УДК 681.3.06

К.т.н., доцент Замятін Д.С., магістрант Ковальчук О.М.

Національний технічний університету України
«Київський політехнічний інститут»

МЕТОД МОДИФІКАЦІЇ АЛГОРИТМУ ВОНГА ДЛЯ ПІДВИЩЕННЯ СТІЙКОСТІ ПРОГРАМНОГО КОДУ ДО ЗЛОМУ

Вступ

На сьогодні, в зв'язку з поширенням інформаційних технологій в суспільстві, набуває актуальності проблема захисту інтелектуальної власності програмного коду. Основною метою захисту є: захист від нелегального використання, приховання авторських алгоритмів, протидія протиправному втручанню в код програмного засобу. Одним з перспективних напрямків є побудова обфускаторів [1] – програм, що ускладнюють розуміння початкового представлення вхідної програми, лишаючи її функціональність при цьому незмінною.

Постановка задачі

Задачу можна сформулювати наступним чином: потрібно розробити алгоритм, який, отримуючи на вхід деяку програму P, трансформував би її в програму P'. При цьому повинні виконуватися три умови: 1) збереження семантики: $-\forall\vec{x}_i \in X$ повинна виконуватися рівність $P(\vec{x}_i) = P'(\vec{x}_i)$, де X - множина допустимих вхідних даних, $\vec{x}_i$ - вектор вхідних значень; 2) обмеження на збільшення кількості ресурсів – розміру коду, пам'яті та швидкодії трансформованої програми в порівнянні з вхідною; 3) стійкість до злому – збільшення складності програми, приховання внутрішньої логіки. На вході та на виході обфускатора програма може бути в різних поданнях.

Аналіз існуючих підходів до обфускації

Обфускація досягається за рахунок ускладнення розуміння початкового коду, заплутування і усунення логічних зв'язків в коді. Таке перетворення залишає функціональність програм незмінною і відкриває широкі можливості застосування обфускації в криптографії та інформаційному захисті. Вперше про криптографічні програми обфускації було згадано в праці [2]. Лише через 20 років у праці [3] було проведено перше докладне дослідження проблеми обфускації програм. Проте,

виявилось, що техніка обфускації програм може бути також використана і в зловмисних цілях для розробки поліморфних вірусів, а також для приховання плагіату програмного забезпечення.

Вперше строге математичне визначення стійкості обфускації програм було запропоноване у праці [1]: обфускація вважається стійкою, якщо будь-який зловмисник може отримати з тексту обфускованої програми за розумний (поліноміальний) час не більше інформації, ніж можна було б отримати, проводячи тестові випробування цієї програми як «чорної скриньки». У цій же праці було доведено існування таких програм, для яких подібна стійкість обфускації в принципі недосяжна.

Всі методи обфускації можна розділити на три види (рис. 1): лексична (зміна зовнішнього вигляду коду програми для ускладнення сприйняття аналізованого коду), обфускація потоку даних (приховання залежності за даними між змінними та функціями програми) і обфускація потоку керування програми (приховання найбільш істотних індивідуальних особливостей програми). Особливий інтерес становить третій вид алгоритмів, що не фіксує конкретного подання програми. У даному випадку можна розглядати програму просто як функцію, що змінює вхідні дані на вихідні.

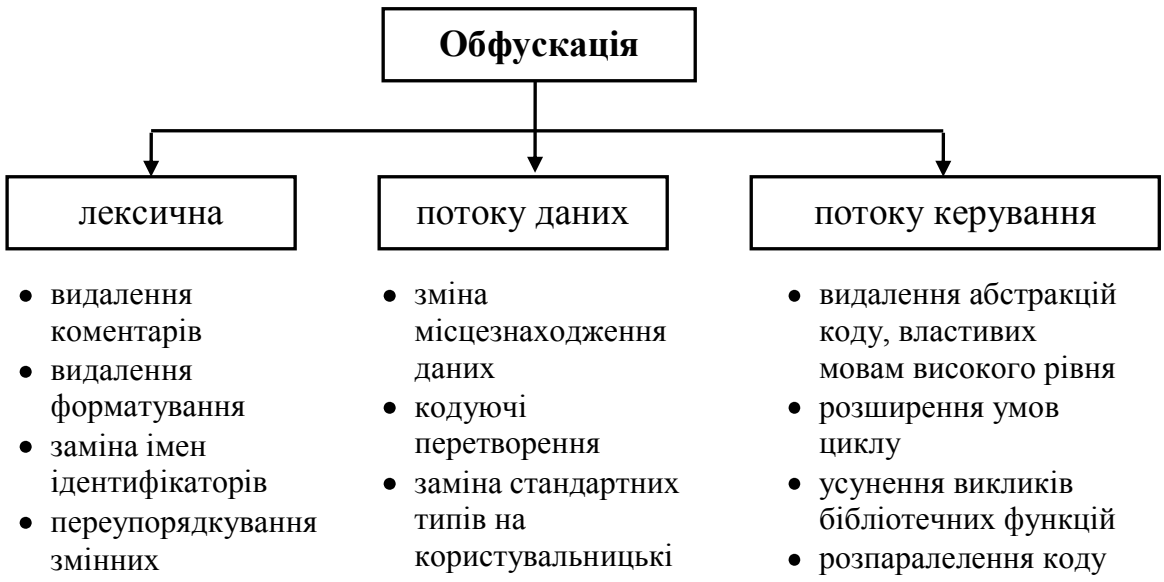

Рис. 1. Класифікація методів обфускації

Опис реалізації

Необхідну послідовність кроків для вирішення поставленої задачі можна розглянути на прикладі алгоритму Колберга (рис. 2), який вважається узагальненим алгоритмом здійснення обфускації. Він не визначає конкретних методів, що повинні бути використанні, а оперує такими загальними поняттями як: вхідна програма *P*; набір стандартних бібліотек *L*, які вона використовує; набір трансформуючих процесів *T*;

визначений фрагмент коду *S*, який витягується з програми *P*, та який безпосередньо буде трансформованим; набір функцій, які будуть визначати ефективність використання визначених трансформуючих процесів *E*; набір функцій, котрі визначатимуть важливість деяких фрагментів коду *I*, що в свою чергу будуть регулювати допустиме збільшення витрати системних ресурсів та необхідний рівень стійкості обфускації.

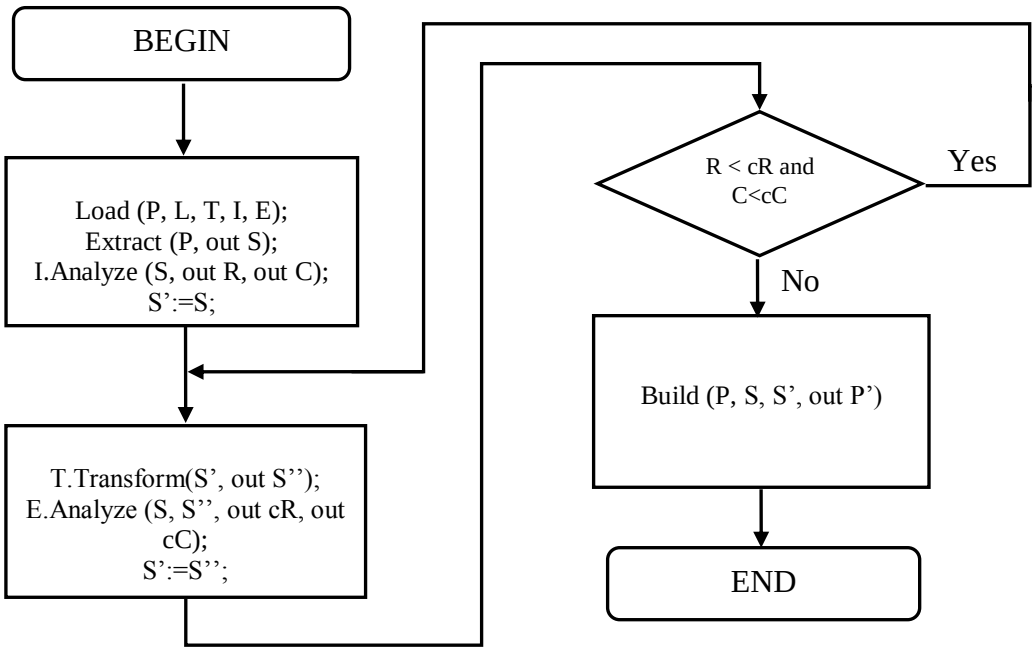

Рис.2. Блок-схема алгоритму Колберга

Одним з прикладів обфускації потоку керування є алгоритм Ченсі Вонга [4]. Він складається з послідовності етапів, що трансформує граф потоку керування до плоского виду. При цьому управління від одного до іншого базового блоку передається не безпосередньо, а через спеціальну змінну, в якій зберігається адреса наступного блоку. Потенціал такого перетворення полягає в тому, що зовнішній вигляд програми істотного змінюється: цикли, умовні переходи вхідної програми набувають іншого вигляду. Через потенційне проходження будь-якого базового блоку як наступного блоку графа потоку керування аналіз потоку даних стане неточним. Дане перетворення, по суті, переводить задачу аналізу графа потоку керування в задачу аналізу потоку даних, яка в загальному випадку алгоритмічно є нерозв'язуваною. Недоліком описаного алгоритму є низька стійкість результату, оскільки в даному випадку «диспетчером» є проста змінна.

Для підвищення стійкості алгоритму Ченсі Вонга пропонується за допомогою перетворення формування диспетчера досягти того, щоб задача статичного пошуку вхідного порядку послідовності базових блоків стала обчислювально-складною. Можна використати підхід включення

обчислювально-складної задачі в диспетчер графа потоку керування. У простому випадку диспетчер можна розглядати як детермінований скінчений автомат. Для збільшення складності його аналізу запропоновано збільшити простір станів диспетчера. Таким чином, результуючий автомат можна отримати прямим добутком автомата переходів вхідної процедури, яка піддається обфускації, та іншого достатньо великого скінченого автомата диспетчера. Задача досягнення деякого стану для такого диспетчера виявляється PSPACE-повною. Задача усунення надлишкових інструкцій з обфускованної програми виявляється PSPACE-складною. При практичній реалізації обфускатора досягнення необхідного рівня стійкості обфускації можна розглядати як задачу побудови заданого рівня складності графа – диспетчера та як задачу внесення надлишкових інструкцій в граф основної програми.

Висновки

Таким чином, проведені дослідження дозволяють зробити висновок, що методи обфускації перетворення графа потоку виконання команд є досить ефективними для захисту програмного коду від злому. Запропонована модифікація алгоритму Ченсі Вонга дозволяє досягти значного підвищення стійкості проти методів автоматичної деобфускації.

Паралельно обфускації програм вирішується суміжне питання: як організувати таке середовище, в якому можна було б виконувати програми без побоювання, що їх внутрішня структура буде вивчена і піддана зміні або витягуванню даних.

Література

1. *Barak B., Goldreich O., Impagliazzo R., Rudich S., Sahai A., Vedhan S., Yang K.* On the (Im)possibility of obfuscating programs // Lecture Notes in Computer Science, v. 2139, 2001. – P. 1-18.
2. *Diffie W., Hellman M.* New directions in cryptography // IEEE Transactions on Information Theory, IT-22(6), 1976. – P.644-654.
3. *Collberg C., Thomborson C., Low D.* A taxonomy of obfuscating transformations // Tech. Report, N 148, Univ. of Auckland, 1997. – P.26-29.
4. *Wang C., Hill J., Knight J,. Davidson J.* Software tamper resistance: obstructing static analysis of programs // Tech. Rep., N 12, Dep. of Comp. Sci., Univ. of Virginia, 2000.– P. 5-7.