DOI 10.18372/2310-5461.50.15670

УДК 519.212.2:681.51: 621.317

С. В. Поперешняк, канд. фіз.-мат. наук, доц.
Київський національний університет імені Тараса Шевченка
orcid.org/0000-0002-0531-9809
e-mail: spopereshnyak@gmail.com;

О. О. Райчев
Київський національний університет імені Тараса Шевченка
orcid.org/0000-0002-4085-5711
e-mail: mileenocktopus@gmail.com

МОДЕЛЬ ЛЕГКОВАГОВОГО ГЕНЕРАТОРА ПСЕВДОВИПАДКОВИХ ЧИСЕЛ ДЛЯ ІНТЕРНЕТУ РЕЧЕЙ

Вступ

Завдяки сучасному розвитку технологій і високому рівню їх інтеграції у різні сфери людської діяльності з'являється необхідність створення програмного забезпечення для різних видів пристроїв з різними операційними системами та обчислювальними можливостями. Звісно, характеристики пристрою чи системного програмного забезпечення не повинні впливати ні на якість, ні на продуктивність, ні на безпеку, що надається самим пристроєм та програмним забезпеченням. Одним з «вузьких місць» сучасних IoT пристроїв є криптографія, адже ці пристрої здебільшого працюють в умовах обмежених обчислювальних можливостей, на що впливають такі фактори як розмір, можливість автономної роботи та навіть розмір батареї. Більшість алгоритмів шифрування базуються на доволі складних розрахунках і генерації випадкових чисел, що не може не впливати на необхідні характеристики пристрою. Використання програмного продукту для створення та тестування легковагових генераторів випадкових чисел може внести неоціненний вклад в розвиток не тільки IoT, а і всієї криптографічної сфери.

Створення будь-якого пристрою або програми починається з дизайну і визначення ключових факторів та обмежень, які дозволять продукту розвиватися у правильному напрямку та надати очікувані результати. Для генераторів випадкових чисел, першим важливим фактором є постановка задачі яку повинен вирішувати генератор [1].

Постановка проблеми

Парадигма інтернету речей (IoT) базується на взаємозв'язках між всюдисущими та гетерогенними мережевими «речами», такими як датчики, виконавчі механізми, смартфони тощо, поширене використання яких можна віднести до передового розвитку комунікацій, сенсорних технологій та мереж можливості, мобільні пристрої та хмарні обчислення тощо [2]. Оскільки в багатьох випадках IoT пов'язаний із повсякденним життям користувача, вимоги щодо безпеки та конфіденційності повинні бути виконані [3]. Однак традиційні заходи безпеки не можуть бути безпосередньо застосовані до IoT через різні стандарти та задіяні комунікаційні технології. Велика кількість обмежених ресурсами вузлів, що зазвичай використовуються в інтернеті речей, потребує використання легких криптографічних примітивів. Легка криптографія складається з групи криптографічних примітивів, призначених для обмежених середовищ [4]. Термін середовище, обмежене ресурсами, використовується для опису платформи розробки, що має зменшений проектний простір. Наприклад, в апаратних реалізаціях для оцінки легких властивостей слід враховувати розмір мікросхеми, енергоспоживання (наприклад, час автономної роботи), апаратну пам'ять, затримку обчислень, пропускну здатність зв'язку тощо. Усі легкі примітивні реалізації апаратних засобів намагаються виконати свої основні функціональні можливості, використовуючи мінімальний розмір апаратних кісток. Ефективність впровадження залежить від складності конструкції, використовуваної технології, пропускної здатності та енергоспоживання. Складність обладнання визначається кількістю логічних шлюзів, необхідних для реалізації генераторів псевдовипадкових чисел.

У разі реалізації програмного забезпечення необхідно звертати увагу на час виконання, споживання оперативної пам'яті (розмір оперативної пам'яті) та розмір коду. Як наслідок, дизайнери обмежені наявними ресурсами і часто ви-

бирають мінімалістичний підхід, не враховуючи ризиків для безпеки, які це може представляти для потенційних користувачів. Справжні генератори випадкових чи псевдовипадкових чисел (СГВЧ та ГПВЧ, відповідно) є двома найважливішими будівельними блоками криптосистем. Вони використовуються для генерування конфіденційних ключів, викликів та довіри. Вони також використовуються в протоколах автентифікації та навіть у контрзаходах проти апаратних атак [5]. У обмежених пристроях програм IoT важко досягти криптографічно захищених ГПВЧ через апаратні/програмні обмеження. В оглядовій літературі можна знайти лише декілька описів цих ГПВЧ, а деякі з них мають проблеми з безпекою [6].

Викладене в цьому розділі доволі повно описує актуальність проблеми легковагових генераторів в IoT пристроях, тому саме для вирішення цієї проблеми буде створено генератор.

Аналіз останніх досліджень і публікацій

В оглядовій літературі запропоновано кілька легковагових криптографічних примітивів для забезпечення безпеки пристроїв, обмежених ресурсами. Розглянемо криптографічно захищені конструкції генераторів псевдовипадкових чисел (ГПВЧ) для пристроїв, обмежених ресурсами.

У праці [2] запропоновано легковаговий ГПВЧ, який належить до сімейства ГПВЧ Trifork [7], структура якого підходить для обмежених ресурсів пристроїв. Структура даного генератора складається з двох зв'язаних відсталих генераторів Фібоначчі, взаємно шифрованих, придатних для забезпечення переважної більшості програм IoT, таких як програми, що використовуються в смарт-картах, тегах ідентифікації радіочастот та бездротових вузлах датчиків.

У праці [8] автори розробили та впровадили СГВЧ, криптографічний генератор псевдовипадкових чисел, який використовує отримані бітові помилки як джерело випадковості у вузлах бездротових датчиків. У праці [9] автори представили вдосконалену версію СГВЧ, запропоновану в праці [10], яка використовує вимірювання, отримані від бездротових вузлів датчиків, як джерела фізичної випадковості. Їх метод використовує розподілений алгоритм виборів лідерів для вибору випадкового джерела даних. Крім того, була оцінена надійність алгоритму СГВЧ проти кількох атак.

Генератори псевдовипадкових чисел для недорогих інтелектуальних пристроїв, таких як вузли датчиків, було представлено в праці [11]. Він базується на поєднанні модифікованих блоків Бруйна та регістра зсуву нелінійних зворотних зв'язків. Два запропоновані екземпляри підходять для захисту недорогих смарт-пристроїв. У [12] важлива відмінна атака на все сімейство ГПВЧ потокових шифрів показує, що майже кожен член цієї родини вразливий до лінійних атак; це може загрожувати безпеці.

Генератор псевдовипадкових чисел з назвою LAMED був представлений в праці [13] для додатків RFID-міток. Його конструкція заснована на алгоритмі генетичного програмування і має внутрішній стан 64 біти, з 32-бітовим ключем і 32-бітним початковим вектором. Модульна алгебра, побітові операції XOR та обертання бітів - основні операції, що використовуються для оновлення внутрішнього стану ГПВЧ. Було запропоновано дві версії генератора. Перший — це 32-розрядний ГПВЧ, а другий — 16-розрядний ГПВЧ. Для перевірки випадковості генераторів використовувались набори статистичних випробувань NIST, ENT та Diehard. Аналіз апаратної складності обох версій генератора підтверджує, що він відповідає вимогам, встановленим недорогою технологією [14], [15].

У праці [16] було запропоновано J3Gen ГПВЧ на основі попередньої роботи в [17]. J3Gen поєднує в собі СГВЧ із тепловим шумом та регістр зсуву динамічного лінійного зворотного зв'язку (DLFSR) з n комірок і має чотири основних блоки: СГВЧ на основі генератора, архітектуру DLFSR, логіку декодування та селектор поліномів. Приблизна апаратна складність цього ГПВЧ підходить для обмежених пристроїв. Розмір ключа захисту відповідає 372 бітам. ГПВЧ J3Gen був успішно підданий криптоаналізу Peinado [6], який показав уразливість алгоритму за допомогою імовірнісної атаки та детермінованої атаки. Перша дозволяє відновити набір поліномів зворотного зв'язку, які становлять секретну інформацію ГПВЧ. Остання дозволяє зловмисникові відновити всю вихідну послідовність ГПВЧ, знаючи лише кілька бітів послідовності.

МЕТА — у роботі запропоновано фізичну модель IoT генератора, яка на своєму прикладі надає широкий огляд факторів та обмежень, що виникають під час проєктування генераторів.

Вибір типу генератора псевдовипадкових чисел

Наступним важливим питанням є вибір між типами генераторів. Генератор псевдовипадкових чисел має свої переваги для поставленого завдання, адже для його роботи необхідна тільки схема, що буде виконувати алгебричну функцію та деяка пам'ять за допомогою якої можна зберегти початкове або проміжне значення алгоритму [1]. Недоліком цього типу генераторів є

те, що він створює випадкові числа набагато гіршої якості (з більшою детермінованістю).

Апаратні генератори випадкових чисел (АГВЧ) потребує значно більшої кількості технічного забезпечення — йому необхідна більша кількість пам'яті (для збереження проміжних даних), більш вимогливий процесор, різноманітні датчики (для вимірювання фізичних процесів), а іноді навіть підсилювач. Перевагою цього типу є те, що створювані числа мають значно нижчу ступінь детермінованості.

Вибір між генераторами є доволі адекватним питанням, не зважаючи на погані характеристики випадковості ГПВЧ, адже залежно від умов навіть ці генератори можуть надавати достатній рівень безпеки. Однак, якщо вирішення проблеми направлене на високі стандарти криптографічної безпеки і відповідно низьку детермінованість, вибір падає на АГВЧ.

Наступним питанням дизайну, що стосується саме АГВЧ, є вибір сенсорів що будуть використовуватися для отримання випадкових чисел з навколишнього середовища. Залежно від існуючих варіантів використання генератора, необхідно чітко визначитися з датчиками, адже датчик світла не буде корисним, якщо пристрій використовується в темному приміщенні, а датчик руху або акселерометр буде надлишковим для стаціонарного пристрою. Це є дуже суттєвим питанням при розробці реальних девайсів, але для створюваної моделі генератора це не є вирішальним фактором.

Іншими не менш важливими при для IoT пристроїв характеристиками є вага, розміри, інтеграція з програмним забезпеченням та іншими пристроями тощо.

Не зважаючи на серйозність цих факторів при розробці пристроїв для кінцевого користувача, робота базується на тому щоб показати більш ефективний підхід до створення легковагових генераторів з використанням створеного програмного продукту для тестування та налагодження, тому ними можна частково чи повністю знехтувати.

Дизайн генератора випадкових чисел

На дизайн генератора більше за все вплинула постановка задачі. Так, типом генератора для моделі було обрано АГВЧ, завдяки тому, що його характеристики дозволять генерувати більш випадкові послідовності. Це рішення в свою чергу викликає необхідність створення фізичної моделі що буде складатися з мікросхеми та декількох сенсорів [1].

Макетну схему та реальну модель генератора можна побачити на рис. 1 та рис. 2 відповідно.

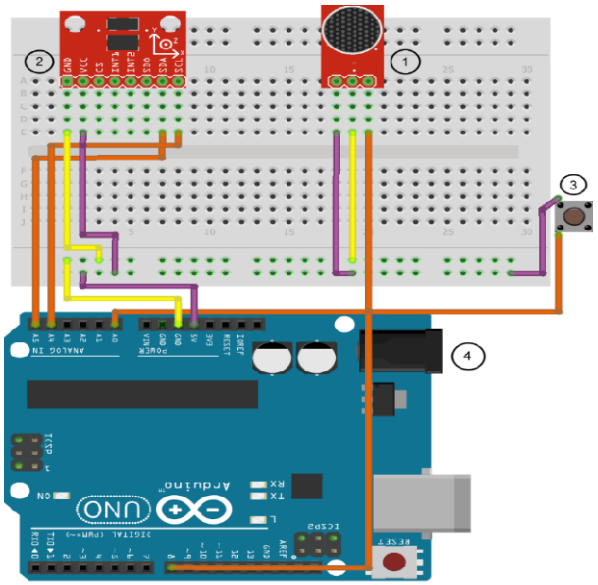

Рис. 1. Макетна схема генератора

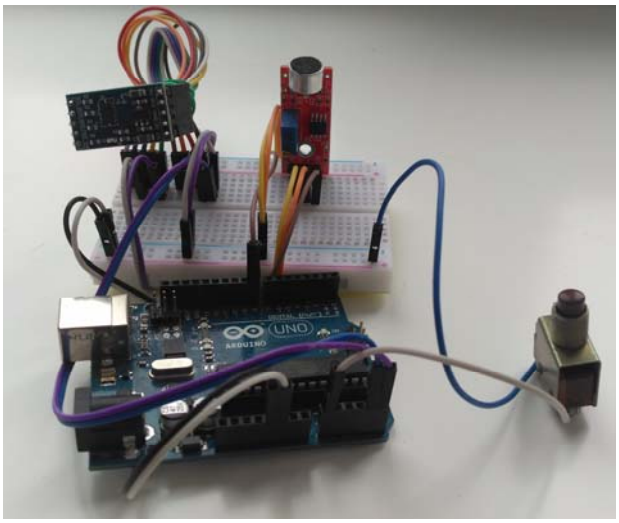

Рис. 2. Фізична модель елементів генератора

За основу генератора та мікросхеми було вирішено використати плату Arduino. Факторами що вплинули на цей вибір є:

- можливість використовувати високорівневу мову програмування;
- наявність великої кількості сенсорів та додаткових пристроїв що можуть бути використані для створення моделі;
- широка підтримка та велика кількість навчальних матеріалів;
- можливість легко вносити зміни та модифікації в модель;

Подібність Arduino до контролеру (порівняно з такими пристроями як Raspberry Pi), а отже більша подібність моделі до реальної схеми.

Використання Arduino надає широкий вибір сенсорів що можуть бути використані для отримання випадкових чисел з навколишнього середовища. Для моделі було обрано датчик звуку та акселерометр. Основним факторами вибору є:

- легкість збору даних у звичному середовищі;
- висока придатність даних сенсорів до використання в генераторах випадкових чисел [18];
- широке використання даних сенсорів у сучасних пристроях.

Дана схема не відображає повну методику підключення елементів між собою, але дає високорівневий огляд компонентів та взаємодії між ними. Різними кольорами на схемі позначено загальні види підключень елементів. Фіолетовий колір — напруга, жовтий — заземлення, а помаранчевий — передача даних.

На реальній моделі можна побачити сенсори, під'єднанні до макетної дошки, саму плату Arduino та кнопку. Положення елементів на моделі відповідають їх положенням на макетній схемі [1].

На схемі наявні такі компоненти:

- Датчик звуку — схема що містить два головних елементи — потенціометр та мікрофон. Потенціометр використовується для налаштування чутливості вимірів, що також може мати свої переваги при генерації випадкових чисел (наприклад, зміна чутливості залежно в навколишнього середовища). Мікрофон використовується щоб вимірювати інтенсивність звуку. За рахунок замірів інтенсивності звуку в рівних інтервалах часу можна згенерувати випадкові числа. Даний датчик підключено до цифрового входу Arduino і він повертає 0 або 1 (0 або 5 вольт) залежно від того, чи перевищує інтенсивність звуку налаштований за допомогою потенціометра рівень;
- акселерометр — схема що використовується для вимірювання сили реакції індукованої прискоренням або гравітацією по трьом осям. Її можна використати щоб зібрати випадкові дані коли пристрій змінює орієнтацію або піддається вібрації чи ударам. Для передачі даних до Arduino використовується два аналогових входи на які приходить напруга від 0 до 1023 вольт. Щоб отримати більш «випадкові» заміри, різницю між ними можна збільшити за допомогою множення на константу;
- кнопка — тригер, що використовується для виклику події генерації випадкового числа. Коли кнопка притиснута, на аналоговий вхід Arduino надається сигнал. У реальному контролері це б виконувалося іншою схемою, що контролює схему генерації випадкових чисел, але для задач моделювання достатньо і такого підходу.
- Arduino — головна схема генератора. Вона підключається до джерела струму і забезпечує всі інші елементи необхідним струмом. Arduino також містить програмний код, який виконується генератором та дозволяє поєднувати всі компоненти схеми для виконання спільної задачі створення випадкових чисел.

Програмне забезпечення генератора випадкових чисел

Схема генератора створена за допомогою Arduino містить всі необхідні компоненти для генерації чисел, але без програмного коду, їх інтеграція між собою не є можливою. Від алгоритму який буде використано також залежить якість генератора. Ці проблеми буде висвітлено в наступному розділі, а зараз розглянемо високорівневий огляд роботи генератора за допомогою блок-схеми (рис. 3).

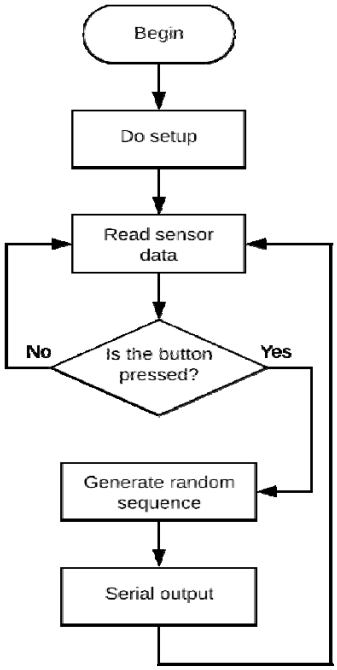

Рис. 3. Блок-схема алгоритму програми генератора

Першим, що може здивувати в блок-схемі, є відсутність кінцевого стану. Це пов'язано зі специфікою Arduino і її подібністю до контролера — програма складається з двох циклів, перший з яких виконується при запуску, а другий є основним циклом і виконується допоки програма не буде перервана.

На самому початку виконання програми генератора виконується ініціалізація, що включає вибір швидкості серійної комунікації та встановлюється з'єднання з акселерометром [1].

Після цього, дані з окремих сенсорів записуються в окремі ділянки пам'яті, з яких вони потім будуть зчитуватися для генерації випадкової послідовності. Якщо кнопка була натиснута, то виконується генерація випадкового числа, що передається на стандартний вивід. У реальній схемі виведення буде виконуватися до іншого пристрою, а тригером може бути електричний сигнал з третього пристрою, але принцип залишається тим самим.

Генерація випадкових чисел для тестування

Перш ніж генерувати випадкові числа, модель генератора повинна виконувати відповідну програму [1]. Модифікувавши програму невеликим чином (для виведення даних), та використовуючи засоби для розробки програм Arduino, можна побачити зміну даних з датчиків в часі (рис. 4). Червоний графік відповідає звуку, а помаранчевий, сірий та чорний — осям *x*, *y* та *z* акселерометра відповідно.

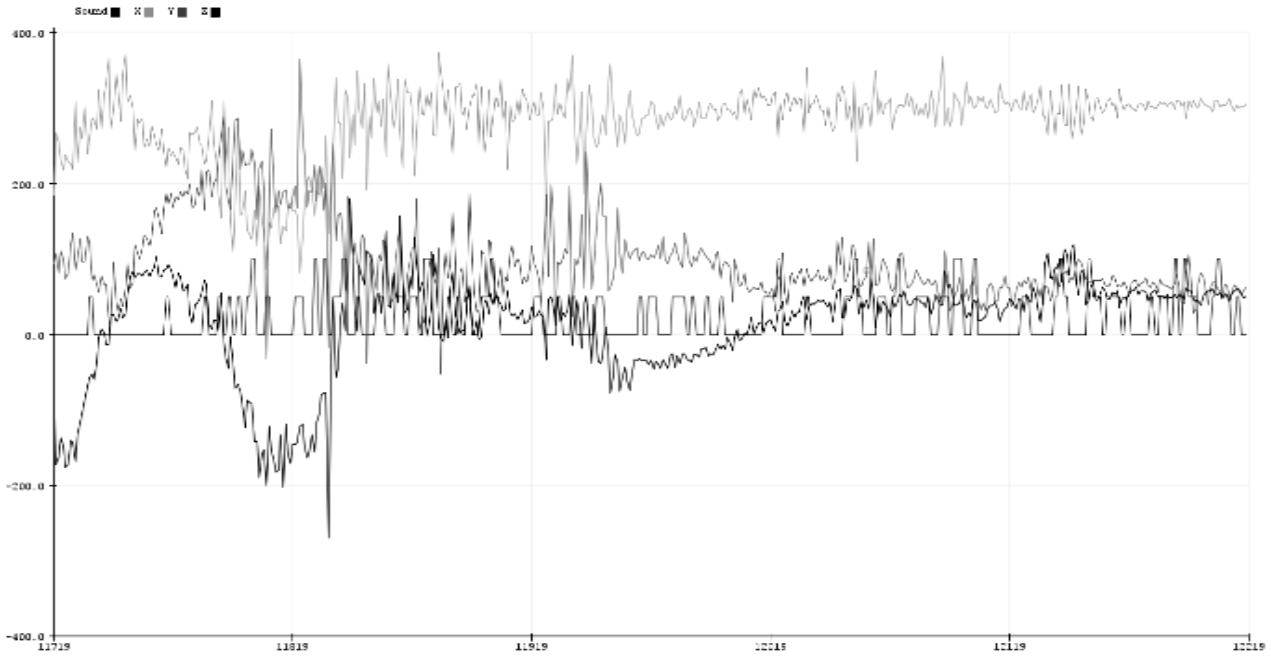

Рис. 4. Зміна вимірів з датчиків в часі

Рис. 4 повною мірою показує переваги АГВЧ — дані є випадковими і вони слабо базуються на попередніх вимірах. Останнім питанням, що може виникнути на шляху до створення якісних випадкових послідовностей, є стійкість алгоритму до відсутності потоку даних або внесення алгоритмом детермінованості в генеровані числа. Ці питання, а також якість створеної моделі можна перевірити за допомогою створеного пакету програм.

Перевірка не була б повною без використання якогось альтернативного методу генерації, адже таким чином можна не тільки показати придатність моделі до реального використання, а і довести що вона показує кращі результати за інші методи генерації.

Щоб забезпечити повну перевірку, буде використано те ж середовище Arduino, але генерація випадкових чисел буде проходити за допомогою вбудованої функції random(), що використовує ГПЧ.

Для цієї задачі створена програма, лістинг якої представлено на рис. 5.

```
void setup() {
  Serial.begin(9600);
}

void loop() {
  long value = random(2147483647);
  Serial.println(toBinary(value + value));
  delay(1000);
}

String toBinary(long n) {
  String r;
  while (n != 0) {
    r = (n % 2 == 0 ? "0" : "1") + r;
    n /= 2;
  }
  return r;
}
```

Рис. 5. Альтернативна програма з ГПВЧ

Варто додати, що обидва генератори створюють послідовності довжини 31 біт, що в більшості мов програмування відповідає типу integer, а в Arduino — типу long. Пам'ять Arduino є доволі обмеженою, що не дозволить генерувати дуже великі послідовності, але це і не є потрібним,

адже в експерименті розглядається саме генерація невеликих послідовностей в контексті легковагових генераторів випадкових чисел.

Тестування згенерованих послідовностей бітів на випадковість буде виконуватися за допомогою методів багатовимірних статистик [19–22]. Тести багатовимірних статистик відрізняються тільки шаблонами, на які перевіряється послідовність. Кожен метод отримує на вхід випадкову бітову послідовність. Для даної послідовності визначається кількість специфічних шаблонів (якщо це визначено методом) і виконується обчислення за допомогою формули специфічної для методу. Випробування NIST не дадуть якісної оцінки результатів, адже більшість з них розраховані на послідовності, довжина яких більша за 100 [23; 24].

Висновки

У статті розглянуто актуальне завдання — побудова легковагового генератора випадкових чисел та аналіз отриманих послідовностей на випадковість. Дослідження випадкових послідовностей та генераторів випадкових чисел є доволі специфічною задачею, але для її вирішення може бути використаний один або декілька з численних пакетів тестів. Однак, виконаний аналіз указує на те, що існуючі тести мають низку недоліків, вирішення яких може зменшити передумови до тестування та покращити точність отриманих результатів.

Фізична модель IoT генератора представлена в роботі, на своєму прикладі надає широкий огляд факторів та обмежень, що виникають під час проєктування генераторів.

Процес тестування та оптимізації генератора рекомендовано проводити з використанням тестів багатовимірних статистик, що ілюструє придатність пакету програм до використання і його інтегральну роль в створенні якісного генератора випадкових чисел, в особливості для використання в IoT пристроях.

Пакет тестів і пакет програм у цілому, рекомендовано до використання при дослідженні генераторів випадкових чисел та послідовностей створених ними на випадковість. Вони можуть бути застосовані в одні з таких областей:

- наукові дослідження — це встановлення залежності між будь-якими експериментальними даними, розробка ГПЧ та АГВЧ (як для IoT пристроїв, так і для загального використання), створення нових методів перевірки послідовності на випадковість;
- криптографія — це перевірка послідовностей згенерованих ГПЧ та АГВЧ, дослідження алгоритмів шифрування;
- розробка та супровід програмних продуктів — тестування ефективності алгоритмів та систем заснованих на випадковості, перевірка криптографічних засобів систем.

ЛІТЕРАТУРА

[1] Райчев О. О. Засіб тестування IoT генераторів випадкових чисел з використанням багатовимірних статистик: бакалаврська дипломна робота. Київський національний університет імені Тараса Шевченка, Київ, 2021.

[2] Orue A. Hernandez E., Luis Martín A., Vitini, F. A Lightweight Pseudorandom Number Generator for Securing the Internet of Things. *IEEE Access*, 2017. Pp. 1–1. 10.1109/ACCESS.2017.2774105. (eng)

[3] Airehrour D., Gutierrez J., Ray S. K., Secure routing for internet of things: A survey. *Journal of Network and Computer Applications*, 66:198–213, 2016. (eng)

[4] Mouha N., "The Design Space of Lightweight Cryptography". *NIST Lightweight Cryptography Workshop*, 2015. https://hal.inria.fr/hal01241013 (eng)

[5] Fischer V. "A Closer Look at Security in Random Number Generators Design". W. Schindler and S. Huss (Ed.). Third International Workshop on Constructive Side-Channel Analysis and Secure Design (COSADE 2012). LNCS 7275, pp. 167–182, 2012. (eng)

[6] Peinado A., Munilla J., Fuster-Sabater A. "EPCGen2 Pseudorandom Number Generators: Analysis of J3Gen". Sensors, 14(4): 6500–6515, 2014. (eng)

[7] Orue A., Hernandez L., Montoya F., "Trifork, a new Pseudorandom Number Generator Based on Lagged Fibonacci Maps". Journal of Computer Science and Engineering, 2(2):46–51, 2010. (eng)

[8] Francillon A., Castelluccia C., "Tinyrng: A cryptographic random number generator for wireless sensors network nodes". IEEE 2007 5th International Symposium on Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks, pp. 1–7, 2007. (eng)

[9] Lo Re G., Milazzo E., Ortolani M., Secure random number generation in wireless sensor networks. ACM Proceedings of the 4th International Conference on Security of Information and Networks (SIN'11), pp. 175–182, 2011. (eng)

[10] Gaglio V., Paola A., Ortolani M., Lo Re G., "A TRNG exploiting multi-source physical data." ACM Proceedings of the 6th Workshop on QoS and Security for Wireless and Mobile Networks (Q2SWinet'10), pp. 82–89, 2010. (eng)

[11] Mandal K., X. Fan, and G. Gong, Design and implementation of Warbler family of lightweight pseudorandom number generators for smart de-

vices. ACM Transactions on Embedded Computing Systems, 15:1–28, 2016. (eng)
[12] Mabin J., G. Sekar, and R. Balasubramanian, Distinguishing Attacks on (Ultra-)Lightweight WG Ciphers. 5th International Workshop Lightweight Cryptography for Security and Privacy (LightSec 2016), LNCS 10058, pp. 45–59, 2017. (eng)
[13] Peris-Lopez P., Hernandez-Castro J. C., Estevez-Tapiador J. M., Ribagorda A. "LAMED – a PRNG for EPC Class-1 Generation-2 RFID specification". Computer Standards and Interfaces, 31(1): 88–97, 2009. (eng)
[14] Markku-Juhani O., Saarinen, D.E. "A Do-It-All-Cipher for RFID: Design Requirements (Extended Abstract)". Cryptology ePrint Archive, Report 2012/317 (2012) (eng)
[15] Martın H., Peris-Lopez P., Tapiador J.E., San Millan E. "An estimator for the ASIC footprint area of lightweight cryptographic algorithms" IEEE Transactions on Industrial Informatics 10(2), 1216–1225 (2014) (eng)
[16] Melia-Segu J., Garcia-Alfaro J., Herrera-Joancomartı J. "J3Gen: A PRNG for low-cost passive RFID" Sensors, 13: 3816–3830, 2013. (eng)
[17] Melia-Segu J., J. Garcia-Alfaro, J. Herrera-Joancomartı, "Multiplepolynomial LFSR based pseudorandom number generator for EPC Gen2 RFID tags". 37th Annual Conference of the IEEE Industrial Electronics Society (IECON 2011), pp. 3820–3825, 2011. (eng)
[18] Ullah I., Meratnia N.and P. J. M. Havinga, "Entropy as a Service: A Lightweight Random Number Generator for Decentralized IoT Applications," 2020 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops), 2020, pp. 1–6, DOI: 10.1109/PerComWorkshops48775.2020.9156205 (eng)
[19] Popereshnyak S., Dimitrov G. "The Testing of Pseudorandom Sequences using Multidimensional Statistics" Proceedings of the 1st International Workshop on Digital Content & Smart Multimedia (DCSMart 2019) Lviv, Ukraine, December 23–25, 2019. p. 151–161 (eng)
[20] Masol V., Popereshnyak S. "Statistical analysis of local sections of bits sequence"s // Journal of Automation and Information Sciences. Vol. 51. 2019. p. 31–45. DOI: 10.1615/JAutomatInf Scien.v51.i10.30 (eng)
[21] Masol V., Popereshnyak S. "Checking the Randomness of Bits Disposition in Local Segments of the (0, 1)-Sequence". *Cybernetics and Systems Analysis*. 56(3). 2020. P. 1–8 DOI: 10.1007/s10559-020-00267-0 (eng)
[22] Popereshnyak S. "The technique for testing short sequences as a component of cryptography on the Internet of Things", CEUR-WS.org/vol/ 2516/paper 11 (eng)
[23] NIST Special Publication 800-57, Elaine Barker, William Barker, William Burr, William Polk, and Miles Smid «Recommendation for Key Management – Part 1: General (Revision 3)», July 2012 (eng)
[24] NIST Special Publication 800-90A, Elaine Barker, John Kelsey, «Recommendation for Random Number Generation Using Deterministic Random Bit Generators», January 2012. (eng)

Поперешняк С. В., Райчев О. О.

МОДЕЛЬ ЛЕГКОВАГОВОГО ГЕНЕРАТОРА ПСЕВДОВИПАДКОВИХ ЧИСЕЛ ДЛЯ ІНТЕРНЕТУ РЕЧЕЙ

Робота присвячена доволі актуальній задачі — дослідженню послідовностей та генераторів випадкових чисел, які працюють на пристроях з обмеженими ресурсами, на випадковість. Аналіз випадкових послідовностей та генераторів випадкових чисел є доволі специфічною задачею, але для її вирішення може бути використаний один або декілька з численних пакетів тестів. Однак, виконаний аналіз вказує на те, що існуючі тести мають низку недоліків, вирішення яких може зменшити передумови до тестування та покращити точність отриманих результатів. В роботі наведено типи генераторів випадкових чисел та їх недоліки. Описано легкий і швидкий генератор псевдовипадкових чисел, корисний для обмежених ресурсів пристроїв, таких як ті, що використовуються в IoT. Стаття присвячена розробці легких генераторів псевдовипадкових чисел. Описано принципи проектування легковагового генератора псевдовипадкових чисел. Сформульовано вимоги до якості легковагового генератора псевдовипадкових чисел. Було розглянуто побудову фізичної моделі легковагового генератора псевдовипадкових чисел. Фізична модель IoT генератора представлена в роботі, на своєму прикладі надає широкий огляд факторів та обмежень, що виникають під час проектування генераторів. Процес тестування та оптимізації легковагового генератора псевдовипадкових чисел рекомендовано проводити з використанням тестів багатовимірних статистик. Дані тести ілюструють придатність пакету програм до використання і його інтегральну роль в створенні якісного генератора випадкових чисел, в особливості для використання в IoT пристроях. Програмний продукт, що було створено в цій роботі може використовуватися для вирішення широкого спектру задач, як уже і було неодноразово зазначено. Одною з найважливіших, та дійсно тою, що може отримати неоціненну користь сферою застосування є криптографія.

Ключові слова: легковаговий генератор псевдовипадкових чисел; тестування; багатовимірна статистика; інтернет речей; криптографія.

Popreshnyak S., Raichev O.

LIGHTWEIGHT PSEUDORANDOM NUMBER GENERATOR MODEL FOR THE INTERNET OF THINGS

The work is devoted to a rather urgent problem — the study of sequences and generators of random numbers, working on devices with limited resources, for randomness. The analysis of random sequences and random number generators is a rather specific task, but one or more of numerous test packages can be used to solve it. However, the analysis performed indicates that the existing tests have a number of drawbacks, the solution of which can reduce the prerequisites for testing and improve the accuracy of the results. The paper presents the types of random number generators and their disadvantages. Described is a lightweight and fast pseudo-random number generator useful for resource-constrained devices such as those used in the IoT. The article is devoted to the development of lightweight pseudo-random number generators. The principles of designing a lightweight pseudo-random number generator are described. Requirements for the quality of a lightweight pseudo-random number generator are formulated. The construction of a physical model of a lightweight pseudo-random number generator was considered. The physical model of the IoT generator, which is presented in the work, by its example provides a broad overview of the factors and limitations that arise in the design of generators. The process of testing and optimizing a lightweight pseudo-random number generator is recommended using tests of multivariate statistics. These tests illustrate the suitability of the software package for use and its integral role in creating a quality random number generator, especially for use in IoT devices. The software product that was created in this work can be used to solve a wide range of tasks, as has already been repeatedly noted. One of the most important, and indeed the one that can receive invaluable benefits in the field of application, is cryptography.

Keywords: lightweight pseudorandom number generators; testing; multidimensional statistics; internet of things; cryptography.

Стаття надійшла до редакції 01.05.2021 р.
Прийнято до друку 09.06.2021 р.