КІБЕРБЕЗПЕКА: освіта, наука, техніка
CYBERSECURITY: EDUCATION, SCIENCE, TECHNIQUE
№ 1 (21), 2023
ISSN 2663 - 4023
DOI 10.28925/2663-4023.2023.21.273285
УДК 004.77

Черненко Роман Миколайович
аспірант кафедри інформаційної та кібернетичної безпеки
імені професора Володимира Бурячка
Київський університет імені Бориса Грінченка, Київ, Україна
ORCID 0000-0002-1439-961X
r.chernenko.asp@kubg.edu.ua

ОЦІНКА ПРОДУКТИВНОСТІ АЛГОРИТМІВ ЛЕГКОЇ КРИПТОГРАФІЇ НА ОБМЕЖЕНИХ 8-БІТНИХ ПРИСТРОЯХ

Анотація. Існують різні алгоритми шифрування які можуть бути реалізовані на обмежених пристроях, проте не всі з них є ефективними. Використання неефективних алгоритмів захисту може призвести до недостатнього рівня захисту інформаційних систем та порушенням їх роботи через брак необхідних ресурсів. Тому, розробка нових моделей захисту даних, що передаються відкритими каналами зв'язку обмеженими пристроями є важливою задачею для забезпечення безпеки інформаційної системи. В роботі розглянуті вимоги до алгоритмів легкої криптографії, сформовані показники вимірювання продуктивності. У статті аналізуються, з точки зору продуктивності та ефективності на пристроях класу 0, з 8-бітними процесорами, сучасні алгоритми легкого шифрування. Згідно з проведеним аналізом, після проведення досліджень та експериментів виявлено, що алгоритм HIGHT демонструє найвищу швидкість шифрування, при цьому споживаючи набільше серед протестованих оперативної пам'яті. Алгоритм XTEA має середні показники за всіма показниками, та в цілому є збалансованим між швидкістю шифрування та необхідними обчислювальними ресурсами для функціонування. Фіналіст конкуру NIST Isap, та переможець, що планується до стандартизації Ascon, демонструють низькі показники ефективності на 8-бітних обмежених пристроях класу 0, через те, що при їх розробці в якості цільової платформи були визначені 64-бітні процесори. PRESENT, в свою чергу, не є ефективним через значні обсяги використовуваних ресурсів та низьку швидкість шифрування.

Ключові слова: Інтернет речей; IoT; мережева безпека; пристрої з обмеженими обчислювальними ресурсами; алгоритми шифрування; ефективність; пропускна здатність.

ВСТУП

Інтернет речей це величезна мережа пристроїв, що активно розвивається. До неї входять різноманітні пристрої від розумних автомобілів до акумуляторних мініатюрних датчиків. Ці пристрої значно відрізняються в плані обчислювальної потужності, а також щодо їх швидкості передачі даних і ємності оперативної пам'яті. IoT можна розглядати як велику екосистему, населену високо диверсифікованими та неоднорідними пристроями, які мають різні форми та розміри. Тому не дивно, що існує десятки різних (і в основному несумісних) мікроконтролерних платформ, операційних систем та стандартів бездротової комунікації для IoT, багато з яких оптимізовані для обслуговування певної області застосування із специфічними вимогами та обмеженнями.

Ця неоднорідність пристроїв IoT різко контрастує з «монокультурою» в мережах класичних комп'ютерів, таких як ПК або ноутбуки, де 64-розрядні архітектури Intel та AMD мають частку ринку практично 100%. Тим не менш, 64-розрядні процесори Intel та AMD представляють лише невелику частку IoT в цілому, в якому (кількісно) домінують мікроконтролери з досить скромними обчислювальними можливостями.

У відповідності до аналізу глобального ринку мікроконтролерів [1] у 2021 році ринок поділений як зображено на рис. 1. При цьому загальний ринок мікроконтролерів продовжить свій ріст із середньорічним показником в 10.10% в період з 2022 року по 2028, при цьому ріст ринку 8 бітних контролерів буде складати 4.7%. Проте найбільший ріст звичайно будуть демонструвати 32 бітні контролери через зниження їх ціни і вимоги в обчислювальних ресурсах.

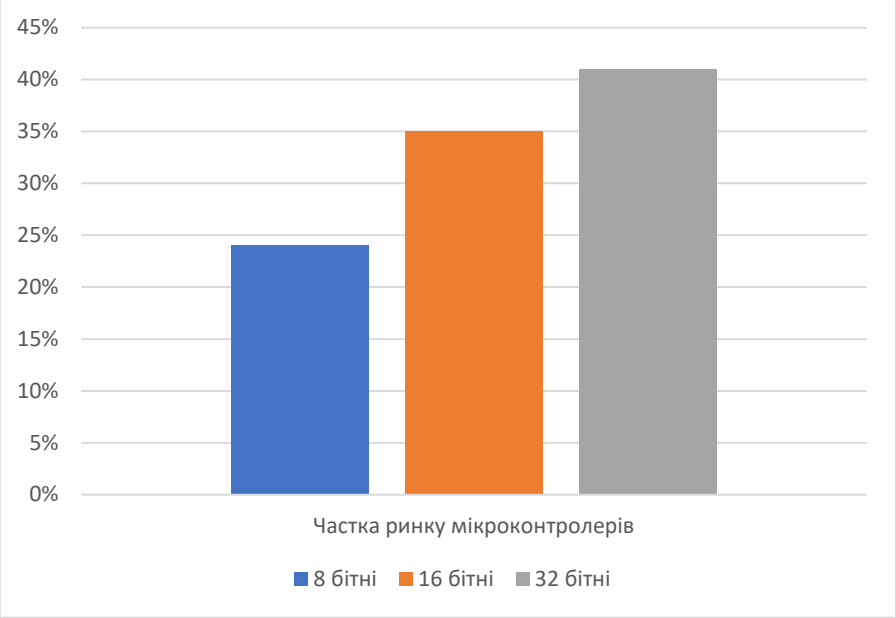

Рис. 1. Ринок мікроконтролерів станом на 2021 рік

Проте ринок 8 бітних контролерів продовжує свій розвиток, окрім цього ці дані лише демонструють продажі мікроконтролерів, та не враховують скільки їх вже існує на сьогоднішній день. Виходячи з такого можна зробити висновок, що більшість контролерів є 8 або 16 бітними, тобто існує необхідність в розробці алгоритмів та методів для захисту дуже обмежених пристроїв, через їх велику кількість та позитивний ріст в майбутньому.

Постановка проблеми. Однією з передових ідей для забезпечення криптозахисту таких пристроїв є «легка криптографія (LWC)». Легка криптографія — це криптографічні алгоритми, методи або протоколи, адаптовані для реалізації в обмежених середовищах, включаючи RFID-мітки, датчики, бездротові смарт-карти, медичні пристрої тощо [2].

Можна вважати, що ідея легкої криптографії з'явилась з початком застосування пристроїв на яких не можуть бути реалізовані стандарти безпеки розроблені для повноцінних комп'ютерів.

Привід для розвитку легкої криптографії (LWC) в основному випливає з її прямого застосування в реальному світі, оскільки вона надає рішення для реальних проблем, з якими стикаються проектувальники систем Інтернету речей. Широко кажучи, легкі криптографічні алгоритми розроблені для досягнення двох основних цілей. Перша мета криптографічного алгоритму — протистояти всім відомим криптоаналітичним атакам і бути таким чином захищеним у моделі чорного ящика. Друга мета — побудувати криптографічний примітив таким чином, щоб його реалізації відповідали чітко визначеному набору обмежень, які залежать від конкретного випадку [3].

Зазвичай до методів і алгоритмів легкої криптографії ставляться такі вимоги [4]:

1. Алгоритми повинні працювати значно краще в обмежених середовищах (апаратні та вбудовані програмні платформи) порівняно із звичайними алгоритмами;
2. Вони повинні бути оптимізовані для ефективної роботи з короткими повідомленнями (наприклад, довжиною до 8 байт);
3. Повинні мати компактні апаратні реалізації та вбудовані програмні реалізації з низьким використанням ОЗУ та ПЗУ;
4. Алгоритми повинні бути гнучкими, щоб підтримувати різні стратегії реалізації (низька енергія, низька потужність, низька затримка);
5. Продуктивність на мікроконтролерах повинна враховувати широкий спектр 8-бітних, 16-бітних і 32-бітних мікроконтролерів архітектур;
6. Для алгоритмів, які мають ключ, попередня обробка ключа (з точки зору часу обчислення та розміру пам'яті) повинна бути ефективною.

Окрім цього при розробці таких методів допускається пріоритизація продуктивності над іншими параметрами.

Оскільки в Інтернеті речей немає єдиної домінуючої платформи мікроконтролерів, важливо, щоб криптографічні алгоритми досягали стабільної продуктивності на широкому спектрі архітектур 8, 16 і 32 біт. Це далеко не проста задача, оскільки, наприклад, 32-бітний мікроконтролер ARM Cortex-M3 має значні архітектурні та мікроархітектурні відмінності в характеристиках, ніж 8-бітний мікроконтролер AVR ATmega. Перший має 16 регістрів, з яких 14 доступні для загального користування, тобто простір загального призначення становить 448 біт. З іншого боку, мікроконтролери AVR мають 32 загальні регістри, але кожен з них може вмістити лише 8 біт даних, що означає, що робочий простір регістрів становить 256 біт. ARM і AVR також значно відрізняються у своїй здатності виконувати багатобітні зсуви та ротації, які є критичними для продуктивності операціями багатьох симетричних криптосистем. Арифметико-логічний блок мікроконтролерів ARM має швидкий ковзний перемикач, який може зсунути або повернути 32-бітне слово на довільний набір біт за один такт. Крім того, зсув або поворот можна поєднати з багатьма іншими арифметико-логічними інструкціями, що означає, що багатобітні зсуви та ротації в основному безкоштовні на ARM. Для архітектур 8 і 16 біт ситуація зовсім інша, оскільки більшість з них мають лише однобітні інструкції зсуву та повороту, що означає, що зсув або поворот регістра на n біт займає (принаймні) n + 1 такт. Таким чином, виконання зсуву або повороту 32-бітного слова на AVR у гіршому випадку може вимагати декількох десятків тактів. Це автоматично робить алгоритми, що були розроблені для 64 та 32 бітних систем, менш ефективними на 8 та 16 бітних системах, через те, що окрім значної кількості ресурсів таким процесорам необхідно виконувати більше операцій.

Це зумовлює необхідність розробки «легких» методів та алгоритмів шифрування, які могли б виконуватись на пристроях з обмеженими обчислювальними ресурсами, особливо на пристроях класу 0 та могли забезпечити надійний рівень захисту даних.

Аналіз останніх досліджень і публікацій. Дослідження і розробка методів легкої криптографії для використання на пристроях Інтернету речей з обмеженими ресурсами швидко розвивалися протягом останнього десятиліття. Основною метою є створення і використання простих криптографічних алгоритмів, які можуть бути застосовані до таких додатків, забезпечуючи при цьому належний рівень безпеки. Для аналізу було відібрано декілька сучасних та відомих алгоритмів легкого шифрування, для визначення їх ефективності функціонування на пристроях класу 0, з 8 та 16 бітними процесорами.

PRESENT — це блочний шифр, орієнтований на апаратне забезпечення, вперше запропонований у 2007 році на Конференції з криптографічного обладнання та вбудованих систем (CHES). PRESENT вибраний до аналізу через його швидку взаємодію та через те, що 2012 році організація ISO та IEC виділили алгоритми PRESENT та CLEFIA у міжнародному стандарті обмеження доступу ISO/IEC 29192-2:2012 [5].

PRINCE [6] — це легкий блочний шифр, опублікований на конференції Asiacrypt 2012 року. Він оптимізований для низької затримки при реалізації в апаратному забезпеченні. Побудований на основі конструкції, подібної Even-Mansour (так званої конструкції FX [7]), і має цікаву особливість: розшифровування можна виконати, повторно використовуючи процес шифрування з дещо іншим ключем. Ця особливість, так звана властивість α-відображення, дає явну перевагу в реалізаціях, що вимагають як шифрування, так і розшифровування. Проте такий підхід зменшує рівень безпеки порівняно з ідеальним шифром, таким чином автори заявили, що безпека шифру гарантується до 2^{127}-n операцій, коли проводиться 2n запитів на шифрування/розшифровування. Це обмеження є дійсним лише для моделі з одним ключем, і автори не зробили жодних заяв щодо моделі пов'язаного ключа.

HIGHT (High security and lightweight) [8] — це ультралегкий алгоритм, який обробляє 64-розрядний блок з 128-бітним ключем за 32 раундів за допомогою компактної функції раунду (без S-блоків) і простих обчислювальних операцій. Найкомпактніша версія використовує 2608 GE для пропускної здатності 188 Кбіт/с [9]. 29 грудня 2006 року був затверджений стандартним алгоритмом шифрування в Південній Кореї [10] Телекомунікаційним Технологічним Об'єднанням (TTA), Південна Корея, за номером стандартизації TTAS.KO-12.0040.

Фіналіст конкурсу NIST-**Isap** — це сімейство шифрів з автентифікацією на основі одноразового ключа з асоційованими даними (AEAD), розроблених з акцентом на стійкість до пасивних атак на стороні каналу [11]. Усі члени сімейства Isap є алгоритмами, що базуються на перестановці, які поєднують варіанти заснованого на губці режиму Isap з одним із декількох опублікованих легких перестановок.

Для аналізу також було обрано відомий алгоритм **Extended Tiny Encryption (XTEA)** — легкий алгоритм, побудований на основі 64-бітної блокової мережі Фейстеля, алгоритм шифрування XTEA був розроблений з оригінального TEA тими ж авторами як розширення, в якому він був вказаний як цінна та інноваційна альтернатива для підвищення безпеки, як доповнений операціями перемішування ключів. Його невеликий розмір коду та низькі вимоги до зберігання дозволяють використовувати його для операцій шифрування програмного забезпечення, які зазвичай розміщуються на невеликих вбудованих системах.

Також було відібрано дуже важливий для аналізу алгоритм **Ascon**, через його перемогу в конкурсі від NIST LWC, та плани щодо стандартизації його як рекомендованого алгоритму для обмежених пристроїв.

Принцип його роботи побудований на конструкції губки розміром 320 біт (що складається з п'яти 64-бітних слів x0, . . . , x4). Ascon існує в двох варіантах, Ascon128 і Ascon-96, з різними рівнями безпеки та параметрами. Для аналізу був обраний саме алгоритм Ascon-128.

Мета статті. Мета статті полягає у аналізі продуктивності відомих криптографічних алгоритмів розроблених для пристроїв з обмеженими обчислювальними ресурсами, на пристроях класу 0, з 8-бітним процесором.

ISSN 2663 - 4023

РЕЗУЛЬТАТИ ДОСЛІДЖЕННЯ

Для реалізації алгоритмів обрано мікроконтролер ATMEGA328p на платформі Arduino (рис. 2). Це дуже розповсюджений контролер для навчання, прототипування та комерційного застосування. Його перевагами є простота в програмуванні та відкритість платформи. Він програмується на мові С з додатковими програмними блоками для керування периферією платформи.

Рис. 2. Мікроконтролер ATMEGA328p на платформі Arduino

ATmega328P — це однокристальний мікроконтролер сімейства megaAVR з архітектурою 8-бітного процесора RISC, що працює на частоті 16 МГц. Має 32 кілобайти вбудованої пам’яті (ROM) і 2 кілобайти оперативної пам’яті (RAM). Такі характеристики повною мірою відповідають класу пристроїв 0, тому є гарним варіантом для проведення оцінки роботи алгоритмів. Живлення контролеру для всіх тестів крім енергоефективності було від USB 3.0 гнізда. Для проведення тесту на енергоефективність мікроконтролер було заживлено від лабораторного джерела живлення з напругою 7 вольт постійного струму через спеціальний інтерфейс живлення на платі.

Для проведення експерименту були обрані «найлегші» реалізації шифрів, в яких залишено лише частину коду з шифруванням та розшифруванням та змінні необхідні для роботи. Алгоритми реалізовані мовою С.

Зважаючи на таке, а також проаналізувавши інші подібні дослідницькі роботи вимоги до ресурсів для програмного забезпечення та апаратного забезпечення можуть бути виміряні з точки зору розміру ключа, розміру блоку тексту, вимог до пам'яті, площі реалізації, затримки, пропускної здатності, споживання енергії та споживання енергії на байт наступним чином:

Вимоги до пам’яті

Зазвичай, вимірюється в KB [12]. RAM потрібно для зберігання проміжних значень, які можуть використовуватися в обчисленнях, а ROM потрібно для зберігання програми/алгоритму та статичних даних, таких як ключ алгоритму, S-box (якщо використовується) тощо [13]. Також можна додати пам’ять для зберігання коду без компіляції.

Площа реалізації (для апаратної реалізації)

Це фізична площа, необхідна для реалізації/запуску алгоритму на платі/схемі, вимірюється в μm². Цей простір може бути задано за допомогою логічних блоків для FPGA або за допомогою GE для ASIC (1GE = 2 вхідний NAND Gate). Зазвичай 200–2000 GE (з 1000–10 000 GE загальної доступної) виділяються для цілей безпеки в економічній мітці RFID [14].

Пропускна здатність

Пропускна здатність, в апаратному забезпеченні, може вимірюватися в термінах обробленого тексту за одиницю часу (бітів за секунду). Пропускна здатність може бути обчислена за формулою:

$$\frac{Ps}{t} \text{ біт/с},$$

де Ps — розмір тексту в бітах, t — витрачений час.

Затримка

Це час, необхідний для отримання шифру з оригінального тексту з точки зору продуктивності апаратного забезпечення [13], тоді як кількість тактів на блок (під час шифрування) визначає затримку програмного забезпечення. Затримку швидкості шифрування або розшифрування можна обчислити використовуючи дані отримані з обчислення пропускної здатності за формулою:

$$\frac{f}{\frac{Ps}{t}/Bs} = \frac{f*Bs*t}{Ps} \text{ (циклів/блок)},$$

де f — частота процесора в герцах, Bs — розмір блока в байтах.

Вимоги до енергії

Кількість енергії, що вимагається схемою для обробки алгоритму, може вимірюватися в мікроватах або в джоулях за секунду.

Споживання енергії на біт

Споживання енергії на біт можна розрахувати наступним чином [12]:

$$Energy[\mu J] = (Latency[cycles/block] * Power[\mu W])/blocksize[bits]$$

У цій формулі затримка використовується в термінах програмної реалізації.

Ефективність

Визначення продуктивності над вимогами до ресурсів. Для апаратного забезпечення вона може бути розрахована за формулою [12]:

$$Hardware\ Efficiency = Throughput[Kbps]/Complexity[KGE]$$

У цій формулі складність означає фізичний простір.

Аналогічно, ефективність програмного забезпечення може бути визначена [12]:

$$Software\ Efficiency = Throughput[Kbps]/CodeSize[KB]$$

У цій формулі розмір коду — це розмір реалізації алгоритму.

Проте при проведені аналізу та оцінки, необхідно враховувати, що допускається компроміс між ефективністю та рівнем захисту, тобто ефективність має більший пріоритет[15].

Таким чином для визначення недоліків та переваг існуючих рішень для захисту обмежених пристроїв, необхідно провести тестування за визначеними параметрами.

Вимоги до пам’яті

Мікроконтролер ATMEGA328p має 32 Кб енергонезалежної пам’яті яка в основному використовується для зберігання програм. Вимірювання зайнятої програмою

місця в пам'яті відбувалось за допомогою вбудованих функцій середовища програмування Arduino ide. Для тестування використовувались мінімальні реалізації алгоритмів, тільки функції шифрування/розшифрування та всі необхідні параметри для їх роботи, результат зображено на рис. 3. Таким чином більше всього місця в пам'яті займає реалізація шифру PRESENT, найменше HIGHT та XTEA. Проте варто зауважити, що розмір програм не перевищує 10% від доступної пам'яті, що є прийнятним. Окрім цього за класифікацією до пристроїв класу 0 належать пристрої в яких об'єм ROM пам'яті до 100 кб, тому всі алгоритми споживають не багато пам'яті залишаючи достатньо місця для корисного навантаження.

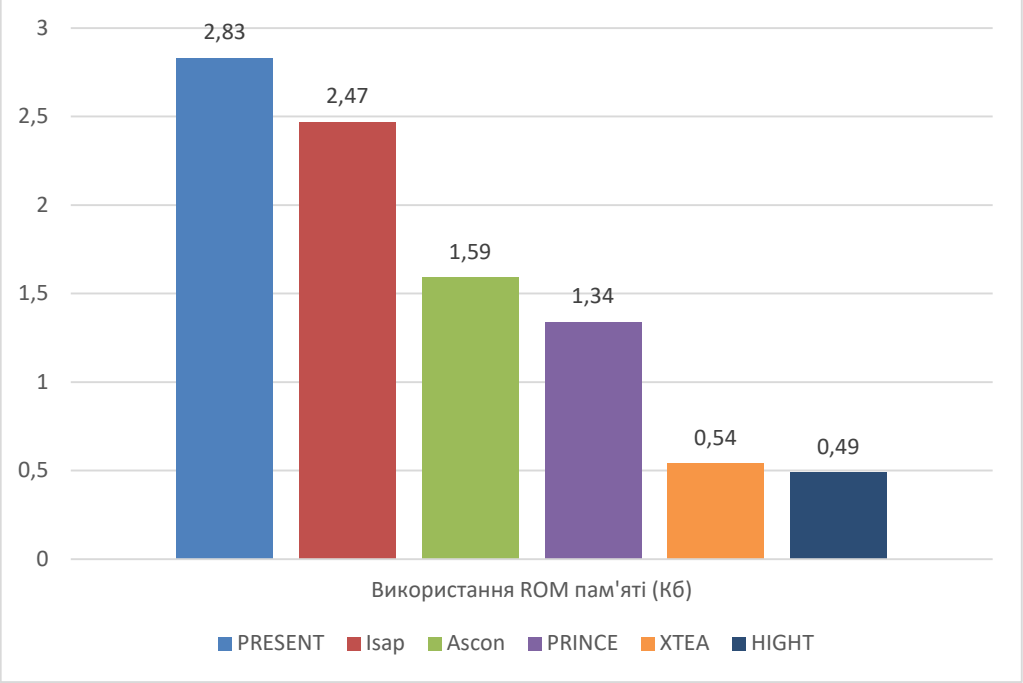

Рис. 3. Використання ROM пам'яті

Під час аналізу споживання RAM пам'яті виявлено, що найбільше оперативної пам'яті використовують алгоритми HIGHT та Isap (рис. 4). Оперативна пам'ять є більш критичним параметром, оскільки в ній зберігаються всі змінні в таких структурах як стек та куча. Пам'ять на мікроконтролерах є критичною ланкою, оскільки якщо в процесі роботи відбудеться її переповнення, наприклад через великі масиви даних, програмний код може працювати непередбачувано, що поставить під загрозу не тільки безпеку, а і правильне виконання команд. На платформі ATMEGA328p доступно 2 кб оперативної пам'яті. Таким чином найефективнішим алгоритмом за цим критерієм виявився XTEA.

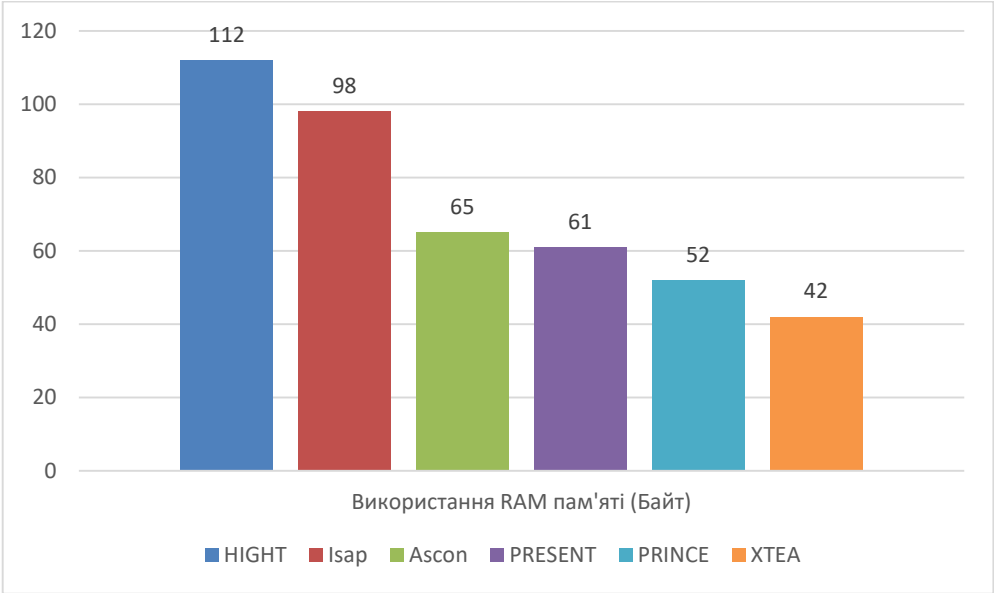

Рис. 4. Використання RAM пам'яті

Показник пропускної здатності (рис. 5), тобто те скільки даних може бути зашифровано за одну секунду, дозволяють побачити реальну ефективність алгоритму. Найшвидшим алгоритмом згідно тестування виявився HIGHT. Проте важливо відмітити, що він при цьому споживає найбільше оперативної пам'яті. Такі шифри як Ascon та Isap, показали дуже низьку швидкість, це зумовлено тим, що вони розроблялись для 64-бітних пристроїв, та мають надто малу продуктивність на 8-бітних та 16-бітних пристроях класу 0, яким є в тому числі ATMEGA328p.

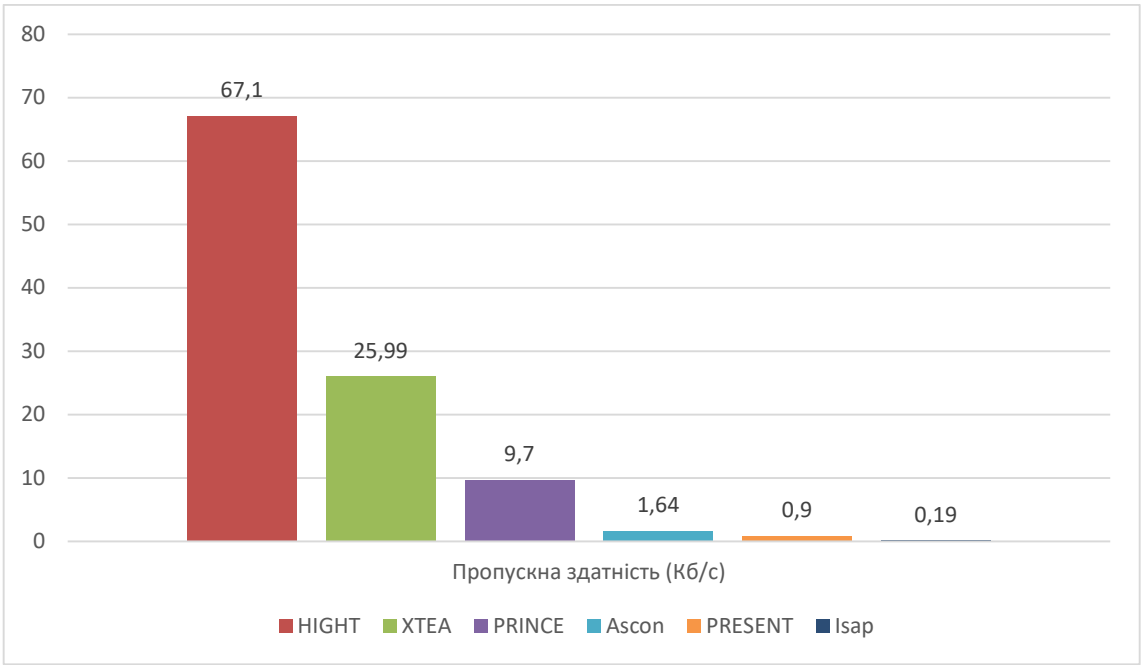

Рис. 5. Пропускна здатність

Програмна затримка (рис. 6) виглядає «дзеркально» до пропускної здатності, адже визначає час необхідний для шифрування блока даних, з поточною швидкістю та частотою процесора. Через це HIGHT знову має найкращий результат.

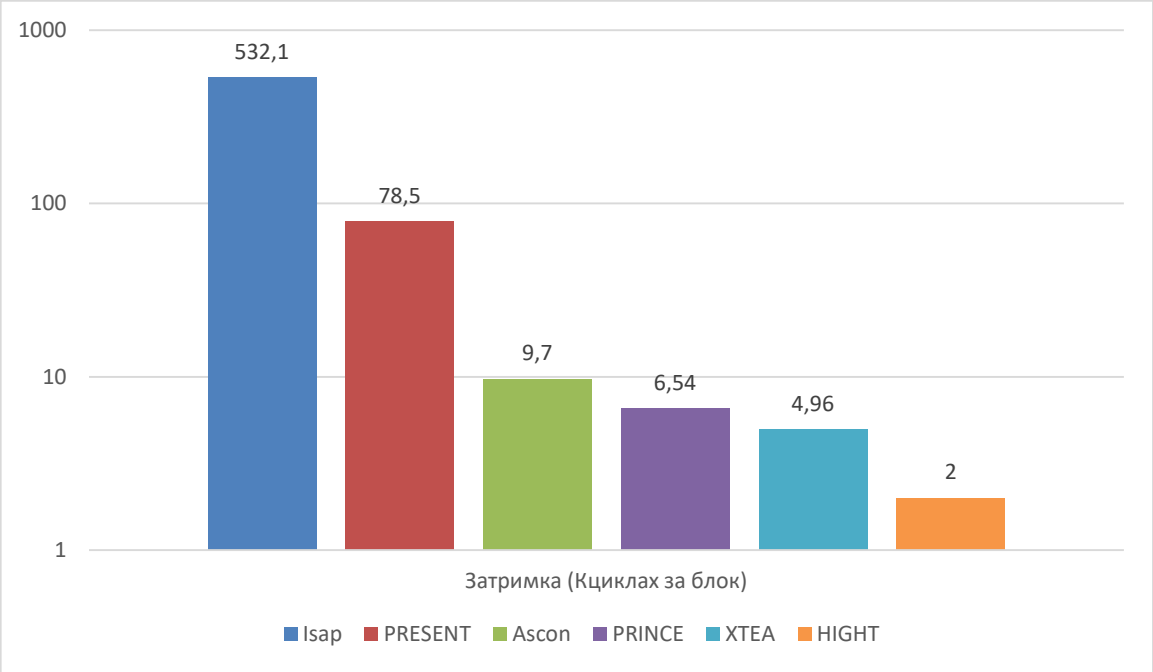

Рис. 6. Програмна затримка

Аналізуючи кількість енергії, що споживається пристроями (рис. 7), можна зробити висновок, що алгоритми Isap та PRESENT споживають надто велику кількість енергії в порівнянні з іншими, при цьому вони демонструють достатньо низьку швидкість на пристроях класу 0. Такий висновок можна зробити і щодо алгоритму Ascon, кількість спожитої ним енергії є значною по відношенню до низької швидкості шифрування. Найменшу кількість енергії споживають алгоритми HIGHT, PRINCE та XTEA.

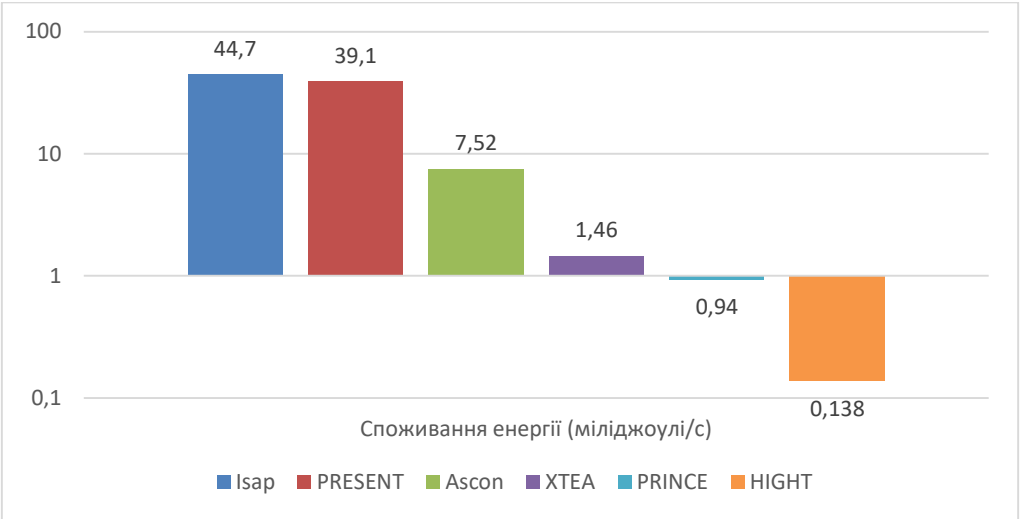

Рис. 7. Споживання енергії за секунду

Якщо аналізувати з точки зору спожитої енергії на біт (рис. 8) результат приблизно такий самий. Алгоритми Isap та PRESENT демонструють надто велике споживання як для швидкості з якою вони шифрують інформацію. При цьому HIGHT демонструє кращий результат через високу швидкість шифрування.

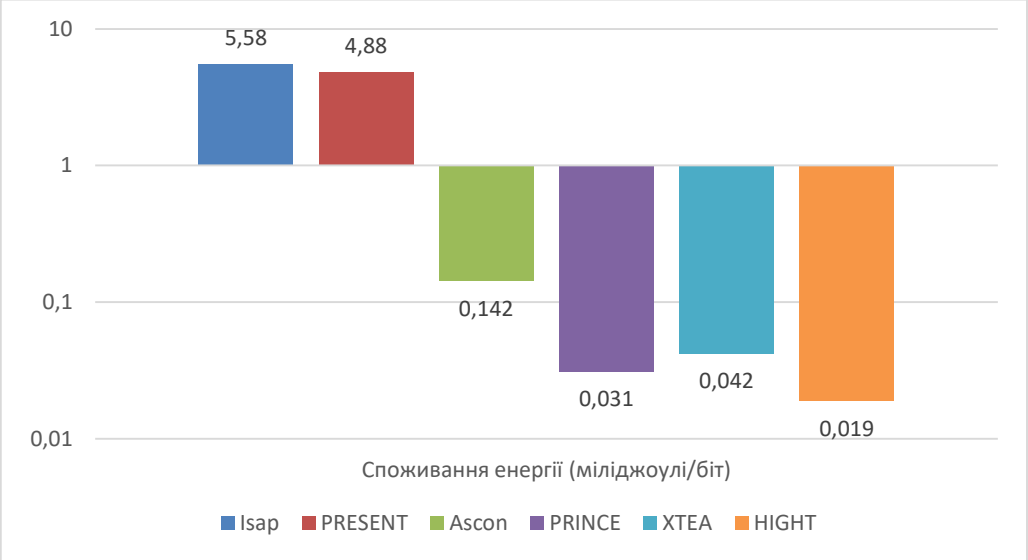

Рис. 8. Споживання енергії на шифрування одного біта

За показником ефективності програмного забезпечення (рис. 9) лідирує HIGHT через те, що цей показник залежить від швидкості шифрування та розміру коду, проте він потребує дуже багато оперативної пам'яті. Алгоритм XTEA має стабільний показник ефективності. Решту алгоритмів можна вважати мало ефективними на пристроях класу 0.

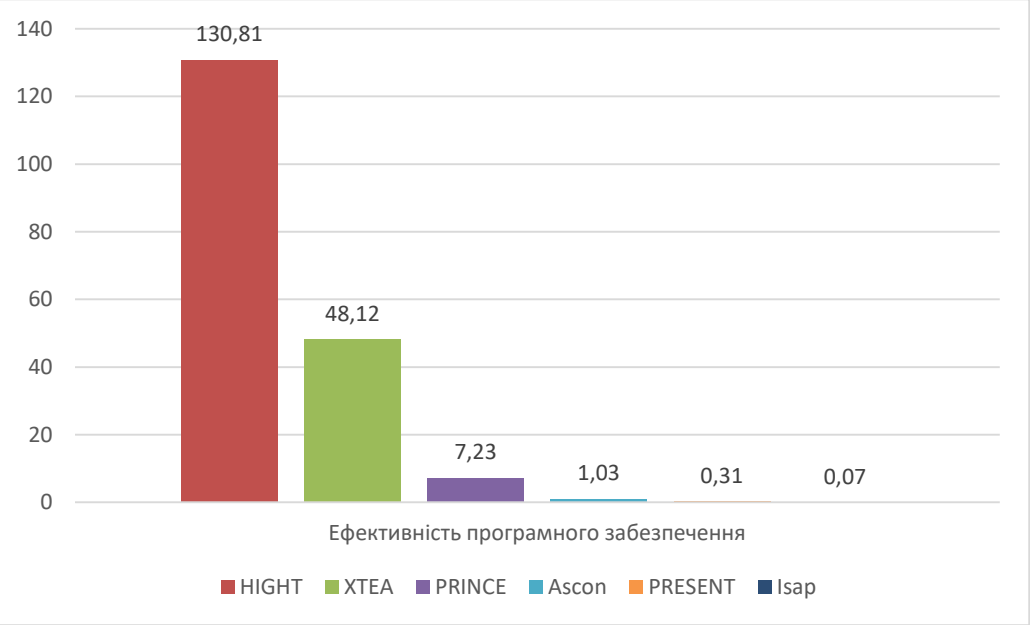

Рис. 9. Ефективність програмного забезпечення

ВИСНОВКИ ТА ПЕРСПЕКТИВИ ПОДАЛЬШИХ ДОСЛІДЖЕНЬ

Відповідно до аналізу найефективнішим алгоритмом з точки зору швидкості шифрування є HIGHT, проте він вимагає значного обсягу оперативної пам'яті у порівнянні з рештою алгоритмів. Окрім цього він має деякі вразливості. Найбільш збалансованим алгоритмом за всіма параметрами можна визнати XTEA при малому розмірі реалізації та споживання пам'яті він демонструє гарний показник швидкості. Алгоритм PRINCE демонструє значно нижчі показники ефективності в порівнянні з XTEA при майже схожих вимогах до ресурсів. Що стосується алгоритмів фіналіста конкурсу NIST Isap, та переможця, що планується до стандартизації Ascon, вони демонструють низькі показники ефективності на 8-бітних пристроях з обмеженими обчислювальними ресурсами класу 0, через те, що при їх розробці в якості цільової платформи були визначені 64-бітні процесори. Щодо алгоритму PRESENT він є надто неефективним через велику кількість ресурсів і надто малу швидкість шифрування.

Таким чином розроблюваний метод криптографічного захисту, з точки зору ефективності на пристроях класу 0, необхідно порівнювати з алгоритмами HIGHT – який найшвидший, XTEA – найзбалансованіший та Ascon – переможець конкурсу NIST.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. *Microcontroller Market Size to Reach USD 69.08 Bn by 2032.* (2023). Precedence Research - Market Research Reports & Consulting Firm. https://www.precedenceresearch.com/microcontroller-mcu-market
2. Katagi, M. & Moriai, S. (2012). Lightweight Cryptography for the Internet of Things. *Sony Corporation*.
3. Ibrahim, N. & Agbinya, J. (2022) A Review of Lightweight Cryptographic Schemes and Fundamental Cryptographic Characteristics of Boolean Functions. *Advances in Internet of Things, 12,* 9–17. https://doi.org/10.4236/ait.2022.121002.
4. *Submission Requirements and Evaluation Criteria for the Lightweight Cryptography Standardization Process* (2018). NIST. https://csrc.nist.gov/csrc/media/Projects/lightweight-cryptography/documents/final-lwc-submission-requirements-august2018.pdf
5. *Information technology — Security techniques — Lightweight cryptography* (29192-2:2012). (2021). https://www.iso.org/standard/56552.html
6. Borghof, J., et al., (2012). PRINCE—A Low-Latency Block Cipher for Pervasive Computing Applications. *International Conference on the Theory and Application of Cryptology and Information Security,* 208-225. https://doi.org/10.1007/978-3-642-34961-4_14
7. Biryukov, A.,. (2011). DES-X (or DESX). *Encyclopedia of Cryptography and Security.*
8. Hong, D., et al. (2006). HIGHT: A new block cipher suitable for low- resource device. *International Workshop on Cryptographic Hardware and Embedded Systems*, 46–59.
9. Lim, Y.-I., et al. (2009). Implementation of HIGHT cryptic circuit for RFID tag, *IEICE Electronics Express, 6(4),* 180–186. https://doi.org/10.1587/elex.6.180
10. South Korea Telecommunications Technology Associations (TTA) (2006). *64-bit Block Cipher HIGHT*. (TTAS.KO-12.0040).
11. Wen, L., et al. (2014). Multidimensional zero- correlation attacks on lightweight block cipher HIGHT: improved crypt- analysis of an ISO standard, *Information Processing Letters, 114(6)*, 322–330.
12. Hatzivasilis, G., et al. (2018). A review of lightweight block ciphers, J. Cryptograph. Eng., 8(2), 141–184.
13. McKay, K., et al. (2017) Report on Lightweight Cryptography (Nistir8114). NIST.
14. Juels, A., & Weis, S.A. (2005). Authenticating pervasive devices with human protocols, Proc. 25th Annu. Int. Cryptol. Conf. 293–308. https://link.springer.com/chapter/10.1007/11535218_18
15. *Submission Requirements and Evaluation Criteria for the Lightweight Cryptography Standardization Process* (2018). NIST. https://csrc.nist.gov/csrc/media/Projects/lightweight-cryptography/documents/final-lwc-submission-requirements-august2018.pdf

КІБЕРБЕЗПЕКА: освіта, наука, техніка
CYBERSECURITY: EDUCATION, SCIENCE, TECHNIQUE
№ 1 (21), 2023
ISSN 2663 - 4023

Roman Chernenko
graduate student of the Department of Information and Cyber Security
named after Professor Volodymyr Buriachok
Borys Grinchenko Kyiv University, Kyiv, Ukraine
ORCID 0000-0002-1439-961X
r.chernenko.asp@kubg.edu.ua

PERFORMANCE EVALUATION OF LIGHTWEIGHT CRYPTOGRAPHY ALGORITHMS ON CONSTRAINED 8-BIT DEVICES

Abstract. Various encryption algorithms can be implemented on constrained devices; however, not all of them are efficient. Employing inefficient security algorithms may lead to insufficient protection levels for information systems and disrupt their functionality due to lack of necessary resources. Therefore, developing new data protection models for transmitting information through open communication channels using constrained devices is a crucial task for ensuring information system security. This paper outlines the requirements for lightweight cryptography algorithms and establishes performance measurement metrics. The article analyzes, in terms of performance and efficiency on class 0 devices with 8-bit processors, modern lightweight encryption algorithms. According to the conducted analysis, research, and experiments, it has been found that the HIGHT algorithm demonstrates the highest encryption speed while consuming the most RAM among the tested algorithms. The XTEA algorithm has average performance metrics across all indicators and is generally balanced between encryption speed and required computational resources for operation. The NIST Isap finalist and the anticipated standardization winner, Ascon, show low efficiency on 8-bit constrained class 0 devices because they were developed targeting 64-bit processors. On the other hand, PRESENT is not efficient due to significant resource usage and low encryption speed.

Keywords: Internet of Things; IoT; network security; constrained devices; encryption algorithms; efficiency; throughput.

REFERENCES (TRANSLATED AND TRANSLITERATED)

1. *Microcontroller Market Size to Reach USD 69.08 Bn by 2032.* (2023). Precedence Research - Market Research Reports & Consulting Firm. https://www.precedenceresearch.com/microcontroller-mcu-market
2. Katagi, M. & Moriai, S. (2012). Lightweight Cryptography for the Internet of Things. *Sony Corporation*.
3. Ibrahim, N. & Agbinya, J. (2022) A Review of Lightweight Cryptographic Schemes and Fundamental Cryptographic Characteristics of Boolean Functions. *Advances in Internet of Things, 12,* 9–17. https://doi.org/10.4236/ait.2022.121002.
4. *Submission Requirements and Evaluation Criteria for the Lightweight Cryptography Standardization Process* (2018). NIST. https://csrc.nist.gov/csrc/media/Projects/lightweight-cryptography/documents/final-lwc-submission-requirements-august2018.pdf
5. *Information technology — Security techniques — Lightweight cryptography* (29192-2:2012). (2021). https://www.iso.org/standard/56552.html
6. Borghof, J., et al., (2012). PRINCE—A Low-Latency Block Cipher for Pervasive Computing Applications. *International Conference on the Theory and Application of Cryptology and Information Security,* 208-225. https://doi.org/10.1007/978-3-642-34961-4_14
7. Biryukov, A.,. (2011). DES-X (or DESX). *Encyclopedia of Cryptography and Security.*
8. Hong, D., et al. (2006). HIGHT: A new block cipher suitable for low- resource device. *International Workshop on Cryptographic Hardware and Embedded Systems*, 46–59.
9. Lim, Y.-I., et al. (2009). Implementation of HIGHT cryptic circuit for RFID tag, *IEICE Electronics Express, 6(4),* 180–186. https://doi.org/10.1587/elex.6.180
10. South Korea Telecommunications Technology Associations (TTA) (2006). *64-bit Block Cipher HIGHT*. (TTAS.KO-12.0040).
11. Wen, L., et al. (2014). Multidimensional zero- correlation attacks on lightweight block cipher HIGHT: improved crypt- analysis of an ISO standard, *Information Processing Letters, 114(6)*, 322–330.
12. Hatzivasilis, G., et al. (2018). A review of lightweight block ciphers, J. Cryptograph. Eng., 8(2), 141–184.
13. McKay, K., et al. (2017) Report on Lightweight Cryptography (Nistir8114). NIST.

14. Juels, A., & Weis, S.A. (2005). Authenticating pervasive devices with human protocols, Proc. 25th Annu. Int. Cryptol. Conf. 293–308. https://link.springer.com/chapter/10.1007/11535218_18
15. *Submission Requirements and Evaluation Criteria for the Lightweight Cryptography Standardization Process* (2018). NIST. https://csrc.nist.gov/csrc/media/Projects/lightweight-cryptography/documents/final-lwc-submission-requirements-august2018.pdf