УДК 004.4

В.О. ЄМЕЛЬЯНОВ[1], **А.В. БІРЮКОВ**[2]

[1] ***Севастопольський інститут банківської справи УАБС НБУ , Україна***
[2] ***Асоціація «ІТ-Україна», Київ, Україна***

ІНСТРУМЕНТАЛЬНИЙ ЗАСІБ ЗАБЕЗПЕЧЕННЯ ІНФОРМАЦІЙНОЇ БЕЗПЕКИ КОНФІДЕНЦІЙНИХ ДАНИХ

Описана проблема захисту конфіденційної інформації. Приведені схеми функціонування інструментального засобу в режимі введення та відновлення конфіденційної інформації. Спроектована структура інструментального засобу, у вигляді програмного забезпечення, для захисту конфіденційних даних. Описано алгоритм генерації ключів для розробленого програмного забезпечення. Запропоновано принцип захисту конфіденційних даних, який полягає в блокуванні роботи центрального процесору на момент відображення паролю. Описано призначення драйверу захисту конфіденційної інформації, виходячи із запропонованого принципу.

Ключові слова: *програмне забезпечення, драйвер захисту, конфіденційна інформація, генерація ключів.*

Вступ та постановка завдання

Однією з важливіших проблем захисту та зберігання інформації є проблема витоку інформації, та персональних даних зокрема.Згідно з дослідженням [1], у ході якого були вивчені наслідки 31 витоку, кожний витік будь-якої інформації у середньому наносить компанії збиток у розмірі 4,8 млн дол. Також [1] внаслідок крадіжки персональних даних (паролі) у руки зловмисників потрапили приватні записи від 2,5 до 263 тис. чоловік з кожної постраждалої організації. У середньому на одну фірму доводиться 26,3 тис. записів, а в сумі це будуть персональні дані майже 815 тис. громадян.

У всесвітній мережі Інтернет існує достатньо схожих програм, що розрізняються за ступенем захисту інформації, способом її зберігання та відображення (інтерфейс користувача), а також способом розповсюдження (Freeware, Shareware, Open-Source та ін.). Призначення всіх цих програм допомогти користувачу надійно зберігати конфіденційні дані, наприклад, паролі для доступу в Інтернет, електронній пошті та іншим сервісам або просто якусь секретну інформацію.

Використання однакового паролю для доступу до різних даних є небезпечним, бо виток пароля чи ключа при доступі до будь-яких даних, одночасно, дає доступ до всієї інформації. Є два можливі рішення цієї проблеми – використовувати різні методи генерації ключа з одного паролю, або використовувати різні паролі. Перший метод потребує великих витрат на створення достатньо крипостійких алгоритмів, а другий метод потребує зусиль від користувача, а саме він полягає в запам'ятовуванні великої кількості інформації.

Для вирішення цих завдань існують різні програмні засоби. Найбільш простим та популярним засобом із захисту конфіденційних даних є програмний продукт SCARABAY. Але у програмі відсутня можливість автоматичного заповнення форм та зберігання різних типів даних. Інші пакети для вирішення подібних завдань Password Commander і Password Boss мають засоби для автоматичного заповнення форм, обидві дозволяють створювати поля для введення особистих даних, але вони не є достатньо стійкими.

Виникає достатньо багатогранна проблема автоматизації зберігання паролів. З одного боку – програмне рішення, що реалізує цю функцію, повинно надійно зберігати конфіденційні дані (повинен використовуватися крипостійкий алгоритм), також повинно бути захищена від витоку інформації в процесі роботи (під час її вводу, виводу), з іншого – бути мобільним, щоб користувач міг у будь який момент скористатися програмою. Таким чином, є необхідність розробки засобу для захисту конфіденційних даних.

1. Проектування механізму функціонування інструментального засобу

Умовно роботу програми можна розділити на два різних алгоритми:

- збереження даних користувачем за допомогою алгоритму шифрування та введеного пароля;

- відновлення даних за введеним паролем та вивід їх користувачу.

Збереження особистих даних користувача здійснюється за такою схемою.

Програма пропонує вибрати файл БД або створити новий і ввести для нього пароль. Після цього відкривається головне вікно програми. Користувач редагує структуру інформації (при цьому дані одразу ж шифруються і зберігаються в файл у зашифрованому вигляді). Поле «Додаткова інформація» як й усі дані структури шифрується ключем структури.

При редагуванні програма запросить ввести особисту інформацію (пароль) та повторно ввести пароль від файлу, за яким генерується ключ інформації. За допомогою ключа інформації дані шифруються і одразу ж зберігаються у файл. Опісля шифрування нешифрована інформація і ключ безпечно видаляються із пам'яті (замінюються нулями або випадковими значеннями).

Розроблений програмний засіб передбачає функцію зберігання паролю (мається на увазі збереження отриманий по паролю ключ для доступу к конфіденційній інформації, але ж в ніякому разі не сам пароль). При виборі цієї опції програма попереджає користувача про можливі наслідки

Структурна схема введення інформації відображена на рис. 1.

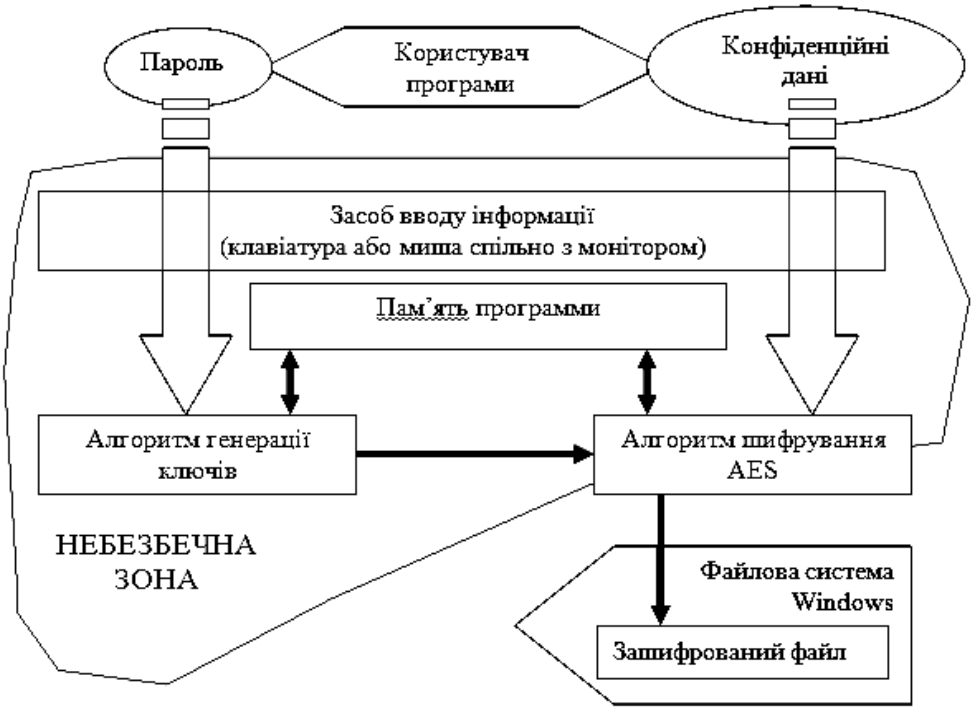

Рис. 1. Структурна схема введення конфіденційної інформації

Програмний засіб має деяку послідовність символів – пароль, та деяку конфіденційну інформацію. За допомогою засобу введення інформації (клавіатура чи миша спільно з монітором) користувач вводить пароль до зашифрованого файлу. За допомогою алгоритму генерації ключів, створюються ключі для шифрування та дешифрування конфіденційної інформації. Створенні ключі зберігаються в оперативній пам'яті. Конфіденційні дані, введені з використанням клавіатури чи миші, за допомогою алгоритму шифрування AES [2], що використовує згенеровані ключі, перетворюються на зашифровану послідовність даних, що можуть бути безпечно збережені та транспортовані.

Червоним кольором на рис. 1 виділена так звана «небезпечна зона», це зона є ненульова вірогідність витоку конфіденційної інформації. Клавіатура може бути просканована та конфіденційні дані можуть бути вилучені з системи. Алгоритм генерації ключів та алгоритм шифрування може бути локалізований в програмному файлі та модифікований сторонніми особами з метою несанкціонованого отримання конфіденційних даних користувача.

Пам'ять програми також потрапляє в «небезпечну зону». Збереження ключів в пам'яті програми є небезпечним, тому що в будь який момент може бути створений дамп пам'яті процесу і ключі доступу до конфіденційних даних можуть отримані безпосередньо з цього дапму.

Структурна схема відновлення інформації відображена на рис. 2.

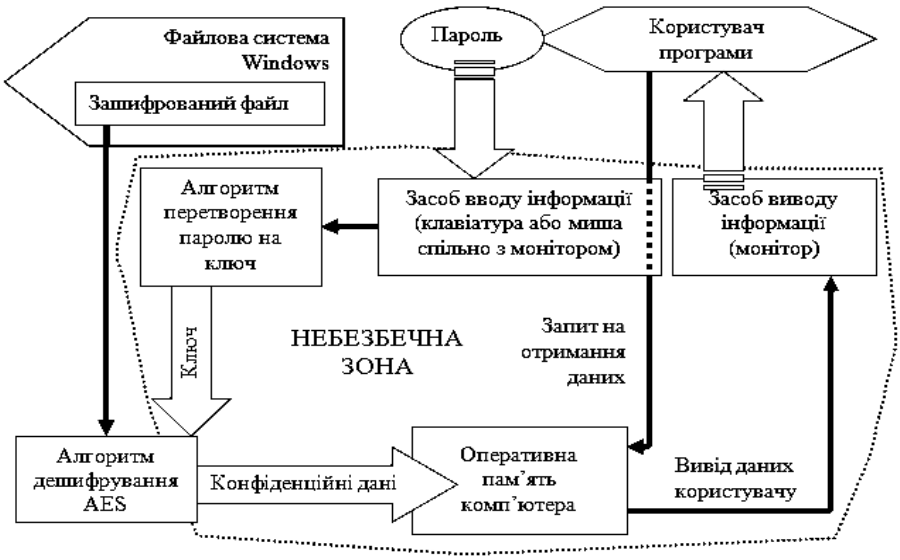

Рис. 2. Структурна схема відновлення конфіденційної інформації

Згідно рис. 2, зашифрований файл зберігається в файловій системі Windows. Користувач передає в систему пароль за допомогою засобу введення інформації. Алгоритм перетворення паролю на ключ генерує ключ структури. За допомогою алгоритму дешифрування AES структура файлу розшифровується та зберігається в оперативній пам'яті. Дані виводяться користувачеві на монітор.

Для вилучення конфіденційних даних, користувач вибирає з отриманої структури елемент. Вводить пароль або використовує раніше генерований та збережений ключ. Дані розшифровуються та виводяться на монітор (рис. 3).

Алгоритм дешифрування AES частково попадає в «небезпечну зону», бо на його вході зашифровані дані, отримання яких не має сенсу для зловмисника за умови, що він не отримав ключ для розшифровки цих даних.

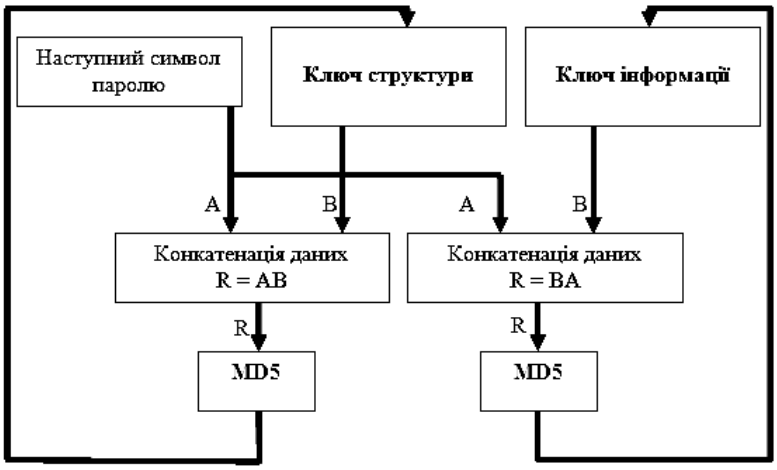

Рис. 3. Структурна схема алгоритму генерації ключів

Відповідно до вибраного алгоритму шифрування AES-128 [2-3], що потребує 128-бітний ключ, вибрано алгоритм MD5 з довжиною 128 біт.

На рисунку 3 представлено запропонований алгоритм генерації ключа інформації.

Відповідно до представленої структурної схеми, генерація ключа відбувається нерозривно з введенням паролю.

Ключі структури та інформації ініціалізуються деяким початковим значенням.

Відповідно до вимог захисту від модифікації програми, це значення є відбитком виконуваного файлу програми за алгоритмом MD5. Якщо опція захисту від модифікації відключена, ключі структури та інформації ініціалізуються нулями.

Після кожного введення символу проводиться конкатенація даних. Для генерації ключа структури введений символ додається перед ключем, для ключа інформації – після ключа. В обох випадках є послідовність з 16 + 1 = 17 байт, для яких вирахо-

вується відбиток за алгоритмом MD5. Таким чином отримуємо наступні ключі – інформації та структури. Зворотна генерація неможлива (тобто неможливо знаючи введений символ отримати значення ключа до введення символу).

2. Інструментальний засіб захисту конфіденційної інформації

Для розробки інструментального засобу використовувалась мова програмування – «C++».

Діаграма компонентів, що відображує структуру розроблюваного програмного засобу, наведено на рис. 4.

Структуру програми умовно можна поділити на три частини:

– драйвер захисту;

– логічна частина (реалізує необхідні базові алгоритми)

– візуальна частина (забезпечує зв'язок користувача з логічною частиною).

Між собою частини зв'язані ієрархічно, візуальна частина використовує інтерфейси логічної, яка в свою чергу використовує драйвер захисту

Захист програми від сканування пам'яті та графічного зображення головної форми реалізоване за допомогою драйверу режиму ядра Windows NT [4, 5].

Принцип роботи захисту полягає в блокуванні роботи центрального процесору на момент відображення паролю. Під блокуванням роботи центрального процесору мається на увазі монопольне його використання драйвером на деякий, вибраний користувачем час.

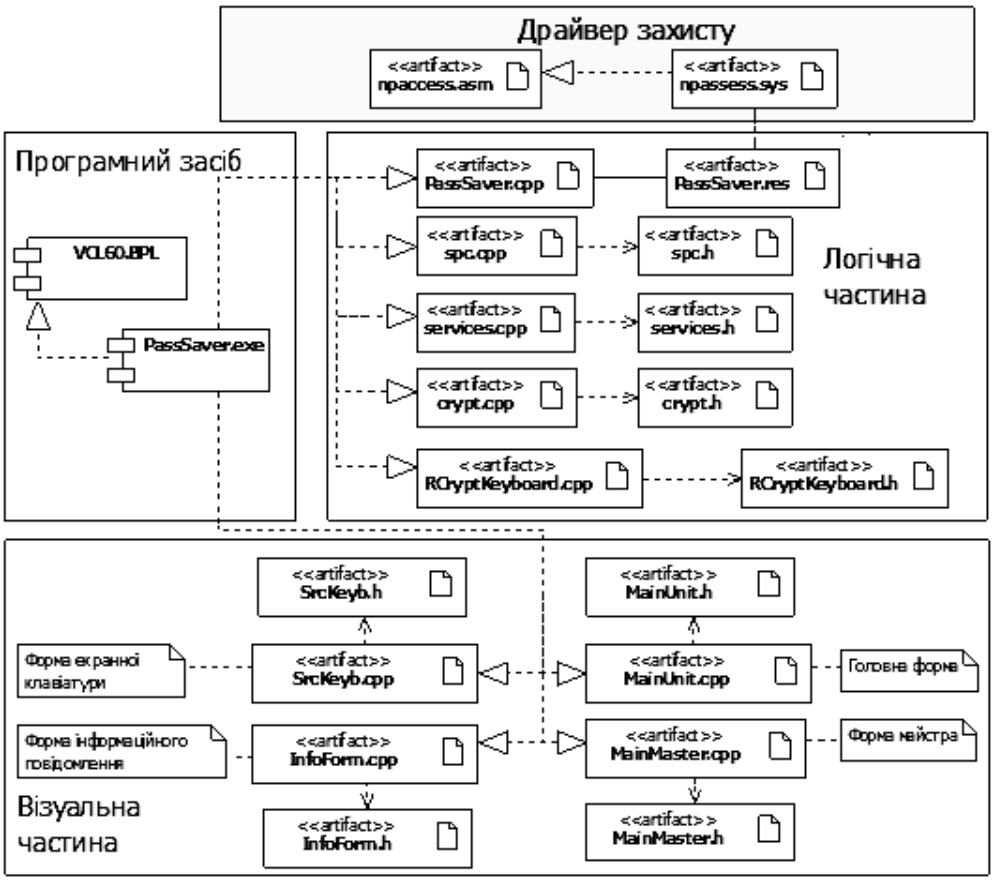

Рис. 4. Структура інструментального засобу захисту конфіденційних даних

Основні моменти в реалізації драйвера – створення віртуального пристрою, обробка пакетів IRP, що формуються диспетчером введення-виведення, і коректне вивантаження драйвера з пам'яті (а саме, видалення віртуального пристрою).

Ініціалізація драйвера полягає в створенні віртуального пристрою. При завантаженні драйвера в пам'ять, викликається процедура DriverEntry, в якій необхідно розмістити відповідний код.

Якщо створення пристрою пройшло без помилок, то наступним нашим кроком буде створення символьного посилання, що вказує на пристрій. Для забезпечення безпеки користувач не може безпосередньо звертатися до пристрою, оскільки об'єкти ядра не доступні режиму користувача. Тому створюється символьне посилання, видиме з режиму користувача, поводження із запитом до якої викликає формування диспетчером завдань IRP-

пакету і напрям його пристрою. Це дозволить коду режиму користувача дістати доступ пристрою.

У режимі ядра Windows не стежить за діями драйверів, і якщо драйвером був виділений будь-який ресурс – наприклад, блок оперативної пам'яті, – то тільки драйвер знає про цей блок пам'яті і при вивантаженні драйвера, інформація про нього втрачається і його вивантаження абсолютно унеможливлюється.

Також актуальна ситуація, коли сегмент коду драйверу буде розформовано, а створений їм пристрій ще буде присутній в системі, звернення до останнього спричинить непередбачувані наслідки – швидше за все це призведе до зависання системи. Тому в процедурі вивантаження драйвера передбачено видалення символьного посилання на пристрій, а також самого пристрою.

Драйвер розроблено за допомогою мови MacroAssembler та додатку KMDKit.

Висновки

В результаті функціонування інструментального засобу була досягнута можливість забезпечення інформаційної безпеки конфіденційних даних, таких як ім'я користувача і його пароль. Алгоритм генерації ключів за паролем, що є складовим методу введення інформації за допомогою радіальної екранної клавіатури, розроблений так, що дає змогу не зберігати пароль протягом його введення до системи, що унеможливлює виток його стороннім особам.

Література

1. 2006 Annual Study: Cost of a Data Breach [Електронний ресурс]. – Режим доступу: http://www.michiganbusiness.us/showcompany.php?id=38903. – 22.03.2012.

2. Фергюсон, Н. Практическая криптография. [Текст] / Н. Фергюсон, Б.Шнайер. – К.: Диалектика, 2004 – 432 с.

3. Масленников, М. Практическая криптография [Текст] / М. Масленников. – СПб.: BHV, 2003. – 458 с.

4. Смит, Г. Windows Driver Foundation: Разработка драйверов [Текст]: пер. с англ. / Г. Смит, П. Орвик. – БХВ-Петербург, 2008. – 880 с.

5. Шрайбер, С.Б. Недокументированные возможности Windows 2000 [Текст] / С.Б. Шрайбер. – СПб.: Питер, 2002. – 544 с.

Надійшла до редакції 22.03.2012

Рецензент: д-р техн. наук, проф. І.А. Жуков, Національний авіаційний університет, Київ, Україна.

ИНСТРУМЕНТАЛЬНОЕ СРЕДСТВО ОБЕСПЕЧЕНИЯ ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ КОНФИДЕНЦИАЛЬНЫХ ДАННЫХ

В.А. Емельянов, А.В. Бирюков

Описана проблема защиты конфиденциальной информации. Приведены схемы функционирования инструментального средства в режиме ввода и восстановления конфиденциальной информации. Спроектированная структура инструментального средства, в виде программного обеспечения для защиты конфиденциальных данных. Описан алгоритм генерации ключей для разработанного программного обеспечения. Предложен принцип защиты конфиденциальных данных, который заключается в блокировании работы центрального процессора на момент отображения пароля. Описано назначение драйвера защиты конфиденциальной информации на основе предложенного принципа.

Ключевые слова: программное обеспечение, драйверы защиты, конфиденциальная информация, генерация ключей.

THE SOFTWARE FOR PROTECTING OF THE CONFIDENTIAL DATA

V.A. Iemelianov, A.V. Biriukov

This article describes the problem of protecting confidential information. The schemes of operation of the tool in the input mode and restore of confidential information are developed. The structure of the software to protect sensitive data are designed. An algorithm for generating keys for the software was developed. The principle of protecting sensitive data, which is blocking the CPU at the time of displaying the password, was developed. The purpose of the security driver to protect confidential information, based on the proposed principle, is described.

Keywords: software, security driver, confidential information, the generation of keys.

Ємельянов Віталій Олександрович – канд. техн. наук, асистент кафедри інформаційних технологій та систем Севастопольського інституту банківської справи УАБС НБУ, Севастополь, Україна, e-mail: v.yemelyanov@gmail.com.

Бірюков Андрій Віталійович – віце-президент Асоціації «ІТ-Україна», радник-консультант Верховної Ради України, Київ, Україна.