18. Коваленко А.С. Визначення понятійного апарату та напрямів досліджень для синтезу систем технічної діагностики інтегрованих інформаційних систем / А.С. Коваленко, О.А. Смірнов, О.В. Коваленко // Комп'ютерне моделювання у наукоємних технологіях (КМНТ-2014): наук.-техн. конф. з міжнар. участю, 28-31 трав. 2014 р., м. Харків: зб. наук. праць. – Харків: ХНУ, 2014. – С. 190-193.
19. Коваленко А.С. Дослідження елементів інтегрованої інформаційної системи / А.С. Коваленко, О.А. Смірнов, О.В. Коваленко // Комбінаторні конфігурації та їх застосування: XVII між нар. наук.-практ. сем., 17-18 квіт. 2015 р., м. Кіровоград: зб. тез – Кіровоград: КНТУ, 2015. – С. 5.
20. Коваленко А.С. Метод автоматизованої перевірки результатів вимірювання параметрів об'єкті в інтегрованої інформаційної системи / А.С. Коваленко, О.А. Смірнов, О.В. Коваленко // Стратегія якості у промисловості і освіті: XI міжнар. конф., 1 – 5 черв. 2015 р., м. Варна, Болгарія.: зб. матер. – Варна: ТУВ, 2015. – С. 423-426.

УДК 004

В. Коваленко, магістр гр. КІ-21М-1,4,
Центральноукраїнський національний технічний університет

ДОСЛІДЖЕННЯ ТА ПРОГРАМНА РЕАЛІЗАЦІЯ СИСТЕМИ ДЛЯ ЗАБЕЗПЕЧЕННЯ КОНФІДЕНЦІЙНОСТІ ІНФОРМАЦІЇ З'ЄМНИХ НОСІЇВ

У статті програмне забезпечення, яке призначено для системи для забезпечення конфіденційності інформації з'ємних носіїв. Метою розробки є дослідження та програмна реалізація системи для забезпечення конфіденційності інформації з'ємних носіїв. Об'єктом дослідження є процес для забезпечення конфіденційності інформації з'ємних носіїв. Предметом дослідження є методи для забезпечення конфіденційності інформації з'ємних носіїв. Методи дослідження базуються на методах захисту інформації, методах математичної статистики, методах розробки програмного забезпечення. Результат роботи – програмна реалізація системи для забезпечення конфіденційності інформації з'ємних носіїв. В процесі роботи над програмною моделлю виконано аналіз існуючих апаратних та програмних засобів. В повній мірі описані всі компоненти розробленого програмного забезпечення.

комп'ютерна інженерія, захисту доступу, з'ємних носії

Постановка проблеми. Широке застосування комп'ютерних технологій і постійне збільшення обсягу інформаційних потоків викликає постійний ріст інтересу до криптографії. Останнім часом збільшується роль програмних засобів захисту інформації, просто модернізуємих, не потребуючих великих фінансових витрат у порівнянні з апаратними криптосистемами [1-5]. Сучасні методи шифрування гарантують практично абсолютний захист даних, але завжди залишається проблема надійності їхньої реалізації. Іншою важливою проблемою застосування криптографії є протиріччя між бажанням громадян захистити свою інформацію й прагненням державних спецслужб мати можливість доступу до деякої інформації для припинення незаконної діяльності [6]. Надзвичайно важко знайти незаперечно оптимальне рішення цієї проблеми. Як оцінити співвідношення втрат законослухняних громадян і організацій від незаконного використання їхньої інформації й збитків держави від неможливості одержання доступу до захищеної інформації окремих груп, що приховують свою незаконну діяльність? Чи можна гарантированно не допустити незаконне використання криптоалгоритмів особами, які порушують і інші закони? Крім того, завжди існують способи схованого зберігання й передачі інформації. Хоча стримування відкритих досліджень в області криптографії й криптоаналізу є найпростішим шляхом, але це принесе значний негативний ефект. Застосування ненадійних засобів не захистить користувачів, але викличе поширення комп'ютерних злочинів, навпроти, виявлення своєчасне виявлення помилок у системах захисту інформації дозволить запобігти збитку [5-

8]. У цей час особливо актуальною стала оцінка вже використовуваних криптоалгоритмів. Завдання визначення ефективності засобів захисту найчастіше більше трудомістка, чим їхня розробка, вимагає наявності спеціальних знань і, як правило, більше високої кваліфікації, ніж завдання розробки. Ці обставини приводять до того, що на ринку з'являється безліч засобів криптографічного захисту інформації, про які ніхто не може сказати нічого певного. При цьому розроблювачі тримають криптоалгоритм (як показує практика, часто нестійкий) у секреті. Однак задача точного визначення даного криптоалгоритму не можуть бути гарантированно складною хоча б тому, що він відомий розроблювачам. Крім того, якщо порушник знайшов спосіб подолання захисту, то не в його інтересах про це заявляти. Тому суспільству повинне бути вигідно відкрите обговорення безпеки систем захисту інформації масового застосування, а приховання розроблювачами криптоалгоритму повинне бути неприпустимим.

На сьогоднішній день існують добре відомі й апробовані криптоалгоритми (як із симетричними, так і несиметричними ключами), криптостійкость яких або доведена математично, або заснована на необхідності рішення математично складного завдання (факторизації, дискретного логарифмування й т.п.) [7-9]. З іншого боку, у комп'ютерному світі весь час з'являється інформація про помилки або "діри" у тій або іншій програмі (у т.ч. що застосовує криптоалгоритми), або про те, що вона була зламана. Це створює недовіру, як до конкретних програм, так і до можливості взагалі захистити що-небудь криптографічними методами не тільки від спецслужб, але й від простих хакерів. Тому знання атак і дір у криптосистемах, а також розуміння причин, по яких вони мали місце, є однією з необхідних умов розробки захищених систем і їхнього використання.

У зв'язку з тим, що на даному етапі часто для переносу інформації використовуються пристрої на flash-накопичувачах, дуже актуальним є завдання їхнього захисту. Існує два підходи до захисту інформації на flash-накопичувачах: з використанням доступу до flash-накопичувача за допомогою біопараметричних характеристик і за допомогою шифрування інформації, що зберігається на flash-накопичувачі.

Аналіз останніх досліджень і публікацій. При аналізі останніх досліджень і публікацій [1-10] було виявлено певні прогалини у забезпеченні системи для забезпечення конфіденційності інформації з’ємних носіїв.

Мета й завдання дослідження. Метою роботи є дослідження та програмна реалізація системи для забезпечення конфіденційності інформації з’ємних носіїв.

Для досягнення поставленої мети визначена програма дослідження, що складається з наступних завдань:

– Огляд існуючих систем для забезпечення конфіденційності інформації з’ємних носіїв.

– Дослідження системи для забезпечення конфіденційності інформації з’ємних носіїв.

– Програмна реалізація системи для забезпечення конфіденційності інформації з’ємних носіїв.

Об’єктом дослідження є процес для забезпечення конфіденційності інформації з’ємних носіїв.

Предметом дослідження є методи для забезпечення конфіденційності інформації з’ємних носіїв.

Методи дослідження базуються на методах захисту інформації, методах математичної статистики, методах розробки програмного забезпечення.

Виклад основного матеріалу.

Опис генератора псевдовипадкових чисел

Послідовності випадкових чисел (ВЧ) є невід'ємним інструментом рішення багатьох математичних завдань, і в тому числі завдань захисту інформації. Для потокових шифрів він використовується у ролі гами для накладання на дані, які потрібно захищати. Найчастіше необхідні послідовності випадкових чисел, які рівномірно й рівноймовірно розподілені на

деякому відрізку. Більшість природних процесів є випадковими, і, відповідно до теорії математичної статистики, вони підкоряються різним законам розподілу випадкових величин. Для породження випадкових чисел, рівномірно розподілених на відрізку [0..*M*], досить для деякого випадкового процесу, що підкоряється закону рівномірного розподілу, увести міру випадкової величини. А потім послідовно проводити експерименти, вимірювати значення випадкової величини й після нормування одержувати необхідну рівномірну випадкову послідовність чисел.

Подібні методи дадуть дуже гарні статистичні результати, але зажадають колосального часу для одержання скільки-небудь довгої послідовності. На щастя, математики розробили рекурентні формули одержання псевдовипадкових чисел (ПВЧ). Назва «псевдовипадкові» обумовлена хоча б тим фактом, що якщо відомо деяке і-*е* число, то по формулі однозначно обчислимо ($i + 1$)-ий елемент послідовності. Більше того, з рекурентної природи знайденої формули виходить, що при тому самому x_0 ми одержимо при повторній генерації ту ж саму послідовність. До того ж, всі арифметичні алгоритми генерації ПВЧ періодичні, тобто існує деяке $p \in N$, для якого виконується $x_i = x_{i+pn}$ при будь-якому натуральному n. Тому будь-який ГПВЧ повинен бути ініціалізований випадковою величиною, якимось зовнішнім джерелом випадкових значень. Такими «апаратними» у ПК генераторами можуть виступати, наприклад, електричні шуми в напівпровідникових пристроях, а також поточна кількість виконаних процесором тактів або поточне значення часу.

Недоліки ГПВЧ

- порівняно короткий період генеруємої послідовності.
- залежність між сусідніми послідовними значеннями.
- нерівномірність розподілу значень, у тому числі через різний ступінь.

Ініціалізація ГПВЧ

Читання поточного значення мілісекунд

Як уже було сказано вище, рекурентну формулу обчислення і-*го* члена ПВЧ при реалізації на ПК можна ініціалізувати значенням мілісекунд поточного часу в момент запуску програми. Для цього можна викликати функцію «2Ch» переривання 21h операційної системи DOS і одержати «майже» випадкове число на відрізку [0..99].

Читання значення лічильника тактів процесора

Починаючи з лінійки процесорів Pentium в архітектурі x86 з'явилася інструкція, що дозволяє прочитати лічильник тактів процесора з моменту останнього скидання. Для інструкції *rdtsc* (Read Time Stamp Counter) заданий машинний код 0F 31. 64-бітне значення лічильника вертається в парі регістрів <EDX:EAX>. Якщо для ініціалізації генератора досить 32-бітного значення, то необхідно використовувати найбільше «чутливі» молодші 32 біта лічильника тактів. При використанні старого компілятора мови асемблера для звертання до 32-бітного регістра EAX знадобиться вручну проставити префікс із кодом 66*h*.

Лінійний конгруентний метод

Рекурентна формула виглядає в такий спосіб:

$$x_n = (ax_{n-1} + c) \bmod m. \qquad (1)$$

Період породжуваної послідовності не перевищує m. Природно, що від вибору параметрів a, c, m і значення першого члена x_0 істотно залежать основні властивості породжуваної послідовності. Довжина періоду буде максимальною (рівна m) тільки в тому випадку, коли:

- НЗД(c, m) = 1 (тобто c і m взаємно прості)
- $a - 1$ кратно всім простим дільникам m
- якщо m кратно 4, то й $(a - 1) \bmod 4 = 0$

Алгоритм Блюма-блюма-Шуба

Алгоритм ГПВЧ, стійкий до зворотних перетворень. Основна рекурентна формула алгоритму:

$$x_n = (x_{n-1})^2 \bmod pq, \qquad z_n = parity\ (x_n), \qquad (2)$$

де p і q – два великих простих числа. Для підвищення якості одержуваної послідовності на черговому кроці вибираються не всі біти x_n, а тільки молодші, або навіть тільки біт парності. З отриманих «випадкових біт» формуються двійкові ПВЧ довільної розрядності. Однієї з особливостей обчислювальної формули є наскрізна можливість обчислити x_n без генерації всіх попередніх членів послідовності.

$$x_n = (x_0)^{2n \bmod (p-1)(q-1)} \bmod pq, \quad (3)$$

Даний алгоритм більше вимогливий до обчислювальних ресурсів, але, з іншого боку, має гарні статистичні характеристики.

Алгоритм xor-shift

Професором університету Флориди Джорджем Марсаглією був розроблений дуже швидкий генератор, що був названий « xor-shift».

$$x = 3456789,\ y = 62436069,\ z = 1288629,\ w = 675123 \quad (4)$$

$$t = (x \text{ xor } (x \text{ shl } 11)),\quad x = y,\quad y = z,\quad z = w, \quad (5)$$

$$\text{xor128} = w = (w \text{ xor } (w \text{ shr } 19)) \text{ xor } (t \text{ xor } (t \text{ shr } 8))) \quad (6)$$

Існує комбінація даного алгоритму з лінійним конгруентним алгоритмом і алгоритмом Фібоначчі із запізнюваннями.

$$x = 123456789,\quad y = 362436000,\quad z = 521288629,\quad c = 7654321, \quad (7)$$

$$x = \text{int64}(69069)\cdot x + 12345,\quad y = y \text{ xor } (y \text{ shl } 13), \quad (8)$$

$$y = y \text{ xor } (y \text{ shr } 17),\quad y = y \text{ xor } (y \text{ shl } 5), \quad (9)$$

$$t = \text{int64}(698769069)\cdot z + c,\quad c = t \text{ shr } 32, \quad (10)$$

$$z = t,\quad \text{Random} = x + y + z, \quad (11)$$

Опис алгоритму шифрування

У якості криптоалгоритму спрямованого на захист інформації, яка утримується в flash-пристрої візьмемо алгоритм RC4. Розглянутий нами криптоалгоритм RC4 відноситься до класу потокових шифрів, які останнім часом стали популярними завдяки високій швидкості роботи. Потокові шифри перетворять відкритий текст у шифротекст по одному біті за операцію. Генератор потоку ключів (іноді називаний генератором із ключем, що біжить) видає потік біт: k_1, k_2, k_3, ..., k_i. Цей потік ключів і потік біт відкритого тексту, p_1, p_2, p_3, ..., p_i, піддаються операції “або, що виключає", і в результаті виходить потік біт шифротексту.

$$c_i = p_i \oplus k_i \quad (12)$$

При дешифруванні операція XOR виконується над бітами шифротексту й тим же самим потоком ключів для відновлення біт відкритого тексту.

$$p_i = c_i \oplus k_i \quad (13)$$

Безпека системи повністю залежить від властивостей генератора потоку ключів. Генератор потоку ключів створює бітовий потік, що схожий на випадковий, але в дійсності детермінований і може бути безпомилково відтворений при дешифруванні. Чим ближче вихід генератора потоку ключів до випадкового, тим більше часу буде потрібно для взлому шифру.

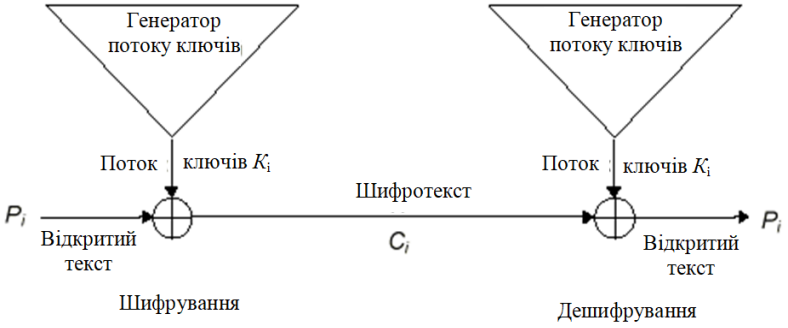

Рисунок 1 – Потоковий шифр

Для всіх потокових шифрів використовуються ключі. Вихід генератора потоку ключів є функцією ключа. Тепер, якщо одержати пару відкритий текст/шифротекст, то можна читати тільки ті повідомлення, які зашифровані тим же ключем.

Потокові шифри особливо корисні для шифрування нескінченних потоків комунікаційного трафіку, наприклад, при записі данних на flash-пам'ять.

Генератор потоку ключів складається із трьох основних частин:

– Внутрішній стан описує поточний стан генератора потоку ключів.

– Два генератори потоку ключів, з однаковим ключем і однаковим внутрішнім станом, видають однакові потоки ключів.

– Функція виходу по внутрішньому стану генерує біт потоку ключів.

– Функція наступного стану по внутрішньому стану генерує новий внутрішній стан.

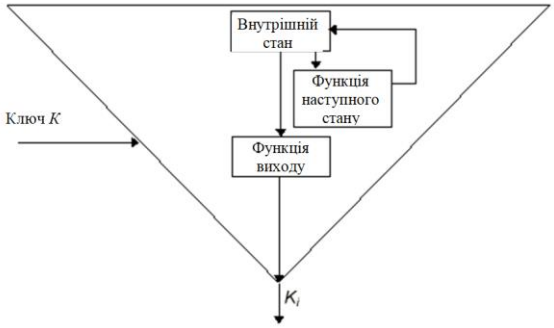

Рисунок 2 – Пристрій генератора потоку ключів

Криптоалгоритм RC4 відноситься до так званих шифрів, що самосинхронізуються. У потокових шифрах, що самосинхронізуються, кожний біт потоку ключів є функцією фіксованого числа попередніх біт шифротексту. Військові називають цей шифр автоключом шифротексту.

Потоковий шифр, що самосинхронізується, показаний на рисунку 3. Внутрішній стан є функцією попередніх n біт шифротексту. Криптографічно складною є вихідна функція, що використовує внутрішній стан для генерації біта потоку ключів.

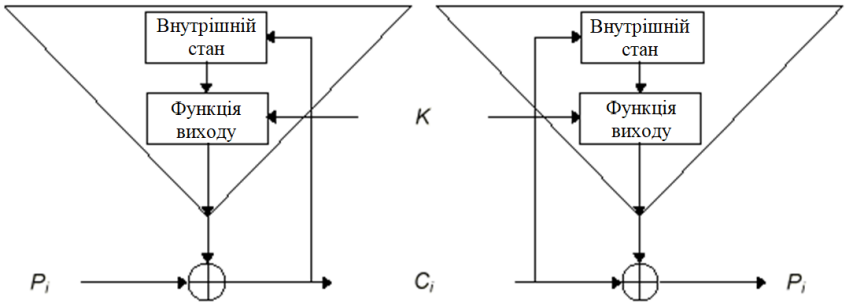

Рисунок 3 – Генератор потоку ключів, що самосинхронізується

Так як внутрішній стан повністю залежить від попередніх *n* шифротексту, дешифруючий генератор потоку ключів автоматично синхронізується з генератором, що шифрує, потоку ключів, прийнявши *n* біт шифротексту. В інтелектуальних реалізаціях цього режиму кожне повідомлення починається випадковим заголовком довжиною *n* біт.

Цей заголовок шифрується, передається й потім розшифровується. Розшифровка буде неправильною, але після цих *n* біт обидва генератори потоку ключів будуть синхронізовані.

Слабкою стороною потокового шифру, що самосинхронізується, є поширення помилки. Для кожного біта шифротексту, зіпсованого при передачі, дешифруючий генератор потоку ключів видає n неправильних біт потоку ключів. Отже, кожному неправильному біту шифротексту відповідають n помилок у відкритому тексті, поки зіпсований біт не перестане впливати на внутрішній стан.

Алгоритм RC4 і його криптоаналіз

Істотне підвищення продуктивності мікропроцесорів в 80-і роки викликало в криптографії посилення інтересу до програмних методів реалізації криптоалгоритмів як можливій альтернативі апаратним схемам на регістрах зрушення.

Одним з найперших подібних криптоалгоритмів, що получили широке поширення, став RC4. Алгоритм RC4 – це потоковий шифр зі змінною довжиною ключа.

Він володіє наступними властивостями:

- адаптивністю для апаратних засобів і програмного забезпечення, що означає використання в ньому тільки примітивних обчислювальних операцій, звичайно присутніх на типових мікропроцесорах;
- алгоритм швидкий, тобто в базисних обчислювальних операціях оператори працюють на повних словах даних;
- адаптивністю на процесори різних довжин слова;
- компактністю в термінах розміру коду, і особливо зручний для процесорів з побайтно-орієнтованою обробкою;
- низькою вимогою до пам'яті, що дозволяє реалізовувати алгоритм на пристроях з обмеженою пам'яттю;
- використанням циклічних зрушень, залежних від даних, з "змінним" числом;
- простотою й легкістю виконання.

У цей час алгоритм RC4 реалізований у десятках комерційних криптографічних продуктів, включаючи Lotus Notes, Apple Computer's AOCE, Oracle Secure SQL, а також є частиною специфікації стандарту стільникового зв'язка CDPD.

Криптогенератор функціонує незалежно від відкритого тексту. Генератор має підстановочну таблицю (S-бокс 8 x 8): S_0, S_1, . . ., S_{255}. Входами генератора є замінені по підстановці числа від 0 до 255, і ця підстановка є функцією від ключа змінюваної довжини. Генератор має два лічильники i і j, ініціалізуємих нульовим значенням.

Для генерації випадкового байта гами виконуються наступні операції:

$$i = (i+1) \bmod 256 \quad (14)$$

$$j = (j+S_i) \bmod 256 \quad (15)$$

$$\text{swap}\ (S_i, S_j) \quad (16)$$

$$t = (S_i+S_j) \bmod 256 \quad (17)$$

$$K = S_t \quad (18)$$

Байт K складається операцією XOR з відкритим текстом для виробітку шифротексту, або із шифротекстом для одержання байта відкритого тексту. Шифрування відбувається досить швидко – приблизно в 10 разів швидше DES-Алгоритму. Ініціалізація S-боксу настільки ж проста. На першому кроці він заповнюється лінійно: S0 = 0, S1 = 1, . . ., S255 = 255.

Потім ще один 256-байтний масив повністю заповнюється ключем, для чого ключ повторюється відповідне число раз залежно від довжини: K0, K1,...,K255. Індекс j обнуляется. Потім:

```
for (i=0; i<= 255; i++)
{
j = (j+Si+Ki) mod 256;
swap (Si , Sj);
}
```

Схема показує, що RC4 може приймати приблизно 2^{1700} ($256! * 256^2$) можливих станів. S-бох повільно змінюється в процесі роботи: параметр i забезпечує зміну кожного елемента, а j відповідає за те, щоб ці елементи змінювалися випадковим образом.

Фактично, RC4 являє собою сімейство алгоритмів, що задаються параметром n, що є позитивним цілим з рекомендованим типовим значенням $n = 8$.

Внутрішній стан генератора RC4 у момент часу t складається з таблиці $S_t = (S_t(L))_{t=0}^{n2\ -1}$, що містить $2^{n\ n-6}$ітних слів і із двох n-бітних слів-покажчиків i_t і j_t. Таким

чином, розмір внутрішньої пам'яті становить $M = n2^n + 2n$ біт. Нехай вихідне n-бітне слово генератора в момент t позначається як Z_t.

Нехай початкові значення $i_0 = j_0 = 0$. Тоді функція наступного стану й функція виходу RC4 для кожного $t > 1$ задається наступними співвідношеннями:

$$i_t = i_{t-1} + 1 \quad (19)$$

$$j_t = j_{t-1} + S_{t-1}(i_t) \quad (20)$$

$$S_t(i_t) = S_{t-1}(j_t) \quad (21)$$

$$S_t(j_t) = S_{t-1}(i_t) \quad (22)$$

$$Z_t = S_t(S_t(i_t) + S_t(j_t)), \quad (23)$$

де всі додавання виконуються по модул. 2^n. Мається на увазі, що всі слова, крім тих, які піддаються перестановці, залишаються тими ж самими. Вихідна послідовність n-бітних слів позначається як $Z_t = (Z_t)_{t=1}^{\infty}$.Початкова таблиця S_0 задається в термінах ключової послідовності:

$$K = (K_L)_{t=0}^{2n-1} \quad (24)$$

з використанням тієї ж самої функції наступного стану, починаючи від таблиці одиничної підстановки $(L)^{2n}{}_{L=0}{}^{2n-1}$. Більш строго, нехай $j_0 = 0$ і для кожного $1 \le t \le 2^n$ обчислюється $j_t = (j_t - 1 + S_{t-1}(t\text{-}1) + K_{t-1}) \bmod 2n$, а потім переставляються місцями $S_{t-1}(t\text{-}1)$ і $S_{t-1}(j_t)$.

На останньому кроці породжується таблиця, що представляє S_0. Ключова послідовність K складається із секретного ключа, що можливо повторюється, і випадкового ключа, переданого у відкритому виді з метою ресинхронизації.

До останнього часу у відкритій літературі практично не було публікацій по криптоаналізу алгоритму RC4. Компанія RSA Data Security оголосила, що шифр має імунітет до методів лінійного й диференціального криптоаналізу, високо не лінієн і не схоже, щоб у нього були короткі цикли.

Відзначається, що для послідовностей, генеруємих RC4, не підходять методи статистичного аналізу. Але, з іншого боку, для блоків, розмір яких перевищує M (розмір внутрішньої пам'яті генератора), завжди існує лінійна статистична слабість або так звана "лінійна модель". Таку модель можна ефективно визначати за допомогою методу апроксимації лінійною послідовною схемою. Лінійна статистична слабість – це лінійне співвідношення між бітами гами, що виконується з імовірністю, що відрізняється від 1/2.

За допомогою методу АЛПС були виведені лінійні моделі для RC4. Метод АЛПС полягає в знаходженні й рішенні послідовної лінійної схеми, що апроксимує генератор гами й приводить до лінійних моделей з відносно великим кореляційним коефіцієнтом c, де ймовірність відповідного лінійного співвідношення між бітами гами становить $(1 + c)/2$. При аналізі використовувалася техніка двійкових похідних. Нехай $Z = (Z_t)_{t=1}^{\infty}$ позначає послідовність самих молодших біт слів виходу RC4, і нехай $Z' = (Z'_t = Z_t + Z_{t+1})_{t=1}^{\infty}$ і $Z'' = (Z''_t = Z_t + Z_{t+2})_{t=1}^{\infty}$ позначають її перші й другу двійкові похідні, відповідно. Показано, що Z' не корелює ні з 1, ні з 0, але Z'' корелює з 1 з кореляційним коефіцієнтом, близьким до $15^* 2\text{-}3^n$ при великих $2n$, де n – довжина ключа. Оскільки довжина вихідної послідовності, необхідна для виявлення статистичної слабості з кореляційним коефіцієнтом c, становить O(c-2), те ця довжина дорівнює приблизно $64^n /225$. Наприклад, якщо $n = 8$, як рекомендується в більшості додатків, те необхідна довжина близька до 2^{40}.

Результати комп'ютерних експериментів погодяться з теоретичними пророкуваннями. Оскільки результуючий коефіцієнт кореляції істотно перевищує величину $2^{M/2}$, те встановлену лінійну модель варто розглядати як статистичну слабість генератора, принаймні в теоретичному аспекті.

Був проведений криптоаналіз узагальненої схеми вузла, що комбінує, з довільним розміром пам'яті. Досліджено кореляційні властивості таких вузлів, обґрунтовані нові конструктивні критерії, пропоновані до схем подібного типу.

Розроблено ефективний метод апроксимації лінійною послідовною схемою для побудови лінійних функцій від входу й виходу з порівняно більшим коефіцієнтом взаємної

кореляції. Це практична процедура, що дозволяє з високою ймовірністю знаходити пари взаємно коррельованих лінійних функцій (від якнайбільше $M + 1$ послідовних вихідних біт і якнайбільше $M + 1$ послідовних векторів входу) з порівняно більшими коефіцієнтами кореляції. Метод АЛПС складається в завданні й рішенні лінійної послідовної схеми (ЛПС), що апроксимує вузол, що комбінує, з пам'яттю. Ця ЛПС має додаткові незбалансовані входи й заснована на лінійних апроксимаціях функції виходу й всіх компонентів функції наступного стану. Лінійна апроксимація булевої функції – це будь-яка лінійна функція, з якою задана булева функція скорельована. Описаний метод застосуємо до довільних вузлів, що комбінують, з пам'яттю без обмежень на функції виходу й наступний стан.

Спочатку відшукуються лінійні апроксимації функції виходу f і кожної з функцій-компонентів функції наступного стану F. Це еквівалентно вираженню кожної із цих $M + 1$ функцій у вигляді суми лінійної функції й незбалансованої функції. Якщо підлягаючої декомпозиції функція вже несбалансована, то можна вибрати константно-нульову лінійну функцію. Якщо підлягаюча декомпозиції функція статистично незалежна від деякої підмножини змінних, то кожна лінійна апроксимація з необхідністю повинна задіяти принаймні одну зі змінних цієї підмножини. Основна вимога – щоб відповідні кореляційні коефіцієнти відрізнялися від нуля. Також бажано, щоб вибиралися лінійні апроксимації з кореляційними коефіцієнтами, абсолютні значення яких близькі до максимального. Кореляційні коефіцієнти можна визначати за допомогою техніки перетворення Уолша.

На наступному кроці, одержавши лінійні апроксимації, у матричній формі записують базові рівняння вузла, що комбінує, з пам'яттю

$$S_{t+1} = A \cdot S_t + B \cdot X_t + \Delta(X_t, S_t),\ t \geq 0, \tag{25}$$

$$y_t = C \cdot S_t + D \cdot X_t + \varepsilon(X_t, S_t),\ t \geq 0, \tag{26}$$

де вектори розглядаються як матриці-стовпці; A, B, C, D – двійкові матриці; а ε і кожний компонент в $D = (d_1, \ldots, d)$ – незбалансовані булеві функції, іменовані функціями шуму. Основна ідея полягає в тому, щоб розглядати $\{\varepsilon(X_t, S_t)\}_{t=0}^{\infty}$ і $\{\delta(X_t, S_t)\}_{t=0}^{\infty}$, $1 \leq i \leq M$, як вхідні послідовності, так що останні рівняння виявляються задаючими неавтономну лінійну машину з кінцевим числом станів або ЛПС, іменовану АЛПС вузла, що комбінує, з пам'яттю. Тоді можна вирішувати цю ЛПС із використанням техніки виробляючих функцій (D-перетворень). Зокрема, нехай S, X, Δ, ε, у позначають виробляючі функції від змінної z для послідовностей $\{S_t\}$, $\{X_t\}$, $\Delta(X_t, S_t)\}$, $\varepsilon(X_t, S_t)\}$, $y_t\}$, відповідно. Тоді рівняння зводяться до виду:

$$y = \left(D - \frac{C \cdot adj(zA - I)B}{\det(zA - I)} \right) X - \frac{C \cdot adj(zA - I)}{\det(zA - I)} (z\Delta + S_0) + \varepsilon \tag{27}$$

де I – одинична матриця, $\det(z - I) = \phi(z)$, $\phi(0) = 1$, – багаточлен, зворотний до характеристичного багаточлена матриці переходів A ступеня, що не перевищує ранг A ($\leq M$); а елементи (приєднаної) матриці $\mathrm{adj}(z - I)$ – це поліноми від z ступеня не більше M-1. Обчислювальна складність для відшукання такого рішення становить $O(M^3(N+1))$. В іншому виді рішення можна переписати як

$$y = \frac{1}{\varphi(z)} \sum_{i=1}^{N} g_i(z) x_i + \frac{1}{\varphi(z)} \sum_{j=1}^{M} h_j(z)(z\delta_j + s_{j0}) + \varepsilon \tag{28}$$

де x_i і δ_j позначають виробляючі функції для $\{x_{it}\}$ і $\{\delta_j(X_t, S_t)\}$, а ступеня поліномів $g_i(z)$ і $h_j(z)$ якнайбільше рівні M і M-1, $1 \leq i \leq N$, $1 \leq j \leq M$, відповідно. Взявши

$$\varphi(z) = \sum_{k=0}^{M} \varphi_k z^k, \quad g_i(z) = \sum_{k=0}^{M} g_{ik} z^k, \quad h_j(z) = \sum_{k=0}^{M-1} h_{jk} z^k \tag{29}$$

рішення можна перетворити до виду:

$$\sum_{k=0}^{M} \varphi_k y_{l-k} = \sum_{i=1}^{N} \sum_{k=0}^{M} g_{ik} x_{i,l-k} + e\left(X_i^{M+1}, S_{l-M}\right), \qquad t \geq M, \tag{30}$$

$$e\left(X_i^{M+1}, S_{l-M}\right) = \sum_{j=1}^{N}\sum_{k=0}^{M-1} h_{jk}\delta_j\left(X_{t-1-k}, S_{t-1-k}\right) + \sum_{k=0}^{M-1}\varphi_k\varepsilon\left(X_{t-k}, S_{t-k}\right), \qquad t \geq M, \qquad (31)$$

де мається на увазі, що вектор стану S_{t-k} – це функція від (X_{t-k-1}^{M-k}, S_{t-M}) для кожного $0 \leq k \leq M$-1. Лінійні функції входу й виходу в (30) скоррельовані тоді й тільки тоді, коли функція шуму *e* незбалансована. Коефіцієнт кореляції не залежить від часу, якщо функція наступного стану збалансована. Якщо ця умова не задовольняється, то кореляційний коефіцієнт може залежати від часу, оскільки від S_t більше не потрібна збалансованість для кожного $t \geq 0$. Функція шуму *e* в (31) визначена як сума індивідуальних шумових функцій, які незбалансовані за умови, що збалансовано функцію наступного стану. Оскільки від індивідуальних шумових функцій не потрібно бути незалежними, у принципі не можна виключати можливість, що коефіцієнт кореляції *e* з константною нульовою функцією дорівнює нулю або дуже близький до цього значення.

У розглянутому випадку індивідуальні шумові функції можна трактувати як булеві функції від $n = MN + N + M$ змінних в ($X^{M+1}{}_t$, S_{t-M}). Отже, за винятком деяких особливих випадків, у загальному випадку можна з високою ймовірністю очікувати, що загальний кореляційний коефіцієнт дуже близький до добутку індивідуальних і, таким чином, відрізняється від нуля. Відповідно, метод АПЛС не тільки з високою ймовірністю дає взаємно коррельовані лінійні функції від входу й виходу, але також дозволяє оцінити значення відповідного кореляційного коефіцієнта, використовуючи незалежність або інші імовірнісні припущення. Оскільки в ідеальному випадку хотілося б одержати такі АЛПС, у яких кореляційні коефіцієнти за абсолютним значенням близькі до максимуму, те індивідуальні кореляційні коефіцієнти повинні бути великими по величині, а кількість шумових членів в (31) повинне бути маленьким.

Звичайно, ці вимоги можуть суперечити один одному. Тому гарним підходом буде повторення процедури АЛПС кілька разів, починаючи з найкращих лінійних апроксимацій для функції виходу й компонент функції наступного стану. Ця процедура може також виконуватися для всіх можливих лінійних апроксимацій, що представляється єдиним систематичним способом перевірити всі кореляції, виявлені в процесі застосування методу АЛПС. У загальному випадку є якнайбільше $(M+1)2^{M+N}$ таких лінійних апроксимацій. Однак, у принципі завжди можна перевірити всі можливі лінійні апроксимації навіть при великому *M*, оскільки в практичних реалізаціях функції виходу й наступного стану залежать від порівняно невеликої кількості змінних або ж складені з таких булевих функцій.

Із практичної точки зору дана лінійна модель може бути використана для виділення по шифротексту генератора RC4 серед інших криптосистем, а також для відновлення параметра *n*. В 2000 році була опублікована стаття присвячена статистичному аналізу потокового генератора RC4, у якій були використані результати роботи для знаходження значення компонент *S*-боксу. Приблизний час роботи цього методу становить 2^{6n}, де *n* – порція біт у вихідному потоці, довжина вихідної послідовності, необхідна для виявлення статистичної слабості, близька до 2^{30}. Отриманий результат указує на істотну слабість генератора й можливість відновити параметри *i* і *n*. *S*-бокс може приймати 2^{nk}, де n_k – число біт ключа.

Розробка структурної схеми

На рисунку 4 зображена структурна схема роботи системи. Схема розділена на три основних компоненти:

- Flash накопичувач;
- розроблена програма;
- користувач.

Коли користувач вставляє в персональний комп'ютер Flash накопичувач, відбувається розпізнання операційною системою типу пристрою й виводу меню вироблених дій.

Розроблена програма перехоплює системне повідомлення операційній системі про виклик меню вироблених дій над Flash накопичувачем і активізує власний інтерфейс програми.

Розроблена програма складається з декількох частин:

- інтерфейсу програми;
- модуля потокового шифрування RC4;
- модуля потокового дешифрування RC4;
- налаштовування програми й захисту програми.

Розроблена програма управляє процесом обміну інформацією між Flash накопичувачем і персональним комп'ютером, використовуючи потоковий алгоритм шифрування інформації RC4. Завдяки такому підходу, можливо використовувати всі існуючі на даний момент Flash накопичувачі не зупиняючись на окремих реалізаціях з підвищеними вимогами захищеності Flash накопичувача (рисунок 4).

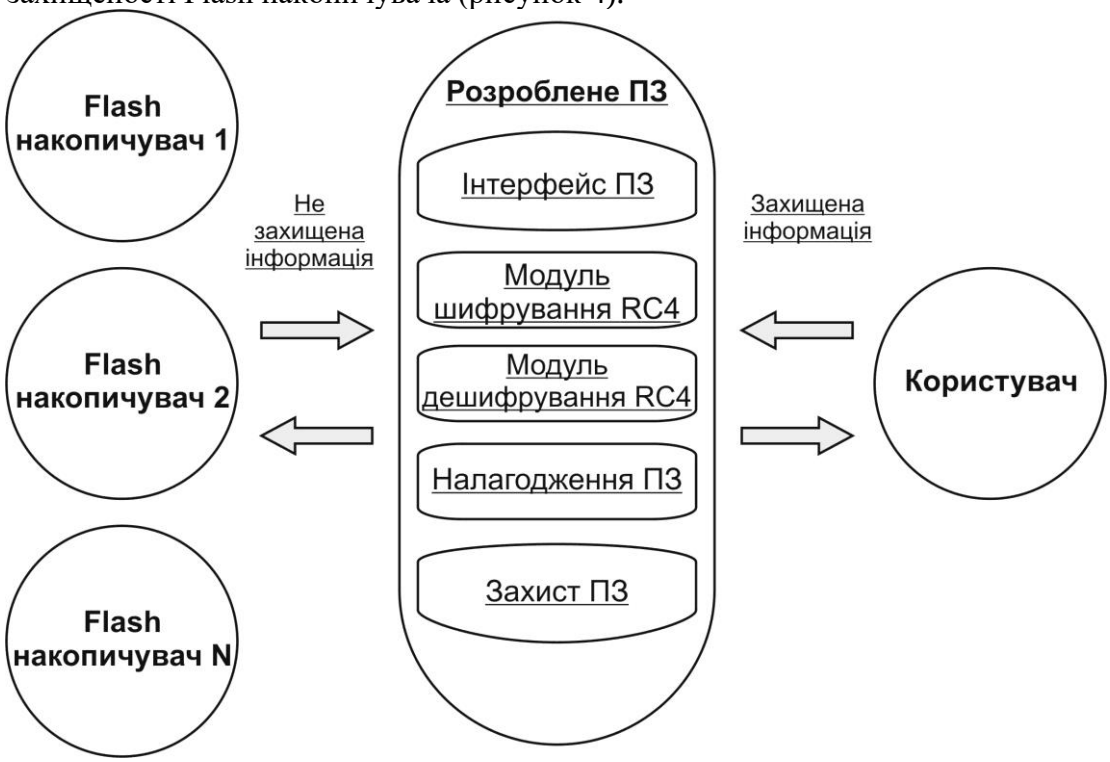

Рисунок 4 – Структурна схема роботи системи

Висновки. У статті теоретичне узагальнення й рішення наукового завдання дослідження методів для забезпечення конфіденційності інформації з’ємних носіїв.

Рішення даного завдання полягало у вирішенні наступних задач:

– Був проведений огляд існуючих систем для забезпечення конфіденційності інформації з’ємних носіїв.

– Досліджена система для забезпечення конфіденційності інформації з’ємних носіїв.

– На основі отриманих результатів досліджень створена програмна реалізація системи для забезпечення конфіденційності інформації з’ємних носіїв.

Розроблені під час виконання випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти алгоритми дозволяють успішно вирішувати завдання для забезпечення конфіденційності інформації з’ємних носіїв.

Проведено аналіз предметної галузі в ході якого були виявлені об'єкти, взаємодія яких носить істотний характер для функціональної діяльності предметної галузі, і їхні основні характеристики; побудована алгоритм і вибраний середовище розробки.

Список літератури

1. Smirnov O., Neskorodieva T., Fedorov E., Rymar P. «Neural Network Modeling Method of Transformations Data of Audit Production with Returnable Waste». CEUR Workshop Proceedings Volume 3101, 2021, Pages 192-207. (Scopus).
2. Smirnov O., Kuznetsov A., Kiian A., Kuznetsova K. «Data hiding scheme based on spread sequence addressing». CEUR Workshop Proceedings Volume 2805, 2020, Pages 44-58. (Scopus).
3. Smirnov, O., Kuznetsov, A., Potii, O., Poluyanenko, N., Stelnyk, I., Mialkovsky, D. «Combining and filtering functions in the framework of nonlinear-feedback shift register». International Journal of Computing; 2020, Volume 19, Issue 2 – Research Institute for Intelligent Computer Systems – 2020. – P. 247-256. (Scopus).
4. Smirnov O., Kuznetsov A., Kiian A., Kuznetsova T. «Non-binary constant weight coding technique». CEUR Workshop Proceedings. Volume 2740, 2020, Pages 102-114. (Scopus).
5. Smirnov O.A., Alimseitova Zh., Adranova A., Akhmetov B., Lakhno V., Zhilkishbayeva G. «Models and algorithms for ensuring functional stability and cybersecurity of virtual cloud resources». Journal of theoretical and applied information technology Vol.98. No 21, 2020, P. 3334-3346. (Scopus).
6. Smirnov O., Kuznetsov A., Arischenko A., Chepurko I., Onikiychuk A., Kuznetsova T. «Pseudorandom sequences for spread spectrum image steganography». CEUR Workshop Proceedings Volume 2654, 2020, Pages 122-131. (Scopus).
7. Smirnov O., Kuznetsov A., Kovalchuk D., Kuznetsova T. «New technique for data hiding in cover images using adaptively generated pseudorandom sequences». CEUR Workshop Proceedings Volume 2654, 2020, Pages 1-14. (Scopus).
8. Smirnov O., Lutsenko M., Kuznetsov A., Kiian A., Kuznetsova T., «Biometric cryptosystems: overview, state-of-the-art and perspective directions». Lecture Notes in Networks and Systems, vol 152. Springer, Cham. 2021, pp 66-84. (Scopus).
9. Smirnov O., Kuznetsov A., Pushkar'ov A., Serhiienko R., Babenko V., Kuznetsova T., «Representation of Cascade Codes in the Frequency Domain». In: Radivilova T., Ageyev D., Kryvinska N. (eds) Data-Centric Business and Applications. Lecture Notes on Data Engineering and Communications Technologies, vol 48. Springer, Cham. 2021. pp 557-587. (Scopus).
10. Smirnov, O., Markovets, O. Vovk, N., Turchyn, Y., «Model of informational support for social network administrators' content creation». CEUR Workshop Proceedings Volume 2616, 2020, Pages 125-136. (Scopus).
11. Smirnov, O., Drieieva, H., Drieiev, O., Polishchuk, Y., Brzhanov, R., Aleksander, M. «Method of fractal traffic generation by a model of generator on the graph». CEUR Workshop Proceedings Volume 2616, 2020, Pages 366-379. (Scopus).
12. Smirnov, O., Shekhanin, K., Kuznetsov, A., Krasnobayev, V. «Detecting Hidden Information in FAT». International Journal of Computer Network and Information Security (IJCNIS). Vol. 12, No. 3, 2020. PP.33-43. (Scopus).
13. Smirnov, O., Drieieva, H., Drieiev, O., Simakhin, V., Bondar, S., Odarchenko, R. «Managing multifractal properties of the binary sequence generated with the Markov chains», CEUR Workshop Proceedings Volume 2608, 2020, Pages 633-645. (Scopus).
14. Smirnov, O., Kuznetsov, A., Gorbacheva, L., Babenko, V., «Hiding data in images using a pseudo-random sequence», CEUR Workshop Proceedings Volume 2608, 2020, Pages 646-660., (Scopus).
15. Smirnov, O., Kuznetsov, A., Kolovanova, I., Kuznetsova, T., «Noise immunity of the algebraic geometric codes». International Journal of Computing; 2019, Volume 18, Issue 4 – Research Institute for Intelligent Computer Systems – 2019. – P. 393-407. (Scopus).
16. Smirnov, O., Ulichev, O., Meleshko, Y., Khokh, V., Goncharenko, I. «Method of Choosing Objects for Informational Influence in Social Networks during Information Campaign Based on the Analytic Hierarchy Process». CEUR Workshop Proceedings, Vol 2588, P. 215-227, 2019. (Scopus).
17. Smirnov, O., Krasnobayev, V., Yanko, A., Kuznetsova, T. «Methods of nulling numbers in the system of residual classes». CEUR Workshop Proceedings, Vol 2588, P. 90-106, 2019. (Scopus).
18. Smirnov, O., Kuznetsov, A., Kovalchuk, D., Pastukhov, M., Kuznetsova, K., Prokopovych-Tkachenko, D., «Discrete Signals with Special Correlation Properties», CEUR Workshop Proceedings Volume 2353, CEUR Workshop Proceedings 2019, Pages 618-629. (Scopus).
19. Smirnov, O., Kuznetsov, A., Kiian, A., Kuznetsova, K., Ivko, T., Prokopovych-Tkachenko, D., «Soft Decoding Based on Ordered Subsets of Verification Equations of Turbo-Productive Codes», CEUR Workshop Proceedings Volume 2353, CEUR Workshop Proceedings 2019, Pages 873-884. (Scopus).
20. Smirnov, O., Kuznetsov, A., Prokopovych-Tkachenko, D. «Hiding Data in Images Using a Pseudo-Random Sequence». ISCI'2020: Information Security in Critical Infrastructures. Collective monograph. Edited by Ivan D. Gorbenko, Victor A. Krasnobayev and Alexandr A. Kuznetsov. ASC Academic Publishing, USA, 2020. pp. 46-59. – ISBN: 978-1-7362833-0-1 (Hardback), ISBN: 978-1-7362833-1-8 (Ebook).