УДК 519.683

О. Н. Молотков, кандидат технічних наук, доцент,
доцент кафедри інформаційних систем
та технологій Академії митної служби України
Д. Є. Костенко, курсант Академії митної служби України

ЗАСТОСУВАННЯ ГЕНЕРАТОРА ПСЕВДОВИПАДКОВИХ ЧИСЕЛ MERSENNE TWISTER ДЛЯ ГЕНЕРАЦІЇ ПАРОЛІВ

Досліджуються характеристики перспективного генератора псевдовипадкових чисел для використання як засобу генерації паролів.

The characteristics of the perspective generator of random numbers are studied for use as means of generation of passwords.

Ключові слова. Генератор псевдовипадкових чисел, генерація паролів доступу, інформаційна безпека.

Вступ. Одне з найважливіших завдань, яке необхідно виконати митній службі України в найближчій перспективі, – це впровадження в митну справу міжнародних норм і правил, що дозволить перейти до застосування єдиних стандартів у галузі митної справи та приєднатися до міжнародних конвенцій з гармонізації і спрощення митних процедур [1, 2].

Виконання цих завдань неможливе без створення багатофункціональної Комплексної системи "Електронна митниця", яка поєднує інформаційно-телекомунікаційні технології та сукупність механізмів їх застосування, грунтується на вимогах Конституції України, законодавства України, міжнародних митних конвенціях та враховує досвід інших країн. Створення зазначеної системи дасть можливість підвищити якість митного регулювання та вдосконалити митне адміністрування [3].

Проектування і створення системи "Електронна митниця" передбачається здійснювати на базі Єдиної автоматизованої інформаційної системи Держмитслужби. Обов'язкові елементи цієї системи такі:

- комплексне впровадження засобів криптографічного та технічного захисту інформації, ідентифікації та автентифікації користувачів;
- захист інформації від несанкціонованого та неконтрольованого ознайомлення, модифікації, знищення, копіювання, поширення;
- обов'язкова автоматична реєстрація результатів ідентифікації та автентифікації користувачів, результатів виконання користувачем операцій з обробки інформації, спроб несанкціонованих дій з інформацією, фактів позбавлення користувачів права на доступ до інформації та її обробку, результатів перевірки цілісності засобів захисту інформації;
- створення комплексних систем захисту інформації на базі типових рішень та відомчого центру сертифікації ключів Держмитслужби.

У зв'язку із цим виникає гостра потреба у створенні якісного та надійного програмного забезпечення для реалізації вищезазначених завдань у митній службі.

Постановка завдання. Найважливішою складовою комплексної системи інформаційної безпеки є підсистема доступу й реєстрації. Функції, виконувані цією підсистемою, мають забезпечити не тільки розмежування користувачів відповідно до їх повноважень, але й установити особистість кожного користувача. Розмежування здійснюється за рахунок застосування механізмів автентифікації, а визначення особистості – механізмів ідентифікації.

Автентифікація й ідентифікація користувачів виконується після введення ними індивідуального пароля й логіна. Як правило, логін для входу в систему є "несекретним", а от відповідний йому пароль тримається в глибокій таємниці. У більш-менш великій організації видачею логінів і паролів займається системний адміністратор (адміністратор безпеки системи), у деяких випадках паролі задають собі самі користувачі. Очевидно, що порушник має бути нездатним підібрати пароль.

На сьогоднішній день найбільш надійним підходом до розв'язання задач автентифікації та ідентифікації є підхід, що базується на генерації паролів за допомогою генераторів псевдовипадкових чисел (ГПВЧ) [4–6].

Будь-яка програма, у тому числі програма генерації паролів, – це кінцевий автомат, вона детермінована й не може відтворити випадкову послідовність. Однак, із практичної точки зору для генерації паролів цілком достатньо детермінованої послідовності, дуже схожої на випадкову.

Якісний генератор псевдовипадкової послідовності (ПВП), орієнтований на використання в системах захисту інформації, має задовольняти такі вимоги:

- непередбачуваність;

• добрі статистичні властивості, ПВП за своїми статистичними властивостями не повинна істотно відрізнятися від істинно випадкової послідовності;

• великий період формованої послідовності: наприклад, при шифруванні для перетворення кожного елемента вхідної послідовності необхідно використовувати свій елемент псевдовипадкової гами;

• ефективна апаратна й програмна реалізація.

Нині розроблена та використовується значна кількість ГПВЧ. Проблема вибору найкращого, стосовно потреб ДМСУ, з найбільш поширених ГПВЧ (а саме: лінійний конгруентний; комбінований; адитивний; генератор, що тасує), розглядалась у працях [5, 6].

Мета цієї статті – доповнення результатів, отриманих у дослідженні [7], та дослідження характеристик перспективного ГПВЧ – Mersenne Twister. Цей генератор ПВП представлено в 1997 р. (автори Мацумото і Нішимура) й описано в [7]. Їхня праця грунтується на властивостях простих чисел Мерсенна і забезпечує швидку генерацію псевдовипадкових чисел. Під час розробки нового генератора автори врахували недоліки, які притаманні іншим генераторам ПВП: малий період; передбачуваність; статистична залежність, що легко прораховується.

Перевагами генератора є колосальний період (2^{19937}–1), який достатній для багатьох практичних завдань, рівномірний розподіл у 623 вимірах (лінійний конгруентний метод дає більш-менш рівномірний розподіл від сили в 5 вимірах), швидка генерація випадкових чисел (у 2–3 рази швидше, ніж стандартні генератори ПВЧ, що використовують лінійний конгруентний метод).

Існує два загальних варіанти алгоритму, запропонованих авторами, які розрізняються тільки розмірністю використовуваного простого числа Мерсенна. Найбільш поширений алгоритм MT19937 [7]. Реалізації цього алгоритму вже використовуються в стандартних бібліотеках для PHP, Python и Рубі.

У праці застосовано такий алгоритм ГПВЧ – Mersenne Twister (далі – алгоритм MT) [7].

Нехай значення $x = (x_{\omega+1}, x_{\omega+2}, \ldots, x_0)$ та $a = (a_{\omega+1}, a_{\omega+2}, \ldots, a_0)$ визначають вектори-слова, які є ω-вимірними рядами векторів двоелементної області $F_2 = \{0,1\}$. Ці значення ідентифіковано машинними словами розміру ω (з найменш суттєвим бітом у правій частині). Алгоритм MT генерує послідовність таких векторів-слів, що утворюють сукупність. Вони є однорідними псевдовипадковими числами між 0 та 2^{ω}–1. Діленням на 2^{ω}–1 можна визначити кожне таке вектор-слово, як реальне число у проміжку [0,1].

Введемо ще деякі позначення, окрім значень x та a. Це константи: цілі числа без знака n (ступінь лінійного повторення розмірності слова), r ($0 \le r \le \omega$–1), m ($1 \le m \le n$). Також на F_2 визначено постійну матрицю A (розмірністю $\omega \times \omega$), форма якої обирається так, щоб множення з її використанням виконувалось якомога швидко.

$$A = \begin{pmatrix} 1 & 1 & \ldots & 1 & 1 \\ 1 & 1 & \ldots & 1 & 1 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 1 & 1 & \ldots & 1 & \vdots \\ a_{\omega-1} & a_{\omega-2} & \cdots & \cdots & a_0 \end{pmatrix}.$$

Значення $x_0, x_1, \ldots x_{n-1}$ відомі, як початкові значення. Змінюючи значення k, починаючи з 0, генератор виробляє такі вектори $x_{n+1}, x_{n+2} \ldots$.

$$x_{k+n} := x_{k+m} \oplus \left(x_i^u \mid x_{i+1}^{ll}\right) A \qquad (k = 0,1,..),$$

де $\oplus$ – множення за модулем два. Алгоритм містить такі кроки.

Крок 0. Введення значень для постійних цілих беззнакових чисел u та ll для визначення розміру слова.

Заносимо в змінну u значення $\underbrace{1\cdots1}_{\omega-r}\underbrace{0\cdots0}_{r}$, що є бітовою маскою для верхніх ω–r бітів. У змінну ll заносимо значення $\underbrace{0\cdots0}_{\omega-r}\underbrace{1\cdots1}_{r}$, що є бітовою маскою для нижніх r бітів.

Заповнюємо матрицю A таким способом: останній рядок становитимуть значення $a_{\omega-1}a_{\omega-2}\cdots a_1a_0$, інші елементи ініціюються одиницями).

Крок 1. У лічильник циклу заносимо нульове значення, масив X заповнюємо деякими ненульовими ініціалізуючими значеннями ($x[0], x[1], \ldots, x[n-1]$).

Крок 2. Розраховуємо $x_i^u \mid x_{i+1}^{ll}$ через визначення рекурентних послідовностей $y = (x[i]\ AND\ u)\ OR\ (x[(i+1) \bmod n]\ AND\ ll)$.

Крок 3. $x[i] = x[(i+1) \bmod n]\ XOR\ (y >> 1)$.

Результат операції *XOR* визначається в такий спосіб:

$$XOR\begin{cases} 0 \text{ if the least significant bit of } y = 0 \\ a \text{ if the least significant bit of } y = 1. \end{cases}$$

Крок 4. Далі до кожного отриманого вектор-слова $x[i]$ вносимо x у чарунку y, проводимо зрушення y на u бітів та додаємо до y:

$$y = x[i]$$
$$y = y \; XOR \; (y >> u).$$

Аналогічно виконуємо таке:

$$y = y \; XOR \; ((y << s) \; AND \; b)$$
$$y = y \; XOR \; ((y << t) \; AND \; c)$$
$$y = y \; XOR \; (y >> l),$$

де l, s, t – це цілі значення, b, c – підходящі бітові маски розміру слова.

Виведення y

Крок 5. Змінюємо лічильник: $i \leftarrow (i+1) \bmod n$.

Крок 6. Перехід до кроку 2.

Слід зазначити, що хоча деякі криптостійкі генератори ПВП або потокові шифри й пропонують набагато більш "випадкові" числа, такі генератори набагато повільніші, ніж звичайні арифметичні й можуть бути непридатні у всякого роду дослідженнях, які потребують щоб процесор був вільний для більш корисних обчислень. У зв'язку із цим у праці аналізувалися властивості не тільки генератора Mersenne Twister, а і його комбінація з генератором, що тасує (MS генератор). Така комбінація дозволить повернути ті ж випадкові числа, що і в Mersenne Twister, але видані в іншому порядку, що забезпечить неможливість визначення параметрів внутрішнього генератора та дозволить позбутися небажаних кореляцій між серіями згенерованих чисел.

Для роботи генератора випадкових чисел необхідно число, або ряд чисел, що його ініціалізує (криптографи називають це джерелом ентропії), тому що за своєю внутрішньою будовою ГПВЧ – це ряд математичних функцій. Таким ініціалізуючим параметром може бути системний час або різні параметри устаткування. Багато ГПВЧ дотепер використовують традиційні методи збирання ентропії, такі як дії користувача (руху миші і т. ін.). Така схема використовується, наприклад, у програмі PGP, можливості якої описані в дослідженні [4]. До появи в процесорах можливості зчитувати значення найчутливішого до найменших змін навколишнього середовища лічильника тактів процесора, збирання ентропії було найбільш уразливим місцем ГПВЧ.

У цей час з'явилася можливість використання набагато швидших джерел ентропії, таких як шум звукової карти або значення лічильника тактів процесора (processor clock counter), які легко зчитуються, наприклад за допомогою інструкції rdtsc (*ReadTimeStamp Counter*) – асемблерної інструкції для платформи ×86, що читає лічильник TSC (Time Stamp Counter) і повертає в регістрах EDX:EAX 64-бітну кількість тактів з моменту останнього скидання процесора. rdtsc підтримується в процесорах Pentium і старше, а також у процесорах Intel.

Результати дослідження. Для перевірки якості ГПВЧ використовувався програмний модуль, розроблений у середовищі Delphi 7.0. Основними кількісними показниками для оцінки роботи генераторів ГПВЧ були результати, отримані за такими чотирма тестами: тест на однорідність, тест на пропуски, покер-тест, тест "збирання купонів" [5, 8]. Процедура тестування була обрана однаковою для всіх тестів. Вона включала такі дії.

Генерувалася репрезентативна кількість випадкових чисел з діапазону від 0 (включно) до 1 (виключно). Одержувані в результаті роботи генераторів значення розбивалися на кілька категорій, підраховувалася кількість значень у кожній категорії та імовірність влучення значення в кожну категорію, на підставі чого перевірялося проходження тесту за критерієм χ^2 з 5 %-ним рівнем значущості. Випадки непроходження тестів для кожного генератора при генерації ПВП наведено в табл. 1.

Під час проведення випробувань використано два варіанти отримання джерела ентропії для ініціалізації роботи ГПВП: системний час (у цьому випадку час запуску програми, що тестує), а також значення лічильника тактів процесора (processor clock counter).

Таблиця 1

Результати випробувань генераторів ПВП

	Тест на однорідність	Тест на пропуски (0.0–0.5)	Тест на пропуски (0.5–1.0)	Тест на пропуски (0.0–0.33)	Тест на пропуски (0.33–0.67)	Тест на пропуски (0.67–1.0)	Тест "Покер"	Тест "Збирання купонів"	Усього за кожним генератором
Системний генератор	95	105	87	90	92	76	54	103	702
Мінімальний стандартний	92	107	108	114	116	109	106	111	863

генератор									
Комбінований генератор [1]	93	86	93	92	92	86	94	99	735
Адитивний генератор	98	92	90	89	89	103	92	93	746
Генератор, що тасує [2]	96	90	100	90	88	90	95	93	742
Генератор MT	74	83	85	86	94	84	77	90	673
MS генератор, що тасує [3]	77	86	90	80	90	74	81	85	663
Усього за кожним тестом	625	649	653	641	661	622	599	674	

Зазначимо, що по-справжньому випадкового генератора серед обраних для дослідження не існує: є тільки менш однорідні й більш однорідні модулі генерації. Чим більша однорідність генератора, тим більші випадкові числа він генеруватиме, тобто буде краще.

Будь-який ГПВП з обмеженими ресурсами рано чи пізно зациклюється. Довжина циклів ГПВП залежить від самого генератора й у середньому становить близько $2^{(n/2)}$, де n – це розмір внутрішнього стану в бітах, хоча лінійні-конгруентні генератори мають максимальні цикли порядку 2^n. Якщо ГПВП може сходитися до занадто коротких циклів, такий ГПВП стає передбачуваним і є непридатним.

Більшість простих арифметичних генераторів хоча й має велику швидкість, але існує багато серйозних недоліків:

- занадто короткий період/періоди;
- послідовні значення не є незалежними;
- деякі біти “менш випадкові”, ніж інші;
- нерівномірний одномірний розподіл;
- оборотність.

Під час дослідження розглянуто декілька простих найпоширеніших генераторів випадкових чисел.

Як видно з таблиці, найгірші результати були отримані для мінімального стандартного генератора (863 негативних результати у сумі для випробування 4 тестами). Тобто цей генератор має бути забракований. Генератор Mersenne Twister пройшов випробування задовільно (673 негативних результати у сумі для випробування 4 тестами), на відміну від інших.

Крім того, просте застосування як початкової послідовності чисел, згенерованих за допомогою мінімальної стандартної для генератора, що тасує, дозволяє зменшити кількість негативних результатів проходження тестів на ≈ 10 %. Беручи до уваги, що генератор, який тасує, повертає точно ті ж випадкові числа, що й Mersenne Twister, дуже цікаво виявити, що під час перевірки його в тестовій програмі регулярність не проявляється. Це свідчить про те, що його можна використовувати для деяких програмних додатків, особливо тих, які потребують пари випадкових чисел. Однак як майбутні дослідження було б доцільно перевірити регулярність не тільки у двовимірній площині, а і в k-вимірному просторі за допомогою спеціальних тестів.

Стосовно впливу початкових значень на кількість достовірних тестів, можна зазначити, що обране початкове значення істотно впливає на випадковість згенерованої послідовності: заміна джерела ентропії для ініціалізації роботи ГПВП із системного часу (час запуску програми, що тестує) на значення лічильника тактів процесора поліпшує результати на ≈ 7 %.

Висновки. На основі проведених випробувань можна зробити загальний висновок стосовно вибору якісного генератора паролів:

– по-перше, показано, що найбільш придатним з тих, які тестуються, є генератор, що являє собою комбінацію генератора Mersenne Twister та генератора, що тасує (це означає, що підхід покращання, тобто подальшої рандомізації вже отриманої послідовності чисел, правильний);

– по-друге, найменш придатним виявився мінімальний стандартний генератор;

– по-третє, при використанні генератора, що тасує, у сукупності з Mersenne Twister, слід пам’ятати про необхідність достатніх ресурсів для збереження та обробки досить великих масивів чисел підвищеної точності, що, порівняно з одним значенням початкового числа типу longint для мінімального стандартного генератора, може здатися занадто величезним обсягом даних.

Це свідчить про те, що прості генератори, які не є криптостійкими, можуть бути використані як складні блоки при побудові інших надійних генераторів.

Отримані результати випробувань генераторів псевдовипадкових послідовностей можуть бути використані працівниками підрозділів Держмитслужби України, які безпосередньо братимуть участь у розробці та впровадженні комплексних засобів криптографічного та технічного захисту інформації, ідентифікації та автентифікації користувачів.

Література

1. Международная Конвенция об упрощении и гармонизации таможенных процедур (Киотская Конвенция) [Текст]. – 1973.

2. Пашко П. В. Електронна митниця – головний механізм забезпечення митної безпеки держави [Текст] / П. В. Пашко // Матеріали 8-ї Міжнародної науково-практичної конференції “Ринок послуг комплексних транспортних систем і проблеми логістики”.

3. Розпорядження КМУ від 17 вересня 2008 р. № 1236-р “Про схвалення Концепції створення багатофункціональної комплексної системи “Електронна митниця” [Текст].

4. Дюбко В. П. Криптоаналіз програмного забезпечення, що реалізує криптографічні алгоритми [Текст] / В. П. Дюбко, В. В. Поліщук, Д. Костенко // Вісник Академії митної служби України. – 2006. – № 4.

5. Кнут Д. Искусство программирования. Получисленные методы [Текст] / Д. Кнут. – 3-е изд. – М. : Вильямс, 2007. – Т. 2. – 832 с.

6. Молотков О. Н. Дослідження генераторів псевдовипадкових чисел як засобу генерації паролів [Текст] / О. Н. Молотков, О. Карнін // Вісник Академії митної служби України. – 2006. – № 4.

7. Makoto Matsumoto and Takuji Nishimura Mersenne Twister: A 623-dimensionally equidistributed uniform pseudorandom number generator // ACM Transactions on Modeling and Computer Simulations: Special Issue on Uniform Random Number Generation. – 1998.

8. Бакнелл Дж. Фундаментальные алгоритмы и структуры данных в Delphi [Текст] : пер. с англ. – СПб. : ДиаСофтЮП, 2003. – 560 с.

---

[1] Базується на мультиплікативних лінійних конгруентних генераторах з різними довжинами циклів.

[2] Створений на основі ПВП, яка була отримана за допомогою мінімального стандартного генератора.

[3] Створений на основі ПВП, яка була отримана за допомогою генератора MT.