УДК 004.056

К.т.н., доцент Марченко О.І., студент Гнатейко В.В.

Національний технічний університет України
«Київський політехнічний інститут»

СПОСІБ РОЗБИТТЯ ВИКОНУВАНОГО ФАЙЛУ НА ФРАГМЕНТИ З МЕТОЮ ЗАХИСТУ ВІД НЕСАНКЦІОНОВАНОГО КОПІЮВАННЯ

Abstract

O.I. Marchenko, associate professor, PhD; V.V. Gnateiko, student
Method to split a binary file into smaller chunks in order to protect against unauthorized copying

This work deals with researching of methods of protecting against unauthorized copying of files and proposing a new technique based on splitting a binary file into smaller chunks. Proposed technique is based on server-client technology, where code is hosted on a server but running on a client. This paper also presents techniques for chunks modification with the goal of security increasing, and some directions for further work are noted as well.

Вступ

Значення та роль програмного забезпечення (ПЗ) в житті сучасного суспільства зростає з кожним роком, тому задача захисту ПЗ є досить актуальною на сьогоднішній день. Найчастіше виникає необхідність захисту ПЗ від нелегального використання, однак останнім часом все частіше також з’являється необхідність захисту від несанкціонованого копіювання (НК).

На сьогодні існують декілька способів, які дозволяють захистити ПЗ від НК, але всі вони мають свої недоліки і не можуть бути використані в загальному випадку. В даній статі пропонується спосіб, який зможе розширити коло випадків захисту ПЗ від НК та покращити швидкодію в деяких з них.

Постановка задачі

Основна мета роботи полягає в дослідженні способів і алгоритмів захисту ПЗ від НК, а також у розробці нового способу на базі виконаного дослідження, який буде більш ефективним за певними показниками.

Термінологія

Обфускація – процес перетворення програми на еквівалентну програму, яка на всіх допустимих вхідних даних видає той самий результат, що й оригінальна програма, але є більш важкою для аналізу, розуміння і модифікації [1].

Аналіз покриття програми – визначення частоти виконання кожної інструкції програмного коду [2] за допомогою виконання програмного коду з різними вхідними даними.

Дизасемблер – комп'ютерна програма, що відновлює програму на мові асемблера з бінарного машинного коду.

Степінь зв'язності програмного коду – показник наскільки зв'язані між собою окремі фрагменти коду.

Хмарні обчислення – це модель забезпечення зручного доступу на вимогу через мережу з довільного комп'ютера до спільного обчислювального ресурсу [3].

Недосяжний код – код, що не буде виконаний ніколи.

Загальна ідея

Усі методи захисту ПЗ поділяються на апаратні та програмні. Перші передбачають застосування додаткового обладнання, а другі – попередньої обробки програмного забезпечення перед тим як воно буде захищено.

Запропонований метод належить до групи програмних методів захисту. Джерелом його ефективності є відсутність попередньої ручної обробки ПЗ, а також потреби модифікації коду цього ПЗ на етапі компіляції. Захист програмного коду від НК досягається завдяки зберіганню ПЗ на сервері, а необхідність попередньої обробки виключається завдяки передаванню програмного коду на клієнт і виконанню його на клієнті. Оскільки єдина можливість НК при такому підході може бути тільки на клієнті, то залишається вирішити задачу захисту програмного коду саме на клієнті. Для рішення цієї задачі програмний код з сервера відсилається фрагментами, а клієнт, після виконання чергового фрагменту, робить запит серверу на пересилання наступного фрагмента. Для коректного розбиття файлу на фрагменти з метою реалізації його захисту потрібне виконання попереднього аналізу на сервері. Оскільки виконуваний файл передається по фрагментах, що завантажуються в міру необхідності, то для копіювання такої фрагментованої програми треба зберегти усі фрагменти на клієнті, а потім об'єднати їх. Для ускладнення НК фрагментованої програми, пропонується виконувати деякі модифікації цих фрагментів коду для того, щоб унеможливити побітове порівняння, а

також знаходження потрібного фрагменту і місця з'єднання фрагментів та їх розташування в програмного коді. Крім того, з метою підвищення швидкості виконання захищеного файлу, потрібно виконати аналіз коду для визначення доцільного розміру фрагмента.

Попередній аналіз виконуваного файлу

Для коректного розбиття файлу на фрагменти попередньо треба виконати аналіз покриття програми та оцінити ступінь зв'язності виконуваного файлу.

Аналіз покриття програми дозволяє знайти місця в програмному коді, які не бажано розривати, а також видалити недосяжний код, якщо цього не зробив компілятор. Аналіз покриття можна зробити дизасемблюванням виконуваного файлу та додаванням лічильників на кожну інструкцію, або на групу інструкцій. Для другого варіанту повинні бути враховані розгалуження та можливі переходи у цьому файлі [2]. Оцінювання ступеня зв'язності виконуваного файлу є більш складним, бо необхідно аналізувати взаємозв'язки його фрагментів, які можуть знаходитися не підряд. В такому випадку треба додавати лічильник не на кожну інструкцію, а на кожний перехід. Таким чином можна буде знайти найбільш зв'язані фрагменти, які по можливості не варто розривати.

Обидва зазначені види аналізу виконуються багато разів на різних наборах тестових даних. В залежності від пріоритету того чи іншого вхідного набору даних, можна зробити розбиття файлу більш коректним та ефективним з точки зору збільшення швидкості його виконання у фрагментованому вигляді.

Модифікація фрагментів виконуваного файлу

Для підвищення захисту пропонується застосувати наступні способи модифікації фрагментів файлу: 1) обфускація фрагменту; 2) динамічний розмір фрагменту; 3) альтернативне розбиття файлу на фрагменти.

Обфускація є досить надійним способом захисту програми, який полягає у заміні одного фрагменту коду на еквівалентний за функціональністю, але суттєво менш зрозумілий фрагмент. На даний момент розроблено досить багато алгоритмів обфускації коду, але оскільки у нашому випадку цей процес потрібно проводити в реальному часі, то складність алгоритмів обфускації може істотно вплинути на швидкодію програмного забезпечення, яке потрібно захистити [1].

Спосіб динамічного розміру фрагменту полягає у додаванні до кінця фрагменту недосяжного коду. Оскільки перехід між будь-якими двома

фрагментами буде робитися через стрибок у пам’яті, то додавання недосяжного коду після команди переходу не змінить ходу виконання фрагменту. З іншого боку додавання недосяжного коду призведе до збільшення розміру фрагменту та часу його передавання з сервера на клієнт.

Спосіб альтернативного розбиття полягає у випадковому з’єднанні двох послідовних фрагментів, або розбитті одного фрагменту на два, що також ускладнить повне копіювання виконуваного файлу.

Всі ці способи незалежні і можуть бути виконані як окремо, так і одночасно над одним фрагментом.

Висновки

Запропонований спосіб є найбільш подібним до способів, що використовуються у моделі хмарного обчислення, оскільки всі вони використовують сервер для зберігання програмного забезпечення [3]. Перевагою запропонованого методу є можливість роботи з обладнанням клієнта. Напрямками подальшого поліпшення даного способу можуть бути наступні:

1) оновлення статистики про покриття та оцінювання зв’язності фрагментів у реальному часі;
2) прогнозування роботи клієнта та відсилання наступного фрагменту клієнту ще до того, як він зробить запит;
3) зберігання частини коду програмного забезпечення на клієнті у випадку, коли ця частина не є пріоритетною для захисту, та коли це суттєво може підвищити швидкодію.

Література

1. *Чернов А.В.* Анализ запутывающих преобразований программ. Сб. "Труды Института системного программирования", под. ред. В. П. Иванникова. М.: ИСП РАН, 2002.

2. *H. Zhu, P. Hall, J. May.* Software Unit Test Coverage and Adequacy. ACM Computing Surveys, 29(4):366-427, December 1997.

3. *Donald Kossmann, Tim Kraska, Simon Loesing.* An evaluation of alternative architectures for transaction processing in the cloud. Proceedings of the 2010 International Conference on Management of Data (SIGMOD 2010), June 6-11, 2010, Indianapolis, Indiana, USA, pp. 579-590