ СБ України від 30.04.04 № 31.

38. Шеннон К.Э. Теория связи в секретных системах / К.Э. Шеннон // Работы по теории связи и кибернетике. – М.: ИЛ, 1963. – С. 333-402.

39. NIST Special Publications 800-22. A Statistical Test Suite for the Validation of Random Number Generators and Pseudo Random Number Generators for Cryptographic Applications, 2000.

40. RIJNDAEL description. Submission to NIST by Joan Daemen, Vincent Rijmen. http://www.csrc.nist.gov/encryption/aes/round1/docs.htm.

41. Mecanismes cryptographiques Version 1.10. Crypto DCSSI @sgdn.pm.gouv.fr, 2006.

42. Feldman P. A practical scheme for non-interactive verifiable secret sharing // Proc. 28th Annu. Symp. on Found. of Comput. Sci. 1987. P. 427-437.

43. ФоменковГ.В. http://www.citforum.ru/cryptography/fortezza/.shtm.