КІБЕРБЕЗПЕКА: освіта, наука, техніка
CYBERSECURITY: EDUCATION, SCIENCE, TECHNIQUE
№ 4 (24), 2024
ISSN 2663 - 4023
DOI 10.28925/2663-4023.2024.24.115132
УДК 004. 056.2:004.421.5

**Нємкова Олена Анатоліївна**
д.т.н., професор, професор кафедри безпеки інформаційних технологій
Національний університет «Львівська політехніка», Львів, Україна
ORCID ID: 0000-0003-0690-2657
*olena.a.niemkova@lpnu.ua*

**Кіх Михайло Васильович**
аспірант кафедри безпеки інформаційних технологій
Національний університет «Львівська політехніка», Львів, Україна
ORCID ID: 0009-0007-1847-1862
*mykhailo.v.kikh@lpnu.ua*

# ПОРІВНЯЛЬНЕ ДОСЛІДЖЕННЯ ТЕСТІВ ДЛЯ ОЦІНКИ СТАТИСТИЧНИХ ХАРАКТЕРИСТИК ГЕНЕРАТОРІВ ВИПАДКОВИХ ТА ПСЕВДОВИПАДКОВИХ ПОСЛІДОВНОСТЕЙ

**Анотація.** У світі інформаційної безпеки, комп'ютерної науки і криптографії питання статистичної безпеки згенерованих послідовностей є вельми важливим. Статистична безпека послідовностей відіграє важливу роль у наступних галузях: криптографія, комп'ютерна безпека, моделювання систем, статистичний аналіз, інформаційна безпека в мережах. Ця стаття присвячена дослідженню та порівнянню наборів тестів для оцінки статистичних властивостей генераторів випадкових та псевдовипадкових послідовностей. Порівняння зосереджено на таких відомих наборах тестів, як NIST, DieHard та TestU01. Ці тести були вибрані для дослідження через їх широке використання та визнану ефективність у вимірюванні якості генераторів. Стаття розглядає різні аспекти цих наборів тестів, включаючи призначення, складність, обсяг, точність оцінки, популярність, проблеми та обмеження, а також новаторство та розвиток. Тести NIST широко використовуються в криптографії та дослідженнях, вони мають ряд підходів для оцінки різних аспектів випадкових послідовностей. Тести DieHard зосереджуються на складних статистичних властивостях та зазвичай застосовуються для більш глибокого аналізу генераторів. З іншого боку, тести TestU01 володіють більшою чутливістю та розгалуженістю, дозволяючи виявити більш широкий спектр недоліків у генераторах випадкових чисел. Порівняльне дослідження тестів NIST, DieHard та TestU01 виявило, що кожен з них має свої переваги та недоліки в оцінці статистичних характеристик генераторів. Детальний огляд різних наборів тестів дозволяє краще зрозуміти їхні переваги та обмеження, що може сприяти вибору належного набору тестів для конкретного завдання. Комплексне використання цих тестів може забезпечити більш точну та повну оцінку якості генераторів. Отримані результати стануть корисною вихідною точкою для подальших досліджень у цьому напрямку та в розробці надійних генераторів. Висновки статті можуть бути корисними для дослідників, розробників програмного забезпечення та інших спеціалістів, які працюють з генераторами випадкових та псевдовипадкових послідовностей.

**Ключові слова:** інформаційна безпека; генератори випадкових послідовностей; генератори псевдовипадкових послідовностей; статистичні тести; NIST; DieHard; TestU01.

## ВСТУП

Тести для перевірки статистичної безпеки генераторів випадкових послідовностей (ГВП) і генераторів псевдовипадкових послідовностей (ГПВП) відіграють важливу роль у різних областях [1] – [17], включаючи:

1. Криптографія. У криптографічних застосуваннях випадкові послідовності (ВП) або псевдовипадкові послідовності (ПВП) використовуються для

генерації ключів [1], [5], [8], [10], і недостатня якість генераторів може стати причиною вразливостей у криптографічних протоколах та системах.

2. Комп'ютерна безпека. У сфері комп'ютерної безпеки ГВП або ГПВП використовуються для створення токенів доступу [3], [6], паролів [6] та ідентифікаторів сесій [3], [10], і ненадійні генератори можуть призвести до недостатнього рівня захисту.
3. Моделювання систем. В області моделювання та симуляцій ГВП або ГПВП використовуються для створення випадкових подій та даних [4], [11], [12], і їхній недолік може призвести до неточних результатів моделювання.
4. Статистичний аналіз. Для статистичного аналізу даних важливо мати надійний ГВП або ГПВП для створення тестових вибірок та випробування гіпотез [8], [13], [14].
5. Інформаційна безпека в мережах. У мережевих протоколах та системах безпеки ВП або ПВП використовуються для генерації токенів ідентифікації [15], підписів [15] – [17] та ключів шифрування [17].

У всіх цих областях надійність, статистична випадковість та відповідність розподілу генерованих чисел є критичними для забезпечення безпеки та надійності систем. Тести статистичної випадковості допомагають оцінити ці характеристики генераторів і виявити можливі проблеми.

**Постановка проблеми.** При виборі генератора випадкових або псевдовипадкових послідовностей для застосування в криптографії, комп'ютерній безпеці, моделюванні систем, статистичному аналізі, інформаційній безпеці в мережах та інших областях, важливо мати надійні засоби для оцінки його якості та статистичних характеристик. У зв'язку з цим існує потреба в систематичному порівняльному аналізі різних методів тестування генераторів випадкових чисел, зокрема, тестів NIST, DieHard та TestU01. Потрібно розглянути різні аспекти цих наборів тестів, включаючи призначення, складність, обсяг, точність оцінки, застосування в практиці, популярність, проблеми та обмеження, а також новаторство та розвиток. Автори аналізують, які набори тестів найбільше підходять для різних сценаріїв використання. Дослідження цих аспектів може допомогти забезпечити краще розуміння та вибір оптимального набору для тестування генераторів.

**Аналіз останніх досліджень і публікацій.** В роботах [1], [5], [11], [18] висвітлені методи генерації і тестування випадкових послідовностей.

Авторами [1], [2], [6], [10], [15], [19] досліджено загальний механізм перевірки статистичних гіпотез, а також коротко описуються загальні принципи роботи статистичних тестів.

Розглянуто питання [3], [5], [11], [12], [14], [20] етапів проходження статистичних тестів. Зокрема, додається список наявних тестів для дослідження статистичних характеристик із детальним описом.

В роботах [1] – [4], [9], [13], [17], [21] – [23] розглядаються практичні основи використання тестів NIST (досліджуються ГПВП і аналізуються результати статистичних характеристик). Також, додається короткий опис набору тестів NIST і принципів проведення тестування.

Авторами [5], [6], [10], [11], [14], [16], [19], [24] – [26] висвітленні практичні основи використання тестів DieHard (досліджуються генератори випадкових числових послідовностей (ГВЧП) і аналізуються результати статистичних характеристик). Також, додається короткий опис пакету тестів DieHard і принципів проведення тестування.

Розглянуто питання [7], [8], [12], [15], [18], [20], [27] – [29] практичного використання тестів TestU01 (досліджуються ГВП, ГПВП та алгоритмів генерації і

аналізуються результати статистичних характеристик). Також, додається короткий опис набору тестів TestU01 і принципів проведення тестування.

**Мета статті.** Провести порівняльне дослідження трьох основних наборів для тестування генераторів випадкових та псевдовипадкових послідовностей: NIST, DieHard та TestU01.

## ТЕОРЕТИЧНІ ОСНОВИ ДОСЛІДЖЕННЯ

Статистичні тести використовується для збору доказів того, що ГВП або ГПВП дійсно створюють числа, які здаються випадковими [1]. Паралельно ми виявляємо вразливості генераторів порівнюючи отримані статистичні дані із уже відомими характеристиками істинно випадкових послідовностей [1], [5].

Вирішення таких завдань ґрунтується на перевірці деяких гіпотез щодо властивостей ВП і ПВП, що виробляються генераторами. Як статистична гіпотеза може використовуватися довільне припущення про характер розподілу та властивості випадкової величини. Істинність або хибність такого припущення підтверджується або відхиляється з допомогою методів математичної статистики [11], [18].

Загальний механізм перевірки статистичних гіпотез полягає у наступному. Висуваються дві гіпотези: нульова ($H_0$) та альтернативна ($H_1$) [18]. Припустимо, нульова гіпотеза полягає в тому, що тестована послідовність дійсно випадкова (з точки зору конкретного тесту), а альтернативна — послідовність не випадкова. Цей механізм використовується у всіх трьох наборах тестів, які ми аналізуємо. Детальніше про перевірку статистичних гіпотез буде описано в підрозділі, де досліджується пакет тестів NIST.

Алгоритм перевірки статистичних гіпотез складається з наступних кроків (рис. 1).


*Рис. 1. Алгоритм перевірки статистичних гіпотез*

Статистичний аналіз послідовностей, як правило, проходить у два етапи [2], [6], [10]. Перший етап можна назвати підготовчим, він трудомісткий, тут виконується основна кількість обчислень [2], [6]:

1. За допомогою генератора, що досліджується, формуються випадкові послідовності;
2. Для кожної послідовності обчислюється статистика тесту. Якщо працює батарея тестів (проводиться відразу кілька тестів), статистика по послідовності обчислюється для кожного тесту;
3. Для кожної послідовності обчислюється ймовірність значущості;
4. Отримані статистики та ймовірності значущості зберігаються.

На другому етапі проводиться обробка отриманих результатів [2], [10]:

1. За допомогою критеріїв згоди перевіряються гіпотези щодо відповідності розподілів статистик та ймовірностей значущості гіпотетичних розподілів;
2. Визначається число послідовностей, що пройшли тест. Будується довірчий інтервал для останньої величини;
3. Приймається рішення про те, чи пройдено тест;
4. Остаточні висновки.

Схематично цей процес виглядатиме приблизно як на рис. 2 [15], [19].

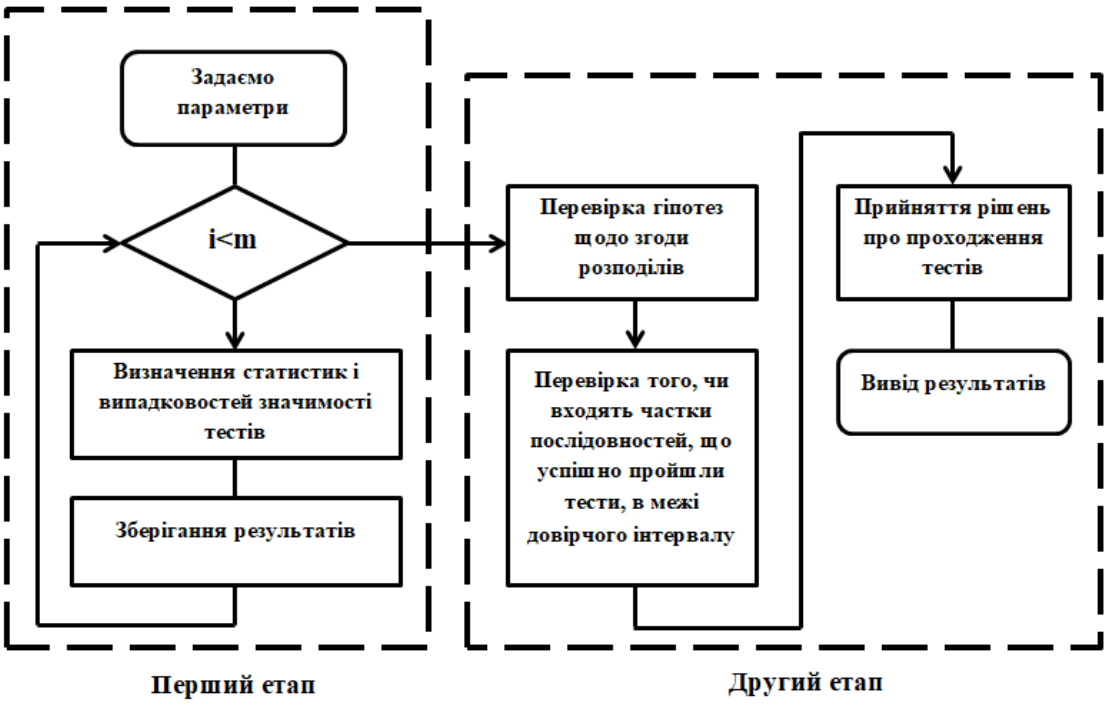


*Рис. 2. Статистичний аналіз послідовностей*

В даний час найбільш відомі наступні набори статистичних тестів (реалізовані у вигляді програмного забезпечення), призначені для статистичного тестування послідовностей [3], [5], [11]:

- NIST Statistical Test Suite (NIST STS) — набір статистичних тестів, розроблений у NIST [3];
- DieHard — набір статистичних тестів, розроблений математиком зі США Джорджем Магсаглією [5];

- TestU01 — набір статистичних тестів, розроблений в Монреальському університеті [11];
- RaBiGeTe — набір статистичних тестів, забезпечений зручним GUI для Windows, розроблений в Італії [3];
- PractRand — набір статистичних тестів, розроблений програмістом із США К. Доті-Хамфрі [3];

Крім перелічених існує ще кілька наборів статистичних тестів (наприклад, Crypt-X, Ent та ін.), які в даний час з різних причин рідко використовуються і не підтримуються [12].

Порівняльне дослідження зосереджено на найбільш часто використовуваних тестах — NIST, DieHard та TestU01.

Крім пакетів, що містять набори тестів для багатостороннього дослідження статистичних властивостей ВП і ПВП, існують окремі тести, спрямовані на більш точний і повний аналіз послідовностей [14], [20].

Завжди необхідно проводити тестування для дослідження статистичної безпеки генераторів, але варто мати на увазі, що воно не здатне замістити криптоаналіз [20]. Важливо також враховувати контекст застосування генераторів та вимоги конкретного застосування при виборі тестів для оцінки їх ефективності.

## РЕЗУЛЬТАТИ ДОСЛІДЖЕННЯ

**Пакет тестів NIST.** The National Institute of Standards and Technology — Національний інститут стандартів і технологій є підрозділом управління з питань технологій США, яке входить до складу одного з агентств Міністерства Торгівлі США [1]. NIST STS (Statistical Test Suite) — це набір статистичних тестів, розроблений Національним інститутом стандартів і технологій (NIST) для оцінки статистичних характеристик ГПВП і ГВП [1] – [4]. Пакет NIST STS вважається найбільш прийнятним з точки зору суворості оцінки властивостей ГПВП, доступним для використання на різних платформах та ефективним за витратами машинного часу [2]. NIST STS забезпечує всебічну перевірку якості ГВП відповідно до обґрунтованої методики та надає критерії для оцінки якості не лише окремого генератора, а й загальної групи [3], [4].

При цьому, застосування інших тестів не виключено, але можна стверджувати, що пакет NIST фактично є ключовим інструментом для дослідження статистичної придатності послідовностей, що використовуються у сфері криптографічного захисту інформації [9].

Цей набір включає 16 різноманітних тестів [4], [13], які призначені для перевірки гіпотези про випадковість двійкових послідовностей, незалежно від їхньої довжини, що може бути згенерованою як апаратними, так і програмними засобами. Тести NIST STS допомагають виявити різноманітні аномалії та недоліки у генераторах, сприяючи створенню надійних інструментів для випадкового генерування чисел у різних областях, включаючи криптографію, статистику, інформаційну безпеку тощо [17].

Тести NIST ґрунтуються на перевірці статистичної гіпотези про випадковість досліджуваної послідовності. У статистичній теорії перевірки гіпотез [11], [18], [24] – [26] дві основні — нульова гіпотеза ($H_0$) та альтернативна гіпотеза ($H_1$):

– Нульова гіпотеза ($H_0$): це припущення або гіпотеза, що досліджувана послідовність є випадковою або що не існує відмінності від випадкової послідовності. У контексті тестів NIST, нульова гіпотеза передбачає, що досліджувана послідовність відповідає стандартам випадковості [22].

– Альтернативна гіпотеза ($H_1$): це альтернативне припущення, яке виражає можливість відхилення від нульової гіпотези. У контексті тестів NIST, альтернативна гіпотеза вказує на те, що досліджувана послідовність не відповідає стандартам випадковості, і містить аномалії або неправильності [23].

Для усіх тестів відбувається порівняння обчисленої статистики з певним заздалегідь теоретично визначеним значенням. Це значення обчислюється для обраного в кожному тесті еталонного розбиття випадкової величини. Як еталонне розбиття використовують [21]:

– Розбиття $\chi^2$ (для 10 тестів із 16) передбачає порівняння ступеня узгодженості частот $F_i$ з відповідними частотами $f_i$ передбачуваного розподілу. Статистичним показником при цьому є величина [21], [22]

$$\chi^2 = \sum_{i=1}^{k} \frac{(F_i - f_i)^2}{f_i}. \quad (1)$$

– Нормальне розбиття, зокрема, стандартне нормальне у двох тестах і одностороннє усічене нормальне у трьох інших тестах. Порівнюється обчислена статистика згенерованої ПВП з визначеним значенням. Статистичні дані центруються і нормуються за середнім квадратичним відхиленням ($z = \frac{x-\mu}{\sigma}$). Обчислення p-value відбувається за допомогою функції помилок, яка є додатковою [23]

$$erfc(z) = \frac{2}{\sqrt{\pi}} \int_z^{\infty} e^{-u^2} du. \quad (2)$$

Обчислення p-value відбувається за допомогою неповної гамма-функції [23]:

$$Q(a,x) = \frac{\Gamma(a,x)}{\Gamma(a)} = \frac{1}{\Gamma(a)} \int_x^{\infty} e^{-t} t^{a-1} dt,\ \Gamma(z) = \int_x^{\infty} e^{-t} t^{z-1} dt. \quad (3)$$

Під час застосування тестів потрібно коректно відбирати показники параметрів у зв'язку з їх параметричністю.

Як уже було згадано, набір включає 16 тестів. Проте, залежно від вхідних параметрів, визначається 189 показників p-value. Їх значення можна представляти як роботу різних тестів. У табл. 1 представлені дані, щодо кількості обрахованих показників p-value із наведенням порядкового номеру тесту у дужках [4], [13]. Також надається короткий опис тесту.

*Таблиця 1*

**Зведені дані з усіх тестів**

| № п/п | Статистичний тест | Кількість значень p-value | Опис |
|---|---|---|---|
| 1 | Частотний (монобітний) тест | 1 (1) | Під час цього тесту аналізується частота появи 1 і 0 в послідовності. Цей тест допомагає виявити відхилення від рівномірного розподілу бітів у вхідній послідовності. |
| 2 | Частотний тест всередині блоку | 1 (2) | Під час цього тесту випадкова послідовність розділяється на блоки, і для кожного блоку аналізується частота появи 1 і 0. Цей аналіз може допомогти виявити нерегулярності або відмінності у частоті появи бітів всередині окремих блоків. |
| 3 | Перевірка накопичених сум | 2 (2–4) | Тест, який оцінює величину відхилення накопичених сум елементів випадкової послідовності від їх теоретичного |

| | | | |
|---|---|---|---|
| | | | очікуваного значення. Під час цього тесту послідовність розбивається на блоки, і для кожного блоку обчислюється накопичена сума. Далі порівнюється фактична накопичена сума з її теоретичним значенням. |
| 4 | Перевірка серій | 1 (5) | Тест, який аналізує кількість безперервних серій однакових бітів (наприклад, послідовності одиниць або нулів) у випадковій послідовності. Цей тест оцінює, наскільки швидко змінюється послідовність бітів у вихідній послідовності. |
| 5 | Перевірка максимальної довжини серії у блоці | 1 (6) | Тест, який аналізує найбільшу кількість однакових бітів, які послідовно зустрічаються у певному блоці даних. Цей тест призначений для виявлення нерегулярностей у вихідній послідовності, таких як велика кількість серій однакових бітів. |
| 6 | Перевірка рангу двійкової матриці | 1 (7) | Тест, який визначає міру узгодження між спостережуваним рангом різних підмножин послідовності бітів та їх теоретично очікуваним рангом у випадковій послідовності. Цей тест може виявити відхилення від очікуваного закону розподілу рангів. |
| 7 | Спектральний тест на основі дискретного перетворення Фур'є | 1 (8) | В цьому тесті використовується дискретне перетворення Фур'є для перетворення послідовності випадкових чисел у частотний домен. Дозволяє виявити випадковість або невипадковість послідовності, а також виявити наявність неперіодичних складових чи інших аномалій. |
| 8 | Перевірка шаблонів, що перекриваються | 1 (9) | Цей тест оцінює кількість збігів або перекриття між шаблонами в послідовності. Він перевіряє, наскільки часто входять взаємопов'язані шаблони у послідовності випадкових чисел. Цей тест допомагає виявити недоліки або аномалії у генераторі випадкових чисел, пов'язані з розподілом шаблонів у послідовності. |
| 9 | Універсальний тест Маурера | 1 (10) | Цей тест оцінює, наскільки послідовності випадкових бітів відповідають вимогам випадковості. Він базується на обчисленні показника складності послідовності, який порівнюється з емпіричними значеннями для випадкових послідовностей. Універсальний тест Маурера може виявити відхилення від випадковості, такі як структурованість, кореляція або інші аномалії. |
| 10 | Ентропійний тест | 1 (11) | У цьому тесті вимірюється кількість інформації, яку містить послідовність, тобто її ентропія. Чим більше ентропія, тим більш непередбачуваною вважається послідовність, що вважається ознакою випадковості. Ентропійний тест може виявити відхилення від випадковості, такі як структурованість, повторюваність або недостатня складність. |
| 11 | Перевірка випадкових відхилень | 8 (12–19) | У цьому тесті досліджується, наскільки послідовність відповідає властивостям випадкового процесу, тобто наскільки вона демонструє випадковість. Під час перевірки випадкових відхилень аналізуються різні статистичні характеристики даних, такі як розподіл частот, автокореляція, ентропія тощо. Ці характеристики порівнюються з очікуваними для випадкового процесу. |
| 12 | Перевірка випадкових відхилень (варіантний) | 18 (20–37) | Цей тест оцінює, наскільки послідовність даних відповідає очікуваному випадковому розподілу і демонструє випадковість. Під час варіантного аналізу перевірки випадкових відхилень аналізуються різні характеристики даних, такі як розподіл частот, кумулятивні суми, кореляція між даними тощо. Потім проводиться порівняння фактичних значень цих характеристик з очікуваними для випадкового процесу. |

| | | | |
|---|---|---|---|
| 13 | Тест на підпослідовності | 2 (38–39) | Під час цього тесту аналізуються різні підмножини або підпослідовності даних на наявність структур або взаємозв'язків, які можуть вказувати на їх нерівномірність або випадковість. Цей тест дозволяє виявити будь-які нерівномірності або відмінності в розподілі даних у великій послідовності. Шляхом аналізу підпослідовностей даних можна виявити відхилення від очікуваного випадкового розподілу, такі як відсутність рівномірності, наявність кореляцій або взаємозв'язків між даними. |
| 14 | Перевірка стиснення за алгоритмом Лемпеля-Зіва | 1 (40) | Під час цієї перевірки аналізуються вхідні дані для виявлення ступеня стиснення, досягнутого за допомогою цього алгоритму. Ця перевірка дозволяє виявити, наскільки добре даний набір даних піддається стисненню і чи є його стиснення оптимальним для даного алгоритму. |
| 15 | Перевірка шаблонів, що не перекриваються | 148 (41–188) | Цей тест аналізує вхідні дані, щоб виявити наявність і розподіл таких шаблонів. Цей тест допомагає визначити, наскільки добре генератор випадкових чисел виробляє послідовності даних, що не мають очевидних залежностей або патернів, що можуть бути передбачені. |
| 16 | Перевірка лінійної складності | 1 (189) | У цьому тесті розглядаються усі можливі лінійні комбінації бітів вхідної послідовності та оцінюється ймовірність виникнення конкретних лінійних залежностей. Основна ідея полягає в тому, щоб перевірити, чи можна побудувати лінійні рівняння, які точно передбачають значення бітів вихідної послідовності. Якщо вдається побудувати такі рівняння, то це свідчить про недостатню складність генератора. |

У тестах застосовуються різні алгоритми і об'єми частин послідовності, що обробляються.

У першій частині тестів досліджуються найпростіші властивості послідовності (робота проводиться з окремими бітами). Тести із другої частини обробляють m-бітові блоки. Блоки довжиною понад 1000 біт досліджують найскладніші тести: перевірка лінійної складності, спектральний тест, перевірка рангу двійкової матриці.

Висновки щодо успішного проходження тесту для усієї вибірки формуються на основі двох критеріїв [22]:

1. Включення обчисленої величини $Pr$, яка представляє частку послідовностей, що успішно пройшли тест у пакеті тестів NIST, в довірчий інтервал
$$\left[(1-\alpha)-3\sqrt{\frac{\alpha(1-\alpha)}{m}},(1-\alpha)+3\sqrt{\frac{\alpha(1-\alpha)}{m}}\right], \quad (4)$$
де m – розмір вибірки.
2. Наскільки рівномірно розподілені p-value від 0 до 1.

Пакетом NIST для перевірки рівномірності обчислюється величина статистики, що розбиває весь інтервал [0,1] на 10 підінтервалів ($k = 10$): [0; 0.1), [0.1; 0.2), ..., [0.9; 1).
$$\chi^2 = \frac{\sum_{i=1}^{k}(v^i - {}^{m}/_{k})^2}{{}^{m}/_{k}}, \quad (5)$$
де $v^i$ представляє собою кількість значень p-value, які потрапляють у кожний інтервал. За критерієм $\chi^2$ проводиться перевірка того, наскільки реальний розподіл значень p-value подібний до теоретичного (рівномірного). У випадку значної кількості пройдених перевірок послідовностями, розподіл цієї статистики повинен наближатися до розподілу $\chi^2$ з числом ступенів $(k-1)$.

**Пакет тестів DieHard.** Тести Diehard представляють собою набір статистичних тестів, призначених для оцінки якості згенерованих випадкових числових послідовностей (ВЧП) [5]. Розроблений Джорджем Магсаглією протягом шести років, цей набір вперше був опублікований у 1995 році на CD-ROM з випадковими числами [5], [6].

Набір DieHard складається з наступних тестів.

*Тест проміжків між днями народження.* Якщо випадково вибрати n днів народження в «році» з m днів і розглянути список проміжків між днями народження, то виявиться, що число j елементів цього списку, що зустрічаються більш ніж один раз, асимптотично розподілені за Пуассоном [5], [11]. Тест полягає в тому, що послідовність, яка тестується, інтерпретується як послідовність таких «днів народження» і перевіряється відповідність розподілу значень j очікуваному для випадкової послідовності. Назва тесту пов'язана із відомим парадоксом днів народження.

*Тест перестановок, які перетинаються.* Розглядає послідовність, що тестується, як послідовність 32-розрядних чисел [10], [16]. Перевіряє розподіл усіх можливих перестановок п'яти послідовних чисел із цієї послідовності (всі 5! перестановок мають бути рівноймовірними).

*Тест рангів матриць.* Тестована послідовність трактується як послідовність бінарних матриць (у різних варіантах тесту використовуються матриці різних розмірів: 6 x 8,31 x 31,32 x 32) [14]. Обчислюються ранги цих матриць. Їхній розподіл перевіряється на відповідність очікуваному.

*Мавпячі тести.* Тестована послідовність інтерпретується як послідовність двійкових слів певної довжини, що перетинаються. Послідовність цих слів поділяється на підпослідовність і обчислюється кількість слів, які до неї не входять. Розподіл цього значення перевіряється на відповідність очікуваному для випадкової послідовності [5], [11].

*Підрахунок числа одиниць у потоці байтів.* Розглядається послідовність одиничних бітів з байтів тестованої послідовності. З неї виходить послідовність букв A, B, C, D, E шляхом заміни 0, 1, 2 на A, 3 на B, 4 на C, 5 на D, а 6, 7, 8 на E [19]. Обчислюються частоти для кожного слова довжини 5 (всього $5^5$ слів). Отримані частоти перевіряються на відповідність ймовірності слів очікуваним для випадкової послідовності.

*Підрахунок числа одиниць у деяких байтах.* Аналогічний попередньому, однак вибирається не кожен байт послідовності, а деякі вказані 8 біт з кожного 32-бітного слова [19].

*Тест на паркування.* У квадраті розміру 100 x 100 випадковим чином (випадкові числа формуються з послідовності, що тестується) розміщуються («паркуються») одиничні кола [24]. Якщо чергове коло перетинається з вже наявним у квадраті колом, то ітерацію закінчують. Усього виконують 12000 таких ітерацій, після чого порівнюють розподіл числа успішно припаркованих кіл з очікуваним (воно має бути близьким до нормального).

*Тест на мінімальну відстань* — 8000 точок випадковим чином (випадкові числа формуються з послідовності, що тестується) розміщуються в квадраті розміру 10000 x 10000, після чого розглядаються відстані між кожною парою точок і вибирається мінімальна серед них [10]. Виконується 100 таких ітерацій. Квадрат мінімальної відстані повинен мати розподіл близький до експоненційного.

*Тест випадкових сфер.* У кубі, ребро якого дорівнює 1000, випадково вибираються 4000 точок (випадкові числа формуються з послідовності, що тестується) [16]. У кожну з цих точок розміщується центр сфери, радіус якої дорівнює відстані до найближчої точки. Таких ітерацій відбувається певна кількість і перевіряється близькість розподілу об'єму кулі, обмеженої мінімально з цих сфер, очікуваного для випадкової послідовності.

ISSN 2663 - 4023

*Тест стиснення.* З послідовності, що тестується, генеруються дійсні числа $u \in [0,1)$, далі знаходиться таке мінімальне i, що $t_i = 1$, де $t_{i+1} = [t_i u], t_0 = 2^{31}$. Проводиться певна кількість таких операцій, після чого розподіл величини перевіряється на відповідність очікуваному [16].

*Тест сум, що перетинаються.* На основі послідовності, що тестується, генерується послідовність дійсних чисел $u_1, u_2, \ldots \in [0,1)$. Обчислюються значення

$$s_i = \sum_{j=i}^{i+100} u_j, i = 1, 2, \ldots \quad (6)$$

Спеціальним чином перевіряється відповідність їхнього розподілу очікуваному [24].

*Тест послідовностей.* На основі послідовності, що тестується, генерується послідовність дійсних чисел. У ній проводиться підрахунок кількості безперервних монотонних (зростаючих та спадних) підпослідовностей [16].

*Тест гри в кістки* — на основі послідовності, що тестується, беруться дані для гри в крепс (craps, вид гри в кістки). Для досить великої кількості таких ігор підраховуються кількість перемог та кількість кидків [11]. Перевіряється відповідність розподілів цих чисел очікуваним.

Зазначимо, що в багатьох тестах перевіряється відповідність розподілу деякої вибірки, обчисленої на основі послідовності, що тестується, розподілу, який очікується в тому випадку, якщо тестована послідовність є випадковою. Найчастіше ця перевірка здійснюється або з урахуванням критерію $\chi^2$ [21], [22], [25] або (рідше) з урахуванням критерію Колмогорова — Смирнова.

Набір тестів DieHard з'явився досить давно (в 1995 році) і здобув велику популярність, оскільки він є фактично першим широко відомим набором статистичних тестів, який можна було використовувати для статистичного тестування криптографічних алгоритмів. Однак він не вільний від недоліків — вибір тестів, що входять до нього, досить неоднозначний і інтерпретація їх результатів буває часом важкою для розуміння, а програмна реалізація страждає помилками [26].

**Набір тестів TestU01.** TestU01 — це набір програмних інструментів, розроблений на мові програмування ANSI C, що надає засоби для емпіричного статистичного тестування ГВП та ГПВП [7]. Ці тести можуть досліджувати генератори, реалізовані користувачем. Також є можливість протестувати генератори, які доступні у бібліотеці. Батареї тестів досліджують двійкові та числові послідовності. Числові дані мають бути рівномірно розподілені від 0 до 1. Також доступна можливість створювати вектори точок, які генеруються різними генераторами [7], [8].

У цій бібліотеці є шість різних батарей: Small Crush, Crush, Big Crush, Rabbit, Alphabit, Block Alphabit [12], [15], [18], [20], [27].

Small Crush — це невелика за обсягом батарея тестів, яка містить лише 10 тестів, але вона є досить швидкою і компактною [12]. Для проведення тестування вона зчитує файл, який містить послідовність бітових даних. Обмеження на розмір файлу становить трохи менше 51,320,000 бітів. Ці тести зазвичай застосовуються для визначення доцільності застосування наступних більш суворих батарей.

До складу батареї Small Crush входять наступні тести: smarsa_BirthdaySpacings, smarsa_MatrixRank; sknuth_Collision, sknuth_Gap, sknuth_SimpPoker, sknuth_CouponCollector, sknuth_MaxOft; svaria_WeightDistrib; sstring_HammingIndep; swalk_RandomWalk1. Назви тестів розпочинаються з літери «s», за якою йде префікс, що вказує на походження тесту. Наприклад, «knuth» вказує на тести, розроблені Дональдом Кнутом, а «marsa» — на тести, розроблені Джорджем Марсалію.

Crush містить у собі 96 достатньо вимогливих тестів таких як класичні тести Кнута і багато інших [15]. Мінімальна кількість об'єму даних для кожного окремого тесту —

30 біт вхідної послідовності, але один із тестів потребує 31 біт. Щоб дослідити послідовність усіма тестами із батареї потрібно 235 біт даних.

Big Crush містить 106 тестів і є найбільш суворим набором випробувань [18]. Для одного із цих тестів потрібно 31 біт вхідної послідовності чисел. Для решти необхідно щоб було хоча б 30 бітів даних як і в інших батареях; інакше їх вважатимуть непройденими. Щоб дослідити послідовність усіма тестами із батареї потрібно 238 біт даних.

Rabbit містить 38 тестів. Ця батарея зосереджена переважно на перевірці двійкових даних, але деякі тести працюють з фіксованими параметрами [20]. Таким чином, дані, розмір яких перевищує 64 мегабайти, можуть призвести до помилки в одному з тестів або до переповнення оперативної пам'яті. Розмір вхідних даних не має значення.

Alphabit і Block Alphabit містять у собі по 17 тестів кожен [27]. Батарея Alphabit була розроблена для перевірки апаратних ГВП. Вона виконує дев'ять послідовних тестів, після кожного з яких вхідні дані переписуються.

На персональному комп'ютері для простого генератора час роботи батареї тестів [28] Small Crush займає кілька хвилин, Crush — близько години, Big Crush — близько десятка годин.

Головна мета будь-якого генератора — пройти тест Big Crush, виконання якого може тривати до 24 годин [18]. Результатом роботи TestU01 є вивід p-value, які необхідні для оцінки статичної безпеки генератора. Також наводяться значення, які виходять за межі інтервалу [0,01,...,0,99]. Недоліком TestU01 є те, що він працює з фіксованою кількістю даних і відкидає молодший біт (для деяких тестів навіть два біти) з 32-розрядних чисел, що перевіряються [29]. Важливо зазначити, що з академічних міркувань TestU01 призначений для перевірки 32-бітних чисел, однак ГВП, які використовуються сьогодні, створюють 64-бітні числа. Дійсно, багато років тому більшість ГВП виробляли, в кращому випадку, 31-бітні випадкові значення. Але результат із 64-бітних чисел можна перетворити на 32-бітні. Фактично, головне занепокоєння викликає p-value.

**Обговорення результатів дослідження.** Всі три набори тестів (NIST, DieHard і TestU01) мають свої переваги, недоліки та обмеження (табл. 2). Вони широко використовуються для оцінки статистичних характеристик ГВП та ГПВП у криптографічних застосуваннях, наукових дослідженнях та аналізі статистики. Вибір конкретного тесту повинен залежати від потреб і умов випробувань, але у всіх випадках важливо ретельно вивчити характеристики кожного тесту для забезпечення точності та ефективності оцінки.

*Таблиця 2*

**Порівняння тестів NIST, DieHard та TestU01**

| Тест / Параметр | NIST | DieHard | TestU01 |
|---|---|---|---|
| Призначення | Оцінка статистичних характеристик ГВП та ГПВП | Оцінка статистичних характеристик ГВЧП | Оцінка статистичних характеристик ГВП, ГПВП та алгоритмів генерації |
| Складність тестів | (Середня) Складність тестів NIST визначається різноманітністю їхніх статистичних методів, | (Висока) Складність тестів DieHard полягає в їхній різноманітності, вимогах до великої кількості вхідних даних та високих | (Дуже Висока) Тести TestU01 відзначаються високою складністю через їхню різноманітність, вимоги до обсягів вхідних |

| | | | |
|---|---|---|---|
| | потребою у складних обчисленнях та аналізі результатів. | обчислювальних вимогах, що робить їх більш вимогливими для використання та розуміння. | даних, точність результатів та потребу уважного аналізу отриманих даних. |
| Обсяг тестів | 16 наборів, у які входять 189 тестів | 15 тестів в пакеті Diehard, деякі повторюються за різними параметрами, видають 213 p-value загалом | 6 батарей, у які входять 284 тести |
| Точність оцінки | (Висока) Точність оцінки тестів NIST залежить від стандартизованості процедур тестування та узгодженості результатів між різними дослідниками та лабораторіями. | (Середня) Якщо використовувати тести DieHard належним чином, з великим обсягом вхідних даних і з урахуванням їхніх особливостей, точність оцінки може бути досить високою. Однак, для більш точної оцінки рекомендується також використовувати інші набори тестів, такі як TestU01, для додаткової перевірки. | (Висока) Якщо використовувати тести TestU01 належним чином, з великим обсягом вхідних даних і з урахуванням їхніх особливостей, точність оцінки може бути досить високою. |
| Популярність | Висока | Середня | Середня |
| Проблеми та обмеження | 1.Тести NIST не можуть виявити всі можливі дефекти.<br>2.Критерії, використовувані в тестах NIST, можуть бути статичними і не охоплювати всі можливі види відхилень, які можуть виникнути.<br>3.Деякі дефекти можуть залишатися непоміченими тестами NIST через їх обмежену чутливість до певних видів відхилень.<br>4.Деякі тести можуть бути чутливими до властивостей вхідних даних, що може призвести до некоректних результатів, якщо вхідні дані не відповідають очікуванням. | 1. Кількість тестів обмежена порівняно з іншими більш сучасними наборами тестів, такими як TestU01.<br>2. Деякі тести DieHard можуть бути менш чутливими до певних видів недоліків у порівнянні з більш сучасними тестами.<br>3. DieHard був створений понад 20 років тому, і потребує модернізації.<br>4. Тести DieHard не можуть враховувати всі можливі сценарії використання генераторів.<br>5. Деякі тести в DieHard можуть мати низьку роздільну здатність, що означає, що вони можуть не виявляти незначних аномалій або недоліків. | 1. TestU01 включає в себе велику кількість тестів, які можуть вимагати значних обчислювальних ресурсів для їх виконання.<br>2. Деякі частини TestU01 можуть бути складними для налаштування, розуміння та використання без відповідного досвіду або експертності у сфері генерації.<br>3. Оскільки TestU01 генерує велику кількість результатів, вони можуть мати обмежену роздільну здатність, що ускладнює виявлення незначних аномалій або недоліків у генерованих послідовностях. |
| Новаторство та розвиток | 1.Одним із новаторських аспектів тестів NIST є їх універсальність та гнучкість. | 1. Новаторство тестів DieHard полягало в тому, що вони вперше враховували широкий спектр статистичних | 1. TestU01 містить широкий набір статистичних тестів, які враховують різноманітні аспекти якості ГВЧ, включаючи |

| | | | |
|---|---|---|---|
| | 2.Вони охоплюють широкий спектр статистичних тестів, які можуть бути застосовані до різних типів вихідних послідовностей.<br>3.Крім того, тести NIST постійно оновлюються та модернізуються з урахуванням останніх досягнень в області криптографії та статистичного аналізу даних.<br>4.Ще одним аспектом новаторства тестів NIST є їх відкритий характер.<br>5.Вони доступні для використання та перевірки будь-яким зацікавленим особам чи організаціям. | властивостей послідовностей випадкових чисел, включаючи рівномірність розподілу, автокореляцію, серійність, а також більш складні властивості, такі як тести на шаблони, які перекриваються, і аналіз довгих підпослідовностей.<br>2. TestU01 був розроблений як наступний етап у розвитку тестів, зокрема, як альтернатива DieHard, які потребували модернізації. | рівномірність розподілу, автокореляцію, серійність, стійкість до криптографічних атак та інші.<br>2. TestU01 побудований у модульній архітектурі, що дозволяє додавати нові тести та розширювати функціональні можливості без значних змін у базовій структурі.<br>3. TestU01 використовує сучасні методи статистичного аналізу даних, включаючи техніки машинного навчання та обробки сигналів.<br>4. TestU01 розроблений для роботи на різних апаратних платформах і операційних системах, що дозволяє використовувати його в різних середовищах без зайвих зусиль.<br>5. TestU01 став стандартом у галузі статистичного тестування генераторів випадкових чисел, що дозволяє користувачам обмінюватися досвідом та знаннями у цій області. |

## ВИСНОВКИ ТА ПЕРСПЕКТИВИ ПОДАЛЬШИХ ДОСЛІДЖЕНЬ

У сучасному світі комп’ютерних технологій, де випадкові та псевдовипадкові послідовності використовуються в широкому спектрі застосувань, важливо мати надійні й ефективні інструменти для перевірки їх якості. Кожен з розглянутих тестів — NIST, DieHard і TestU01 — має свої особливості, які варто враховувати при їхньому використанні.

У нашому порівняльному дослідженні тестів NIST, DieHard та TestU01 для оцінки статистичних характеристик генераторів ми виявили, що кожен з цих тестів має свої переваги та обмеження. Тести NIST широко використовуються в індустрії та дослідженнях, вони мають ряд підходів для оцінки різних аспектів псевдовипадкових послідовностей. Тести DieHard зосереджуються на складних статистичних властивостях та зазвичай застосовуються для більш глибокого аналізу послідовностей. З іншого боку, тести TestU01 володіють більшою чутливістю та розгалуженістю, дозволяючи виявити більш широкий спектр недоліків. У табл. 2 ми систематизували і порівняли їхні особливості, сильні та слабкі сторони для того щоб дослідникам було легше оцінити кожен тест. Це може допомогти зробити висновки щодо використання певних тестів у конкретних ситуаціях та обрати найбільш підходящий тест для конкретних потреб дослідження.

Враховуючи ці фактори, вибір конкретного набору тестів повинен залежати від конкретних потреб і умов випробувань. Важливо ретельно аналізувати властивості кожного тесту та його відповідність поставленим завданням, щоб забезпечити надійність та ефективність оцінки ГВП і ГПВП.

Загальний висновок полягає в тому, що комбінація різних тестів може бути найкращим підходом для оцінки якості ГВП або ГПВП, забезпечуючи більш об'єктивні та надійні результати. Однак важливо також враховувати контекст застосування генераторів та вимоги конкретного застосування при виборі тестів для оцінки їх ефективності.

Стаття є корисним ресурсом для спеціалістів або організацій, які працюють у сфері криптографії, комп'ютерної безпеки, моделювання систем, статистичного аналізу та інформаційної безпеки в мережах. Результати дослідження тестів NIST, DieHard і TestU01 для оцінки статистичних характеристик будуть використані у подальшому дослідженні для розроблення генерування псевдовипадкових послідовностей з покращеними характеристиками для систем кібербезпеки.

## СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Almaraz Luengo, E., & Román Villaizán, J. (2023). Cryptographically Secured Pseudo-Random Number Generators: Analysis and Testing with NIST Statistical Test Suite. *Mathematics, 11(23),* 4812. https://doi.org/10.3390/math11234812
2. Mahalingam, H., Rethinam, S., Janakiraman, S., & Rengarajan, A. (2023). Non-Identical Inverter Rings as an Entropy Source: NIST-90B-Verified TRNG Architecture on FPGAs for IoT Device Integrity. *Mathematics, 11(4),* 1049. https://doi.org/10.3390/math11041049
3. Kim, Y., & Yeom, Y. (2021). Accelerated implementation for testing IID assumption of NIST SP 800-90B using GPU. *PeerJ Computer Science*, *7*, e404. https://doi.org/10.7717/peerj-cs.404
4. Meitei, H. B., & Kumar, M. (2023). Implementation of a secure wireless communication system using true random number generator for internet of things. *Indonesian Journal of Electrical Engineering and Computer Science*, *30(2),* 982. https://doi.org/10.11591/ijeecs.v30.i2.pp982-992
5. Al-Daraiseh, A., Sanjalawe, Y., Al-E'mari, S., Fraihat, S., Bany Taha, M., & Al-Muhammed, M. (2023). Cryptographic Grade Chaotic Random Number Generator Based on Tent-Map. *Journal of Sensor and Actuator Networks, 12(5),* 73. https://doi.org/10.3390/jsan12050073
6. Vaskova, A., Lopez-Ongil, C., Millan, E. S., Jimenez-Horas, A., & Entrena, L. (2011). Accelerating secure circuit design with hardware implementation of Diehard Battery of tests of randomness. *2011 IEEE 17th International On-Line Testing Symposium*, 179–181. https://doi.org/10.1109/IOLTS.2011.5993835
7. L'Ecuyer, P., & Simard, R. (2007). TestU01: A C library for empirical testing of random number generators. *ACM Transactions on Mathematical Software*, *33(4),* 1–40. https://doi.org/10.1145/1268776.1268777
8. Suciu, A., Toma, R. A., & Marton, K. (2012). Parallel implementation of the TestU01 statistical test suite. *IEEE 8th International Conference on Intelligent Computer Communication and Processing*, 317–322. https://doi.org/10.1109/ICCP.2012.6356206
9. Almaraz Luengo, E., & Gragera, C. (2023). Critical Analysis of Beta Random Variable Generation Methods. *Mathematics, 11(24),* 4893. https://doi.org/10.3390/math11244893
10. Sriram, V., Srikamakshi, M., Jegadish Kumar, K. J., & Nagarajan, K. K. (2020). Randomness Analysis of YUGAM-128 Using Diehard Test Suite. *Innovative Data Communication Technologies and Application, 46,* 600–607. https://doi.org/10.1007/978-3-030-38040-3_68
11. Araki, S., Wu, J.-H., & Yan, J.-J. (2024). A Novel Design of Random Number Generators Using Chaos-Based Extremum Coding. *IEEE Access*, *12*, 24039–24047. https://doi.org/10.1109/ACCESS.2024.3365638
12. Sleem, L., & Couturier, R. (2020). TestU01 and Practrand: Tools for a randomness evaluation for famous multimedia ciphers. *Multimedia Tools and Applications*, *79(33–34),* 24075–24088. https://doi.org/10.1007/s11042-020-09108-w

13. Chen, Y., Tian, Y., Zhou, R., Castro, D. M., Guo, D., & Zhou, Q. (2024). NDSTRNG: Non-deterministic Sampling-based True Random Number Generator on SoC FPGA Systems. *IEEE Transactions on Computers*, 1–14. https://doi.org/10.1109/TC.2024.3365955
14. Lecca, C., Zegarra, A., & Santisteban, J. (2024). Random Number Generator Based on Hopfield Neural Network with Xorshift and Genetic Algorithms. *Advances in Computational Intelligence*, *14391*, 283–295. https://doi.org/10.1007/978-3-031-47765-2_21
15. Li, Y., Wang, Q., & Yu, S. (2024). A novel hybrid scheme for chaotic image encryption. *Physica Scripta*, *99(4)*, 045244. https://doi.org/10.1088/1402-4896/ad3171
16. Abdulhameed, H. A., Abdalmaaen, H. F., Mohammed, A. T., Mosleh, M. F., & Abdulhameed, A. A. (2022). A Lightweight Hybrid Cryptographic Algorithm for WSNs Tested by the Diehard Tests and the Raspberry Pi. *2022 International Conference on Computer Science and Software Engineering (CSASE)*, 271–276. https://doi.org/10.1109/CSASE51777.2022.9759589
17. Kumar, V., & Pravinkumar, P. (2023). Quantum random number generator on IBM QX. *Journal of Cryptographic Engineering*. https://doi.org/10.1007/s13389-023-00341-1
18. Aldossari, H., & Mascagni, M. (2022). Scrambling additive lagged-Fibonacci generators. *Monte Carlo Methods and Applications*, *28(3)*, 199–210. https://doi.org/10.1515/mcma-2022-2115
19. Hussein, S. N., & Al-Alak, S. M. (2021). Secret Keys Extraction Using Light Weight Schemes for Data Ciphering. *Journal of Physics: Conference Series*, *1999(1)*, 012114. https://doi.org/10.1088/1742-6596/1999/1/012114
20. Duda, C. K., Meier, K. A., & Newell, R. T. (2023). Development of a High Min-Entropy Quantum Random Number Generator Based on Amplified Spontaneous Emission. *Entropy*, *25(5)*, 731. https://doi.org/10.3390/e25050731
21. Zhao, W., & Ma, C. (2024). Modification of Intertwining Logistic Map and a Novel Pseudo Random Number Generator. *Symmetry*, *16(2)*, 169. https://doi.org/10.3390/sym16020169
22. Isakov, O. V., & Voitusik, S. S. (2023). Comparative analysis of digital noise generated by additive Fibonacci generators. *Ukrainian Journal of Information Technology*, *5(1)*, 67–76. https://doi.org/10.23939/ujit2023.01.067
23. Gafsi, M., Hafsa, A., & Machout, M. (2024). Hardware implementation of digital pseudo-random number generators for real-time applications. *Signal, Image and Video Processing*. https://doi.org/10.1007/s11760-024-03082-8
24. Ryan, C., Kshirsagar, M., Vaidya, G., Cunningham, A., & Sivaraman, R. (2022). Design of a cryptographically secure pseudo random number generator with grammatical evolution. *Scientific Reports*, *12(1)*, 8602. https://doi.org/10.1038/s41598-022-11613-x
25. Hussein, S. N., Obaid, A. H., & Jabbar, A. (2022). Encryption Symmetric secret Key in Wireless Sensor Network Using AES Algorithm. *Iraqi Journal of Science*, 5037–5045. https://doi.org/10.24996/ijs.2022.63.11.38
26. Semankiv, M. (2016). Assessment of statistical properties random numbers. *Bulletin of the National Technical University 'KhPI'. A Series of 'Information and Modeling'*, *0(21)*. https://doi.org/10.20998/2411-0558.2016.21.12
27. Álvarez, R., Martínez, F., & Zamora, A. (2022). Improving the Statistical Qualities of Pseudo Random Number Generators. *Symmetry*, *14(2)*, 269. https://doi.org/10.3390/sym14020269
28. Palacios-Luengas, L., Marcelín-Jiménez, R., Rodriguez-Colina, E., Pascoe-Chalke, M., Jiménez-Ramírez, O., & Vázquez-Medina, R. (2021). Function Composition from Sine Function and Skew Tent Map and Its Application to Pseudorandom Number Generators. *Applied Sciences*, *11(13)*, 5769. https://doi.org/10.3390/app11135769
29. Hurley-Smith, D., & Hernandez-Castro, J. (2020). Quantum Leap and Crash: Searching and Finding Bias in Quantum Random Number Generators. *ACM Transactions on Privacy and Security*, *23(3)*, 1–25. https://doi.org/10.1145/3398726

КІБЕРБЕЗПЕКА: освіта, наука, техніка
CYBERSECURITY: EDUCATION, SCIENCE, TECHNIQUE
№ 4 (24), 2024
ISSN 2663 - 4023

**Olena Niemkova**
Doctor of Sciences, Professor, Professor of the
Department of Information Technology Security
Lviv Polytechnic National University, Lviv, Ukraine
ORCID ID: 0000-0003-0690-2657
*olena.a.niemkova@lpnu.ua*

**Mykhailo Kikh**
Graduate student of the Department of Information Technology Security
Lviv Polytechnic National University, Lviv, Ukraine
ORCID ID: 0009-0007-1847-1862
*mykhailo.v.kikh@lpnu.ua*

# COMPARATIVE STUDY OF TESTS FOR ASSESSMENT OF STATISTICAL CHARACTERISTICS OF RANDOM AND PSEUDO-RANDOM SEQUENCE GENERATORS

**Abstract.** In the world of information security, computer science and cryptography, the issue of statistical security of generated sequences is very important. Statistical sequence security plays an important role in the following fields: cryptography, computer security, system modeling, statistical analysis, and information security in networks. This article is devoted to the study and comparison of test sets for evaluating the statistical properties of random and pseudorandom sequence generators. The comparison focuses on well-known test suites such as NIST, DieHard, and TestU01. These tests were selected for study because of their widespread use and recognized effectiveness in measuring the quality of generators. The article examines various aspects of these test suites, including purpose, complexity, scope, scoring accuracy, popularity, challenges and limitations, and innovation and development. NIST tests are widely used in cryptography and research, and they take a number of approaches to evaluate different aspects of random sequences. DieHard tests focus on complex statistical properties and are usually used for more in-depth analysis of generators. On the other hand, TestU01 tests have greater sensitivity and branching, allowing to detect a wider range of flaws in random number generators. A comparative study of the NIST, DieHard, and TestU01 tests revealed that each of them has its advantages and disadvantages in evaluating the statistical characteristics of generators. A detailed review of different test suites provides a better understanding of their strengths and limitations, which can help in choosing the right suite of tests for a particular task. The integrated use of these tests can provide a more accurate and complete assessment of the quality of generators. The obtained results will be a useful starting point for further research in this direction and in the development of reliable generators. The conclusions of the article may be useful for researchers, software developers, and other specialists who work with random and pseudo-random sequence generators.

**Keywords:** informational security; generators of random sequences; generators of pseudorandom sequences; statistical tests; NIST; DieHard; TestU01.

## REFERENCES (TRANSLATED AND TRANSLITERATED)

1. Almaraz Luengo, E., & Román Villaizán, J. (2023). Cryptographically Secured Pseudo-Random Number Generators: Analysis and Testing with NIST Statistical Test Suite. *Mathematics, 11(23),* 4812. https://doi.org/10.3390/math11234812
2. Mahalingam, H., Rethinam, S., Janakiraman, S., & Rengarajan, A. (2023). Non-Identical Inverter Rings as an Entropy Source: NIST-90B-Verified TRNG Architecture on FPGAs for IoT Device Integrity. *Mathematics, 11(4),* 1049. https://doi.org/10.3390/math11041049
3. Kim, Y., & Yeom, Y. (2021). Accelerated implementation for testing IID assumption of NIST SP 800-90B using GPU. *PeerJ Computer Science, 7,* e404. https://doi.org/10.7717/peerj-cs.404

4. Meitei, H. B., & Kumar, M. (2023). Implementation of a secure wireless communication system using true random number generator for internet of things. *Indonesian Journal of Electrical Engineering and Computer Science*, *30(2)*, 982. https://doi.org/10.11591/ijeecs.v30.i2.pp982-992
5. Al-Daraiseh, A., Sanjalawe, Y., Al-E'mari, S., Fraihat, S., Bany Taha, M., & Al-Muhammed, M. (2023). Cryptographic Grade Chaotic Random Number Generator Based on Tent-Map. *Journal of Sensor and Actuator Networks*, *12(5)*, 73. https://doi.org/10.3390/jsan12050073
6. Vaskova, A., Lopez-Ongil, C., Millan, E. S., Jimenez-Horas, A., & Entrena, L. (2011). Accelerating secure circuit design with hardware implementation of Diehard Battery of tests of randomness. *2011 IEEE 17th International On-Line Testing Symposium*, 179–181. https://doi.org/10.1109/IOLTS.2011.5993835
7. L'Ecuyer, P., & Simard, R. (2007). TestU01: A C library for empirical testing of random number generators. *ACM Transactions on Mathematical Software*, *33(4)*, 1–40. https://doi.org/10.1145/1268776.1268777
8. Suciu, A., Toma, R. A., & Marton, K. (2012). Parallel implementation of the TestU01 statistical test suite. *IEEE 8th International Conference on Intelligent Computer Communication and Processing*, 317–322. https://doi.org/10.1109/ICCP.2012.6356206
9. Almaraz Luengo, E., & Gragera, C. (2023). Critical Analysis of Beta Random Variable Generation Methods. *Mathematics*, *11(24)*, 4893. https://doi.org/10.3390/math11244893
10. Sriram, V., Srikamakshi, M., Jegadish Kumar, K. J., & Nagarajan, K. K. (2020). Randomness Analysis of YUGAM-128 Using Diehard Test Suite. *Innovative Data Communication Technologies and Application*, *46*, 600–607. https://doi.org/10.1007/978-3-030-38040-3_68
11. Araki, S., Wu, J.-H., & Yan, J.-J. (2024). A Novel Design of Random Number Generators Using Chaos-Based Extremum Coding. *IEEE Access*, *12*, 24039–24047. https://doi.org/10.1109/ACCESS.2024.3365638
12. Sleem, L., & Couturier, R. (2020). TestU01 and Practrand: Tools for a randomness evaluation for famous multimedia ciphers. *Multimedia Tools and Applications*, *79(33–34)*, 24075–24088. https://doi.org/10.1007/s11042-020-09108-w
13. Chen, Y., Tian, Y., Zhou, R., Castro, D. M., Guo, D., & Zhou, Q. (2024). NDSTRNG: Non-deterministic Sampling-based True Random Number Generator on SoC FPGA Systems. *IEEE Transactions on Computers*, 1–14. https://doi.org/10.1109/TC.2024.3365955
14. Lecca, C., Zegarra, A., & Santisteban, J. (2024). Random Number Generator Based on Hopfield Neural Network with Xorshift and Genetic Algorithms. *Advances in Computational Intelligence*, *14391*, 283–295. https://doi.org/10.1007/978-3-031-47765-2_21
15. Li, Y., Wang, Q., & Yu, S. (2024). A novel hybrid scheme for chaotic image encryption. *Physica Scripta*, *99(4)*, 045244. https://doi.org/10.1088/1402-4896/ad3171
16. Abdulhameed, H. A., Abdalmaaen, H. F., Mohammed, A. T., Mosleh, M. F., & Abdulhameed, A. A. (2022). A Lightweight Hybrid Cryptographic Algorithm for WSNs Tested by the Diehard Tests and the Raspberry Pi. *2022 International Conference on Computer Science and Software Engineering (CSASE)*, 271–276. https://doi.org/10.1109/CSASE51777.2022.9759589
17. Kumar, V., & Pravinkumar, P. (2023). Quantum random number generator on IBM QX. *Journal of Cryptographic Engineering*. https://doi.org/10.1007/s13389-023-00341-1
18. Aldossari, H., & Mascagni, M. (2022). Scrambling additive lagged-Fibonacci generators. *Monte Carlo Methods and Applications*, *28(3)*, 199–210. https://doi.org/10.1515/mcma-2022-2115
19. Hussein, S. N., & Al-Alak, S. M. (2021). Secret Keys Extraction Using Light Weight Schemes for Data Ciphering. *Journal of Physics: Conference Series*, *1999(1)*, 012114. https://doi.org/10.1088/1742-6596/1999/1/012114
20. Duda, C. K., Meier, K. A., & Newell, R. T. (2023). Development of a High Min-Entropy Quantum Random Number Generator Based on Amplified Spontaneous Emission. *Entropy*, *25(5)*, 731. https://doi.org/10.3390/e25050731
21. Zhao, W., & Ma, C. (2024). Modification of Intertwining Logistic Map and a Novel Pseudo Random Number Generator. *Symmetry*, *16(2)*, 169. https://doi.org/10.3390/sym16020169
22. Isakov, O. V., & Voitusik, S. S. (2023). Comparative analysis of digital noise generated by additive Fibonacci generators. *Ukrainian Journal of Information Technology*, *5(1)*, 67–76. https://doi.org/10.23939/ujit2023.01.067
23. Gafsi, M., Hafsa, A., & Machout, M. (2024). Hardware implementation of digital pseudo-random number generators for real-time applications. *Signal, Image and Video Processing*. https://doi.org/10.1007/s11760-024-03082-8

24. Ryan, C., Kshirsagar, M., Vaidya, G., Cunningham, A., & Sivaraman, R. (2022). Design of a cryptographically secure pseudo random number generator with grammatical evolution. *Scientific Reports, 12(1),* 8602. https://doi.org/10.1038/s41598-022-11613-x
25. Hussein, S. N., Obaid, A. H., & Jabbar, A. (2022). Encryption Symmetric secret Key in Wireless Sensor Network Using AES Algorithm. *Iraqi Journal of Science*, 5037–5045. https://doi.org/10.24996/ijs.2022.63.11.38
26. Semankiv, M. (2016). Assessment of statistical properties random numbers. *Bulletin of the National Technical University 'KhPI'. A Series of 'Information and Modeling'*, *0(21).* https://doi.org/10.20998/2411-0558.2016.21.12
27. Álvarez, R., Martínez, F., & Zamora, A. (2022). Improving the Statistical Qualities of Pseudo Random Number Generators. *Symmetry*, *14(2),* 269. https://doi.org/10.3390/sym14020269
28. Palacios-Luengas, L., Marcelín-Jiménez, R., Rodriguez-Colina, E., Pascoe-Chalke, M., Jiménez-Ramírez, O., & Vázquez-Medina, R. (2021). Function Composition from Sine Function and Skew Tent Map and Its Application to Pseudorandom Number Generators. *Applied Sciences*, *11(13),* 5769. https://doi.org/10.3390/app11135769
29. Hurley-Smith, D., & Hernandez-Castro, J. (2020). Quantum Leap and Crash: Searching and Finding Bias in Quantum Random Number Generators. *ACM Transactions on Privacy and Security*, *23(3),* 1–25. https://doi.org/10.1145/3398726