ISSN 2225-5036 (Print), ISSN 2411-071X (Online)
http://infosecurity.nau.edu.ua; http://jrnl.nau.edu.ua/index.php/Infosecurity

DOI: 10.18372/2225-5036.27.16003

ДОСЛІДЖЕННЯ ТА ТЕСТУВАННЯ ЛЕГКОВАГОВИХ ГЕНЕРАТОРІВ ПСЕВДОВИПАДКОВИХ ЧИСЕЛ ДЛЯ ІНТЕРНЕТУ РЕЧЕЙ

Світлана Поперешняк, Олексій Райчев

Київський національний університет імені Тараса Шевченка, Україна

Поперешняк Світлана Володимирівна, к.ф.-м.н., доцент.
Рік та місце народження: 1980 рік, м. Кіровоград, Україна.
Освіта: Кіровоградський державний педагогічний університет імені Володимира Винниченка, 2002 рік.
Посада: доцент кафедри програмних систем і технологій факультету інформаційних технологій Київського національного університету імені Тараса Шевченка.
Наукові інтереси: програмна інженерія, автоматизація процесів виробництва, інформаційні технології, захист інформації, використання багатовимірних статистик для тестування бітової послідовності на випадковість.
Публікації: більше 100 наукових публікацій, серед яких навчальні посібники, наукові статті, матеріали та тези доповідей на конференціях.
E-mail: spopereshnyak@gmail.com.
Orcid ID: 0000-0002-0531-9809.

Райчев Олексій Олегович, магістр
Рік та місце народження: 2000 рік, м. Київ, Україна.
Освіта: Київський національний університет імені Тараса Шевченка, 2021 рік.
Посада: магістр кафедри програмних систем і технологій факультету інформаційних технологій Київського національного університету імені Тараса Шевченка.
Наукові інтереси: програмна інженерія, статистичні методи, інформаційні технології, захист інформації.
Публікації: 6 наукових публікацій.
E-mail: mileenocktopus@gmail.com.
Orcid ID: 0000-0002-4085-5711.

Анотація. *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. Робота присвячена доволі актуальній задачі – дослідженню генераторів випадкових чисел, які працюють на пристроях з обмеженими ресурсами, та послідовностей невеликої довжини на випадковість. В роботі було розглянуто побудову фізичної моделі легковагового генератора псевдовипадкових чисел. Використання багатовимірних статистик як основи для випробувань, дозволяє краще дослідити послідовність на випадковість, за рахунок оцінки одночасно декількох характеристик послідовності. Тести багатовимірних статистик засновані на дослідженні входжень шаблонів в послідовність і допомагають виявляти приховані залежності між даними та неякісні генератори. Головною перевагою цих тестів є їх ефективність на послідовностях короткої довжини, тому вони вирішують одну з проблем існуючих тестів, полегшуючи передумови до випробувань. 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 генератора з використанням тестів багатовимірних статистик ілюструє 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.*

Вступ

Генератор псевдовипадкових чисел (ГПВЧ) - це механізм генерації випадкових чисел у комп'ютері. Його називають псевдовипадковим, оскільки отримати справжнє випадкове число за допомогою комп'ютера дуже важко і дорого.

Теоретично справжнє випадкове число можна отримати з таких джерел ентропії [1]:

- навколишній шум;
- радіоактивний розпад;
- шуми струмів в електричних ланцюгах;
- вимірювання реакції користувача (рух миші) тощо.

Інтернет речей (IoT) широко застосовується багатьма галузями для величезної кількості програм [2]. За останнє десятиліття кількість пристроїв IoT зросла в геометричній прогресії [3], і очікується, що вона зросте ще більше.

Наприклад, в інтелектуальній логістиці, де піддони вбудовані в розумні датчики для аналітичних цілей (виявляють, передбачають і запобігають різним подіям, пов'язаним з логістикою).

У розумній логістиці існує складний ланцюжок зацікавлених сторін, для яких безпека та конфіденційність є ключовими питаннями.

Отже, дані, генеровані цими розумними датчиками, які є обмеженими, та повинні бути захищеними. Централізоване управління цими датчиками є надзвичайно складним, тому потреба у розподіленому механізмі безпеки зростає [4].

У розподіленому механізмі кожен пристрій здатний генерувати власні випадкові числа без необхідності конфігурації центральної сторони або вручну.

Аналіз існуючих досліджень

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 роботі [5] автори розробили та впровадили істинний (справжній) генератор псевдовипадкових чисел (СГВЧ), криптографічний генератор псевдовипадкових чисел, який використовує отримані бітові помилки як джерело випадковості у вузлах бездротових датчиків.

У роботі [6] автори представили вдосконалену версію СГВЧ, запропоновану в роботі [7], 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.

ГПВЧ для недорогих інтелектуальних пристроїв, таких як вузли датчиків, було представлено в [8]. Він базується на поєднанні модифікованих блоків Бруйна та регістра зсуву нелінійних зворотних зв'язків. Два запропоновані екземпляри підходять для захисту недорогих смарт-пристроїв. У [9] важлива відмінна атака на все сімейство ГПВЧ потокових шифрів показує, що майже кожен член цієї родини вразливий до лінійних атак; це може загрожувати безпеці. ГПВЧ з назвою LAMED був представлений в [10] 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 [11], [12]. У [13] було запропоновано J3Gen ГПВЧ на основі попередньої роботи в [14]. J3Gen поєднує в собі TRNG 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 [15, 16], 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 – в роботі запропоновано фізичну модель IoT генератора, яка на своєму прикладі надає широкий огляд факторів та обмежень, що виникають під час проектування генераторів. Процес тестування та оптимізації генератора з використанням тестів багатовимірних статистик ілюструє придатність пакету програм до використання і його інтегральну роль в створенні якісного генератора випадкових чисел, в особливості для використання в IoT пристроях.

В роботі розглянуто критерії для перевірки на випадковість бітових послідовностей невеликої довжини (до 100 біт). Даний підхід доцільно використовувати

для тестування полегшеного генератора псевдовипадкових чисел в пристроях з певними обмеженнями на ресурси.

Основна частина дослідження

Властивості генератора псевдовипадкових чисел. Перерахуємо деякі властивості, які повинен мати генератор [17]. Звичайно, потрібно, щоб він виробляв послідовність з рівномірним розподілом на (0, 1).

Це саме по собі є досить абстрактною математичною вимогою; перші дві властивості, наведені нижче, роблять це більш практичним.

1. Пройдіть емпіричні статистичні тести Це тести, де генерується довга послідовність випадкових чисел, а потім потрібно пройти різні статистичні тести, щоб перевірити гіпотезу про те, що числа рівномірно розподілені на [0, 1] і є незалежними.

2. Математична основа За генераторами випадкових чисел (принаймні деякими з них) стоїть багато математичної теорії, включаючи властивості, які повинні створити хороший генератор випадкових чисел.

3. Швидкість (при обмеженій пам'яті) Більшість моделювань вимагають величезної кількості випадкових чисел. Можливо, доведеться сформувати велику кількість вибірок, і генерація кожної вибірки часто передбачає багаторазовий виклик генератора випадкових чисел (ГВЧ). Тож ГВЧ повинен бути швидким.

4. Кілька потоків Легко використовувати паралельні обчислення, але для цього потрібно запустити кілька копій генератора випадкових чисел. Тож потрібно переконатися, що всі ці різні потоки не залежать один від одного.

5. Практичні проблеми Простота установки та легкість генерування. Деякі генератори випадкових чисел досить короткі і займають лише кілька рядків коду. Інші значно складніші. Потрібно переконатися, що отримуються послідовності належним чином.

6. Відтворюваність Для налагодження та тестування потрібно мати можливість генерувати однаковий потік випадкових чисел неодноразово.

Парадокс випадковості

Багатьом захищеним системам необхідно часто генерувати випадкові числа, що суперечить бажанням зменшити кількість подій, які можуть виявити вимірні дані побічного каналу. Щоб створити безпечні секретні ключі, система також повинна використовувати різні початкові числа випадкових чисел кожен раз, коли вона генерує ключ. Таким чином, випливає питання як можна узгодити вимоги до частої генерації випадкових чисел з необхідністю мінімізувати активність для захисту системи від аналізу даних побічного каналу. Одна з проблем, пов'язаних з виконанням цього в цифрових комп'ютерах, полягає в тому, що вони спроектовані так, щоб бути детермінованими, працювати з двійковими даними, щоб усунути невизначеності при маніпулюванні аналоговими сигналами. Створення справжньої випадковості в таких схемах є надзвичайно складною задачею.

Багато різних підходів, які можна для цього використати мають низку недоліків. Які в свою чергу обмежують їх застосовність. Багато ранніх ГВЧ покладалися на час дня, математично комбінуючи цифри, що представляють секунди, десяті, соті і навіть тисячні секунди, для отримання явно випадкового початкового числа.

Однак це початкове число насправді можна розглядати тільки як псевдовипадкове число, тому що результат генерується за допомогою детермінованого процесу, який, якщо його вводити в один і той же час дня, поверне ту ж послідовність «випадкових» чисел. Підхід не проходить перевірку на незалежність, тому що результат передбачуваний. Зловмисник, який може вгадати або, що ще гірше, встановити годинник часу, має можливість значно обмежити кількість можливих початкових значень, використовуваних ГПСЧ, і, отже, може використовувати цю слабкість, щоб порушити безпеку системи. Багато невеликих вбудованих систем, що представляють пристрої IoT, не мають навіть постійних годинників, зберігають своє значення під час перезавантаження або циклів включення живлення. Часто зустрічаються такі системи, які використовують такі функції, як лічильники циклів, для заповнення своїх генераторів випадкових чисел, що призводить до того, що багато копії одного і того ж пристрою генерують одну і ту ж послідовність випадкових чисел з моменту їх скидання. Ці системи просто чекають, щоб їх використовували.

Дизайн генератора випадкових чисел

Одне з основних питань, які потрібно вирішити перед проектуванням та розробкою генератора випадкових чисел, а саме апаратного генератора випадкових чисел (АГВЧ) є вибір датчиків та сенсорів, що будуть використовуватися для генерації випадкових чисел використовуючи фактори навколишнього середовища.

В залежності від обраних факторів навколишнього середовища та варіантів використання генератора необхідно визначитися з датчиками, які будуть збирати інформацію з навколишнього середовища, адже датчик світла не буде корисним, якщо пристрій використовується в темному приміщенні, а датчик руху або акселерометр буде надлишковим для стаціонарного пристрою. Для створюваної моделі генератора це питання треба врахувати, але воно не є настільки критичним як при розробці реального девайсу.

Важливими характеристиками при розробці пристроїв для Інтернету речей є такі як: вага, розміри, об’єм пам’яті, інтеграція з програмним забезпеченням та іншими пристроями тощо.

Мета даної роботи показати підхід до створення легковагового генератора, а також процес тестування, налагодження та оптимізації створеного генератора з використанням програмного засобу для тестування послідовності малої довжини на випадковість з використанням багатовимірних статистик [18], тому, не зважаючи на серйозність цих факторів при розробці пристроїв для кінцевого користувача, ними можна частково знехтувати.

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.

Реальну модель генератора можна побачити на рис. 1.

За основу генератора та мікросхеми було вирішено використати плату Arduino та для моделі було обрано датчик звуку та акселерометр [17].

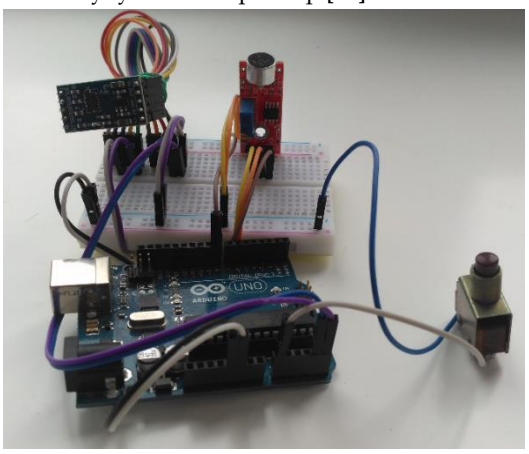

Рис. 1. Фізична модель елементів генератора

На реальній моделі можна побачити сенсори під'єднані до макетної дошки, саму плату Arudino та кнопку. На схемі наявні такі компоненти:

- Датчик звуку;
- Акселерометр;
- Кнопка – тригер;
- Arduino – головна схема генератора.

Варто додати, що генератор створює послідовності довжини 31 біт (короткі послідовності). 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.

Методи тестування з використанням багатовимірних статистик

В даному розділі приведемо набори тестів, які будемо використовувати для дослідження згенерованих послідовностей та оптимізації моделі легковагового генератора. Тести багатовимірних статистик відрізняються тільки шаблонами, на які перевіряється послідовність [18-21]. Кожен метод отримує на вхід випадкову величину:

$$\gamma_1, \gamma_2 \ldots, \gamma_n, \quad \text{де } \gamma_i \in \{0, 1\},\ i = 1,2\ldots,n,\ n > 0. \tag{1}$$

Для даної величини визначається кількість специфічних шаблонів k_1, k_2 та k_3 (якщо це визначено методом) і виконується обчислення за допомогою формули специфічної для методу. Перший тест виконується, щоб знайти спільну вірогідність появи подій $k_1 = \eta(tt^*)$ та $k_2 = \eta(t1t^*) + \eta(t0t^*)$, при $t \in \{0, 1\}$, $t^*=1 - t$:

$$\mathrm{P}\left\{\eta\left(tt^*\right) = k_1, \eta\left(t1t^*\right) + \eta\left(t0t^*\right) = k_2\right\} = \sum_{m_1=k_1}^{n-k_1} p^{m_1} q^{m_0} \sum \prod_{i=0}^{1} C_{k_1}^{\delta_i} C_{m_i-k_1}^{k_1-\delta_i}, \tag{2}$$

де n – довжина бітової послідовності, p – вірогідність появи t, q – вірогідність появи t^* ($q = 1 - p$), $m_0 = n - m_1$, $\sum$ - сума по всім комбінаціям δ_0 та δ_1, таким, що: $\delta_0 + \delta_1 = 2k_1 + k_2$.

Другий метод тестування знаходить спільну вірогідність появи подій $k_1 = \eta(tt^*)$ та $k_2 = \eta(ttt^*)$:

$$\mathrm{P}\left\{\eta\left(tt^*\right) = k_1, \eta\left(ttt^*\right) = k_2\right\} = \sum_{m_1=k_1}^{n-k_1} p^{m_1} q^{m_0} C_{k_1}^{k_2} C_{m_1-k_1}^{k_2} C_{m_0}^{k_1}. \tag{3}$$

Третій метод оцінює вірогідність появи шаблонів $k_1 = \eta(tt^*)$, $k_2 = \eta(t1t^*)$ *та* $k_3 = \eta(t0t^*)$:

$$\mathrm{P}\left\{\eta\left(tt^*\right) = k_1, \eta\left(t1t^*\right) = k_2, \eta\left(t0t^*\right) = k_3\right\} = \sum_{m_1=k_1}^{n-k_1} p^{m_1} q^{m_0} C_{k_1}^{k_2} C_{k_1}^{k_3} C_{m_1-k_1}^{k_2} C_{m_0-k_1}^{k_3}. \tag{4}$$

За допомогою четвертого методу можна визначити вірогідність подій $k_1 = \eta(tt^*)$ та $k_2 = \eta(t1t)+ \eta(t0t)$.

$$\mathrm{P}\left\{\eta\left(tt^*\right) = k_1, \eta\left(t1t\right) + \eta\left(t0t\right) = k_2\right\} = \sum_{m_1=k_1}^{n-k_1} p^{m_1} q^{m_0} \times \sum \sum_{i \in \{k_1, k_1+1\}} C_{i-1}^{\delta_0} C_i^{\delta_1 - m_1 + 2i} C_{m_0-i+1}^{k_1-\delta_0} Z\left(m_1 - i, m_1 - i - \delta_1\right),$$

де $\sum$ - сума по всім комбінаціям δ_0 та δ_1, таким, що:

$$\delta_0 + \delta_1 = k_2,\ Z(a,b) = \begin{Bmatrix} C_{a-1}^{b-1}, \text{ якщо } a \ge b \ge 0; \\ 1, \text{ якщо } a = b = 0; \\ 0, \text{ в іншому випадку} \end{Bmatrix}.$$

За допомогою п'ятого методу можна визначити вірогідність подій $k_1 = \eta(tt^*)$ та $k_2 = \eta(ttt)$:

$$\mathrm{P}\left\{\eta\left(tt^*\right) = k_1, \eta\left(ttt\right) = k_2\right\} = \sum_{m_1=k_1}^{n-k_1} p^{m_1} q^{m_0} C_{m_0}^{k_1} \times \sum_{i \in \{k_1, k_1+1\}} C_i^{m_1-k_2-i} Z\left(m_1 - i, m_1 - i - k_2\right). \tag{5}$$

Шостий метод можна використати для того щоб знайти спільну вірогідність подій $k_1 = \eta(tt^*)$ та $k_2 = \eta(tt^*t)$:

$$\mathrm{P}\left\{\eta(tt^*)=k_1,\eta(tt^*t)=k_2\right\}=$$

$$\sum_{m_1=k_1}^{n-k_1} p^{m_1}q^{m_0}\sum_{i\in\{k_1,k_1+1\}} C_i^{k_2}C_{m_0-i}^{k_1-k_2}Z\left(m_1,i+1\right). \quad (8)$$

Сьомий метод шукає спільну вірогідність подій $k_1 = \eta(tt^*)$, $k_2 = \eta(ttt)$ та $k_3 = \eta(tt^*t)$:

$$\mathrm{P}\left\{\eta(tt^*)=k_1,\eta(ttt)=k_2,\eta(tt^*t)=k_3\right\}=\sum_{m_1=k_1}^{n-k_1} p^{m_1}q^{m_0}\times$$

$$\times\sum_{i\in\{k_1,k_1+1\}} C_i^{k_2-m_1+2i}C_{i-1}^{k_3}C_{m_0-i+1}^{k_1-k_3}Z\left(m_1-i,m_1-i-k_2\right). \quad (9)$$

Восьмий метод виконується, щоб знайти спільну вірогідність подій $k_1 = \eta(tt)$ та $k_2 = \eta(t^*t\,t^*)$:

$$\mathbf{P}\left\{\eta(tt)=k_1,\eta(t^*tt^*)=k_2\right\}=\sum_{m_1=0}^{n} p^{m_1}q^{m_0}\times$$

$$\left\{C_{a-2}^{k_2}C_{k_1+1}^{a-k_2+1}Z\left(m_0,a-1\right)+C_a^{k_2}C_{m_0-1}^{a}Z\left(k_1,a-k_2\right)+\right.$$

$$\left.+2C_{a-1}^{k_2}C_{k_1}^{a-k_2-1}C_{m_0-1}^{a-1}+\chi\left(a-1=k_2=m_0=0\right)\right\},$$

де $\chi(E)$ – індикатор події E, $a = m_1 - k_1$.

Тестування, оптимізація та інтерпретація результатів

Тестування згенерованих послідовностей бітів на випадковість буде виконуватися за допомогою методів багатовимірних статистик. Випробування NIST не дадуть якісної оцінки результатів, адже більшість з них розраховані на послідовності, довжина яких більша за 100.

Загальний процес тестування, що буде використано при оцінці генераторів побудовано наступним чином:

– запустити програму за допомогою Arduino та згенерувати деяку кількість випадкових послідовностей;

– виконати тестування послідовностей з використанням восьми методів багатовимірних статистик та реалізованих в пакеті програм;

– знайти відношення між результатами отриманими на попередньому етапі та максимальними значеннями для тесту і відповідної довжини послідовності. Це значення можна знайти за допомогою визначення найбільш вірогідної кількості входжень шаблонів в послідовність;

– зробити висновки про вірогідність послідовності.

Перед тим як розпочати тестування, розглянемо табл. 1, в якій наведено максимальна вірогідність для кожного з тестів багатовимірних статистик. Щоб оцінити наскільки та чи інша послідовність є випадковою, необхідно знайти відношення результатів відповідного тесту до максимальної вірогідності цього тесту. Якщо відношення є більшим за 0,8 то послідовність можна вважати випадковою, якщо ні – вона не є випадковою. Результати тестування 10 послідовностей бітів створених псевдовипадковим генератором на Arduino наведено в табл. 2.

Таблиця 1

Максимальні вірогідності для тестів багатовимірних статистик та довжини послідовності 31

№ Тесту	Найбільша вірогідність
1	0,0699318274855613
2	0,0847552437335252
3	0,0277345534414052
4	0,0368665847927331
5	0,0499579892493784
6	0,0855854991823434
7	0,0172293558716774
8	0,0369482999667525

Таблиця 2

Результати тестування ГПЧ

№ Тесту	1	2	3	4	5	6	7	8
Частина послідовностей що пройшли тест	30%	30%	10%	20%	30%	0%	0%	20%

Результати тестування явно вказують на те, що такі послідовності мають низький рівень випадковості. Оцінити дані можна більш точно, якщо розглянути як показує себе АГВЧ створений з використанням сенсорів (табл. 3). Результати є істотно кращими за ГПЧ і мають загальний позитивний тренд в контексті випадковості. Результати тестування АГВЧ хоч і є позитивними, вказують на те, що цей генератор не створює послідовності належної якості і що його можна потенційно покращити. Однією з проблем зчитування даних з датчиків є те, що дані не змінюються різко і можуть мати загальний тренд.

Таблиця 3

Результати тестування АГВЧ

№ Тесту	1	2	3	4	5	6	7	8
Частина послідовностей що пройшли тест	30%	50%	30%	30%	30%	20%	10%	20%

Цю проблему можна вирішити за допомогою внесення додаткової випадковості в процес генерації послідовності, або винесення детермінованості за межі алгоритму. Одним з підходів, що може бути використано, є Fisher-Yates shuffle [9] – алгоритм, що можна використати для змішування результатів отриманих з датчиків. Результати явно вказують на те, що змішування даних отриманих з сенсорів мало позитивний вплив на випадковість послідовностей створюваних генератором. Наостанок, розглянемо послідовності згенеровані під час

того, як генератор та сенсори знаходилися в відносному спокої (табл. 5).

Таблиця 4

Результати тестування АГВЧ, що використовує алгоритм Fisher-Yates shuffle

№ Тесту	1	2	3	4	5	6	7	8
Частина послідовностей що пройшли тест	60%	50%	30%	50%	50%	40%	30%	40%

Таблиця 5

Результати тестування АГВЧ в стані відносного спокою

№ Тесту	1	2	3	4	5	6	7	8
Частина послідовностей що пройшли тест	20%	20%	10%	10%	10%	20%	10%	10%

Як можна побачити з результатів, коли сенсори не отримують постійних випадкових даних, в послідовностях що створює генератор з'являється висока детермінованість. Відповідно, даний генератор потребує поліпшення на випадок ситуації відносного спокою.

Практичні рекомендації

Перед тим як застосовувати ланцюжки випадкових чисел для моделювання реальної практичної задачі бажано дотримуватися наступних порад:

– Не використовувати вбудований ГПВ, якщо невідомо як він сконструйований, алгоритм його роботи і як він був протестований.

– Потрібно сконструювати свій код таким чином, щоб було легко змінити його в створеному легковаговому генераторі псевдовипадкових чисел, який ви використовуєте.

– Бажано використовувати два різні ГПВЧ і порівнювати їх результати. Це не втрата часу, оскільки завжди можна поєднати два набори даних. Бажано сконструювати другий генератор таким чином, щоб він використовував відмінні від першого генератора фактори навколишнього середовища та інші датчики для збору цих факторів. Або якщо не планується розробка ще одного генератора, то можна скористатися підходом при якому використовується лише кожне п'яте значення випадкового числа з генератора.

– Доцільно не використовувати занадто багато випадкових чисел з генератора порівняно з його періодом.

Висновки

Аналіз галузі розробки та побудови легковагових генераторів дозволив обґрунтувати необхідність побудови пакетів тестів для перевірки на випадковість послідовностей, які є результатом генерації пристроїв з певними обмеженнями.

Виконаний аналіз вказує на те, що існуючі тести мають низку недоліків, вирішення яких може зменшити передумови до тестування та покращити точність отриманих результатів. Робота присвячена доволі актуальній задачі – дослідженню генераторів випадкових чисел, які працюють на пристроях з обмеженими ресурсами, та послідовностей невеликої довжини на випадковість.

В роботі було розглянуто побудову фізичної моделі легковагового генератора псевдовипадкових чисел. Використання багатовимірних статистик як основи для випробувань, дозволяє краще дослідити послідовність на випадковість, за рахунок оцінки одночасно декількох характеристик послідовності. Тести багатовимірних статистик засновані на дослідженні входжень шаблонів в послідовність і допомагають виявляти приховані залежності між даними та неякісні генератори. Головною перевагою цих тестів є їх ефективність на послідовностях короткої довжини, тому вони вирішують одну з проблем існуючих тестів, полегшуючи передумови до випробувань.

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 генератора з використанням тестів багатовимірних статистик ілюструє 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.

Література

[1] Chugunkov I. V., Novikova O. Y., Perevozchikov V. A. and Troitskiy S. S., "The development and researching of lightweight pseudorandom number generators," *2016 IEEE NW Russia Young Researchers in Electrical and Electronic Engineering Conference (EIConRusNW)*, 2016. - pp. 185-189.

[2] Ullah I., Meratnia N. and Havinga P. J. M., "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.

[3] Dinca L. M., Hancke G., "Behavioural sensor data as randomness source for iot devices", *2017 IEEE 26th International Symposium on Industrial Electronics (ISIE), June 2017*, 2017. - pp. 2038-2043.

[4] Francis L., Hancke G., Mayes K. and Markantonakis K., "Potential misuse of nfc enabled mobile phones with embedded security elements as contactless attack platforms", *2009 International Conference for Internet Technology and Secured Transactions (ICITST)*, Nov 2009, 2009. - pp. 1-8.

[5] 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), 2010. – pp. 46-51.

[6] 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*, 2007. - pp.1–7.

[7] 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)*, 2011. - pp.175– 182.

[8] 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)*, 2010. - pp.82–89.

[9] Mandal K., X. Fan, and G. Gong, Design and implementation of Warbler family of lightweight pseudorandom number generators for smart devices. *ACM Transactions on Embedded Computing Systems*, 2016. – pp. 1-28.

[10] 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, 2017. - pp.45–59.

[11] 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), 2009. –pp. 88-97.

[12] Markku - Juhani O., Saarinen, D.E. "A Do-It-All-Cipher for RFID: Design Requirements (Extended Abstract)". *Cryptology ePrint Archive, Report* 2012/317, 2012. – 54 p.

[13] 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), 2014. – pp.1216–1225.

[14] Melia-Segu J., Garcia-Alfaro J., Herrera-Joancomartı J. "J3Gen: *A PRNG for low-cost passive RFID" Sensors*, 2013. - pp. 3816–3830.

[15] 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)*, 2011. - pp. 3820–3825.

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

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

[18] Поперешняк С.В. Програмний засіб для тестування бітової послідовності малої довжини на випадковість // *Безпека інформації*. – т. 27 (2), 2020. – С. 80-86.

[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. - pp. 151-161.

[20] Masol V., Popereshnyak S. Statistical analysis of local sections of bits sequences // *Journal of Automation and Information Sciences*. Vol. 51, 2019. - pp. 31-45.

[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. - pp. 1-8.

УДК 519.212.2 : 681.51

Поперешняк С.В., Райчев А.О. Исследования и тестирование легковесных генераторов псевдослучайных чисел для интернета вещей

Аннотация. *Анализ случайных последовательностей и генераторов случайных чисел является довольно специфической задачей, но для ее решения может быть использован один или несколько из многочисленных пакетов тестов. Однако, выполненный анализ указывает на то, что существующие методы тестирования имеют ряд недостатков, решение которых может уменьшить предпосылки к тестированию и улучшить точность полученных результатов. Работа посвящена довольно актуальной задачи - исследованию генераторов случайных чисел, работающих на устройствах с ограниченными ресурсами, и последовательностей небольшой длины на случайность. В работе было рассмотрено построение физической модели легковесного генератора псевдослучайных чисел. Использование многомерных статистик в качестве основы для испытаний, позволяет лучше исследовать последовательность на случайность, за счет оценки одновременно нескольких характеристик последовательности. Тесты многомерных статистик, которые основанные на исследовании*

вхождений нескольких шаблонов в последовательность, помогают выявлять скрытые зависимости между данными и некачественные генераторы. Главным преимуществом этих тестов является их эффективность на последовательностях короткой длины, поэтому они решают одну из проблем существующих тестов, облегчая предпосылки к испытаниям. Представленная в работе физическая модель IoT генератора на своем примере предоставляет широкий обзор факторов и ограничений, которые возникают при проектировании генераторов. Процесс тестирования и оптимизации генератора с использованием тестов многомерных статистик иллюстрирует пригодность пакета программ к использованию и его интегральную роль в создании качественного генератора случайных чисел, в особенности для использования в IoT устройствах. Программный продукт, что был создан в этой работе может использоваться для решения широкого спектра задач, как уже и было неоднократно отмечено. Одной из важнейших, и действительно той, что может получить неоценимую пользу сферой применения является криптография.

Ключевые слова: *легковесный генератор псевдослучайных чисел, тестирование, многомерная статистика, интернет вещей, криптография.*

Popereshnyak S. V., Raichev O.O. The research and testing of lightweight pseudorandom number generators for the Internet of Things

Abstract. *The analysis of random sequences and random number generators is a rather specific task, but one or more of the numerous test packages can be used to solve it. However, the analysis performed indicates that the existing testing methods have a number of disadvantages, the solution of which can reduce the prerequisites for testing and improve the accuracy of the results. The work is devoted to a rather urgent problem - the study of random number generators operating on devices with limited resources, and short-length sequences for randomness. The paper considered the construction of a physical model of a lightweight pseudo-random number generator. By using multivariate statistics as a basis for testing, it is possible to better investigate a sequence for randomness by evaluating several characteristics of the sequence simultaneously. Tests of multivariate statistics, which are based on the study of occurrences of several patterns in a sequence, help to reveal hidden dependencies between data and low-quality generators. The main advantage of these tests is their effectiveness on short sequences, so they solve one of the problems of existing tests, facilitating the prerequisites for testing. The physical model of the IoT generator 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 the generator using tests of multivariate statistics illustrates 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 generator, testing, multivariate statistics, internet of things, cryptography.*

Поперешняк Світлана Володимирівна, к.ф.-м.н., доцент, доцент кафедри програмних систем і технологій Київського національного університету імені Тараса Шевченка.
Поперешняк Светлана Владимировна, к.ф.-м.н., доцент, доцент кафедры программных систем и технологий Киевского национального университета имени Тараса Шевченко.
Popereshnyak Svitlana, Ph.D., assistant Professor of the Department of Software Systems and Technologies, Taras Shevchenko National University of Kyiv.

Райчев Олексій Олегович, магістр кафедри програмних систем і технологій Київського національного університету імені Тараса Шевченка.
Райчев Алексей Олегович, магистр кафедры программных систем и технологий Киевского национального университета имени Тараса Шевченко.
Raichev Oleksiy, Master of the Department of Software Systems and Technologies, Taras Shevchenko National University of Kyiv.

Отримано 20 *червня 2021 року,* затверджено редколегією *27 серпня 2021 року*