

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

КРИПТОАНАЛІЗ. КРИПТОГРАФІЧНІ ПРОТОКОЛИ

Навчальний посібник
для студентів спеціальності
123 «комп'ютерна інженерія»

Ужгород – 2021

УДК 004.056.55, 003.26

Посібник з курсу «Компютерна криптографія» призначено для студентів інженерно-технічного факультету ДВНЗ «УжНУ» спеціальності 123-«комп'ютерна інженерія».

Укладачі: Гапак О. М. – канд. пед. наук, доцент кафедри комп'ютерних систем та мереж ДВНЗ «УжНУ»

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

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

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

ЗМІСТ

ВСТУП	5
1 КРИПТОАНАЛІЗ: СУЧАСНИЙ СТАН І ПЕРСПЕКТИВИ РОЗВИТКУ	6
1.1 Основні поняття криптоаналізу.....	6
1.2 Універсальні методи криптоаналізу.....	10
2 ОСНОВИ КРИПТОАНАЛІЗУ	15
2.1 Первинний аналіз шифровки.....	15
2.2 Конкретні приклади криптоаналізу.....	16
3 КРИПТОАНАЛІЗ ШИФРІВ НА МОНОАЛФАВІТНИХ ПІДСТАНОВКАХ	19
3.1 Криптоаналіз шифрів за методом простої заміни.....	19
3.2 Криптоаналіз шифру Цезаря.....	22
4 КРИПТОАНАЛІЗ ШИФРУ ВІЖІНЕРА	24
5 КРИПТОАНАЛІЗ ПОТОКОВИХ ШИФРІВ	29
5.1 Дешифрування методом гамування.....	30
5.2 Методи криптоаналізу поточкових шифрів.....	30
6 ЛІНІЙНИЙ ТА ДИФЕРЕНЦІАЛЬНИЙ КРИПТОАНАЛІЗ	33
6.1 Розкриття шифрів блокових алгоритмів.....	33
6.2 Лінійний криптоаналіз.....	33
6.3 Диференціальний криптоаналіз.....	38
6.4 Напрямки розвитку лінійного та диференціального крипто аналізу...41	
7 КРИПТОАНАЛІЗ СИСТЕМ З ВІДКРИТИМ КЛЮЧЕМ	42
7.1 Атаки на криптосистему RSA.....	43
7.2 Атаки на ЦЕП RSA.....	44
8 КРИПТОАНАЛІЗ ЗА ПОБІЧНИМИ КАНАЛАМИ	46
9 ВИКОРИСТАННЯ НОВИХ ТЕХНОЛОГІЙ У КРИПТОАНАЛІЗІ	50
9.1 Нейронні мережі.....	50

9.2	Генетичні алгоритми.....	51
9.3	Квантові комп'ютери.....	53
10	КРИПТОГРАФІЧНІ ПРОТОКОЛИ.....	56
10.1	Специфіка взаємодії абонентів.....	56
10.2	Приклади криптографічних протоколів.....	57
10.3	Інтерактивні системи доведення.....	58
10.4	Протоколи аутентифікація.....	59
10.5	Захищені обчислення.....	64
10.6	Спеціальні види електронного підпису.....	65
11	КРИПТОГРАФІЧНІ СХЕМИ ПОДІЛУ СЕКРЕТУ.....	68
11.1	Поділ секрету.....	68
11.2	Порогова схема поділу секрету.....	69
11.3	Схема обчислення ключа доступу.....	70
11.4	Метод «розшарування» зображень.....	72
11.5	Протоколи поділу секрету, що перевіряється.....	72
11.6	Протоколи конфіденційних обчислень.	73
	ЛІТЕРАТУРА.....	75
	ДОДАТКИ	

ВСТУП

Мета курсу – ознайомлення студентів із основами криптоаналізу та криптографічними протоколами.

Завдання дисципліни – сформувані погляд на захист інформації і криптографію як на систематичну науково-практичну діяльність, що носить прикладний характер. Сформувані базисні теоретичні поняття та практичні навички щодо проведення криптоаналізу класичних шифрів, блочних шифрів, асиметричних криптосистем та організацію криптографічних протоколів.

Вивчення даної дисципліни базується на знанні студентами курсу «Програмування», «Теорія інформації і кодування», «Теорія ймовірності і математична статистика», «Захист інформації в комп'ютерних системах», «Дискретна математика».

У результаті вивчення дисципліни студент повинен знати: основні види симетричних та асиметричних криптоалгоритмів, основні методи криптоаналізу та види протоколів. Вміти: застосовувати математичні методи описання і дослідження криптосистем; аналізувати криптосистеми, оцінювати їх стійкість, вміло застосовувати основні методи криптоаналізу.

Для засвоєння матеріалу курсу необхідна активна самостійна робота студентів. Криптоаналіз шифрів DES, ГОСТ 28147-89, IDEA та інші потребують великих ресурсів і для початківців є дуже складними. У той же час на прикладах класичних шифрів можна проілюструвати деякі важливі прийоми і методи криптоаналізу. Як показує практика, студенти після аналізу шифрів перестановки, простої заміни та Віженера впевнено і досить швидко входять до кола ідей сучасної криптографії. Таким чином, цей посібник виконує пропедевтическую функцію. Після аналізу класичних шифрів студенти можуть успішно вивчати сучасні блокові алгоритми шифрування і їм стають доступними ідеї лінійного і диференціального криптоаналізу.

Проблема побудови криптографічних протоколів нерозривно пов'язана з дослідженнями рівня їх безпеки. Разом з пошуком слабких місць і доопрацюванням протоколів відбувалася формалізація самого поняття безпеки, що забезпечується даним протоколом. Для багатьох протоколів, що спочатку здавалися надійними, були знайдені вразливості та розроблені атаки, що їх використовують.

Криптографічні протоколи є основним об'єктом досліджень в теоретичній криптографії. Одна з основних областей застосувань криптографічних протоколів – банківські платіжні системи, де замість платіжних доручень на папері використовується їх електронна форма. Але платіжні доручення – лише один з численних типів документів, що знаходяться в обороті в сфері бізнесу. Адже існують ще документи, з якими працюють державні органи і громадські організації, юридичні документи і т. д. Весь документообіг подається в електронному форматі, тому виникає необхідність захисту цієї інформації під час її збереження та передачі.

1 КРИПТОАНАЛІЗ: СУЧАСНИЙ СТАН І ПЕРСПЕКТИВИ РОЗВИТКУ

1.1. Основні поняття криптоаналізу

Основоположник сучасної криптографії Клод Шеннон довів, що якщо на будь-який вихідний текст накласти ключ довжиною не менше, ніж довжина тексту, то такий шифр стійкий до розкриття. Однак використання такого способу шифрування («одноразовий блокнот») у більшості випадків є дуже довгим і дорогим.

Вибір необхідного ступеня захисту інформації і засобів його забезпечення є важливою задачею і повинен враховувати ряд параметрів: рівень секретності інформації; її ціну; час, на протязі якого вона повинна зберігатися в таємниці і т.д.

За деякими проведеними оцінками, економічні втрати від зловмисних атак на банківські системи по всьому світу становить щорічно більше 130 млрд. доларів.

Як відомо, не всі присутні на ринку криптографічні засоби забезпечують відповідний рівень захисту. Тому, що задача визначення ефективності системи захисту при використанні криптографічних методів захисту найчастіше більш складна, а ніж розробка самої системи захисту. Часто аналіз нового шифру є науковою або інженерною задачею.

Сучасна криптологія – змагання методів шифрування і криптоаналізу.

Криптоаналіз – це наука про розкриття вихідного тексту зашифрованого повідомлення без доступу до ключа. Успішний аналіз може розкрити вихідний текст або ключ. Він дозволяє також знайти слабкі місця в криптосистемі, що, у кінцевому рахунку, веде до тих же результатів.

Фундаментальне правило криптоаналізу, уперше сформульоване голландцем А. Керкхоффом ще в XIX столітті полягає в тому, що стійкість криптосистеми повинна визначатися тільки таємністю ключа. Іншими словами, правило Керкхоффа полягає в тому, що весь алгоритм шифрування, крім значення секретного ключа, відомий для супротивника. Це обумовлено тим, що криптосистема, що реалізує сімейство криптографічних перетворень, звичайно розглядається як відкрита система. Такий підхід відбиває дуже важливий принцип технології захисту інформації: захищеність системи не повинна залежати від таємності чого-небудь такого, що неможливо швидко змінити у випадку витoku секретної інформації. Звичайно криптосистема являє собою сукупність апаратних і програмних засобів, яку можна змінити тільки при значних витратах часу і засобів, тоді як ключ є легко змінюваним об'єктом. Саме тому стійкість криптосистеми визначається тільки таємністю ключа.

Інше майже загальноприйняте допущення в криптоаналізі полягає в тому, що криптоаналітик має у своєму розпорядженні шифротексти повідомлень.

Існує чотири основних типи криптографічних атак. Звичайно, усі вони формулюються в припущенні, що криптоаналітику відомо застосовуваний алгоритм шифрування і шифротексти повідомлень. Перерахуємо ці криптоаналітичні атаки.

1) Криптоаналітична атака при наявності тільки відомого шифртексту. Криптоаналітик має тільки шифртексти C_1, C_2, \dots, C_m декількох повідомлень, причому усі вони зашифровані з використанням того самого алгоритму шифрування E_K . Робота криптоаналітика полягає в тому, щоб розкрити вихідні тексти M_1, M_2, \dots, M_m , або, ще краще, обчислити ключ K , використаний для шифрування цих повідомлень, для того, щоб розшифрувати й інші повідомлення, зашифровані цим ключем.

2) Криптоаналітична атака при наявності відомого відкритого тексту. Криптоаналітик має доступ не тільки до шифротекстів C_1, C_2, \dots, C_m декількох повідомлень, але також до відкритих текстів M_1, M_2, \dots, M_m цих повідомлень. Його робота полягає в пошуку ключа, використовуваного при шифруванні цих повідомлень, або алгоритму розшифрування D_K будь-яких нових повідомлень, зашифрованих тим же самим ключем.

3) Криптоаналітична атака при можливості вибору відкритого тексту. Криптоаналітик не тільки має доступ до шифрів-текстів C_1, C_2, \dots, C_m і зв'язаним з ними відкритим текстом M_1, M_2, \dots, M_m декількох повідомлень, але і може за бажанням обирати відкриті тексти, що потім одержує в зашифрованому виді. Такий криптоаналіз виходить більш могутнім у порівнянні з криптоаналітиком за відомим відкритим текстом, тому що криптоаналітик може вибрати для шифрування такі блоки відкритого тексту, що дадуть більше інформації про ключ. Робота криптоаналітика складається в пошуку ключа, використаного для шифрування повідомлень, або алгоритму розшифрування D_K нових повідомлень, зашифрованих тим же ключем.

4) Криптоаналітична атака з адаптивним вибором відкритого тексту. Це особливий варіант атаки з вибором відкритого тексту. Криптоаналітик може не тільки обирати відкритий текст, що потім шифрується, але і змінювати свій вибір у залежності від результатів попереднього шифрування. При криптоаналізі з простим вибором відкритого тексту криптоаналітик звичайно може вибирати кілька великих блоків відкритого тексту для їхнього шифрування; при криптоаналізі з адаптивним вибором відкритого тексту він має можливість вибрати спочатку більш дрібний спробний блок відкритого тексту, потім вибрати наступний блок у залежності від результатів першого вибору, і т.д. Ця атака надає йому ще більше можливостей, чим попередні типи атак.

Крім перерахованих основних типів криптоаналітичних атак, можна відзначити, принаймні, ще два типи.

5) Криптоаналітична атака з використанням обраного шифротекста. Криптоаналітик може обирати для розшифрування різні шифротексти C_1, C_2, \dots, C_m і має доступ до розшифрованих відкритих текстів M_1, M_2, \dots, M_m . Наприклад, він одержав доступ до захищеного від несанкціонованого розкриття блоку, що виконує автоматичне розшифрування. Робота аналітика полягає в відшуканні ключа. Цей тип криптоаналізу становить особливий інтерес для розкриття алгоритмів з відкритим ключем.

6) Криптоаналітична атака методом повного перебору всіх можливих ключів. Ця атака припускає використання криптоаналітиком відомого шифротексту і здійснюється за допомогою повного перебору всіх можливих

ключів з перевіркою, чи є осмисленим відкритий текст, що виходить. Такий підхід вимагає залучення великих обчислювальних ресурсів і іноді називається силовою атакою. Існують і інші, менш розповсюджені, криптоаналітичні атаки

Основні принципи криптоаналізу

1. Принцип Керкхоффа. Тільки супротивник може судити про криптостійкість системи.

2. Принцип Керкхоффа-Шеннона. Супротивник знає використовувану систему з точністю до ключової інформації.

3. Принцип Жеверже. Поверхневі ускладнення системи можуть бути ілюзійними, так як породжують хибні оцінки її стійкості.

4. При оцінці криптостійкості необхідно враховувати можливі криптографічні помилки та інші порушення безпеки.

Криптограф Ларс Кнудсен пропонує наступну класифікацію успішних результатів криптоаналізу блочних шифрів в залежності від об'єму і якості секретної інформації, яку вдалося отримати [1]:

- повний злом – криптоаналітик обчислює ключ;
- глобальна дедукція – криптоаналітик розробляє функціональний еквівалент досліджуваного алгоритму, що дозволяє зашифровувати і розшифровувати інформацію без знання ключа;
- часткова дедукція – криптоаналітику вдається розшифрувати деякі повідомлення;
- інформаційна дедукція – криптоаналітик отримує деяку інформацію про відкритий текст чи ключ.

Як правило, криптоаналіз розпочинається із спроб злому (атаки) спрощеної версії алгоритму, після чого результати розповсюджуються на повну версію.

Класифікація атак

X – відкритий текст, Y – шифротекст, k_1, k_2 – ключі шифрування і дешифрування (є випадки, коли вони рівні), E_k – функція шифрування, D_k – функція дешифрування.

– **Атака на основі шифротексту.** Криптоаналітик має тільки шифртексти декількох повідомлень, причому усі вони зашифровані з використанням того самого алгоритму шифрування E_k . Робота криптоаналітика полягає в тому, щоб розкрити вихідні тексти, або, ще краще, обчислити ключ k_1 , використаний для шифрування цих повідомлень, для того, щоб розшифрувати й інші повідомлення, зашифровані цим ключем.

– **Атака на основі відкритого тексту.** Криптоаналітик має доступ не тільки до шифротекстів декількох повідомлень, але також до відкритих текстів цих повідомлень. Його робота полягає в пошуку ключа k_1 , використовуваного при шифруванні цих повідомлень, або алгоритму розшифрування D_k будь-яких нових повідомлень, зашифрованих тим же самим ключем.

– **Атака на основі підбраного відкритого тексту.** Криптоаналітик не тільки має доступ до шифротекстів і зв'язаним з ними відкритим текстам декількох повідомлень, але і може за бажанням обирати відкриті тексти, що потім одержує в зашифрованому виді. Такий криптоаналіз виходить більш могутнім у порівнянні з криптоаналітиком за відомим відкритим текстом, тому що

криптоаналітик може вибрати для шифрування такі блоки відкритого тексту, що дадуть більше інформації про ключ. Робота криптоаналітика складається в пошуку ключа, використаного для шифрування повідомлень k_1 , або алгоритму расшифрування D_k нових повідомлень, зашифрованих тим же ключем.

– **Криптоаналітична атака з адаптивним вибором відкритого тексту.** Це особливий варіант атаки з вибором відкритого тексту. Криптоаналітик може не тільки обирати відкритий текст, що потім шифрується, але і змінювати свій вибір у залежності від результатів попереднього шифрування. При криптоаналізі з простим вибором відкритого тексту криптоаналітик звичайно може вибирати кілька великих блоків відкритого тексту для їхнього шифрування; при криптоаналізі з адаптивним вибором відкритого тексту він має можливість вибрати спочатку більш дрібний спробний блок відкритого тексту, потім вибрати наступний блок у залежності від результатів першого вибору, і т.д. Ця атака надає йому ще більше можливостей, чим попередні типи атак.

Існують і інші, менш розповсюджені, криптоаналітичні атаки.

Вказані атаки називаються пасивними, вони вивчають повідомлення незалежно від самої системи.

Засоби дозволяють зловмиснику активно втручатися в процес передачі повідомлення за допомогою імітації та підміни. Атака імітації – зловмисник вкладає в канал зв'язку сфабриковане тим «шифроване» повідомлення. При цьому він розраховує на те, що отримувач сприйме це повідомлення як справжнє. Атака підміни – супротивник спостерігаючи за повідомленням, що передається по каналу, виймає його і передає замінене.

Властивість самого шифру протистояти активним атакам називають **імітостійкістю** шифру.

Властивість системи протистояти будь-яким атакам називається її **стійкістю**. Кількісно стійкість визначається складністю криптоалгоритму. Універсальний метод прямого перебору всієї множини ключів дозволяє зробити оцінку зверху для стійкості алгоритмів.

Розрізняють стійкість ключа (складність розкрити найкращим методом); стійкість безключового читання (складність нав'язування неправдивої інформації найкращим алгоритмом).

Аналогічно розглядають стійкість криптоалгоритму, протоколу, алгоритму генерування і розподілу ключів. У залежності від складності злому алгоритми забезпечують різні ступені захисту. **Існують безумовно стійкі (теоретично стійкі), доведено стійкі і припустимо стійкі.**

Теоретично стійкий криптоалгоритм – такий алгоритм, коли ніякий метод криптоаналізу не дозволяє не тільки визначити ключ і відкритий текст, але й навіть отримати деяку інформацію про них. Такі системи на практиці незручні (симетричні системи із одноразовим використанням ключа потребують більшої захищеної пам'яті для збереження ключів, системи квантової криптографії потребують волоконно-оптичних каналів зв'язку, які є дорогими). Крім того доведення їх стійкості виходить за межі області математики і фізики. Тому абсолютно стійкі шифри використовуються тільки в межах зв'язку з невеликим

об'ємом інформації, що передається. Зазвичай, ці мережі використовуються для передачі особливо важливої державної інформації.

Стійкість **доведено стійких криптоалгоритмів** визначається складністю розв'язання відомої математичної задачі. Приклад таких криптосистем: Діффі-Хеллмана – складність дискретного логарифмування, RSA – складність розкладу великого числа на множники та інші.

Припустимо стійкі криптоалгоритми засновані на складності розв'язання частинної математичної задачі. Приклади: FEAL, AES та інші. Ці шифри характеризує порівняно мала вивченість математичних задач, на яких базується їх стійкість. Однак, такі шифри мають велику гнучкість, що дозволяє при визначенні слабих місць не відмовлятися від алгоритму, а провести його доробку.

1.2. Універсальні методи криптоаналізу

Метод повного перебору ключа.

Часто криптоаналітики розкривають шифри на компютерах методом перебору ключів. У процесі криптоаналізу доводиться перебирати більше мільярдів ключів зі швидкістю тисяча ключів у секунду.

Якщо впорядкувати множину ключів $K=\{k_1, k_2, \dots, k_n\}$ то повний перебір потребує $|K|$ операцій, де перевірка одного ключа одна операція. Ймовірність правильного вибору ключа – $\frac{1}{|K|}$.

Алгоритми повного перебору допускають розпаралелення, що дозволяє значно прискорити процес знаходження ключа. Є різні підходи до реалізації розпаралелення: метод конвеєру, поділ задачі, «китайська лотерея», «криптографічні водорості».

Конвеєр. По-перше, побудова конвеєра. Нехай алгоритм співвідношення $E_k(x)=y$ представимо у вигляді детермінованого ланцюжка найпростіших дій (операцій):

$$O_1, O_2, \dots, O_N$$

Візьмемо N процесорів A_1, A_2, \dots, A_N , поставимо їх порядок і припустимо, що i -ий процесор виконує три однакові за часом операції:

- 1) прийом даних від $(i - 1)$ -го процесора;
- 2) виконання операції O_i ;
- 3) передача даних наступного $(i + 1)$ -го процесора.

Тоді конвеєр з N послідовно з'єднаних паралельно і синхронно працюючих процесорів працює зі швидкістю $\frac{V}{3}$, де V швидкість виконання однієї операції процесором.

Поділ задач. Другий напрямок розпаралелення полягає в тому, що множина K розбивається на неперетинаючі підмножини K_1, K_2, \dots, K_Q . Система з Q машин перебирає ключі так, що i -ий машина здійснює перебір ключів з безлічі K_i , $i=1, \dots, Q$. Система припиняє роботу, якщо одна з машин знайшла ключ. Найбільшою складністю у викладеному підході є організація поділу ключової множини. Однак, якщо організувати пошук ключа таким чином, що при кожному

черговому випробуванні кожен з N процесорів стартує з випадкової точки, то час випробування збільшиться, але схема значно спроститься. Середнє число кроків випробування N процесорами (машинами) з ключів множини K в цьому випадку становить $\frac{|K|}{N}$.

Реалізація такого паралелізму передбачає різні рішення. Саме очевидне рішення - створення комп'ютерного вірусу для розповсюдження програми - зломщика в глобальній мережі. Вірус повинен використовувати періоди простою комп'ютера (за даними досліджень, комп'ютер простоює 70-90% часу) для здійснення перебору по множині ключів. Рано чи пізно один із заражених комп'ютерів виявить шуканий ключ (необхідно передбачити механізм оповіщення зловмисника); з ростом продуктивності комп'ютерів і швидкості розповсюдження вірусів загроза успішного результату такої атаки зростає.

У літературі описані більш оригінальні ідеї розпаралелення обчислень: «Китайська лотерея», створення «криптоаналітичних» водоростей і тварин.

Китайська лотерея припускає, що кожен радіоприймач і телевізор вбудована мікросхема, запрограмована на автоматичну перевірку різних множин ключів після отримання ефіру пари відкритий текст/шифротекст.

Використання біотехнологій в перспективі зробить можливим здійснення більш ефективних атак. Розглянемо вигадане тварина під назвою «DESозавр». Воно складається з оптично прозорих біологічних клітин, які вміють тестувати можливі ключі. По якомусь широкомовному оптичному каналу в клітини передаються пари відкритий текст/шифротекст. Розв'язок переносяться до органів мовлення DESозавра спеціальними клітинами, які подорожують по кровоносній системі тваринного. В доісторичні часи середній динозавр складався приблизно з 10^{14} клітин (без мікробів). Якщо кожна клітина може виконувати мільйон шифрувань в секунду, злом 56-бітного ключа займе $7 * 10^{-4}$ с, а 64-бітового - не більш 0,2 сек. Інший біологічний підхід полягає у створенні методами генної інженерії криптоаналітичних водоростей, які вміють розкривати криптографічні алгоритми методом повного перебору. Водорості можуть покривати великі простори, що теоретично дозволить створити якусь подібну розподіленого комп'ютера з величезним числом процесорів.

Тепер розглянемо випадок, коли криптоаналітик здійснює атаку на основі тільки шифртексту. При здійсненні спроби визначення ключа шифру за криптограммою шляхом її розшифрування на різних ключах вимагається деяким чином аналізувати вихідні дані алгоритму і перевіряти їх «змістовність». В якості об'єкта шифрування може виступати графічний файл або програма, в цьому випадку задача визначення «змістовності» вихідних даних стає дуже важкою. Коли ж відомо, що відкритий текст являє собою текст на природній мові, проаналізувати результат і впізнати успішний результат дешифрування порівняно нескладно, тим більше що нерідко криптоаналітик володіє деякою апіорною інформацією про зміст повідомлення. Потрібно за невеликим відрізком тексту вирішити, що собою являє дешифрований текст: змістовне повідомлення або набір випадкових символів. Однак вручну виконати аналіз безлічі фрагментів дешифрованих текстів неможливо. Тому завдання виділення змістовного тексту

(тобто виявлення правильного дешифрованого тексту) вирішують з допомогою компютера. В цьому випадку використовують теоретичні положення, розроблені в кінці XIX століття петербурзьким математиком Марковим А. А., - так звані ланцюги Маркова.

Зловмисник зацікавлений в отриманні певної ймовірнісної інформації про вихідний текст повідомлення. Наприклад, відомий факт написання тексту англійською мовою надає криптоаналітику певну апріорну інформацію про це повідомленні навіть до аналізу шифровки. У цьому випадку він заздалегідь знає, що слово «HELLO» є більш вірогідним початком повідомлення, ніж набір букв «FGHKM». Тому однією з цілей криптоаналізу може бути збільшення інформації, що відноситься до кожного можливого повідомленням. Припустимо, противник перехопив шифровку «ABCCD» і знає (або не передбачає), що використаний шифр – це шифр простої заміни. Аналіз шифрування дозволяє зробити висновок, що вихідне повідомлення складається з п'яти букв, причому на третій і четвертій позиціях стоїть одна і та ж буква, а інші відмінні від неї і різні між собою. Противник не може вважати, що це повідомлення «HELLO», тому що є й інші можливі повідомлення, наприклад, «TEDDY». Однак апостеріорні ймовірності таких відкритих текстів зростають щодо їх апріорних ймовірностей. У той же час апостеріорна ймовірність таких відкритих текстів, як «PEACE» або «GATES», знижується до нуля незалежно від їх апріорних ймовірностей. За Шенноном, криптосистема є досконалою, якщо після аналізу закритих текстів апостеріорні ймовірності можливих відкритих текстів залишаються такими ж, якими були їх апріорні ймовірності.

Можна припустити, що із зростанням потужності комп'ютерів розрядність ключа, достатня для забезпечення безпеки інформації проти атаки методом повного перебору, буде необмежено зростати. Однак це не так. Існують фундаментальні обмеження обчислювальної потужності, накладені структурою всесвіту: наприклад, швидкість передачі будь-якого сигналу не може перевищувати швидкість поширення світла у вакуумі, а кількість атомів у Всесвіті (з яких, в кінцевому рахунку, складаються комп'ютери) величезна, але скінчена.

Атака на ключі.

Однією із вразливостей криптосистем є використання слабких ключів, які не забезпечують достатнього рівня захисту. Відповідно до вже згаданого принципу Кіркхгофа, стійкість криптоалгоритма визначається секретністю ключа. Слабкий ключ – це ключ, не забезпечує достатнього рівня захисту, чи використовує в шифруванні закономірності, які можуть бути зламані. Зазвичай вважається, що алгоритм симетричного шифрування повинен по можливості не мати слабких ключів. Якщо це неможливо, то кількість слабких ключів має бути мінімальною. Тим не менш, всі слабкі ключі, якщо їх не можливо уникнути, мають бути відомими (як у алгоритмі DES).

Генератор випадкових чисел – ще одне слабе місце криптосистем. Це означає, що, якщо для генерування ключів використовується криптографічний слабкий алгоритм, незалежно від використовуваного шифру вся система буде нестійкою. Якісний ключ, призначений для використання в рамках симетричної криптосистеми, являє собою випадковий двійковий набір. Якщо потрібно ключ

розрядністю n , в процесі його генерування з однаковою ймовірністю повинен виходити з будь-яких 2^n можливих варіантів. Генерування ключів для асиметричних криптосистем – процедура складна, так як ключі, вживані в таких системах, повинні володіти певними математичними властивостями. Наприклад, у випадку системи RSA модуль шифрування являє собою добуток двох великих простих чисел.

За допомогою генераторів псевдовипадкових можна будувати стійкі криптосистеми. Хороші генератори випадкових чисел складні в розробці, так як їх надійність часто залежить від особливостей апаратного та програмного забезпечення. У зв'язку з цим ведеться пошук способів побудови ефективних генераторів псевдовипадкових на основі різних криптографічних припущень. Для генерування ключової інформації, призначеної для використання в рамках симетричної криптосистеми, використовуються такі методи (у порядку зростання якості):

- програмне генерування, яке передбачає обчислення чергового псевдовипадкового числа як функції поточного часу, послідовності символів, введених користувачем, особливостей його клавіатурного почерку і т. д.;
- програмне генерування, засноване на моделюванні якісного псевдовипадкового генератора з рівномірним законом розподілу;
- апаратне генерування з використанням якісного псевдовипадкового генератора;
- апаратне генерування з використанням генераторів випадкових послідовностей, побудованих на основі фізичних генераторів шуму і якісних генераторів псевдовипадкових чисел.

Показником ефективності є кількість операцій, що витрачаються на обчислення кожного чергового біта псевдовипадковою послідовністю. Псевдовипадковий генератор характеризуються періодом, розкидом, а також необхідністю його ініціалізувати. Малий період і поганий розкид ставляться до недоліків псевдовипадкового генератора. У випадку малого періоду (коли випадкових значень, що виробляються генератором, менше, ніж можливих значень ключа) зломисник може скоротити час пошуку ключа, перебираючи не самі ключі, а псевдовипадкові числа, генеруючи з них ключі. Невисока якість програмних методів формування пояснюється також необхідністю захисту від руйнуючих програмних дій.

Кращий спосіб генерування множини випадкових бітів – витяг їх з природно випадкових подій реального світу. Часто такий метод вимагає наявності спеціальної апаратури, але можна реалізувати його і на комп'ютерах. Як випадкові величини можна також розглядати інтервали між натисканнями клавіш клавіатури. Головний недолік таких систем – можливі закономірності в згенерованій послідовності. Використовуються фізичні процеси можуть бути випадкові, однак використання вимірювальних інструментів може призвести до появи проблем: зміщення, відхилення або кореляції між бітами. Обійти ці недоліки можна, використовуючи не один, а кілька випадкових джерел. В якості випадкових подій Брюс Шнайер пропонує розглядати, наприклад:

- моменти натискання на клавіші;

- моменти надходження команд від миші;
- номер сектора, час дня і затримку пошуку для кожної дискової операції;
- фактичної положення курсору миші;
- вміст поточного виведеного на екран зображення;
- момент закінчення завантаження процесора;
- час надходження мережових пакетів і т. д.

Застосовуючи до цих подій односпрямовану функцію, можна зберігати отримані випадкові величини в накопичувачі і при необхідності їх вилучати.

Частотний аналіз.

Визначення окремих символів і їх сполучень. Цей аналіз використовує статистичні і лінгвістичні методи для отримання додаткової інформації про ключ.

Лінійний аналіз.

Заміна нелінійної функції криптографічного алгоритму деяким лінійним аналогом. Найчастіше застосовується для симетричних алгоритмів.

Різницевий аналіз.

Різницевий або диференціальний аналіз використовує аналіз пари відкритих текстів, що мають визначені відмінності, після їх шифрування. Використовується для блочних систем і загальним методом.

Методи безключового читання.

Метод читання в колонках реалізує спосіб відновлення тексту, зашифрованого нерівномірною гамою.

Метод Полларда. Метод колізій для хеш-функцій. Метод «зустрічі посередині» та інші.

2 ОСНОВИ КРИПТОАНАЛІЗУ

Опишемо деякі найбільш прості та ефективні методики розкриття зашифрованого тексту. Ці методики, звичайно, не дають гарантії розкриття будь-якого шифру, бо криптоаналіз це скоріше мистецтво, засноване на інтуїції і досвіді, ніж точна наука, заснована на законах і формулах. Однак ці методики слугують інструментом злому, свого роду набором відмичок, які можуть принести результат лише в руках досвідченого майстра. Деякі методики застосовуються для розкриття ручних шифрів, а деякі для розкриття машинних. В обох випадках злом зазвичай відбувається з використанням обчислювальної техніки, що дозволяє позбавитися від рутинного перебору ключів за ручним методом. При цьому вдалий лозунг «працювати повинна машина, а думати повинна людина». При традиційному криптоаналізі, результат роботи часто визначається кваліфікацією зломщика, а при машинному – швидкістю обчислювальної системи. Оптимальне співвідношення цих двох факторів є необхідною умовою успіху криптоаналізу[4].

Завжди процес криптоаналізу починається зі збору різноманітних відомостей про зашифрований документ:

- на якій мові написаний оригінал;
- які лінгвістичні особливості даної мови (вживання артиклів і займенників, узгодження відмінків, правила утворення суфіксів і префіксів, порядок слідування слів у реченні, ...);
- яка мінімальна змістовна одиниця інформації (біт, символ, слово,...);
- які імовірно слова і фрази можуть знаходитися в тексті і в якій послідовності;
- яка приблизно довжина вихідного тексту;
- які ймовірно методи шифрування могли бути застосовані;
- яка службова інформація може знаходитися в тексті (контрольна сума пакета, адресна інформація, дата передачі);
- на який апаратурі шифрувався текст.

Такі і подібні дані збираються агентурними або аналітичними шляхами, що часто значно полегшує поставлене перед зломщиком завдання.

2.1. Первинний аналіз шифровки.

Первинний аналіз шифровки – дія, яка проводиться завжди при зломі будь-якого шифру. Так як будь-яка перехоплена шифровка (папірус з ієрогліфами, клаптик паперу з колонками цифр, повідомлення на морзянці, закодований файл або протокол активності комп'ютерної мережі) в кінцевому підсумку набрана за допомогою певного алфавіту (ієрогліфічного, цифрового, літерного, байтового), то можна, по-перше, оцінити інформативність всього повідомлення, по-друге, проаналізувати діаграму розподілу ймовірностей появи окремих символів алфавіту в тексті [4].

Інформативність повідомлення в тому сенсі, який запропонував Шеннон, визначається за формулі $H = \left| \sum P_i \cdot \log_2(P_i) \right|$, де P_i - частота появи i -го символу в тексті. Простіше кажучи, чим вище інформативність, тим більше щільність корисної інформації в обсязі повідомлення. Крім того, інформативність визначає теоретично максимальний коефіцієнт стиснення повідомлення архіватором, якщо літери в повідомленні будуть зустрічатися з рівною ймовірністю, то коефіцієнт стиснення буде ще більшим. Таке трактування інформативності має три особливості:

1. Зовсім не важливий смисловий зміст аналізованого тексту.
2. Вважається, що символи в початковому тексті не, залежать від попередніх символів.
3. Заздалегідь відомий алфавіт вихідного тексту.

Другим пунктом після визначення інформативності шифровки є побудова **діаграми розподілу ймовірностей прояву окремих символів в тексті шифровки**. Переважно діаграму представляють у вигляді лінійного графіка, де по одній осі відкладені порядкові номери символів в алфавіті, а по іншій нормалізовані ймовірності прояву цих символів у тексті шифровки.

Ймовірності появи окремих символів англійської, української, російської мов стандартного літературного тексту є у вільному доступі. Ймовірності наведені в процентному поданні та округлені до цілого.

Твердження про те, що ймовірність прояву чергового символу в середньостатистичному тексті є величина незалежна неправильно, так як деяких поєднань символів в мові взагалі може не бути. Правильніше сказати, що ймовірність появи певного символу в черговій позиції тексту залежить від попереднього символу. Якщо припустити, що алфавіт повідомлення складається з символів української мови і пробілу, то неважко вивести таблицю ймовірностей появи конкретних біграм в довільному українському тексті. Аналогічно з українськими біграмами, можна скласти таблицю ймовірностей прояву англійських біграм в англійському тексті та інше.

Крім біграм, можна побудувати аналогічні таблиці для триграм, тетраграмми і т.д.

Отже, після всього сказаного, можна зробити висновок, що діаграма розподілу ймовірностей прояву символів і інформативність можуть дати важливі непрямі відомості про мову шифровки або хоча б про можливий тип шифрування.

2.2. Конкретні приклади криптоаналізу

Якщо за результатами первинного аналізу шифровки був успішно визначений метод шифрування, то можна вважати, що ми зробили близько 20% всієї роботи. Якщо методами шифрування виявилися різні види підстановок, перестановок шифрування біграм або шифрування по системі Віжінера, а довжина шифровки досить велика, то можна сказати, що дешифрування вихідного тексту діло техніки. Якщо найбільш імовірним методом шифрування виявилось

гамування, а ще гірше взбивка, то доведеться визнати, що робота по розкриттю шрифту тільки починається і радіти ще дуже рано.

Ми ніколи не доб'ємося розкриття шрифту, якщо в результаті первинного аналізу шифровки ми зробили неправильне припущення про метод шифрування.

2.2.1. Криптоаналіз шифрів за методом перестановок

Перед описом методу злому шифрування за перестановкою перерахуємо кілька властивостей даного методу шифрування, якими треба буде скористатися.

По-перше, якою б не була перестановка, у неї завжди є максимальний радіус. Іншими словами, символ може переміщатися з поточної позиції не більше ніж на відстань R вперед або назад.

По-друге, для розрахунку зсуву конкретного символу відносно його положення в початковому тексті зазвичай використовується проста математична формула, а не повна матриця перестановок, так як в протилежному випадку обсяг ключової інформації для шифрування буде близьким до об'єму вихідного тексту.

І по-третє, математична формула для перестановки символів повинна бути однозначною, тобто кожному символу в початковому тексті повинен відповідати тільки один символ тексту шифровки і не повинно бути накладок.

Розкриття шифру перестановки зазвичай здійснюють перебором можливих формул і аналізом одержуваного відкритого тексту. В якості аргументів у формулах фігурують значення поточної позиції i і максимального радіуса R . В якості операцій над аргументами у формулах можуть використовуватися додавання, віднімання, побітової операції, множення, зведення в ступінь, взяття залишку за модулем R , а також різні комбінації цих операцій.

Криптоаналіз шифру стовпцевої перестановки

Під час розв'язання даної задачі необхідно відновити початковий порядок слідування букв тексту. Для цього використовують аналіз сумісності символів (додаток Г) та таблиці частот біграм відповідної мови (додаток В) для першого рядка таблиці.

Приклад 2.1

Розшифрувати текст «ИСВИІПНКУАПЬОТІРІУВКИТПАЦ», зашифрований стовпцевою перестановкою, без ключа (ключем є розмір таблиці та порядок перестановки стовпців).

Текст містить 25 символів, тому, ймовірно, можемо розглядати таблицю розміром 5x5. Таким чином, запишемо текст по СТОВПЦЯХ у таблицю. Можемо припустити, що мова на якій написаний текст українська(буква Ї).

И	П	П	Р	И
С	Н	Ь	І	Т
В	К	О	У	П
И	У	Т	В	А
Ї	А	І	К	Ц

Розглянемо у першому рядку можливі біграми і згідно додатку В запишемо їх ймовірності: ПР– **0,0071** ПИ–0, 0011 РИ – **0,0057**

ИР – 0,0017 РП – 0,0001 ИП – 0,0007

Із даного аналізу бачимо, що найбільш можливими є сполучення: ПР, РИ. Тому перший рядок, ймовірно, буде – ПРИПИ. Важливо визначити прядок букв, які повторюються, наприклад:

И	П	П	Р	И
5	1	4	2	3

Переставимо стовпці згідно цього припущення:

П	Р	И	П	И
Н	І	Т	Ь	С
К	У	П	О	В
У	В	А	Т	И
А	К	Ц	І	Ї

У результаті цієї перестановки ми отримали розшифрований текст: ПРИПИНІТЬ СКУПОВУВАТИ АКЦІЇ.

Подвійна перестановка

Використовуємо для розшифрування попередній аналіз, із врахуванням того, що його можна робити над будь-яким рядком таблиці.

Приклад 2.2

Розшифрувати текст «ыоечттоу_снсорчтрнаидьн_е», зашифрований подвійною перестановкою, без ключа (ключем є розмір таблиці та порядок перестановки стовпців та рядків).

Текст містить 25 символів, тому, ймовірно, можемо розглядати таблицю розміром 5x5. Запишемо текст по рядках у таблицю. Також можемо припустити, що мова на якій написаний текст російська (буква ы)

ы	о	е	ч	т
т	о	у	_	с
н	с	о	р	ч
т	р	н	а	и
д	ь	н	_	е

Розглянемо у першому рядку можливі біграми і згідно додатку В запишемо їх частоти: ое – 15; оч – 12; ет – 33; те – 31; ео – 7; ты – 11; че – 23. Згідно проведеного аналізу, переставимо стовпці у порядку 24351.

о	ч	е	т	ы
о	_	у	с	т
с	р	о	ч	н
р	а	н	и	т
ь	_	н	е	д

Після цього переставимо рядки у порядку 32451. У результаті цієї перестановки ми отримали розшифрований текст: **срочно устранить недочеты.**

3 МОНОАЛФАВІТНІ ПІДСТАНОВКИ

3.1. Криптоаналіз шифрів за методом простої заміни.

Дешифрування проводиться в 7 етапів:

1) В якості одиниці інформації приймається 1 символ для простої підстановки або 2 символи для шифрування біграм.

2) Будується діаграма розподілу примітивних елементів для шифрованого тексту.

3) Побудована діаграма порівнюється з діаграмою розподілу примітивних елементів стандартного літературного тексту.

4) Вручну або автоматично робиться припущення про відповідність примітивних елементів вихідного тексту і тексту шифровки, причому аналіз починається з елементів, які найчастіше зустрічаються (найбільш високі піки на графіку), а закінчується елементами, які найрідше зустрічаються.

5) Після зворотної заміни примітивних елементів шифрованого тексту на відповідні їм примітивні елементи вихідного тексту, виходить досить читабельний текст вихідного повідомлення.

6) Якщо вихідне повідомлення містить помилки, або зовсім не схоже на зв'язний осмислений текст, то це значить, що ми зробили помилкове припущення про відповідності елементів і слід повернутися до пункту 4.

7) Аналіз закінчується тоді, коли отриманий текст не містить помилок та є осмисленим.

Підстановки на основі простої заміни не приховують статистичні властивості тексту. Тому криптоаналіз шифру простої заміни заснований на використанні статистичних закономірностей мови (Додаток Б). Додатково можна використовувати аналіз k-грам (в основному біграм), враховувати властивості мови (суфікси, префікси, подвоєння, сполучники та інше) (Додатки В, Г).

Приклад 3.1

Розшифрувати текст, зашифрований за допомогою простої заміни:

9216811398918112 8158136519195518 1681705566, 66 401281666518
581316551270819181 1681581340 19654070816681-13591819554219819181
9216819116595840 1365 66148158708119656459191998 13591819816481915584
6626168186192678136665 86401461-98702684 662614 14559864611981581355
9255149216267212581366 1365 8116916519558065785584 81866465581355
58136566 19591281506426662612 865980 662670811626581365191998
70811292'17135916198188 13591819557026. 806558138158406665191998 1365
16818011261659191998 9265167040 8186422658641766656461198188
13591819557026 40 66585518 583559166518 14559864611981581355 5813656426
195981861855141955581317 146498 1981161265646119819181
3540197078558119406665191998 9255149216267212581366,
8116916519558065785584, 40581365198166, 8065706465145566 815866551326,
81188116811926 801481168166'98 13818581.

1655807059 801681581365191998 705564617081581355
70811292'17135916198188 13591819557026 1365 669216816665145059191998
1981665513195518 92168191166512192618 921681144070135566
1959122619404259 92162680668114261361 1481 801681581365191998
928113165986 40 665514928166551419811240 706514168166811240
806586598092594259191917 5813164070134016, 987055 9216261486656426
70811292'1713591626, 658681 9281198166266426 5558194017422684
70811292'17135916192684 35811914. 5861819181141955, 70816426
70811292'17135916 1959 921681581381 9281125542192670
92166578556619267065, 65 55195813164012591913 40921665666455191998,
801965421981 801681586426 662612819126 1481 987081581355 168186811326
70811292'17135916198188 13591819557026. 80 168158138112 6626128191 1481
706665645535557065785588 708116265813406665425566
70811292'17135916198188 13591819557026 1365 9216819116651219819181
806586598092594259191998, 928158136572 92261365191998 921681
706665645535557065785517 921681911665125558135566 1365
551950591959165566, 987055 806586598092594240171361 987055581940
168186811340 70811292'17135916198188 13591819557026. 58651259 146498
1681801681867026 70811292'17135916192618 582658135912 1365 1259165950,
8818 584092168166811440 1365 8186586440918166406665191998
928170642670651955 66269240587019267026 58925978556564611981581355
„70811292'171359161955 58265813591226 1365 125916595055”.

8065 146519261226 918164816619819181 40921665666455191998
58136513265813267026 40 80657065169265135861705584 81866465581355 8559 40
2007 16817855 1965 86656465195855 171626142642192618 81585586 1365 8818
66551481701659126459192618 5813164070134016192618
9255141681801455645566 196564554240666564815861 9281196514 30 1326589842
81142619267861 8186422658641766656461198188 13591819557026 40585518
6626145566 1365 1281142635557065785584. 196514656455 705564617055581959
80865564611159191998 8186422658641766656461198188 13591819557026
925916595813656481 86401326 65701340656461192612, 196513811255581361
92815813656481 92261365191998 921681 593559701326661955581361 8888
662670811626581365191998.

Представимо таблицю частот символів: перший рядок список зашифрованих букв (кожній відповідає одне число), другий - кількість зашифрованих символів у тексті, третій - відношення кількості зашифрованих

си

	81		19		13		65		55		26		16		58		59		70		66		92		64		40	
	133		113		91		88		85		76		73		64		61		61		55		49		42		39	
	0.100		0.085		0.068		0.066		0.064		0.057		0.055		0.048		0.046		0.046		0.041		0.037		0.032		0.029	

Як бачимо, найбільш частіше в тексті зустрічається число 81, за допомогою таблиці «Середньостатистичні частоти букв та пропуску між словами в українській мові» це буде буква «о». Також зробимо припущення, що 98 це буква «я», тому що перед цією буквою є апостроф. Ще бачимо що 8888 дві однакові букви і робимо припущення що це буква «ї». 19 - буде буква «н», тому що вона досить часто зустрічається в тексті, вона повинна бути приголосна і за таблицею біграм сполучення з буквою «о» найімовірніша.

```
9216013я91012 0581365нн5518 160705566, 66 40120666518 5813165512700910 160581340 н6540700660-135918н5542но910 921609116595840
1365 661405870он656459ннн 135918но640915584 662616086н2678136665 86401461-я702684 662614 1455я6461но581355 92551492162672125
81366 1365 0169165н558065785584 0866465581355 58136566 н59120506426662612 865980 66267001626581365ннн 7001292'17135916ної 135
918н557026. 80655813058406665ннн 1365 1608011261659ннн 9265167040 086422658641766656461ної 135918н557026 40 66585518 58355916
6518 1455я6461но581355 5813656426 н59086185514н55581317 1464я н01612656461но910 3540н707855он406665ннн 9255149216267212581366
, 0169165н558065785584, 40581365но66, 8065706465145566 05866551326, 018016он26 8014016066'я 130850.
1655807059 80160581365ннн 70556461700581355 7001292'17135916ної 135918н557026 1365 66921606665145059ннн но665513н5518 9216091
166512н2618 92160144070135566 н591226н404259 9216268066014261361 140 80160581365ннн 92013165986 40 665514920665514но1240 7065
141606601240 806586598092594259нн17 5813164070134016, я7055 9216261486656426 7001292'1713591626, 65860 920но66266426 5558н401
7422684 7001292'17135916н2684 35он14. 5861091014н55, 7006426 7001292'17135916 н59 9216058130 920125542н2670 921665785566н2670
65, 65 55н581316401259н13 40921665666455ннн, 80н6542но 80160586426 66261209126 140 я700581355 1608601326 7001292'17135916ної
135918н557026. 80 1605813012 662612091 140 7066656455355570657855ї 70016265813406665425566 7001292'17135916ної 135918н557026
1365 9216091166512но910 806586598092594259ннн, 92058136572 92261365ннн 92160 706665645535557065785517 92160911665125558135566
1365 55н5059н59165566, я7055 806586598092594240171361 я705558н40 1608601340 7001292'17135916ної 135918н557026. 58651259 1464
я 16080160867026 7001292'17135916н2618 582658135912 1365 1259165950, ї18 5840921606601440 1365 08658644091066406665ннн 920706
4267065н55 662692405870н267026 5892597855656461но581355 „7001292'17135916н55 58265813591226 1365 125916595055".
8065 1465н261226 91064066но910 40921665666455ннн 58136513265813267026 40 80657065169265135861705584 0866465581355 8559 40 200
7 1607855 н65 86656465н5855 171626142642н2618 0585586 1365 ї18 6655140701659126459н2618 5813164070134016н2618 925514160801455
645566 н656455424066656405861 920н6514 30 132658я42 01426н267861 086422658641766656461ної 135918н557026 40585518 6626145566 1
365 120142635557065785584. н6514656455 70556461705558н59 80865564611159ннн 086422658641766656461ної 135918н557026 92591659581
365640 86401326 65701340656461н2612, н651301255581361 920581365640 92261365ннн 92160 59355970132666н55581361 її 6626700162658
1365ннн.
```

Рисунок 3.1 Частково розшифрований текст

66 повинна бути приголосна, тому що вона досить часто зустрічається з буквою «о». За таблицею біграм і частот букв в українській мові робимовисновок, що це буква «в», 18 – «х», тому що з буквою «ї» в слові з двох букв має велику імовірність, 13 = «т» - часто зустрічається буква і з тексту вона повина бути приголосна, 65 = «а», 16 = «д» і т.д.

92160та91012 о58танн5518 1607055в, в 4012ова18 58т16551270о91о 16058т40 на4070ово-т5918н5542но91о 921609116595840 та вдо5870о на6459ння т5918но640915584 в2616086н2678тва 8640д61-я702684 в26д д55я6461но58т55 9255д921626721258тв та о1691ан5580а785584 о8 664а58т55 58тав н5912о506426в2612 865980 в2670о162658танн 70о1292'17т5916ноі т5918н557026. 80а58то5840вання та 1608011261659 нн 92а167040 о864226586417ва6461ноі т5918н557026 40 в585518 58355916а18 д55я6461но58т55 58та6426 н590861855дн5558т17 д64я но 1612а6461но91о 3540н707855он40вання 9255д921626721258тв, о1691ан5580а785584, 4058танов, 80а7064ад55в о58в55т26, о18о16он26 80 до16ов'я то85о.

1655807059 8016058танн 7055646170о58т55 70о1292'17т5916ноі т5918н557026 та в9216овад5059ння нов55тн5518 921609116а12н2618 92 16од4070т55в н591226н404259 92162680вод26т61 до 8016058танн 92от165986 40 в55д92ов55дно1240 70ад16ово1240 80а86598092594259н н17 58т164070т4016, я7055 921626д86а6426 70о1292'17т591626, а86о 92онов266426 5558н4017422684 70о1292'17т5916н2684 35онд. 586 1о91одн55, 70о6426 70о1292'17т5916 н59 9216058то 92о125542н2670 9216а7855вн2670а, а 55н58т16401259нт 409216ав6455ння, 80на42н о 80160586426 в2612о9126 до я70о58т55 16086от26 70о1292'17т5916ноі т5918н557026. 80 16058то12 в2612о91 до 70ва6455355570а7855 і 70о162658т40ва4255в 70о1292'17т5916ноі т5918н557026 та 921609116а12но91о 80а86598092594259ння, 92о58та72 9226танн 9216о 70 ва6455355570а785517 921609116а125558т55в та 55н5059н591655в, я7055 80а8659809259424017т61 я705558н40 16086от40 70о1292'17т591 6ноі т5918н557026. 58а1259 д64я 16080160867026 70о1292'17т5916н2618 582658т5912 та 1259165950, і18 58409216овод40 та о8658644 091ов40вання 92о70642670ан55 в2692405870н267026 5892597855а6461но58т55 „70о1292'17т5916н55 582658т591226 та 125916595055”.

80а дан261226 91о64овно91о 409216ав6455ння 58тат2658т267026 40 80а70а1692ат5861705584 о8664а58т55 8559 40 2007 1607855 на 86а 64ан5855 171626д2642н2618 о585586 та і18 в55до701659126459н2618 58т164070т4016н2618 9255д16080д556455в на64554240ва6405861 92 онад 30 т2658я42 од26н267861 о864226586417ва6461ноі т5918н557026 40585518 в26д55в та 12од26355570а785584. нада6455 7055646170 5558н59 80865564611159ння о864226586417ва6461ноі т5918н557026 9259165958та64о 8640т26 а70т40а6461н2612, нато125558т61 92о58та 64о 9226танн 9216о 59355970т26вн5558т61 іі в2670о162658танн.

Рисунок 3.2 Наступний крок заміни

Далі використовуючи таблиці біграм і частоти букв в українській мові, а також використовуючи контексту тексту завдяки відгаданим слів ми повністю розшифруємо текст.

протягом останніх років, в умовах стрімкого росту науково-технічного прогресу та вдосконалення технологій виробництва будь-як ий вид діяльності підприємств та організацій області став неможливим без використання комп'ютерної техніки. застосування та р озширення парку обчислювальної техніки у всіх сферах діяльності стали необхідністю для нормального функціонування підприємств , організацій, установ, закладів освіти, охорони здоров'я тощо.

різке зростання кількості комп'ютерної техніки та впровадження новітніх програмних продуктів неминує призводить до зростання потреб у відповідному кадровому забезпеченню структур, які придбали комп'ютери, або поновили існуючий комп'ютерний фонд. сьо годні, коли комп'ютер не просто помічник працівника, а інструмент управління, значно зросли вимоги до якості роботи комп'ютер ної техніки. з ростом вимог до кваліфікації користувачів комп'ютерної техніки та програмного забезпечення, постає питання про кваліфікацію програмістів та інженерів, які забезпечують якісну роботу комп'ютерної техніки. саме для розробки комп'ютерних систем та мереж, їх супроводу та обслуговування покликані випускники спеціальності „комп'ютерні системи та мережі”.

за даними головного управління статистики у закарпатській області ще у 2007 році на балансі юридичних осіб та їх відокремлени х структурних підрозділів налічувалось понад 30 тисяч одиниць обчислювальної техніки усіх видів та модифікацій. надалі кількі сне збільшення обчислювальної техніки перестало бути актуальним, натомість постало питання про ефективність її використання.

Рисунок 3.3 Розшифрований текст

3.2. Криптоаналіз шифру Цезаря

Криптосистема Цезаря визначається рівнянням $y_i = (x_i + k) \bmod m$, де $i = \overline{1, n}$, y_i – порядок букви в алфавіті криптограми, x_i – порядок букви в алфавіті відкритого тексту, k – ключ шифру, який може змінюватися від 1 до m , n – довжина криптограми, m – потужність алфавіту (кількість букв).

Ключ можна знайти за допомогою декількох методів:

– Частотного аналізу, знайти букву, яка найчастіше зустрічається у шифрі і порівняти відстань від неї до букви, яка найчастіше зустрічається у алфавіті відповідної мови.

– Здійснити атаку повним перебором на множині всіх можливих ключів.

– Обчислити ключ за допомогою формули:

$$k^* = \arg \max_k l(k), \quad l(k) = \sum_{j=0}^{m-1} v_{(j+k) \bmod m} \cdot \log_2 p(j), \quad v_j \text{ — частота } j\text{-ої букви у}$$

криптограмі, $p(j)$ – ймовірність j -ої букви у алфавіті.

Приклад 3.2

Розшифрувати криптограму зашифровану за допомогою шифру Цезаря.

«pelcgbbybtlvfpelcgbtenculnaqpelcgbnanylfvf»

Перетворимо її у числову, нумерація букв починається із нуля:

«15 4 11 2 6 1 24 1 19 11 21 5 15 4 11 2 6 1 19 4 13 2 20 11 12 0 16 15 4 11 2 6 1 13 0 13 24 11 5 21 5».

Складемо таблицю ймовірнісних характеристик англійського алфавіту

буква	a	b	c	d	e	f	g	h	i	j	k	l	m	n
j	0	1	2	3	4	5	6	7	8	9	10	11	12	13
$\text{Log}_2 p(j)$	-3.6	-6	-5	-4.7	-3	-5.4	-5.7	-4.2	-3.8	-9.3	-7.2	-4.6	-5.3	-3.8
буква	o	p	q	r	s	t	u	v	w	x	y	z		
j	14	15	16	17	18	19	20	21	22	23	24	25		
$\text{Log}_2 p(j)$	-3.7	-5.6	-9.8	-4	-3.9	-3.4	-5.2	-6.7	-5.7	-9	-5.9	-10.1		

Знайдемо частотні характеристики криптограми

буква	a	b	c	d	e	f	g	h	i	j	k	l	m	n
j	0	1	2	3	4	5	6	7	8	9	10	11	12	13
v_j	2	4	4	0	4	3	3	0	0	0	0	6	0	4
буква	o	p	q	r	s	t	u	v	w	x	y	z		
j	14	15	16	17	18	19	20	21	22	23	24	25		
v_j	0	3	1	0	0	2	1	2	0	0	2	0		

Обчисливши та побудувавши графік логарифмічної функції правдоподібності $l(k)$, знайдемо ключ рівний 13.

k	0	1	2	3	4	5	6	7	8
L(k)	-203,5	-230,8	-193,4	-219,8	-234,2	-258,6	-207,3	-211,2	-235,2

k		12	13	14	15
L(k)		-260,2	-185,4	-253,5	-211,8

Зсунемо всі букви криптограми на 13 позицій і отримаємо вихідний текст:

cryptologyis cruptography and cryptoanalysis

4 КРИПТОАНАЛІЗ ШИФРУ ВІЖІНЕРА

Поліалфавітні підстановки маскують справжню частоту появи символів у шифрі, тому вони значно надійніші як моно алфавітні. Однак, метод частотного аналізу можна застосувати і для них. Шифр Віжінера – поліалфавітна підстановка із використанням одного алфавіту (кожна буква визначає свій алфавіт шифрування). Також цю криптосистему можна розглядати як шифр гамування із використанням періодичної гами малого періоду.

Шифрування можна записати: $E_i = (M_i + K_{i(\text{mod } u)}) \text{mod } L$, де C_i , M_i – числові еквіваленти символів криптограми та початкового тексту, K_i – числовий еквівалент букви ключа, L – потужність алфавіту, u – довжина ключа.

Розшифрування запишемо у вигляді: $M_i = (C_i - K_{i(\text{mod } u)}) \text{mod } L$. Аналогічно, знаходження ключа можна записати як: $K_i = (C_i - M_i) \text{mod } L$.

Криптоаналіз даного шифру можна розбити на такі основні кроки:

1. Знаходимо довжину ключа, якщо вона невідома за допомогою одного із методів (Казискі, Фрідмана).

2. Розбиваємо криптограму на блоки так, щоб число символів у кожному блоці дорівнювало довжині ключа.

3. Символи криптограми, які займають однакове положення у блоці (стовпці) мають однакове зміщення (ключ). Аналіз відносно кожного такого стовпця криптограми дозволить визначити зміщення у всіх блоках, а саме, ключ шифрування. Такий метод має назву читання у колонках.

Визначення довжини ключа:

- Метод Казискі

Тест заснований на простому спостереженні про те, що два однакові відрізки відкритого тексту, які стоять один від одного на відстані кратній u будуть однаково зашифровані. Внаслідок цього в шифрі знаходяться повторення (не менше трьох символів). Знаходять відстані між ними d_1, d_2, \dots, d_k , потім знаходять їх найбільший спільний дільник – d і вважають, що довжина ключа $u = \kappa d$, де κ – деякий коефіцієнт. Якщо співпадінь багато, то $u = d$.

- Перший метод Фрідмана

Для криптограми обчислюється практичний індекс відповідності:

$$IB_{pr} = \frac{\sum_{i=1}^L (f_i - 1) f_i}{N(N-1)}, \text{ де } N - \text{кількість символів у криптограмі, } f_i - \text{кількість}$$

відповідної букви у криптограмі, L – потужність алфавіту.

Цей індекс порівнюють із теоретичним індексом відповідності, результати очікувань якого для різних довжин ключа містяться у таблиці (наприклад, для російського алфавіту+пропуск). Аналогічні таблиці є для різних алфавітів.

Довжина ключа	Min IB _{теор}	Max IB _{теор}
1	0,0544	0,0550
2	0,0395	0,044
3	0,0355	0,0405
4	0,0350	0,0390
5	0,0335	0,0385
6	0,0325	0,0365
7	0,0315	0,0350
8 і т.д.		

Цей метод ефективний при довжині ключа менше п'яти.

- **Другий метод Фрідмана**- випробування можливих періодів по наступній схемі: із вихідної криптограми y_1, y_2, \dots, y_n виписується d (вибирається будь-як). Тоді для кожної послідовності обчислюється індекс відповідності:

$y_1, y_{1+d}, y_{1+2d}, \dots$ \ стовпці

$y_2, y_{2+d}, y_{2+2d}, \dots$

.....

$y_n, y_{n+d}, y_{n+2d}, \dots$

Якщо всі індекси відповідності в середньому приблизно рівні $\frac{1}{d} \sum_{i=1}^L p_i^2$, то величину d сприймають за довжину ключа, інакше випробовують інший d . Другий метод Фрідмана ефективний для $d \leq 30$.

Приклад 4.1

Знайти довжину ключа, сам ключ та розшифрувати текст, написаний на російській мові.

влцдугтжбюцхъяррмшбрхцзооэцгбрыцмйфктъюьмшэсяцпунуящэйтаьэдкци
брыцгбрпачкъуцпъбъсэгкцъгуушарцёвърюуююэкаабрняфукабъарпяъафкъиьжяф
фнйояфывбнэнфуюгбръсшьжэтбзёчюьюръегофкбъчябашвёуъюаднчжчужцёэвлр
нчулбюпцуруньшсэюъзкцхъяррнрювяспэмасчкпэужъжыатуфуярюравртубурьпэ
щлафоуфбюацмнубсюкйтаьэджюнооэгюожбгкбрънцэпотчмёодзцвбцщщвщепчдчд
ръюьскасэгъппэгюкдойрсервоопчщоказръббнэугнялёкьсрбёуыэбдэулбюасшоуэт
ъшкрсдугэфлбубуьчнчтртпэгюкиугюэмэгюккъьпэгяапуфуэърадзьжчюрмфцхраю
юанчёчюьыхъьцомэфъцпоирькнщпэтэузуябашуцбаыэйчдфрпэцърьцьцпоилуфэд
цойэдятррачкубуфнйтаьэдкцкрннцюабугюуубурьпйюэжтгюркующюуфъэгясуои
чщщчдцсфырэдщъуяфшёчцюрщвяхвмкршрпгюопэуцчйтаьэдкцибрыцяжтюрбу
этэбдующэубъибрювъежагибргабрымпуноцшяжцечкфодщюъжшйуъцхчщвуэдл
дъэгясуахзцэбдэулькнъщбжяцъърёдъвъювлрнуяфуоухфекьгцччгэъжтанопчынаж
пачкъуьмэнкйрэфщэъбуд

1. Визначаємо довжину ключа за методом Казискі.

У цьому тексті знайдено чотирикратне повторення буквосполучення «бръ». Визначимо відстань між ними і знайдемо найбільший спільний дільник цих відстаней. У результаті отримаємо: 35, 85, 510. НСД = 5.

Отже, з певною ймовірністю можна визначити, що довжина ключового слова дорівнює 5.

2. Для підтвердження гіпотези використаємо математичну статистику для визначення довжини ключового слова. Для цього запишемо шифр-текст у таблицю с 5 стовпцями, враховуючи із попереднього пункту, що довжина ключового слова дорівнює 5.

Фрагмент розбитого тексту на 5 колонок:

Y1	Y2	Y3	Y4	Y5
В	л	ц	д	у
Т	ж	б	ю	ц
Х	ь	я	р	р
М	ш	б	р	х
Ц	э	о	о	э
Ц	г	б	р	ь
Ц	м	й	ф	к
Т	ь	ь	ю	ь
М	ш	э	с	я
Ц	п	у	н	у
Я	щ	э	й	т
А	ь	э	д	к
Ц	и	б	р	ь
Ц	г	б	р	п
А	ч	к	ь	у
Ц	п	ь	б	ь
С	э	г	к	ц
ь	г	у	у	щ
А	р	ц	е	э
В	ь	р	ю	у
О	ю	э	к	а
А	э	б	р	п
Я	ф	у	к	а
Б	ь	а	р	п
Я	ь	а	ф	к
ь	и	ь	ж	я
Ф	ф	и	й	о
Я	ф	м	и	б
Н	э	и	ф	у
Ю	г	б	р	ь
С	ш	ь	ж	э
Т	б	э	е	ч
Ю	ь	ю	р	ь
Е	г	о	ф	к
Б	ь	ч	я	б
А	ш	и	е	э
У	ь	ь	ю	а
Д	и	ч	ж	ч
У	ж	ц	е	э
В	л	р	и	ч
У	л	б	ю	п

3. Обчислимо Індикси відповідностей букв у кожному із стовпців таблиці, для встановлення точної довжини ключа. Для цього порахуємо частоту повторення букв в кожному стовпці. Таблиця складається із 5 стовпців, так як на попередньому етапі нами було встановлено, що ключове слово має довжину 5.

4. Далі можна зробити частотний аналіз кожної колонки, вибрати найчастіші букви і замінити їх на відповідну букву алфавіту (таких літер може бути кілька). Перебравши всі можливі варіанти, отримаємо ключове слово.

Частота повторення букв в стовпцях: 1 стовпець (загальна кількість букв $m=198$).

	0	1	2	3	4	5	6	7	8	9	10	11	12
буква	а	б	в	г	д	е	ё	ж	з	и	й	к	л
кількість	17	2	10	16	14	7	0	1	1	3	2	1	0

	13	14	15	16	17	18	19	20	21	22	23	24	25
буква	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш
кількість	3	4	1	0	1	16	9	14	5	5	23	0	0

	26	27	28	29	30	31	32
буква	щ	ъ	ы	ь	э	ю	я
кількість	5	10	3	2	2	10	11

$$I_{B1}(X) = \frac{\sum_{i=0}^{n-1} f_i(f_i - 1)}{m(m - 1)} = \frac{17 * 16 + 2 * 1 + \dots 11 * 10}{198 * 197} = 0,05676$$

Аналогічно складаємо таблицю для кожного стовпця і обчислюємо індекс відповідності:

$$I_{B2}(X) = 0,05896$$

$$I_{B3}(X) = 0,0634$$

$$I_{B4}(X) = 0,0581$$

$$I_{B5}(X) = 0,0723$$

За другим методом Фрідмана можна сказати, що довжина ключового слова вибрана правильно і дорівнює 5.

Так як кожний із стовпців є результатом шифрування фрагменту відкритого тексту простою заміною, то можемо оцінити ключове слово.

У першому стовпці найчастіше зустрічаються букви «ц»-порядковий номер 23 та «а» - порядковий номер 0 (їх може бути і більша кількість, тоді доцільно брати всі). Ці букви можуть бути буквою «о» після шифрування, тому щоб знайти ключ, використаємо формулу $K_i = (C_i - M_i) \bmod L$.

Тоді: $K_1 = (23 - 15) \bmod 33 = 8$, а це буква «з» або $K_1 = (0 - 15) \bmod 33 = 18$, а це буква «с».

У другій колонці «ъ»-«л», у третій «б», «у», «э»- «р», «е», «о», у четвертій – «ж» та «р» - «ш» та «в», і у п'ятій – «б», «у», «э» - «р», «е», «о».

Склавши таку відповідність для кожного стовпця, отримаємо:

$$\begin{pmatrix} 3 \\ c \end{pmatrix} + (л) + \begin{pmatrix} p \\ e \\ 0 \end{pmatrix} + \begin{pmatrix} ш \\ в \end{pmatrix} + \begin{pmatrix} p \\ e \\ 0 \end{pmatrix}$$

Перебравши всі можливі комбінації: злушу, зливо, слево, **слово**....слошу....

Знайдемо смислове ключове слово «**слово**»

Розшифруємо за допомогою формули: $M_i = (C_i - K_{i(modu)}) \bmod L$.

развебытьздоровымтожесамоечтонебытьбольнымопределенноздоровьеэто нечтобольшедлянасфизическоездоровьеэтоисостояниеиспособностьиэнергиязаниматьсятемчтонамнеобходимополучатьприэтомудовольствиеивыздоравливатьбезвсякойпомощиздоровьепарадоксальновынеможенепосредственнозаставитьсебястатьздоровымвамостаетсятольконаблюдатьзатемкакудивительнаяспособностьвашегоорганизмаисцелятьсебяначинаетдействоватьсамасобойивашебогатствоилибедностьжестокостьилидобродетельностьнеимеютздесьповидимомуникакогозначенияздоровьеэто нечтопозитивноеононеозначаетотказотудовольствияздоровьеявляетсяестественнымследствиемнашегоображизниивзаимоотношенийдиетыокружающейобстановкиздоровьеэто не предметсобственностиэто процессэто то что мыделаемрезультатнашихмыслейичувствэтообразсуществованияинтересночтонаправлениемедицинскихисследованийвсебольшеи большеотклоняетсяявсторонутойобластикотораядосихпорсчиталасьсферойдеятельно

5 КРИПТОАНАЛІЗ ПОТОКОВИХ ШИФРІВ

Потокові шифри перетворюють відкритий текст в шифротекст і навпаки побітно.

Більшість робіт, присвячених блочним шифрам орієнтовані на аналіз і синтез DES-подібних алгоритмів, а у поточковому крипто аналізі такого центру нема. Синтезні розв'язки та методи «злому» в поточкових шифрах різноманітні. У потокових шифрах сформувалась теорія побудови методів «злому», також визначений набір вимог, яким повинні задовольняти якісні схеми. Від стійких схем вимагається: великі періоди ПВП, висока лінійна складність ПВП.

Методи крипто аналізу поточкових шифрів реалізуються на основі одного із підходів [1]:

1. Використання статичних зв'язків гами.

В першу чергу досліджуються ймовірнісні характеристики гами. Її структура може мати деякі недоліки, які дозволяють отримати деяку інформацію про ключ. Найбільш очевидно- період ключа. Крім того, якщо генератор ПВП побудований некоректно, то згенеровані ним елементи можуть мати пряму або непряму залежність, що гарантує успіх крипто аналітику у зломі шифру.

2. Лінеаризація

Цей підхід зв'язаний із спробувати лінеаризувати рівняння гамутворення, тобто зведення задачі знаходження ключа до розв'язання деякої системи лінійних рівнянь (лінійний та диференціальний аналіз). При такому підході, основну відіграє лінійна складність досліджуваних послідовностей. Значення лінійної складності ПВП визначає порядок системи ліній рівнянь, яку потрібно розв'язати.

3. Використання комбінаторної залежності між елементами гами

Під час дослідження криптографічних властивостей поточкових шифрів, крім алгебраїчних і статистичних властивостей гами, необхідно враховувати наявність комбінаторної залежності між елементами гами. Наприклад, під час використання лінійної рекурентної послідовності із малим числом ненульових коефіцієнтів в законі рекурсії може виникнути ситуація, коли більшість елементів гами залежить від невеликої кількості елементів ключа. В такому випадку крипто аналітик має можливість припустити, що може знайти частину ключа, маючи статистичні властивості відкритих текстів.

Дослідження схем крипто аналізу для поточкових шифрів відбувається більш динамічно, ніж блочних. Таким дослідженням приділяється увага у європейських криптографічних центрах, у США робиться нахил у бік блочних шифрів.

Криптологічні дослідження поточкових шифрів являються джерелом ряду задач для фундаментальних напрямів дискретної математики:

- Задача аналізу властивостей регістра зсуву з лінійним оберненим зв'язком стимулювали дослідження лінійних рекурентних послідовностей над полями і кільцями.

- Задачі пошуку статистичних зв'язків між входом-виходом вузла, який реалізує дискретну функцію – побудову функції із заданими властивостями.

5.1. Дешифрування за методом гамування

У даного методу шифрування є два вразливих місця.

По-перше, якщо датчик випадкових чисел, використовуваний для створення гами побудований неграмотно і послідовно генеруються їм числа мають пряму чи непряму залежність один від одного, то це практично гарантує криптоаналітика успіх у зломі. Злом робиться таким чином:

1. Припускається, що в початковому тексті є будь-яка характерна фраза або окреме слово (чим довший фраза, тим краще) наприклад фраза «ЧЕКАЙТЕ подальших вказівок». Необхідно сказати, що переважна більшість документів в даний час зберігаються у вигляді файлів стандартних форматів, а формат файлу практично завжди визначається його текстовим заголовком або чим-небудь в тому ж роді.

2. У тексті шифровки шляхом послідовного підбору шукається рядок символів такої ж довжини, закон формування якої відповідає відомій особливості датчика ПСЧ. Нагадаємо, що текст шифровки виходить складанням за модулю 2 вихідного тексту і псевдовипадковою гами. Перший символ рядка вважається породжує, а наступні символи ітераційно обчислюються з нього.

3. Якщо шукана стрічка виявлена, то переходимо до наступного пункту, а якщо ні, то пробуємо знайти в тексті іншу характерну фразу.

4. Коли рядок символів виявлена, все інше вже просто і ясно - треба тільки добудувати псевдовипадкову послідовність вперед і назад, тим самим, відкривши вихідний текст.

Другий спосіб злому шифровки гамуванням припускає наявність можливості з боку криптоаналітика або його співника особисто посилати тестові повідомлення за криптографічним каналу, а потім, перехопивши їх, аналізувати вироблені шифроалгоритмом зміни. У такій ситуації, шляхом вирішення детермінованою математичної системи рівнянь, стає відомим алгоритм генерації навіть найкращого генератора ПВП. Далі, обґрунтовано вважаючи, що алгоритм генерації не буде змінюватися і в найближчому майбутньому, коли за криптографічним каналу буде передаватися не тестове повідомлення, а корисна інформація, дії по-злому будуть повністю аналогічні діям з попереднього методу.

5.2. Методи криптоаналізу поточкових шифрів

Розглянемо методи криптоаналізу сучасних поточкових шифрів (Таблиця 5.1). Варто зазначити, що більшість методів криптоаналізу дають змогу отримати результат лише, з певною ймовірністю, водночас повторення криптоаналітичного експерименту з іншими вхідними параметрами дає змогу підвищити ймовірність [7].

Таблиця 5.1

Методи криптоаналізу сучасних поточкових шифрів

Потоковий шифр	Дата створення	Розробник	Атака		Криптостійкість на даний час	Застосування
			Найбільш відома	Обчислювальна складність		
A5/1, A5/2	1987, 1989	Частина GSMстандарту	На основі відомих відкритих текстів (райдужні таблиці, —time-memory tradeoff)	2^{39}	ні	Шифрування голосу в GSM мережах
GEA5/1, GEA5/2	1993	Частина GSMстандарту	Атака —розділяй і володарюй (—time-memory trade off)	2^{45}	ні	Шифрування голосу в GSM мережах
FISH	1993	Siemens	Текстова атака	2^{11}	ні	Телефонія
RC4	1987	Рон Райвест	На основі відомих відкритих текстів	$2^{13} (2^{33})$	ні	Протоколи WEP SSL,
Scream	2002	Шай Холевї, Дон коперсміт та Чаранжит Жутала	-	-	так	Шифрування даних на жорстких магнітних дисках
SEAL	1997	U.S. Patent 5,454,039, U.S. Patent 5,675,652	На основі відомих відкритих текстів	-	ні	Шифрування даних на жорстких магнітних дисках
Salsa20	До 2004	Деніел Бернстейн	Метод ймовірно нейтральних бітів	2^{251} для 8 раундів	так	Проект eSTREAM (EU ECRYPT)
SOSEMA NUK	До 2004	Олівер Біллет, Ніколас Куртуа та ін. (без патентних обмежень)	-	-	так	3G-мережі

Найчастіше в ході криптоаналітичних досліджень використовуються диференціальний та лінійний методи, а також —mod n-атака та метод пов'язаних ключів. До відносно нових належить метод алгебраїчного криптоаналізу, який був

уперше запропонований Н. Куртуа [7]. На думку авторів даний метод є одним із найперспективніших, хоча його практичне використання й передбачає подальші дослідження. До його ключових особливостей у порівнянні з іншими варто віднести: універсальність (підходить як для потокових, так і для блокових шифрів), здатність масштабуватися, можливості подальшого вдосконалення. Алгебраїчні криптоаналітичні методи — це методи, які передбачають представлення криптографічних перетворень ключа, вхідних та вихідних даних для шифрування у вигляді деякого рівняння. Тоді сукупності таких перетворень формують систему рівнянь. На сьогодні є реалізації даного підходу для криптосистем, які мають практичне використання. Проте, його застосування для широкого кола шифросистем у багатьох випадках носить теоретичний характер, або не є в повній мірі досліджене на даний час. Загальна схема алгебраїчного криптоаналізу наведена на рисунку 5.1.

Можливість використання даного методу пов'язана з необхідністю розв'язання цілої низки теоретичних та прикладних задач, зокрема:

1. Представлення алгоритму шифрування у вигляді системи рівнянь у аналітичній формі.
 2. Перетворення аналітичної форми криптоалгоритму до кон'юнктивної нормальної форми.
 3. Розв'язання системи рівнянь у кон'юнктивній нормальній формі.
- Якщо п.1 і п.2 вимагають чіткої математичної формалізації криптоаналітичної системи, то п.3 пов'язаний з чисельним розв'язком відповідної системи рівнянь великої обчислювальної складності.

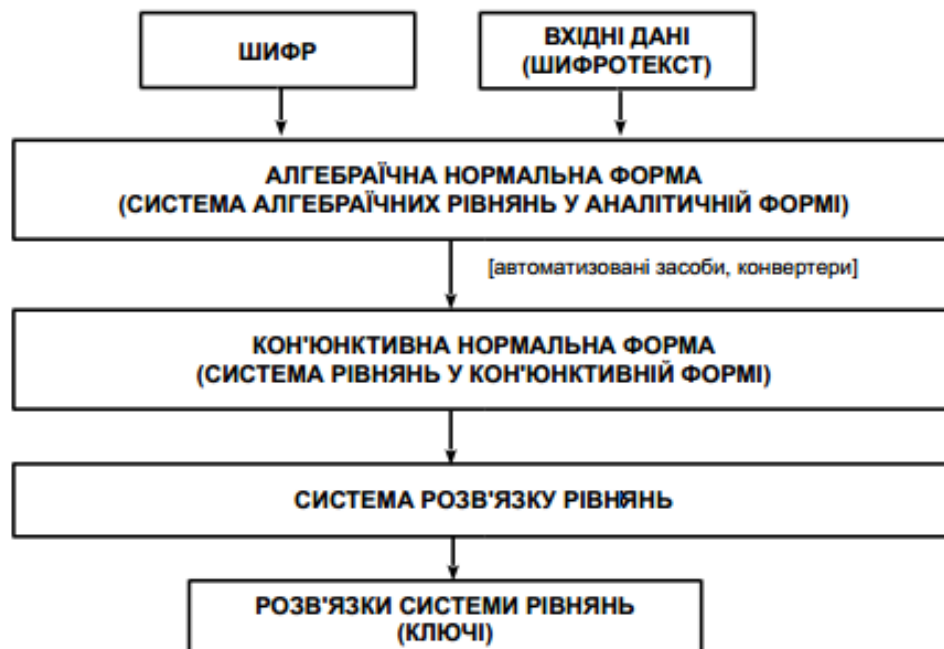


Рисунок 5.1 Загальна схема алгебраїчного криптоаналізу

6 ЛІНІЙНИЙ ТА ДИФЕРЕНЦІАЛЬНИЙ КРИПТОАНАЛІЗ

6.1. Розкриття шифрів блокових алгоритмів.

Одразу слід зазначити, що вірогідність успіху в дешифрування криптосистем, що належать до цієї групи методів досить мала і відмінна від нуля лише в разі використання ключів малої довжини. Основний метод злому, як це не сумно - повним перебором можливих ключів. На практиці, саме повним перебором були здійснені вдалі спроби розкриття алгоритму DES з 56-бітовим ключем. Для розкриття алгоритму DES будувалися спеціалізовані DES процесори і комп'ютери з мультипроцесорною архітектурою, тому, що цей алгоритм допускає повне розпаралелення. Проте, з розширенням ключа шифрування до 512, а сьогодні вже й до 1024 бітів злом шифрів не можливий.

Якщо у криптоаналітика є можливість (що на практиці практично не реально) самому виробляти шифрування заздалегідь відомих йому вихідних текстів і аналізувати одержувані тексти шифровок, та до того ж одним і тим же секретним ключем, то його завдання помітно спрощується. Для визначення невідомого йому секретного ключа в подібному випадку винайдені методики лінійного та диференційного аналізу. Лінійний аналіз заснований на аналізі статистичних параметрів вихідного і шифрованого текстів, а диференціальний аналіз відстежує зміни, що відбуваються в тексті шифровки, викликані зміною одного біта у вихідному тексті.

Єдине, що можна порадити людині, що займається розкриттям криптосистем даного типу, так це оптимізувати процес детектування осмисленості тексту. Слід знати, що при дешифруванні шифровки неправильним ключем, не виникає явною математичної або логічної помилки, а просто виходить помилковий, нічого не значущий текст або навіть не текст, а просто безглуздий набір символів. Для того щоб хоч якось позбавити оператора від необхідності прочитання всієї множини пропонованих машиною варіантів вихідних текстів, криптоаналітики вводять функцію P порогової ймовірності осмисленості тексту. Функція P визначає, скільки слів в отриманому тексті належить до загальноновживаних, а скільки ні, і якщо відсоток загальноновживаних слів більше порогового значення (зазвичай 85-95%), то текст можна пред'явити для подальшого знайомства людині. А вже вона виносить остаточний вердикт. При такому підході є звичайно ймовірність, що істинний вихідний текст буде забракованим як не смисловий (якщо в ньому занадто багато екзотичних слів), або що випадково попадеться смисловий текст, який буде дезінформацією. Однак іншого способу фільтра величезного обсягу інформації просто немає[2, 5].

6.2. Лінійний криптоаналіз

Метод лінійного криптоаналізу розроблений в 1993 році японським криптологом Мітсуро Матсуї. У початковому вигляді цей метод застосований до криптосистеми DES, в наш час створюються нові модифікації цього методу .

Ідея методу лінійного криптоаналізу заснована на тому, що існує можливість замінити нелінійну функцію криптографічного перетворення її лінійним аналогом. Лінійний криптоаналіз базується на знанні криптоаналітиком пар «відкритого тексту - криптограми», а також алгоритму шифрування.

Будемо рахувати, що при генеруванні вихідного тексту X , x_i – випадкові біти незалежні та однаково ймовірні $P(x_i = 1) = p$, $P(x_i = 0) = 1 - p$, $p = 0,5$.

Лінійним статистичним аналогом (або наближеним лінійним аналогом) називається вираз :

$$\lambda(X, Y) = \sum_{i=1}^n a_i x_i \oplus \sum_{j=1}^n b_j y_j = \sum_{k=1}^L c_k k_k \quad (6.1)$$

Ймовірність злому шифру

$$P\{\lambda(X, f(X, K)) = \sum_{k=1}^L c_k k_k\} = 0,5 + \Delta \quad (6.2)$$

Величина $\Delta = |1 - 2p|$ називається ефективністю лінійного аналогу, а коефіцієнти $a_i = 0$ або 1 ; $b_i = 0$ або 1 ; $c_k = 0$ або 1 – параметрами лінійного аналогу. По суті Δ характеризує степінь лінійності функції криптографічного перетворення та має максимальне значення $\Delta_{max} = 0,5$.

При застосуванні лінійного криптоаналізу вирішуються дві взаємопов'язані задачі:

1) Знаходження ефективного лінійного статистичного аналогу і обчислення його ймовірності.

2) Визначення ключа (або декількох біт ключа) з використанням ефективного лінійного статистичного аналогу.

Практична реалізація метода лінійного криптоаналізу зв'язана з реалізацією наступних послідовних кроків.

1. Ретельно аналізується криптографічна функція та визначається множина лінійних статистичних аналогів. На цьому кроці в першу чергу аналізується S - блоки функції ускладнення $\Psi(o)$. Для цього формується таблиця значень $Q_t(i, j)$, де: $t = 0, 1$ – номер S – блока, $i = 1, 4$, $j = 1, 2$. Значення $Q_t(i, j)$ представляє собою кількість збігів суми за $\text{mod} 2$ деяких бітів вхідних даних з сумою за $\text{mod} 2$ деяких бітів вихідних даних. В ході аналізу простежуються всі можливі комбінації двійкових векторів i, j . Кожна пара векторів використовується в якості маски, яка накладається на можливі пари «вхід – вихід» S – блоку.

Ці маски вказують на біти входу та виходу, які необхідно додати по $\text{mod} 2$, а після цього порівняти отриманні результати. Далі проводиться аналіз отриманих таблиць $Q_t(i, j)$ та знаходяться такі значення i^*, j^* , для яких виконується умова :

$$Q_t(i^*, j^*): \max |Q_t(i, j) - n_x|, n_x - \text{довжина під блоку (для S-DES дорівнює 8)}$$

У відповідності з отриманою парою i^*, j^* , та враховуючи в схемі алгоритму шифрування перестановки та додавання за модулем 2, формується ефективний лінійний аналог:

$$\lambda^*(X, Y) = \sum_{i=1}^n a_i^* x_i \oplus \sum_{j=1}^n b_j^* y_j = \sum_{k=1}^L c_k^* k_k, \text{ тоді } P_{\text{еф}} = \frac{Q(i^*, j^*)}{2^{n_x}} \quad (6.3)$$

2. Генеруємо множину незалежних вихідних текстів $X^{(1)}, X^{(2)}, \dots, X^{(M)}$ та реєструємо відповідні їм криптограми $Y^{(1)}, Y^{(2)}, \dots, Y^{(M)}$.

3. Для кожної пари $X^{(m)}, Y^{(m)}$, $m = \overline{1, M}$ обчислюються значення лівої частини ефективного лінійного статистичного аналогу :

$$\lambda^*(X^{(m)}, Y^{(m)}) = \sum_{i=1}^n a_i^* x_i^m \oplus \sum_{i=1}^n b_i^* y_i^m \quad (6.4)$$

4. Визначається частота отримання «1» при обчисленні М значень:

$$v = \frac{1}{M} \sum_{m=1}^M \lambda^*(X^{(m)}, Y^{(m)}) \quad (6.5)$$

Та будується оцінка максимальної правдоподібності у відповідності з правилом :

$$d = \begin{cases} 1, & v \geq 0,5 \\ 0, & v < 0,5 \end{cases} \quad (6.6)$$

Наприклад, N- кількість відкритих текстів (30), Т - кількість відкритих текстів, для яких ліва частина дорівнює 0 (22). Якщо $T > N/2$ і $p < 0.5$, тоді права частина дорівнює 1, інакше права частина дорівнює 0. Якщо $T < N/2$ і $p < 0.5$, тоді права частина дорівнює 0, інакше права частина дорівнює 1.

Обчислення кроку 3 та 4 виконується для всіх сформованих аналогів.

5. Будується система лінійних рівнянь, причому кожне рівняння системи приставляє собою рівність правій частині (6.4) та відповідності значенню (6.6)

$$\sum_{k=1}^L c_k^* k_k = d \quad (6.7)$$

Єдине рішення отриманої системи (6.7) використовується як оцінка ключа $k^* = k_1^*, k_2^*, \dots, k_L^*$.

Таким чином, на кроці 1 розв'язується задачі лінійного крипто аналізу, а кроки 2-5 забезпечується зв'язання другої задачі.

Криптоаналіз на базі S-DES (придатний до шифрів, які містять петлю Фейтеля).

Алгоритм шифрування: 2 раунди, 8 бітні блоки, 2 блоки заміни F_k (ф-я ускладнення), 10 бітний ключ (8 бітні раундові ключі)

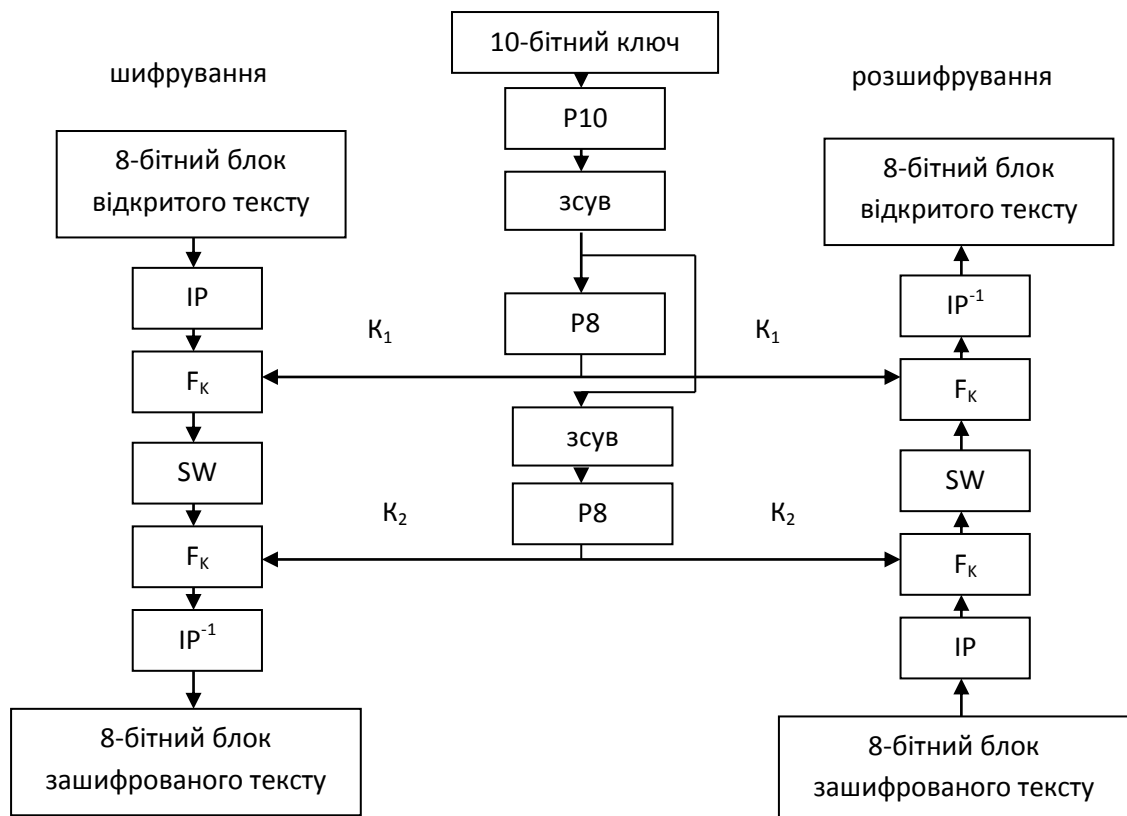


Рисунок 6.1 Схема алгоритму S-DES

У відповідності з алгоритмом лінійного криптоаналізу спочатку проводиться аналіз S-блоків — ф-я ускладнення.

$S_0 =$	№	0	1	2	3
	0	1	0	2	3
	1	3	1	0	2
	2	2	0	3	1
	3	1	3	2	0

$S_1 =$	№	0	1	2	3
	0	0	3	1	2
	1	3	2	0	1
	2	1	0	3	2
	3	2	1	3	0

Для простоти виконаємо лише 1 раунд. 4 біти \rightarrow 2 біти. Пара 0000 — 00 не виконається, бо ключ потрібно додавати.

Результати аналізу S_0 -блока та S_1 -блока (таблиця 6.1).

Таблиця 6.1

Аналіз S -блоків

Q_0					Q_1				
i		Знач i			i		Знач i		
S_{ij}		01	10	11	S_{ij}		01	10	11
000		8	8	8	0001		8	8	8
0010		8	6	6	0010		6	8	10
0011		8	6	6	0011		6	1	6
0100		6	10	8	0100		8	8	8
0101		14	10	8	0101		8	12	12
0110		6	8	6	0110		10	8	6
0111		6	8	6	0111		10	8	6
1000		8	8	8	1000		8	8	8
1001		8	8	8	1001		8	8	8
1010		8	10	10	1010		10	12	10
1011		8	10	2	1011		10	8	6
1100		6	10	8	1100		8	4	12
1101		6	10	8	1101		8	8	8
1110		6	4	10	1110		6	8	10
1111		6	12	10	1111		14	8	10

Як бачимо в $Q_0(i, j)$ можна виділити 4-ефек. аналоги (i,j):

(0101,01), (1011,11), (1110,10), (1111,10)

У $Q_1(i, j)$ — 7 ефек. (i,j):

(0011,10), (0101,10), (0101,11), (1010,10), (1100,10), (1100,11), (1111,01)

Розглянемо першу пару $S_0(0101,01)$. Очевидно, що $x_1 \oplus x_3 = Y_1$ вик з ймов $p = 1^4/16$, а відповідно $\Delta = |1 - 2p| = 3/4$. Тоді можна записати 1-е рівняння.

$$x_1 \oplus x_3 \oplus Y_1 = k_1 \oplus k_3 \quad (6.1)$$

Аналогічно записуємо інші рівняння.

$$\begin{aligned}x'_0 \oplus x'_2 \oplus x'_3 \oplus Y'_0 \oplus Y'_1 &= k_0 \oplus k_2 \oplus k_3 \\x'_0 \oplus x'_2 \oplus x'_3 \oplus Y'_0 &= k_0 \oplus k_1 \oplus k_2 \\x'_0 \oplus x'_2 \oplus x'_3 \oplus Y'_0 &= k_0 \oplus k_1 \oplus k_2 \oplus k_3\end{aligned}\tag{6.2}$$

Із блока S_1 складемо сім рівнянь.

$$\begin{aligned}x'_6 \oplus x'_7 \oplus Y'_2 &= k_6 \oplus k_7 \\x'_5 \oplus x'_7 \oplus Y'_2 &= k_5 \oplus k_7 \\x'_5 \oplus x'_7 \oplus Y'_2 \oplus Y'_3 &= k_5 \oplus k_7 \\x'_4 \oplus x'_6 \oplus Y'_2 &= k_4 \oplus k_6 \\x'_4 \oplus x'_5 \oplus Y'_2 &= k_4 \oplus k_5 \\x'_4 \oplus x'_5 \oplus Y'_2 \oplus Y'_3 &= k_4 \oplus k_5 \\x'_4 \oplus x'_5 \oplus x'_6 \oplus x'_7 \oplus Y'_3 &= k_4 \oplus k_5 \oplus k_6 \oplus k_7\end{aligned}\tag{6.3}$$

Розглянемо початковий текст $x_0, x_1, x_2, x_3, x_4, x_5, x_6, x_7$ до S-перетворень включно.

- Виконаємо перестановку IP(7 6 4 0 2 5 1 3). Після цього отримуємо $x_7, x_6, x_4, x_0, x_2, x_5, x_1, x_3$.

- Потім текст розбивають на ліву і праву частини : x_7, x_6, x_4, x_0 і x_2, x_5, x_1, x_3 .

- Права частина перетворюється за функцією розширення E отримуємо $(x_3, x_2, x_5, x_1, x_5, x_1, x_3, x_2)$ який додаємо \oplus до ключа $k_0 k_1 k_2 k_3 | k_4 k_5 k_6 k_7$. На блок S_0 подаємо $x_3 x_2 x_5 x_1 \oplus k_0 k_1 k_2 k_3$, на S_1 : $x_5 x_1 x_3 x_2 \oplus k_4 k_5 k_6 k_7$.

- Для S_0 блока отримаємо вхідні елементи x'_0, x'_1, x'_2, x'_3
 $x'_0 = x_3 \oplus k_0, x'_1 = x_2 \oplus k_1, x'_2 = x_5 \oplus k_2, x'_3 = x_1 \oplus k_3$

- Для S_1 блока: x'_4, x'_5, x'_6, x'_7
 $x'_4 = x_5 \oplus k_4, x'_5 = x_4 \oplus k_5, x'_6 = x_3 \oplus k_6, x'_7 = x_2 \oplus k_7$

Нехай криптограма має вигляд $Y_0 Y_1 Y_2 Y_3 Y_4 Y_5 Y_6 Y_7$. Вигляд інформ. блоку після кінц. перестановки $IP^{-1}(7 6 4 0 2 5 1 3)$ має вигляд $Y_7 Y_6 Y_4 Y_0 Y_2 Y_5 Y_1 Y_3$ (бо йдемо назад, тому $IP^{-1} = IP$)

- Розбиваємо блок на ліву: $Y_7 Y_6 Y_4 Y_0$ і праву: $Y_2 Y_5 Y_1 Y_3$ частини.

Права частина отримується в результаті $L \oplus F^{R+K}$ лівої частини вихідного тексту після перестановки IP($x_7 x_6 x_4 x_0$) з текстом $(x_2 x_5 x_1 x_3)$ отримуємо Y'_0, Y'_1, Y'_2, Y'_3 , після перестановки P(1 0 3 2) отримуємо Y'_1, Y'_0, Y'_3, Y'_2 . В результаті можна записати:

$$Y'_1 = x_7 \oplus Y_2, Y'_0 = x_6 \oplus Y_5, Y'_3 = x_4 \oplus Y_1, Y'_2 = x_6 \oplus Y_3\tag{6.6}$$

- Підкладемо отримані рівняння (6.4- 6.6) у (6.1-6.3) і отримаємо лінійні рівняння статичних аналогів. Отримані результати запишемо в таблицю 6.2.

На цьому етапі алгоритму лінійного крипто аналіз завершений. Вибравши в якості ключа шифрування $k = 0010001111$ – довільним чином, реалізуємо 2, 3, 4 етапи:

У результаті отримаємо систему рівнянь (п'ятий етап):

$$\begin{aligned}k_0 + k_2 + k_5 &= 0, k_1 + k_3 = 0, k_0 + k_1 + k_2 = 0, k_0 + k_1 + k_2 + k_3 = 0, k_6 + k_7 = 1 \\k_5 + k_7 &= 1, k_4 + k_6 = 1, k_4 + k_5 = 1, k_4 + k_5 + k_6 + k_7 = 0\end{aligned}$$

Лінійні статистичні аналоги

№ блока	Множина рівнянь	P	$\Delta = 1 - 2p $
S_0	$x_1 \oplus x_3 \oplus \dot{Y}_1 = k_1 \oplus k_3$	7/8	3/4
	$\dot{x}_0 \oplus \dot{x}_2 \oplus x_3 \oplus \dot{Y}_0 \oplus \dot{Y}_1 = k_0 \oplus k_2 \oplus k_3$	1/8	3/4
	$\dot{x}_0 \oplus \dot{x}_2 \oplus x_3 \oplus \dot{Y}_0 = k_0 \oplus k_1 \oplus k_2$	1/4	1/2
	$\dot{x}_0 \oplus \dot{x}_2 \oplus x_3 \oplus \dot{Y}_0 = k_0 \oplus k_1 \oplus k_2 \oplus k_3$	3/4	1/2
S_1	$\dot{x}_6 \oplus \dot{x}_7 \oplus \dot{Y}_2 = k_6 \oplus k_7$	3/4	1/2
	$\dot{x}_5 \oplus \dot{x}_7 \oplus \dot{Y}_2 = k_5 \oplus k_7$	3/4	1/2
	$\dot{x}_5 \oplus \dot{x}_7 \oplus \dot{Y}_2 \oplus \dot{Y}_3 = k_5 \oplus k_7$	3/4	1/2
	$\dot{x}_4 \oplus \dot{x}_6 \oplus \dot{Y}_2 = k_4 \oplus k_6$	3/4	1/2
	$\dot{x}_4 \oplus \dot{x}_5 \oplus \dot{Y}_2 = k_4 \oplus k_5$	1/4	1/2
	$\dot{x}_4 \oplus \dot{x}_5 \oplus \dot{Y}_2 \oplus \dot{Y}_3 = k_4 \oplus k_5$	3/4	1/2
	$\dot{x}_4 \oplus \dot{x}_5 \oplus \dot{x}_6 \oplus \dot{x}_7 \oplus \dot{Y}_3 = k_4 \oplus k_5 \oplus k_6 \oplus k_7$	7/8	3/4

Після розв'язання системи лінійних рівнянь, отримаємо:

$$\begin{aligned}
 k_1 &= 10101001, k_2 = 01010101, k_3 = 10011001, k_4 = 01100101 \\
 k_5 &= 10100101, k_6 = 01011001, k_7 = 10010101, k_8 = 01101001 \\
 k_9 &= 10101010, k_{10} = 01010110, k_{11} = 10011010, k_{12} = 01100110 \\
 k_{13} &= 10100110, k_{14} = 01011010, k_{15} = 10010110, k_{16} = 01101010
 \end{aligned}$$

Аналіз розв'язку системи дозволяє вилучити раундовий ключ $k_1 = 10101001$, а потім виконати алгоритм формування ключів в S-DES і методом повного перебору – ключ криптосистеми [5].

6.3. Диференціальний криптоаналіз

Метод диференційного (різницевого) криптоаналізу запропонував Є.Байхам і А.Шамір, і за думкою ряду спеціалістів компанії IBM, є загальним методом криптоаналізу блочно – ітераційних криптосистем. Ідея полягає в аналізі процесу зміни відмінностей для пар відкритих текстів $\Delta X = X \oplus X'$, маючих певні вихідні відмінності, в процесі проходження через цикли шифрування з одним і тим же ключем.

Розбір методу диференціального криптоаналізу розглянемо на прикладі S-DES.

Нехай задана пара входів X та X' , з відмінністю $\Delta X = X \oplus X'$. Відомі перестановки IP та перестановки з розширенням E , а отже, відомі і відмінності ΔA на вході блоків заміни S_0 та S_1 . Виходи Y і Y' відомі, отже, відомі і відмінності $\Delta Y = Y \oplus Y'$. При відомих перестановках IP^{-1} і P відомі і відмінності ΔC на виході блоків заміни S_0 і S_1 .

Доведено, що для будь-якого заданого ΔA не всі значення ΔC рівновірогідні. Комбінації ΔA і ΔC дозволяють допустити значення бітів для $E(X) \oplus k_i$ і $E(X') \oplus k_i$. То, що $E(X)$ і $E(X')$ відомі, дає інформацію про k_i .

Відмінність різних пар відкритих текстів приводить до відмінності отриманих криптограм з певною вірогідністю. Ці ймовірності можливо знайти,

побудувавши таблиці для кожного з блоків заміни. Таблиці будуються за наступним принципом: по вертикалі розташовуються всі можливі комбінації ΔA , по горизонталі – всі можливі комбінації ΔC , а на перетині – число відповідностей даного ΔC даному ΔA .

Кількість найбільших збігів вказує нам пару ΔA і ΔC , з допомогою яких можливо визначити секретний ключ. Пара відкритих текстів, відповідних даним ΔA і ΔC називаються правильною парою, а пара відкритих текстів, невідповідних даним ΔA і ΔC - неправильною парою. Правильна пара підкаже правильний ключ циклу, а неправильна пара – випадковий. Щоб знайти правильний ключ, необхідно просто зібрати достатнє число припущень. Один з під ключів буде зустрічатися частіше, ніж всі інші. Фактичний правильний під ключ з'явиться з всіх можливих випадкових під ключів.

Розглянемо цей приклад диференціального криптоаналізу:

Аналізуються блоки заміни, формуються таблиці залежностей ΔA і ΔC .

На вхід блоку заміни подається 4 біти, що їх розмірність суми за модулем 2 теж не більше 4 біти. Таким чином, діапазон ΔA (0000–1111). 0000 – не враховується для аналізу, бо повинна відрізнитися хоча б одним бітом, тому діапазон для ΔA (0001 – 1111). Кожне таке значення може бути отримане 16 можливими комбінаціями вхідних даних блоків заміни.

Наприклад, $\Delta A = 0001$ може бути отримане наступним чином $0000 \oplus 0001$, $0001 \oplus 0000$, $0010 \oplus 0011$, ... $1111 \oplus 1110$.

Сума виходів ΔC , отриманих після проходження через блоки заміни не завжди рівна блоку заміни іншої пари. Результати аналізу S_0, S_1 блоків записані в таблиці 6.3.

У цій таблиці визначаємо найкращі ΔA і ΔC .

Для S_0 : (0100,01); (1011,11); (1111,10),

для S_1 : (0110,10); (1001,10); (1111,00).

Так як ΔA дорівнює сумі за модулем 2 переставлених і розширених біт, то можна виділити єдине $\Delta A = 1111|1111$, яке відповідає $\Delta C = 10|00$.

Знаючи найкраще значення пари ($\Delta A, \Delta C$) можна знайти ключ. Для цього нам необхідно декілька пар відкритих текстів (X_1, X_2), таких що:

$$\Delta A = E(X_1) \oplus E(X_2) = 11111111,$$

$$\text{а } \Delta C = S(E(X_1)) \oplus S(E(X_2)) = 1000.$$

Для того, щоб із зашифрованого X_1 вилучити $S(E(X_1))$ необхідно до останніх 4-х біт шифру повідомлення додати перші 4-ри біти, потім врахувати перестановку. Результати записані в таблицю 6.4 [5].

Таблиця 6.3

$\Delta C(\Delta A)$ в блоках									
S_0	ΔC				S_1	ΔC			
ΔA	00	01	10	11	ΔA	00	01	10	11
0001	6	6	2	2	0001	2	6	2	6
0010	0	4	8	4	0010	0	8	0	8
0011	2	2	6	6	0011	6	2	6	2
0100	0	12	0	4	0100	0	8	4	4
0101	6	6	2	2	0101	6	2	6	2
0110	0	0	8	8	0110	0	0	12	4
0111	2	2	6	6	0111	2	6	2	6
1000	4	8	4	0	1000	2	6	2	6
1001	2	2	6	6	1001	0	0	12	4
1010	2	2	6	6	1010	6	2	6	2
1011	0	4	0	12	1011	0	8	4	4
1100	6	6	2	2	1100	6	2	6	2
1101	8	4	0	4	1101	4	4	0	8
1110	2	2	6	6	1110	2	6	2	6
1111	4	0	12	0	1111	12	4	0	0

Таблиця 6.4

Пари (x_1, x_2) , що відповідають найкращим $(\Delta A, \Delta C)$

№	X1	E(X1)	S(E(X1))	Y1
1	00011001	11000011	1011	01000110
2	11111001	11000011	1011	00000000
	X2	E(X2)	S(E(X2))	Y2
1	01000110	00111100	0011	11010111
2	01000110	00111100	0011	11010111

Для блоку S_0 :

Пара (00011001, 01000110): $1100 \oplus k_1$ дає на виході 10; $0011 \oplus k_1$ дає 00.

На виході блоку S_0 значення 10 отримуємо в тому випадку, коли на його вході 0100, 0101, 1000, 1101, а значення 00 – при входних 0010, 0111, 1010, 1011.

Складаємо можливі варіанти

$1100 \oplus k_1=0100, k_1=1000$	$1100 \oplus k_1=0101, k_1=1001$
$1100 \oplus k_1=1000, k_1=0100$	$1100 \oplus k_1=1101, k_1=0001$
$1100 \oplus k_1=0010, k_1=0001$	$1100 \oplus k_1=0111, k_1=0100$
$1100 \oplus k_1=1010, k_1=1001$	$1100 \oplus k_1=1011, k_1=1000$

Пару (11111001, 01000110) розглядати не потрібно, так як перші чотири біти E(X1) та E(X2) будуть аналогічні тим самим бітам, як і в попередньої пари.

Для блоку S_1 :

Пара (00011001, 01000110): $1100 \oplus k_2$ дає на виході 11; $0011 \oplus k_2$ дає 11.

На виході блоку S_1 значення 11 отримуємо в тому випадку, коли на його вході 0001, 0010, 1100, 1101, а значення 11- 0001, 0010, 1101. Складаємо можливі варіанти

$1100 \oplus k_2=0001, k_2=1101$	$1100 \oplus k_2=0010, k_2=1110$
$1100 \oplus k_2=1101, k_2=0001$	$0011 \oplus k_2=0001, k_2=0010$
$0011 \oplus k_2=0010, k_2=0001$	$0011 \oplus k_2=1100, k_2=1111$
$0011 \oplus k_2=1101, k_2=1110$	$1100 \oplus k_1=1011, k_2=1000$

Аналогічно пару (11111001, 01000110) розглядати не потрібно.

Обєднаємо результати аналізу і отримаємо найбільш ймовірні ключі:

$k_1=10000001, k_2=10010001, k_3=01000001, k_4=00010001, k_5=10001110, k_6=10011110, k_7=01001110, k_8=00011110$.

У результаті перевірки k_7 виявився раундовим ключем.

6.4 Напрямки розвитку лінійного та диференціального криптоаналізу

1) \oplus замінюють на $\oplus 2^{32}$.

2) У 1993 році ізраїльський математик Бен-Ароста і Біхам запропонували знаходити різниці характеристики, рахуючи, що ключ приймає не всі можливі значення, а лише значення із деякої підмножини. Цей метод отримав назву «метод умовних диференціалів».

3) У 1994 р. датський математик Ларс Кнурсен запропонував будувати по аналогії метод «зрізаних диференціалів», слідкувавши лише за частиною бітів.

4) У 1994 р. швейцарський криптограф Лан показав, що для побудови методу можна замість пар використовувати різниці характеристики вищих порядків.

5) У 1998р. Біхам, Бірюков і Шамір помітили, щодля побудови криптоаналізу можна використовувати диференціали, які мають не підвищену ймовірність появи, а понижену – 0(неможливі диференціали). Цим методом була визначена слабкість крипто алгоритму skipjack.

7 КРИПТОАНАЛІЗ СИСТЕМ З ВІДКРИТИМ КЛЮЧЕМ

Для систем з відкритим ключем застосовуються лінійний і диференціальний методи криптоаналізу, при умові, що зломщик системи має можливість безперешкодно подавати на вхід криптосистеми свої вхідні дані і аналізувати отриманий результат шифрування. Додатково до цієї умови він повинен бути впевнений, що шуканий ним ключ шифрування не змінюється в процесі спроб, а також в тому, що він не зміниться і після того, як шифровані повідомлення почне посилати легальний відправник. У будь-якому іншому випадку дані два методи просто безсилі. Зайвим підтвердженням тому служать заяви фірм-розробників криптосистем про величезні грошові винагороди, які чекають вдалого зломщика у разі злому крипто алгоритму.

Крім загальноприйнятих методів лінійного і диференціального криптоаналізу, до кожної конкретної системи з відкритим ключем можуть застосовуватись свої спеціалізовані методи злому. Так, наприклад, у деяких криптосистем є «слабкі» ключі. Імовірність випадкового вибору таких ключів повинна бути зіставлена з необхідним ступенем криптозахищеності і за умови загрози, в криптосистемах повинно бути передбачене відбраковування «слабких» ключів. Однак, якщо таке відбраковування в криптосистемах не передбачене, то можна організувати полювання на «слабкий» ключ.

Практично всі використовувані алгоритми асиметричного криптоаналізу засновані на задачах факторизації і дискретного логарифмування в різних алгебраїчних структурах. Незважаючи на те, що приналежність цих задач до класу NP-повних задач не доведена, на сьогоднішній день не знайдений поліноміальний алгоритм їх розв'язку [1].

Криптоаналіз систем шифрування на основі складної задачі дискретного логарифмування.

Останні досягнення теорії складності показали, що загальна проблема логарифмування в кінцевих полях не може рахуватись досить міцним фундаментом. Найбільш ефективні на сьогоднішній день алгоритми мають вже експоненціальну, або субекспоненціальну часову складність. Це алгоритми «index-calculus» вперше запропоновані Адлеманом. Основна ідея полягає в тому, що якщо

$$\prod_{i=1}^m x_i = \prod_{j=1}^n y_j$$

для деяких елементів скінченного поля Z_p , то

$$\sum_{i=1}^m \log_a x_i = \sum_{j=1}^n \log_a y_j \bmod (p-1) \quad (7.1)$$

Отримавши багато співвідношень (1) можна розв'язати систему лінійних рівнянь відносно $\log_a x_i$ та $\log_a y_j$ в кільці залишків Z_{p-1} .

Алгоритмів, які здійснюють дискретне логарифмування на еліптичних кривих в загальному випадку, хоча б субекспоненціальної складності, на сьогоднішній день не існує. Відомі роботи «Криптоаналіз систем шифрування,

заснованих на складності задачі факторизації» У. А. Семаєва, в яких він розглядає метод ідейно близький до методу Адлемана.

Криптоаналіз систем шифрування, заснованих на складності задачі факторизації.

Ці методи зводяться до пошуку ефективніших способів розкладання цілих чисел на множники. Найбільш очевидний метод розкладу числа n на множники – перебір простих дільників, що не перевищують \sqrt{n} . Є також інші методи, наприклад метод Ферма, який заснований на представленні числа n у вигляді різних квадратів числа n :

$$n = a^2 - b^2.$$

По складності, як простий метод, так і метод Ферма оцінюються величиною $O(\sqrt{n})$, однак останній метод може бути ефективніший, коли прості множники близькі один до одного. Для великих чисел n час роботи таких алгоритмів стає неприйнятним.

Є більш ефективні методи розкладу числа на прості множники – це методи, які мають субекспоненціальну складність. Наприклад, ймовірнісний алгоритм Хенстри за допомогою еліптичних кривих, метод Діксона та інші.

7.1 Атаки на криптосистему RSA

7.1.1 Розклад n на множники. Метод Ферма

Використаємо близькі прості значення p і q , де $n = p * q$.

Припустимо, що $p > q$. Тоді можемо записати $x = \frac{p+q}{2}$; $y = \frac{p-q}{2}$, де $n = x^2 - y^2$. Для знаходження p і q достатньо підібрати x і y , що задовольняють дане рівняння. У такому випадку:

$$\begin{cases} p = x + y \\ q = x - y \end{cases}$$

Для початку знаходимо $x = \sqrt{n}$, та обернемо до нього найближче ціле x_1 , знаходимо y_1 , і тоді, поки не виконується умова - x , y – цілі; $n = x^2 - y^2$.

Приклад 7.1

Нехай $n = 851$.

$$x_0 = \sqrt{n} = 29,17.$$

$$x_1 = 30 ; y_1^2 = 30^2 - 851 = 49 \rightarrow y_1 = 7.$$

Отже,

$$\begin{cases} p = x + y = 30 + 7 = 37 \\ q = x - y = 30 - 7 = 23 \end{cases}$$

$$\text{Перевірка: } 23 * 37 = 851.$$

Приклад 7.2

Нехай $n = 1219$.

$$x_0 = \sqrt{n} = 34,9.$$

$$x_1 = 35 ; y_1^2 = 35^2 - 1219 = 6 \rightarrow y_1 = \sqrt{6} -.$$

$$x_2 = 36 ; y_2^2 = 36^2 - 1219 = 77 \rightarrow y_2 = \sqrt{77} -.$$

$$x_3 = 37 ; y_3^2 = 37^2 - 1219 = 150 \rightarrow y_3 = \sqrt{150} -.$$

$$x_4 = 38; y_4^2 = 38^2 - 1219 = 225 \rightarrow y_4 = 15.$$

Отже,

$$\begin{cases} p = x + y = 38 + 15 = 53 \\ q = x - y = 38 - 15 = 23 \end{cases}$$

$$\text{Перевірка: } 53 * 23 = 1219.$$

Захист від такої крипто атаки: n – дуже велике число.

7.1.2 Атака з вибраним шифром

Нехай ми на стороні А перехопили шифр $C = E_e(M)$.

Оберемо $r < n$, НСД(n, r) = 1.

$$\text{Обчислюємо } x = r^e \bmod(n), \begin{cases} x^d = r^{e \cdot d} \bmod(n) = r \bmod(n) \\ x^d = r \bmod(n) \\ y = x \cdot C \bmod(n) \end{cases}$$

$$t = r^{-1} \bmod(n) \text{ (тобто } t \cdot r = 1 \bmod(n))$$

Просимо абонента В підписати y приватним ключем d . Тоді, отримуємо

$$m = y^d \bmod(n).$$

Далі із рівняння $tm \bmod(n)$ знаходимо M . Доведемо це:

$$\begin{aligned} t \cdot m \bmod(n) &= r^{-1} y^d \bmod(n) = r^{-1} x^d c^d \bmod(n) = \\ &= r^{-1} r \cdot c^d \bmod(n) = 1 \cdot c^d \bmod(n) = c^d \bmod(n) = M. \end{aligned}$$

Приклад 7.3

А	В
<p>Перехопив: $C = 26; n = 33; e = 3$.</p> <p>Вибрав $r < n$: $r = 10$.</p> <p>Обчислює: $x = r^e \bmod(n) = 10^3 \bmod(33) = 10$ $y = x \cdot c \bmod(n) = 10 \cdot 26 \bmod(33) = 29$ $t \cdot r = 1 \bmod(n) \rightarrow 10 \cdot t = 1 \bmod(33) \rightarrow t = 10$</p>	<p>Секретні $\begin{cases} M = 5 \\ d = 7 \end{cases}$</p>
Просить В підписати $y = 29$	$m = y^d \bmod(n) = 29^7 \bmod(33) = 17$
Із рівняння $t \cdot m \bmod(n) = 10 \cdot 17 \bmod(33) = 5$ знаходимо $M = 5$.	

Захист від такої атаки:

- пари ключів для шифрування та ЦЕП повинні бути різні;
- підписувати лише хеш $h(M)$.

7.2 Атаки на ЦЕП RSA

7.2.1 Мультиплікативна атака

Якщо зломиснику потрібно отримати ЦЕП до повідомлення M , він створює два чи більше повідомлень M_1, M_2 , так що $M = M_1 \cdot M_2 \bmod(n)$ і підписує кожне окремо:

$$\begin{aligned} M_1^d \bmod(n) \\ M_2^d \bmod(n) \end{aligned}$$

$$\begin{aligned}\text{Далі отримує } M^d &= (M_1^d \bmod(n)) \cdot (M_2^d \bmod(n)) = \\ &= (M_1 \cdot M_2)^d \bmod(n).\end{aligned}$$

Приклад 7.4

$$M_1 = 2, M_2 = 4; M = 8, n = 33$$

// Секретне: $d = 7$

$$M_1^d \bmod(n) = 2^7 \bmod(33) = 29;$$

$$M_2^d \bmod(n) = 4^7 \bmod(33) = 16;$$

$$M = 29 \cdot 16 \bmod(33) = 2 \quad // \text{що теж саме, що і}$$

$$M = 8^7 \bmod(33) = 2$$

7.2.2 Атака на спільний модуль

Всім відомий спільний модуль n і є різні пари e та d . Ця схема працює до тих пір, поки два користувачі не зашифрують одне й теж саме повідомлення на різних ключах, тоді можлива атака:

M, e_1, e_2, n

M – повідомлення (секретне);

e_1, e_2, n – ключі, спільний модуль (відкриті).

Два користувачі його зашифрували:

$$C_1 = M^{e_1} \bmod(n) \text{ і } C_2 = M^{e_2} \bmod(n).$$

Криптоаналітик володіє знаннями про n, e_1, e_2, C_1, C_2 і може виконати атаку наступним чином: За алгоритмом Евкліда знаходить r і s із рівняння:

$$s \cdot e_1 + r \cdot e_2 = 1 \bmod(n) \quad (r < 0, s > 0)$$

$$\text{Знаходить } C_2^{-1} \bmod(n) \left(C_2^{-1} \cdot C_2 \bmod(n) \right)$$

Тоді знаходить M .

$$C_1^s \cdot (C_2^{-1})^{-r} \bmod(n) = M.$$

Доведемо це

$$\begin{aligned}(M^{e_1})^s \cdot ((M^{e_2})^{-1})^{-r} \bmod(n) &= M^{e_1 \cdot s} \cdot M^{e_2 \cdot r} \bmod(n) = M^{e_1 \cdot s + e_2 \cdot r} \bmod(n) \\ &= M^1 \bmod(n) = M.\end{aligned}$$

Захист – задавати спільний модуль для групи користувачів не можна.

7.2.3 Атака дешифрування ітераціями

Перехоплену криптограму повторно шифрують на публічному ключі і при деякій кількості ітерацій отримують вихідне повідомлення (зв'язний текст). Кількість ітерацій залежить від довжини ключа (e) і модуля (n).

Приклад 7.5

$$\text{Нехай } C = 29; n = 33; e = 7.$$

// Секретна інформація: $M = 2$

$$\text{I ітерація: } M_1 = 29^7 \bmod(33) = 17;$$

$$\text{II ітерація: } M_2 = 17^7 \bmod(33) = 8;$$

$$\text{III ітерація: } M_3 = 8^7 \bmod(33) = 2 \rightarrow M.$$

8 КРИПТОАНАЛІЗ ЗА ПОБІЧНИМИ КАНАЛАМИ

Останнім часом одним з найбільш актуальних напрямків криптоаналізу стало здійснення атак, що використовують особливості реалізації та робочого середовища. Атаки по стороннім, або побічним, каналах — це вид криптографічних атак, що використовують інформацію, отриману по стороннім або побічним каналам. Під інформацією з побічних каналів розуміється інформація, яка може бути отримана з пристрою шифрування і не є при цьому ні відкритим текстом, ні шифротекстом[1].

Майже всі здійснені на практиці вдалі атаки на криптосистеми використовують слабкості в реалізації та розміщення механізмів криптоалгоритма. Такі атаки засновані на кореляції між значеннями фізичних параметрів, вимірюваних в різні моменти під час обчислень (споживання енергії, час обчислень, електромагнітне випромінювання тощо), і внутрішнім станом обчислювального пристрою, що має відношення до секретного ключа. На практиці атаки по побічних каналах на багато порядків більш ефективні, ніж традиційні атаки, засновані на математичному аналізі. При цьому атаки по побічних каналах використовують особливості реалізації (тому їх іноді називають також називають атаками на реалізацію - *implementation attacks*) для отримання секретних параметрів, задіяних в обчисленнях. Такий підхід менш узагальнений, оскільки прив'язаний до конкретної реалізації, але найчастіше більш потужний, ніж класичний криптоаналіз.

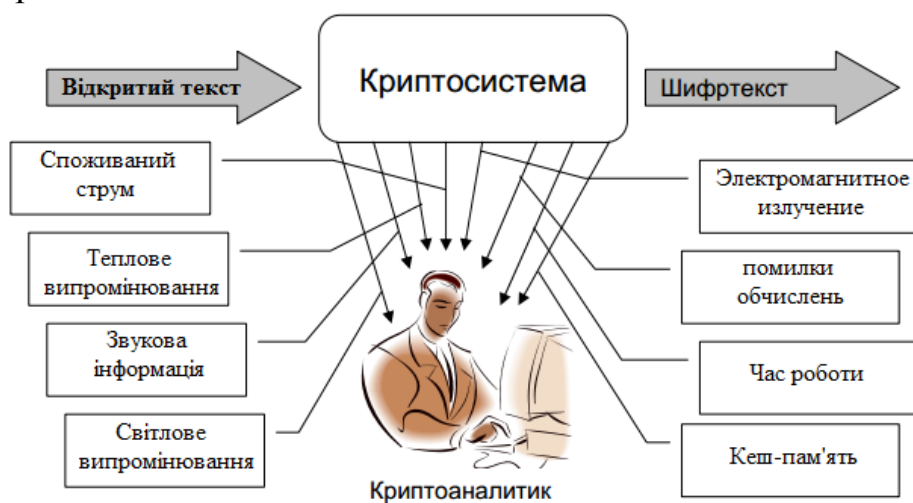


Рисунок 8.1 – Види атак

В останні роки різко зросла кількість криптографічних атак, що використовують особливості реалізації та робочого середовища. Наприклад, противник може відстежувати енергію, споживану смарт-картою, коли вона виконує операції з закритим ключем, такі, як розшифрування або генерація підпису. Противник може також заміряти час, що витрачається на виконання криптографічної операції, або аналізувати поведінку криптографічного пристрою при виникненні певних помилок. Побічну інформацію на практиці зібрати часом нескладно, тому потрібно обов'язково враховувати таку загрозу при оцінці захищеності системи.

Атаки по побічних каналах трьом класифікуються за такими типами:

- За контролем над обчислювальним процесом: пасивні та активні.
- за способом доступу до модулю: агресивні (invasive), напів агресивні (semi-invasive) і неагресивні (non-invasive).
- за методом, що застосовується в процесі аналізу: прості – simple side channel attack (SSCA) та різницеві – differential side channel attack (DSCA).

На сьогоднішній день виділено більше десяти побічних каналів. Атаки розрізняються по виду використовуваного побічного каналу (рис. 1): атаки по часу виконання (Timing Attacks), атаки по енергоспоживанню (Power Analysis Attacks), атаки за помилками обчислень (Fault Attacks), атаки по електромагнітному випромінюванню (ElectroMagnetic Analysis), атаки за помилками в каналі зв'язку (Error Message Attacks). Існують і більш витончені види атак: атаки з кеш-пам'яті (Cache-based Attacks), акустичні атаки (Acoustic Attacks), атаки по світлового випромінювання (Visible Light Attacks).

Атака за часом

Атака за часом – спосіб отримання якоїсь прихованої інформації шляхом точного вимірювання часу, який потрібен користувачу для виконання криптографічних операцій. Це – найперша з атак по побічних каналах, що з'явилася в цивільній криптографії. Найчастіше час обробки даних в криптосистемах трохи змінюється в залежності від вхідних значень (наприклад, відкритого тексту або шифртекста). Це є наслідком оптимізації продуктивності та широкого кола інших причин. Атака за часом заснована на вимірі часу, необхідного модуля шифрування для виконання операції шифрування. Ця інформація може вести до розкриття інформації про секретному ключі. Наприклад, ретельно вимірюючи час, необхідний для виконання операцій з секретним ключем, атакуючий може знайти точне значення експоненти в алгоритмі Diffie-Hellman.

Атаки за часом наробили багато шуму в пресі в 1995 році: закриті ключі RSA можуть бути відновлені вимірюванням відносних інтервалів часу, витрачених на створення криптографічних операцій. Ці атаки були успішно застосовані до карток із мікропроцесорами та іншими засобам надійної ідентифікації, а також до серверів електронної комерції в Мережі.

Атаки за потужністю

Атака з аналізу потужності придатна переважно для апаратної реалізації криптографічних засобів і успішно застосовується при зломі смарт-карт і інших систем, в яких зберігається секретний ключ.

Щоб виміряти споживану схемою потужність, необхідно послідовно з ланцюгом живлення або заземлення підключити резистор малого опору (наприклад, 50 Ом). Падіння напруги, поділена на опір, дасть силу струму. Сучасні лабораторії розпорядженні устаткування, здатним виконувати цифрові вимірювання напруги на виключно високих частотах понад 1 ГГц) і з чудовою точністю (похибка менше 1%).

Атаки за потужністю може бути розділена на просту (Simple Power Analysis, SPA) і різницеву (Differential Power Analysis, DPA). Метою SPA є отримання інформації про конкретних виконуваних інструкціях в системі і про конкретних

оброблюваних даних. У загальному випадку SPA може дати відомості про роботу пристрою, так і інформацію про ключі. Для здійснення цієї атаки криптоаналитик повинен мати у своєму розпорядженні точні дані про реалізацію пристрою. Цей метод використовує безпосередні дані вимірювань, зібрані під час виконання криптографічних операцій. Згідно, проста атака за потужністю для смарт-карт зазвичай займає декілька секунд, в той час як різницева атака за потужністю може зайняти кілька годин.

На відміну від простих атак, різницеві атаки, засновані на аналізі споживаної потужності, мають на увазі не тільки візуальне подання споживаної потужності, але також статистичний аналіз і статистичні методи виправлення помилок для отримання інформації про ключі. Більш того, DPA часто не потребує даних про конкретної реалізації і в якості альтернативи використовує статистичні методи аналізу. Різницевий аналіз потужності – одне з найпотужніших засобів для проведення атак, що використовують побічні канали, причому ця атака вимагає дуже маленьких витрат.

Атаки за помилками обчислень

Помилки апаратного забезпечення, що з'являються під час роботи відповідного криптографічного модуля, або помилкові вихідні дані можуть стати важливими побічними каналами і іноді істотно збільшують уразливість шифру до криптоаналізу. Криптоаналіз на основі формування випадкових апаратних помилок - це вид нападу на шифри у разі, коли ймовірний порушник має можливість надати на шифратор зовнішній фізичний вплив і викликати одиночні помилки в процесі шифрування одного блоку даних. Атаки за помилками на криптографічні алгоритми вивчаються з 1996 року, і з того часу майже всі криптографічні алгоритми були піддані атакам такого виду.

Здійсненність атаки за помилками (або, принаймні, її ефективність залежить від можливостей зломисника викликати помилки в системі спеціально або користуватися збоями природного походження. Помилки найбільш часто відбуваються через стрибків напруги, збоїв годин або через випромінювань різних типів. Розгляд питання стійкості до цього методу особливо актуально для шифраторів, що застосовуються в інтелектуальних електронних картках. В основному помилки класифікуються по наступних аспектах:

- Точність, яку порушник може досягти при виборі часу і місця, де з'являється помилка під час роботи криптографічного модуля.
- Довжина даних, на які впливає помилка; наприклад, тільки один біт.
- Сталість помилки; чи є помилка короткочасною або постійною.
- Тип помилки; такі як зміна одного біта; зміна одного біта, але тільки в одному напрямку (наприклад, з 1 на 0); зміна біта на випадкове значення та ін.

Загалом, успішна атака за помилками на криптографічні модулі або пристрою вимагає двох кроків: крок створення помилки і крок використання помилки. Помилки можуть бути викликані в смарт-картах шляхом зовнішнього впливу на неї та переміщення в неправильні умови. Деякі з них – аномальне і раптове зниження або підвищення напруги, частоти, температури, випромінювання, освітлення та ін.

Різницевий аналіз помилок полягає у вивченні результату роботи алгоритму шифрування в нормальних і ненормальних умовах при одному і тому ж вході (відкритому тексті). Ненормальні умови зазвичай виходять створенням помилки в процесі (короткочасна помилка) або перед процесом (постійна помилка) роботи. Різницевий аналіз помилок широко вивчені з теоретичної точки зору і здаються застосовними майже до всіх симетричних криптосистем.

Для ГОСТ 28147-89 була показана можливість розкриття ключа і таблиць підстановки з допомогою криптоаналізу на основі формування випадкових апаратних помилок. DES, RC5 і інші шифри також є уразливими по відношенню до цього виду криптоаналізу, тому при їх використанні необхідно забезпечити захист апаратури від нав'язування збоїв.

Атаки по електромагнітному випромінюванню

Виконання обчислювальних операцій на комп'ютері пов'язане з виділенням електромагнітного випромінювання. Вимірюючи і аналізуючи це випромінювання, порушник може отримати значну інформацію про запусчених обчисленнях і використовуваних даних. Атаки по електромагнітному аналізу можуть бути також поділені на дві великі категорії: прості (SEMA) і диференціальні (DEMA).

9 ВИКОРИСТАННЯ НОВИХ ТЕХНОЛОГІЙ В КРИПТОАНАЛІЗІ

На даний момент ці методи не призвели до будь-яких серйозних проривів в криптоаналізі, вони складають більш академічний інтерес, ніж практичний. Однак, ці методи заслуговують на увагу хоча б завдяки їх оригінальності. Крім того, цілком можливо, що з часом, їх важливість у криптології зросте.

9.1. Нейронні мережі

Криптографічну систему можна розглядати як «Чорний ящик», тобто пристрій або програму про внутрішню структуру якої нічого не відомо, але подаючи сигнали на дані входу, ви можете отримати реакцію на виході. Завдання криптоаналізу – ідентифікація системи, тобто, визначення його структури на основі сигналів на його вході і отриманих на виході. Одним з інструментів для вирішення цього завдання може бути теорія нейронної мережі, яка подані у [1].

Штучна нейронна мережа – це математична модель, а також пристрій паралельних обчислень, які є системою підключених і взаємодіючих між собою процесорів (штучних нейронів). Такі процесори зазвичай дуже прості, особливо порівняно з процесорами, які використовуються в персональних комп'ютерах. Кожен процесор подібний до мережі має справу тільки з сигналами, які він отримує періодично, і сигнали, якого він періодично надсилає іншим процесорам. Однак, будучи об'єднаними в досить велику мережу з керованою взаємодією, локально такі прості процесори здатні виконувати досить складні завдання. Концепція виникла при вивченні процесів, що відбуваються в мозку, під час роздумів, і спробі змодельовати ці процеси. Моделі називаються штучних нейронними мережами (ANNs). Проста схема генератора нейронної мережі зображена на рис 1. (чорним позначено елементи входу білим елементи виходу).

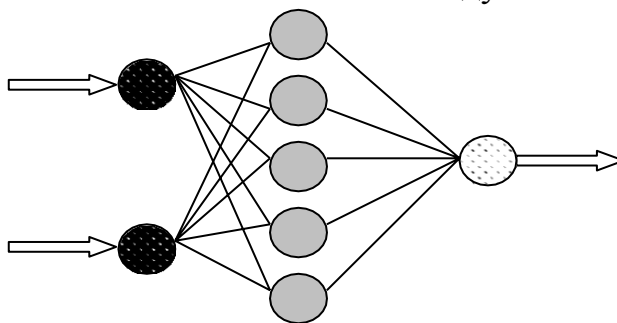


Рисунок 9.1. – Структура нейронної мережі

Брюс Шнайер в його роботі «Прикладна криптографія» [8] став на бік песимізму щодо застосування нейромереж у криптоаналізі: процес злому не залишає місця для навчання: або можна відкрити ключ, або ні (у крайньому випадку це вірно при взломі будь-якого надійного алгоритму). Нейронні мережі добре працюють в структурованих середовищах, дозволяючи для навчання, але не у високоентропійному, імовіровипадковому світі криптографії. Однак, дослідження в цьому напрямку тривають.

У літературі є так звана «атака чорного ящика» для класичних і потокових криптосистеми на основі побудови моделі «нейророзпізнавача чорного ящика». Є дві переслідувані мети: по-перше, визначення відкритого ключа і зашифрованих текстів; по-друге, створення нейродосліджуваної криптосистеми.

Ідентифікація системи представляється у визначенні правил функціонування системи «Чорний ящик» на вході та виході даних і апроксимації невідомої функції шляхом нейромережевої моделі. Першим кроком є виділення нейронної моделі, яка характеризується конкретною архітектурою та алгоритмом навчання. Вибір здійснюється методом проб і помилок. Багато відомих пар відкритих текстів і криптограм поділено на два набори, один з яких використовується для навчання мережі а інший використовується, щоб перевірити на відповідність отриманої моделі вказано точності. Оптимальною моделлю вважається модель, яка має мінімальне число нейронів при цьому задовольняє критеріям.

За допомогою описаної системи здійснився крипто аналіз шифру Віжінера і поточних шифрів. Під час встановлення порогу помилки на рівні 10^{-5} вдалося отримати модель криптосистеми, яка видає результат з 100% точністю. Переваги такого підходу у порівнянні з іншими сучасними методами є незалежність результату від природної мови та її статистичних характеристик, оскільки для навчання система використовує адаптивний навчальний процес.

У літературі [10] наводиться приклад застосування нейронних мереж до злому DES. В процесі навчання використовувалось 2240 пар відкритих текстів та криптограм, що дозволило отримати результати з точністю 98%. У авторів цієї роботи входило у плани також здійснити криптоаналіз AES, використовуючи розроблену стратегію.

9.2. Генетичні алгоритми

Одним з перших шифрів на основі задачі про складання ранцю був запропонований Хеллманом і Марклі в 1978 році. Це була одна з перших спроб створення системи шифрування з відкритим ключем. Незважаючи на те, що проблема складання ранцю належить до класу NP-повних, було показано, що більшість версій алгоритму були нестійкими. У 1983 році Брикель запропонував спосіб взлому криптосистеми на основі ранця низької щільності. Роком пізніше Шамір розробив поліноміальний алгоритм для атаки на початкову ранцеву систему. Після цього, було запропоновано багато інших систем на основі алгоритму ранця: кілька послідовних ранців, ранці Грем Шаміра (Garham-Шамір), і т. д. Для всіх цих систем, були розроблені алгоритми взлому. Є ще один метод криптоаналізу шифру алгоритму укладки ранцю, відмінною особливістю якого є універсальність, тобто можливість використання будь-якої версії "ранцевої криптосистеми", а також зручність в роботі. Метод ґрунтується на використанні генетичних алгоритмів.

Генетичні алгоритми були розроблені Джоном Холландом і являють собою модифікацію так званого «Еволюційного програмування». Ідеєю Холланда було створення алгоритму пошуку через механізми природного відбору, відомого з біології, або алгоритм «спрямованого» випадкового пошуку. На етапі ініціалізації у вас є популяція можливих рішень. На основі цієї популяції з'являється нове покоління рішення, які, у свою чергу, служать «джерелом матеріалу» для наступного покоління.

Цикл генетичного алгоритму загального вигляду представлений на рисунку 9.2: стадії відбору, схрещення та мутації. Найкращі представники покоління вибрано для відтворення популяції, таким чином, що за припущенням кожне нове покоління повинне містити краще рішення, ніж попереднє. У багатьох випадках це вірно.

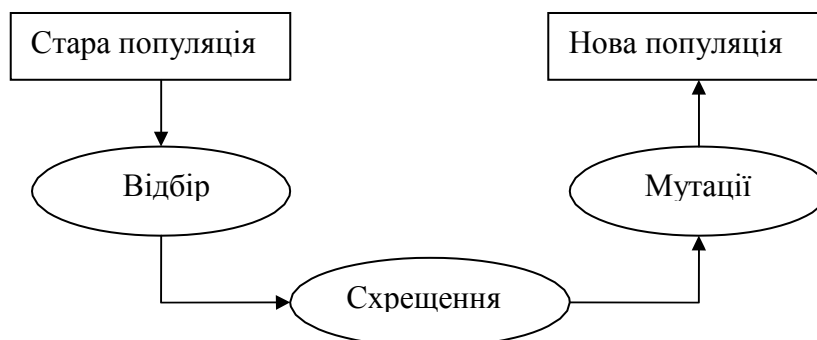


Рисунок 9.2 - Цикл генетичного алгоритму

Популяція складається з набору двійкових рядків. Кожен рядок представляє двійкові рішення та називається хромосомами. Перший етап генетичного алгоритму **Відбір**. У процесі відбору визначається рядок, який використовуватиметься для створення нового покоління. "Батьки" вибрані навмання, але кращі особи популяції мають більше шансів бути вибраними. Таким чином, алгоритм рухається в перспективному напрямку пошуків. Наступний етап схрещення. Схрещення полягає в тому, що для пари виділених рядків довжиною r кожен обирається випадковим чином число $s \in \{1, \dots, r\}$. «Батьки» обмінюються бітами від $s+1$ до r ; таким чином отримуємо хромосоми нащадків.

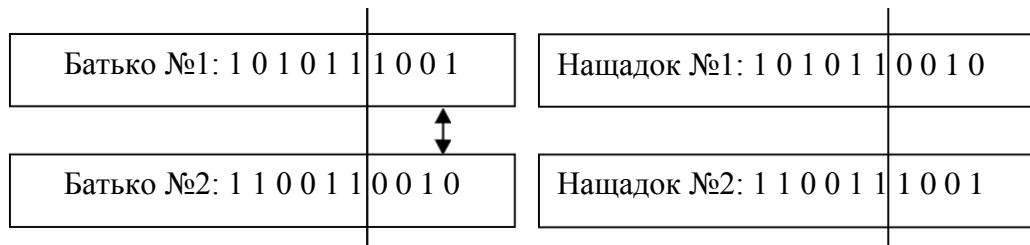


Рисунок 9.3

Заключним етапом є мутація. Під час ініціалізації алгоритму встановлюється фіксована невелика ймовірність мутації новоствореної хромосом (рис. 9.3).

Ці етапи повторюються до досягнення умови виходу з циклу (така умова може бути, наприклад, перевищення максимальної кількості населення).

Розглянемо формулювання задачі укладки ранцю. Дано набір об'єктів різної ваги; питання в тому, чи можна поставити деякі з цих елементів в рюкзак так, що його вага мала деяке значення? Більш формально, завдання сформульовано наступним чином: Дано набір значень M_1, M_2, \dots, M_n і сумарне значення S , необхідно знайти значення b_i такі що: $S = b_1M_1 + b_2M_2 + \dots + b_nM_n$.

Тут b_i може бути нулем або одиницею. Значення $b_i=1$ означає, що i -й предмет кладуть в ранець, а $b_i=0$ - не кладуть. Тому, вміст ранцю представлений у формі хромосом, біти яких відповідають значенням b_i . Функція вибору "кращої хромосоми" оцінює близькість ваги конкретного ранцю до вказаного числа. Значення функції знаходяться в діапазоні $[0; 1]$, де 1 означає точне співпадіння з бажаною вагою. Якщо вага одного ранцю перевищує цільове значення S на деяке число x , а вага другого, навпаки, менша на таку ж кількість x , тому "кращим" є останній ранець. Більш формально, ця функція описується нижче:

1. Обчислити максимальні розбіжності, які можуть виникнути між випадковими хромосомами та цільовими значеннями S : $\Delta_{\max} = \max(S, \tilde{S} - S)$, де \tilde{S} - сума всіх компонентів, які можна використовувати при складанні ранця.

2. Розрахувати вагу ранцю, який відповідає поточній хромосомі і позначити як S' .

3. Якщо $S' \leq S$, то "якість" хромосоми оцінюється значенням: $\alpha = 1 - \sqrt{\left| \frac{S' - S}{S} \right|}$.

4. Якщо $S' > S$, то $\alpha = 1 - \sqrt{\left| \frac{S' - S}{\Delta_{\max}} \right|}$.

Загальний алгоритм має вигляд:

1. Створює випадкова популяція бінарних хромосом.
2. Для кожної хромосоми знаходиться значення α (функція оцінки).
3. На основі отриманих коефіцієнтів відбувається природній відбір.
4. До вибраних на 3-му етапі особ застосовують схрещення.
5. Нащадки піддаються мутації.
6. Нова популяція аналізується, виділяються «кращі хромосоми».

Процес припиняється, коли число поколінь буде перевищувати деяке вказане число; «кращий хромосом» використовуватимуться для злому шифру. Для простого шифру підстановки, який було досліджено, алгоритм показав гарні результати: для отримання оптимальної точки, тобто секретного ключа, алгоритм необхідно дослідити в середньому не більше 2% ключового простору.

Генетичні алгоритми, успішно використовуються у криптоаналізі шифрів підстановок та перестановок.

9.3. Квантові комп'ютери

За допомогою квантового комп'ютера можна здійснювати розрахунки, які не можна реалізувати на сьогоденішніх класичних комп'ютерах. У квантовій фізиці стан частинок характеризується так званою хвильовою функцією ψ , яка

приймає комплексне значення, що називається амплітудою. Аргументом хвильової функції є час, а також набір параметрів (наприклад, координати частинки). У теорії алгоритмів працюють із скінченними об'єктами. Тому із нескінченних моделей перейдемо до скінчених, враховуючи аргументи хвильової функції дискретно.

Для простоти розглянемо випадок, де хвильова функція $\psi(x, t)$ залежить тільки від координатами x і часу t , які ми зафіксуємо. Нехай x може прийняти тільки два значення: 0 і 1. Це відповідає випадку, де частинки можуть бути тільки в двох різних точках. Можемо подивитися на цей випадок, з іншого боку. Припустимо, що у нас є тільки один біт для зберігання інформації про місцезнаходження частинок, що знаходиться у деякому проміжку $[0;1]$. Потім ми приймемо $x=0$, якщо вона в лівій половині цього відрізка, і $x=1$ якщо в правій. Розглянемо дві додаткові функції: $|0\rangle$ і $|1\rangle$. Перша дорівнює 1, якщо $x=0$ і нулю при $x=1$, а друга навпаки. Будь-яку хвильову функцію $\psi(x)$ можна в єдиним способом записати в формі $\lambda_0 |0\rangle + \lambda_1 |1\rangle$. Цей запис відповідає розкладанню вектора двовимірного комплексного простору за базисом $|0\rangle$ і $|1\rangle$ [1].

Якщо вважати ці вектори ортогональними і одиничними, ми отримуємо, що хвильова функція є вектором двовимірного інтегрованого простору з виділеним ортонормованим базисом. Така частинка називається **квантовим бітом**, або **кубітом**. Єдиний спосіб дізнатися, в якій точці знаходиться ця частинка на даний момент часу – це виміряти його хвильову функцію. Вимірювання дасть нам будь-який із векторів $|0\rangle$, $|1\rangle$.

кожен із ймовірністю $|\alpha_0|^2$ або $|\alpha_1|^2$ відповідно. λ_0 і λ_1 - *амплітуда*, пов'язана з умовою нормалізації: $|\alpha_0|^2 + |\alpha_1|^2 = 1$ (сумарна ймовірність дорівнює 1).

Система з n кубітів має простір станів виміру 2^n . Саме цей експоненціальний ріст простору стану в залежності від кількості часток надає перевагу в швидкості обчислень на квантових комп'ютерах в порівнянні з класичними.

У 1994 році Пітер Шор виявив на так званий «обмежено- ймовірнісний» алгоритм факторизації, який дає змогу розкласти на множники число N за поліноміальний час $O(\log N^3)$, витрачаючи при цьому $O(\log N)$ місця на квантовому комп'ютері. У більшості алгоритмів, включаючи алгоритм Шора, використовується стандартний спосіб розкладання на множники, зведення до задачі знаходження періоду функції. Шор використовує квантовий паралелізм для отримання суперпозиції всіх значень функції за один крок. Потім він здійснює квантове перетворення Фур'є, результатом, які для класичного перетворення Фур'є є функція аргумент, якої є кратним величині зворотній доперіоду. З високою вірогідністю вимірювання стану повертає період, який в свою чергу, використовується для розкладання цілого числа N . Вище сказане розкриває суть квантового алгоритму в дуже спрощеній формі. Проблема полягає в тому, що квантове перетворення Фур'є засноване на основі швидкого перетворення Фур'є і таким чином, у більшості випадків дає лише приблизні результати.

Алгоритм Шорра, розкладання чисел на множники був, мабуть, основним досягненням в області квантових обчислень алгоритмів. Це був не тільки великий успіх в математиці. Від цього моменту почали збільшувати фінансування робіт зі створення квантових комп'ютерів.

Ефективність алгоритму Шорра, було поставлено під сумнів японськими вченими з SHARP. Справа в тому, що як Шор так і всі математики, що працюють в галузі квантових алгоритмів, говорять про кількість операцій, хоча і практично важливо тільки час обчислень, який визначили японці. Дискретне перетворення Фур'є (ДПФ) на квантовому комп'ютері використовуються як один за одним перетворення Адамара над окремими кубітами і операції умовного повороту фази в одному кубіті j в залежності від стану іншого для кубіта на кут $\theta = \frac{\pi}{2^{k-j}}$. Якщо число кубітів дорівнює n , то мінімальний кут $\frac{\pi}{2^{n-1}}$. Нехай на цю операцію ми витратимо часу τ_{min} тоді на операцію повороту на кут $\frac{\pi}{2}$ для сусідніх кубітів, ми витрачаємо час порядку $\tau = \tau_{min} 2^n$.

Експоненціальна залежність отримана японськими вченими, зводить на ніщо переваги алгоритму Шора про час розрахунків.

Але співробітник ФТІАН Леонід Федічкін вказав на опубліковану в 1996 році роботу фінських авторів. Вони вивчили вплив шуму на точність ДПФ. Таке перетворення вимагає експоненціально великих (щодо кількості кубітів n) динамічних діапазонів кутів Θ . Що станеться, якщо в свою чергу при малих кутах нахилу забиваються шумом? З'ясувалося, що необхідна точність операцій фазового зсув стані кубіту допускає усунення операції повороту фази на малі кути. Розрахунки Федічкіна показують, що виключення цієї операції зберігає поліноміальну залежність алгоритму Шора від кількості кубітів n .

Так як алгоритм Шора, працює тільки на квантовому комп'ютері, немає в даний час технічних засобів, які дозволяють за поліноміальний час розкласти досить велике число на множники. Таким чином, найбільш важливим питанням залишається створення квантового комп'ютера. Алгоритм Шора, надзвичайно простий і потребує набагато скромнішого апаратного забезпечення, ніж те що потрібне для універсального квантового комп'ютера. Таким чином, цілком імовірно, що квантовий пристрій до розкладів, буде побудовано задовго до того, як весь спектр квантових обчислень стане технологічно можливим. На сьогоднішній день є конкретні результати. Так IBM продемонструвала використання створеного в лабораторії компанії семикубітного квантового комп'ютера для факторизації чисел по алгоритму Шора. Хоча вирішена ним задача навряд може здивувати уяву (комп'ютер правильно визначив, що дільники числа 15 є числа 5 і 3), це є найскладнішим розрахунком за всю історію квантових комп'ютерів.

10 КРИПТОГРАФІЧНІ ПРОТОКОЛИ

10.1. Специфіка взаємодії віддалених абонентів

Успіхи, досягнуті в розробці схем цифрового підпису та відкритого розподілу ключів, дозволили застосувати ці ідеї також і до інших місій взаємодії віддалених абонентів. Так виник великий новий напрямок теоретичної криптографії – криптографічні протоколи.

У класичній Шеннонівській моделі системи секретного зв'язку є два (повністю довіряючі один одному учасники), яким необхідно передавати між собою інформацію, яка не призначена для третіх осіб. Вирішується завдання захисту секретної інформації від зовнішнього противника.

Об'єктом вивчення теорії криптографічних протоколів являються віддалені абоненти, взаємодіючі, як правило, по відкритих каналах зв'язку. Мета взаємодії абонентів полягає в розв'язанні конкретного секретною завдання. У цій ситуації противником може виявитися будь-який з абонентів або декілька абонентів, що вступили в змову, що переслідують власні цілі. При цьому противник в різних задачах може мати різні можливості: наприклад, може взаємодіяти з абонентами від імені інших абонентів або втручатися в обміни інформацією між абонентами і т. д. Тому криптографічні протоколи повинні захищати їх учасників не тільки від зовнішнього противника, а й від нечесних дій партнерів.

Є відмінності криптографічних протоколів від криптосистем, з яких можна виділити наступні:

- учасники протоколу, взагалі кажучи, не довіряють один одному;
- в протоколі може бути більше двох учасників;
- протоколи можуть бути інтерактивними, тобто є багатораундовий обмін повідомленнями між учасниками.

На жаль, поняття криптографічного протоколу неможливо формалізувати.

Визначення. Під протоколом (не обов'язково криптографічним) зазвичай розуміють розподілений алгоритм, що включає в себе:

- сукупність алгоритмів для кожного з учасників,
- специфікації форматів повідомлень, що пересилаються між учасниками,
- специфікації синхронізації дій учасників,
- опис дій при виникненні збоїв.

На останній елемент цього списку слід звернути особливу увагу, оскільки його часто випускають із уваги, а некоректний повтор може повністю розвалити безпеку учасників навіть у стійкому криптографічному протоколі.

Всі типи криптографічних протоколів можна умовно розділити на дві групи: **прикладні протоколи і примітивні** [6].

Прикладний протокол вирішує конкретну задачу, яка виникає (або може виникнути) на практиці.

Примітивні протоколи використовуються як своєрідні «будівельні блоки» при розробці прикладних протоколів.

За останнє десятиріччя криптографічні протоколи перетворились в основний об'єкт досліджень в теоретичній криптографії. Одна з основних областей застосувань криптографічних протоколів – банківські платіжні системи, де замість платіжних доручень на папері використовується їх електронна форма. Вигоди від такої заміни настільки відчутні, що, мабуть, банки від неї вже ніколи не відмовляться, які б технічні та криптографічні труднощі при цьому ні виникали. Але платіжні доручення – лише один з численних типів документів, що знаходяться в обороті в сфері бізнесу. Адже існують ще документи, з якими працюють державні органи і громадські організації, юридичні документи і т. д. В останні роки чітко простежується тенденція перекладу всього документообігу в електронну форму.

У дослідженнях багатьох типів криптографічних протоколів зроблені тільки перші кроки і ще багато математичних проблем належить вирішити, перш ніж криптографічні протоколи ввійдуть до повсюдного використання.

10.2. Приклади криптографічних протоколів.

Познайомимось з деякими типами криптографічних протоколів, а також колом математичних завдань, що виникають при дослідженні їх стійкості. Розглянемо кілька прикладів завдань, що вирішуються віддаленими абонентами.

1. Протокол підписання контракту. Взаємодіють два абоненти, які не довіряють один одному. Вони хочуть підписати контракт. Це треба зробити так, щоб не допустити таку ситуацію: один з абонентів отримав підпис іншого, а сам не підписався.

2. Протокол ідентифікації абонента. Взаємодіють два абонента А і В. Абонент А хоче довести абоненту В, що він саме А, а не противник. Типовий приклад: А - власник інтелектуальної смарт -картки (кредитної карти, карти доступу в закриті приміщення, комп'ютерний ключ), В - банк, комп'ютер охорони, адміністратор мережі.

3. Протокол візантійської угоди. Взаємодіють кілька віддалених абонентів, які отримали накази з одного центру. Частина абонентів, включаючи центр, можуть бути противниками. Необхідно виробити єдину стратегію дій, виграшну для абонентів. Це завдання прийнято називати задачею про **візантійських генералів** [6].

Опишемо приклад, якому це завдання зобов'язане своєю назвою. Ніч перед великою битвою. Візантійська армія складається з n легіонів, кожен з яких підпорядковується своєму генералу. Крім того, у армії є головнокомандувач, який керує генералами. Однак імперія знаходиться в занепаді і одна третина генералів, включаючи головнокомандуючого, можуть бути зрадниками. Протягом ночі кожен з генералів отримує від головнокомандувача наказ про дії на ранок, причому можливі два варіанти наказу: «атакувати» або «відступати». Якщо всі чесні генерали атакують, то вони перемагають. Якщо всі вони відступають, то їм вдається зберігати армію. Але якщо частина з них атакує, а частина відступає, то вони зазнають поразки. Якщо головнокомандувач виявиться зрадником, то він може дати різним генералам різні накази, тому накази головнокомандувача не

варто виконувати беззаперечно. Якщо кожен генерал буде діяти незалежно від інших, результати можуть виявитися плачевними. Очевидно, що генерали потребують обміну інформації один з одним (щодо отриманих наказів) з тим, щоб прийти до угоди.

4. Протокол вгадування підкинутої монети по телефону. Взаємодіють два, які не довіряють один одному, абоненти. Вони хочуть кинути жереб за допомогою монети. Це треба зробити так, щоб абонент, який підкидає монету, не міг змінити результат підкидання після отримання здогадки від абонента, який вгадує цей результат.

Опишемо один з найпростіших протоколів підкидання монети по телефону (так звана схема Блюма - Мікалі). Для його реалізації у абонентів А і В має бути одностороння функція $y=f(x)$, яка задовольняє таким умовам:

1) X – скінчена множина цілих чисел, яка містить однакову кількість парних і непарних чисел;

2) будь-які числа x_1, x_2 із X , що мають один образ $f(x_1)=f(x_2)$, мають одну парність;

3) по заданому образу $f(x)$ «важко» обчислити парність невідомого аргументу x .

Роль підкидання монети грає випадковий і рівноймовірний вибір елемента x із X , а роль орла і решки - парність і непарність x відповідно. Нехай А - абонент, підкидає монету, а В - абонент, що вгадує результат. Протокол складається з наступних кроків:

1) А вибирає x («підкидає монету»), зашифровує x , тобто обчислює $y=f(x)$, і посилає у абоненту В;

2) В отримує y , намагається вгадати парність x і посилає свою здогадку абоненту А;

3) А отримує здогад від В і повідомляє В, вгадав він, посилаючи йому вибране число x ;

4) В перевіряє, чи не обманює А, обчислюючи значення $f(x)$ і порівнюючи його з отриманим на другому кроці значенням y .

10.3. Інтерактивна система доведення

Осмислення різних протоколів і методів їх побудови привело в 1985-1986 рр. до появи двох плідних математичних моделей - **інтерактивної системи доказів і доведень із нульовим розголошенням**. Математичні дослідження цих нових об'єктів дозволили довести багато тверджень, вельми корисних при розробці криптографічних протоколів.

Визначення. Під інтерактивною системою доказів (P, V, S) розуміють протокол взаємодії двох абонентів:

P (prover) - доводить і V (verifier) - перевіряє.

Абонент P хоче довести абоненту V , що твердження S істинно. При цьому абонент V самостійно, без допомоги абонента P , не може перевірити твердження S (тому V і називається перевіряючим). Абонент P може бути і противником, який хоче довести абоненту V , що твердження S істинно, хоча насправді воно

помилково. Протокол може складатися з багатьох раундів обміну повідомленнями між абонентами P і V і повинен задовольняти двом умовам:

1) повнота - якщо S дійсно істинно, то абонент P переконає абонента V визнати це;

2) коректність - якщо S неправдиво, абонент P «навіть чи» переконає абонента V , що S істинно.

Тут словами «навіть чи» ми для простоти замінили точне математичне формулювання.

Підкреслимо, що у визначенні системи (P, V, S) не допускалось, що абонент V може бути противником. А якщо це так і перевіряючий хоче «вивідати» у доводячого якусь нову корисну для себе інформацію про твердження S ? У цьому випадку абонент P , природно, може не хотіти, щоб це трапилося в результаті роботи даного протоколу. Протокол (P, V, S) , який вирішує таке завдання, називається **доведенням з нульовим розголошенням** і повинен задовольняти, крім умов 1) і 2), ще й такий умові:

3) нульове розголошення - в результаті роботи протоколу (P, V, S) абонент V не збільшить свої знання про твердження S або, іншими словами, не зможе витягти жодної інформації про те, чому S істинно.

10.4. Протоколи аутентифікації

Одне із застосувань інтерактивних систем доказів - протоколи аутентифікації, коли один з абонентів повинен довести другому свій статус, але при цьому зробити так, щоб ніхто інший, який спостерігає за цим протоколом не зміг зробити те ж саме. Такий, один з найбільш важливих і поширених типів криптографічних протоколів називають схемами аутентифікації.

Прикладом протоколу аутентифікації може служити всім відома казка про вовка і семеро козенят. У цьому протоколі коза виступає в якості того, хто доводить, а семеро козенят - в якості перевіряючого. Протокол покликаний забезпечувати цілісність. У даному випадку - сімох козенят. У казці описана атака на протокол. Противником виступає вовк, і цей противник активний: спочатку він підслуховує виконання протоколу, потім намагається сам пройти аутентифікацію в якості доводячої (кози) і при цьому накопичує й аналізує отримувану інформацію. Протокол виявився нестійким проти активного противника.

Більш серйозно призначення і суть протоколів аутентифікації (або ідентифікації) легко зрозуміти на наступному прикладі. Уявімо собі інформаційну систему, яка працює в комп'ютерній мережі і забезпечує доступ до деяких даних. У адміністратора системи є список всіх її користувачів разом з співставленим кожному з них набором повноважень, на основі яких здійснюється розмежування доступу до ресурсів системи. Ресурсами можуть бути, наприклад, деякі фрагменти інформації, а також функції виконувані системою. Одним користувачам може бути дозволено читати одну частину інформації, іншим - іншу її частину, а третім - ще й вносити до неї зміни. У даному контексті під забезпеченням цілісності розуміється попередження доступу до системи осіб, які не є її користувачами, а також попередження доступу користувачів до тих

ресурсів, на які у них немає повноважень. Найбільш поширений метод розмежування доступу - парольний захист не забезпечує достатньої стійкості. Тому перейдемо до криптографічної постановки завдання.

Для того, щоб протокол аутентифікації був стійким, досить, щоб він був доведенням з нульовим розголошенням. У протоколі є два учасника - Аліса, яка повинна довести свою автентичність, і Боб, який цю автентичність повинен перевірити. В Аліси є два ключі - загальнодоступний відкритий K_v і секретний K_s . Фактично Алісі потрібно довести, що вона знає - K_s , і зробити це таким чином, щоб цей доказ можна було перевірити, знаючи тільки K_v .

Схема аутентифікації Шнорра

Одним з найбільш ефективних практичних протоколів аутентифікації вважають протокол Шнорра.

Нехай p і q - прості числа такі, що q ділить $p-1$ (Шнорр пропонує використовувати p довжини порядку 512 біт і q - довжини близько 140 біт).

$g \in \mathbb{Z}_p$ таке, що $g^q = 1 \pmod p$, $g \neq 1$, де $\mathbb{Z}_p = \{0, 1, \dots, p-1\}$ - множина залишків за модулем p .

$x \in \mathbb{Z}_q$ - секретний ключ, де $\mathbb{Z}_q = \{0, 1, \dots, q-1\}$ - множина залишків за модулем q . $y = g^x \pmod p$ - відкритий ключ.

Визначення x по y - це завдання дискретного логарифмування, для якої на даний момент не відомі ефективні алгоритми.

Відкриті ключі всіх учасників схеми повинні публікуватися таким чином, щоб виключалася можливість їх підміни (таке збереження ключів називається загальнодоступним сертифікованим довідником). Ця проблема, яку часто називають проблемою автентичності відкритих ключів, складає окремий предмет досліджень у криптографії.

У протоколі Шнорра кількість раундів дорівнює трьом. Наведемо алгоритм протоколу по кроках:

1. В якості секретного ключа схеми аутентифікації Аліса вибирає випадкове число k із множини $\{1, \dots, q-1\}$, обчислює відкритий ключ $r = g^k \pmod p$ і посилає r Бобу.

2. Боб вибирає випадковий запит e з множини $\{0, \dots, 2^t-1\}$, де t -деякий параметр, і посилає e Алісі.

3. Аліса обчислює $s = k + xe \pmod q$ і посилає s Бобу.

4. Боб перевіряє співвідношення $r = g^s y^e \pmod p$ і, якщо воно виконується, приймає доказ, інакше - відкидає.

Перша з вимог до стійкості протоколів аутентифікації - коректність, означає, що противник, що знає тільки відкритий ключ y , може пройти аутентифікацію лише з нескінченно малою ймовірністю.

Нескладний аналіз показує, що коректність протоколу Шнорра залежить від вибраного значення параметра t .

Ця неформально викладена ідея була використана Шнорром для доказу поліноміальної зведеності завдання дискретного логарифмування до задачі, що стоїть перед пасивним противником, тобто таким, який намагається пройти аутентифікацію, знаючи лише відкритий ключ. Іншими словами, доведено, що в

припущенні труднощі завдання дискретного логарифмування схема аутентифікації Шнорра є стійкою проти пасивного противника, тобто коректною.

Активний противник може провести деяку кількість сеансів виконання протоколу в якості перевіряючого з чесним доводячим (або підслухати такі виконання) і після цього спробувати атакувати схему аутентифікації. Для стійкості проти активного противника достатньо, щоб протокол аутентифікації був доказом з нульовим розголошенням. Однак властивість нульового розголошення для схеми Шнорра досі нікому довести не вдалося. Більше того, на даний момент відомий єдиний метод доказу властивості нульового розголошення - так званий метод «чорного ящика». У цьому методі моделююча машина використовує алгоритм перевіряючого лише в якості оракула, тобто, не аналізуючи сам цей алгоритм, подає йому на вхід будь-які значення за своїм вибором і отримує відповідні вихідні значення.

Доведено, що трираундові доведення з нульовим розголошенням, в яких остання властивість усталюється методом «чорного ящика», існують лише в тривіальному випадку, тобто коли перевіряючий може самостійно, без всякої допомоги доводячого перевірити істинність затверджується. Відносно схеми Шнорра з цього результату впливає, що або існує ефективний алгоритм дискретного логарифмування, або властивість нульового розголошення цього протоколу не може бути доведено методом «чорного ящика». Питання про існування доказів з нульовим розголошенням, для яких властивість нульового розголошення не може бути доведено методом «чорного ящика», залишається відкритим.

Неважко показати, що схема Шнорра володіє більш слабкою властивістю - властивістю нульового розголошення щодо чесного перевіряючого. У цьому випадку досить побудувати моделюючу машину тільки для чесного перевіряючого, який на кроці 2 і справді вибирає випадковий запит e з множини $\{0, \dots, 2^t - 1\}$.

Властивості нульового розголошення щодо чесного перевіряючого може виявитися досить, якщо схема аутентифікації використовується, наприклад, для контролю за доступом в приміщення, що охороняється. У цьому випадку Аліса - це перепустка, виконана у вигляді інтелектуальної картки, а Боб - комп'ютер охорони. У такій ситуації головне завдання - забезпечити коректність схеми аутентифікації, а захищатися від нечесного перевіряючого безглуздо. Що ж до властивості нульового розголошення щодо чесного перевіряючого, то воно видається далеко не зайвим, оскільки дозволяє убезпечитися від противника, який може спробувати підслуховувати сеанси виконання протоколу з метою виготовлення фальшивого пропуску.

Ніяка математично строго доведена властивість криптографічного протоколу не може гарантувати його безпеки у всіх випадках. Справді, навіть протокол доказів з нульовим розголошенням не захищає від наступної атаки на схему аутентифікації, відомої в криптографічній літературі під назвою «мафіозна загроза». У даному сценарії є чотири учасники: А, В, С і D. Припустимо, що А зайшов у кафе випити чашечку кави і розплачується за допомогою кредитної картки. Для виконання будь-якої банківської операції картка повинна

ідентифікувати себе, тобто виконати протокол аутентифікації. Але власник кафе, В, - мафіозі, спільник якого С в той же самий момент знаходиться в магазині ювеліра D і намагається купити діамант, також за допомогою кредитної картки. При цьому С «представляється» як А, а D просить довести це за допомогою протоколу аутентифікації. Пристрій зчитування для карток у В і картка у С - це спеціально виготовлені приймально-передавальні пристрої, які лише пересилають повідомлення між А і D. У результаті А, обмінюючись повідомленнями з В, насправді ідентифікує себе для D. Отже, тільки математичного обґрунтування стійкості того чи іншого криптографічного протоколу недостатньо. Для практичного застосування конкретного протоколу потрібні ще зусилля спеціалістів в області інформаційної безпеки з аналізу умов його застосування.

Діалогові доведення з нульовим розголошенням

Приклади таких доведень:

- Доведення знань ізоморфізма графів.
- Доведення знань розкладу складного числа на множники (наприклад, Фіама-Шаміра).
- Доведення знань дискретного логарифма.
- Доведення правильності вибору складного числа.

Доведення знання розв'язку певної задачі Діффі-Хеллмана з нульовим розголошенням

Припустимо, що G_1 – це скінченна адитивна підгрупа точок еліптичної кривої E з N елементів, P – базова точка цієї кривої, aP та bP елементи G_1 , за якими гравець А розв'язав задачу Діффі-Хеллмана, тобто знайшов точку кривої abP . Гравцю А потрібно довести перевіряльнику В, що він справді знає розв'язок цієї задачі, не допомагаючи йому знайти це значення. При цьому припустимо, що перевіряльнику В відомий порядок q групи G_1 . Послідовність кроків, які повинні зробити гравці, така:

- 1) Гравець А випадково вибирає число e та надсилає В значення $V'=ebP$.
- 2) В вибирає випадковий біт α . Якщо $\alpha=1$, то гравець А повинен розкрити значення e і гравець В може перевірити, що дійсно $V'=ebP$.
- 3) Якщо $\alpha=0$, то гравець А повинен обчислити та передати В значення $eabP$, гравець В повинен перевірити рівність $e(P, eabP) = e(aP, V')$.
- 4) Кроки 1–3 повторюються доти, доки гравець В не переконається, що А дійсно знає значення abP .

Властивість повноти тривіальним чином впливає з конструкції самого протоколу, оскільки виникає тотожність $e(P, eabP) = e(P, P)^{eab} = e(aP, P)^{eb} = e(aP, V')$.

Якщо гравець А насправді не знає значення abP , то він може давати правильну відповідь не більше ніж при одному варіанті значення α . Так, якщо виконуючи крок 1 гравець А сподівається, що α прийме значення 1, то він може, не знаючи abP , послати В значення $V' = ebP$. Якщо А сподівається, що α прийме значення 0, то він може послати значення $V' = eP$ (тоді на кроці 3 він повинен замість $eabP$ послати значення eaP , тому що $e(P, eaP) = e(aP, eP) = e(aP, V')$). Однак якщо в цьому випадку α прийме значення 1, гравець А не зможе надати правильну відповідь, бо не знає значення eb^{-1} , яке потрібно скалярно помножити на точку bP , щоб отримати V' ($V' = eP = eb^{-1}bP$). Ймовірність того, що гравець В прийме

доведення дорівнює $1/2^m$ при m ітераціях протоколу, це обґрунтовує властивість коректності. Необхідно зауважити, що при доведенні стійкості криптографічних протоколів будують зв'язок певної важко обчислювальної задачі до задачі, яка стоїть перед супротивником. Тобто якщо супротивник зможе зламувати протокол, то алгоритм, який використовує супротивник, можна використовувати для обчислювання певної важко обчислювальної задачі. Так зробив, наприклад, Шнор зі схемою аутентифікації, де здійснюється зведення задачі обчислення дискретного логарифму до задачі, яку повинен розв'язувати противник. У нашому випадку противник, не знаючи значення abP , повинен уміти обчислювати значення $eabP$. Якщо припустити, що він це вміє якось робити, то тоді він зможе розв'язувати задачу Діффі–Хеллмана, тобто обчислювати abP множенням e^{-1} на $eabP$. Властивість нульового розголошення можна обґрунтувати використовуючи відомий метод імітації, який використовується в доведеннях наявності цієї властивості в протоколах такого типу. Нехай деяка третя сторона C не знає значення розв'язку задачі Діффі–Хеллмана abP при відомих P , aP та bP , однак знає заздалегідь, яке значення прийме випадковий біт α . Тоді вона може імітувати гравця A , посылаючи $B' = ebP$ перед тим, як α прийме значення 1 і $B' = eP$ перед тим, як α прийме значення 0. Та інформація, яку гравець B отримуватиме, не відрізнятиметься від тієї інформації, яку він міг би отримувати від гравця A . При цьому третя сторона C не може передати ніякої інформації щодо розв'язку задачі, тому що сама не знає його.

Доведення знання розкладу складного числа на множники (Фіама-Шаміра)

Цей протокол заснований на складності задачі добування квадратного кореня за модулем великого складного числа n із невідомим розкладом на множники.

A доводить B знання секрету за допомогою t ітерацій наступного три крокового протоколу. Довірений центр T вибирає модуль n та повідомляє його всім тим, хто доводить, при цьому множники p і q залишаються секретними.

Кожний A вибирає секрет $1 < s < n-1$, який є взаємно простим із n . Далі обчислює значення $v = s^2 \bmod n$, яке називає своїм відкритим ключем.

Наступні три кроки повторюються незалежно t раз, причому B приймає доведення A про знання секрету, якщо всі ітерації призводять до позитивної відповіді.

1. A вибирає z , $1 < z < n-1$ і відправляє B число $x = z^2 \bmod n$.
2. B випадково вибирає біт c і відправляє його A .
3. A обчислює і відправляє для B число ($y = z$, якщо $c = 0$) або ($y = zs \bmod n$, $c = 1$).

B дає позитивну відповідь, якщо $y \neq 0$ і $y^2 = xv^c \bmod n$.

Бездіалогові доведення з нульовим розголошенням

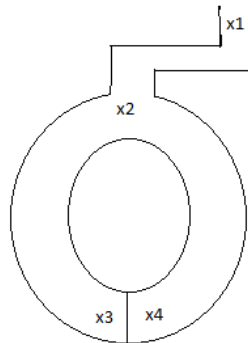
Апарат доведень із нульовим розголошенням дозволяє забезпечувати доведення знань секретної інформації і без діалогу. Нехай претендент знає доведення деякої теореми і хоче переконати верифікатора в тому, що теорема доведена, не надаючи йому доведення. Можливі два випадки:

1. Доведення може бути представлене лише одному верифікатору, і тільки він може перевірити правильність доведення.

2. Доведення може бути перевірене кожним, в такому випадку говорять про відкритих перевіряючих без діалогових доведень (як в ЦЕП).

Прикладом такого доведення є схема ідентифікації Гілли-Кіскатера.

Жан-Жак Кіскатер та Луї Гілли пояснюють нульове розголошення історією у пищері. У пищері є секрет – таємні двері. Той хто знає чарівні слова може відкрити таємні двері між x_3 та x_4 . Для всіх інших обива проходи ведуть до глухого кута.



А знає секрет і хоче довести це В, але не розкриваючи секрет.

- В знаходиться у точці x_1 .
- А проходить весь шлях по печері або до точки x_3 , або до x_4 .
- Після того як А зайшов в печеру, В переходить у точку x_2 .
- В кричить до А, просячи його, щоб А вийшов:
 - з лівого проходу;
 - з правого проходу.
- А виконує бажання В, при необхідності використовує таємні слова, щоб відкрити двері.
- А і В повторюють це від 1 до n разів.

Зауваження

В не може переконати третю сторону в правильності доведення.

10.5 Захищені обчислення

До захищених обчислень відносяться різні протоколи, метою яких є спільне обчислення деякого значення таким чином, щоб приховати це значення від усіх або деяких учасників. Розглянемо два протоколи захищених обчислень.

Приховування інформації від оракула

Нехай учасник В - оракул, який вміє обчислювати значення деякої односторонньої (важкообчислюваної) функції f . А хоче звернутися до В за обчисленням $f(x)$, але не хоче розкривати значення x .

Параметри протоколу: конструкція (K, E, D) , що відповідає вимогам до криптосистеми з секретним ключем і з функцією дешифрування, що задовольняє умові $D_k(f(E_k(x)))=f(x)$.

Алгоритм протоколу:

1. А вибирає випадковим чином ключ k , шифрує x на цьому ключі: $E_k(x)$, і посилає його В.

2. В обчислює значення $f(E_k(x))$ функції від зашифрованого значення і відсилає його А.

3. А дешифрує отримане значення і отримує за умовою $D_k(f(E_k(x)))=f(x)$ значення функції.

Криптосистема з відкритим ключем тут не потрібна, тому що не потрібно пересилати оракула ні значення ключа, ні алгоритм дешифрування. Основна умова - наявність функцій шифрування і дешифрування, що задовольняють умові $D_k(f(E_k(x)))=f(x)$.

Прикладом такого протоколу може бути протокол обчислення функції $f(x)$ - дискретного логарифма x в Z_p^* за основою g , (g -утворюючий елемент Z_p^*). У цьому випадку функції шифрування і дешифрування можна вибрати:

$$y=E_k(x)=x g^k \bmod p \text{ і } D_k(y)=(y-k) \bmod (p-1), k \in Z_{p-1}^*.$$

Задача про двох мільйонерів .

Два мільйонера хочуть з'ясувати у кого більше мільйонів, але так, щоб ніхто не дізнався, скільки їх у іншого.

Параметри протоколу :

Нехай a, b - секретні значення учасників А і В.

У протоколі потрібна криптосистема (K, E, D) з відкритим ключем і число m , таке, що $a \leq m$ і $b \leq m$. Нехай k - ключ учасника В.

Алгоритм протоколу :

1. А вибирає велике просте число x , обчислює $t=E_k(x)$ - a і посилає його В.

2. В вибирає просте число p , яке задовольняє вказаній нижче умові і обчислює послідовність чисел $S=(u_1, \dots, u_m, p)$ наступним чином:

$$u_i = \begin{cases} z_i, & i \leq b \\ z_i = 1, & i > b \end{cases}, \text{ де } z_i = D_k(t+i) \bmod p,$$

p вибирається так, щоб для всіх $i \neq j$ виконувалося $|z_i - z_j| > 1$.

Зауважимо, що при цьому $u_a = x$.

3. А посилає В повідомлення « $a \leq b$ », якщо $u_a = x \bmod p$ і « $a > b$ » в іншому випадку.

Цей протокол має очевидні недоліки:

– Чим більше m , тим складніше підібрати p , яке задовольняє умовам.

– Учасник А дізнається результат раніше, ніж В і може відмовитися від виконання 3-го кроку.

10.6 Спеціальні види електронного підпису.

Для вирішення завдань аутентифікації і забезпечення цілісності інформації використовується цифровий підпис. Звичайні протоколи електронного підпису, що розробляються для практичного застосування, є неінтерактивними (тобто весь обмін повідомленнями полягає в передачі відправником одержувачу підписаного повідомлення). Це викликано насамперед вимогами до ефективності.

Проте виникають ситуації в яких до електронного підпису пред'являються додаткові вимоги. Розглянемо спеціальні види електронного підпису, в яких використовуються багаторазові протоколи.

Сліпий підпис

Сеанс протоколу сліпого підпису полягає в тому, щоб учасник В підписав запропоноване учасником А повідомлення, не отримавши інформації про це повідомлення. У паперовому документообігу така схема може бути реалізована за допомогою запечатаного конверта, в якому знаходиться документ, а поверх нього лист копії. Такий підпис використовується, наприклад, у фінансовій криптографії для забезпечення властивості «невідстежуваності» електронних грошей.

У цифровому варіанті сліпий підпис являє собою окремий випадок приховування інформації від оракула. Повідомлення m шляхом шифрування упаковується в цифровий конверт $P(m)$, який підписується $S(P(m))$. При розкритті конверта виходить підпис для m : $P^{-1}(S(P(m)))$.

Наведемо реалізацію сліпого підпису RSA.

Протокол створення сліпого підпису RSA під повідомленням $m \in Z_m$

Ключі учасника В: (n, e) - відкритий і (n, d) - секретний .

1. $A \rightarrow B \quad P(m) = mr^e \bmod n, r \in Z_n^*$

2. $B \rightarrow A \quad S(P(m)) = (P(m))^d \bmod n$

А обчислює $P^{-1}(S(P(m))) = r^{-1}S(P(m)) = m^d \bmod n$

Як правило, необхідно, щоб В підписував лише повідомлення з деякої множини «прийнятних» повідомлень M , але не знав, яке саме повідомлення він підписав.

Протокол створення сліпого підпису RSA під прийнятним повідомленням $m \in M$.

Ключі учасника В: (n, e) - відкритий і (n, d) - секретний.

Нехай $m_1, \dots, m_k \in M$ - повідомлення, для одного з яких буде отриманий цифровий підпис.

1. $A \rightarrow B \quad (P_1(m_1), \dots, P_k(m_k))$, де $P_i(m_i) = m_i^e \bmod n, r_i \in Z_n^* \quad i = 1, \dots, k$.

2. $B \rightarrow A$ вибирає номер конверта $j \in \{1, \dots, k\}$, який підпише.

3. $A \rightarrow B$ для будь-якого $i \neq j$, А відкриває всі конверти, крім j -го, В перевіряє, що $i \neq j$ повідомлення прийнятні $m_i \in M$.

4. $B \rightarrow A \quad S(P_j(m_j)) = P(m_j)^d \bmod n$.

А розкриває підписаний конверт, тобто обчислює підпис для m :

$P^{-1}(S(P(m_j))) = r^{-1}S(P(m_j)) = m_j^d \bmod n$.

Імовірність того, що А вдасться отримати підпис під неприйнятним повідомленням, не перебільшує k^{-1} .

Невидимий підпис.

Можна зробити так, щоб підпис доводячого учасника Р (prover) не міг би бути перевірений перевіряючим V (verifier) без участі Р. При цьому вийде так званий невидимий підпис або незаперечний підпис (undeniable signature, термін ввів David Chaum, вперше запровадив таку схему).

Застосування невидимого підпису:

– надання фірмою можливості перевірити справжність розповсюджуваного програмного продукту лише зареєстрованим користувачам;

– здійснення безпаперового документообігу всередині організації так, щоб у разі витоку документу за межі організації не можна було довести його автентичність.

Параметри системи :

Група Z_p , де p - просте число. Нехай $g \in Z_p$ - утворюючий елемент (ділить $p-1$). Ключі учасника P : $x \in Z_p$ - закритий ключ, $y = g^x$ - відкритий ключ.

Створення підпису :

$z = S(h) = h^x \bmod p$, де $h \in Z_p$ - хеш-функція повідомлення.

1. $P \rightarrow V$ число $w = h^a g^b \bmod p$, де $a, b \in Z_p$ - вибрані P числа.

2. $V \rightarrow P$ пару чисел (r, s) , де $r = w g^t = h^a g^{b+t} \bmod p$, де $t \in Z_p$ - вибране V число і $s = r^x \bmod p$.

3. $P \rightarrow V$ пару чисел (a, b) і P перевіряє, що $w = h^a g^b \bmod p$.

4. $V \rightarrow P$ число t і V перевіряє, що $r = h^a g^{b+t}$ і $s = z^a y^{b+t} \bmod p$.

Твердження (надійність протоколу). Навіть володіючи необмежено обчислювальною потужністю, P не може з вірогідністю, більшою p , дати вірні відповіді при неправильному підписі.

Твердження (безпека протоколу). У ході протоколу не відбувається розкриття інформації про закритий ключ P , оскільки V може самостійно побудувати всі повідомлення протоколу з тим же розподілом, але без участі P .

11 КРИПТОГРАФІЧНІ СХЕМИ ПОДІЛУ СЕКРЕТУ

11.1 Поділ секрету

Криптографічні схеми поділу секрету були незалежно відкриті Шаміром (A. Shamir) і Блеклі (GR Blakley). Основне призначення протоколів або схем поділу секрету - управління ключами. У багатьох практичних додатках доступність інформації залежить від одного єдиного секретного ключа. Якщо ключ яким-небудь чином загублений (втрачений носій із записаним ключем, зруйнована пам'ять, в якій зберігається ключ, і т.д.), доступ до інформації блокується. Схеми розділу секрету дозволяють вирішити сформульовану проблему. Ідея полягає в поділі секретного ключа на компоненти з наступним їх розподілом серед легального кола учасників. Відновлення ключа можливо тільки в тому випадку, якщо деяка легальна коаліція учасників об'єднає свої ключові компоненти [8].

З точки зору математики завдання поділу секрету є насамперед комбінаторною задачею. Відзначимо, що перша постановка задачі була сформульована у вигляді такої комбінаторної проблеми. Розглянемо ситуацію, в якій одинадцять вчених працюють над секретним проектом. Робочі документи проекту замкнені в сейфі. Умова задачі - тільки шість і більше вчених, об'єднавшись і пред'явивши ключі, можуть відімкнути сейф і отримати доступ до документів.

Постановка завдання зводиться до оцінки найменшого числа замків сейфа і найменшого числа ключів у кожного з учених. В якості розв'язання пропонується наступне міркування. За умовою існує як мінімум один замок, який не може бути відкритий жодним ученим з довільної п'ятірки. Таким чином, кожен з шести інших вчених має ключ від цього замка. Причому немає необхідності в більш ніж одному подібному замку для коаліції з п'яти вчених. Таким чином, мінімальне число замків і ключів оцінюється як число сполучень $C_{11}^5 = 462$ і $C_{11-1}^5 = 252$ відповідно. Очевидно, що пропоноване комбінаторне вирішення є неприйнятним з практичної точки зору.

Криптографічні схеми поділу секрету пропонують інше вирішення проблеми. Розглянемо її як **груповий криптографічний протокол** з деякою множиною учасників.

Зазвичай в протоколах ми розглядали пару взаємодіючих учасників (A, B). У групових протоколах замість A будемо розглядати групи учасників (A_1, \dots, A_k, C), де учасники вибираються з множини можливих учасників $U = \{A_1, \dots, A_l\}$ (їх іноді називають процесорами). При цьому всі A_i і B взаємодіють тільки з виділеним учасником C. Учасник C називається комбінатором (combiner) або дилером (лідером) і доданий для досягнення анонімності, щоб B не міг дізнатися про склад групи A_i .

На множині U виділяється структура доступу Γ - множина підмножин U, які називаються дозволеними коаліціями. Множина Γ зазвичай монотонна по

включенню: якщо $V \subset V' \wedge V' \in \Gamma$, то $V \in \Gamma$. Протокол повинен бути працездатний, якщо $\{A_{i1}, \dots, A_{ik}\} \in \Gamma$.

Як і в інших типах криптографічних протоколів, в протоколі розділу секрету учасники, взагалі кажучи, не довіряють один одному, і кожен з них може виявитися противником, в тому числі і дилер. Груповий криптографічний протокол повинен відповідати наступним властивостям.

- Надійність. Для будь-якої дозволеної коаліції $\{A_{i1}, \dots, A_{ik}\} \in \Gamma$ протокол $((A_{i1}, \dots, A_{ik}, C), B)$ задовольняє вимогам надійності і безпеки.

- Групова безпека. Для будь-якої дозволеної коаліції $\{A_{i1}, \dots, A_{ik}\} \in \Gamma$ ніякі змовники $A_{i1}^*, \dots, A_{ik}^*, C^*$ такі, що будь-який A_i^* і C^* має доступ до секретів всіх A_i і C , не можуть порушити надійність системи.

Надалі будемо розглядати коаліції учасників як підмножину множини номерів учасників $I \in \Gamma \subset \{1, \dots, l\} = L$.

Крім схем поділу секретів групові протоколи найчастіше використовуються для вирішення наступних завдань:

- групове обчислення значення деякої функції - може застосовуватися для побудови групових криптосистем, групового підпису, групової ідентифікації /протоколи конфіденційних обчислень /;
- групова генерація псевдовипадкових чисел;
- анонімна пошта та віддалене таємне голосування та інше.

11.2 Порогова схема поділу секрету

Мета протоколу поділу секрету (secret sharing) – забезпечити відновлення секретного значення $S \in K$ тільки дозволеної коаліції учасників, кожен з яких має значення, яке називають часткою (share) або тінню (shadow) секрету $s_i \in T$.

Передбачається, що схема поділу секрету включає n учасників і керується авторизованим дилером. Основна функція дилера - поділ секрету на n компонентів («тіней») і розподіл їх серед учасників так, що будь-які m (і більше) учасників, зібравшись разом і пред'явивши тіні, можуть відновити секретний ключ. Причому будь-які $(m-1)$ і менш учасників не можуть цього зробити.

Криптографічні схеми поділу секрету відомі також під назвою $m - із - n$ - схем або (m, n) - порогових схем.

Визначення. Множина $\{s_1, \dots, s_n\}$, що задовольняє наступним двом умовам, називається (m, n) - порогові. схемою поділу секрету.

1. Секрет S легко може бути обчислений за будь-яким значенням s_i .

2. Знання будь-яких $(m-1)$ значень s_i не дозволяє визначити секрет S .

Компроміс між криптостійкістю і гнучкістю схеми регулюється вибором параметрів m, n .

Протокол складається з двох фаз.

На фазі поділу секрету дилер, що знає деякий секрет S , генерує n часток секрету s_1, \dots, s_n і посилає s_i процесору A_i по захищеному каналу зв'язку. На фазі відновлення секрету будь-яка підмножина з не менш ніж m процесорів однозначно відновлює секрет, обмінюючись повідомленнями по захищених

каналах зв'язку. А будь-яка підмножина з не більше ніж $(m-1)$ процесорів не може відновити секрет.

Визначення. Схемою поділу секрету з множиною секретів K і множиною часток T називається пара алгоритмів (D, R) , таких, що

– $D(S)$ - імовірнісний поліноміальний алгоритм роздачі, який виходячи з секрету $s \in K$ обчислює набір часток учасників $s_i \in T$.

– $R(I, x_1, \dots, x_I)$ - детермінований поліноміальний алгоритм відновлення секрету такий, що $D(S) = (s_1, \dots, s_n) \rightarrow S = R(I, x_1, \dots, x_I)$.

Для кожної дозволеної коаліції I алгоритм задає функцію

$\varphi_I(x_1, \dots, x_I) = R(I, x_1, \dots, x_I)$, для кожної недозволеної коаліції I всі можливі значення S рівноймовірні: $\forall I \notin U, \forall x_i \in T, P(s=x, \forall i \in I, s_i=x_i) = P(s=x)$.

Визначення. Схема поділу секрету називається досконалою, якщо довільна коаліція або повністю розкриває секрет, або в результаті не отримує про нього ніякої апостеріорної інформації.

11.3 Схема обчислення ключа доступу

1. Схема обчислення ключа доступу при поділі секрету між двома учасниками.

Розглянемо найпростіший варіант досконалої схеми. Нехай є первинний секретний ключ і два учасники, що утворюють легальну коаліцію для отримання доступу до секретної інформації. Жоден з учасників не знає первинного секретного ключа, і тільки об'єднання тіней, якими володіють учасники, дозволяє відновити первинний ключ і виконати дешифрування.

Практична реалізація ідеї заснована на властивостях арифметики за модулем два. Розглянемо первинний ключ S як послідовність двійкових символів (біт) довжини n . Тоді тінь першого учасника s_1 обчислюється шляхом побітного додавання первинного ключа і шумової (випадкової) послідовності s_2 двійкових символів довжини n . Та ж шумова послідовність s_2 використовується як тінь іншого учасника. Отже, первинний ключ S може бути обчислений додаванням за модулем 2 тіней учасників: $S = s_1 + s_2$.

Оцінка затрат розкриття первинного ключа для кожного з учасників, так само як і для нелегального зловмисника, становить 2^n спроб дешифрування в гіршому випадку і 2^{n-1} в середньому.

Схема Шаміра

Схема поділу секрету, запропонована Шаміром, побудована за принципом поліноміальної інтерполяції. Нехай заданий многочлен степеня $(m-1)$ над кінцевим полем Галуа $GF(q)$ з q елементів ($GF(q) = \mathbb{Z}_q$, де q - просте).

Секретний ключ задається вільним членом a_0 , всі інші коефіцієнти многочлена - випадкові елементи поля. Поле $GF(q)$ відомо всім учасникам. Кожна з n тіней представляє собою точку (x_i, y_i) кривої, що описується многочленом $P(x)$, $x_i \neq 0$.

Скориставшись інтерполяційною формулою Лагранжа, можна відновити вихідний многочлен (i , таким чином, секрет a_0) по будь-яких m точках (тінях).

При цьому ймовірність розкриття секрету в разі довільних $(m-1)$ тіней оцінюється як q^{-1} , тобто в результаті інтерполяції по $(m-1)$ точці секретом може бути будь-який елемент поля з рівною вірогідністю. Отже, схема Шаміра є досконалою.

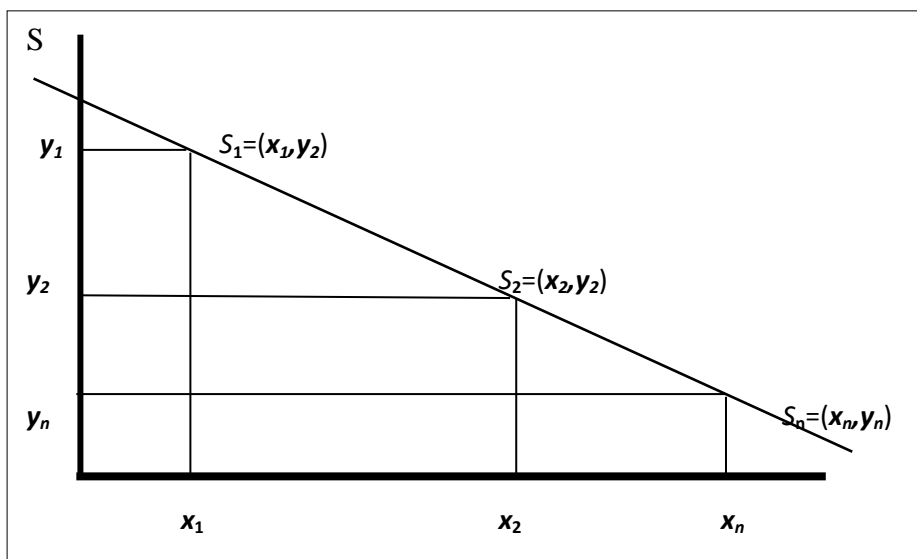


Рисунок 11.1 - Відновлення секрету за двома тінями

На рисунку 11.1 представлений спеціальний випадок; $m=2$ (для відновлення секрету необхідно дві тіні). Таким чином, двочлен описує деяку пряму, перетинає з віссю y в точці s (секрет). Кожна тінь - точка на прямій. Отже, секрет може бути відновлений за двома довільним тіням, так як для однозначного визначення прямої достатньо двох довільних точок.

У разі задання однієї тіні в якості шуканого секрету може бути обрана будь-яка точка на осі y , так як через одну точку можна провести множину різних прямих, що перетинаються з віссю y в довільних точках.

Схема Блеклі

Схема поділу секрету Блеклі має геометричну природу. Секрет являє собою точку в m - вимірному просторі. Відзначимо, що для випадку $m>2$ всі геометричні побудови виконуються над скінченим полем $GF(q)$. Кожна з n тіней задається як гіперплощина в m - вимірному просторі. Визначення секрету зводиться до знаходження точки перетину m гіперплощин.

На рисунку 11.2 представлений спеціальний випадок схеми Блеклі. Кожна тінь - пряма, що проходить через цю точку. Для відновлення секрету S (точки на площині) необхідно мати принаймні дві тіні.

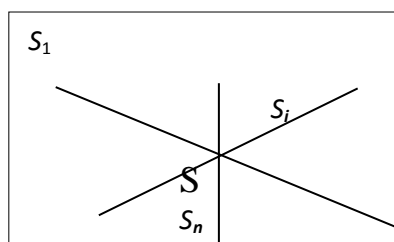


Рисунок 11.2 – Схема Блеклі

11.4 Метод «розшаровування» зображення

Наор і Шамір розробили оригінальний варіант схеми розподілу секрету. Автори схеми запропонували наступну постановку завдання. Нехай є секретне зображення (в деякому графічному форматі), яке необхідно розподілити серед n учасників.

Для цього зображення «розшаровується» на n складових (тіней) таким чином, що об'єднання будь-яких m з них дозволяє відновити зображення. Очевидно, що жодна з тіней не дає уявлення про вихідне зображення. Більше того, ніякі $(m-1)$ і менш тіней не дозволяють відновити вихідне зображення. Можливий варіант конструктивної реалізації схеми полягає в «розшаровуванні» зображення на чорні та білі пікселі з наступною їх обробкою.

Схема є досконалою і проста в реалізації. Додаткова модифікація дозволяє одержувати в якості тіней не «шумові», а цілком змістовні зображення (наприклад, зображення пейзажу або будівлі тощо), що дозволяє приховати сам факт поділу секрету.

11.5 Протокол поділу секрету, що перевіряється

У звичайних схемах поділу секрету розглядається пасивний супротивник, а саме, противником є не більш ніж $(m-1)$ процесорів, які, об'єднавши свої частки, намагаються отримати будь-яку інформацію про значення секрету.

Якщо ж на фазі відновлення секрету діє активний противник, можливі такі ситуації:

- дилер намагається роздати процесорам неправильні тіні,
- процесори A_i переслідують мету зірвати відновлення секрету S , посилаючи чесним учасникам замість часткою секрету будь-яку іншу інформацію.

Подібна дія призведе до того, що секрет не буде правильно відновлений. При цьому неможливо визначити, хто з учасників коаліції винен в цьому. У гіршому випадку учасники просто не визначать, що відновлений секрет не відповідає вихідному розділеному секрету.

Для захисту в подібній ситуації використовується протокол поділу секрету, що перевіряється або верифікована схема поділу секрету.

Можливе вирішення полягає в застосуванні доказів з нульовим розкриттям, що дозволяє одному учаснику довести коректність виконаних ним обчислень і знання тіні, отриманої в результаті поділу секрету, іншому учаснику схеми, не розкриваючи при цьому самої тіні. Даний протокол дозволяє вирішувати наступну практичну задачу: кожен з учасників може переконатися в тому, що отримана ним тінь, об'єднана з тінями інших учасників, дозволяє гарантовано відновити розділений секрет.

Протокол поділу секрету, що перевіряється за схемою Шаміра

Для прикладу розглянемо схему Шаміра і ситуацію, в якій коаліція з m учасників намагається відновити секрет S за умови, що один з учасників - брехун і надає «помилкову» тінь (не ту, що була отримана в результаті поділу секрету).

Фаза поділу секрету починається з того, що дилер публікує секрет S в «зашифрованому» вигляді (точніше - виконує прив'язку до рядка S , за аналогією з прив'язкою до біту).

Дилер вибирає випадковий поліном степеня $m - 1$: $Q(x) = a_0 + a_1 * x + \dots + a_{m-1} * x^{m-1}$, де $a_0 = S$, обчислює $r_i = g^{a_i} \bmod p$ ($i = 1, \dots, n$) і публікує r_1, \dots, r_m .

За допомогою цієї інформації кожен процесор A_j може перевірити, що значення s_j , отримане ним від дилера, дійсно є часткою секрету S . Для цього він повинен перевірити рівність

$$g^{s_j} = r_0 (r_1)^j \dots (r_{m-1})^{j^{m-1}} \bmod q.$$

$$\text{Справді: } r_0 r_1^j r_2^{j^2} \dots r_{m-1}^{j^{m-1}} = g^{a_0} g^{a_1 j} g^{a_2 j^2} \dots g^{a_{m-1} j^{m-1}} = g^{Q(j)} \bmod q.$$

На фазі роздачі секрету дилер обчислює $s_j = Q(j)$, $j = 1, \dots, n$ і посилає це значення (частки або тіні секрету) процесору A_j по захищеному каналу.

Конструкцію протоколу для фази відновлення секрету розглянемо в найбільш простому випадку, коли дилер чесний. На цій фазі кожний процесор A_j надсилає кожному іншому процесору A_i свою частку s_j .

Будь-який чесний учасник A_i , отримавши деяке значення s_j від A_j , перевіряє це значення, як описано вище, і відкидає всі частки s_j , які не пройшли перевірку. Оскільки чесних учасників не менше m , A_i отримає принаймні m правильних часткою секрету.

Використовуючи алгоритм відновлення секрету зі схеми Шаміра, відновлюємо значення S . Частки секрету є гіперплощинами в просторі многочленів над полем Z_q . Для відновлення секрету будується єдина точка перетину m гіперплощин за допомогою інтерполяційного полінома Лагранжа:

$$Q'(x) = \sum_{i \in I} Q(x_i) \pi_i, \quad \pi_i = \prod_{j \in I, j \neq i} \frac{x - x_j}{x_i - x_j}.$$

На відміну від звичайних схем поділу секрету стійкість даного протоколу ґрунтується на припущенні про обчислювані труднощі завдання дискретного логарифмування. Тому, якщо в звичайних схемах поділу секрету потрібно, щоб будь-яка підмножина учасників, що не становить кворуму, не отримала ніякої інформації про секрет, то в багатьох схемах перевіряючого розподілу секрету така підмножина лише «не може відновити» секрет в тому сенсі, що для його відновлення потрібно вирішити деяку гіпотетично важку обчислювану задачу. У розглянутому вище прикладі будь-який учасник міг би дізнатися секрет S , якби він умів обчислювати дискретні логарифми.

11.6 Протоколи конфіденційного обчислення

Припустимо, що за допомогою протоколу поділу секрету розділені два секрети S_1 і S_2 і що обидва ці секрети є числами. Тепер уявімо собі ситуацію, що після цього було потрібно розділити секрет $S = S_1 + S_2$. Це може зробити дилер за допомогою того ж протоколу. А чи можуть процесори виконати те ж саме без участі дилера?

Нехай $Q_1(x) = a_0 + a_1 * x + \dots + a_{m-1} * x^{m-1}$ і $Q_2(x) = b_0 + b_1 * x + \dots + b_{m-1} * x^{m-1}$ – поліноми, які використовувалися для розділення секретів S_1 і S_2 відповідно.

Нехай $g_i^{(1)} = g^{a_i} \bmod q$ і $g_i^{(2)} = g^{b_i} \bmod q$, $i=0, \dots, m-1$ – значення, використовуються для перевірки правильності часток секрету.

Нехай $s_j^{(1)} = Q_1(j)$ і $s_j^{(2)} = Q_2(j)$, $j=1, \dots, n$ – частки секретів S_1 і S_2 , отримані процесором A_j .

Ясно, що $Q(x) = Q_1(x) + Q_2(x)$ – поліном степеня $m-1$ і $Q(0) = S$.

Тому кожен процесор A_j може обчислити частку s_j секрету S просто за формулою $s_j = s_j^{(1)} + s_j^{(2)}$.

Ці частки перевіряються за допомогою значень $g_i = g_i^{(1)} g_i^{(2)} \bmod q$.

Виконуючи такого роду обчислення над долями секретів, процесори можуть обчислити будь-яку функцію над кінцевим полем «перевіряється чином». Типова задача тут така. Потрібно обчислити значення функції f на деякому наборі значень аргументів y_1, \dots, y_k . За допомогою схеми перевіряється поділу секрету обчислюються частки $x_i^{(1)}, \dots, x_i^{(k)}$, ($i=1, \dots, n$) цих значень. На початку виконання протоколу частка x_i відома процесору A_i і тільки йому. Протокол повинен забезпечувати обчислення значення

$$f(x_i^{(1)}, \dots, x_i^{(k)}) = f(y_1, \dots, y_k)$$

таким чином, щоб для деякого параметра m :

1) у результаті виконання протоколу будь-яка підмножина з не більше ніж $m-1$ процесорів не отримувало жодної інформації про значення x_i інших процесорів (крім тієї, яка впливає з відомих їм часток) і значення функції $f(x_i^{(1)}, \dots, x_i^{(k)})$,

2) при будь-яких діях нечесних учасників інші учасники обчислюють правильне значення $f(y_1, \dots, y_k)$, якщо тільки кількість чесних учасників не менш m .

ЛІТЕРАТУРА

1. С. М. Авдошин. Криптоанализ: современное состояние и перспективы развития/ С. М. Авдошин, А. А. Савельева. [Электронный ресурс].- режим доступа: <http://docplayer.ru/25985958-Kriptoanaliz-sovremennoe-sostoyanie-i-perspektivy-razvitiya.html>.
2. Э. А. Болелов Пособие к выполнению лабораторных работ по дисциплине «криптографические методы защиты информации» / Болелов Э.А. – Москва, 2010. – 33с.
3. Гапак О.М. Захист інформації в комп'ютерних системах. Підручник / О.М. Гапак, С.І. Балога .- Ужгород: видавництво ПП «АУТДОР-ШАРК», 2021. – 184 с.
4. Гундарь К. Ю. Защита информации в компьютерных системах / К. Ю. Гундарь, А. Ю. Гундарь, Д. А. Янишевский. – К.: «Корнейчук», 2000.– 152с.
5. Жданов О.Н. Криптоанализ классических шифров / О.Н. Жданов,. Куденкова И.А. – Красноярск, 2008. – 107с.
6. Зукова О.Л. Основы криптографической защиты информации. Учебное пособие. / О.Л. Зукова. —М., 2005. — 207 с.
7. Кравець О. Підвищення ефективності крипто аналізу сучасних поточкових шифрів / О. Кравець, С. Лупенко, А. Луцків // Вісник Національного університету "Львівська політехніка". – 2012. – № 741 : Автоматика, вимірювання та керування. – С. 240–245.
8. Шнайер Б. Прикладная криптография: Протоколы, алгоритмы и исходные тексты на языке С / Б. Шнайер.– М. : Триумф, 2002. – 816 с.
9. Остапов С. Е. Основи криптографії: навчальний посібник / С. Е. Остапов, Л. О. Валь. – Чернівці: Книги–XXI, 2008. – 188 с.
10. Haranadh Gavara, Harendra Kumar Mishra, Surendra Kumar Y. Cryptanalysis using Neural Networks // Available via netlab.cs.iitm.ernet.in/cs650/2006/TermPapers/Group9.pdf.

ДОДАТОК А

Таблиця Віжінера для української мови (алфавит Z32- 32 букви и пропуск)

[illegible]

ДОДАТОК Б

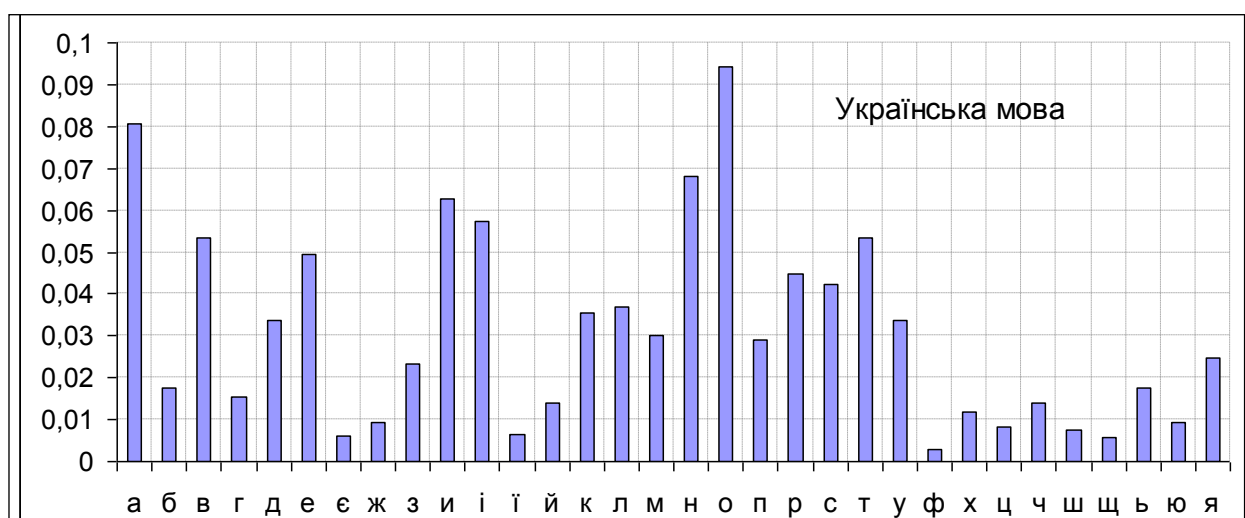
Таблиця Б1

Ранжовані частоти використання букв української мови

О	0,0942	р	0,0448	я	0,0248	ж	0,0093
А	0,0807	с	0,0424	з	0,0232	ю	0,0093
Н	0,0681	л	0,0369	б	0,0177	ц	0,0083
И	0,0626	к	0,0354	ь	0,0177	ш	0,0076
І	0,0575	д	0,0338	г	0,0155	ї	0,0065
В	0,0535	у	0,0336	ч	0,0141	є	0,0061
Т	0,0535	м	0,0303	й	0,0138	щ	0,0056
Е	0,0495	п	0,0290	х	0,0119	ф	0,0028

Діаграма Б2

Гістограма частот букв алфавіту української мови



Таблиця Б3

Середньостатистичні частоти букв та пропуску між словами в українській мові

—	0,138	і	0,044	л	0,027	г	0,013	ж	0,007
о	0,086	р	0,043	л	0,027	ч	0,011	ю	0,008
н	0,068	е	0,042	п	0,025	х	0,011	є	0,005
а	0,064	с	0,037	з	0,020	ї	0,010	ш	0,004
и	0,055	к	0,033	я	0,019	ц	0,010	ф	0,003
в	0,046	м	0,029	ь	0,016	ш	0,005	г	0,000
т	0,045	у	0,027	б	0,013	й	0,009		

Таблиця Б4

Середньостатистичні частоти букв без пропуску між словами в українській мові

о	0,100	р	0,050	л	0,031	ч	0,012	ш	0,006
н	0,079	е	0,048	п	0,029	х	0,013	є	0,005
а	0,075	с	0,043	з	0,023	ц	0,012	ф	0,004

и	0,064	к	0,039	я	0,021	ї	0,011	ш	0,004
в	0,053	м	0,034	ь	0,018	й	0,010	г	0,000
і	0,052	л	0,032	б	0,015	ю	0,009		
т	0,052	у	0,032	г	0,015	ж	0,008		

Таблиця Б5

Таблиця розподілу літер в англійській мові

Буква	Частота	Буква	Частота	Буква	Частота
a	0.0804	b	0.0154	c	0.0306
d	0.0399	e	0.1251	f	0.0230
g	0.0196	h	0.0549	i	0.0726
j	0.0016	k	0.0067	l	0.0414
m	0.0253	n	0.0709	o	0.0760
p	0.0200	q	0.0011	r	0.0612
s	0.0654	t	0.0925	u	0.0271
v	0.0099	w	0.0192	x	0.0019

у	0.0173				
---	--------	--	--	--	--

Таблица Б6

Таблица розподілу літер в російській мові

Буква	Частота	Буква	Частота	Буква	Частота
а	0.062	л	0.035	ц	0.004
б	0.014	м	0.026	ч	0.012
в	0.038	н	0.053	ш	0.006
г	0.013	о	0.090	щ	0.003
д	0.025	п	0.023	ы	0.016
е, ё	0.072	р	0.040	ь, ъ	0.014
ж	0.007	с	0.045	э	0.003
з	0.016	т	0.053	ю	0.006
и	0.062	у	0.021	я	0.018
й	0.010	ф	0.002	пробіл	0.172
к	0.028	х	0.009		

ДОДАТОК В

Таблиця В1

**Середньостатистичні частоти повторюваності біграм в українській мові із
врахуванням пропуску між словами**

	А	Б	В	Г	Ґ	Д	Е	Є	Ж
А	0,0000	0,0010	0,0030	0,0013	0,0000	0,0019	0,0000	0,0015	0,0008
Б	0,0014	0,0000	0,0001	0,0001	0,0000	0,0000	0,0013	0,0002	0,0000
В	0,0068	0,0000	0,0002	0,0000	*	0,0008	0,0017	0,0000	0,0004
Г	0,0020	0,0000	0,0000	0,0000	*	0,0000	0,0002	0,0000	0,0000
Ґ	0,0000	*	0,0000	*	*	0,0000	0,0000	0,0000	*
Д	0,0021	0,0003	0,0005	0,0001	0,0000	0,0001	0,0019	0,0000	0,0008
Е	0,0003	0,0004	0,0009	0,0006	0,0000	0,0016	0,0000	0,0000	0,0010
Є	0,0000	0,0000	0,0001	0,0000	0,0000	0,0002	0,0000	0,0000	0,0000
Ж	0,0008	0,0000	0,0000	0,0000	0,0000	0,0001	0,0021	*	0,0000
З	0,0051	0,0009	0,0013	0,0002	0,0000	0,0004	0,0004	0,0000	0,0000
И	0,0000	0,0003	0,0030	0,0003	0,0000	0,0007	0,0000	0,0002	0,0001
І	0,0003	0,0005	0,0046	0,0004	0,0000	0,0047	0,0000	0,0004	0,0004
Ї	*	0,0000	0,0002	0,0000	*	0,0000	*	*	0,0000
Й	0,0000	0,0002	0,0001	0,0000	0,0000	0,0002	0,0000	0,0000	0,0000
К	0,0035	0,0000	0,0011	0,0000	*	0,0000	0,0003	*	0,0000
Л	0,0037	0,0000	0,0000	0,0001	0,0000	0,0002	0,0030	0,0000	0,0000
М	0,0040	0,0000	0,0001	0,0000	0,0000	0,0000	0,0024	0,0000	0,0001
Н	0,0118	0,0000	0,0001	0,0000	0,0000	0,0004	0,0044	0,0001	0,0000
О	0,0000	0,0044	0,0090	0,0060	0,0000	0,0051	0,0002	0,0003	0,0018
П	0,0012	0,0000	0,0000	0,0000	*	0,0001	0,0033	0,0000	0,0000
Р	0,0065	0,0001	0,0004	0,0005	0,0000	0,0001	0,0056	0,0000	0,0004
С	0,0008	0,0000	0,0013	0,0000	*	0,0000	0,0010	0,0000	0,0000
Т	0,0065	0,0000	0,0023	0,0000	*	0,0000	0,0057	0,0001	0,0001
У	0,0002	0,0002	0,0021	0,0006	0,0000	0,0016	0,0000	0,0006	0,0004
Ф	0,0004	0,0000	0,0000	0,0000	*	0,0000	0,0003	0,0000	*
Х	0,0007	0,0000	0,0001	0,0000	*	0,0001	0,0000	*	*
Ц	0,0002	0,0000	0,0000	0,0000	*	0,0000	0,0027	*	*
Ч	0,0024	0,0000	0,0000	0,0000	*	0,0000	0,0015	*	*
Ш	0,0003	0,0000	0,0001	0,0000	*	0,0000	0,0009	*	*
Щ	0,0001	*	0,0000	*	*	0,0000	0,0006	*	*
Ь	0,0000	0,0003	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000
Ю	0,0000	0,0000	0,0005	0,0000	0,0000	0,0003	0,0000	0,0003	0,0000
Я	0,0000	0,0000	0,0006	0,0005	*	0,0005	0,0000	0,0002	0,0002
—	0,0031	0,0044	0,0140	0,0021	0,0000	0,0077	0,0021	0,0007	0,0006

	З	И	І	Ї	Й	К	Л	М	Н
А	0,0013	*	0,0000	0,0011	0,0012	0,0028	0,0058	0,0035	0,0073
Б	0,0000	0,0007	0,0011	0,0000	0,0000	0,0001	0,0010	0,0002	0,0009
В	0,0002	0,0075	0,0052	0,0000	0,0000	0,0004	0,0013	0,0001	0,0030
Г	0,0000	0,0003	0,0007	*	*	0,0001	0,0009	0,0000	0,0003
Ґ	0,0000	0,0000	0,0000	0,0000	*	0,0000	0,0000	0,0000	0,0000
Д	0,0001	0,0022	0,0019	0,0000	0,0000	0,0006	0,0009	0,0002	0,0029
Е	0,0013	0,0000	0,0000	0,0001	0,0005	0,0027	0,0014	0,0024	0,0070
Є	0,0000	*	*	0,0002	*	0,0001	0,0000	0,0005	0,0001
Ж	0,0000	0,0007	0,0002	*	*	0,0001	0,0004	0,0000	0,0009
З	0,0000	0,0005	0,0005	0,0000	0,0000	0,0004	0,0004	0,0007	0,0022
И	0,0008	*	0,0000	0,0002	0,0025	0,0028	0,0016	0,0034	0,0033
І	0,0017	*	0,0000	0,0023	0,0022	0,0010	0,0022	0,0005	0,0033
Ї	0,0000	*	*	0,0007	0,0000	0,0002	0,0000	0,0002	0,0009
Й	0,0000	*	0,0000	0,0000	*	0,0001	0,0002	0,0001	0,0009
К	0,0000	0,0046	0,0024	*	*	0,0000	0,0011	0,0000	0,0006
Л	0,0001	0,0048	0,0030	*	*	0,0002	0,0000	0,0000	0,0001
М	0,0000	0,0047	0,0025	0,0000	*	0,0004	0,0002	0,0000	0,0005
Н	0,0001	0,0089	0,0061	*	*	0,0010	0,0000	0,0000	0,0061
О	0,0024	*	0,0000	0,0036	0,0002	0,0022	0,0031	0,0070	0,0048
П	0,0001	0,0011	0,0022	*	*	0,0000	0,0015	0,0000	0,0001
Р	0,0002	0,0057	0,0033	0,0001	0,0000	0,0013	0,0000	0,0017	0,0014
С	0,0000	0,0021	0,0011	*	0,0000	0,0014	0,0019	0,0001	0,0019
Т	0,0000	0,0070	0,0032	0,0000	*	0,0008	0,0001	0,0000	0,0013
У	0,0003	*	0,0000	0,0000	0,0001	0,0025	0,0015	0,0009	0,0007
Ф	0,0000	0,0000	0,0005	*	*	0,0000	0,0000	0,0000	0,0000
Х	0,0000	0,0002	0,0007	*	*	0,0000	0,0000	0,0000	0,0005
Ц	0,0000	0,0003	0,0042	*	*	0,0000	0,0001	0,0000	0,0001
Ч	0,0000	0,0022	0,0003	*	*	0,0001	0,0001	0,0000	0,0031
Ш	0,0000	0,0010	0,0004	*	*	0,0002	0,0002	0,0000	0,0004
Щ	0,0000	0,0001	0,0001	*	*	*	*	*	0,0000
Ь	0,0000	*	0,0000	0,0000	0,0000	0,0042	0,0000	0,0002	0,0021
Ю	0,0000	*	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000
Я	0,0006	*	0,0000	0,0000	0,0000	0,0024	0,0002	0,0010	0,0008
—	0,0106	0,0000	0,0048	0,0012	0,0020	0,0055	0,0011	0,0065	0,0104

	О	П	Р	С	Т	У	Ф	Х	Ц
А	0,0000	0,0010	0,0030	0,0034	0,0041	0,0004	0,0001	0,0017	0,0021
Б	0,0014	0,0000	0,0006	0,0004	0,0001	0,0029	*	0,0002	0,0000
В	0,0052	0,0004	0,0002	0,0016	0,0003	0,0010	0,0000	0,0001	0,0001
Г	0,0063	0,0000	0,0012	0,0000	0,0000	0,0005	0,0000	0,0001	0,0000
Г	0,0000	*	0,0001	*	*	0,0000	0,0000	*	*
Д	0,0053	0,0005	0,0008	0,0003	0,0001	0,0021	0,0000	0,0001	0,0000
Е	0,0008	0,0007	0,0087	0,0014	0,0015	0,0000	0,0002	0,0006	0,0006
Є	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000
Ж	0,0002	0,0000	0,0000	*	0,0000	0,0005	0,0000	0,0000	0,0001
З	0,0009	0,0005	0,0006	0,0001	0,0000	0,0008	0,0000	0,0000	0,0000
И	0,0000	0,0007	0,0017	0,0043	0,0031	0,0000	0,0001	0,0052	0,0012
І	0,0003	0,0001	0,0007	0,0022	0,0015	0,0000	0,0001	0,0003	0,0001
Ї	0,0000	*	0,0000	0,0000	0,0000	*	0,0000	0,0005	0,0000
Й	0,0005	0,0000	0,0000	0,0008	0,0001	0,0000	0,0000	0,0000	0,0000
К	0,0096	0,0000	0,0025	0,0004	0,0017	0,0023	*	0,0000	0,0008
Л	0,0033	0,0000	0,0000	0,0000	0,0001	0,0008	0,0000	0,0000	0,0000
М	0,0046	0,0010	0,0001	0,0003	0,0000	0,0026	0,0000	0,0000	0,0000
Н	0,0115	0,0000	0,0000	0,0018	0,0021	0,0020	0,0008	0,0000	0,0005
О	0,0001	0,0017	0,0064	0,0057	0,0021	0,0001	0,0002	0,0007	0,0008
П	0,0070	0,0000	0,0071	0,0002	0,0001	0,0004	0,0000	0,0000	0,0001
Р	0,0095	0,0001	0,0000	0,0007	0,0007	0,0021	0,0000	0,0003	0,0001
С	0,0015	0,0020	0,0000	0,0000	0,0115	0,0018	0,0002	0,0002	0,0004
Т	0,0051	0,0000	0,0028	0,0002	0,0005	0,0022	0,0000	0,0000	0,0001
У	0,0000	0,0008	0,0011	0,0012	0,0017	0,0000	0,0000	0,0002	0,0000
Ф	0,0014	0,0000	0,0003	0,0000	0,0000	0,0003	0,0000	*	0,0000
Х	0,0014	0,0000	0,0003	0,0000	0,0001	0,0002	0,0000	0,0000	0,0000
Ц	0,0001	0,0000	0,0000	0,0000	0,0007	0,0000	0,0000	0,0000	0,0001
Ч	0,0004	*	0,0000	0,0000	0,0000	0,0003	0,0000	0,0000	0,0000
Ш	0,0005	0,0000	0,0000	0,0000	0,0003	0,0003	0,0000	0,0000	0,0000
Щ	0,0028	*	*	*	*	0,0001	*	*	*
Ь	0,0015	0,0000	0,0000	0,0019	0,0004	*	0,0000	0,0000	0,0000
Ю	0,0000	0,0000	0,0000	0,0000	0,0022	0,0000	0,0000	0,0000	0,0000
Я	0,0000	0,0000	0,0002	0,0002	0,0009	0,0000	0,0000	0,0004	0,0000
—	0,0045	0,0159	0,0051	0,0095	0,0080	0,0038	0,0016	0,0009	0,0030

	Ч	Ш	Щ	Ь	Ю	Я	—
А	0,0011	0,0004	0,0001	*	0,0009	0,0001	0,0133
Б	0,0000	0,0001	0,0000	*	0,0000	0,0000	0,0003
В	0,0003	0,0001	0,0000	*	0,0000	0,0008	0,0078
Г	0,0000	0,0000	*	*	0,0000	0,0000	0,0002
Г	0,0000	0,0000	*	*	*	0,0000	0,0000
Д	0,0002	0,0000	0,0000	0,0001	0,0000	0,0003	0,0020
Е	0,0004	0,0001	0,0000	*	0,0001	0,0001	0,0060
Є	0,0000	0,0000	0,0000	*	0,0003	0,0000	0,0022
Ж	0,0000	0,0000	0,0000	*	0,0000	0,0000	0,0008
З	0,0000	0,0000	0,0000	0,0004	0,0000	0,0002	0,0032
И	0,0018	0,0004	0,0006	*	0,0000	0,0004	0,0161
І	0,0010	0,0008	0,0001	*	0,0004	0,0009	0,0116
Ї	0,0000	0,0000	0,0000	*	*	*	0,0070
Й	0,0000	0,0001	0,0000	*	0,0000	0,0000	0,0057
К	0,0000	0,0000	0,0001	0,0000	0,0000	0,0000	0,0024
Л	0,0000	*	*	0,0044	0,0010	0,0023	0,0003
М	0,0008	0,0000	0,0000	0,0000	0,0000	0,0002	0,0044
Н	0,0001	0,0005	0,0000	0,0013	0,0005	0,0056	0,0018
О	0,0013	0,0005	0,0001	0,0000	0,0022	0,0002	0,0139
П	0,0000	0,0000	0,0000	*	0,0006	0,0001	0,0001
Р	0,0000	0,0005	0,0000	0,0002	0,0002	0,0007	0,0008
С	0,0000	0,0000	*	0,0038	0,0000	0,0033	0,0007
Т	0,0000	0,0000	0,0000	0,0050	0,0002	0,0005	0,0012
У	0,0006	0,0001	0,0001	*	0,0006	0,0000	0,0095
Ф	0,0000	0,0000	*	0,0000	0,0000	0,0000	0,0000
Х	0,0000	0,0000	*	*	*	*	0,0071
Ц	*	0,0000	*	0,0007	0,0002	0,0005	0,0001
Ч	0,0001	0,0000	0,0000	0,0000	0,0000	0,0001	0,0002
Ш	0,0000	0,0001	*	0,0000	0,0000	0,0000	0,0002
Щ	*	*	0,0000	*	*	*	0,0001
Ь	0,0000	0,0005	0,0000	*	0,0000	0,0000	0,0045
Ю	0,0006	0,0000	0,0000	*	0,0002	*	0,0036
Я	0,0001	0,0000	0,0000	*	0,0001	0,0000	0,0098
—	0,0021	0,0006	0,0028	*	0,0000	0,0024	*

Таблиця частот біграм російської мови

	А	Б	В	Г	Д	Е	Ж	З	И	И	К	Л	М	Н	О	П
А	2	12	35	8	14	7	6	15	7	7	19	27	19	45	3	11
Б	5					9	1		6			6		2	21	
В	35	1	5	3	3	32		2	17		7	10	3	9	58	6
Г	7				3	3			5		1	5		1	50	
Д	25		3	1	1	29	1	1	13		1	5	1	13	22	3
Е	2	9	18	11	27	7	5	10	6	15	13	35	24	63	7	16
Ж	5	1			6	12			5					6		
З	35	1	7	1	5	3			4		2	1	2	9	9	1
И	4	6	22	5	10	21	2	23	19	11	19	21	20	32	8	13
И	1	1	4	1	3		1	2	4		5	1	2	7	9	7
К	24	1	4	1		4	1	1	26		1	4	1	2	66	2
Л	25	1	1	1	1	33	2	1	36		1	2	1	8	30	2
М	18	2	4	1	1	21	1	2	23		3	1	3	7	19	5
Н	54	1	2	3	3	34			58		3		1	24	67	2
О	1	28	84	32	47	15	7	18	12	29	19	41	38	30	9	18
П	7					15			4			9		1	46	

	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ы	Ь	Э	Ю	Я
А	26	31	27	3	1	10	6	7	10	1			2	6	9
Б	8	1		6						1	11				2

В	6	19	6	7		1	1	2	4	1	18	1	2		3
Г	7			2											
Д	6	8	1	10			1	1	1		5	1			1
Е	39	37	33	3	1	8	3	7	3	3			1	1	2
Ж		1													
З	3	1		2							4				4
И	11	29	29	3	1	17	3	11	1	1			1	3	17
И	3	10	2				1	3	2						
К	10	3	7	10			1								
Л		3	1	6		4		1			2	30		4	9
М	2	5	3	9	1			2			5	1	1		3
Н	1	9	9	7	1		5	2			36	3			5
О	43	50	39	3	2	5	2	12	4	3			2	3	2
П	41	1		6							2				2

	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П
Р	55	1	4	4	3	37	3	1	24		3	1	3	7	56	2
С	8	1	7	1	2	25			6		40	13	3	9	27	11
Т	35	1	27	1	3	31		1	28		5	1	1	11	56	4
У	1	4	4	4	11	2	6	3	2		8	5	5	5	1	5
Ф	2					2			2						1	
Х	4	1	4	1	3	1		2	3		4	3	3	4	18	5
Ц	3					7			10		2				1	
Ч	12					23			13		2			6		
Ш	5					11			14		1	2		2	2	
Щ	3					8			6					1		
Ы		1	9	1	3	12		2	4	7	-3	6	6	3	2	10
Ь		2	4	1	1	2		2	2		6		3	13	2	4
Э											1			1		
Ю		2	1	2	1			3	1		1		1	1	1	3
Я	1	3	9	1	3	3	1	5	3	2	3	3	4	6	3	6

	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ы	Ь	Э	Ю	Я
Р	1	5	9	16		1	1	1	2		8	3			5
С	4	11	82	6		1	1	2	2		1	8			17
Т	26	18	2	10				1			И	21			4
У	7	14	7			1		8	3	2				9	1
Ф	1	1													
Х	3	4	2	2	1			1							
Ц				1							1				
Ч			7	1					1			1			
Ш				1								1			
Щ				1											
Ы	3	9	4	1		16		1	2						
Ь	1	11	3					1	4				1	3	1
Э		1	9												
Ю	1	1	7				1	1		4					
Я	3	6	10			2	1	4	1	1			1	1	1

Таблица В2

Таблица частот біграм англійської мови

	A	B	C	D	E	F	G	H	I	J	K	L	M
A	4	20	28	52	2	11	28	4	32	4	6	62	23
B	13	0	0	0	55	0	0	0	8	2	0	22	0
C	32	0	7	1	69	0	0	33	17	0	10	9	1
D	40	16	9	5	65	18	3	9	56	0	1	4	15
E	84	20	55	125	51	40	19	16	50	1	4	55	54
F	19	3	5	1	19	21	1	3	30	2	0	11	1
G	20	4	3	2	35	1	3	15	18	0	0	5	1
H	101	1	3	0	270	5	1	6	57	0	0	0	3
I	40	7	51	23	25	9	11	3	0	0	2	38	25
J	3	0	0	0	5	0	0	0	1	0	0	0	0
K	1	0	0	0	11	0	0	0	13	0	0	0	0
L	44	2	5	12	62	7	5	2	42	1	1	53	2
M	52	14	1	0	64	0	0	3	37	0	0	0	7

	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	167	2	14	0	83	76	127	7	25	8	1	9	1
B	0	11	0	0	15	4	2	13	0	0	0	15	0
C	0	50	3	0	10	0	28	11	0	0	0	3	0
D	6	16	4	0	21	18	53	19	5	15	0	3	0
E	146	35	37	6	191	149	65	9	26	31	12	5	0
F	0	51	0	0	26	8	47	6	3	3	0	2	0
G	4	21	1	1	20	9	21	9	0	5	0	1	0
H	2	44	1	0	3	10	18	6	0	5	0	3	0
I	202	56	12	1	46	79	117	1	22	0	4	0	3
J	0	4	0	0	0	0	0	3	0	0	0	0	0
K	2	2	0	0	0	6	2	1	0	2	0	1	0

L	2	25	1	1	2	16	23	9	0	1	0	33	0
M	1	17	18	1	2	12	3	8	0	1	0	2	0

	A	B	C	D	E	F	G	H	I	J	K	L	M
N	42	10	47	122	63	19	106	12	30	1	6	6	9
O	7	12	14	17	5	95	3	5	14	0	0	19	41
P	19	1	0	0	37	0	0	4	8	0	0	15	1
Q	0	0	0	0	0	0	0	0	0	0	0	0	0
R	83	8	16	23	169	4	8	8	77	1	10	5	26
S	65	9	17	9	73	13	1	47	75	3	0	7	11
T	57	22	7	1	76	5	2	330	126	1	0	14	10
U	11	5	9	6	9	1	6	0	9	0	1	19	5
V	7	0	0	0	72	0	0	0	28	0	0	0	0
W	36	1	1	0	38	0	0	33	36	0	0	4	1
X	1	0	2	0	0	1	0	0	3	0	0	0	0
Y	14	5	4	2	7	12	2	6	10	0	0	3	7
Z	1	0	0	0	4	0	0	0	0	0	0	0	0

	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
N	7	54	7	1	7	44	124	6	1	15	0	12	0
O	134	13	23	0	91	23	42	55	16	28	0	4	1
P	0	27	9	0	33	14	7	6	0	0	0	0	0
Q	0	0	0	0	0	0	0	17	0	0	0	0	0
R	16	60	4	0	24	37	55	6	11	4	0	28	0
S	12	56	17	6	9	48	116	35	1	28	0	4	0
T	6	79	7	0	49	50	56	21	2	27	0	24	0
U	31	1	15	0	47	39	31	0	3	0	0	0	0
V	0	5	0	0	0	0	0	0	0	0	0	3	0
W	8	15	0	0	0	4	2	0	0	1	0	0	0
X	0	1	5	0	0	0	3	0	0	1	0	0	0
Y	5	17	3	0	4	16	30	0	0	5	0	0	0
Z	0	0	0	0	0	0	0	0	0	0	0	0	0

ДОДАТОК Г

Таблиця Г1

Сполученість букв української мови

Голосні	Приго- лосні	Ліворуч	Буква	Праворуч	Голосні	Приго- лосні
0,012	0,988	З,Т,Р,В,Н	А	__,Н,Л,Т,М	0,064	0,936
0,807	0,193	І,З,А,__,О	Б	У,О,А,Е,І	0,750	0,250
0,801	0,199	А,И,І,О,__	В	__,И,А,І,О	0,772	0,228
0,906	0,094	У,Е,А,__,О	Г	О,А,Р,Л,І	0,823	0,177
0,500	0,500	Б,__	Ґ	Р	0,214	0,786
0,894	0,106	У,А,І,О,__	Д	О,Н,И,А,У	0,678	0,322
0,004	0,996	Л,П,Н,Р,Т	Е	Р,Н,__,К,М	0,040	0,960
0,911	0,089	О,І,У,__,А	Є	__,Т,М,Ю,Д	0,197	0,803
0,737	0,263	__,Д,А,Е,О	Ж	Е,Н,А,__,Й	0,716	0,284
0,957	0,043	Е,А,І,О,__	З	А,__,Н,В,О	0,558	0,442
0,000	1,000	Л,Р,Т,В,Н	И	__,Х,С,М,Н	0,017	0,983
0,001	0,999	Р,Ц,__,В,Н	І	__,Д,В,Н,Ї	0,109	0,891
0,994	0,006	Ї,А,__,І,О	Ї	__,Н,Ї,Х,В	0,205	0,795
0,995	0,005	Е,А,__,І,И	Й	__,Н,С,О,Л	0,204	0,796
0,581	0,419	Е,И,А,Ь	К	О,И,А,Р,__	0,792	0,208
0,660	0,340	И,С,І,О,А	Л	И,Ь,А,О,І	0,859	0,141
0,853	0,147	Е,И,А,__,О	М	И,О,__,А,У	0,913	0,087
0,517	0,483	О,Н,Е,А,__,	Н	А,О,И,Н,І	0,800	0,200
0,011	0,989	Г,П,Р,К,Н	О	__,В,М,Р,Г	0,086	0,914
0,596	0,404	А,М,О,С,__,	П	Р,О,Е,І,Л	0,683	0,317
0,580	0,420	А,__,О,П,Е	Р	О,А,И,Е,І	0,813	0,187
0,700	0,300	І,А,И,О,__,	С	Т,Ь,Я,И,П	0,355	0,645
0,514	0,486	Ю,И,А,__,С	Т	И,А,Е,О,Ь	0,709	0,291
0,015	0,985	Т,К,М,Б,__,	У	__,К,В,Т,Д	0,063	0,937
0,416	0,584	Е,О,С,Н,__,	Ф	О,І,А,Е,Р	0,876	0,124
0,922	0,078	Е,О,__,А,И	Х	__,О,І,А,Н	0,740	0,260
0,625	0,375	О,К,И,А,__,	Ц	І,Е,Ь,Т,Я	0,832	0,168
0,886	0,114	І,А,О,И,__,	Ч	Н,А,И,Е,О	0,758	0,242
0,588	0,412	О,Р,Н,__,І	Ш	И,Е,О,Н,І	0,736	0,264
0,826	0,174	І,О,К,И,__,	Щ	О,Е,А,И,__,	1,000	0,000
0,000	1,000	Ц,Н,С,Л,Т	Ь	__,К,Н,С,О	0,142	0,858
0,640	0,360	У,П,А,Л,О	Ю	__,Т,Ч,В,Д	0,094	0,906
0,106	0,894	І,Л,__,С,Н	Я	__,К,М,Т,Н	0,030	0,970
0,676	0,324	Я,І,А,О,И	__	П,В,З,Н,С	0,160	0,840

Сполученість букв російської мови

Г	С	Слева		Справа	Г	С
3	97	л, д, к, т, в, р, н	А	л, н, с, т, р, в, к, м	12	88
80	20	я, е, у, н, а, о	Б	о, ы, е, а, р, у	81	19
68	32	я, т, а, е, н, о	В	о, а, н, ы, с, н, л, р	60	40
78	22	р, у, а, н, е, о	Г	о, а, р, л, н, в	69	31
72	28	р, я, у, а, н, е, о	Д	е, а, н, о, н, у, р, в	68	32
19	81	м, н, л, д, т, р, н	Е	н, т, р, с, л, в, м, н	12	88
83	17	р, е, н, а, у, о	Ж	е, н, д, а, н	71	29
89	11	о, е, а, н	З	а, н, в, о, м, д	51	49
27	73	р, т, м, н, о, л, н	И	с, н, в, н, е, м, к, з	25	75
55	45	ь, в, е, о, а, н, с	К	о, а, н, р, у, т, л, е	73	27
77	23	г, в, ы, н, е, о, а	Л	н, е, о, а, ь, я, ю, у	75	25
80	20	я, ы, а, н, е, о	М	н, е, о, у, а, н, п, ы	73	27
55	45	д, ь, н, о, а, н, е	Н	о, а, н, е, ы, н, у	80	20
11	89	р, п, к, в, т, н	О	в, с, т, р, н, д, н, м	15	85
65	35	в, с, у, а, н, е, о	П	о, р, е, а, у, н, л	68	32
55	45	н, к, т, а, п, о, е	Р	а, е, о, н, у, я, ы, н	80	20
69	31	с, т, в, а, е, н, о	С	т, к, о, я, е, ь, с, н	32	68
57	43	ч, у, н, а, е, о, с	Т	о, а, е, н, ь, в, р, с	63	37
15	85	п, т, к, д, н, м, р	У	т, п, с, д, н, ю, ж	16	84
70	30	н, а, е, о, н	Ф	н, е, о, а, е, о, а	81	19
90	10	у, е, о, а, ы, н	Х	о, н, с, н, в, п, р	43	57
69	31	е, ю, н, а, н	Ц	н, е, а, ы	93	7
82	18	е, а, у, н, о	Ч	е, н, т, н	66	34
67	33	ь, у, ы, е, о, а, н, в	Ш	е, н, н, а, о, л	68	32
84	16	е, б, а, я, ю	Щ	е, н, а	97	3
0	100	м, р, т, с, б, в, н	Ы	л, х, е, м, н, в, с, н	56	44
0	100	н, с, т, л	Ь	н, к, в, п, с, е, о, н	24	76
14	86	с, ы, м, л, д, т, р, н	Э	н, т, р, с, к	0	100
58	42	ь, о, а, н, л, у	Ю	д, т, ш, ц, н, п	11	89
43	57	о, н, р, л, а, н, с	Я	в, с, т, п, д, к, м, л	16	84

Сполученість букв англійської мови

Г	С	Слева		Справа	Г	С
19	81	l,c,d,m,n,s,w,t,r,e,h	A	n,t,s,r,l,d,c,m	6	94
55	45	y,b,n,t,u,d,o,s,a,e	B	e,l,u,o,a,y,b,r	70	30

61	39	u,o,s,n,a,i,l,e	C	h,o,e,a,i,t,r,l,k	59	41
52	48	r,i,l,a,n,e	D	e,i,t,a,o,u	54	46
8	92	c,b,e,m,v,d,s,l,n,t,r,h	E	r,d,s,n,a,t,m,e,c,o	21	79
69	31	s,n,f,d,a,i,e,o	F	t,o,e,i,a,r,f,u	52	48
36	64	o,d,u,r,i,e,a,n	G	e,h,o,r,a,t,f,w,i,s	42	58
7	93	g,e,w,s,c,t	H	e,a,i,o	90	10
13	87	f,m,w,e,n,l,d,s,r,h,t	I	n,t,s,o,c,r,e,m,a,l	17	83
28	72	y,w,t,s,n,e,c,b,a,c	J	u,o,a,e,m,w	88	12
53	47	y,u,i,n,a,r,o,c	K	e,i,n,a,t,s	68	32
52	48	m,p,t,i,b,u,o,e,l,a	L	e,i,y,o,a,d,u	65	35
69	31	s,d,m,r,i,a,o,e	M	e,a,o,i,p,m	71	29
89	11	u,e,o,a,i	N	d,t,g,e,a,s,o,i,c	32	68
21	79	o,d,l,p,h,n,e,c,f,s,i,r,t	O	n,f,r,u,t,m,l,s,w,o	18	82
47	53	r,l,t,n,i,p,m,a,o,u,e,s	P	o,e,a,r,l,u,p,t,i,s	59	41
20	80	o,n,l,e,d,r,s	Q	u	100	0
70	30	p,i,u,t,a,o,e	R	e,o,a,t,i,s,y	61	39
48	52	d,t,o,u,r,n,s,i,a,e	S	t,e,o,i,s,a,h,p,u	41	59
43	57	u,o,d,t,f,e,i,n,s,a	T	h,i,o,e,a,t,r	38	62
35	65	p,f,t,l,b,d,s,o	И	n,s,t,r,l,p,b,c	8	92
88	12	r,u,o,a,i,e	V	e,i,o,a	99	1
48	52	g,d,y,n,s,t,o,e	W	a,h,i,e,o,n	80	20
95	5	u,n,i,e	X	p,t,i,a,u,c,k,o	38	62
24	76	b,n,a,t,e,r,l	Y	a,o,s,t,w,h,i,e,d,m	38	62
88	12	o,n,a,i	Z	e,i,w	86	14

ДОДАТОК Д

Таблиця Д1

Ймовірності характеристики англійського алфавіту

Буква	A	B	C	D	E	F	G	H	I
Символ j	0	1	2	3	4	5	6	7	8
Log₂ p(j)	-3.6	-6.0	-5.0	-4.7	-3.0	-5.4	-5.7	-4.2	-3.8
Буква	J	K	L	M	N	O	P	Q	R
Символ j	9	10	11	12	13	14	15	16	17
Log₂ p(j)	-9.3	-7.2	-4.6	-5.3	-3.8	-3.7	-5.6	-9.8	-4.0
Буква	S	T	U	V	W	X	Y	Z	
Символ j	18	19	20	21	2	23	24	25	
Log₂ p(j)	-3.9	-3.4	-5.2	-6.7	-5.7	-9.0	-5.9	-10.1	