https://orcid.org/0000-0003-4326-3030
https://orcid.org/0000-0002-8698-0972
https://orcid.org/0000-0002-2855-3729

УДК 004.41

Д.В. РАТОВ[*], В.Г. ІВАНОВ[*], Л.А. ЛИГІНА[*]

СТВОРЕННЯ МЕРЕЖЕВОЇ СИСТЕМИ АВТОРИЗАЦІЇ ДЛЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

[*]Східноукраїнський національний університет імені Володимира Даля, м. Сєвєродонецьк, Україна

Анотація. *Незважаючи на всі зусилля різних організацій на чолі з Business Software Alliance (BSA) – торгової асоціації, що представляє інтереси ряду найбільших у світі розробників програмного забезпечення, в останні роки триває ріст комп'ютерного піратства. У продовження досліджень і розробки інформаційних систем у статті розглядається програмний метод забезпечення захисту розроблюваного програмного забезпечення, призначений для збереження конфіденційності даних від несанкціонованого доступу. Під захистом в інформаційних технологіях розуміють сукупність дій, спрямованих на протидію нелегальному копіюванню й несанкціонованому доступу. Розглянутий метод полягає в авторизації користувача програмного забезпечення чи інформаційної системи через Інтернет і допускає первинну активацію продукту на обчислювальній машині користувача, а також авторизацію користувача на сервері при кожній активації інформаційної системи чи програмного забезпечення. Така авторизація дозволяє розроблювачеві стежити за статистикою використання інформаційної системи, виявляти випадки порушення ліцензій, позбавляти ліцензій несумлінних користувачів, а також гнучко контролювати ліцензійну політику. У даній роботі запропонований протокол контролю авторизації програмного забезпечення, який заснований на використанні прив'язки до апаратного забезпечення обчислювальної машини й шифрування 128-бітним алгоритмом хешування MD5. Програмною імплементацією є динамічна бібліотека, що інтегрується із програмним забезпеченням користувача й контролює його авторизацію і роботу функціонала, php-сценарії взаємодії динамічної бібліотеки й бази даних MySQL, php-сценарій створення web-форми адміністрування, android-додаток адміністрування авторизації програмного забезпечення. Одержаний засіб програмного захисту при розробці програмного забезпечення та інформаційних систем є своєчасним і необхідним механізмом для контролю за санкціонованим використанням програмного продукту.*

Ключові слова: *авторизація, захист програмного забезпечення, програмно-апаратні засоби захисту, ідентифікація, аутентифікація, динамічна бібліотека, протокол авторизації, web-програмування, інформаційні технології, інженерія програмного забезпечення, комп'ютерні науки.*

Abstract. *Despite the efforts of various organizations led by the Business Software Alliance (BSA), a trade association representing some of the world's largest software developers, computer piracy has continued to grow in recent years. In the continuation of research and development of information systems the article considers a software method of the development software protection, designed to preserve the confidentiality of data from unauthorized access. In information technology protection is understood as a set of actions aimed at combating illegal copying and unauthorized access. The given method consists in authorizing the user of a software or an information system via the Internet and allowing the initial activation of the product on the user's computer, as well as user authorization on the server during each activating of the information system or software. Such authorization allows the developer to monitor statistics on the use of the information system, detect cases of license violations, revoke the licenses of dishonest users, as well as flexibly control the licensing policy. This paper proposes a software authorization control protocol which is based on the use of binding to computer hardware and encryption by 128-bit MD5 hashing algorithm. The software implementation is a dynamic library that integrates with the user's software and controls its authorization and functionality, php-scripts of interaction between the dynamic library and MySQL database, php-script for the creation of a web-form of administration, Android application of software authorization administration. The obtained means of software protection in the development of software and information systems is a timely and necessary mechanism for monitoring the authorized use of the software product.*

ISSN 1028-9763. Математичні машини і системи. 2021. № 2

Keywords: *authorization, software protection, software and hardware protection, identification, authentication, dynamic library, authorization protocol, web-programming, information technology, software engineering, computer science.*

DOI: 10.34121/1028-9763-2021-2-35-44

1. Вступ

Виробники програмного забезпечення зазнають досить великих збитків через нелегальне використання їх продукції. Найчастіше юридичні методи боротьби із правопорушниками не є ефективними, тому для захисту своїх інтересів розроблювачам програмного забезпечення (ПЗ) доцільно звертатися до технічних засобів захисту програмного продукту від нелегального використання.

До технічних методів захисту ПЗ відносять програмні й програмно-апаратні засоби, а також використання програм у вигляді онлайн-сервісів [3, 4]. До програмних методів відносяться методи [4], реалізовані софтверним і алгоритмічним шляхом, що припускають використання прив'язки програми до конфігурації обладнання, на якому її інсталювали. В таких методах не враховуються фізичні характеристики носіїв інформації чи спеціальне устаткування. Прив'язка відбувається в момент установки програмного забезпечення й вимагає або введення ліцензійного ключа, або проходження процедури активації – одержання одноразового кодового ключа, що залежить від конфігурації обладнання користувача й придатного для використання тільки на цьому комп'ютері. У першому випадку нічого не заважає користувачеві незаконно поширювати продукт разом з ліцензійним ключем. У другому випадку сумлінному користувачеві необхідно погодитися з незручностями, які виникають при кожній зміні обладнання, на якому він використовує програмне забезпечення, а несумлінний користувач має можливість незаконно використовувати й поширювати програмне забезпечення разом з так званою віртуальною машиною [5], що програмно імітує необхідне конфігураційне обладнання.

Метою статті є створення засобу програмного захисту для розроблюваного програмного забезпечення чи інформаційної системи, які можуть бути задіяні розроблювачами програмного забезпечення при реалізації захисту від несанкціонованого використання й контролю авторизації програмних продуктів.

Постановка проблеми. Для рішення задачі забезпечення захисту розроблюваного програмного забезпечення необхідна сукупність обладнань і дій, призначених для збереження конфіденційності даних від несанкціонованого доступу. Найбільш перспективним для захисту програмного забезпечення є метод авторизації через Інтернет.

2. Засоби захисту програмного забезпечення

Методи захисту програмного забезпечення можна розділити на програмні й апаратні. Отже, засобами захисту програмного забезпечення є технічне і апаратне обладнання, а також програмні рішення, що дозволяють позбавити програмне забезпечення від несанкціонованого впливу ззовні.

Програмно-апаратні засоби захисту [3, 4] припускають перевірку наявності деяких апаратних засобів, які поставляються разом із програмою й для яких неможливо або дуже важко виготовити копію. Наприклад, широко використовують спеціально виготовлені CD, DVD і апаратні електронні ключі, що підключаються до USB-портів комп'ютера, щоб ідентифікувати оригінальну версію програми й захистити продукт від нелегального використання. Такі методи вважаються більш надійними, ніж програмні, але вони мають істотні недоліки. По-перше, таке програмне забезпечення можна поставляти тільки в «коробковій» версії, по-друге, такий засіб захисту може суттєво збільшити вартість програмного продукту, по-третє, користувачеві також потрібно погодитися з деякими незручностями. У зв'язку з цим даний метод захисту не придатний для більшості програмних продуктів. Крім

того, відомі [4] кількаразові випадки злому захисту обох типів і подальшого нелегального й безперешкодного поширення програмного забезпечення.

Надійним засобом від копіювання програм можна вважати роботу їх як онлайн-сервісів, так званих Software as a service (SaaS) [6], які припускають виконання програмного забезпечення на стороні виробника, не доступній для користувача. Користувач же може тільки ввести вхідні й одержати вихідні дані через веб-інтерфейс. Такий вид захисту мало придатний для програмного забезпечення, яке вимагає більших об'ємів вхідних або вихідних даних, і в кожному разі він не зможе замінити традиційні способи поширення програм.

Програмні засоби забезпечення захисту інформації представляють комплексну сукупність програмних дій, призначених для збереження конфіденційності даних. Засоби програмного захисту можна поділити за такими категоріями:

1) власний захист: засоби, вбудовані безпосередньо у програмне забезпечення для захисту інформації на всіх стадіях його використання;

2) засоби, що є частиною операційної системи – брандмауери;

3) засоби захисту з запитом інформації – для здійснення повноцінної роботи із програмним продуктом потрібно ввести код доступу;

4) активний захист – використовується при спробах несанкціонованого впровадження у програмне середовище, наприклад, неправильним введенням пароля або порушенням порядку авторизації;

5) захист від шкідливого програмного забезпечення за допомогою спеціалізованих антивірусних програм;

6) пасивний захист – пошук відомостей, що підтверджують факт спроби вторгнення у програму й розкрадання даних.

Найбільш ефективними й такими, які часто використовуються, є такі технічні засоби захисту програмного забезпечення:

1. Локальний програмний захист – застосування серійних номерів (ключів), які можуть бути використані тільки для однієї копії програми. Цей метод ефективно використовувався тоді, коли програми поширювалися тільки на фізичних носіях (наприклад, на компакт-дисках). На коробці з диском був надрукований серійний номер, який підходить тільки до даної копії програми. З поширенням мереж очевидним недоліком стала проблема поширення образів дисків і серійних номерів по мережі. Тому в даний момент метод використовується тільки в комплексі з іншими методами.

2. Мережевий програмний захист:

a) локальний, здійснюється шляхом сканування мережі на предмет наявності двох однакових реєстраційних ключів у різних користувачів і їх блокування. Недолік захисту полягає в тому, що брандмауер можна налаштувати так, щоб він не пропускав пакети, які належать захищеній програмі. Крім того, програма може взаємодіяти по мережі (наприклад, організація мережевої гри). У цьому випадку брандмауер повинен пропускати такий трафік;

b) глобальний, здійснюється за допомогою використання віддалених серверів, з якими повинна зв'язатися копія програмного продукту користувача для забезпечення її нормальної працездатності. У випадку, якщо ключ, що використовується, є не унікальним або забороненим, можливість роботи із програмою буде заблокована.

Також потрібно відзначати існування можливості проведення аналізу загроз і функціональності з використанням спеціальних протоколів, які показують зв'язки між можливими причинами відмови елементів джерела, їх впливом на функціонування системи управління і наслідками відмови на функції системи. Раціональними для оцінки ймовірності відмов є методи дерев відмов (FTA – fault tree analysis), що визначають імовірність ініціюючих небезпечних подій і метод дерев подій (ETA – event tree analysis) для урахування відмов систем захисту і визначення сценаріїв наслідків таких відмов.

3. Метод авторизації через глобальну мережу

В інформаційних системах під ідентифікацією розуміють процедуру, у результаті виконання якої для суб'єкта ідентифікації виявляється його ідентифікатор, однозначно ідентифікуючий цього суб'єкта в інформаційній системі [20]. Для виконання процедури ідентифікації в інформаційній системі суб'єктові попередньо повинен бути призначений відповідний ідентифікатор, тобто проведена реєстрація суб'єкта в інформаційній системі.

Процедура ідентифікації прямо пов'язана з аутентифікацією (процедурою перевірки дійсності): суб'єкт проходить процедуру аутентифікації, і якщо аутентифікація успішна, то інформаційна система на основі факторів аутентифікації визначає ідентифікатор суб'єкта [21]. При цьому вірогідність ідентифікації повністю визначається рівнем вірогідності виконаної процедури аутентифікації.

В інформаційних системах під авторизацією (англ. authorization – дозвіл, уповноваження) розуміють надання певній особі або групі осіб прав на виконання певних дій, а також процес перевірки (підтвердження) даних прав при спробі виконання цих дій [21, 22].

Достатньо перспективним для захисту програм є метод авторизації через Інтернет [11, 23], який припускає первинну активацію продукту на обчислювальній машині користувача, а також авторизацію користувача на сервері розроблювачів при кожному запуску програми. Даний метод має мале поширення й готових рішень у відкритому доступі не існує [23]. Авторизація користувача при кожному запуску програми дозволяє розроблювачеві стежити за статистикою використання програми, виявляти випадки порушення ліцензій, позбавляти ліцензій несумлінних користувачів, а також гнучко змінювати ліцензійну політику.

Як обмеження на використання безплатних і умовно безкоштовних програм може служити введення обмеження на легальне використання програмного продукту, наприклад, обмеження на кількість запусків програми на місяць або обмеження на кількість переглянутих і створених документів за деякий проміжок часу або недопущення одночасного запуску програми на декількох комп'ютерах [23]. Такі обмеження повинні бути прописані в ліцензійній угоді разом із санкціями за їхні порушення. Під санкцією може матися на увазі як повне, так і тимчасове позбавлення ліцензії. Перевірка порушення лімітів повинна проводитися в недоступному для користувача модулі контролю ліцензій, наприклад, віддаленому сервері, куди програма повинна посилати дані й перевіряти наявність дозволу на запуск. Якщо раптом нелегальна копія програми виявиться опублікованою у публічному місці, то ліміти досить швидко будуть перевищені, й ліцензія буде заблокована. Для того щоб організувати такий захист, необхідно організувати безпечний обмін по відкритому каналу зв'язку між програмою й модулем, у якому несумлінний користувач може читати й змінювати будь-які дані, що пересилаються. У зв'язку з поширенням доступу до Інтернету такий обмін можна організувати досить просто, не викликаючи значних незручностей у користувачів. Крім того, програма може контролювати спробу зміни коду й посилати повідомлення серверу. Для реалізації такої схеми розглянемо протокол передачі даних між динамічною бібліотекою, що підключається до програми, й віддаленим сервером.

4. Протокол авторизації програмного забезпечення

У даній роботі запропонований протокол контролю авторизації програмного забезпечення, заснований на використанні прив'язки до апаратного забезпечення обчислювальної машини й шифрування 128-бітним алгоритмом хешування MD5 відкритого ключа [12, 24]. Схема взаємодії програмного забезпечення з сервером представлена на рис. 1.

Розглянемо алгоритм контролю авторизації додатка CheckRun.exe, який інстальований на комп’ютері користувача User. Віддалений сервер Server належить розроблювачам додатка або їх довіреній особі.

За протоколом авторизації програмного забезпечення генерується відкритий ключ (key_client), який буде зашифрований 128-бітним алгоритмом хешування MD5. Закритий ключ (key_server) повинен зберігатися на сервері Server. При першому запуску CheckRun.exe програма одержує ідентифікатор Identification із сервера Server при умові, що адміністратор дав загальний дозвіл на роботу в системі й додавання нових користувачів. Щоразу, коли користувач User намагається виконати одну з дій, закріплених для контролю розроблювачем (наприклад, запуск програми, створення, відкриття або збереження документа), програма CheckRun.exe з підключеною динамічною бібліотекою RequestRespons.dll надсилає запит серверу Server про можливість продовжити роботу, зробивши такі кроки передачі даних (рис. 1):

1. Програма CheckRun.exe за допомогою динамічної бібліотеки RequestRespons.dll перевіряє підключення до інтернету і у випадку його відсутності приховує робочий функціонал програми й видає відповідне повідомлення.

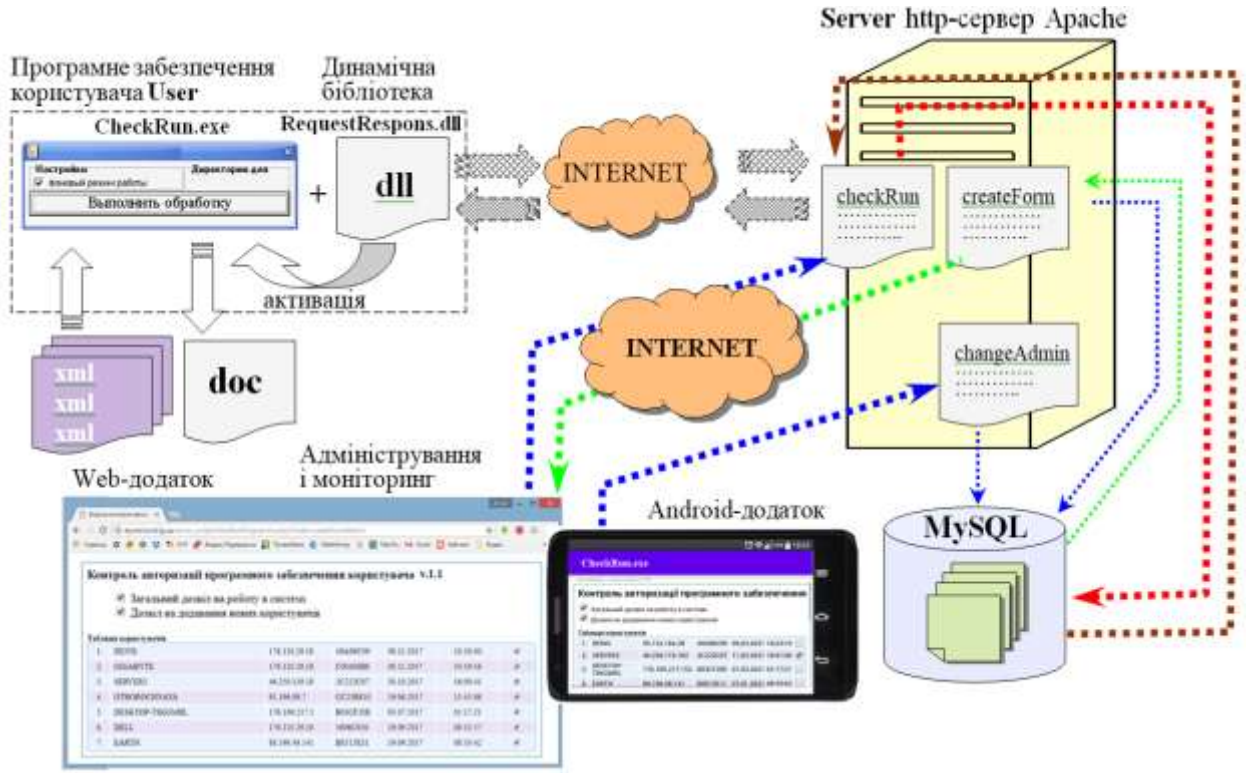

Рисунок 1 – Схема обміну пакетами між програмним забезпеченням, для якого організовується контроль авторизації з віддаленим сервером

2. В об’єкті admin із класу classRequest бібліотеки RequestRespons.dll формується параметр Comp – ім'я локального комп'ютера й параметр IP – IP-адреси локального комп'ютера.

3. В об’єкті admin із класу classRequest бібліотеки RequestRespons.dll обчислюється параметр SerNum – прив'язка до апаратного забезпечення обчислювальної машини, де вона встановлена (4-байтний серійний номер системного диска C).

4. Програма CheckRun.exe за допомогою динамічної бібліотеки RequestRespons.dll передає пакет із параметрами key_client, Mode, IP, Comp, SerNum сервера Server.

5. Сервер Server перевіряє можливість використання програми користувача з ідентифікатором Identification, який визначається іменем локального комп'ютера Comp, IP-адресою локального комп’ютера – IP, серійним номером системного диска SerNum і порівнює зашифрований ключ key_client із ключем key_server.

6. Сервер Server відправляє пакет із параметром f_status об’єкта admin із класу classRequest бібліотеки RequestRespons.dll, зв’язаної із програмним забезпеченням Check-Run.exe.

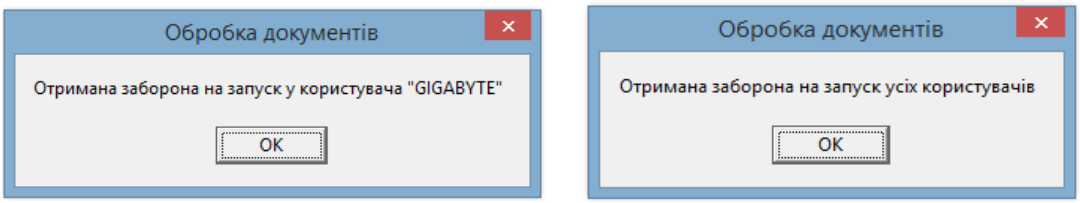

Рисунок 2 – Реакція бібліотеки на заборонний пакет
активації програми CheckRun.exe

7. Програма CheckRun.exe за допомогою динамічної бібліотеки RequestRespons.dll одержує пакет від сервера Server. Якщо в параметрі f_status підтверджена можливість запуску, то бібліотека RequestRespons.dll активує головну форму програми CheckRun.exe й відображає робочу панель, а якщо ні, то видає повідомлення і завершує роботу програми CheckRun.exe (рис. 2).

5. Програмна реалізація об'єктної моделі

В інтегрованому середовищі розробки Visual Studio розроблена динамічна бібліотека, яка інтегрується з користувацьким програмним забезпеченням і контролює його авторизацію.

Інтерфейс класу classRequest, методи якого реалізують протокол авторизації програмного забезпечення, наведено на рис. 3. Як механізм шифрування обраний 128-бітний алгоритм хешування MD5 [12, 24].

```
namespace Control
{
    public struct AnsverServer {...}
    public class classRequest
    {
        private string URL;
        private AnsverServer locAnsver;
        // конструктор класса
        public classRequest(string url = "http://www.google.com") { this.URL = url; }
        // метод получения хоста по URL
        private string Host(string URL) {...}
        // пакет данных, возвращенный сервером
        public virtual AnsverServer ansver { get { return locAnsver; } }
        public virtual string GetPCName { get { return Environment.UserName; } }
        public virtual string GetSerialNum { get { return Environment.MachineName; } }
        // метод проверки подключения к Internet
        public virtual bool isInternetConected{...}
        // метод отсылки пакета на сервер + формирование пакета-ответа
        public virtual void Query(string addUrl) {...}
    } // class classRequest
}
```

Рисунок 3 – Клас classRequest

Клас classRequest (рис. 3) складається з:

1. Приватного поля URL, в якому задається адреса web-ресурсу для пінгування наявності підключення до глобальної мережі.
2. Приватної структури locAnsver, в якій формується відповідь від сервера.
3. Приватного методу Host, в якому з адреси ресурсу розраховується хост ресурсу.
4. Конструктора класу з попередньою ініціалізацією.

5. Загальнодоступних методів GetPCName, GetSerialNum, які повертають ім’я і серійний номер комп’ютера.

6. Загальнодоступного методу isInternetConected, який повертає ознаку підключення до глобальної мережі.

7. Загальнодоступного методу Query, що виконує GET-запит до віддаленого сервера.

Рисунок 4 – Упровадження механізму авторизації у проєкт програми CheckRun.exe

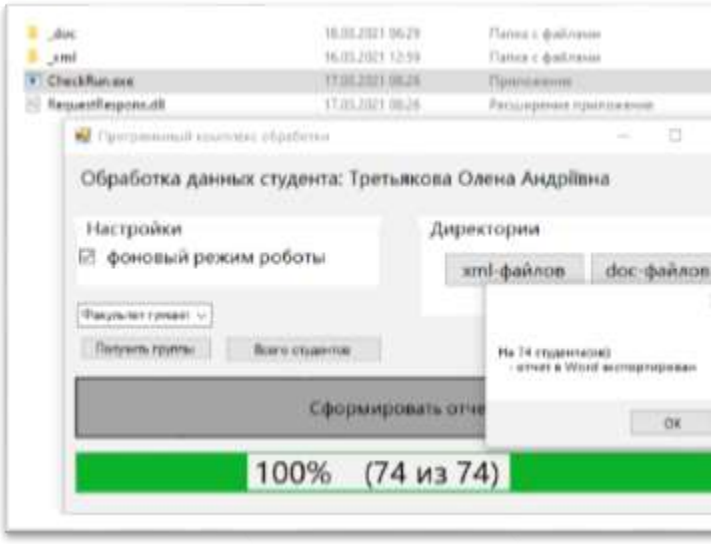

Рисунок 5 – Програмне забезпечення CheckRun.exe в активному режимі роботи

Згідно з розробленим протоколом авторизації програмного забезпечення, для використання функціонала динамічної бібліотеки RequestRespons.dll у проєкті програми CheckRun.exe необхідно змінити сценарій роботи (рис. 4). Замість ініціалізації головної форми програми (1) створюється екземпляр класу classRequest – admin (2) для пінгування наявності підключення до глобальної мережі (3) і для передачі й отримання пакетів від віддаленого сервера (4). Створений об’єкт дозволяє один із трьох можливих сценаріїв подальшої роботи програми:

1. При отриманні дозвільного пакета активації програма CheckRun.exe повідомить про успішну авторизацію (5) і виконає стандартну роботу програми (рис. 5).

2. При відсутності інтернету програма Check_Run.exe повідомить (7) і деактивує можливість стандартної роботи.

3. При отриманні забороненого пакета активації програма CheckRun.exe сформує повідомлення від сервера (6) і деактивує можливість стандартної роботи (рис. 2).

Серверна частина програмного комплексу реалізована мовою програмування PHP із можливістю запитів до сервера баз даних MySQL.

6. Аналіз розробленого протоколу

Зв'язок із сервером впроваджений об'єктом класу classRequest може здійснювати через глобальну мережу Інтернет за іменем його домену або IP-адресою.

Дозвіл на активацію програми делегується віддаленим сервером і може ґрунтуватися на таких даних: кількість зроблених успішних запусків ПЗ, час використання програми, оплачений користувачем баланс, кількість створених docx чи xlsx-файлів при використанні програми та ін.

Перед першим використанням ПЗ користувача програма CheckRun.exe одержує ідентифікатор і у випадку, якщо адміністратор дав загальний дозвіл на роботу в системі й додавання нових користувачів, активує свій функціонал. За допомогою web-форми адміністрування авторизації ПЗ або android-додатка (рис. 6) адміністратор системи може контролювати і виконувати моніторинг роботи ПЗ користувача – CheckRun.exe. Цей процес здійснюється коректуванням списка ознак роботи й функціонала ПЗ у окремих користувачів чи загальною забороною активації ПЗ (рис. 6).

Для унікальності реєстраційних даних для даного ідентифікатора в базі даних MySQL зберігаються ім'я локального комп'ютера, IP-адреса локального комп'ютера і 4-байтний серійний номер системного диска.

Рисунок 6 – Web-форма і android-додаток адміністрування авторизації ПЗ

Проаналізуємо розроблений протокол від таких видів атак [7, 23]:

1) атака підбору пароля. Для зменшення ймовірності підбору пароля користувача при реалізації протоколу введений лічильник неуспішних входів у систему й блокування на якийсь час можливості авторизації користувача після деякого числа невдалих входів;

2) колізія. Для унеможливлення використання раніше отриманих від сервера пакетів використовуються унікальні числа, генеровані з часу, та дати й зашифровані алгоритмом хешування MD5;

3) атака підміни відкритого ключа. Для неможливості підміни відкритого ключа віддаленого сервера в коді програми використовується алгоритм хешування.

7. Висновки

1. У ході виконання роботи була розроблена об'єктна модель ідентифікації та аутентифікації й запропонований протокол авторизації програмного забезпечення.
2. В інтегрованому середовищі розробки Visual Studio розроблена динамічна бібліотека, що інтегрується із програмним забезпеченням користувача і контролює його авторизацію і роботу функціонала.
3. Для препроцесора гіпертексту PHP, що працює на http-сервері Apache, розроблені сценарії взаємодії динамічної бібліотеки й бази даних MySQL, написаний сценарій створення web-форми адміністрування, а в інтегрованому середовищі розробки Android Studio створений додаток адміністрування авторизації ПЗ.
4. Отримані результати роботи й засіб технічного захисту для розроблювального програмного забезпечення можуть бути практично використані розроблювачами програмного забезпечення при реалізації його захисту від несанкціонованого використання й контролю авторизації програмних продуктів. Авторизація користувача при кожному запуску програми дозволяє розроблювачеві стежити за статистикою використання його програми, виявляти випадки порушення ліцензій, а також позбавляти ліцензій несумлінних користувачів.

СПИСОК ДЖЕРЕЛ

1. Ratov D. Architectural paradigm of the interactive interface module in the cloud technology model. *Applied Computer Science*. 2020. Vol. 16, N 4. P. 48–55.
2. Ратов Д.В. Модель модуля інтерфейсу користувача інформаційної web-системи. *Математичні машини і системи*. 2020. № 4. С. 74–81.
3. Варлатая С.К., Шаханова М.В. Программно-аппаратная защита информации: уч. пособ. Владивосток: Изд. ДВГТУ, 2007. 318 с.
4. Скляров Д.В. Искусство защиты и взлома информации. СПб.: Бхв-Петербург, 2004. 288 с.
5. Віртуальна машина. URL: http://ru.wikipedia.org/wiki/ Віртуальна_машина (дата звернення: 25.03.2021).
6. Software as a service. URL: http://ru.wikipedia.org/wiki/Software_as_a_service (дата звернення: 25.03.2021).
7. Захист від піратства й ліцензування ПО Starforce. URL: http://www.star-force.ru/ (дата звернення: 25.03.2021).
8. Захист програм Hasp. URL: http://www.aladdin-rd.ru/ (дата звернення: 25.03.2021).
9. Сервіс Software Activation Service. URL: http://www.softactivation.com/asp/getstarted.asp (дата звернення: 25.03.2021).
10. Захист Crypline. URL: http://crypline.ru (дата звернення: 25.03.2021).
11. Аналіз ринку засобів захисту від копіювання й злому програмних засобів. URL: http://citforum.ru/security/articles/analis/ (дата звернення: 25.03.2021).
12. Молдовян Н.А., Молдовян А.А. Введення у криптосистеми з відкритим ключем. Спб.: Бхв-Петербург, 2005. 288 с.
13. Буинцев Д.Н. Метод защиты программных средств на основе запутывающих преобразований: дис. … канд. техн. наук: 05.13.19. Томськ, 2006. 121 с.
14. Зворотна розробка. URL: https://uk.wikipedia.org/wiki/Зворотна_розробка (дата звернення: 25.03.2021).
15. Система програмного захисту додатків від несанкціонованого копіювання Asprotect. URL: https://jak.koshachek.com/articles/asprotect-32-sistema-programnogo-zahistu-32-bitnih.html (дата звернення: 25.03.2021).
16. Дшхунян В.Л., Шаньгин В.Ф. Электронная идентификация. Бесконтактные электронные идентификаторы и смарт-карты. М.: «НТ Прес», 2004. 695 с.
17. GMP Install Instruction for Windows Platform. URL: http://cs.nyu.edu/exact/core/gmp/ (дата звернення: 25.03.2021).
18. RSA class. URL: http://code.google.com/p/phpjsrsa/source/browse/trunk/rsa.class.php?r=6 (дата звернення: 25.03.2021).

19. Студопедія. URL: https://studopedia.su/15_3511_zashchita-po.html (дата звернення: 25.03.2021).
20. Wikipedia. URL: https://ru.wikipedia.org/wiki/Идентификация (дата звернення: 25.03.2021).
21. Wikipedia. URL: https://ru.wikipedia.org/wiki/Авторизация (дата звернення: 25.03.2021).
22. Авторизация пользователя. URL: https://dic.academic.ru/dic.nsf/business/17457 (дата звернення: 25.03.2021).
23. Метод авторизации через интернет для защиты shareware программ. URL: http://irbis-nbuv.gov.ua/cgi-bin/irbis_nbuv/cgiirbis_64.exe?C21COM=2&I21DBN=UJRN&P21DBN=&IMAGE_FILE_DOWNLOAD=1&Image_file_name=PDF/Vejpte_2014_4(2)__4.pdf (дата звернення: 25.03.2021).
24. MD5. URL: https://ru.wikipedia.org/wiki/MD5 (дата звернення: 25.03.2021).
25. Ivanov V., Baturin O., Lyfar V., Mytrokhin S., Lyhina L. Construction of Methods for Ensuring the Required Level of Safety Integrity in the Automated Systems of Control Over Technological Processes. *Eastern-European Journal of Enterprise Technologies*. 2019. N 2 (102). P. 70–78.

Стаття надійшла до редакції 25.03.2021