УДК 519.722
DOI https://doi.org/10.32782/2663-5941/2023.2.1/29

Макарова Л.М.
Національний університет кораблебудування імені адмірала Макарова

Камінський С.С.
Національний університет кораблебудування імені адмірала Макарова

Бризгалов М.В.
Національний університет кораблебудування імені адмірала Макарова

РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ ЗНАХОДЖЕННЯ ВНЕСЕНИХ ЗМІН В КОД ВИКОНУВАНИХ ФАЙЛІВ

Метою даної роботи є розробка програмного забезпечення для знаходження внесених змін в код виконуваних файлів. Відомо що, розвиток сучасних комп'ютерних технологій породжує безліч нових можливостей та оптимізації звичайних процесів. Однак в основі кожного процесу лежить передача даних і виконання коду на обчислювальній машині, що включає в себе комплексні процеси перетворення коду в машину мову. Чим більшими і складнішими стають ці системи, тим вище стають вимоги до безпеки на кожному етапі розвитку і тим більше потенційних слабкостей у них виявляється. Однією з таких вразливостей є ін'єкція коду у виконуваний файл. Ці випадки можуть призвести до величезних втрат, потенційної небезпеки втрати важливих даних та їхнього потрапляння до третіх осіб.

Досліджено різні метрики для визначення авторства коду програмного забезпечення, отже, для виявлення впровадження шкідливого коду. У статті розглянуто один із способів вирішення цієї проблеми, а саме підрахунок метрик вихідного або виконуваного програмного коду. Визначено кількісну метрику ентропію, яка добре підходить для перевірки авторства програмного коду та показано, що використовуючи дану метрику можна з високою ймовірністю стверджувати, написана нова ділянка коду розробником програми чи ні.

Результатом роботи є програмне забезпечення для знаходження внесених змін в код виконуваних файлів DisEn, побудоване на .NET Framework з використання мови програмування C#, яке дозволяє перевіряти виконувані файли на наявність змін і допомагає користувачам визначити, чи був файл змінений автором або шкідливим програмним забезпеченням. Основними функціональними можливостями програми DisEn є: дизасемблювання виконуваних файлів для отримання асемблерного коду, обчислення значення ентропії для кожної команди, зіставлення даних із попередньою версією файлу (за наявності), аналіз отриманих даних з точки зору різниці ентропії, відображення отриманих даних у вигляді таблиць та графіків, збереження зліпків файлів за іменами для подальшого порівняння на предмет авторства змін.

Ключові слова: *ентропія, дизасемблювання, виконуваний файл, зміна коду, програмний додаток.*

Постановка проблеми. Наявність вірусів або інших шкідливих модифікацій у програмному коді становить серйозний ризик для безпеки комп'ютерних систем. Якщо ці зміни не виявити і не усунути, вони можуть поставити під загрозу конфіденційність, цілісність і доступність системних ресурсів. Це може призвести до катастрофічних наслідків, таких як крадіжка персональних даних, фінансове шахрайство або навіть кібератаки на інші системи [1].

Щоб запобігти таким інцидентам, дуже важливо якомога швидше виявляти та усувати вірусні зміни в програмах. Однак це завдання є доволі складним, оскільки зловмисники можуть використовувати різні методи, щоб приховати або замаскувати свої модифікації. Як наслідок, для вирішення цієї проблеми також використовуються різні підходи, такі як аналіз коду, виявлення вторгнень, мережевий моніторинг та інші.

Аналіз коду передбачає перевірку програмного коду на наявність підозрілих паттернів, таких як використання шкідливих функцій або бібліотек. Цей підхід може бути ручним або автоматизованим і може вимагати спеціалізованих навичок та інструментів. Виявлення вторгнень передбачає моніторинг системних подій, таких як зміни файлів, мережеві з'єднання або системні виклики, для виявлення підозрілих дій, які можуть свідчити про зміну вірусом. Цей підхід може використовувати алгоритми, засновані на правилах або машинному

навчанні для виявлення аномалій і сповіщення про них [2].

Моніторинг мережі передбачає аналіз трафіку між програмою та іншими системами для виявлення зловмисних дій, таких як витік даних або командно-контрольна комунікація. Цей підхід може використовувати такі інструменти, як сніфери пакетів, системи запобігання вторгненням або брандмауери для моніторингу та блокування підозрілого трафіку [3].

Загалом, виявлення та усунення вірусних змін у програмах є критично важливим завданням у забезпеченні безпеки комп'ютерних систем. Застосовуючи різні підходи та не втрачаючи пильності, системні адміністратори можуть захистити свої системи від широкого спектру загроз і вберегти їх від шкоди.

Аналіз останніх досліджень і публікацій. Одним із способів виявлення авторства змін у тексті, зокрема, і комп`ютерної програми, є застосування кількісної метрики – ентропії, що добре підходить для визначення авторства програмного коду, тобто, використовуючи дану метрику, можна з високою ймовірністю стверджувати, написана нова ділянка коду автором програми чи є чужою вставкою.

Інформаційна ентропія Шеннона [4] була обрана як основа для розрахунку завдяки тому, що вона дає змогу отримувати кількісні результати, які зручно порівнювати один з одним і, тим самим, визначати авторство зміни файлу.

Елементами для розрахунку метрики виступають асемблерні команди виконуваних файлів. Тож, щоб отримати ці команди, треба спочатку дизасемблювати файл.

В першу чергу були розглянуті такі існуючі рішення для дизасемблювання, як IDA Pro Disassembler [5] та Immunity Debugger [6].

IDA Pro Disassembler [5] – інтерактивний дизасемблер, який широко використовується для реверс-інжинірингу. Він відрізняється винятковою гнучкістю, наявністю вбудованої командної мови, підтримує безліч форматів файлів для великої кількості процесорів і операційних систем.

Immunity Debugger [6] – це новий потужний спосіб написання експлойтів, аналізу шкідливих програм та аналізу бінарних файлів. Він заснований на надійному інтерфейсі користувача з графічними функціями. Розроблений спеціально для створення множин з великим і добре підтримуваним інтерфейсом Python API для легкої розширюваності.

Однак вищеназвані дизасемблери надто важкі і великі. Тому були знайденні більш легкі там швидкі аналоги.

Capstone [7] – це легкий мультиплатформенний фреймворк для дизасемблювання. Його головна особливість – це мультиархітектурність: Arm, Arm64 (Armv8), BPF, Ethereum Virtual Machine, M68K, M680X, Mips, MOS65XX, PowerPC, RISCV, Sparc, SystemZ, TMS320C64X, Web Assembly, XCore & X86 (включаючи X86_64).

Але даний дизасемблер також не відповідає потребам, оскільки в результаті його роботи не повертається повний набір команд. Крім того, він має обмеження на розмір виконуваних файлів та може займати багато часу для обробки файлів розміром більше 5 Мб, а іноді призводить до аварійного завершення програми. Незважаючи на ці недоліки, даний дизасемблер може працювати з будь-яким типом файлів.

Інший дизасемблер, який був розглянутий, є утилітою, що йде разом з Visual Studio 2019. Програма дампу двійкових файлів DUMPBIN. EXE [8] відображає двійкові файли у форматі текстового файлу із командами асемблера. Вона не має обмежень на розмір вхідного виконуваного файлу і швидко їх дизасемблює, але вона може працювати лише с файлами типу .exe. Проте, цей дизасемблер повністю виконує потрібні задачі, тому саме він став основним інструментом дизасемблювання виконуваних файлів.

Постановка завдання. Метою роботи є розробка програмного забезпечення для знаходження внесених змін в код виконуваних файлів на основі обчислення ентропії дизасембльованих виконуваних файлів та порівняння її з попередніми значеннями. Для того, щоб досягти цієї мети, необхідно виконати декілька завдань.

По-перше, необхідно проаналізувати існуючі дизасемблери для виконуваних файлів та визначити найбільш відповідний вимогам роботи.

Після того, як відповідний дизасемблер знайдено, необхідно розробити програму для аналізу дизасембльованого коду. Програма повинна обчислювати метрику ентропії для всіх команд дизасембльованого коду і зберігати попередні значення для порівняння.

Нарешті, дані, отримані в програмі, повинні бути проаналізовані для того, щоб зробити висновки щодо авторства змін, внесених до коду, порівнюючи поточну метрику ентропії з попередніми значеннями: чи були зміни внесені автором, чи кимось іншим.

Виклад основного матеріалу дослідження. Для досягнення мети роботи будемо використовувати значення ентропії [9]. Поняття ентропії вперше було введено Рудольфом Клаузіусом

в термодинаміці для визначення міри зворотного розсіювання енергії. Ентропію найчастіше розуміють як «непотрібну енергію» у системі, тобто енергію, яка не виконує жодної роботи. Вона визначена в термінах теорії ймовірності та використовується для розрахунку міри невизначеності будь-якого досвіду (випробування), який може мати різні результати.

Ентропія – міра невизначеності чи непередбачуваності інформації, невизначеність появи будь-якого символу первинного алфавіту. За відсутності інформаційних втрат ентропія чисельно дорівнює кількості інформації [10]. Наприклад, у послідовності літер, що становлять якусь фразу різними мовами, різні літери з'являються з різною частотою, тому невизначеність появи для деяких літер менша, ніж для інших. Якщо врахувати деякі поєднання літер, які зустрічаються дуже рідко, то невизначеність зменшується ще сильніше. Ентропія також може бути застосована до програмного забезпечення як міра взаємодії. Так, розглядаючи ентропію вихідного коду, який може бути розбитий на дрібніші сегменти (рядки, функції, змінні, команди і т.д.), можна визначати авторство тієї чи іншої частини коду. Таким чином, можна обчислити внесок різних авторів у написану програму. Для розрахунку ентропії в теорії інформації використовується наступна формула [10]:

$$H(i) = -p_i \cdot \log_2(p_i), \qquad (1)$$

де: i – можливі стани,

p_i – ймовірність появи i-го стану.

У випадку з програмним кодом у ролі станів виступають команди, і, відповідно, ймовірність їх появи в коді. Також ентропійний метод використовується для вирішення таких завдань, як пошук зашифрованого або запакованого шкідливого програмного забезпечення.

В результаті роботи було розроблене програмне забезпечення DisEn, яке побудоване на .NET Framework з інтерфейсом WPF з використанням мови програмування C#. Основним завданням цього програмного забезпечення є визначення авторства внесених змін в код виконуваних файлів на основі обчислення ентропії дизасембльованих виконуваних файлів. Це дозволяє користувачам визначити, чи був файл змінений автором або якимось шкідливим програмним забезпеченням [11].

Програма дозволяє дизасемблювати необхідний виконуваний файл, отримуючи таким чином асемблерний код, розрахувати значення ентропії для кожної команди та проаналізувати отримані дані з точки зору різниці ентропії. Ці дані відображаються у відповідній таблиці та на діаграмі у основному вікні програми. Програма зберігає зліпки файлів за іменами для подальшого порівняння на предмет авторства змін.

На вхід програмі подається файл типу .exe, який треба проаналізувати. Алгоритм роботи програми наступний.

1) Дизасемблювання вхідного файлу. Дизасемблювання здійснюється за допомогою утиліти dumpbin.exe [8], яка входить до складу Microsoft Visual Studio. Команда для дизасемблювання: "dumpbin.exe /disasm /out:NAME.txt NAME.exe", де /disasm – опція дизасемблювання; /out:NAME.txt – ім'я вихідного файлу, в який буде записано результат; NAME.exe – ім'я файлу, який необхідно дизасемблювати. В результаті отримуємо файл формату .txt, в якому будемо підраховувати кількість команд. Також є можливість переглянути дизасембльований код програми, який зберігається в текстовому файлі.

2) Розрахунок значення ентропії кожної команди для дизасембльованого файлу. Розрахунок здійснюється за формулою (1). Усі отримані значення ентропії записуються у файл.

3) Розрахунок різниці ентропії. Зчитуємо значення ентропії для кожної команди оригінального файлу та відповідного йому зміненого, підраховуємо різницю для них, записуємо результат у файл.

4) Порівняння різниці ентропії відповідних команд з експериментальними пороговими значеннями для цих команд.

На наступних рисунках можна побачити роботу розробленого програмного забезпечення DisEn. В якості прикладу для аналізу було взято програму для рендерингу 3D-об'єктів, створену авторами цієї статті (перша версія відображає лише каркас, наступна версія реалізувала растеризацію трикутників).

На рис. 1 приведено частину дизасембльованого виконуваного файлу Renderer.exe, результат роботи збережено у файл Renderer.txt.

На рис. 2 приведено результат аналізу дизасембльованого файлу Renderer.txt: розраховано значення ентропії для кожної команди та побудовано відповідну гістограму. Ці дані представлені у двох частинах вікна програми: зліва інформація виводиться у формі таблиці, відображаючи кількість викликів кожної команди і розраховане значення ентропії; праворуч у форматі стовпчатої діаграми зображено значення ентропії для кожної команди.

Програмне забезпечення DisEn зберігає інформацію про попередню версію яку вважаємо еталонною, або зліпок файлу за найменуванням.

Рис. 1. Результат дизасемблювання файлу Renderer.exe

Рис. 2. Результат аналізу дизасембльованого файлу Renderer.txt.

Якщо якийсь файл вже було опрацьовано і буде завантажено новий з такою самою назвою, буде проведено порівняння цих файлів. На основному вікні буде відображено інформацію про значення ентропії команд двох файлів.

На рис. 3 приведено інформацію про дві версії файлу, що аналізується: у верхній частині вікна – про поточну версію, у нижній частині – про попередню.

За необхідності користувач може відкрити додаткове вікно та деталізувати інформацію про різницю в значенні ентропії, кількості команд та розміру файлів. Такий приклад наведено на рис. 4.

Головна мета програмного забезпечення DisEn – це знаходження оптимального способу для розрізнення виконавчих файлів, що змінилися автором або шкідливим програмним забезпеченням. Прикладом того, наскільки файли можуть відрізнятись один від одного, може слугувати порівняння виконавчого файлу гри та інсталятора. На рисунках 5 та 6 можна побачити різницю між цими двома файлами.

Таким чином, на основі такої різниці можна визначати, наскільки виконавчий файл був змінений і чи може це бути авторським втручанням чи шкідливою ін'єкцією коду.

Висновки. Розроблене програмне забезпечення DisEn допомагає знаходити зміни коду виконуваних файлів та робити висновки на предмет авторства цих змін за рахунок аналізу дизасембльованого коду виконуваних файлів, обчислення ентропії та порівняння з попередніми значеннями. Розроблена

Рис. 3. Приклад порівняння двох версій файлу, що аналізується

Рис. 4. Детальна інформація про різницю ентропій між двома файлами

Рис. 5. Приклад порівняння між грою та інсталятором

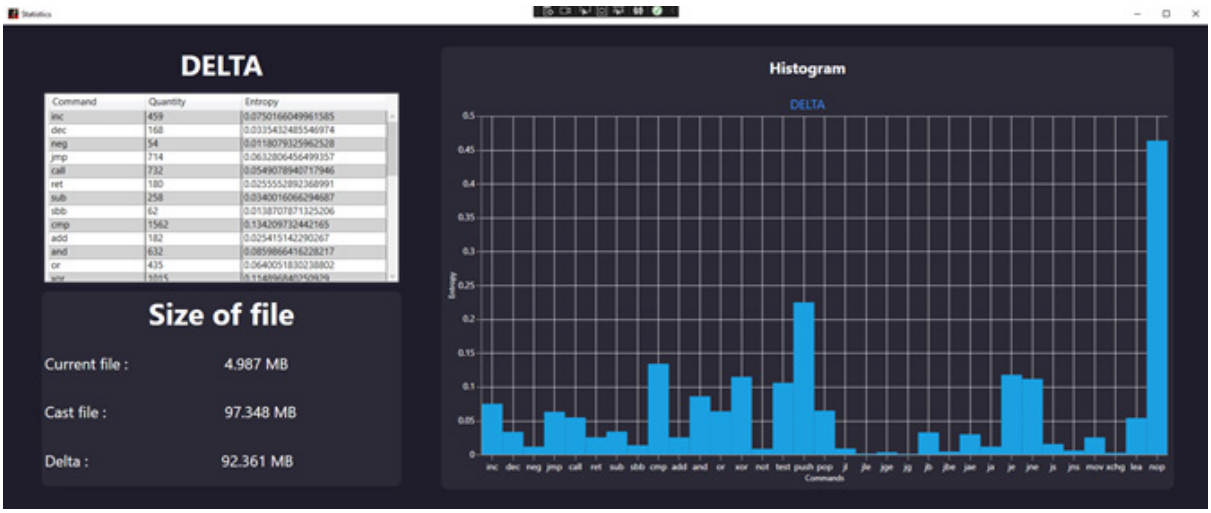

Рис. 6. Приклад детальної інформації про порівняння між грою та інсталятором

програма може бути використана, зокрема, для валідації існуючих файлів. В подальшому передбачається удосконалення розробленої програми за рахунок додавання математичної моделі для прогнозування авторства внесених змін. Також планується реалізувати можливість налаштування списку команд для розрахунку ентропії для більш гнучкого використання програми.

Список літератури:

1. Каплун В.А., Майданюк В.П. Захист операційних систем. Навчальний посібник. Вінниця: ВНТУ, 2006. 180 с.

2. Що таке ін'єкція коду в Windows? URL: https://www.thefastcode.com/uk-uah/article/what-is-code-injection-on-windows (дата звернення 10.11.2022).

3. Бурячок В.Л., Аносов А.О., Семко В.В., Соколов В.Ю., Складанний П.М. Технології забезпечення безпеки мережевої інфраструктури. К.: КУБГ, 2019. 218 с.

4. Shannon C.E., Weaver W. The Mathematical Theory of Communication. The University of Illinois Press, 1971. 144 p.

5. IDA Pro. URL: https://hex-rays.com/ida-pro/ (дата звернення 17.11.2022).

6. Immunity Debugger. URL: https://www.immunityinc.com/products/debugger/ (дата звернення 19.11.2022).

7. Capstone. The Ultimate Disassembler. URL: https://www.capstone-engine.org/ (дата звернення 21.11.2022).

8. Microsoft Visual Studio disassembler dumpbin. URL: https://learn.microsoft.com/en-us/cpp/build/reference/dumpbin-reference?view=msvc-170 (дата звернення 23.11.2022).

9. Энтропия. Как хаос помогает искать вирусы. URL: https://xakep.ru/2021/01/29/viruses-entropy/ (дата звернення 05.11.2022).

10. Шарапов О.Д., Дербенцев В.Д., Семьонов Д.Є. Економічна кібернетика. Навч. посібник. К.: КНЕУ, 2004. 231 с.

11. Макарова Л.М., Камінський С.С., Бризгалов М.В. Використання ентропії для знаходження внесення змін коду виконуваних файлів. *Інформаційні технології: моделі, алгоритми, системи (ITMAS-2022): Матеріали III Всеукраїнської науково-практичної інтернет-конференції* (26-28 жовтня 2022 р.). Миколаїв: НУК імені адмірала Макарова, 2022. С.77-79.

Makarova L.M., Kaminsky S.S., Bryzgalov M.V. DEVELOPMENT OF SOFTWARE FOR DETECTING CHANGES MADE TO THE CODE OF EXECUTABLE FILES

The aim of this work is to develop software for detecting changes made to the code of executable files. It is known that the development of modern computer technologies gives rise to many new opportunities and optimizations of common processes. However, each process is based on data transfer and code execution on a computer, which includes complex processes of converting code into machine language. The larger and more complex these systems become, the higher the security requirements at each stage of development and the more

potential weaknesses they have. One such vulnerability is code injection into an executable file. These cases can lead to huge losses, the potential danger of losing important data and its exposure to third parties.

Various metrics have been studied to determine the authorship of software code, and thus to detect the introduction of malicious code. The article considers one of the ways to solve this problem, namely, counting metrics of source or executable program code. The quantitative metric entropy is defined, which is well suited for verifying the authorship of program code, and it is shown that using this metric it is possible to state with high probability whether a new section of code was written by the program developer or not.

The result of the work is DisEn, a software for detecting changes made to the code of executable files built on the .NET Framework using the C# programming language, which allows you to check executable files for changes and helps users determine whether the file has been modified by the author or malware. The main functionalities of DisEn are: disassembling executable files to obtain assembly code, calculating the entropy value for each command, comparing the data with the previous version of the file (if available), analyzing the data in terms of entropy differences, displaying the data in the form of tables and graphs, saving file copies by name for further comparison for the authorship of changes.

Key words: *entropy, disassembly, executable file, code change, software application.*