**Задача 3: Проверка лицензионного ключа**

**Исходный код с пояснениями**

python

Copy

Download

import os *# Модуль для работы с файловой системой*

def check\_license():

*# Проверяем наличие файла лицензии*

if not os.path.exists("license.key"):

print("Файл лицензии не найден!")

return False *# Возвращаем False, если файл отсутствует*

*# Читаем и проверяем ключ*

with open("license.key") as f:

key = f.read().strip() *# Удаляем лишние пробелы и переносы*

return key == "ABCD-1234-EFGH" *# Сравниваем с эталонным ключом*

def main():

*# Проверяем лицензию*

if not check\_license():

print("Программа заблокирована!")

return *# Завершаем работу, если лицензия недействительна*

*# Основная логика программы*

print("Программа запущена!")

*# Здесь может быть основной код приложения*

if \_\_name\_\_ == "\_\_main\_\_":

main() *# Запуск программы*

**Тестирование функционала**

| **Сценарий** | **Результат** | **Описание** |
| --- | --- | --- |
| Файл license.key отсутствует | Файл лицензии не найден! Программа заблокирована! | Лицензия не обнаружена, доступ запрещён |
| Файл есть, ключ верный (ABCD-1234-EFGH) | Программа запущена! | Лицензия подтверждена |
| Файл есть, ключ неверный (XXXX-0000-YYYY) | Программа заблокирована! | Неверный лицензионный ключ |

**Задача 4: Вопросы по защите программного обеспечения**

**1. Основные цели защиты ПО**

* Усложнение реверс-инжиниринга
* Борьба с нелегальным распространением
* Защита интеллектуальной собственности

**2. Важность защиты для бизнеса**

* Сохранение уникальных технологий
* Предотвращение утечек конфиденциальной информации
* Обеспечение соблюдения лицензионных соглашений

**3. Риски при отсутствии защиты**

* Утечка алгоритмов и данных
* Внедрение вредоносных модификаций
* Финансовые потери из-за пиратского использования

**4. Реверс-инжиниринг и связанные угрозы**

* **Что это?** Анализ исполняемого кода для восстановления исходной логики.
* **Угрозы:** Кража технологий, создание уязвимостей.

**5. Методы реверс-инжиниринга**

* **Статический анализ** (Ghidra, IDA Pro)
* **Динамический анализ** (отладчики: x64dbg, OllyDbg)

**6. Уязвимости в незащищённом ПО**

* Жёстко прописанные пароли (password = "12345")
* Отсутствие проверки лицензий
* Уязвимости переполнения буфера

**7. Популярные методы защиты**

* **Обфускация** (PyArmor, Obfuscator-LLVM)
* **Шифрование** критичных строк и алгоритмов
* **Лицензирование** (HWID, онлайн-активация)

**8. Шифрование данных**

* **Симметричное** (AES, один ключ)
* **Асимметричное** (RSA, пара ключей)

**9. Обфускация кода**

* Делает код трудночитаемым для анализа.
* **Инструменты:** PyArmor (Python), ProGuard (Java).

**10. Лицензирование как метод защиты**

* **Оффлайн-лицензии** (файл license.key)
* **Онлайн-проверка** (серверная валидация)
* **Привязка к оборудованию** (HWID)

**11. Тестирование защиты**

* Декомпиляция (uncompyle6, Ghidra)
* Фаззинг-тесты (проверка на некорректные данные)

**12. Аудит безопасности**

* Поиск уязвимостей перед релизом.
* **Инструменты:** SonarQube, Checkmarx.

**13. Автоматизированное тестирование защиты**

* Интеграция в CI/CD-пайплайны.

**14. Успешные кейсы защиты**

* **Denuvo** (защита игр от взлома)
* **Spotify** (онлайн-лицензирование)

**15. Адаптация защиты под тип ПО**

* **Мобильные приложения:** Защита от модификаций (Frida, Xposed)
* **Десктоп:** Техники против отладки

**16. Примеры взломов**

* **Cyberpunk 2077** (уязвимые скрипты)
* **Утечка исходников Windows XP** (2004 г.)

**17. Машинное обучение в защите ПО**

* Обнаружение аномалий в работе программы.

**18. Юридические аспекты**

* **DMCA (США)** – запрет на обход защиты.
* **GDPR (ЕС)** – защита пользовательских данных.

**19. Новые тенденции**

* **RASP** (самозащита приложений в runtime)
* **Контейнеризация** (изоляция компонентов)

**20. Влияние IoT и AI**

* **Уязвимости IoT:** Слабое шифрование.
* **Риски AI:** Генерация поддельных лицензий нейросетями.

**Вывод**

Базовая проверка лицензии уязвима к редактированию файлов. Для надёжности необходим комплексный подход:

* **Обфускация**
* **Шифрование**
* **Серверная проверка**  
  Регулярный аудит безопасности критически важен для предотвращения угроз.