УДК 004.056.55

О. Ю. Нагирняк
В. А. Каплун

ЗАСІБ ГЕНЕРУВАННЯ КЛЮЧОВОЇ ІНФОРМАЦІЇ ДЛЯ ЗАХИСТУ ПРОГРАМ ВІД НЕСАНКЦІОНОВАНОГО КОПІЮВАННЯ

Вінницький національний технічний університет

Анотація: *У статті йдеться про один зі способів захисту програмного забезпечення від несанкціонованого копіювання (НСК). Для реалізації цього способу захисту використано ключ, генерування якого базується на унікальних параметрах конкретного комп'ютера, а для унеможливлювання його копіювання зловмисниками ключ зберігається у хешованому вигляді.*

Ключові слова: *захист від несанкціонованого копіювання, ключова інформація, генерування ключового файлу.*

Abstract *This article discusses one way to protect software from unauthorized copying (UC). To implement this method of protection, a key is used, the generation of which is based on the unique parameters of a specific computer. and to prevent it from being copied by attackers, the key is stored in a hashed form.*

Keywords*: protection against unauthorized copying, key information, key file generation.*

Захист програмного забезпечення в наш час не втрачає своєї актуальності. Не зважаючи на те, захист від несанкціонованого копіювання не є достатньо стійким способом захисту програм, для певного кола задач він все ж може бути успішно застосований [1]. Інша справа, що даний вид захисту необхідно осучаснити і зробити його зручним для використання. Саме для цього і пропонується деяке удосконалення популярного методу захисту програм шляхом прив'язки до характеристик комп'ютера, на якому цей програмний засіб буде експлуатуватися.

Засіб для захисту програм від НСК пропонується реалізувати за допомогою вбудовування у програму модуля прив'язки до ключового файлу. Для створення ключового файлу вирішено обрати серійні номери усіх логічних дисків комп'ютера або\і параметри інших ресурсів комп'ютерної системи. Дана інформація є унікальною, що дозволяє здійснити прив'язку програми для певного пристрою і певної операційної системи, і змінюється лише в випадку заміни накопичувача або інших складових комп'ютера [1]. Звичайно, у цьому випадку захист програми необхідно здійснити заново (тобто, «переприв'язати»).

Для попередження копіювання ключової інформації та обходу захисту, інформацію вирішено записувати в файл у хешованому вигляді, що значно ускладнює процес зламу.

Розроблений засіб захисту працює таким чином. При першому запуску створюється файл з даними, а при наступних інформація з цього файлу порівнюється з реальними параметрами системи. Для визначення першого запуску програми, користувачу при дистрибуції програмного засобу видається унікальний хешований текст, який повинен збігатись з текстом в коді захисту. Після виконання процедури першого запуску дистрибутивний ключ видаляється.

1) За замовчуванням вважається що в доступі відмовлено (*access=false*). При спробі відкриття головного робочого вікна спрацьовує функція, що намагається відкрити конкретний файл *keyFile* в конкретній директорії, в якому міститься, як вважається, згенерована при першому запуску ключова інформація. Далі алгоритм порівнює зчитані з файлу *keyFile* дані з реальними параметрами системи та видає результат порівняння (рис. 1).
2) При співпадінні даних надається доступ (*access=true)* до роботи програми, в іншому випадку користувачеві вказується де саме повинен знаходитись файл перевірки, а також пропонується пройти процедуру першого запуску програми, якщо це відповідає дійсності.

3) При ініціюванні процедури першого запуску викликається функція, що порівнює ключ дистрибутивного текстового файлу з ключем записаним в змінних в коді програми, при збігу створюється файл з хешованою системною інформацією, що буде порівнюватись при наступних запусках, дистрибутивний файл видаляється, надається доступ, в випадку помилки на будь-якому з етапів цього пункту в доступі відмовляється.

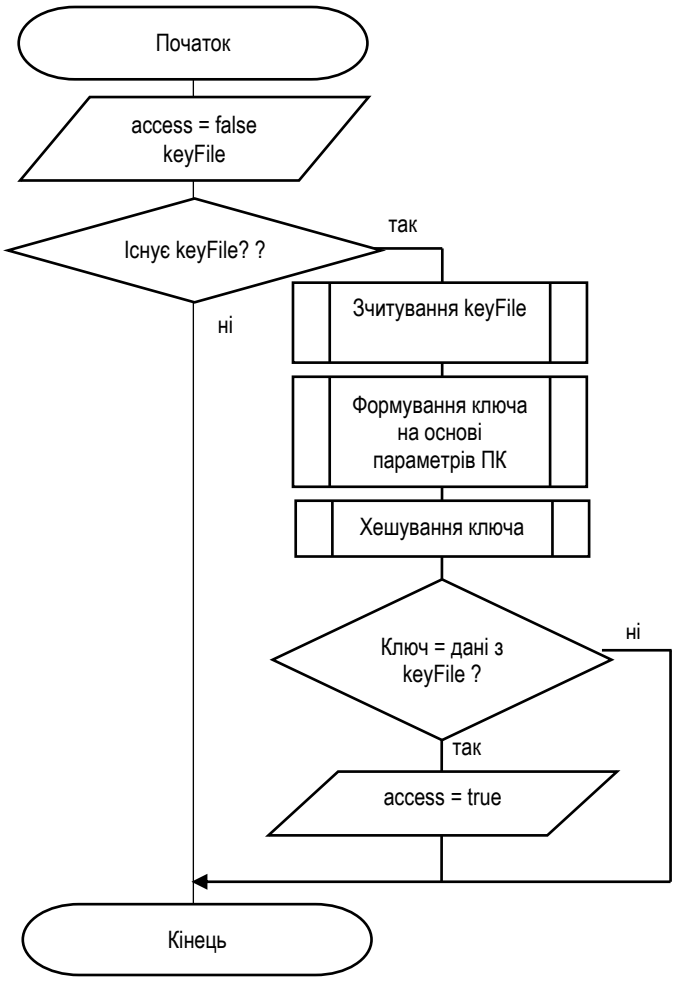

Рисунок 1 – Схема перевірки ключа

Результати тестування розробки

При розповсюдженні (продажі) захищеного програмного засобу користувачу для санкціонованого доступу до робочого вікна програми буде видаватись текстовий файл з даними для процедури першого запуску [2]. На початку роботи захищуваної програми буде здійснено спробу знайти файл з хешованою інформацією прив'язки до системи, але очікувано для першого запуску не знаходить його. Програма вказує шлях, де повинен знаходитись файл перевірки прив'язки, а далі у випадку першого запуску пропонує пройти відповідну процедуру.

Якщо користувач впевнений в наявності файлу перевірки прив'язки, він може відхилити процедуру, впевнитись в знаходженні файлу в необхідній директорії та повторно відкрити робоче вікно програми.

В іншому випадку, при наявності дистрибутивного текстового файлу, слід погодитись на процедуру першого запуску. Програма надасть можливість вказати шлях до дистрибутивного файлу. Після вибору коректного файлу програма перевіряє співпадіння даних. Далі при позитивному результаті створює новий файл з даними прив'язки до системи, видаляє дистрибутивний файл і, в разі успішного виконання всіх операцій, надає доступ. При наступних запусках алгоритм буде автоматично та непомітно для користувача перевіряти на співпадіння дані з файлу прив'язки та фактичні системні параметри і, в разі наявності коректного файлу в коректній директорії, надавати доступ.

Робота програми при спробах обійти захист

Під спробою обійти захист мається на увазі спроба несанкціонованого доступу до робочого вікна програми [3]. Такі умови передбачають відсутність у умовного користувача легального дистрибутивного текстового файлу, лише сам програмний засіб. Отож найпростішим варіантом було б скопіювати разом з програмою і файл перевірки прив'язки та спробувати запустити програму. Але, оскільки прив'язка відбувалась до унікальних для кожної системи номерів логічних дисків (або інших параметрів), програма видасть повідомлення про помилку.

Іншим варіантом обходу захисту була б спроба підробити дистрибутивний файл та пройти процедуру першого запуску, для цього можливо застосувати реверсивну інженерію та добути зі змінних в коді фрагменти ключа. Але завдяки застосуванню обфускації не одразу зрозуміло, яким чином фрагменти поєднуються в один ключ [4], тому при послідовному з'єднанні фрагментів та запису їх в текстовий файл і спробі пройти процедуру першого запуску програма видасть повідомлення про помилку.

Авжеж, при більш детальному аналізі коду досить легко зрозуміти логіку програми та зламати захист, але пересічному користувачу це може видатись не під силу.

Також для зламу можна застосувати соціальну інженерію та скопіювати дистрибутивний файл до його видалення, чому в свою чергу можна запобігти шляхом використання певної кількості різних варіантів дистрибуційних ключів.

Висновки

Таким чином, розроблено засіб для захисту програм від несанкціонованого копіювання і використання. Розроблений захист поєднує в собі декілька методів захисту:

- прив'язка до унікальних характеристик комп'ютера та операційного середовища;
- прив'язка до ключового файлу, який, до того ж, знаходиться у хешованому вигляді;
- обфускація даних ключового файлу, що робить складною задачу реверсивної інженерії.

Тестування розробки довело ефективність використаних способів захисту, а також зручність користування захищеним програмним засобом. Передбачено утруднення можливості зламу та обходу захисту.

Список використаних джерел

1. Кавун С. В. Інформаційна безпека: підручник / С. В. Кавун. - Харків : ХНЕУ, 2009. - 368 с.
2. Юричев Д., Введение в reverse engineering для начинающих. – 2020. – 1056 с. - [Електронний ресурс] – Режим доступу: URL: https://www.twirpx.com /file/2715490/.
3. Панов А. Реверсинг и защита программ от взлома – СПб.: BHV, 2006 – 230 с.
4. Каплун В. А., Дмитришин О. В., Баришев Ю. В. Захист програмного забезпечення : лабораторний практикум – Вінниця : ВНТУ, 2016. – 75 с.

Нагирняк Олег Юрійович - студент групи 1 БС-18 б, факультет інформаційних технологій та комп'ютерної інженерії, Вінницький національний технічний університет, Вінниця, e-mail: olegnagirnjak@gmail.com.

Каплун Валентина Аполінаріївна, ст. викл. кафедри захисту інформації, Вінницький національний технічний університет, Вінниця, valuka8379@gmail.com.

Nagirnyak Oleg – student group 1 BS-18 b, Department of Information Technology and Computer Engineering, Vinnytsia National Technical University, Vinnytsia, olegnagirnjak@gmail.com.

Valentyna A. Kaplun – Lecturer of the Chair of Safety of Infomation and Communication Systems, Vinnytsia National Technical University, Vinnytsia, valuka8379@gmail.com.