УДК 004.056.5

Марковський О. П., Русанова О. В., Олієвський А. А. *НТУУ «КПІ ім. Ігора Сікорського», Київ*
Черевик В. М. *Державний університет телекомунікацій, Київ*

МЕТОД ПРИСКОРЕНОГО МОДУЛЯРНОГО ЕКСПОНЕНЦІЮВАННЯ З ВИКОРИСТАННЯМ ПЕРЕДОБЧИСЛЕНЬ

Запропоновано метод прискорення модулярного експоненціювання – базової операції мережевих протоколів захисту інформації. Дано математичне обґрунтування методу та описана процедура обчислення модулярної експоненти з використанням передобчислень. Розроблено рекомендації для вибору оптимальних параметрів програмної реалізації запропонованого методу. Метод забезпечує прискорення обчислення модулярної експоненти приблизно в півтора рази.

***Ключові слова**: комп'ютерна арифметика, модулярне множення, модулярне експоненціювання, передобчислення, протоколи захисту інформації в мережах.*

Markovskyi O. P., Rusanova O. V., Olievskyi A. A. *NTUU "Igor Sykorsky KPI", Kyiv*
Cherevyk V. M. *State University of Telecommunications, Kyiv*

METHOD FOR SPEED UP MODULAR EXPONENT CALCULATION BY USING PRECOMPUTATIONS

The article proposes a method to speed up modular exponentiation A^E mod M – the base operation of network data protection protocols. The proposed method is solving this issue by using precomputation that allows to cut down by half the numbers of operation of modular multiplications on retention of numbers modular quadrate operations. Elaborated method envisages to separate the exponent code E into m-bit length fragments. In context of precomputation all $q = 2^m - 1$ possible values of A^2 mod M, A^3 mod M,…, A^q mod M are calculated. In contrast to existing modular exponentiation method, the proposed one allows to process the whole fragment of exponent code. The article includes mathematical background of the proposed approach. It proves existence of the optimal value for length m fragments of exponent code. The mathematical way to determinate the optimal value of m is presented. Obtained results can be used for optimization of structure modular exponentiation calculation. The results derived by theoretical way have been confirmed by the results of experimental researches that are presented in the article. The proposed procedure of modular exponent calculation with precomputations are described in details and illustrated by numerical example. A recommendation has been worked out for choosing the optimal parameters of software for implementation of the proposed method. A comparative analysis of the proposed methods of modular exponent calculation has been executed. In theoretical and experimental way it is proved that the proposed method provides an acceleration of modular exponentiation by approximately 50%.

***Keywords**: computer arithmetic, modular multiplication, modular exponentiation, precomputation, network data protection protocols.*

Марковский А. П., Русанова О. В., Олиевский А. А. *НТУУ «КПИ им. Игоря Сикорского», Киев*
Черевик В. М. *Государственный университет телекоммуникаций, Киев*

МЕТОД УСКОРЕНИЯ МОДУЛЯРНОГО ЭКСПОНЕНЦИРОВАНИЯ С ИСПОЛЬЗОВАНИЕМ ПРЕДВЫЧИСЛЕНИЙ

Предложен метод ускорения модулярного экспоненцирования – базовой операции сетевых протоколов защиты информации. Дано математическое обоснование метода и описана процедура вычисления модулярной экспоненты с использованием предвычислений. Разработаны рекомендации для выбора оптимальных параметров программной реализации предложенного метода. Метод обеспечивает ускорение вычисления модулярной экспоненты приблизительно в полтора раза.

***Ключевые слова**: компьютерная арифметика, модулярное умножение, модулярное экспоненцирование, предвычисления, протоколы защиты информации в сетях.*

1. Вступна частина

Постановка задачі. Важливою передумовою ефективної взаємодії термінальних пристроїв в комп'ютерних мережах є забезпечення високої продуктивності реалізації ними існуючих мережевих протоколів. Найбільш критичними з точки зору обчислювальної реалізації протоколів мережевого обміну є операції модулярної арифметики, що виконуються над числами великої розрядності, яка значно перевищує розрядність процесорів. На теперішній час, для досягнення прийнятного для більшості застосувань рівня захищеності, необхідна довжина чисел становить 1024-2048 розрядів з перспективою її зростання в найближчі роки до 4096.

Тенденція зростання пропускної здатності каналів передачі даних комп'ютерних мереж вимагає адекватного зростання швидкості реалізації протоколів захисту інформації як на комп'ютерах загального призначення, так і на малорозрядних мікроконтролерах.

Наведені фактори визначають розробку нових підходів до прискорення програмної реалізації операцій модулярної арифметики при їх застосуванні в протоколах захисту інформації, як важливу та актуальну проблему, від вирішення якої значною мірою залежить ефективність локальних та глобальних комп'ютерних мереж.

Аналіз літературних джерел. Базовою обчислювальною операцією широкого кола мережевих протоколів захисту інформації є модулярне експоненціювання, тобто обчислення $A^E \bmod M$, де A, E і M – числа розрядністю n, значно більшою за розрядність процесора.

Процес о модулярного експоненціювання зводиться до послідовного виконання n циклів, у кожному із яких виконується операція модулярного піднесення до квадрату результату операції попереднього циклу і, залежно від поточного біта степені E, здійснюється операція модулярного множення. Залежно від порядку, в якому організована обробка розрядів коду експоненти E існують два базових алгоритми експоненціювання [1]: починаючи зі *старших* або *молодших* розрядів коду експоненти.

Алгоритм обробки розрядів експоненти починаючи зі старших розрядів в нотаціях мови C++ має такий вигляд:

1. $R = 1$.
2. for ($j=n-1; j >=0; j$ --)
{
2.1. $R = R \cdot R \bmod M$
2.2. if ($e^j == 1$)
$R = R \cdot A \bmod M$
}
3. Результат: R.

При цьому, під час кожної ітерації циклу виконується модулярне піднесення числа в квадрат і множення на постійне число, рівне A, що створює потенційні передумови для підвищення швидкості множення. Недоліком є те, що всі операції виконуються строго послідовно й лежать на критичному шляху [2].

Алгоритм, який передбачає обробку розрядів степені E починаючи із молодших розрядів в нотаціях мови C++ має вигляд:

1. $R = 1$, $Q = 1$.
2. for ($j=0; j <n; j$ ++)
{
2.1. $R = R \cdot R \bmod M$
2.2. if ($e_j == 1$)
$Q = Q \cdot R$
}
3. Результат: Q.

Аналіз обох базових алгоритмів показує, що час їх реалізації визначається сумою:

$$n \cdot t_{sq} + 0.5 \cdot n \cdot t_m,$$

де t_{sq} – час виконання операції модулярного піднесення до квадрату;

t_m – час виконання операції модулярного множення [3].

В рамках другого з розглянутих алгоритмів модулярного експоненціювання існує потенційна можливість розпаралелювання обчислень. В роботі [4] описаний варіант такого розпаралелювання. Показано, що на рівні операцій модулярного множення максимальний рівень паралелізму не перевищує 2-х. Відповідно, прискорення обчислювання модулярної експоненти досягається за рахунок використання двох процесорів, один з яких реалізує потік операцій модулярного піднесення до квадрату, а інший – модулярного множення.

Проведений аналіз обчислювальних процедур модулярного експоненціювання показав, що прискорення їх виконання може бути досягнуто за рахунок розпаралелювання на різних рівнях. Більшість робіт [5-7], присвячених вирішенню проблеми прискорення операції модулярного експоненціювання, орієнтовані на рівень процесорних операцій, на які розкладаються операції модулярного піднесення до квадрату і модулярного множення. Зокрема, при експоненціювання 2048-розрядних чисел на 32-розрядних процесорах виконується 128 операцій процесорного множення, які можуть виконуватися паралельно [6].

У свою чергу, час виконання модулярного множення визначається двома складовими: часом, необхідним для реалізації власне множення, і часом, який витрачається на модулярну редукцію, тобто залишку від ділення результату множення на модуль M. У класичному множенні модулярна редукція реалізується з використанням операції ділення і, відповідно, друга складова відіграє значну роль. Значна ефективність обчислювальної реалізації модулярного множення досягається при використанні алгоритму Монтгомері [8], в якому модулярна редукція зводиться до зсуву на k розрядів. Інший підхід до зменшення часу редукції запропоновано в роботі [3]. Цей підхід базується на тому, що на практиці, модуль, що є частиною відкритого ключа, практично не змінюється. Це надає змогу виділити операції, що залежать від модуля, обчислити їх результати один раз і використовувати при кожному модулярному експоненціюванні.

На сьогоднішній день розроблено ряд методів прискореної реалізації модулярного експоненціювання [8, 9], які реалізують можливості розпаралелювання прискорення обчислення модулярної експоненти на рівні процесорних операцій.

Невирішені питання. На основі проведеного аналізу літературних джерел можна зробити наступні висновки. Основним резервом підвищення швидкості обчислювальної реалізації базової операції мережевих протоколів захисту інформації є організація паралельної обробки. Проте цей підхід не можу бути використаний для прискорення модулярного експоненціювання на малопотужних мікроконтролерах – термінальних пристроях широкого кола мереж, що використовуються на практиці. Таким чином, існуючі методи прискорення обчислення модулярної експоненти не вирішують цю проблему для вельми широкого класу обчислювальних пристроїв, які мають підтримувати протоколи мережевого захисту даних.

Мета та задачі дослідження. Мета досліджень полягає в прискоренні виконання критичної для протоколів мережевого захисту інформації операції модулярного експоненціювання.

Для досягнення поставленої мети в роботі розв’язуються такі наукові задачі:

– розробка методу прискореного обчислення модулярної експоненти за рахунок організації обробки декількох розрядів коду експоненти та використання передобчислень, які дозволяють зменшити кількість операцій модулярного множення;

– визначення оптимальних параметрів методу – зокрема кількості розрядів коду експоненти, які оброблюються одночасно, виходячи з критерію досягнення максимуму прискорення обчислення модулярної експоненти;

– теоретичне та експериментальне дослідження ефективності методу прискореної обчислення модулярної експоненти, порівняльний аналіз показників його ефективності з відомими методами реалізації модулярного експоненціювання.

2 Метод прискорення обчислення модулярної експоненти з використанням передобчислень

Для досягнення поставленої мети – прискорення обчислення $A^E \bmod M$ – пропонується розділити n-розрядний код експоненти

$$E = \{e_1, e_2, \ldots, e_n\},\ \forall j=1,2,\ldots,n:\ e_j \in \{0,1\},$$

на n/m m-розрядних фрагментів

$$f_1=\{e_1,e_2,\ldots,e_m\}, f_2=\{e_{m+1},e_{m+2},\ldots,e_{2\cdot m}\},\ldots, f_{n/m}=\{e_{n-m},e_{n-m+1},\ldots,e_n\}.$$

Задля зменшення кількості операцій модулярного множення пропонується перед виконанням процедури модулярного експоненціювання здійснити обчислення m-2 значень:

$$A^2 \bmod M,\ A^3 \bmod M,\ldots,\ A^q \bmod M, \text{ де } q=2^m-1.$$

Отримані результати зберігаються в таблиці передобчислень:

$$T[0]=1,\ T[1]=A,\ T[2]=A^2 \bmod M,\ \ldots,\ T[q]=A^q \bmod M.$$

Після вказаних передобчислень пропонується наступна процедура модулярного експоненціювання.

1) Номеру k поточного фрагменту експоненти присвоюється значення n/m-номеру останнього фрагменту $f_{n/m}$, що містить старші розряди експоненти E: $k=n/m$. Змінна R встановлюється в одиницю: $R=1$.

2) Змінній R присвоюється значенню модулярного добутку її попереднього значення на d_k-е значення з таблиці передобчислень:

$$R = R\cdot T(d_k) \bmod M,$$

де d_k – число, утворене розрядами експоненти, що складають фрагмент

$$f_k : d_k = 2^{m-1}\cdot e_{(k-1)\cdot m}+2^{m-2}\cdot e_{(k-1)\cdot m+1}+\ldots+e_{k\cdot m}.$$

Змінна i встановлюється в одиницю: $i=1$.

3) Виконується модулярне піднесення до квадрату змінної R: $R = R^2 \bmod M$ та інкремент змінної i: $i = i+1$.

4) Якщо $i<m$, здійснюється повернення на повторне виконання п. 3.

5) Виконуються декремент змінної k: $k=k-1$. Якщо $k>1$ повернення на повторне виконання п. 2.

6) Обчислюється модулярний добуток змінної R на значення таблиці передобчислень, що адресується кодом d_1 – числом, утвореним молодшими m розрядами коду експоненти, що складають фрагмент f_1 :

$$R = R\cdot T(d_1) \bmod M = R\cdot T(e_1+2\cdot e_2+\ldots+2^{m-1}\cdot e_m) \bmod M.$$

Отриманий в результаті код $R = A^E \bmod M$.

Запропонована процедура обчислення модулярної експоненти проілюстрована наступним прикладом. Нехай, потрібно обчислити $143^{307} \bmod 311 = 116$, відповідно $A=143$, $M=311$, $E=307 = 100\ 110\ 011_2$. Код експоненти розділяється на три фрагменти по три розряди: $m=3$, $n=9$, $f_1 = \{0,1,1\}$, $f_2 = \{1,1,0\}$, $f_3 = \{1,0,0\}$, $d_1 = 3$, $d_2 = 6$, $d_3 = 4$. $T[0]=1$, $T[2]= A = 143$. Результати передобчислень показані в табл. 1.

Результати передобчислень Табл. 1

d	Табличне значення $T(d)$	
	Змістовне	Числове
0	A^0	1
1	A^1	143
2	A^2 mod M	234
3	A^3 mod M	185
4	A^4 mod M	20
5	A^5 mod M	61
6	A^6 mod M	15
7	A^7 mod M	279

Обробка фрагментів починається зі старшого з них, код якого співвідноситься з числом $d_3 = 4$: при k =3 початкове значення R встановлюється в одиницю. Потім обчислюється його добуток з табличним кодом

$$T(4)=20: \quad R = R\cdot T(d_3)\text{mod}M = 1\cdot T(4)\text{modM} = 20\text{mod}311 = 20.$$

В рамках п. 3-4 виконується три цикли піднесення до квадрату R, так, що отримане значення дорівнює:

$$R = R^8 \text{mod} M = 20^8 \text{mod} 311 = 168.$$

При обробці другого фрагменту, код якого співвідноситься з числом d_2=6, k=2: обчислюється модулярний добуток числа R на табличне значення $T(d_2)$:

$$R = R\cdot T(d_2)\text{mod}M = R\cdot T(6)\text{mod}M = 15\cdot 168\text{mod}311 = 32.$$

Отриманий результат трьома циклами модулярного піднесення до квадрату трансформується до вигляду:

$$R = R^8 \text{ mod } M = 32^8 \text{ mod } 311 = 83.$$

Для молодшого фрагменту d_1= 3. Тому при k=1 у відповідності з п.6 обчислюється модулярний добуток поточного значення R на табличне значення $T(d_1)$:

$$R = R\cdot T(d_1) \text{ mod } M = R\cdot T(3) \text{ mod } M = 185\cdot 83 \text{ mod } 311 = 116.$$

В результаті в змінній R формується код результату 143^{307} mod 311.

3 Аналіз ефективності та оптимізація

Обробка кожного фрагменту коду експоненти зводиться до піднесення попереднього результату в ступінь 2^m та модулярного множення на табличне значення передобчислення, вибір якого визначається кодом фрагменту. Таким чином, час обробки кожного з n/m -1 фрагментів (крім останнього) визначається часом виконання m операцій модулярного піднесення до квадрату та одного модулярного множення. Для останнього фрагменту час його обробки визначається часом виконання однієї операції модулярного множення. Якщо позначити через t_{sq} час виконання операції модулярного піднесення до квадрату, а через t_m – час виконання операції модулярного множення, то загальний час безпосереднього модулярного експоненціювання визначається як сума:

$$(n/m \text{ -1})\cdot m\cdot t_{sq} + n/m \cdot t_m.$$

При виконанні передбачених методом передобчислень 2^m–2 значень $A^2\text{mod}M$, $A^3\text{modM},\ldots,A^q\text{mod}M$ виконується 2^{m-1}–2 операцій модулярного множення і така ж кількість операцій модулярного піднесення до квадрату.

Загальний час T_1 виконання модулярного експоненціювання з урахуванням передобчислень визначається формулою:

$$T_1 = (2^{m-1} - 1)\cdot t_{sq} + (2^{m-1} - 1)\cdot t_m + (n - m)\cdot t_{sq} + \frac{n}{m}\cdot t_{sq}. \qquad (1)$$

Реально операції модулярного множення та модулярного піднесення до квадрату, що виконуються над числами великої розрядності (1024 або 2048), яка значно перевищує розрядність процесора, виконуються шляхом множення секцій співмножників, довжина яких дорівнює розрядності процесора. Якщо позначити через s кількість секцій чисел, то модулярне множення потребує s^2 операцій процесорного множення, оскільки кожна секція множимого множиться на кожну секцію множника. При модулярному піднесенні до квадрату кількість операцій процесорного множення значно менша за рахунок того, що добуток i-тої секції першого числа на j-ту секцію другого дорівнює добутку i-тої секції другого числа на j-ту секцію першого, де $i,j\in\{1,2,\ldots,s\}$. Загальна кількість операцій процесорних множень зменшується до $(s\text{-}1)\cdot s/2$ [3]. Таким чином, загальна кількість операцій процесорного множення при модулярному піднесенні до квадрату зменшується в порівнянні з модулярним множенням в γ раз, де $\gamma = t_m/t_{sq} = 2\cdot s/(s\text{-}1) \approx 2$. Наприклад, при n=1024 число секцій s=32 і чисельне значення γ = 2.06. Виходячи з викладеного, можна вважати, що $t_{sq} \approx 0.5\cdot t_m$.

З урахування наведеного, формула (1) може бути трансформована до наступного вигляду:

$$T_1 = ((2^{m-1} - 1) \cdot \frac{3}{2} + (n - m) \cdot \frac{1}{2} + \frac{n}{m}) \cdot t_m = \xi \cdot t_m \quad , \qquad (2)$$

де ξ – приведене значення кількості операцій модулярного множення.

Отримана формула (2) може бути використана для визначення оптимальної довжини фрагменту при якій значення T_1 досягає мінімуму. Оптимальне значення m' може бути отримане як розв'язання рівняння, яке прирівнює нулю диференціал $T_1(m)$ по m:

$$\frac{dT_{1\cdot}}{dm} = 2^{m-1} \cdot \ln 2 \cdot \frac{3}{2} - \frac{n}{m^2} - \frac{1}{2} = 0 \ . \qquad (3)$$

Чисельний розв'язок рівняння (3) дає такі значення m' для розрядностей n, що реально використовуються в протоколах захисту інформації: для n=1024 оптимальне значення m'=5.57, для n=2048 оптимальне значення m'= 6.45, для n=4096 оптимальне значення m'=7.19.

Вважаючи, що розрядність m фрагменту коду експоненти являє собою ціле, бажано парне число, достатньо просто прорахувати значення формули (2) для обмеженого кола актуальних для задач практики значень n та m. Результати відповідних розрахунків наведено в табл. 2, де представлена залежність коефіцієнту β прискорення обчислення модулярної експоненти $A^E \bmod M$ від довжини n чисел A, E і M та розрядності m фрагментів.

При використанні класичного алгоритму модулярного експоненціювання виконується n операцій модулярного піднесення до квадрату та, у середньому, $n/2$ операцій модулярного множення [6], відповідно, час виконання експоненціювання становить $n \cdot t_{sq} + n/2 \cdot t_m \approx n \cdot t_m$.

Відповідно, для оцінки ефективності запропонованого методу модулярного експоненціювання доцільно використати коефіцієнт прискорення $\beta = n \cdot t_m / T_1 = n/\xi$.

Залежність коефіцієнту β від довжини чисел та розрядності фрагментів Табл. 2

Розрядність n чисел	Розрядність m фрагментів	Приведена кількість ξ операцій множення	Коефіцієнт прискорення β
1024	4	778	1.32
	6	727	1.41
	8	828	1.24
	10	1377	0.74
2048	4	1548	1.32
	6	1410	1.45
	8	1468	1.40
	10	1992	1.02
4096	4	3082	1.33
	6	2776	1.47
	8	2748	1.49
	10	3220	1.27

Аналіз даних таблиці 2 показує, що оптимальне значення m' для всіх розрядностей n, що використовуються на практиці близьке до 8-ми: m' = 8. Використання довжини фрагменту 6 для n=1024, яке теоретично забезпечує максимальний коефіцієнт прискорення, не є ступенем 2 і не дозволяє використовувати фрагменти рівної довжини. При 8-розрядних фрагментах коефіцієнт β прискорення обчислення модулярної експоненти за рахунок використання передобчислень збільшується зі зростанням розрядності n від 1.24 при n=1024 до 1.49 при n=4096.

Подальше прискорення модулярного експоненціювання в рамках пропонованого методу може бути досягнуте за рахунок виконання не всього об'єму передобчислень. Іншими словами виконується передобчислення лише тих степенів A, котрі співпадають з реальними кодами фрагментів експоненти. При цьому, суттєво зменшиться складова часу, зумовлена операціями

передобчислень, причому ступінь μ зменшення залежить від розрядності *n* чисел та довжини *m* фрагменту. Проведенні дослідження показали, що розрядності фрагменту, який відповідає оптимальному значенню m' =8 для *n*=1024 (кількість фрагментів – 127), в середньому потрібно обчислювати в рамках передобчислень лише 99.66 значень, тобто в μ = 99.66/256 = 0.39 рази менше ніж в базовому варіанті. Для *n*=2048 (кількість фрагментів – 256) середня кількість оригінальних значень фрагментів коду експоненти становить 162 з 256 можливих, відповідно значення μ становить: μ = 162/256 = 0.63. Для *n*=4096 значення μ близьке до одиниці.

Технологічно виконати вибіркові передобчислення неважко, оскільки в реальних застосуваннях код експоненти є практично незмінним. При застосуванні в рамках запропонованого методу вибіркових передобчислень час T_2 модулярного експоненціювання визначається формулою:

$$T_2 = ((2^{m-1} - 1) \cdot \frac{3 \cdot \mu}{2} + (n - m) \cdot \frac{1}{2} + \frac{n}{m}) \cdot t_m \quad , \tag{4}$$

Результати розрахунків приведеної кількості модулярних множень, виконані згідно формули (4), а також коефіцієнта β прискорення у порівнянні з класичним алгоритмом модулярного експоненціювання для близьких до оптимального значень *m* зведено до табл. 3, в якій представлена залежність коефіцієнту β прискорення обчислення $A^E \bmod M$ від довжини *n* чисел *A, E* і *M* та розрядності *m* фрагментів при вибіркових передобчисленнях.

Залежність коефіцієнту β від *n* та *m* при вибіркових передобчисленнях Табл. 3

Розрядність *n* чисел	Розрядність *m* фрагментів	Коефіцієнт μ зменшення об'єму передобчислень	Приведена кількість ξ операцій множення	Коефіцієнт прискорення β
1024	4	1	778	1.32
	6	0.92	724	1.42
	8	0.39	710	1.45
	10	0.12	688	1.48
2048	4	1	1548	1.32
	6	1	1410	1.45
	8	0.62	1392	1.47
	10	0.22	1395	1.45
4096	4	1	3082	1.33
	6	1	2776	1.47
	8	1	2748	1.49
	10	0.74	2753	1.48

Аналіз даних таблиці 3 показує, що використання вибіркових передобчислень має наслідком зміщення оптимальних значень m' в бік їх збільшення у порівнянні з теоретичними, які є рішеннями рівняння (3). З даних таблиці 2 випливає, що для всіх розрядностей *n* найбільш доцільним є застосування 8-розрядних фрагментів. При цьому використання вибіркових передобчислень дозволяє прискорити обчислення модулярної експоненти на 20% для *n*=1024 і на 5% для *n*=2048. Додатковим чинником ефективності застосування вибіркових передобчислень є зменшення потрібного об'єму пам'яті для зберігання таблиці передобчислень.

4 Висновки

В результаті проведених досліджень, направлених на пошук шляхів прискорення виконання операції модулярного експоненціювання над числами, розрядність яких значно перевищує розрядність процесора, запропоновано метод обчислення модулярної експоненти з використанням передобчислень.

Розроблений метод передбачає організацію обчислення модулярної експоненти в два етапи: на *першому з*дійснюється передобчислення множини шаблонних значень модулярної експоненти, а на *другому* виконується безпосереднє обчислення модулярної експоненти з множенням проміжних результатів не на число, що підноситься до ступені, а на один з шаблонів, що дозволяє зменшити кількість операцій модулярного множення і, відповідно, прискорити процес модулярного експоненціювання.

Теоретичні та експериментальні дослідження ефективності запропонованого методу довели, що його використання забезпечує прискорення програмної реалізації модулярного експоненціювання приблизно на 50%.

Розроблений метод орієнтовано, насамперед, для термінальних пристроїв мереж – малопотужних мікроконтролерів та смарт-карт, які підтримують мережеві протоколи захисту інформації.

Список використаної літератури

1. Харин Ю. С. Математические основы криптологии / Ю. С. Харин, В. И. Берник, Г. В. Матвеев. – Минск: Изд-во БГУ, 1999. – 319 с.

2. Самофалов К. Г. Эффективная реализация мультипликативных операций модулярной арифметики в системах защиты информации / К. Г. Самофалов, Г. М. Луцкий, А. П. Марковский // Proceeding of International scientific conference UNITECH-09. Gabrovo, November 20-21, 2009. – Technical University of Gabrovo. – 2009. – Vol. 1. – P. 435-437.

3. Bardis N. G. Accelerated modular multiplication algorithm of large word length numbers with a fixed module / N. G. Bardis, A. Drigas, A. P. Markovskyi, I. Vrettaros // Communications in computer and information science 111, knowledge management, information systems, E-learning, and sustainability research. Third world summit on the knowledge society, WSKS 2010, Corfu, Greece. – September 2010. – 2010.

4. Стіренко С. Г. Спосіб прискореного обчислення модулярної експоненти / С. Г. Стіренко, О. П. Марковський, Захаріудакіс Лефтеріс, Л. Д. Міщенко // Вісник Національного технічного університету України «КПІ» Інформатика, управління та обчислювальна техніка. – Київ: ТОО «ВЕК+», 2017. – № 65. – С. 110-115.

5. Самофалов К. Г. Ускоренная реализация модулярного экспоненцирования на малоразрядных микропроцессорах и встроенных микроконтроллерах / К. Г. Самофалов, Рамзи Анвар Салиба Сунна // Проблеми інформатизації та управління. Збірник наукових праць: Випуск 4(15). – Київ: НАУ, 2005. – С. 144-153.

6. Can Xiang. Verifiable and secure outsourcing schemes of modular exponentiations using one untrusted cloud server and their application / Can Xiang // IACR Cryptology e-Print Archive. – 2014. – P. 500-508.

7. Bardis N. G. Secure implementation of modular exponentiation on cloud computing resources / N. G. Bardis, O. P.Markovskyi // Proceeding of International conference applied mathematics, computational science and systems engineering. Athens, Greece, October 6-8, – 2017. – P.90-96.

8. Костенко Ю. В. Метод защищенного модулярного экспоненцирования на удаленных компьютерных системах / Ю. В. Костенко, А. П. Марковский, О. В. Русанова // Вісник Національного технічного університету України «КПІ». Інформатика, управління та обчислювальна техніка. – Київ: ТОО «ВЕК+». – 2016. – № 64. – С. 51-54.

9. Markovskyi O. P. Secure modular exponentiation in cloud systems / Oleksandr P. Markovskyi, Nikolaos Bardis, Nikolaos Doukas, Sergej Kirilenko // Proceedings of The congress on information technology, computational and experimental physics (CITCEP 2015). – December 18-20, 2015. – Krakow, Poland. – P. 266-269.

References

1. Harin J. S., Berdnic V. I., Matveev G. V. "Mathematical basics of cryptology." *Minsk. BGU* (1999): 319.

2. Samofavov K. G., Luckyi G. M., Markovskyi O. P. "Effectiveness organization of multiplicative modular arithmetic operation in data protection systems." *Proceeding of International scientific conference UNITECH-09. Gabrovo* (November 20-21, 2009): 435-437.

3. Bardis N. G., Drigas A., Markovskyi A. P., Vrettaros I. "Accelerated Modular Multiplication Algorithm of Large Word Length Numbers with a Fixed Module." *Communications in computer and information science 111, knowledge management, information systems, e-learning, and sustainability research. Third world summit on the knowledge society, WSKS 2010, Corfu, Greece* (September 10-14, 2010): 573-581.

4. Stirenko S. G., Markovskyi O. P., Zahariydakis L., Michenko L. D. "Methor for speed up modular exponentiation calculate." *Proceeding of National Technical University of Ukraine "KPI". Informatica, control and computer technic* 65 (2017): 110-115.

5. Samofalov K. G. Ramzi Anvar Syliba Sunna. "Accelerated implementation of modular exponentiation on low bit microprocessors, embedded microcontrollers." *Problems of informatics and control* 4(15), (2005):144-153.

6. Can Xiang. "Verifiable and secure outsourcing schemes of modular exponentiations using one untrusted cloud server and their application." *IACR Cryptology ePrint Archive* (2014): 500-508.

7. Bardis N. G, Markovskyi O. P. "Secure implementation of modular exponentiation on cloud computing resources". *Proceeding of International conference applied mathematics, computational science and systems engineering, Athens, Greece* (October 6-8, 2017): 90-96.

8. Kostenko J. V., Markovskyi A. P., Rusanova O. V. "Method for protected modular exponentiation on remote systems." *Proceeding of National Technical University of Ukraine "KPI" Informatica, control and computer technic* 64 (2016): 51-54.

9. Markovskyi O. P., Badis N., Doukas N., Kirilenko S. "Secure modular exponentiation in cloud systems." *Proceedings of The congress on information technology, computational and experimental physics (CITCEP 2015), Krakow, Poland* (18-20 December 2015): 266-269.

Автори статті

Марковський Олександр Петрович – кандидат технічних наук, доцент кафедри обчислювальної техніки, Національний технічний університет України «Київський політехнічний інститут ім. Ігора Сікорського», Київ, Україна. Тел.+380(96) 710 85 34. E-mail: markovskyy@i.ua.

Русанова Ольга Веніамінівна – кандидат технічних наук, доцент кафедри обчислювальної техніки, Національний технічний університет України «Київський політехнічний інститут ім. Ігора Сікорського», Київ, Україна. Тел. +380 (97) 410 56 87 E-mail: olga.rusanova.v@gmail.com.

Олієвський Андрій Анатолійович – студент кафедри обчислювальної техніки, Національний технічний університет України «Київський політехнічний інститут ім. Ігора Сікорського», Київ, Україна. Тел. +380 (96) 855 96 04. E-mail: androlyc@gmail.com.

Черевик Вячеслав Михайлович – кандидат технічних наук, професор кафедри комп'ютерної інженерії, Державний університет телекомунікацій, Київ. Тел.: +380 (68) 357 88 48. E-mail: wmcherevik@ukr.net.

Authors of the article

Markovskyi Olexsandr Petrovych – candidate of science (technical), docent of computer technical department, National Technical University of Ukraine "Igor Sykorsky Kiev Polytechnic Institute", Kyiv, Ukraine. Tel.: +380(96) 710 85 34. E-mail: markovskyy@i.ua

Rusanova Olha Veniaminivna – candidate of science (technical), docent of computer technic department, National Technical University of Ukraine "Igor Sykorsky Kiev Polytechnic Institute", Kyiv, Ukraine. Tel.: +380(97) 410 56 87. E-mail: olga.rusanova.v@gmail/com

Olievskyi Andii Anatoliiovych – student of computer technic department, National Technical University of Ukraine "Igor Sykorsky Kiev Polytechnic Institute", Kyiv, Ukraine. Tel.: +380 (96) 855 96 04. E-mail: androlyc@ gmail.com

Cherevyk Viacheslav Mykhailovych – candidate of science (technical), professor of computer sciences department, State University of Telecommunications, Kyiv. Tel.: +380 (68) 357 88 48. E-mail: wmcherevik@ukr.net.

Дата надходження в редакцію: 04.01.2018 р.

Рецензент:
доктор технічних наук, професор К. С. Козелкова
Державний університет телекомунікацій, Київ