УДК 004.63

Ярова М. С.
Карпінець В. В.

МЕТОДИКА ЗАХИСТУ ВИХІДНОГО ПРОГРАМНОГО КОДУ ВІД НЕСАНКЦІОНОВАНОГО КОПІЮВАННЯ З ВИКОРИСТАННЯМ МАТЕМАТИЧНОЇ МОДЕЛІ ЛЕВЕНШТЕЙНА

Вінницький національний технічний університет

Анотація

У даній роботі розглянуто основні принципи та структуру методики захисту вихідного програмного коду від несанкціонованого копіювання за допомогою порівняння версій файлів з використанням методів динамічного програмування та аналізу подібності стрічок на основі показника відстані Левенштейна.

Ключові слова: *відстань Левенштейна, редакційна відстань, динамічне програмування, порівняння версій*

Abstract

This paper considers the basic principles and structure of the method of protecting the source code from unauthorized copying by comparing file versions using dynamic programming methods and analysis of tape similarity based on the Levenstein distance.

Keywords: *Levenstein distance, editorial distance, dynamic programming, version comparison*

Вступ

Питання захисту авторського права на програмний продукт та протидії загрозам компрометації розробки з боку зловмисника все частіше постають перед розробниками програмного забезпечення. Викрадений код піддається видозміні з боку злочинця і надалі доведення автентичності та авторства розробки є непростою задачею. Зокрема, у європейській судовій системі не вдасться довести авторство вихідного коду, якщо його було піддано рефакторингу – абсолютно ідентичні послідовності коду з перейменованими змінними вважаються різними вихідними кодами. Тож, існує потреба механізмах, що дозволяють проводити розслідування випадків порушення авторських прав на програмну розробку шляхом порівняння версій вихідного коду. Метрикою в такому дослідженні виступає редакційна відстань або відстань Левенштейна.

Методика порівняння версій файлів з використанням динамічного програмування на основі відстані Левенштейна

Сучасні системи контролю версій (SVN, Mercurial, Git), в першу чергу, базуються на алгоритмі пошуку найдовшої співпадаючої послідовності (LCS, Longest common Subsequence). Найдовша подібна послідовність символів двох файлів вважається незмінною частиною, а решта ділянок є видаленими (якщо належать оригінальній версії) і доданими (якщо належать оновленій версії). Такі алгоритми досить вибагливі в контексті затрат пам’яті, тому значно зручніше в якості одиниць порівняння використовувати абзаци. Їх можна порівнювати безпосередньо, по стрічках, або ж застосувати до абзацу хешування і порівнювати уже результати хеш-функцій. Однак, оскільки неможливо виключити імовірність появи колізій при хешуванні, при співпадінні хешу необхідно повернутись до порівняння стрічок безпосередньо. [1]

Складнішою є реалізація порівняння змінених ділянок, що присутні в обох версіях файлу, що порівнюються. У цьому випадку розраховують оптимальну відповідність для абзаців зміненої ділянки, для того щоб визначити їх відповідність версіям. Для оцінки схожості кожної пари абзаців можна застосувати алгоритм розрахунку так званої редакційної відстані або відстані Левенштейна. Це метрика, що вимірює різницю між двома послідовностями символів. Вона визначається як мінімальна кількість односимвольних операцій вставки (додавання одного символа), видаленні (видалення одного

символа) та заміни (заміна одного символа іншим), які необхідно виконати для перетворення однієї послідовності символів в іншу. Кожна з цих операцій додає одиницю до відстані. Виконуючи ці три операції, алгоритм намагається змінити першу стрічку так, щоб вона відповідала іншій. [2]

Після обрахунку редакційної відстані застосовується метод динамічного програмування. Перадбачається визначення оптимального значення відповідності між абзацами зміненої ділянки.

1. Ініціалізується матриця, що вимірює в комірці (m,n) відстань Левенштейна між префіксом m одного слова та префіксом n іншого слова. Матрицю можна заповнювати з лівого верхнього до правого нижнього кута.

2. Кожен перехід по горизонталі чи вертикалі відповідає вставці чи видаленню відповідно. Вартість кожної операції встановлюється рівною 1.

3. Діагональний стрибок може вартувати 1, якщо два символи в стрічці і в стовпці не співпадають, або 0, якщо вони співпадають. Кожна комірка завжди локально мінімізує вартість.

4. В результаті, число яке отримуємо в правому нижньому куті матриці і є відстанню Левенштейна між обома словами. [3]

В контексті розробки програмних механізмів для визначення відстані між окремими словами чи абзацами за алгоритмом Левенштейна відмічають деяку недоліки у порівнянні з іншими математичними моделями непарного пошуку: при перестановці місцями слів та частин слів отримують порівняно великі відстані; відстань між абсолютно різними короткими словами виявляється невеликою, в той час як відстань між дуже схожими довгими словами виявляється досить значною. [4]

При визначенні результатів динамічного програмування найменш оптимальними вважаються результати обрахунків, при яких всі абзаци зміненої ділянки у старій версії вважаються видаленими, а всі абзаци у новій версії – доданими, тобто жодна стрічка не відповідає жодній стрічці. Решта варіантів розміщення будуть вважатись більш оптимальними.

Далі, визначивши відповідність абзаців версій файлу можна застосувати посимвольний алгоритм порівняння для точного обрахунку змін всередині абзацу.

Висновки

У роботі представлено дослідження методики порівняння версій файлів при проведенні розслідувань випадків реалізації загроз несанкціонованого копіювання вихідного коду програм з подальшим застосуванням рефакторингу. Визначено, що в основі більшості сучасних систем контролю версій лежать алгоритми пошуку найдовшої співпадаючої послідовності та пошуку оптимальної відповідність для абзаців зміненої ділянки. Другий етап полягає в обрахунку засобами динамічного програмування метрики схожості пари стрічок – відстані Левенштейна, яка визначає мінімальну кількість односимвольних операцій необхідних для перетворення однієї послідовності символів у іншу. Менша редакційна відстань між порівнюваними версіями свідчить про максимальне запозичення між ними.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. О подходах к сравнению версий файлов *PVSM* : веб-сайт URL: https://www.pvsm.ru/razrabotka/6099#begin (дата звернення: 22.02.2021).
2. String similarity — the basic know your algorithms guide! *ITNEXT* : веб-сайт URL: https://itnext.io/string-similarity-the-basic-know-your-algorithms-guide-3de3d7346227 (дата звернення: 22.02.2021).
3. The Levenshtein Algorithm *CUELOGIC* : веб-сайт URL: https://www.pvsm.ru/razrabotka/6099#begin (дата звернення: 22.02.2021).
4. Расстояние Левенштейна *Википедия* : веб-сайт URL: https://cutt.ly/UlceZGd (дата звернення: 22.02.2021).

Ярова Марія Сергіївна – студентка групи УБ-17б, Факультет менеджменту і інформаційної безпеки, Вінницький національний технічний університет, Вінниця, e-mail: fm.ub17b.yarova@gmail.com

Науковий керівник: ***Карпінець Василь Васильович*** — кандидат технічних наук, доцент, завідувач кафедри менеджменту та безпеки інформаційних систем, Вінницький національний технічний університет, Вінниця.

Yarova Maria S. - student of UB-17b group, Faculty of Management and Information Security, Vinnytsia National Technical University, Vinnytsia, e-mail: fm.ub17b.yarova@gmail.com.

Supervisor: ***Karpinets Vasyl V.*** — Ph. D., assistant professor, Head of the Department of Management and Security of Information Systems, Vinnytsa National Technical University, Vinnytsia.