УДК 004.056

О. В. Гнатюк

ОБФУСКАЦІЯ ЯК ОДИН З МЕТОДІВ ЗАХИСТУ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Вінницький національний технічний університет;

Анотація

У статті здійснено наведено результати дослідженняз одного з методів захисту програмного забезпечення – обфускації. Розглянуто деякі види затемнення, які відрізняються один від одного способом дії. Визначено переваги та слабкі місця даного методу захисту.

Ключові слова: обфускація, захист програмного забезпечення, атаки з боку зловмисного хосту.

Abstract

The article presents the results of research a method of software protection OJEC chenie - obfuskatsiyi. We consider certain types of eclipses, which differ from one mode of action. Objectified cheno advantages and disadvantages of this method of protection.

Keywords: obfuscation, protection software, attack from abuse intentional host.

В даний час досить актуальною є проблема захисту програмного забезпечення. Даючи користувачам доступ до файлів встановлення програм, компанії неминуче розкривають свої професійні секрети і напрацювання, і ніщо не зупиняє конкурентів від копіювання та крадіжки чужих алгоритмів, а так само дослідження програми заради її злому. У більшості випадків для обходу захисту зловмиснику потрібно вивчити принцип роботи його коду, і те, як він взаємодіє з програмою, що підлягає захисту, цей процес вивчення називається процесом реверсивної (зворотної) інженерії.

Обфускація ("obfuscation" – заплутування) – це один з методів захисту програмного коду, який дозволяє ускладнити процес реверсивної інженерії коду. Суть процесу обфускації полягає в тому, щоб заплутати програмний код і усунути більшість логічних зв'язків в ньому, тобто трансформувати його так, щоб досліджуваний код був дуже важкий для вивчення і модифікації стороннім и особами (будь то хакери, або програмісти, які збираються дізнатися унікальний алгоритм роботи захищуваної програми) [1].

Доцільним є використання обфускації для захисту програмного забезпечення від зловмисного хоста. Це галузь, в якій вся міць шифрування виявляється повністю беззахисною перед реверсивною інженерією. Єдиним способом, який міг би ускладнити зворотний аналіз, є заплутування семантики, приховування логіки роботи програми з тим, щоб перешкодити атакуючому зрозуміти особливості її роботи.

«Заплутування» коду може здійснюватися на декількох рівнях: на рівні алгоритму, на рівні сирцевого (вихідного) тексту та/або на рівні асемблерного тексту.

Користувачу сирцевий код програми може бути доступний на JavaScript, VBScript і подібних скрипт-мовах. В цьому випадку форматуванням тексту й заміною імен можна зробити текст менш читабельним.

Зазвичай обфускація на рівні машинного коду зменшує швидкість виконання і, відповідно, збільшує час виконання програми. Тому вона застосовується в критичних для безпеки, але не критичних для швидкості місцях програми, таких як, наприклад, перевірка реєстраційного коду, серійного номеру продукту. Найпростіший спосіб обфускації машинного коду – вставлення в нього недіючих конструкцій.

На рівні проміжного коду обфускацію застосовують в таких мовах програмування, як Java, NetP і інших мовах платформи. NET. На відміну від таких мов, як C++ і Pascal, які компілюють в машинний код, ці мови компілюють сирцевий код в проміжний код (байт-код), який містить достатньо інформації для відновлення сирцевого коду. З цієї причини для цих мов застосовується обфускація проміжного коду.

Перетворення, що заплутують, можна розділити на декілька груп залежно від того, на трансформацію якого з компонентів програми вони націлені.

1) *Перетворення форматування*, які змінюють тільки зовнішній вигляд програми. До цієї групи належать перетворення, що видаляють коментарі, відступи в тексті програми або перейменовують ідентифікатори. Цей вид перетворень є найпростішим і в той же час забезпечує найменший захист.
2) *Перетворення структур даних*, які змінюють структури даних, з якими працює програма. До цієї групи належать, наприклад, перетворення, що змінюють ієрархію успадкування класів в програмі, або перетворення, що об'єднують скалярні змінні одного типу в масив.
3) *Перетворення потоку управління програми*, які змінюють структуру її графа потоку керування. До таких перетворень відносяться такі як внесення недосяжного коду, мертвого коду (код в програмі виконується, але його виконання ніяк не впливає на результат роботи програми), надлишкового коду (він виконується, і результат його виконання використовується в подальшому в програмі, але такий код можна спростити або зовсім видалити, оскільки обчислюється або константне значення, або значення, обчислене раніше), переплетення та клонування функції [2].
4) *Превентивні перетворення*, націлені проти певних методів декомпіляції програм або ті, що використовують помилки в певних інструментальних засобах декомпіляції [3].

До недоліків обфускації відносять втрату гнучкості коду, тобто код після обфускації може стати більш залежним від платформи або компілятора, та ускладнення в процесі знаходження та усунення вад програмного засобу.

Істотним плюсом заплутування є його повна автономність. Заплутування оперує тільки з виконуваним кодом і не вимагає для своєї роботи ніяких додаткових модулів (як програмних, так і апаратних).

Однак, заплутування не є універсальним методом. Неможливо побудувати перетворення, яке буде ефективно заплутувати будь-яку вхідну програму. Проте, на практиці для більшості програмних засобів можна побудувати заплутуючі перетворення, які будуть забезпечувати досить високий рівень захисту [4].

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Обфускация и защита программных продуктов[Електронний ресурс]. – Режим доступу : URL : http://citforum.ck.ua/security/articles/obfus/– Назва з екрану.
2. C. Collberg, C. Thomborson, D. Low. A Taxonomy of Obfuscating Transformations. Department of Computer Science, The University of Auckland. – 1997. http://www.cs.arizona.edu/~collberg/Research/Publications/CollbergThomborsonLow97a.
3. Чернов А.В. Анализ запутывающих преобразований программ. Труды Института Системного программирования РАН [Електронний ресурс]. – Режим доступу : URL : http://citforum.ck.ua/security/articles/analysis/ – Назва з екрану.
4. А. В. Анисимов, И. Ю. Иванов Подходы к защите программного обеспечения от атак злонамеренного хоста. – 2005. – С. 49-50.

Ольга Володимирівна Гнатюк – студентка групи БС-14 б, факультет інформаційних технологій та комп'ютерної інженерії, Вінницький національний технічний університет, Вінниця, e-mail: olichka05.02@mail.ru.

Науковий керівник: ***Каплун Валентина Аполінарівна*** – старший викладач кафедри захисту інформації, Вінницький національний технічний університет, м. Вінниця.

Olha V. Hnatiuk – Department of Information technology and computer engineering, Vinnytsia National Technical University, Vinnytsia, e-mail: olichka05.02@mail.ru.

Supervisor: ***Valentyna A. Kaplun*** – Lecturer of the Chair of Safety of Infomation and Communication Systems, Vinnytsia National Technical University, Vinnytsia.