СУЧАСНІ ІНФОРМАЦІЙНІ СИСТЕМИ І ТЕХНОЛОГІЇ

Матеріали

VII Всеукраїнської

науково-практичної інтернет-конференції

студентів, аспірантів та молодих вчених

за тематикою:

«Сучасні комп`ютерні системи та мережі в управлінні»

29 листопада 2024 р.

Хмельницький

Міністерство освіти і науки України
Херсонський національний технічний університет
Вінницький національний технічний університет
Криворізький національний університет
Кременчуцький національний університет ім. М. Остроградського
Хмельницький національний університет
Львівський національний університет природокористування

Матеріали

VII Всеукраїнської

науково-практичної інтернет-конференції

молодих вчених та студентів

«Сучасні інформаційні системи та технології»

за тематикою:

«Сучасні комп'ютерні системи та мережі в управлінні»

29 листопада 2024 року

Хмельницький

УДК 330.111.66:005.8
С 91

С 91 **Сучасні інформаційні системи та технології**: матеріали VII Всеукр. наук.-практ. інтернет-конф. за тематикою «Сучасні комп'ютерні системи та мережі в управлінні» (29 листопада 2024 р., м. Херсон, м. Хмельницький) / за ред. А. А. Григорової. – Херсон: Книжкове видавництво ФОП Вишемирський В. С., 2024. – 263 с.

ISBN 978-617-8187-36-1 (електронне видання)

Доповіді наукової конференції містять результати наступних досліджень: сучасні тенденції розвитку інформаційних технологій; впровадження інновацій та сучасних технологій; моделювання та оптимізація систем управління; інформаційні технології в науці, освіті, економіці, логістиці, туристичній сфері, транспорті; новітні технології в енергетичних системах та в галузі енергозбереження.

Роботи друкуються в авторській редакції, в збірці максимально зменшено втручання в обсяг та структуру відібраних до друку матеріалів. Редакційна колегія не несе відповідальність за достовірність статистичної та іншої інформації, що надано в рукописах, та залишає за собою право не розподіляти поглядів деяких авторів на ті чи інші питання.

Збірник становить інтерес для студентів, аспірантів, викладачів та наукових працівників.

УДК 330.111.66:005.8

ISBN 978–617–8187–36–1 (електронне видання)

3. Алаєв С. В., Бойченко О. О. “Забезпечення безпеки передачі даних у сучасних інформаційних системах”. Київ: Національний технічний університет України, 2022.

УДК 004.421.2

Малярова Д.М.
здобувач вищої освіти факультету КІУ

Маляров М.В.
к.т.н., доцент, доцент кафедри автоматичних систем безпеки та інформаційних технологій

ПОРІВНЯЛЬНИЙ АНАЛІЗ РЕАЛІЗАЦІЙ ЛЕСИЧНОЇ ОБФУСКАЦІЇ, ЯК ЗАХИСТУ ВІД СТАТИЧНОГО ДОСЛІДЖЕННЯ

Харківський національний університет радіоелектроніки, Україна
Національний університет цивільного захисту України, Україна

Постановка проблеми

Інформаційні технології стали важливою складовою сучасного суспільства, надаючи широкий спектр можливостей для обробки, зберігання та передачі інформації. Розвиток інформаційних технологій значно підвищив доступність інформації, але водночас відкрив нові вразливості для несанкціонованого доступу та копіювання програмного забезпечення, що може завдати шкоди як розробникам, так і кінцевим користувачам. У контексті збереження комерційних та інтелектуальних інтересів розробників виникає необхідність забезпечення захисту програмного забезпечення.

Наразі існують статичні засоби дослідження, які використовуються для аналізу роботи програмного забезпечення. Ці засоби оперують початковим кодом програми та, після опрацювання, будують її алгоритм. Використання статичних засобів є доволі універсальними в тому значенні, що теоретично можуть одержати алгоритм усієї програми, у тому числі і тих блоків, які ніколи не отримують управління [1].

Захист програмного забезпечення від статичного дослідження спрямований на ускладнення або навіть унеможливлення декомпіляції та запобігання несанкціонованим змінам у коді. Серед наведених методів захисту особливої уваги заслуговує обфускація – ефективний спосіб ускладнення аналізу програмного забезпечення.

Аналіз останніх досліджень та публікацій

На цей час основні методи захисту програмного забезпечення, передбачають ускладнення доступу до коду та його структури. Статичні засоби дослідження та лексична обфускація наведені в роботах Баришева Ю. В, Дмитришина О. В, Дудатьєва А. В., Каплун В. А та Семеренко В. П.

На відміну від шифрування, яке приховує код повністю, лексична обфускація спрямована на зміну структури коду таким чином, щоб він залишався функціональним, але водночас значно ускладнювався. Алгоритми обфускації передбачають використання хаотичних переходів в різні частини коду, впровадження помилкових процедур-«пустишок», холості цикли, спотворення кількості реальних параметрів процедур програмного забезпечення, розкидання ділянок коду по різних областях оперативного запам’ятовуючого пристрою [2].

Наразі в інтернеті існує багато різний сайтів для проведення обфускації, серед яких можна відзначити Pythonium [3], Toolfk [4] та Oxyry Python Obfuscator [5]. Використання їх алгоритмів дозволяє обфускувати код та використовувати їх функціонал для захисту своїх програм. Але в той же час їх притаманні недоліки, до яких можна віднести сповільнення виконання коду, порушення його працездатності або обфусукацію тільки частини програмного коду тощо.

Постановка задачі

Дана робота спрямована на розроблення програмної реалізації лексичної обфускації програмного коду (розробленого на мові Python), з максимальним дотриманням принципів основних складових обфускації та перевіркою працездатності програми та проведенню порівняльного аналізу з наведеними раніше трьома онлайн ресурсам.

Виклад основного матеріалу

Можна виділити мінімум два аспекти, які дозволяють визначити, що робота обфускатора була виконана недаремно [1]:

– час, витрачений на розуміння коду зловмисником перевищує час, протягом якого актуальність алгоритму залишається значущою;

– вартість деобфускації перевищує вартість самого продукту.

Крім того, існує кілька різновидів обфускації, які спрямовані на досягнення різних цілей у процесі захисту коду [1]:

– лексична обфускація – перетворення форматування, які змінюють лише зовнішній вигляд програми;

– обфускація даних – перетворення структури даних, що змінюють спосіб організації та представлення даних, якими оперує програма;

– обфускація графа потоку керування – заміна виконуваної логіки недетермінованою та додавання випадкового зайвого заплутаного коду.

Лексична обфускація є ефективним засобом, який застосовується для маскування структури коду на рівні ідентифікаторів, що ускладнює розуміння та дослідження програми під час статичного аналізу. Вона включає в себе наступні складові [1, 6]:

1 видалення необов’язкових конструкцій мови програмування (видалення усіх коментарів в коді або зміна їх на дезінформуючи або видалення різноманітних пробілів, відступів, які зазвичай використовуються для кращого візуального сприйняття коду);

2 додавання різноманітних (так званих, сміттєвих) операцій;

3 зміна розміщення блоків (функцій, процедур) програми так, щоб це ні в якому разі не вплинуло на її дієздатність;

4 заміна імен ідентифікаторів (змінних, масивів, структур, функцій і т. д.) на набори символів, які важко сприймати людині.

При розробці програмної реалізації було передбачено кілька важливих етапів обфускації коду та реалізовані наступні функції:

1 запит назви файлу та читання коду, який треба обфускувати;

3 видалення коментарів та зайвих пробілів;

4 додавання надлишкового та «мертвого» коду;

5 констант на вирази;

6 застосування символьної обфускації ідентифікаторів;

7 додавання дезінформуючих коментарів;

8 запис обфускованого коду у новий файл.

Сформулювавши ідею програми, обравши середовище розробки та спроектувавши структуру програми за допомогою PyCharm Community Edition 2022.3.3 була написана, протестована і відлагоджена програмна реалізація лексичної обфускації даних, що реалізує введення назви файлу, з кодом, який треба обфускувати, проходження усіх кроків обфускації та збереження результату у новому файлі. Тестування програми дозволило отримати практичне підтвердження деяким теоретичним положенням лексичної обфускації.

Наприклад, було підтверджено успішну обфускацію, зниження читабельності та збереження функціональності, продемонстровано додавання зайвих коментарів і «випадкових» рядків та додавання зайвих циклів і умов.

Порівняємо результати обфускації коду розробленого програмного засобу з наявними в інтернеті [3, 4, 5], на прикладі тестової програми, що написана на мові програмування Python. Опишемо кожну реалізацію обфускації.

1 Код, обфускований за допомогою розробленої програми:

Функції: Видалення коментарів та зайвих пробілів, додавання надлишкового та мертвого коду, перетворення константних значень на вирази, використання символьної обфускації, додавання дезінформуючих коментарів.

Особливості: використання зайвих коментарів та перетворення константних значень на вирази (1 перетворилося на (1 + 10 - 10)).

Недоліки: Виконання коду може трохи сповільнюватися через надлишковий код, що може бути відчутно у великих проектах.

2 Код, обфускований за допомогою Pythonium:

Функції: Видалення коментарів та зайвих пробілів, використання символьної обфускації.

Особливості: використання односимвольних змінних після символьної обфускації.

Недоліки: Код не працює, оскільки не витримані відступи для Python при перевизначенні вбудованих функцій або об'єктів, що призводить до синтаксичних помилок. Крім того, заміна значень на односимвольні змінні працює не зовсім коректно, особливо коли в коді вже використовуються односимвольні змінні, що порушує логіку програми. При виведенні тексту залишаються не обфусковані змінні, наприклад, print(f"Число {number} просте?", A(I)), де number не замінено.

3 Код, обфускований за допомогою Toolfk:

Функції: Видалення коментарів та зайвих пробілів, використання символьної обфускації.

Особливості: використання довгих змінних після символьної обфускації.

Недоліки: Код не працює, оскільки при виведенні тексту залишаються не обфусковані змінні. Наприклад, у рядку print(f"Факторіал {n} дорівнює", xSNyxzDAEfGOKcxuOllqSiivicOFOMoa(MmvlnZFDztjXWwuraDgBwplEYAFjpSnI)), змінна n не обфускована, що призводить до помилок.

4 Код, обфускований за допомогою Oxyry Python Obfuscator:

Функції: Видалення коментарів та зайвих пробілів, використання символьної обфускації.

Особливості: використання довгих специфічних змінних після символьної обфускації (складаються з символу «O» та «0»).

Недоліки: Додає коментарі з номерами початкових рядків, що у деяких випадках може полегшити відстеження вихідного коду.

Висновки

Запропонована програмна реалізація забезпечує високий рівень захисту за рахунок символьної обфускації та додаткових оманливих коментарів, а головне, зберігає функціональність коду. Було виявлено деякі недоліки сторонніх інструментів – недостатня перевірка функціональності після обфускації, що робить їхній результат нестабільним або непридатним до використання (код, обфускований за допомогою Pythonium та Toolfk не працював після обфускації). Наприкінці зазначимо, що розроблена програмна реалізація лексичної обфускації працює коректно та дійсно може ускладнити статичний аналіз та забезпечити захист програмного забезпечення розробника від дослідження.

Перелік джерел посилання:

1. Каплун В. А., Дмитришин О. В., Баришев Ю. В. Захист програмного забезпечення. Частина 2: навчальний посібник. Вінниця: ВНТУ, 2014. 105 с.

2. Дудатьєв А. В., Каплун В. А., Семеренко В. П. Захист програмного забезпечення. Частина 1: навчальний посібник. Вінниця: ВНТУ, 2005. 140 с.

3. Python obfuscator online. Pythonium – Python, What else ? URL: https://pythonium.net/obfuscator (дата звернення: 02.11.2024).

4. Python obfuscator & encryption online tools. Toolfk. URL: https://www.toolfk.com/tools/online-python-confuse.html (дата звернення: 02.11.2024).

5. Oxyry Python Obfuscator – The most reliable python obfuscator in the world. Oxyry Python Obfuscator – the power to protect your python source code. URL: https://pyob.oxyry.com (дата звернення: 02.11.2024).

6. Каплун В. А., Дмитришин О. В., Баришев Ю. В. Захист програмного забезпечення: лабораторний практикум. Вінниця: ВНТУ, 2017. 75 с.