Лабораторна робота №2

«Реалізація алгоритмів генерації ключів гібридних криптосистем»

Чашницька М., Ткаченко А., Ковалевський О.

6 грудня 2021 р.

Мета роботи: «Дослідження алгоритмів генерації псевдовипадкових послідовностей, тестування простоти чисел та генерації простих чисел з точки зору їх ефективності за часом та можливості використання для генерації ключів асиметричних криптосистем»

Хід роботи

Проаналізувати стійкість реалізації генератору ПВЧ та генераторів ключів для бібліотеки PyCrypto на платформі Linux.

Теоретична частина

PyCrypto

PyCryptodome — це автономний Python пакет низькорівневих криптографічних примітивів. Він підтримує Python 2.7, Python 3.5 і більш нові версії, а також PyPy. Для швидшої роботи алгоритмів з відкритим ключем у системах Unix вам слід встановити GMP у вашій системі. PyCryptodome — це форк PyCrypto. PyCryptodome не є обгорткою окремої бібліотеки C, як OpenSSL. Більшість алгоритмів реалізовано на чистому Python. Тільки ті частини, для яких надзвичайно важливою є їхня швидкодія (наприклад, блочні шифри), реалізуються як розширення C.

Генератори псевдовипадкових чисел бібліотеки PyCrypto беруть випадкові числа із генератору ОС. Бібліотека PyCrypto має такі методи генерації псевдовипадкових чисел:

1. Crypto.Random.get_random_bytes(k) повертає k псевдовипадкових байтів.
2. Crypto.Random.random.getrandbits(k) повертає псевдовипадкове ціле число, що має бітову довжину k.
3. Crypto.Random.random.randrange(start, stop, step) повертає псевдовипадкове ціле число x таке, що $start < x < stop$.
4. Crypto.Random.random.choice(seq) повертає випадковий елемент послідовності *seq*.
5. Crypto.Random.random.shuffle(seq) випадковим чином перемішує послідовність *seq*.
6. Crypto.Random.random.sample(population, k) обирає k різних випадкових елементів із *population*.

Практична частина

Тестування генераторів псевдовипадкових чисел відбувалося за допомогою тестів *NIST*. Для тестування генерувалися блоки довжиною $1024, 10000, 100000, 640000, 1000000$ бітів за допомогою методу Crypto.Random.random.getrandbits, бо з нашої точки зору він є головним методом генерації псевдовипадкових бітів. Для генерації псевдовипадкових послідовностей було написано наступний Python скрипт:

Лістинг 1: Алгоритм генерації псевдовипадкових послідовностей

```
from Crypto.Random.random import getrandbits
from Crypto.Random.random import randint

s = bin(getrandbits(1024))[2:]

with open('bits.txt', 'w') as f:
        f.write(s)

print('OK')
```

Для перевірки була використана Python реалізація тестів псевдовипадкових послідовностей NIST. Внаслідок тестування отримано результати, наведені у Додатку А.

Висновки

Внаслідок аналізу отриманих результатів зроблено висновок, що генератор псевдовипадкових чисел бібліотеки PyCrypto не є гарним за більшістю критеріїв NIST. На зображеннях, що наведені у Додатку А, можна побачити, що зі збільшенням довжини послідовності псевдовипадкових бітів кількість провалених тестів збільшується.

В результаті виконання практикуму надано рекомендацію використовувати бібліотеку PyCrypto зі сторонніми генераторами псевдовипадкових чисел.

Додаток А

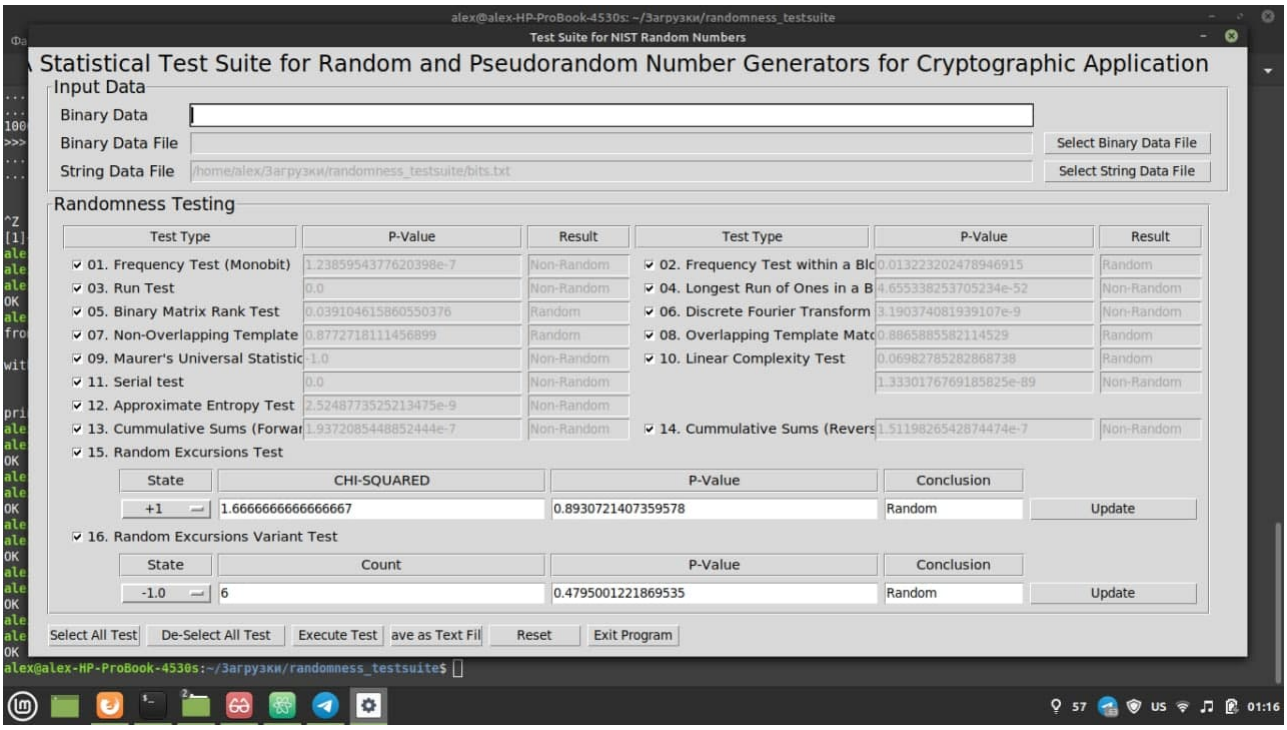

Рис. 1: Результати для 1024 бітів

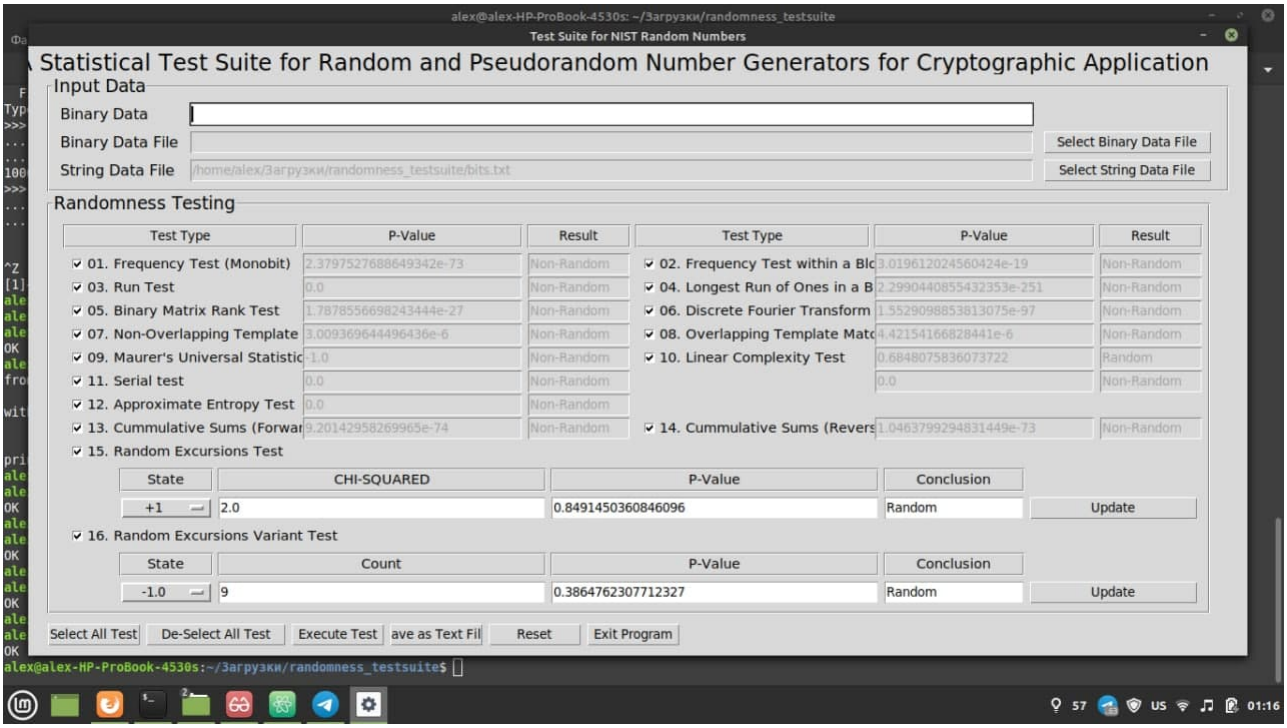

Рис. 2: Результати для 10000 бітів

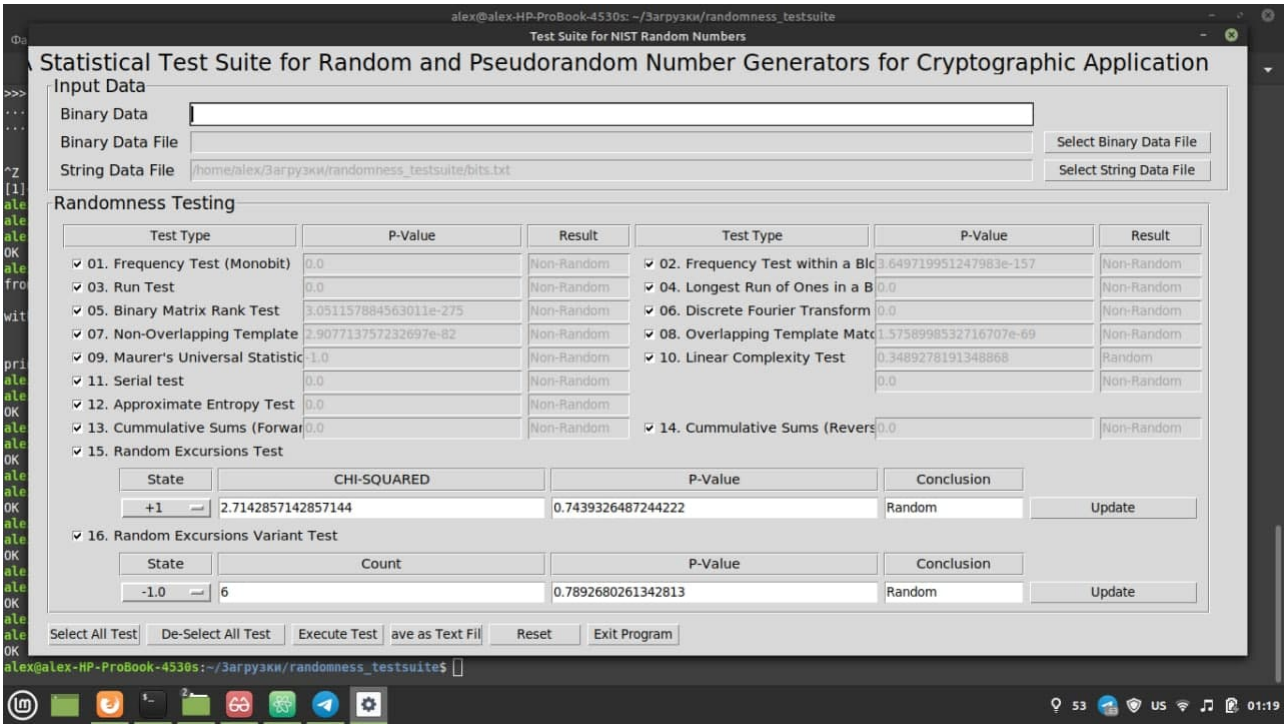

Рис. 3: Результати для 100000 бітів

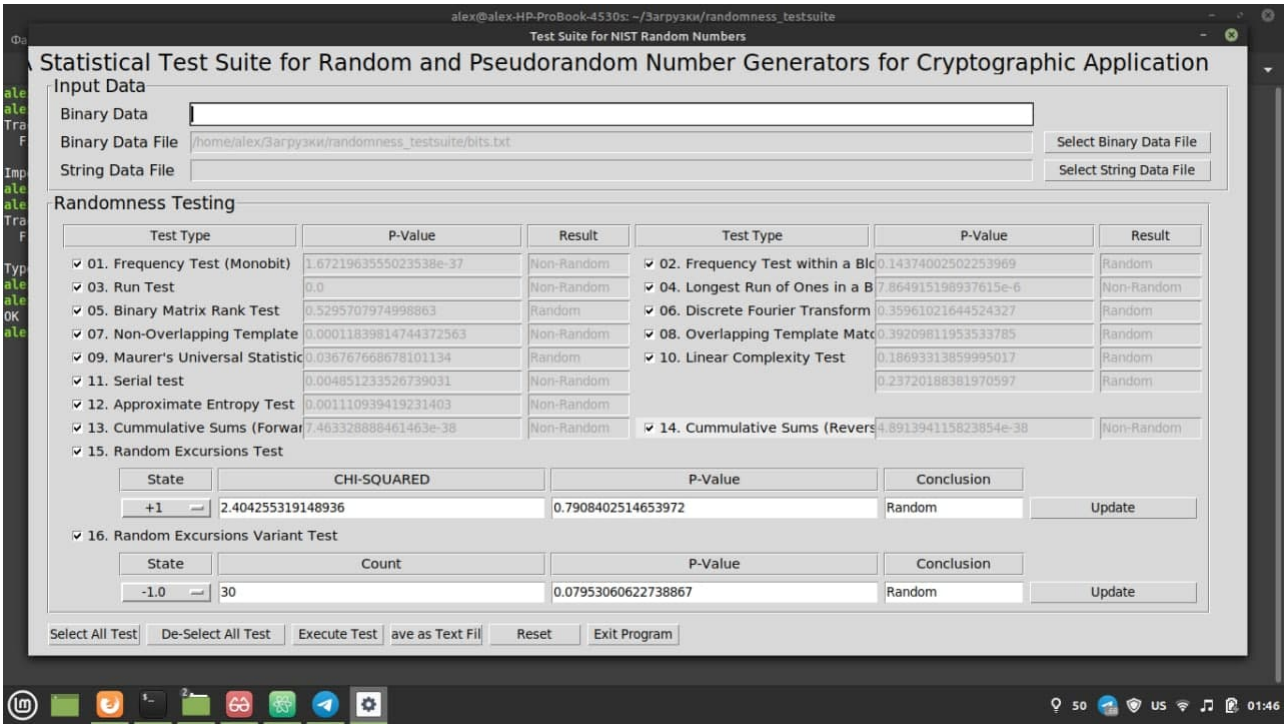

Рис. 4: Результати для 640000 бітів

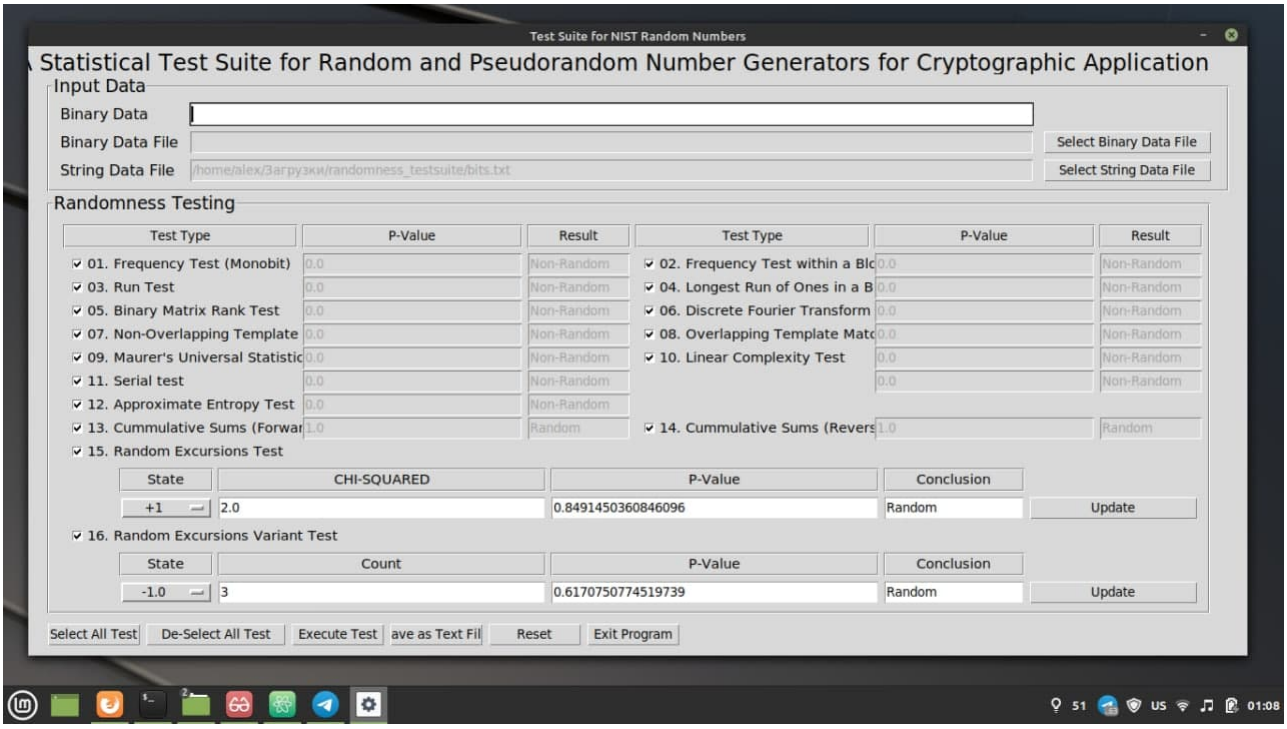

Рис. 5: Результати для 1000000 бітів