МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний університет «Чернігівська політехніка»

Навчально-науковий інститут електроних та інформаційних технологій

Кафедра кібербезпеки та математичного моделювання

**ЗВІТ**

про виконання лабораторної роботи

з дисципліни «Основи криптографічного захисту інформації»

Виконав: здобувач 3 курсу, групи КБ-221 Регент Анастасія

Перевірив: Шелест Михайло Євгенович

НУ «Чернігівська політехніка» 2024

Лабораторна робота № 2

**Мета:** отриматидосвід в оцінюванні випадкових послідовностей, обчисленні ентропії, відновлювані параметри некриптографічних ГПВЧ.

Хід роботи

Завдання № 1

**Оцінка ентропії**

У цьому завданні вчимося обчислювати ентропію тексту.

Необхідно обчислити вручну значення ентропії наданого тексту.

Крім того, напишіть програму на Python, яка аналізує вхідний файл та обчислює ентропію його вмісту.

**Припустимо, текстовий файл містить наступний текст:**

AAAAABBBBBCCCCCDDDDEEEEE

**Використайте такі кроки:**

1. Розрахуйте частоти символів у файлі.

2. Обчисліть ентропію за формулою Шеннона: H = , де — частота символу i.

**Формат виконання:**

1. Надайте значення ентропії, обчислену вручну.

2. Завантажте програму для обчислення ентропії у ваш репозиторій.

3. Додайте файл із розрахунками ентропії для наданого тексту.

**Дайте відповідь:**

1. Який рівень ентропії у вхідному файлі?

Рівень ентропії для наданого тексту AAAAABBBBBCCCCCDDDDEEEEE становить 2.317 біт.

2. Що це говорить про випадковість або надлишковість тексту?

Випадковість – ентропія 2.317 біт вказує, що текст має структурованість, а не є абсолютно випадковим (є закономірності у розподілі символів).

Надлишковість – низька ентропія свідчить про надлишковість у тексті, що дозволяє зменшити його розмір за допомогою стиснення.

A–5, B–5, C–5, D–4, E–5.

Загальна кількість символів: 24

|  |  |
| --- | --- |
| **Ймовірність** | |
|  | (1) |
|  | (2) |
|  | (3) |
|  | (4) |
|  | (5) |
| **Формула Шеннона** |  |
|  | (6) |
| **Обчислення ентропії** |  |
|  | (7) |
|  | (8) |
|  | (9) |
|  | (10) |
|  | (11) |
|  | (12) |
|  | (13) |
|  | (14) |

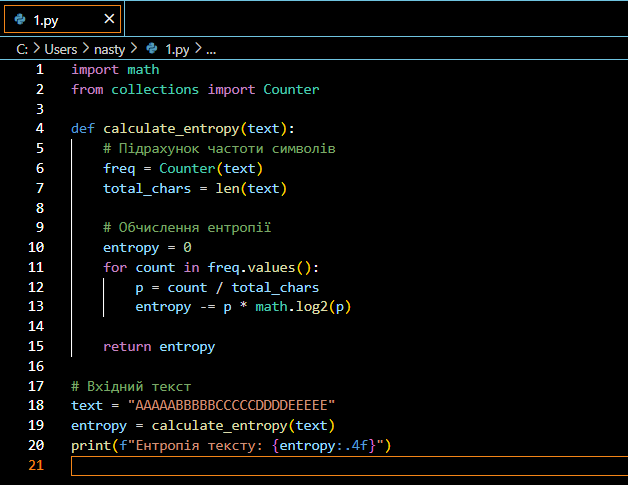


Рисунок 1 – Код

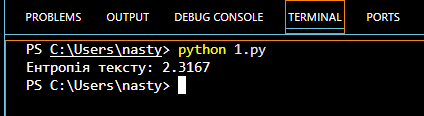


Рисунок 2 – Результат

Завдання № 2

**Тестування послідовностей на статистичну випадковість**

У цьому завданні досліджуємо статистичні властивості псевдовипадкових послідовностей, згенерованих різним чином.

У мові програмування Python є 2 різних модулі для генерування псевдовипадкових чисел: **random** та **secrets**. Модуль **random** використовує для генерування алгоритм Mersenne Twister, що не є криптографічно стійким. Модуль **secrets** користується системними засобами генерування криптографічно стійких псевдовипадкових чисел (/dev/urandom на Linux та macOS).

**Ваше завдання**:

1. Згенерувати та записати у файл псевдовипадкові послідовності довжиною щонайменше 1ГБ, використовуючи наступні ГПВЧ:

1. random.Random.randint()

2. secrets.randbits()

3. OpenSSL

2. У файлах варто зберігати дані в бінарному виді.

3. З OpenSSL можна згенерувати послідовність просто з командного рядка:

openssl rand -out rnd-openssl.bin 1000000000

Послідовність згенерована OpenSSL буде вашою еталонною послідовністю.

4. Протестувати згенеровані послідовності статистичним тестом [dieharder](https://webhome.phy.duke.edu/~rgb/General/dieharder.php).

5. Пакет dieharder доступний в репозиторіях Ubuntu. Створіть віртуальну машину або контейнер з Ubuntu будь-яким зручним для вас способом.

6. Запустіть пакет тестів dieharder для кожної послідовності та збережіть звіт в окремий файл. Наприклад для послідовності згенерованої OpenSSL:

dieharder -a -g 201 -f rnd-openssl.txt

7. Використовуйте ті самі параметри dieharder й для інших послідовностей.

**Формат виконання**

Надайте звіти тестування для кожної з послідовностей. Дайте відповіді на наступні запитання:

1. Скільки з тестів провалила кожна з послідовностей?

2.Чи можна за результатами статистичних тестів визначити генератор псевдовипадкових чисел що є криптографічно стійким від звичайного?

Так, можна.

Додаткова інформація: утиліти що дозволяють встановити віртуальну машину або контейнер з Ubuntu:

1. [VirtualBox](https://www.virtualbox.org/wiki/Downloads) (Linux, Windows, macOS)

2. [Podman](https://podman.io/) (Linux, Windows, macOS)

3. [UTM](https://docs.getutm.app/installation/ios/) (macOS)

Якщо є складності з Ubuntu, можно скористатися працювати з Pyhton у середовищі Windows, наприклад, з Colabs Notebooks (буде значно довше, але усе вийде).

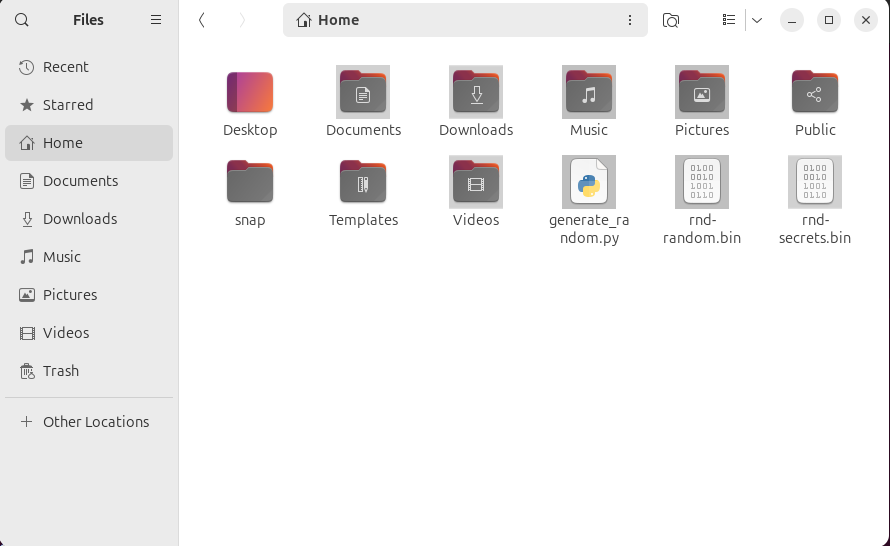


Рисунок 3 – Папки



Рисунок 4 – Код 2.1



Рисунок 5 – Код 2.2

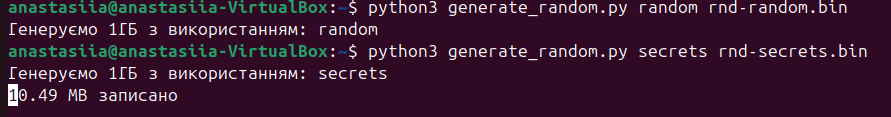


Рисунок 6 – Процес

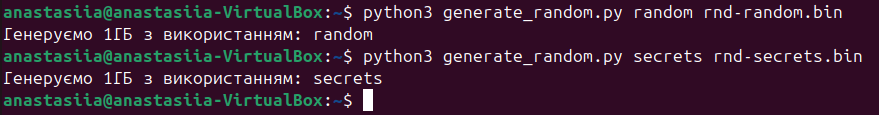


Рисунок 7 – Результат

Завдання № 3

**Некриптографічний ГПВЧ**

У цьому завданні досліджуємо вразливості генератора псевдовипадкових чисел Mersenne Twister (MT19937) при використанні його для генерації криптографічних ключів.

Це завдання підкреслює важливість вибору відповідних криптографічних інструментів та методів для забезпечення надійного захисту інформації. Ви отримаєте практичний досвід у виявленні та експлуатації вразливостей, що виникають при неправильному використанні генераторів псевдовипадкових чисел та режимів шифрування.

Вам надано ряд файлів. Файл **data.jpg.enc**, який був зашифрований використовуючи функцію encrypt з файлу **encryption.py**. Ви не маєте ключа, що використовувався для зашифрування, проте маєте послідовність декількох псевдовипадкових чисел (точніше 624), згенерованих тим же генератором за якийсь час до того як ним було згенеровано ключ. Ці випадкові числа збережено у файлі **sequence.txt**.

Спробуйте віднайти ключ шифрування та розшифрувати файл **data.jpg.enc**.

**Додаткова інформація:**

1. Почніть з аналізу функцій шифрування в **encryption.py**. Чи надійні вони?

2. Зверніть увагу на використаний генератор псевдовипадкових чисел.

3. Можете користуватися функціями як зашифрування, так і розшифрування з **encryption.py**.

**4. MT19937Reverse** стане в нагоді. Дослідить як він працює.

5. Зашифрований файл **data.jpg.enc** є зображенням у форматі jpg. За вдалого розшифрування ви зможете продивитись його.

6. Зображення формату jpg мають ідентифікаційний заголовок 0xFF 0xD8 в hex форматі, або ж  ‘a╪’ в ASCII.

7. Зберегти розшифровані дані можна використовуючи наступний код:

with open(filename + ".jpg", "wb") as f:

f.write(data)

Для глибшого розуміння теми рекомендується ознайомитися з додатковими матеріалами, такими як відео "Random Facts About Mersenne Twisters" від Тома Лістона (<https://youtu.be/Jo5Nlbqd-Vg>), де детально розглядаються особливості та вразливості генератора Mersenne Twister.

**Формат виконання:**

1. Збережіть знайдений ключ у форматі hex в текстовий файл.

2.Додайте у створений вами репозиторій з домашнім завданням наступні файли:

1. Текстовий файл зі знайденим ключем шифрування у форматі HEX.

2. Розшифроване зображення jpg.

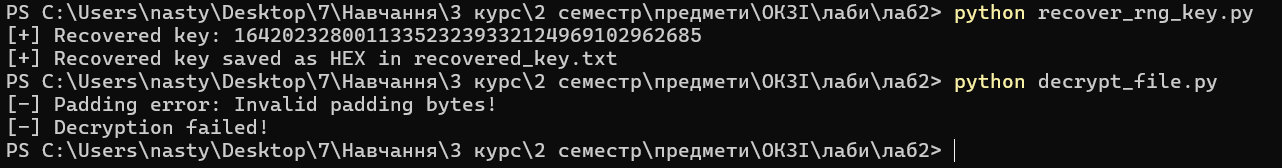


Рисунок 8 – Результат

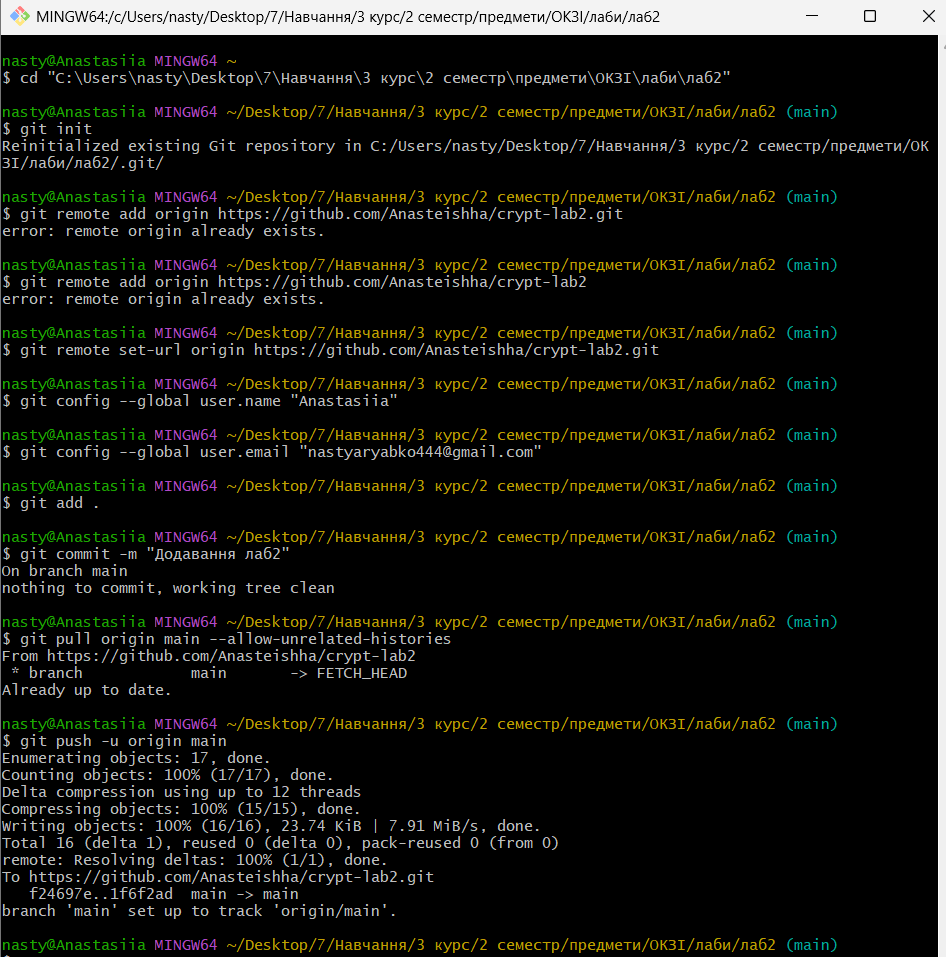


Рисунок 9 – Git

**Посилання на репозиторій:** <https://github.com/Anasteishha/crypt-lab2>

Контрольні запитання

1. Яка різниця між випадковими та псевдовипадковими числами?

Різниця між випадковими та псевдовипадковими числами полягає в тому що випадкові числа генеруються природними процесами, а псевдовипадкові числа генеруються алгоритмами, які надають результати, які здаються випадковими, але насправді є детермінованими.

2.Чому важливо використовувати криптографічно стійкі генератори псевдовипадкових чисел (CSPRNG)?  
CSPRNG гарантують високий рівень непередбачуваності та стійкість до зловживань, що критично для криптографії, оскільки слабкий генератор може бути передбачений і зламаний.

3.Які властивості повинні мати криптографічно стійкі генератори псевдовипадкових чисел?

Непередбачуваність, стійкість до відновлення вихідних значень та стійкість до атак, де відомі деякі попередні значення.

4. Наведіть приклади задач, де необхідно використовувати випадкові або псевдовипадкові числа.

Шифрування, генерація ключів, тестування безпеки та імітаційне моделювання.

5. Що таке "ентропія" у контексті випадкових чисел і чому вона важлива?  
Ентропія вимірює ступінь непередбачуваності або випадковості системи. Висока ентропія забезпечує кращу випадковість, що важливо для криптографії та генерації випадкових чисел.

6. Що означає поняття "кількісне вимірювання інформації"?

Вимірювання кількості інформації в одиницях, таких як біти, що допомагає оцінити, скільки нового знання чи різноманіття містить джерело.

7. Що таке ентропія за Шенноном і як її обчислити?

Ентропія Шеннона вимірює інформацію, яка міститься в повідомленні, на противагу до тієї частини повідомлення, яка є визначеною (або передбачуваною).

|  |  |
| --- | --- |
|  | (6.2) |

8.Як змінюється кількість інформації, коли підвищується ймовірність події?

Кількість інформації зменшується, коли ймовірність події зростає, оскільки вона стає більш передбачуваною.

9. У чому відмінність між шифруванням та кодуванням?

Шифрування — це процес перетворення даних у вигляд, що не може бути прочитаний без ключа, а кодування — це перетворення даних у формат для збереження чи передавання, але без приховування змісту.

10. Опишіть основні моделі атак у криптографії.

1. Атака з обраним відкритим текстом (Chosen Plaintext Attack, CPA);

2. Атака з обраним шифртекстом (Chosen Ciphertext Attack, CCA);

3. Атака за допомогою знань про внутрішню структуру алгоритму.

11. Чим відрізняється "атака на основі шифртексту" від "атаки з обраним відкритим текстом"?

Атака на основі шифртексту – це коли атакуючий має доступ до шифртексту і намагається отримати ключ чи розшифрувати дані, а ось атака з обраним відкритим текстом, коли атакуючий може обрати відкриті тексти для шифрування і спостерігати за їх шифртекстами.

12. Які властивості має криптосистема, щоб бути стійкою до атак?

1. Безпека повинна базуватися на складних математичних проблемах;

2. Шифрування та ключі повинні бути достатньо сильними;

3. Потрібен захист від всіх типів відомих атак.

13. Поясніть поняття "пряма секретність" та її значення.

Пряма секретність – це властивість криптосистеми, при якій неможливо отримати відкритий текст без знання ключа, навіть якщо атакуючий має доступ до великої кількості шифрованих даних.

14. Як визначити, чи є алгоритм криптографічно стійким?  
 Алгоритм вважається криптографічно стійким, якщо його безпека не може бути порушена при допомозі сучасних обчислювальних потужностей, і він не піддається відомим математичним атакам.