УДК 004.052.42

DOI: 10.18372/2073-4751.80.19774

Русанова О.В., к.т.н.,
orcid.org/0000-0003-0145-3012,
e-mail: olga.rusanova@gmail.com,
Марковський О.П., к.т.н.,
orcid.org/0000-0003-3483-4233,
e-mail: markovskyy@i.ua,
Вовк В.В.,
orcid.org/0009-0002-2303-3131,
e-mail: vovk_vlad@lll.kpi.ua

МЕТОД МОДУЛЯРНОГО ЕКСПОНЕНЦІЮВАННЯ З ЗАХИСТОМ ВІД АТАК АНАЛІЗОМ ДИНАМІКИ СПОЖИВАННЯ ПОТУЖНОСТІ

**Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»**

Вступ

Визначальними рисами розвитку інформаційних технологій останнього десятиліття стали динамічний прогрес Інтернету та якісно нові досягнення в мікроелектроніці. Це стимулювало розвиток та активне впровадження систем інтелектуального віддаленого контролю об'єктами реального світу на базі термінальних мікроконтролерів, здатних взаємодіяти з мережею Інтернет. Відповідні технології отримали назву Інтернету речей (*Internet of Things – IoT*). Початкова орієнтація *IoT* на управління побутовими приладами швидко змінилась на контроль за віддаленим технологічним обладнанням широкого кола призначень. Сьогодні технології *IoT* активно застосовуються в системах віддаленого контролю за об'єктами критичної інфраструктури, управління транспортом, дистанційного управління технологічним обладнанням, охоронної сигналізації та відео спостереження [1].

Оскільки в перелічених застосуваннях для обміну даними використовується потенційно вразливе середовище – мережа Інтернет, вони потребують застосування криптографічних засобів захисту. Для цього використовується широкий спектр стандартизованих протоколів захисту інформації, більшість з яких базуються на криптографічних алгоритмах з відкритим ключем.

Разом з тим, специфіка наведених систем не виключає можливості фізичного доступу сторонніх осіб до термінальних пристроїв. Це зумовлює небезпеку незаконної реконструкції секретних ключів шляхом відслідковування та аналізу динаміки споживання потужності термінальним мікроконтролером під час виконання криптографічного алгоритму.

Зазначений метод злому криптографічних алгоритмів захисту даних, якими обмінюються термінальні пристрої систем віддаленого управління є найдешевшим, технологічно простим, а тому являє реальну небезпеку для зазначених систем [2]. Особливо вразливими до нього є криптографічні алгоритми з відкритим ключем, в основі яких лежить операція модулярного експоненціювання, при цьому розряди експоненти для переважної більшості сучасних криптографічних алгоритмів з відкритим ключем і є секретним ключем.

В силу того, що при використанні класичного алгоритму модулярного експоненціювання, порядок виконання команд напряму залежить від значень розрядів експоненти, то застосування методу аналізу динаміки споживання потужності дозволяє відносно просто реконструювати секретний код ключа.

З розширенням застосування систем віддаленого керування на базі технологій *IoT* у критичних та безпекових сферах

життєдіяльності людства (таких як безпілотні літальні апарати, критична інфраструктура тощо), гострота зазначеної проблеми об'єктивно зростає та вимагає спеціальних заходів протидії спробам стороннього втручання в роботу зазначених систем через отримання доступу до секретних ключів аналізом динаміки зміни споживаної потужності термінальних обчислювальних платформ.

В рамках таких заходів першочергове місце займає нових методів реалізації базової операції криптографії з відкритим ключем – модулярного експоненціювання, які виключають залежність між порядком виконання команд на термінальному пристрої та значенням секретного коду ключа.

Таким чином, наукова задача ефективної протидії реконструкції секретних компонентів криптосистем аналізом динаміки споживання потужності шляхом розробки організації модулярного експоненціювання, яка виключає залежність порядку виконання команд від розрядів секретного коду ключа, є актуальною з огляду на сучасний стан розвитку інформаційних технологій.

Аналіз проблеми захисту даних від їх реконструкції за динамікою споживання потужності

Технології незаконної реконструкції даних за результатами аналізу динаміки споживання потужності термінальними комп'ютерними платформами стали активно застосовуватися починаючи з останнього десятиліття минулого століття [2].

Ці технології базуються на залежності споживання потужності мікроконтролером в кожен момент часу як від команди, яка на ньому виконується, так і від даних, що оброблюються цією командою. Реальна можливість відновлення даних, що оброблюються на комп'ютерній платформі за результатами аналізу динаміки споживання ним потужності існує лише для платформ, в яких реалізується обробка лише одного процесу [3].

На сьогоднішній день технологія відновлення даних за результатами аналізу динаміки споживання потужності термінальним мікроконтролером пройшла тривалий шлях розвитку. В останні роки для розпізнавання команд та даних за результатами вимірювань споживання потужності широко застосовуються механізми штучного інтелекту [4].

Історично першим різновидом цієї технології став простий аналіз динаміки споживання потужності (*Simple Power Analysis – SPA*). Ця технологія базується на розпізнаванні окремих команд за даними осцилограми споживання потужності. Іншими словами, технологія *SPA* дозволяє з визначеною ймовірністю виявити факт виконання певних команд. Якщо послідовність їх виконання залежить від даних, що оброблюються програмою, технологія *SPA* дозволяє реконструювати значення цих даних. Технологія передбачає наявність програми обробки даних.

Другий різновид технологій відновлення даних за аналізом динаміки споживання потужності комп'ютерної платформи отримав назву диференційного аналізу потужності (*Differential Power Analysis – DPA*). Ця технологія базується на статистичній обробці великої кількості діаграм споживання потужності з метою виявлення "критичних точок", які несуть певну інформацію про дані програми [2].

За умови використання ефективних алгоритмів розпізнавання команд або груп команд, що виконуються на термінальному мікроконтролері, технологія *SPA* дозволяє відновити секретні дані за однією діаграмою споживання потужності. З позицій безпечного виконання криптографічних алгоритмів на термінальному мікроконтролері, технологія *SPA* особливо небезпечна для базової операції криптографії з відкритим ключем – модулярного експоненціювання, тобто обчислення $A^E \bmod M$. В реальних механізмах криптографічного захисту з відкритим ключем ця операція виконується над n-розрядними числами, причому значення n на порядки перевищує розрядність процесора – r [6]. На сьогоднішній день в переважній більшості практичних застосувань використовується розрядність $n = 4096$.

Операція модулярного експоненціювання полягає у реалізації одного з двох різновидів класичного алгоритму. Класичний алгоритм передбачає виконання n циклів, операції у яких залежать від значення бітів коду експоненти:

$$E = 2^n \cdot e_{n-1} + \ldots + 2 \cdot e_2 + e_1,\ \forall j = 1,2,\ldots,n\text{-}1 : e_j \in \{0,1\}.$$

В першому різновиді класичного алгоритму код E сканується від старшого до молодшого розряду, тобто від e_{n-1} до e_1. Виконання алгоритму починається зі встановлення лічильника $j = n$ та початкового значення результату $R = 1$. У кожному j-тому циклі значення R підноситься до квадрату по модулю M: $R = R^2 \bmod M$. Якщо значення поточного біту експоненти $e_j = 1$, то R множиться на значення A по модулю M: $R = R \cdot A \bmod M$.

В другому різновиді класичного алгоритму модулярного експоненціювання сканування коду експоненти виконується з молодшого розряду до старшого, тобто від e_1 до e_{n-1}. Відповідно, лічильник j номеру розряду експоненти на початку виконання другого різновиду алгоритму встановлюється рівним одиниці: $j = 1$. На відміну від першого різновиду алгоритму, у другому використовуються дві змінні.

Значення першої з них у кожному з циклів позначаються як $D_1, D_2, \ldots, D_n$. Початковому значенню D_1 присвоюється значення A: $D_1 = A$. Другій змінній R на початку присвоюється значення одиниці: $R = 1$.

У кожному j-му циклі спочатку обчислюється оновлене значення R. При цьому, наступне значення R залежить від значення поточного біту ключа: якщо $e_j = 1$, то реалізується множення R на D_1 по модулю M: $R = R \cdot D_j \bmod M$ після цього у рамках j-го циклу обчислюється наступне $(j+1)$ значення D_{j+1} шляхом піднесення поточного значення D_j до квадрату по модулю M: $D_{j+1} = D_j^2 \bmod M$, а лічильник j збільшується на одиницю. По закінченню n циклів, у змінній R сформовано результат модулярного експоненціювання $A^E \bmod M$.

Аналіз обох розглянутих вище різновидів класичного алгоритму модулярного експоненціювання показав, що виконання операції модулярного множення прямо пов'язане зі значенням поточного біту секретного коду експоненти e_j. За цифровою осцилограмою динаміки споживання потужності термінальним мікроконтролером під час виконання модулярного експоненціювання можна доволі просто відслідкувати операції модулярного множення в циклах, а отже і відновити розряди секретного коду ключа [6].

Зважаючи на значну вразливість криптографії з відкритим ключем, яка використовує модулярне експоненціювання, до зазначеного методу аналізу динаміки споживання потужності – *SPA*, до теперішнього часу розроблено низку методів протидії атакам такого типу.

Найбільш простий метод протидії полягає в виконанні операцій модулярного множення незалежно від значення поточного біту секретного коду експоненти e_j [7]. Реалізація цього підходу передбачає створення та використання змінної X "фальшивого результату", у котру перезаписується результат модулярного множення попереднього значення X на змінну D_j на ітераціях при $e_j = 0$. Недоліком цього методу є збільшення часу виконання алгоритму на 25%, тобто витрата значної кількості додаткових ресурсів.

Ще один підхід до захисту операції модулярного експоненціювання від *SPA* полягає в такій організації модулярного піднесення до квадрату яка б не дозволяла розпізнавати різницю між цією операцією та модулярним множенням. На практиці застосування цього підходу має наслідком сповільнення приблизно на 18% виконання модулярного експоненціювання, тому що не використовується резерв швидкої реалізації цієї операції за рахунок властивостей симетрії модулярного квадрату [8].

Група методів захищеного обчислення модулярної експоненти базується на розкладенні коду експоненти на адитивно-мультиплікативні складові [9]. Основа перевага такого підходу полягає у можливостях організації програмного поліморфізму при виконанні операції модулярного

експоненціювання. Окрім захисту від *SPA*, це забезпечує достатньо ефективний захист від *DPA*. Суттєвим недоліком такого підходу є втрата близько 250% продуктивності, зумовленої дублюванням обчислення *D* та роботою генератора псевдовипадкових кодів програми.

Інший підхід до протидії *SPA* при виконанні модулярного експоненціювання на термінальному мікроконтролері полягає у залученні до обчислень віддалених комп'ютерних систем з використанням хмарних технологій [10, 11]. У таких рішеннях застосування технологій *SPA* та *DPA* вбачається практично неможливим за рахунок віддаленої реалізації значної частини обчислень. Додатковою перевагою є збільшення швидкості обчислень, зумовленої використанням високопродуктивних систем. Запровадження зазначених удосконалень супроводжується значними ресурсними витратами, необхідними для досягнення відповідного рівня захисту від незаконного відновлення секретних компонентів модулярного експоненціювання за даними, які передаються у хмарні системи.

Проведений огляд показав, що основною передумовою результативного застосування технологій *SPA* для протизаконного доступу до ключів механізмів захисту на основі несиметричної криптографії є зв'язок між індексом циклу (тобто номером біту секретного коду експоненти) обчислення базової операції такої криптографії – модулярного експоненціювання та моментами виконання операцій модулярного множення. Відповідно, для захисту механізмів криптографічного захисту з відкритим ключем від атак *SPA* потрібно порушити вказаний зв'язок. Для цього в відомих методах застосовуються спеціальні заходи, що утруднюють розпізнавання за осцилограмою споживання потужності модулярних операцій піднесення до квадрату та множення.

Загальний недолік відомих методів протидії *SPA* полягає в тому, що вони використовують ресурси, які для більшості практичних застосувань є критичними для їх ефективності. В повній мірі це стосується методів, які реалізують захист від *SPA* при реалізації модулярного експоненціювання за рахунок його сповільнення.

Таким чином, проведений огляд показав, що існуючі методи протидії *SPA* при реалізації на термінальних контролерах базової операції несиметричної криптографії – модулярного експоненціювання, не повною мірою відповідають сучасним вимогам, які диктуються динамічним розширенням використанням систем контролю віддаленими об'єктами на основі технологій *IoT*.

Мета досліджень

Мета досліджень полягає в підвищенні ефективності захисту ключів від їх незаконної реконструкції аналізом динаміки споживання потужності термінальними мікроконтролерами під час реалізації на них модулярного експоненціювання – базової операції несиметричної криптографії за рахунок організації його виконання, яка виключає часову залежність між розрядами коду експоненти та моментами здійснення операцій модулярного множення.

Організація модулярного експоненціювання стійка до зламу аналізом динаміки споживання потужності

Для досягнення поставленої мети пропонується спеціальна організація протидії незаконній реконструкції показника модулярної експоненти аналізом споживання потужності термінальним мікроконтролером при її обчисленні.

Основна ідея запропонованої організації полягає у виключенні залежності між значеннями показника степені модулярної експоненти та послідовністю виконання команд при її обчисленні.

Аналіз класичного алгоритму модулярного експоненціювання $A^E \bmod M$ з молодших розрядів коду *E* експоненти свідчить про те, що послідовність обчислення значень *D*, тобто модулярних експонент числа *A* з показниками, що є степенями двійки $D_1 = A^{2^0} \bmod M$, $D_2 = A^{2^1} \bmod M$, … , $D_{n-1} = A^{2^{n-1}} \bmod M$ не

залежить від послідовності обчислень значень R поточного результату. Це надає можливості рознесення в часі, тобто порушення синхронізації, обчислення значень $D_1,D_2,\ldots,D_{n-1}$ та обчислень послідовних значень поточного результату шляхом зберігання в пам'яті потрібних для цього значень D.

Для тимчасового зберігання операндів модулярних множень, тобто значень $D_1,D_2,\ldots,D_{n-1}$ запропонований метод передбачає використання пам'яті W яка містить d комірок, причому $d>k_e$, де k_e – кількість одиничних компонент експоненти E.

Зважаючи на те, що значення модулярної експоненти у переважній більшості криптографічних алгоритмів з відкритим ключем являє собою закритий ключ, то його можна вважати практично незмінним.

У зв'язку з цим, його зберігання пропонується у вигляді спеціальної структури Q яка складається з n полів: $Q = \{q_1, q_2,..., q_n\}$. Кожне j-те, $j \in \{1,2,\ldots,n\}$, поле містить дві компоненти: адресу a_j і тег активації операції множення b_j, $q_j = <a_j, b_j>$.

Адресна компонента a_j вказує адресу у пам'яті W куди записується обчислене значення ваги D_j на j-тому циклі. Тег активація b_j операції множення набуває одиничного значення, коли на j-тому циклі здійснюється операція модулярного множення.

Значення компонент структури Q визначаються наступним чином:

1. $\forall j \in \{1,2,..n\}$; якщо $e_j = 1$, то $0 \le a_j \le k_e$; $\forall i \in \{1,2,\ldots,n\}$, $i \ne j$; i,j: $a_i \ne a_j$;
2. $\forall j \in \{1,2,..n\}$; якщо $e_j = 0$, то $k_e \le a_j \le d$.

Структура Q не залежить від операнду A модулярної експоненти $A^E \bmod M$ і формується лише один раз. Головна ціль використання структури Q полягає в виключенні операцій аналізу поточних бітів коду експоненти в ході їх сканування при реалізації класичного алгоритму обчислення модулярної експоненти. Використання структури Q дозволяє визначити в випадковому порядку комірки пам'яті W для зберігання операндів віднесених у часі виконання операцій модулярного множення. Значення $D_1,D_2,\ldots,D_{n-1}$, які не використовуються для обчислення модулярних добутків зберігаються в іншій невеликій групі комірок пам'яті з можливістю перезапису. Це дозволяє досягти того, що при різних значеннях поточного біту експоненти E в програмі реалізується однакова послідовність операцій. Фактично, за запропонованим методом, в ході обчислення модулярної експоненти здійснюється сканування не коду експоненти E, а компонент описаної вище структури Q.

Тег активації операцій множення дозволяє організувати обчислення модулярних добутків, для яких сформовані дані набудь-якому циклі виконання модулярного експоненціювання. Зокрема, в варіанті коли $\forall\, l=1,2,\ldots,n-1$: $b_l=0$, $b_n=1$, всі операції модулярного множення реалізуються по закінченні обчислення всіх значень $D_1,D_2,\ldots,D_{n-1}$. В цьому варіанті в пам'яті зберігається k_e потрібних для обчислення модулярних добутків значень $D_1,D_2,\ldots,D_{n-1}$.

Рівень захищеності в такому варіанті визначається об'ємом перебору для відновлення локацій операцій модулярного множення, тобто кількістю Y варіантів локалізації k_e одиниць в n-розрядному двійковому коді:

$$Y = C_n^{k_e} = \frac{n!}{(n-k_e!)\cdot n!}. \qquad (1)$$

З використанням відомої формули Стірлінга вираз (1) для визначення чисельного значення Y може бути трансформований до наступного вигляду:

$$Y = \frac{n^n \cdot \sqrt{2\pi \cdot n}}{k_e^{k_e} \cdot (n-k_e)^{k_e} \cdot 2 \cdot \pi \cdot \sqrt{k_e \cdot (n-k_e)}} = \\ = \left(\frac{n}{n-k_e}\right)^{n-k_e} \cdot \left(\frac{n}{k_e}\right)^{k_e} \cdot \sqrt{\frac{n}{2\pi \cdot k_e \cdot (n-k_e)}} \qquad (2)$$

На практиці, при великих значеннях n, число k_e в коді експоненти E практично дорівнює половині n, тобто $k_e = 0{,}5 \cdot n$. Після підстановки вказаного значення

$k_e = 0{,}5 \cdot n$ в формулу (2), остання набуває точного вигляду:

$$Y = 2^n \cdot \sqrt{\frac{2}{\pi \cdot n}}. \qquad (3)$$

Зокрема, при n = 4096 формула (3) дає значення Y рівне 10^{1231}. Очевидно, що такий об'єм перебору можливих значень Y повністю виключає можливість його практичної реалізації. Разом з цим, таке велике значення об'єму Y перебору для відновлення коду елементи E свідчить про недоцільність збереження в пам'ять всіх значень D, потрібних для обчислень результату R модулярного експоненціювання, оскільки таке рішення потребує значних за обсягом об'ємів пам'яті.

З цих позицій, більш ефективним виглядає організація обчислення модулярної експоненти, яка передбачає чередування циклів обчислення D і поточного результату R. При цьому довжина такого циклу визначається вимогами до рівня U захищеності, зумовленого специфікою конкретного застосування. При цьому чисельне значення U визначається граничним об'ємом перебору, який потрібно здійснити для відновлення коду експоненти E, який в більшості криптографічних протоколів виступає в якості секретного ключа.

Якщо рознесення виконання модулярних піднесень до квадрату та модулярних множень організується в межах фрагменту обробки n/L розрядів коду експоненти, то кількість $Y(L)$ варіантів локалізації k_e одиниць у n-розрядному коді E за умови що кожен n/L-розрядний фрагмент коду E містить k_e / L одиниць визначається наступною формулою:

$$Y(L) = (2^{\frac{n}{L}} \cdot \sqrt{\frac{2 \cdot L}{\pi \cdot n}})^L = 2^n \cdot (\frac{2 \cdot L}{\pi \cdot n})^{\frac{L}{2}}. \quad (4)$$

Об'єм $X(L)$ пам'яті необхідний при цьому для зберігання значень D, потрібних для обчислення всіх віднесених модулярних множень становить, в середньому, $X_L = n^2 / L$ біт.

Визначення чисельних значень L може здійснюватися як за заданими обмеженнями на об'єм X' пам'яті, так і за зумовленим специфікою конкретного застосування рівним захищеності Y':

$$Y' \le 2^n \cdot \left(\frac{2 \cdot L}{\pi \cdot n}\right)^{\frac{L}{2}}; \quad X' \le \frac{n^2}{L}. \qquad (5)$$

Наприклад, якщо значенні n=4096 об'єм X' пам'яті, яка може бути виділена на термінальному мікроконтролері для зберігання значень D обмежена обсягом 64 Кбайтів, тобто $X' = 2^{19}$ бітів, то відповідно значення L обчислюється по формулі (5) у вигляді: $L=n^2 \cdot (X')^{-1} = 2^{24} \cdot 2^{-19} = 2^5 = 32$. Тоді об'єм перебору для відновлення коду експоненти E, обчислений за формулою (5) становить: $Y' = 2^n \cdot 0{,}005^{32} = 10^{1231} \cdot 10^{-71} = 10^{1157}$

Цілком очевидним є те, що таке значення Y' практично виключає можливість відновлення коду експоненти за результатами *SPA*.

Розроблена в рамках запропонованого методу формалізована процедура обчислення модерної експоненти $A^E \ mod \ M$ реалізується на основі першого класичного алгоритму модулярного експоненціювання, тобто використовує дві змінні D та R. Робота запропонованої процедури модулярного експоненціювання зводиться до виконання наступної послідовності дій:

1. Встановлюються початкові значення змінних процедури: ваги D_1 : $D_1 = A$ та результату R: R = 1. Стартове значення індексу j циклу встановлюється рівним одиниці: j = 1.

2. Значення D_j записується у пам'ять W за адресою a_j.

3. Виконується обрахунок наступного значення ваги D_{j+1} шляхом модулярного піднесення до квадрату попереднього значення ваги D_j : $D_{j+1} = D_j^2 \ mod \ M$. Перевіряється значення тегу активації операції множення: якщо b_j рівне нулю, то відбувається перехід на п. 7.

4. Лічильник k кількості множень встановлюється в нуль: k=0.

5. Поточне значення результату R множиться по модулю M на код, що читається з пам'яті W за адресою k: $R=R\cdot W[k]$ *mod* M.

6. Значення k збільшується на одиницю: $k=k+1$. Якщо обчислене значення k не перевищує k_e/L, тобто $k < k_e/L$, виконується перехід на повторне виконання пункту 7.

7. Здійснюється інкремент індексу циклу: $j = j + 1$, якщо значення j менше за n, тобто $j < n$, здійснюється повернення на повторне виконання п. 2.

8. Кінець. Результат модулярного експоненціювання сформовано у змінній: $R=A^E$ *mod* M.

Як випливає з викладеної процедури, в ній, на відміну від класичного алгоритму не використовуються умовні переходи за значенням поточного біту коду експоненти. Це означає, що всі операції в процедурі не залежать від бітів секретного коду експоненти.

Робота запропонованого методу модулярного експоненціювання, стійкого до аналізу динаміки споживання потужності, ілюструється наступним прикладом. Нехай модуль M дорівнює 413-ти, а n відповідно дорівнює восьми: $n = 8$. Закритий ключ криптосистеми E обрано рівним 89: $E = 89 = 1011001_2$. Відповідно, кількість k_e одиниць в коді E становить чотири: $k_e=4$. Виходячи з заданих обмежень на об'єм пам'яті термінального мікроконтролера визначено, що значення L дорівнює двом: $L=2$.

Це означає, що при $L = 2$ кількість комірок у пам'яті W, орієнтованих на зберігання значимих кодів ваги D становить $k_e / 2 = 2$. Для зберігання незначимих кодів ваги обрано кінцеві дві комірки. Відповідно, загальна кількість d комірок виділеної пам'яті W дорівнює чотирьом: $d=k_e/2+2=4$. Адресація виділених комірок пам'яті W здійснюється від нуля до трьох, тобто нумерація адрес комірок від 0 до 3.

Згідно з викладеним методом, випадковим чином визначається заповнення структура Q у наступному вигляді:

$$Q = \{\langle 1,0\rangle,\langle 3,0\rangle,\langle 2,0\rangle,\langle 0,1\rangle,\langle 0,0\rangle,\langle 3,0\rangle,\langle 1,1\rangle\},$$

де першому значенню відповідає адреса комірки 1, а друге значення сигналізує про виконання проміжного модулярного множення.

Це означає, що початкове значення D_1 на нульовому кроці зберігається у пам'яті W у комірці з адресою 1. При обчисленні модулярної експоненти, наприклад 103^{83} *mod* 413 = 129 значення A дорівнює 103: $A = 103$. У рамках пункту 1 процедури обчислення, початкове значення D_1 рівне значенню A, тобто дорівнює 103: $D_1=103$. Початкове значення R результату встановлюється рівним одиниці: $R = 1$. Поточне значення індексу циклу j встановлюється рівним одиниці: $j = 1$

У рамках п. 2 процедури, поточне значення D_1 записується у пам'ять W за адресою першої компоненти структури Q, яка дорівнює 1. Тобто у першу комірку пам'яті записується значення $D_1=103$: $W[1]=103$. В його рамках п.3 здійснюється модулярне піднесення до квадрату поточного значення D_1: $D_2=D_1$ *mod* $M = 103^2$ *mod* 413 = 284. Значення b_1 рівне нулю, отже наступним виконується п.7 процедури, в якому інкрементується значення індексу j циклу та повернення на повторне виконання п.2. Подальші обчислення при значеннях j від 1 до 3 здійснюються аналогічним чином і отримані результати обчислення значень D розміщуються в пам'яті як показано в таблиці 1.

Таблиця 1. Записані в пам'яті W коди після перших циклів 0-3

Адреса	0	1	2	3
Код	186	103	121	284

На третьому кроці виконання описаної вище процедури, тобто при $j = 3$, значення тегу b_j активації модулярного множення дорівнює одиниці: $b_3=1$. Це означає що в рамках п. 4, в лічильник k кількості операцій модулярного множення встановлюється в нуль: $k=0$. З використанням цього лічильника організується двократне виконання п. 5-6 процедури. При першому виконанні п.5, тобто при $k=0$ здійснюється обчислення $R=R\cdot W[0]$ *mod* $M = 1\cdot 186$ *mod* 413 = 186. При другому виконанні п.5,

тобто при k =1 реалізується модулярне множення $R=R\cdot W[1]\ mod\ M$ = 186·103 *mod* 413 = 160.

Після виконання описаних двох циклів модулярного множення, після досягнення лічильником k значення $k_e/2=2$, в рамках п.7 процедури здійснюється інкремент індексу j циклу, в результаті чого він стає рівним чотирьом: $j=4$. Після цього, оскільки $j<n$, здійснюється перехід на повторне виконання п.2.

Відповідно, обчислене раніше значення $D_4 = 317$ записується в пам'ять за нульовою адресою: $W[0]=317$. В наступному п.3. здійснюється модулярне піднесення до квадрату поточного значення D_4: $D_5=D_4\ mod\ M = 317^2\ mod\ 413 = 130$. В силу того, що біт b_4 рівний нулю: $b_4=0$ виконується перехід на п.7, в рамках якого значення індексу j збільшується на одиницю: $j = j+1 =5$. Так, як $j<n$, відбувається повернення на виконання п. 2, в якому обчислене значення $D_5 = 130$ зберігається в память за адресою $a_5=3$: $W[3]=130$. В наступному п.3 відбувається модулярне піднесення до квадрату поточного значення D_5: $D_6=D_5\ mod\ M = 130^2\ mod\ 413 = 380$. Так як біт b_5 дорівнює нулю: $b_5=0$, то виконується перехід на п.7, в рамках якого значення індексу j збільшується на одиницю: $j = j+1 =6$. Так, як $j<n$, відбувається повернення на виконання п. 2, в якому обчислене значення $D_6 = 380$ записується в пам'ять за адресою $a_6=1$: $W[1]=380$. Після цього в пам'яті W записані коди, які представлені в таблиці 2.

Таблиця 2. Записані в пам'яті W коди після перших циклів 4-6

Адреса	0	1	2	3
Код	317	380	121	130

Оскільки біт b_6 активації множень дорівнює одиниці: $b_6=1$, то в п.4-6 процедури послідовно виконуються дві операції модулярного множення: $R=R\cdot W[0]\ mod\ M = 160\cdot 317\ mod\ 413 = 334$ та $R=R\cdot W[1]\ mod\ M = 334\cdot 380\ mod\ 413 = 129$. Отриманий код $R=129$ є результатом модулярного експоненціювання $103^{83}\ mod\ 413$.

Оцінка ефективності

Ефективність запропонованого підходу до безпечної реалізації базової операції криптографії з відкритим ключем – модулярного експоненціювання на термінальних обчислювальних платформмах, визначається за такими критеріями:

- Рівнем захищеності від спроб реконструювати код експоненти E або інформаційної чи диференційної компоненти A з використанням технологій простого чи диференційного аналізу динаміки споживання потужності термінальною обчислювальною платформою, під час виконання на ній операції модулярного експоненціювання;
- Впливом запропонованої організації захисту від атак на секретні компоненти операції модулярного експоненціювання на час реалізації її на термінальному обладнанні систем віддаленого контролю та управлянні;
- Об'ємом додаткових ресурсів, зокрема пам'яті, для реалізації запропонованого методу з огляду на можливості термінальних обчислювальних платформ.

Рівень захищеності від спроб незаконного відтворення секретних компонентів операції модулярного експоненціювання під її реалізації на термальних мікроконтролерах аналізом динаміки споживання ними потужності є основним з поміж наведених вище критеріїв. В запропонованій організації обчислення модулярної експоненти відсутні операції тестування поточних бітів коду експоненти. Відповідно, виключено залежність факту виконання будь-яких команд мікроконтролера від бітів секретного коду експоненти. В теоретичному плані, це досягається за рахунок рознесення в часі виконання операції модулярного множення поточного результату R на відповідну вагу D та її формування. В попередньому розділі показано, що об'єм перебору в реальних умовах для визначення зміщення в часі операції модулярного множення вимірюється величиною близькою до 10^{1200}, що робить такий перебір практично нездійсненним. Таким чином, можливість застосування

SPA для незаконної реконструкції коду експоненти запропонованим методом повністю виключена.

Разом з тим, певні можливості для результативного застосування *DPA* залишаються. Найбільш критичним в цьому плані є моменти запису ваги *D* після її формування в пам'ять. Статистичними методами можна, з певною ймовірністю, встановити адреси, за якими здійснюється запис в пам'ять кодів *D*. Проведені експериментальні дослідження показали, що при кількості вибірок, яка дорівнює 10^3 ймовірність відновлення одного розряду адреси, за якою здійснюється запис в пам'ять лежать в інтервалі від 0.62 до 0.74. Це означає, що кількість проб, для реконструкції адреси за даними *DPA* складає величину близьку до $5.5 \cdot 10^2$. Для того, що виключити таку можливість пропонується через 100-300 циклів виконання програми змінювати адресні компоненти структури *Q*. За цих умов виключається можливість формування стійкої статистики, не обхідної для результативної роботи *DPA*. Вказана зміна адресних компонентів може виконуватися спеціальним програмним модулем. Результати експерименттальних досліджень у цьому напрямку повністю підтвердили практичну ефективність такого заходу.

Значимим критерієм ефективності для систем контролю та управління віддаленими об'єктами є час реалізації модулярного експоненціювання. Особливо важливий цей чинник для систем, що працюють в реальному часі, до класу яких відноситься більшість термінальних мікроконтролерів. Час обчислення модулярної експоненти визначається кількістю мультиплікативних операцій модулярної арифметики, що виконуються на *n*–розрядними числами. Розрядність *n* визначається вимогами щодо безпеки криптографії з відкритими ключами і на сьогодні для більшості застосувань воно прийнято рівним 4096. Оскільки на практиці чисельне значення *n* значно більше за розрядність *r* мікроконтролера, адитивні операції, а також логічні операції, мають на декілька порядків нижчу обчислювальну складність в порівнянні з мультиплікативними. Тому при оцінці часових характеристик, доцільним вважається урахування лише часу виконання мультиплікативних операцій модулярної арифметики над *n*-розрядними числами. Операція модулярного піднесення до квадрату зазвичай виконується швидше, ніж операція модулярного множення над різними числами. В обох різновидах класичного алгоритму модулярного експоненціювання часові характеристики визначаються часом виконання *n* операцій модулярного піднесення до квадрату та *n*/2 операцій модулярного множення.

В першій фазі запропонованого методу виконується *n* операцій модулярного піднесення до квадрату, час виконання яких на 3-4 порядки вищий у порівнянні з часом запису результату у пам'ять. В другій фазі виконується k_e, яке в середньому дорівнює *n*/2, операцій модулярного множення. Таким чином, загальна кількість мультиплікативних операцій модулярної арифметики запропонованого методу дорівнює числу відповідних операцій класичної реалізації модулярного експоненціювання. Тобто запропонована реалізація не потребує додаткових часових ресурсів у порівнянні з класичною.

Ефект захисту від простого аналізу динаміки споживання потужності в запропонованому алгоритмі досягається за рахунок використання додаткових ресурсів пам'яті. З описаного вище випливає, що об'єм пам'яті, що використовується для тимчасового зберігання операндів віднесених в часі операцій модулярного множення може бути за рахунок відповідного вибору кількості *L* рівнів зменшений відповідно до характерристик конкретної моделі термінального мікроконтролера.

Висновки

За результатами досліджень, націлених на підвищення ефективності захисту ключів від їх незаконної реконструкції аналізом динаміки споживання потужності термінальними мікроконтролерами під час реалізації на них модулярного експоненціювання *n*-розрядних чисел – базової

операції несиметричної криптографії можна зробити наступні висновки.

Аналіз можливостей протидії простому аналізу динаміки споживання потужності показав, що найбільш ефективний шлях порушення залежності між бітами коду експоненти та моментами виконання модулярного множення полягає в віднесенні в часі виконання цієї операції зі збереженням в пам'яті необхідних даних для її реалізації.

Теоретично обґрунтовано, розроблено та досліджено метод обчислення модулярної експоненти, який відрізняється рознесенням в часі обробки j-го розряду коду експоненти, $j \in \{1,2,\ldots,n\}$, та операції модулярного множення, яка співвідноситься з цим розрядом, що не дозволяє визначити за фактом її виконання визначити його значення і тим самим забезпечує захист секретного коду показника від незаконного відновлення за аналізом динаміки споживання потужності термінальним мікроконтролером в процесі виконання на ньому модулярного експоненціювання – базової операції криптографії з відкритим ключем.

Доведено, що запропонований метод забезпечує захист як від простого аналізу динаміки споживання потужності, так і від диференційного аналізу без впливу на час виконання операції модулярного експоненціювання, а за рахунок використання додаткової пам'яті, об’єм якої не є критичним для термінальних мікроконтролерів.

Розроблений метод орієнтовано для застосування систем віддаленого управління на базі *IoT*, до термінальних пристроїв яких можливий сторонній доступ.

Література

1. Meneghello F. et al. IoT: Internet of Threats. A Survey of Practical Security Vulnerabilities in Real IoT Devices. *IEEE Internet of Things Journal*. 2019. Vol. 6, no. 5. P. 8182–8201. DOI: 11.1109/JIOT.2019. 2935189.

2. Mangard S., Oswald E., Popp T. Power Analysis Attacks. Revealing the Secrets of Smart Cards. Springer, 2007. 338 p.

3. Standaert F-X. et al. An Overview of Power Analysis Attack Against Field Programmable Gate Arrays. *Proceeding of the IEEE*. 2006. Vol. 92, no. 2. P. 383–394. DOI: 10.1109/JPROC 2005.862437.

4. Lerman L., Bontempi G., Markowitch O. Power analysis attack: An approach based on machine learning. *International Journal of Applied Cryptography*. 2014. Vol. 3, no. 2. P. 134–148. DOI: 10.1504/IJACT.2014.062722.

5. Kocher P., Jaffe J., Jun B. Differential Power Analysis. *Lecture Notes in Computer Science. Vol. 1666. Advances in Cryptology - CRYPTO '99. 19th Annual International Cryptology Conference, Santa Barbara, California, USA, August 15-19, 1999 Proceedings* / ed. by M. Wiener. Berlin, 1999. P. 388–397.

6. Messerges T. S., Dabbish E. A., Sloan R. H. Power Analysis Attacks of Modular Exponentiation in Smartcards. *Lecture Notes in Computer Science. Vol. 1717. Cryptographic Hardware and Embedded Systems. First International Workshop, CHES'99 Worcester, MA, USA, August 12-13, 1999 Proceedings* / ed. by C. K. Koc, C. Paar. Berlin, 1999. P. 144–157.

7. Clavier C., Joye M. Universal exponentiation algorithm - A first step to toward provable SPA-resistance. *Lecture Notes in Computer Science. Vol. 2162. Cryptographic Hardware and Embedded Systems - CHES 2001 Third International Workshop, Paris, France, May 14-16, 2001 Proceedings* / ed. by C. K. Koc, D. Naccache, C. Paar. Berlin, 2001. P. 300–308.

8. Shanmugham S. R., Paramasivam S. Survey on power analysis attacks and its impact on intelligent sensor networks. *IET Wireless Sensor Systems*. 2018. Vol. 8, no. 6. P. 295–304.

9. Prasad N. D., Avirneni, Somani A. K. Countering Power Analysis Attacks Using Realiable and Aggressive Designs. *IEEE Transaction on Computers*. 2014. Vol. 63, no. 6. P. 1408–1420. DOI: 10.1109/TC.2013.9.

10. Borges J. et al. A Secure Cloud Computing Method for Rapid Implementation of Cryptographic Data Protection in IoT.

2023 13th International Conference on Dependable Systems, Services and Technologies (DESSERT) : proceedings, Athens, Greece, 13–15 October 2023 / IEEE. 2023. P. 674–677. DOI: 10.1109/DESSERT61349.2023.10416477.

11. Русанова О. В., Гайдукевич О. В. Метод розподіленого модулярного експоненціювання на термінальних мікроконтролерах IoT з захищеним залученням хмарних обчислень. *Проблеми інформатизації та управління*. 2024. № 2(78). С. 91–103. DOI: 10.18372/2073-4751. 78.18966.

Русанова О.В., Марковський О.П., Вовк В.В.

МЕТОД МОДУЛЯРНОГО ЕКСПОНЕНЦІЮВАННЯ З ЗАХИСТОМ ВІД АТАК АНАЛІЗОМ ДИНАМІКИ СПОЖИВАННЯ ПОТУЖНОСТІ

В статті запропоновано метод модулярного експоненціювання на термінальних мікроконтролерах, який забезпечує захист від відновлення секретних операндів аналізом динаміки споживання потужності. Метод базується на рознесення в часі обробки бітів коду експоненти і операцій модулярного множення, що з ними співвідносяться за рахунок збереження потрібних для них операндів в пам'яті. Це стає на заваді співставленню виявлених по діаграмі споживання потужності модулярних множень зі значеннями бітів секретного коду експоненти. Викладено формалізовану процедуру модулярного експоненціювання, робота якої ілюстрована прикладом. Розроблено методику вибору параметрів процедуру з урахуванням обмежень на об'єм пам'яті.

Теоретично та експериментально доведено, що запропонований метод забезпечує захист як від простого так і від диференційного аналізу динаміки споживання потужністю без впливу на час обчислення модулярної експоненти.

***Ключові слова**: атаки аналізом споживання потужності; простий аналіз споживання потужності; диференційний аналіз споживання потужності; модулярне експоненціювання,*

Rusanova O.V., Markovskyi O.P., Vovk V.V.

MODULAR EXPONENTIATION METHOD WITH PROTECTION AGAINST POWER ANALYSIS

The article proposes a method of modular exponentiation on terminal microcontrollers, which provides protection against recovery of secret operands by power analysis. The method is based on the separation in time of processing of the exponent code bits and correspondent modular multiplication operations by storing the operands necessary for multiplication in memory. This prevents the comparison of the modular multiplications detected from the power consumption diagram with the values of the secret exponent code bits. A formalized procedure for modular exponentiation is presented, the functionality of which is illustrated by an example. A method for selecting the procedure parameters is developed in view of microcontroller embedded memory limitations.

It has been theoretically and experimentally proved by the proposed method, which provides protection against both simple and differential analysis of the power analysis without affecting the time of calculating the modular exponent.

***Keywords**: power analysis attacks; simple power analysis; differential power analysis; modular exponentiation.*