Министерство образования Республики Беларусь

Учреждение образования
БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерных систем и сетей

Кафедра информатики

Дисциплина: Информационные сети. Основы безопасности

ОТЧЁТ
к лабораторной работе №7
на тему

ЗАЩИТА ПО ОТ НЕСАНКЦИОНИРОВАННОГО ИСПОЛЬЗОВАНИЯ

Выполнил: студент гр. 253503
Тимошевич К.С.

Проверил: ассистент кафедры информатики Герчик А.В.

Минск 2025

СОДЕРЖАНИЕ

1 ПОСТАНОВКА ЗАДАЧИ

Целью данной лабораторной работы является исследование и практическое применение методов обфускации исходного кода на языке *Python* для защиты программного обеспечения от несанкционированного использования, анализа и модификации [1].

Актуальность работы обусловлена необходимостью обеспечения безопасности программных продуктов, предотвращения обратной инженерии и защиты интеллектуальной собственности в условиях возрастающих киберугроз. В рамках работы предполагается изучить и реализовать методы обфускации, такие как переименование идентификаторов, добавление избыточного кода и шифрование строковых литералов. Переименование идентификаторов заключается в замене имен переменных, функций и других элементов на случайные строки, что затрудняет понимание логики программы. Добавление избыточного кода предполагает внедрение в программу лишнего бессмысленного кода, который не влияет на функциональность, но усложняет анализ [2].

Шифрование строковых литералов включает преобразование строковых констант в зашифрованный вид с использованием алгоритмов кодирования, таких как *Base64*, что затрудняет их прямое чтение. Объектом исследования выступает исходный код программы на языке *Python*, а предметом исследования – методы и алгоритмы обфускации, применяемые для защиты кода от анализа и несанкционированного использования.

Практическая часть работы включает разработку и реализацию функций для обфускации кода, применение этих методов к примеру исходного кода, анализ измененного кода на предмет читаемости и сложности анализа, а также проверку сохранения функциональности программы после обфускации.

2 ХОД ВЫПОЛНЕНИЯ РАБОТЫ

Программа выполняет обфускацию исходного кода на языке *Python*, применяя три основных метода: переименование идентификаторов, добавление избыточного кода и шифрование строковых литералов. В начале программа принимает на вход исходный код в виде строки. Первым этапом выполняется функция *rename_identifiers*, которая анализирует код, находит все идентификаторы. Каждый идентификатор заменяется на случайно сгенерированную строку длиной 8 символов. Далее выполняется функция *add_junk*, которая добавляет в код избыточные. Эти конструкции не влияют на функциональность программы, но усложняют ее анализ. Третий этап – функция *encrypt_strings*, которая находит все строковые литералы в коде, шифрует их с использованием алгоритма *Base64* и заменяет оригинальные строки на вызовы функции *base64.b64decode* для их расшифровки. После выполнения всех этапов обфускации программа возвращает модифицированный код, который сохраняет исходную функциональность, но становится значительно сложнее для чтения и анализа (рисунок 2.1).

```
def UNTHSgZB(TboSROxq, MEKGWQqJ):
    WSQeXGNx = TboSROxq + MEKGWQqJ
    print(base64.b64decode("UmVzdWx0Og==").decode(), WSQeXGNx)
    return WSQeXGNx
    for _ in range(10): pass


    x = [i**2 for i in range(100) if i % 3 == 0]

UNTHSgZB(5, 3)
```

Рисунок 2.1 – Вывод обфусцированного кода

На рисунке 2.2 показано выполнение сначала обфусцированного кода, а затем оригинального. Оба варианта выводят одинаковый результат, что подтверждает сохранение функциональности программы после обфускации.

```
User@KARINICH MINGW64 /d/6_SEM/ИСОБ/LR7/LR7_Timoshevich (main)
$ py test_obfuscated_code.py
Result: 8

User@KARINICH MINGW64 /d/6_SEM/ИСОБ/LR7/LR7_Timoshevich (main)
$ py test_normal_function.py
Result: 8
```

Рисунок 2.2 – Результат выполнения программ

Таким образом, программа успешно демонстрирует применение методов обфускации для защиты кода от несанкционированного использования и анализа.

ЗАКЛЮЧЕНИЕ

В ходе выполнения лабораторной работы был разработан инструмент для обфускации кода на *Python*, который успешно применяет три основных метода: переименование идентификаторов, добавление избыточного кода и шифрование строк. Программа продемонстрировала свою работоспособность на примере функции *calculate*, которая складывает два числа и выводит результат. Обфусцированный код сохранил свою функциональность, но стал значительно сложнее для чтения и анализа.

Метод переименования идентификаторов заменил имена переменных и функций на случайные строки, что затруднило понимание логики программы. Добавление избыточного кода, такого как циклы и вывод случайных значений, создало информационный шум, который усложнил анализ. Шифрование строк с использованием *Base64* сделало невозможным прямое извлечение текстовых данных из кода.

Однако обфускация не обеспечивает полной защиты от целенаправленного анализа, так как опытный злоумышленник может восстановить логику программы. Также стоит отметить, что добавление избыточного кода и шифрование строк могут незначительно увеличить время выполнения программы и объем кода.

Несмотря на эти ограничения, разработанный инструмент может быть полезен для базовой защиты кода от несанкционированного использования и анализа, особенно в небольших проектах.

Таким образом, проведенная работа подтвердила, что обфускация является эффективным методом начального уровня для защиты программного обеспечения.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

[1] Обфускация кода — что, как и зачем [Электронный ресурс]. – Режим доступа: https://habr.com/ru/articles/735812/. – Дата доступа: 11.03.2025

[2] Обфускация: что нужно знать [Электронный ресурс]. – Режим доступа: https://otus.ru/journal/obfuskaciya-chto-nuzhno-znat-razrabotchiku/. – Дата доступа: 11.03.2025