Пара №13

Задача 1: *Исходный скрипт*

pip install pyarmor

*pyarmor gen -O dist auth.py*

*Задача 2:*

*pip install pyinstaller # Установка PyInstaller*

pyinstaller --onefile --key=MySecretKey auth.py # Создание запакованного исполняемого файла

pip install uncompyle6 # Проверка защиты

uncompyle6 dist/auth # декомпилирование исполняемого файла

Пара №14

Задача 3: *Простая проверка лицензии*

1) Импортирует модуль os, который предоставляет функции для взаимодействия с операционной системой, такие как проверка существования файлов.

3)def check\_license():: Определяет функцию check\_license, которая будет проверять наличие и корректность файла лицензии.

4)def check\_license():: Определяет функцию check\_license, которая будет проверять наличие и корректность файла лицензии.

5) Если файл не найден, выводит сообщение "Файл лицензии не найден!".

6) Возвращает False, указывая на то, что проверка лицензии не удалась.

8) Открывает файл "license.key" для чтения.

9) Читает содержимое файла, удаляя лишние пробелы и символы новой строки.

10) Сравнивает прочитанный ключ с ожидаемым значением "ABCD-1234-EFGH" и возвращает результат сравнения.

12) Определяет функцию main, которая является точкой входа в программу.

13) Вызывает функцию check\_license и проверяет её результат.

14) Если проверка лицензии не удалась, выводит сообщение "Программа заблокирована!"

15) Возвращает управление, завершая выполнение функции main.

17) Если проверка лицензии прошла успешно, выводит сообщение "Программа запущена!".

20) Проверяет, является ли данный скрипт основной программой (а не импортированным модулем).

21) Если условие истинно, вызывает функцию main, запуская основной код программы.

Пустые строки в коде 2, 7, 11, 16, 18 и 22, используются для улучшения читаемости кода.

1. Программа проверит наличие файла "license.key".

Поскольку файла нет, она выведет сообщение "Файл лицензии не найден!".

Затем программа выведет "Программа заблокирована!" и завершит выполнение.

2.Программа найдет файл "license.key" и прочитает его содержимое.

Поскольку ключ в файле совпадает с ожидаемым значением "ABCD-1234-EFGH", функция check\_license() вернет True.

Программа выведет "Программа запущена!" и продолжит выполнение основного кода.

3. Программа найдет файл "license.key" и прочитает его содержимое.

Поскольку ключ в файле не совпадает с ожидаемым значением, функция check\_license() вернет False.

Программа выведет "Программа заблокирована!" и завершит выполнение.

Задача 4:

1 )Основные цели включают предотвращение несанкционированного доступа, защиты интеллектуальной собственности, обеспечение целостности и конфиденциальности данных, а также предотвращение модификации и копирования кода.

2) Защита программного обеспечения важна для бизнеса, так как она помогает предотвратить финансовые потери, защитить репутацию компании, обеспечить соответствие нормативным требованиям и сохранить конкурентное преимущество.

3) Риски включают утечку конфиденциальных данных, финансовые потери, юридические последствия, потерю доверия клиентов и возможность использования уязвимостей злоумышленниками.

4) Реверс-инжиниринг — это процесс анализа программы с целью понять её работу, часто с использованием декомпиляции и дизассемблирования. Он угрожает безопасности, так как позволяет злоумышленникам находить уязвимости и копировать код.

5) Основные методы включают статический анализ (анализ кода без его выполнения), динамический анализ (анализ во время выполнения), декомпиляцию и дизассемблирование.

6) Уязвимости могут включать буферные переполнения, инъекции кода, утечки памяти, недостаточную проверку входных данных и слабые механизмы аутентификации.

7) Наиболее распространенные методы включают шифрование, обфускацию кода, использование цифровых подписей, лицензирование и регулярные аудиты безопасности.

8) Шифрование данных преобразует информацию в нечитаемый формат с использованием ключей. Основные виды включают симметричное шифрование (один ключ для шифрования и дешифрования) и асимметричное шифрование (публичный и приватный ключи).

9) Обфускация кода — это процесс преобразования исходного кода в трудночитаемый вид без изменения его функциональности. Это помогает защитить программы, усложняя процесс реверс-инжиниринга.

10) Инструменты для обфускации кода включают ProGuard для Java, Obfuscator-LLVM для C/C++, JavaScript Obfuscator для JavaScript и другие специализированные инструменты для различных языков программирования. |

11) Лицензирование может использоваться для контроля доступа к программному обеспечению, ограничения количества установок, проверки подлинности пользователей и обеспечения соответствия условиям использования. |

12) Наиболее эффективные типы лицензий включают лицензии на основе аппаратного ключа, сетевые лицензии, лицензии на основе подписки и лицензии с активацией через интернет. |

13) Тестирование защитных механизмов включает использование статического и динамического анализа кода, пентестинг, фаззинг и аудит безопасности. |

14) Регулярные аудиты безопасности помогают выявлять и устранять уязвимости, обеспечивать соответствие нормативным требованиям, улучшать механизмы защиты и предотвращать потенциальные атаки.

15) Автоматизированное тестирование позволяет быстро и эффективно выявлять уязвимости, проверять соответствие стандартам безопасности и оценивать эффективность защитных механизмов.

16) Примеры успешных практик включают использование многофакторной аутентификации, регулярное обновление программного обеспечения, применение современных методов шифрования и обфускации, а также проведение регулярных аудитов безопасности.

17) Методы защиты могут адаптироваться в зависимости от типа программного обеспечения, учитывая его специфику, например, использование различных методов шифрования для мобильных и десктопных приложений, или применение специализированных инструментов обфускации для веб-приложений.

18) Известные случаи включают атаки на уязвимости в операционных системах, таких как WannaCry, атаки на веб-приложения, такие как взлом Equifax, и эксплуатация уязвимостей в мобильных приложениях.

19) Современные технологии, такие как машинное обучение, могут улучшить защиту программ, автоматизируя обнаружение уязвимостей, предсказывая потенциальные атаки и адаптируя механизмы защиты в реальном времени.

20) Юридические аспекты включают соблюдение законов о защите данных, лицензионные соглашения, патенты и авторские права, а также соответствие международным стандартам безопасности.

21) Новые тренды включают использование искусственного интеллекта для обнаружения угроз, развитие технологий блокчейн для защиты данных, увеличение использования облачных технологий для безопасности и развитие методов защиты на основе поведенческого анализа.

22) Развитие технологий требует постоянного обновления методов защиты, таких как улучшение алгоритмов обфускации, использование более сложных методов шифрования и внедрение новых технологий обнаружения вторжений.

23) С развитием IoT и AI могут появиться новые уязвимости, связанные с недостаточной защитой устройств IoT, утечкой данных в облачных сервисах, атаками на алгоритмы машинного обучения и эксплуатацией уязвимостей в системах искусственного интеллекта.