43. ПІДХОДИ ЗАХИСТУ ПРОГРАМНИХ ПРОДУКТІВ ВІД НЕСАНКЦІОНОВАНОГО ДОСТУПУ

Г.В. Олійник
Національний університет харчових технологій

В наш час необхідність використання систем захисту програмного забезпечення обумовлена рядом чинників, серед яких слід виділити: незаконне використання алгоритмів, що є інтелектуальною власністю автора, несанкціонований доступ, використання і модифікація, нелегальне розповсюдження та збут програмного забезпечення [1].

У зв'язку з цим активно створюються і розвиваються системи захисту програмних продуктів. Проте, разом з цим також удосконалюються методи подолання такого захисту. Програмний продукт, захищений за допомогою популярних засобів, таких як Extreme Protector, Armadillo, значно збільшується за розміром, стає нестабільним, виникають конфлікти, чисельні помилки, різко втрачається швидкодія [2].

Для забезпечення захисту програмного продукту, без ризику зниження надійності роботи, запропоновано методику, що базується на використанні відомих методів.

До основних використаних методів захисту програмного коду належать: антивідлагоджування — захист від засобів відлагоджування та дослідження; шифрування — криптографічне перетворення вихідних текстів на різних етапах експлуатації; обусфікація — внесення заплутуючих перетворень; поліморфність — генерування різних версій бінарного коду одного алгоритму.

Важливим та найпершим кроком при створенні системи захисту програмного продукту є ускладнення та мінімізація можливості відновлення вихідного програмного коду при її дослідженні, а також недопустимість простого доступу до контрольних параметрів, встановлення своїх значень змінним, блокування роботи певних програмних блоків. Адже призначені для рішення таких задач інтерактивні програмні засоби, наприклад, IDA PRO, дозволяють не тільки отримати повний доступ, а й точно відновити алгоритми роботи та внести будь-які зміни.

Дослідження програмного продукту може відбуватися у статичному або динамічному режимі. Суть першого зводиться до дослідження програмного коду, для отримання якого вихідний програмний модуль дизасемблюють. Динамічний

режим передбачає трасування програми, тобто виконання з використанням спеціальних засобів, які включають можливість покрокового режиму, отримання доступів до регістрів, областей пам'яті, виконання зупинки програми за певною адресою тощо.

Найефективніший спосіб боротьби з дизасемблером — шифрування [3]. Причому шифрування динамічне: невеликі частини програмного коду розшифровуються за мірою необхідності, а після виконання знову зашифровуються. Для підвищення безпеки при реалізації даного методу необхідно використовувати декілька криптостійких алгоритмів шифрування, віртуальні процедури розшифрування, визначення ефективних адрес, що вимагає значних часових витрат.

Технологія поліморфності дозволяє проводити перетворення з бінарним кодом об'єкту, що захищається, шляхом вбудовування неінформативних для загального алгоритму програми частинами між основними інструкціями. Даний процес поділяється на два етапи: аналіз коду; генерація інструкцій на основі проведеного аналізу. Для аналізу бінарного коду необхідний певний рівень абстракції від машинного коду до людини, досяжний з використанням методики його декомпіляції. Для розробки поліморфного генератора, здатного до вбудовування в довільну ділянку коду, необхідно проводити не тільки облік задіяної пам'яті та регістрів, але й аналіз поточної ділянки коду.

Найчастіше таку технологію для захисту використовують поліморфні комп'ютерні віруси, для виявлення та дослідження яких, як правило, використовуються спеціально розроблені для кожного окремого коду алгоритми аналізу. Виявлення таких вірусів за допомогою так званих вірусних масок — специфічних ділянок постійного коду, неможливе. Це досягається шифруванням основного коду вірусу з непостійним ключем і випадковим набором команд розшифровувача або заміною самого виконуваного коду вірусу.

Суть методу, що ґрунтується на використанні самоідентифікуючих алгоритмів, полягає в тому, що виконувані програмні коди отримуються самою програмою безпосередньо у процесі її виконання. Незважаючи на деякі труднощі розробки, частково пов'язані з неможливістю реалізації певних управляючих процедур мовами програмування високого рівня, даний метод неодноразово продемонстрував свою ефективність [3].

Проведення практичних випробувань з використанням поєднання елементів розглянутих методів захисту дає змогу стверджувати про створення гнучкого, ефективного та надійного захисту програмного продукту.

ЛІТЕРАТУРА

1. *Золотарев В.В.* Метод исследования программных средств защиты информации на основе компонентной модели информационной среды [Текст] / В.В. Золотарев // Изв. ЮФУ: Технич. науки. — 2008. — № 8. — С. 87 – 94.

2. *Кукарцев А.М.* Методика защиты программного кода от несанкционированной модификации и исследования посредством его хеширования [Текст] / А.М. Кукарцев, И.А. Лубкин // Вестн. СибГАУ. — 2008. — № 1. — С. 56 – 60.

3. *Проскурин В.Г.* Защита программ и данных [Текст] / В.Г. Проскурин. — М: Академия, 2011. — 208 с.

Науковий керівник: С.В. Грибков