Theories, Architecture and Languages. Amsterdam, The Netherlands, August 8–9, 1994; Eds. M.J. Wooldridge and N.R. Jennings). Proceedings. Springer Verlag. – 1994.– P. 245–259. 5. Ferguson I.A. Integrated Control and Coordinated Behaviour: A case for Agent Models. In: Intelligent Agents. ECAI-94 Workshop on Agent Theories, Architecture and Languages. 6. R.E.Fikes and N.Nilsson. STRIPS: A new Approach to the Application of Theorem Proving to Problem Solving. Artificial Intelligence, 5(2). – 1971. – P. 189–208. 7. Wooldridge M. and Jennings N.R. Agent Theories, Architectures, and Languages: A Survey. In: Intelligent Agents. ECAI-94 Workshop on Agent Theories, Architecture and Languages. Amsterdam, The Netherlands, August 8–9, 1994; Eds. M.J. Wooldridge and N.R. Jennings. Proceedings. Springer Verlag. – 1994. – P. 3–39. 8. Dunin-Keplicz B. and Treuer J. Compositional Formal Specification of Multi-Agent System In:Intelligent Agents. ECAI-94 Workshop on Agent Theories, Architecture and Languages. Amsterdam, The Netherlands, August 8–9, 1994; Eds. M.J. Wooldridge and N.R. Jennings. Proceedings. Springer Verlag. – 1994.– P. 102–117. 9. Городецкий В.И., Грушинский М.С., Хабалов А.В. Многоагентные системы // Новости искусственного интеллекта. – 1997 – № 1. 10. Тарасов В.Б. Агенты, многоагентные системы, виртуальные сообщества: стратегическое направление в информатике и искусственном интеллекте // Новости искусственного интеллекта. – 1998. – № 2. – С. 5–63. 11. Городецкий В. И. Многоагентные системы: современное состояние исследований и перспективы применения // Труды конференции по ИИ. – 1996. – С. 36–45.

УДК 004.31, 004.056.55, 003.26

Л.М. Коркішко
Тернопільський державний економічний університет,
кафедра безпеки інформаційних технологій

БАЗОВІ ЛОГІЧНІ ЕЛЕМЕНТИ ДЛЯ КОМП'ЮТЕРНИХ ПРИСТРОЇВ ЗАХИСТУ ІНФОРМАЦІЇ

Запропоновано узагальнений параметризований метод для побудови базових логічних елементів (логічного множення та додавання), призначених для використання у комп'ютерних пристроях захисту інформації.

It is proposed a generalized parameterized method for creation of basic logical elements (logical AND and logical OR) intended for usage in computer devices for information protection.

Вступ

Із розширенням галузей застосування криптографічних перетворень у сучасному житті (банківські транзакції, смарт-карти, персональні комунікаційні пристрої тощо) значно зростає роль конфіденційності даних. Компрометування цих даних (наприклад, отримання зловмисником відомостей про ці дані) створює можливість реалізації загроз безпеки для їх власника, наприклад, його фінансових втрат. Для отримання відомостей про конфіденційні дані, які використовуються у криптографічних перетвореннях, зловмисник може використати інженерно-криптографічні атаки за побічними каналами витоку інформації [1–8]. Комп'ютерні пристрої захисту інформації уможливлюють витік інформації через сигнал про споживану потужність під час криптографічних перетворень з використанням конфіденційних даних (ключів шифрування чи цифрового підпису).

Для отримання відомостей про використовувані конфіденційні дані з сигналу про споживану потужність комп'ютерного пристрою використовують спеціальні методи аналізу, так званий "диференційний аналіз споживаної потужності" (ДАСП) [8]. Ці методи аналізу характеризуються

своєю складністю – кількістю вибірок у часі, які аналізуються. Складність аналізу характеризується його "порядком". У найпростішому випадку ДАСП першого порядку для виявлення відомостей про конфіденційні дані потребує значень сигналу про споживану потужність пристрою в один момент часу. Із збільшенням порядку ДАСП кількість вибірок аналізу і кількість даних для аналізу значно зростає.

Сучасний стан техніки збирання даних та методів їхнього аналізу дає змогу проводити ДАСП першого і, частково, другого порядку за прийнятний час. Відомі роботи з успішного отримання інформації про ключі шифрування алгоритмів AES, RSA, ECC тощо, навіть із використанням спеціальних методів захисту від атак ДАСП першого порядку [9–12]. Тому разом з розвитком математичного апарату аналізу та покращання характеристик систем збирання даних про споживану потужність, актуальним завданням є розвиток існуючих і створення нових методів захисту від ДАСП першого та вищих порядків.

Постановка задачі

Ефективним методом захисту від ДАСП першого і вищих порядків є метод "маскування" даних, які підлягають обробці, за допомогою техніки розділення таємниці [13]. При цьому базовою операцією для маскування є додавання за модулем 2 (так зване "логічне маскування"). Згадану операцію обрано для маскування даних з погляду простоти та ефективності її реалізації як у спеціалізованому обладнанні, так і за допомогою програмних засобів.

Виходячи з необхідності створення засобів виконання криптографічних перетворень, стійких до ДАСП, необхідно розв'язати задачу створення узагальненого методу маскування даних для захисту базових елементів (логічного множення і додавання) комп'ютерних пристроїв захисту інформації від ДАСП першого і вищого порядків. Вибір саме такого переліку базових елементів зумовлений тим, що операції логічного множення і додавання входять до переліку базових логічних операцій (поряд з операцією інвертування), які використовуються для виконання криптографічних алгоритмів на найнижчому рівні – як апаратно, так і програмно.

Атаки ДАСП першого та вищих порядків

Модель витоку інформації для проведення пасивних інженерно-криптографічних атак

Для проведення інженерно-криптографічних атак приймемо, що [8]:

– аргументами двомісних логічних операцій є конфіденційні дані K і відкритий текст P;

– комп'ютерний пристрій, який реалізує алгоритм криптографічного перетворення з використанням двомісних логічних операцій, уможливлює витік інформації про Хемінгову вагу результату S;

– витік інформації здійснюється через споживаний струм, а тому і через споживану пристроєм потужність;

– пристрій споживає більший струм під час обробки даних з більшою Хемінговою вагою, залежність споживаного струму від Хемінгової ваги є лінійною.

Нехай споживання потужності у момент часу j подано у вигляді $P[j]$. Для моделювання каналу витоку інформації у сигналі $P[j]$ скористаємося лінійною залежністю, запропонованою у [8]:

$$P[j] = \varepsilon \cdot d[j] + L + n, \tag{1}$$

де $d[j]$ репрезентує Хемінгову вагу результату, який отримується у момент часу j, ε – внесок у споживану потужність кожної одиниці Хемінгової ваги даних, L – споживана постійна загальна потужність, n – шум з нульовим середнім значенням.

Атака на засоби реалізації логічної операції додавання за модулем 2

Нехай j позначає момент часу, коли виконується операція додавання за модулем 2. Тоді сума $S = K \oplus P$, де K – N-бітовий невідомий доданок, P – N-бітовий відкритий текст. Розглянемо атаку, запропоновану в [8], на N-бітовий суматор за модулем 2, метою якої є визначення бітів K без відомостей про значення бітів S. Припустимо, що залежність між споживаною потужністю у

момент часу *j* і Хемінговою вагою результату, який отримується, описується виразом (1). Тоді узагальнений алгоритм атаки на реалізацію операції додавання за модулем 2 є таким:

```
Для і від 0 до N-1 {
      Для b=0 до 1 {
            Обчислити усереднене значення сигналу споживаної потужності
      Ab[j] {
            Встановити і-й біт P рівним b;
            Встановити решту бітів P у випадкові значення;
            Зібрати дані про споживану потужність пристрою;
            }
      }
     Обчислити диференційний сигнал T[j] = A0[j] – A1[j];
     Якщо T[j] > 0, то і-й біт K є "1", якщо T[j] < 0 то і-й біт K є "0";
}
```

Результативність цієї атаки грунтується на незалежності очікуваного значення Хемінгової ваги результату додавання за модулем 2 від позиції біту, який піддається аналізу.

Стійкість методу маскування до ДАСП першого і вищих порядків

Найпростіший метод маскування операндів для виконання логічної операції додавання за модулем 2 (XOR) передбачає застосування додаткової операції XOR до відкритих/конфіденційних даних та випадкових даних (маски) (рис. 1).

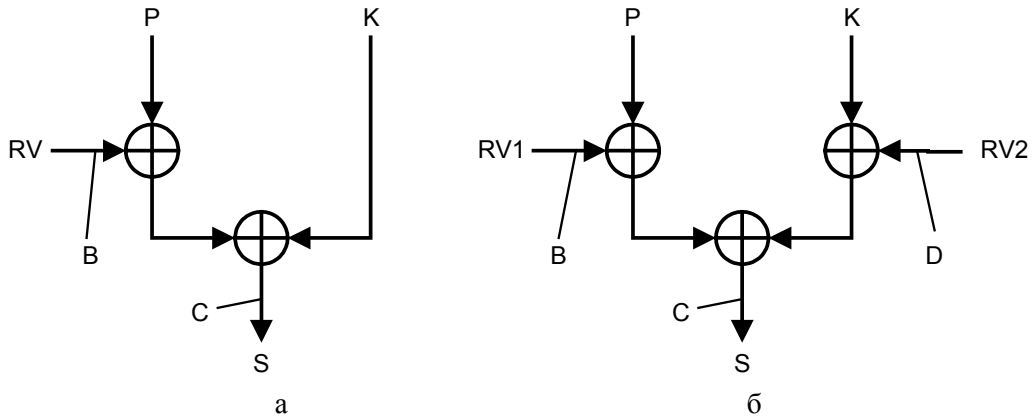

Рис. 1. Маскування операнда як захист від ДАСП:
а – маскування відкритих даних; б – додаткове маскування ключа;
P – відкритий текст; K – ключ; RV, RV1, RV2 – випадкові числа

Захист операції XOR з використанням маскування відкритого операнда (рис. 1, *а*) дає змогу уникнути атаки ДАСП першого порядку. Однак, такий захист можна подолати за допомогою ДАСП другого порядку. Для цього необхідно отримати інформацію про споживану потужність пристрою під час генерування випадкової маски *RV* (точка *B*) та протягом обчислення результату *S* (точка *C*). Маніпулюючи відкритим текстом *P* та отримуючи інформацію про Хемінгову вагу *RV* і *S* із сигналу про споживану потужність пристрою, можна встановити значення бітів ключа *K*.

Додаткове маскування ключа *K* (рис. 1, *б*) дає змогу уникнути успішної реалізації атаки ДАСП другого порядку. Однак, якщо додатково у зловмисника є доступ до інформації про Хемінгову вагу маски *RV2* (точка *D*), то маніпулюванням відкритим текстом *P* можна успішно реалізувати атаку ДАСП третього порядку та встановити значення бітів ключа *K*.

Можна запропонувати альтернативні методи захисту від атак ДАСП вищих порядків шляхом проведення операцій із більшою кількістю масок (наприклад, рис. 2, *а*, б).

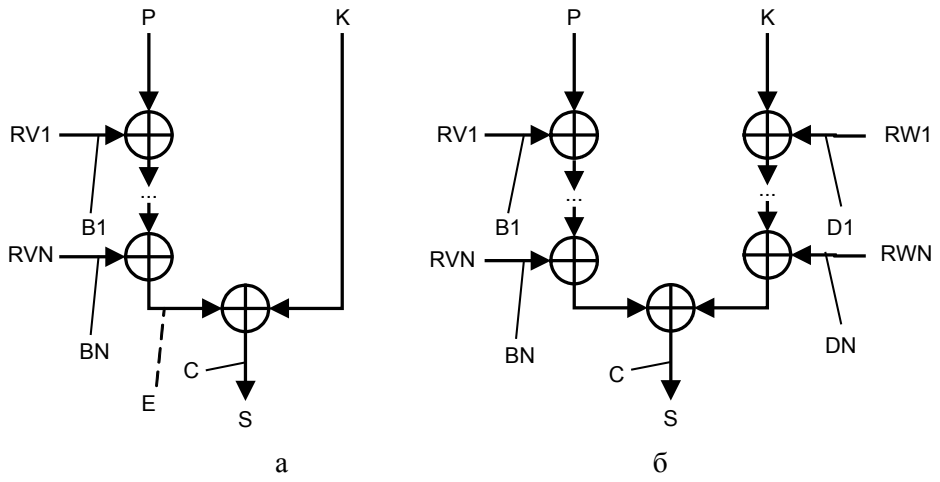

Рис. 2. Приклади захисту від атак ДАСП вищих порядків за допомогою додаткових масок: а – в плечі відкритого тексту; б – комбінована

Наведені на рис. 2 приклади захисту від атак ДАСП вищих порядків дають змогу уникнути виявлення ключа *K* при проведенні атаки ДАСП *N-1* порядку (рис. 2, *а*) чи атаки ДАСП *2N-1* порядку (рис. 2, *б*), де *N* – кількість масок. Разом з тим, якщо у зловмисника є можливість отримати інформацію про Хемінгову вагу у точці *E* (рис. 2, *а*) чи аналогічних точках (рис. 2, *б*), то наведені приклади захисту можна обійти за допомогою атаки ДАСП другого порядку. Така особливість атакування зумовлена надлишковістю інформації про Хемінгову у точках *B1,…,BN*, оскільки необхідна інформація міститься у точці *E*. З метою уникнення витоку інформації з точки *E* усі обчислення організують спеціальним чином, наприклад, шляхом рандомізування послідовності виконання цих обчислень.

Отже, успішне проведення атак ДАСП вищих порядків істотно залежить від можливих вибірок витоку інформації про Хемінгову вагу операндів, які піддаються обробці. При цьому, істотного значення набуває послідовність виконання операцій з маскування даних і роздільна здатність апаратури для отримання інформації з сигналу про споживану потужність пристрою. Тому розробники комп'ютерних засобів для реалізації криптографічних алгоритмів, стійких до атак ДАСП вищих порядків, повинні приділяти значну увагу послідовності обробки відкритих даних, масок та ключа.

Виходячи з наведених передумов атакування реалізацій криптографічних алгоритмів, розробимо алгебраїчні методи захисту двомісних логічних операцій від атак ДАСП вищих порядків. При цьому звернемо увагу на послідовність обчислення та використання проміжних результатів.

Захист двомісних логічних операцій від проведення атаки ДАСП *n*-го порядку

Для захисту двомісних логічних операцій від проведення атак ДАСП *n*-го порядку скористаємося алгебраїчним методом, запропонованим у [14]. За цим методом можна побудувати захист від проведення атак ДАСП першого порядку для операції логічного множення. При проведенні атак ДАСП вищого порядку згаданий захист не є ефективним. Тому розробимо параметризований алгебраїчний метод захисту двомісних логічних операцій (логічного множення і додавання) від проведення атак ДАСП вищого порядку, де параметром буде номер порядку атаки ДАСП.

Захист операції логічного множення від проведення атаки ДАСП n-го порядку

Для цього припустимо, що *a* і *b* є реальними даними, над якими необхідно виконати двомісну логічну операцію. Для уникнення успішного проведення атак ДАСП скористаємося представленням *a* і *b* у вигляді: $\widetilde{a} = a \oplus x$, $\widetilde{b} = b \oplus y$, де *x* і *y* є маски операндів – випадкові числа з рівномірним законом розподілу ймовірностей. Тоді обчислити добуток $a \cdot b$ необхідно так, щоб уникнути використання чи появи реальних даних у проміжних чи кінцевих результатах обчислень.

Згідно з [14], вираз для обчислення маскованого результату логічного множення двох маскованих даних, який є стійким до атаки ДАСП першого порядку, має вигляд:

$$(a \cdot b) \oplus z = \widetilde{a} \cdot \widetilde{b} \oplus (\widetilde{a} \cdot y \oplus (\widetilde{b} \cdot x \oplus (x \cdot y \oplus z))). \quad (2)$$

У (2) кінцевий результат обчислення є маскованим за допомогою нової маски z, яка має ті самі властивості, що й x і y.

Для захисту обчислень від успішних атак ДАСП другого порядку скористаємося двома масками для кожного аргументу. Тоді a і b можна подати у вигляді: $\widetilde{a} = a \oplus x_1 \oplus x_2$, $\widetilde{b} = b \oplus y_1 \oplus y_2$. Скориставшись алгебраїчними властивостями операцій логічного множення і додавання за модулем 2, отримаємо вираз:

$$a \cdot b = \widetilde{a} \cdot \widetilde{b} \oplus \widetilde{a} \cdot y_1 \oplus \widetilde{a} \cdot y_2 \oplus \widetilde{b} \cdot x_1 \oplus \widetilde{b} \cdot x_2 \oplus x_1 \cdot y_1 \oplus x_1 \cdot y_2 \oplus x_2 \cdot y_1 \oplus x_2 \cdot y_2.$$

Цей вираз модифікуємо для введення двох масок з метою отримання остаточного результату:

$$\begin{aligned}(a \cdot b) \oplus z_1 \oplus z_2 = {} & \widetilde{a} \cdot \widetilde{b} \oplus \widetilde{a} \cdot y_1 \oplus \widetilde{a} \cdot y_2 \oplus \widetilde{b} \cdot x_1 \oplus \widetilde{b} \cdot x_2 \oplus \\ & \oplus x_1 \cdot y_1 \oplus x_1 \cdot y_2 \oplus x_2 \cdot y_1 \oplus x_2 \cdot y_2 \oplus z_1 \oplus z_2\end{aligned}. \quad (3)$$

Аналогічно до (2) можна розставити дужки для виконання операцій. При цьому можна запропонувати декілька варіантів групування операцій, які відрізнятимуться порядком використання масок z_1 і z_2.

Подавши a і b у вигляді $\widetilde{a} = a \oplus x_1 \oplus x_2 \oplus x_3$, $\widetilde{b} = b \oplus y_1 \oplus y_2 \oplus y_3$, отримаємо вираз для маскованого результату:

$$\begin{aligned}(a \cdot b) \oplus z_1 \oplus z_2 \oplus z_3 = {} & \widetilde{a} \cdot \widetilde{b} \oplus \widetilde{a} \cdot y_1 \oplus \widetilde{a} \cdot y_2 \oplus \widetilde{a} \cdot y_3 \oplus \\ & \oplus \widetilde{b} \cdot x_1 \oplus \widetilde{b} \cdot x_2 \oplus \widetilde{b} \cdot x_3 \oplus \\ & \oplus x_1 \cdot y_1 \oplus x_1 \cdot y_2 \oplus x_1 \cdot y_3 \oplus \\ & \oplus x_2 \cdot y_1 \oplus x_2 \cdot y_2 \oplus x_2 \cdot y_3 \oplus \\ & \oplus x_3 \cdot y_1 \oplus x_3 \cdot y_2 \oplus x_3 \cdot y_3 \oplus \\ & \oplus z_1 \oplus z_2 \oplus z_3\end{aligned}. \quad (4)$$

За виразом (4) можна виконати операцію логічного множення над двома маскованими операндами так, щоб уникнути успішного проведення атаки ДАСП третього порядку. В результаті подальшого узагальнення виразів (2)–(4) отримуємо вираз для захисту обчислень від успішних атак ДАСП вищих порядків.

Нехай дані a і b є маскованими з використанням n масок: $\widetilde{a} = a \oplus x_1 \oplus \ldots \oplus x_n$, і $\widetilde{b} = b \oplus y_1 \oplus \ldots \oplus y_n$. Тоді узагальнений вираз для виконання операції логічного множення над цими даними буде таким:

$$a \cdot b \oplus \bigoplus_{i=1}^{n} z_i = \widetilde{a} \cdot \widetilde{b} \oplus \bigoplus_{i=1}^{n} x_i \cdot \widetilde{b} \oplus \bigoplus_{j=1}^{n} y_j \cdot \widetilde{a} \oplus \bigoplus_{\substack{i=1 \\ j=1}}^{n} x_i \cdot y_j \oplus \bigoplus_{i=1}^{n} z_i, \quad (5)$$

де $z = z_1, \ldots, z_n$ – маски результату обчислення.

Захист операції логічного додавання від проведення атаки ДАСП n-го порядку

Використовуючи аналогічний підхід, побудуємо вирази для виконання операції логічного додавання. Вираз для обчислення маскованого результату логічного додавання двох маскованих даних, який є стійким до атаки ДАСП першого порядку, матиме вигляд:

$$(a \vee b) \oplus z = \widetilde{a} \vee \widetilde{b} \oplus \widetilde{a} \cdot y \oplus \widetilde{b} \cdot x \oplus x \cdot y \oplus x \oplus y \oplus z. \quad (6)$$

З метою захисту результатів остаточних і проміжних обчислень від успішних атак ДАСП другого порядку скористаємося двома масками для кожного аргументу. Тоді a і b можна записати

так: $\widetilde{a} = a \oplus x_1 \oplus x_2$, $\widetilde{b} = b \oplus y_1 \oplus y_2$. Скориставшись алгебраїчними властивостями операцій логічного додавання і додавання за модулем 2, отримаємо вираз:

$$a \vee b \oplus z_1 \oplus z_2 = \widetilde{a} \vee \widetilde{b} \oplus \widetilde{a} \cdot y_1 \oplus \widetilde{a} \cdot y_2 \oplus \widetilde{b} \cdot x_1 \oplus \widetilde{b} \cdot x_2 \oplus x_1 \cdot y_1 \oplus x_1 \cdot y_2 \oplus \\ \oplus x_2 \cdot y_1 \oplus x_2 \cdot y_2 \oplus x_1 \oplus x_2 \oplus y_1 \oplus y_2 \oplus z_1 \oplus z_2 . \quad (7)$$

Представивши a і b у вигляді $\widetilde{a} = a \oplus x_1 \oplus x_2 \oplus x_3$, $\widetilde{b} = b \oplus y_1 \oplus y_2 \oplus y_3$, отримаємо вираз для маскованого результату:

$$(a \vee b) \oplus z_1 \oplus z_2 \oplus z_3 = \widetilde{a} \vee \widetilde{b} \oplus \widetilde{a} \cdot y_1 \oplus \widetilde{a} \cdot y_2 \oplus \widetilde{a} \cdot y_3 \oplus \\ \oplus \widetilde{b} \cdot x_1 \oplus \widetilde{b} \cdot x_2 \oplus \widetilde{b} \cdot x_3 \oplus \\ \oplus x_1 \cdot y_1 \oplus x_1 \cdot y_2 \oplus x_1 \cdot y_3 \oplus \\ \oplus x_2 \cdot y_1 \oplus x_2 \cdot y_2 \oplus x_2 \cdot y_3 \oplus \\ \oplus x_3 \cdot y_1 \oplus x_3 \cdot y_2 \oplus x_3 \cdot y_3 \oplus \\ \oplus x_1 \oplus x_2 \oplus x_3 \oplus y_1 \oplus y_2 \oplus y_3 \oplus \\ \oplus z_1 \oplus z_2 \oplus z_3 . \quad (8)$$

Вираз (8) дає змогу виконати операцію логічного додавання над двома маскованими операндами так, щоб уникнути успішного проведення атаки ДАСП третього порядку. В результаті подальшого узагальнення виразів (6)–(8) отримуємо вираз для захисту обчислень від успішних атак ДАСП вищих порядків.

Нехай дані a і b є маскованими з використанням n масок: $\widetilde{a} = a \oplus x_1 \oplus \ldots \oplus x_n$, $\widetilde{b} = b \oplus y_1 \oplus \ldots \oplus y_n$. Тоді узагальнений вираз для виконання операції логічного додавання над цими даними набуде вигляду:

$$(a \vee b) \oplus \bigoplus_{i=1}^{n} z_i = \widetilde{a} \vee \widetilde{b} \oplus \bigoplus_{i=1}^{n} x_i \cdot \widetilde{b} \oplus \bigoplus_{j=1}^{n} y_j \cdot \widetilde{a} \oplus \bigoplus_{\substack{i=1 \\ j=1}}^{n} x_i \cdot y_j \oplus \bigoplus_{i=1}^{n} x_i \oplus \bigoplus_{j=1}^{n} y_j \oplus \bigoplus_{i=1}^{n} z_i , \quad (9)$$

де $z = z_1, \ldots, z_n$ – маски результату обчислення.

Отже, використовуючи вирази (5) і (9), можна створити базові логічні операційні пристрої для подальшої побудови на їхній основі комп'ютерних засобів для виконання криптографічних перетворень, які є захищеними від успішного проведення атак ДАСП вищих порядків. Необхідний рівень захисту (порядок атаки ДАСП) задається шляхом відповідного вибору кількості масок, які використовуються для подання даних і результатів.

Оцінка складності захисту двомісних логічних операцій від проведення атак ДАСП вищих порядків

Для оцінки складності захисту двомісних логічних операцій від проведення атак ДАСП вищих порядків скористаємося виразами (5) і (9). Оцінимо часову, апаратну та місткістну характеристики складності [15]. При цьому приймемо, що порядок виконання складових операцій цих виразів не впливає на характеристики складності.

Оцінка часової складності полягає у визначенні довжини критичного шляху обробки даних. Для виразів (5) і (9) критичний шлях визначається порядком виконання обчислень. Уникнення успішного проведення атак ДАСП вищих порядків можливе лише для спеціального порядку використання масок результату z та обробки даних.

Для виразу (5) доданки $\widetilde{a} \cdot \widetilde{b}$, $x_i \cdot \widetilde{b}$, $y_j \cdot \widetilde{a}$ та $x_i \cdot y_j$ можна обчислити паралельно, а маску результату вводити поступово, починаючи з $x_i \cdot y_j$ і закінчуючи $\widetilde{a} \cdot \widetilde{b}$. Тому часова складність

виконання виразу (5) t_{MAND} залежатиме від тривалості виконання однієї операції логічного множення однобітових даних $t_{\wedge}$, додавання за модулем 2 однобітових даних $t_{\oplus}$, необхідного для захисту від атак ДАСП порядку n>1, і становитиме

$$t_{MAND}(n) = 3nt_{\oplus} + n^2 t_{\oplus}, \tag{10}$$

а часова складність виконання виразу (5) для захисту від атак ДАСП першого порядку (n=1) дорівнюватиме

$$t_{MAND} = t_{\wedge} + 4t_{\oplus}.$$

Аналогічно, для виразу (9) доданки $\tilde{a} \vee \tilde{b}$, $x_i \cdot \tilde{b}$, $y_j \cdot \tilde{a}$, $x_i \cdot y_j$ можна обчислювати паралельно, а маску результату вводити поступово, починаючи з x_i, y_j і закінчуючи $\tilde{a} \vee \tilde{b}$. Часова складність виконання виразу (9) t_{MOR} залежатиме від тривалості виконання однієї операції логічного додавання однобітових даних $t_{\vee}$, логічного множення однобітових даних $t_{\wedge}$, додавання за модулем 2 однобітових даних $t_{\oplus}$, необхідного порядку захисту від атак ДАСП n:

$$t_{MOR}(n) = 5nt_{\oplus} + n^2 t_{\oplus}. \tag{11}$$

Паралельне виконання виразів (5) і (9) можна використати для апаратної реалізації захисту від проведення атак ДАСП вищих порядків. Для побудови графіків залежності часової складності виконання захищених маскованих операцій від порядку атаки ДАСП припустимо, що $t_{\wedge} \approx t_{\oplus}$ (рис. 3).

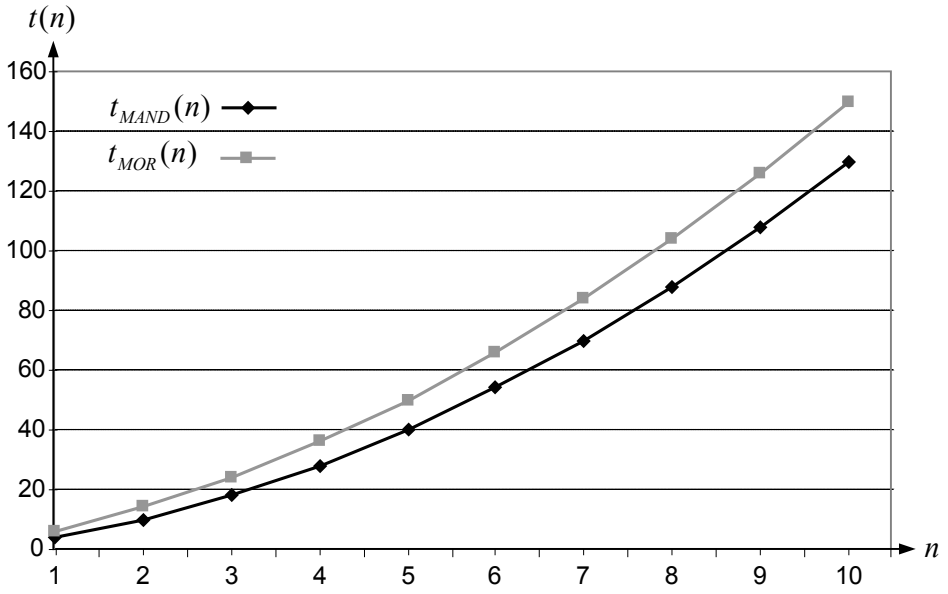

Рис. 3. Графік залежності часової складності паралельного виконання захищених маскованих операцій від порядку атаки ДАСП

З наведених на рис. 3 графіків випливає, що із використанням запропонованого способу захисту від атак ДАСП вищих порядків час виконання (затримка виконання) маскованих операцій зростатиме пропорційно до квадрата порядку атаки ДАСП. При цьому за однакових порядків атаки ДАСП часова складність виконання операції логічного множення над маскованими даними є меншою за відповідну часову складність для виконання операції логічного додавання.

При альтернативному виконанні усіх операцій послідовно (однак у заданому порядку) згідно з виразом (5), часова складність виконання захищеної маскованої операції логічного множення становитиме

$$t_{MAND}(n) = t_{\wedge} + nt_{\oplus} + 2n(t_{\oplus} + t_{\wedge}) + n^2(t_{\oplus} + t_{\wedge}). \tag{12}$$

Аналогічно, для виразу (9) отримаємо:

$$t_{MOR}(n) = t_{\vee} + 3nt_{\oplus} + 2n(t_{\oplus} + t_{\wedge}) + n^2(t_{\oplus} + t_{\wedge}). \quad (13)$$

Послідовне виконання виразів (5) і (9) можна використати під час програмної реалізації захисту від проведення атак ДАСП вищих порядків на програмованих процесорах. Оскільки у програмованих процесорах час виконання інструкцій логічних операцій є приблизно однаковий і майже не залежить від типу цих логічних операцій, то для побудови графіків залежності часової складності виконання захищених маскованих операцій від порядку атаки ДАСП припустимо, що $t_{\wedge} \approx t_{\oplus} \approx t_{\vee}$ (рис. 4).

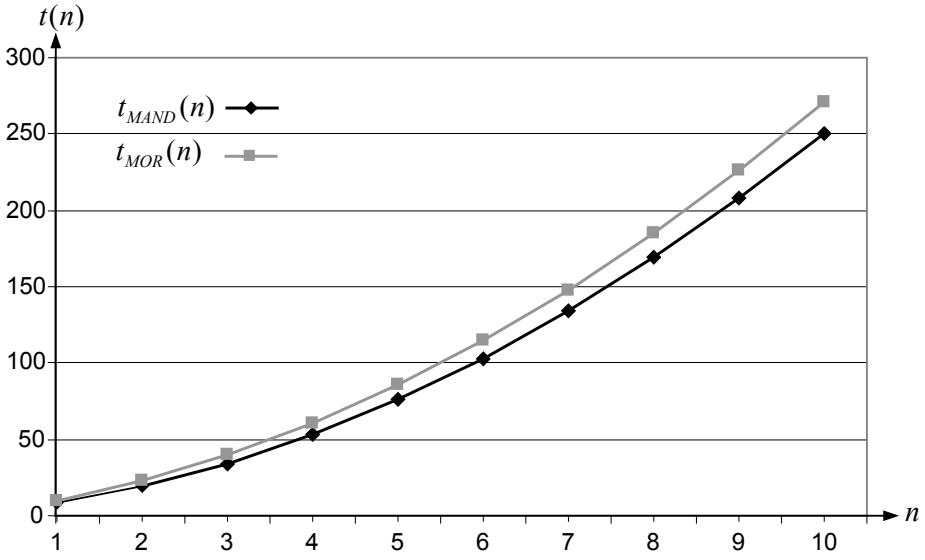

Рис. 4. Графік залежності часової складності послідовного виконання захищених маскованих операцій від порядку атаки ДАСП

З наведених на рис. 4 графіків випливає, що із використанням запропонованого способу захисту від атак ДАСП вищих порядків час виконання (затримка виконання) маскованих операцій зростатиме аналогічно до складності паралельного способу виконання – пропорційно до квадрата порядку атаки ДАСП. При цьому, за однакових порядків атаки ДАСП часова складність виконання операції логічного множення над маскованими даними є меншою за відповідну часову складність для виконання операції логічного додавання.

Апаратну складність оцінюють як кількість умовних (типових) елементів, необхідних для виконання заданого алгоритму дій. Оцінимо апаратну складність виразів (5) і (9) через кількість двовходових логічних елементів логічного множення $N_{\wedge}$, логічного додавання $N_{\vee}$, додавання за модулем 2 $N_{\oplus}$ (див. таблицю).

Апаратна складність реалізації маскованого логічного множення і додавання

Операція	$N_{\vee}$	$N_{\wedge}$	$N_{\oplus}$
Логічне множення	0	$1+2n+n^2$	$3n+n^2$
Логічне додавання	1	$2n+n^2$	$5n+n^2$

Практична оцінка апаратної складності реалізації захищених маскованих логічних операцій від порядку атаки ДАСП полягає у підрахунку загальної кількості логічних елементів, необхідних для цієї реалізації (рис. 5).

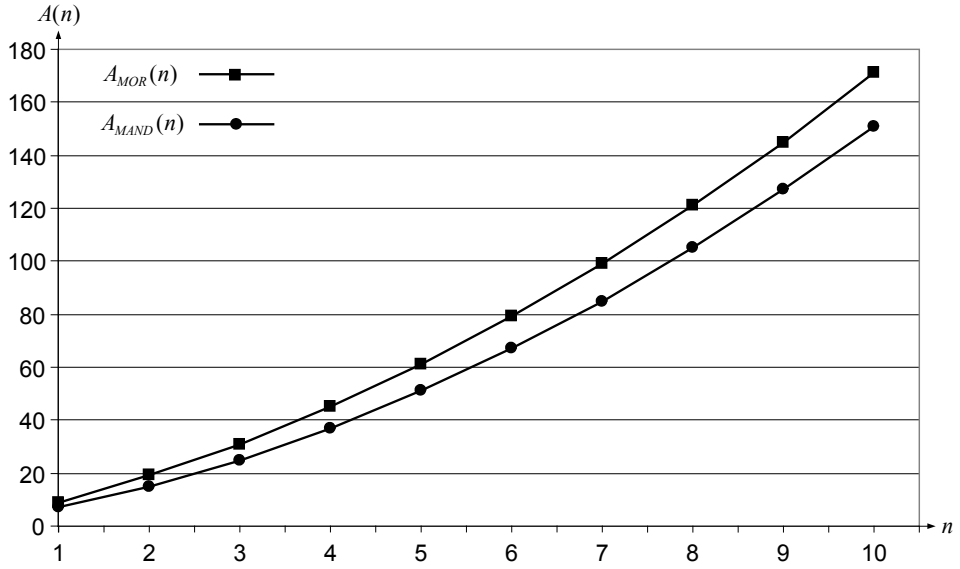

Рис. 5. Графік залежності апаратної складності реалізації захищених маскованих логічних операцій від порядку атаки ДАСП

З наведених на рис. 5 графіків випливає, що із використанням запропонованого способу захисту від атак ДАСП вищих порядків апаратна складність реалізації маскованих операцій зростатиме пропорційно до квадрата порядку атаки ДАСП. При цьому, за однакових порядків атаки ДАСП апаратна складність виконання операції логічного множення над маскованими даними є меншою за відповідну апаратну складність для виконання операції логічного додавання.

Місткістна складність оцінюється у кількості елементів пам'яті, необхідних для реалізації обчислень. Для проведення такої оцінки приймемо, що обчислення згідно з формулами (5) і (9) реалізуються у вигляді наперед обчисленої таблиці T_{op} для усіх можливих варіантів вхідних даних і відповідних масок (рис. 6).

Зауважимо, що для однакових n розмірність вхідних даних у виразах (5) і (9) є однаковою. Тому місткістна складність табличного виконання цих виразів у бітах є однаковою і задається виразом

$$W(n) = 2^{3n+2}. \qquad (14)$$

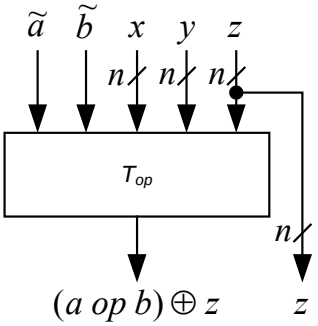

Рис. 6. Табличне виконання захищених маскованих логічних операцій

З виразу (14) випливає, що із зростанням порядку атаки ДАСП місткістна складність виконання захищених маскованих логічних операцій пропорційна до степені, що дорівнює порядку атаки ДАСП. Табличне виконання цих операцій доцільно використовувати як для програмної, так і для апаратної реалізації захисту логічних операцій від атак ДАСП вищих порядків.

Висновки

У роботі запропоновано узагальнений метод побудови базових логічних елементів (логічного множення та додавання) для комп'ютерних засобів захисту інформації, стійких до атак ДАСП вищих порядків. Особливістю запропонованого методу є його параметризування. Параметром захисту є порядок атаки ДАСП, від якої необхідно захистити комп'ютерні засоби виконання криптографічних перетворень. Оскільки основну увагу приділено захисту перетворень на найнижчому рівні – рівні базових логічних операцій, то запропонований метод можна використати для побудови як програмованих, так і апаратних комп'ютерних засобів виконання криптографічних перетворень.

У роботі проаналізовано складність запропонованого методу захисту. Зокрема, проаналізовано часову, апаратну та місткістну характеристики складності для двох способів виконання обчислень –

паралельного і послідовного. Паралельний спосіб виконання доцільно використовувати для апаратної реалізації захисту від проведення атак ДАСП вищих порядків. Послідовний спосіб доцільно використовувати при програмній реалізації захисту від проведення атак ДАСП вищих порядків на програмованих процесорах. Встановлено, що із використанням паралельного або послідовного способів виконання запропонованого методу захисту від атак ДАСП вищих порядків час виконання (затримка виконання) маскованих операцій зростатиме пропорційно до квадрата порядку атаки ДАСП. При цьому, за однакових порядків атаки ДАСП часова складність виконання операції логічного множення над маскованими даними є меншою за відповідну часову складність для виконання операції логічного додавання.

Оцінкою та аналізом апаратної складності запропонованого методу встановити, що апаратна складність реалізації маскованих операцій зростає пропорційно до квадрату порядку атаки ДАСП. При цьому, за однакових порядків атаки ДАСП апаратна складність виконання операції логічного множення над маскованими даними є меншою за відповідну апаратну складність для виконання операції логічного додавання.

Результати аналізу місткісної складності виконання запропонованого методу дали змогу встановити, що із ростом порядку атаки ДАСП місткістна складність виконання захищених маскованих логічних операцій пропорційна до степеня, що дорівнює порядку атаки ДАСП. Табличне виконання цих операцій доцільно використовувати як для програмної, так і апаратної реалізації захисту логічних операцій від ДАСП вищих порядків.

1. Kelsey J., Schneier B., Wagner D., Hall C., Side Channel Cryptanalysis of Product Ciphers // In 5th European Symposium on Research in Computer Security – ESORICS '98, vol. 1485 of Lecture Notes in Computer Science. – Springer-Verlag Berlin Heidelberg, 1998. – P. 97–110. 2. Clavier C., Coron J.-S., Dabbous N., Differential power analysis in the presence of hardware countermeasures / C.K. Koc, C. Paar, Eds., Cryptographic Hardware and Embedded Systems – CHESS 2000, vol. 1956 of Lecture Notes in Computer Science. – Springer-Verlag Berlin Heidelberg, 2000. – P. 252–263. 3. Kocher P., Jaffe J., Jun B., Differential Power Analysis // In proceedings of International conference CRYPTO'99, vol. 1666 of Lecture Notes in Computer Science. – Springer-Verlag Berlin Heidelberg, 1999. – P. 388–397. 4. Messerges T., Dabbish E., Sloan R., Eximining smart-card security under the threat of power analysis attack // IEEE Transactions on computers. – 2002. – Vol. 51, No 5. – P. 541–552. 5. Messerges T., Dabbish E., Sloan R., Power analysis attacks of modular exponentiation in smartcards / C.K. Koc, C. Paar, Eds., Cryptographic Hardware and Embedded Systems – CHESS 1999, vol. 1717 of Lecture Notes in Computer Science. – Springer-Verlag Berlin Heidelberg, 1999. – P. 144–157. 6. Akkar, M., Giraud, C. An implementation of DES and AES, secure against some attacks // In Proc. Cryptographic Hardware and Embedded Systems – CHES 2001, volume 2162 of Lecture Notes in Computer Science. – Springer-Verlag Berlin Heidelberg, 2001. – P. 309–318. 7. Akkar M., Bevan R., Dischamp P., Moyart D., Power analysis, what is now possible / T. Okamoto, Eds., International conference ASIACRYPT 2000, vol. 1976 of Lecture Notes in Computer Science. – Springer-Verlag Berlin Heidelberg, 2000. – P. 489–502. 8. Messerges T., Using second-order power analysis to attack DPA resistant software / C.K. Koc, C. Paar, Eds., Cryptographic Hardware and Embedded Systems – CHESS 2000, vol. 1956 of Lecture Notes in Computer Science. – Springer-Verlag Berlin Heidelberg, 2000. – P. 238–251. 9. Lv J., Han Y., Enhanced DES Implementation Secure against High-Order Differential Power Analysis in Smartcards // In Proceedings of ACISP'05 – Tenth Australian Conference on Information Security and Privacy, Volume 3574 of Lecture Notes in Computer Science, Springer-Verlag Berlin Heidelberg, 2005. – P. 195–206. 10. Oswald E., Mangard S., Herbst C., Tillich S., Practical Second-Order DPA Attacks for Masked Smart Card Implementations of Block Ciphers // In Proc. CT-RSA 2006, Vol. 3860 of Lecture Notes in Computer Science, Springer-Verlag Berlin Heidelberg, 2006 – P. 192–207. 11. Shin J.H., Park D.J., Lee P. J., DPA attack on the Improved Ha-Moon Algorithm // In Proc. Information Security Applications, 6th International workshop – WISA 2005, vol. 3786 of Lecture Notes in Computer Science, Springer-Verlag Berlin Heidelberg, 2006. – P. 283–291. 12. Katsuyuki O., Kouichi S., A Second-Order DPA Attack Breaks a Window-Method Based Countermeasure against Side Channel Attacks, Lecture Notes in Computer Science, Volume 2433 of Lecture Notes in Computer Science, Springer-Verlag Berlin Heidelberg, 2002. –

P. 389–401. 13. Koecher et al., Using Unpredictable Information to Minimize Leakage from Smartcards and other Cryptosystems., USA patent 6327661., Dec. 4, 2001. – 14 p. 14. Trichina E., Korksihko T., Secure AES Hardware Module for Resource Constrained Devices, // Lecture Notes in Computer Science, Vol. 3313, Springer-Verlag Berlin Heidelberg, 2005. – P. 215–229. 15. Черкаський М. Складність апаратно-програмних комп'ютерних засобів // Сучасні проблеми в комп'ютерних науках. Contemporary Computing in Ukraine CCU'2000: Зб. наук. праць. – Львів, 2000. – С. 58–67.

УДК 004.627

В.Т. Кремінь, М.П. Кушнір
Національний університет "Львівська політехніка",
кафедра електронних обчислювальних машин

МЕТОДИ СТИСКУ ЕЛЕКТРОКАРДІОСИГНАЛІВ

Оцінено та порівняно трансформувальні методи стиску електрокардіосигналів. Наведено техніки кодування, в яких використано 1D та 2D дискретне косинусне перетворення (ДКП), 1D та 2D дискретне вейвлетне перетворення (ДВП) для того, щоб спочатку перетворити сигнал у іншу частотно-часову форму, яка краще підходить для виявлення та видалення надлишковостей. Величину та якість компресії оцінюють традиційними показниками, тобто ступенем стиску (CR) та середньоквадратичним відхиленням (PRD) відповідно.

This paper introduces an estimation and comparison of the transformation methods for the compression of Electrocardiogram (ECG) signals. The presented coding techniques using 1D and 2D Discrete Cosine Transform (DCT), 1D and 2D Discrete Wavelet Transform (DWT) to convert the signal firstly to some other time-frequency representation better suited for detecting and removing redundancy. A quantity and quality of compression is evaluated by traditional measures, i.e. Compression Ratio (CR) and Percent Root mean square Difference (PRD) respectively.

Вступ

Сьогодні проблему ефективної діагностики серцевих захворювань вирішують такими методами, як холтерівське моніторування та телемедицина. Остання безпосередньо використовує різноманітні комунікаційні технології для дистанційного медичного догляду. Довготривале накопичення та безпровідне передавання ЕКГ даних невеликими переносними пристроями сприяють легкому діагностуванню кардіологічних хвороб. Раціональне подання сигналу є особливо важливим за умов обмеження об'єму пам'яті та смуги пропускання каналу передачі. Тому залишається актуальним питання щодо використання ефективних методів компресії ЕКГ, які дають змогу зменшити надлишковість нестаціонарного та квазіперіодичного кардіосигналу та зберігатимуть клінічно важливі характеристики, такі як Р-хвиля, QRS комплекс та Т-хвиля.

Аналіз останніх досліджень і публікацій

Серед існуючих алгоритмів стиску ЕКГ даних можна виділити три основні категорії [1]:

1. Прямі методи (direct methods). Це методи, за допомогою яких з метою виявлення та видалення надлишкової інформації аналізують відліки оригінального сигналу. До цієї групи належать такі алгоритми, як AZTEC, TP, CORTES, FAN.