УДК 004.519.217

Д. А. МАЕВСКИЙ, Ю. П. ЧЕРБАДЖИ

Одесский национальный политехнический университет, Украина

ОПРЕДЕЛЕНИЕ АВТОРСТВА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ПО ИСХОДНОМУ КОДУ ПРОГРАММ

В работе предложен способ определения авторства программного обеспечения по исходному коду программ. В основе определения авторства лежит специально разработанная система, состоящая из 100 метрик, отражающих «почерк создателя» программного обеспечения. Разработано программное обеспечение, позволяющее автоматизировать подсчет метрик для исходных кодов программ. На основе метрик составлен «профиль почерка» пяти разных программистов по текстам трех разработанных ими программных систем и проверено соответствие этому профилю других программ, написанных как данными, так и другими программистами. Показано, что по разработанному профилю можно достоверно осуществлять идентификацию автора программного обеспечения.

***Ключевые слова**: программное обеспечение, атрибуция текста, почерк автора, метрики авторства, метрики исходного кода.*

Введение

Человечество живет и развивается по своим, еще не познанным законам. Одним из проявлений этих законов есть тот факт, что любое изобретение, изначально направленное на благо всего человечества, рано или поздно начинает использоваться с прямо противоположной целью – для нанесения вреда отдельному человеку. Примеров тому множество.

Открыв на заре своего существования огонь, человек понял, что его можно использовать не только для обогрева своего дома, но и для того, чтобы сжечь дом соседа.

В 1867 году Альфред Нобель изобрел динамит, предполагая использовать его для облегчения горнопроходческих работ. А 28 лет спустя он же, ужаснувшись от применения своего изобретения в военном деле, учредил премию, ежегодно присуждаемую «тем, кто за прошедший год внес наиболее существенный вклад в науку, литературу или дело мира и чья деятельность принесла наибольшую пользу человечеству» [1].

В 1919 году Эрнест Резерфорд открыл ядерные реакции, мечтая поставить атомную энергию на службу человеку. А 26 лет спустя, 6 августа 1945 года атомная бомба «Малыш» («Little Boy») унесла жизни 140 тысяч человек в Хиросиме.

В начале семидесятых годов двадцатого века был создан первый персональный компьютер и начала реализовываться мечта Билла Гейтса - поставить простой и доступный компьютер в каждый дом. А сегодня…

Сегодня компьютерным информационным системам поручают управление различными сложными объектами, зачастую объектами критического назначения. Сегодня, чтобы нарушить электроснабжение целого региона, устроить аварию на атомной электростанции или на химическом комбинате, не обязательно необходимо непосредственное вмешательство человека. Сегодня уже не нужно тратить время и деньги на подготовку диверсантов, переход ими границы и внедрение на соответствующее предприятие. Сегодня для этого достаточно вмешаться в работу программного обеспечения, которое управляет производственными процессами на этом предприятии.

Именно поэтому программное обеспечение, особенно критического назначения, часто становится объектом разнообразных кибер-атак. Так, например, в 2010 году была обнаружена высокоточная «компьютерная бомба» Stuxnet, с помощью которой помощью были физически выведены из строя урановые центрифуги на иранском заводе по обогащению ядерного топлива. Осенью 2011 года обнаружен шпионский вирус Duqu, а весной 2012 иранские инженеры обнаружили атаку на свои нефтеперегонные заводы многокомпонентного шпионского вируса Flame, который собирал и передавал на удаленные сервера техническую документацию [2].

Помимо таких точечных атак на объекты критической инфраструктуры, возможны массированные атаки на жизненно важные объекты целых государств. Вот что пишет по этому поводу бывший советник администрации президента США, специалист по борьбе с терроризмом Ричард Аллан Кларк. «Уязвимость США происходит от её интернет-зависимости в сфере снабжения населения, банков-

ской системы, автоматизации транспортных и энергетических сетей и других современных систем. Угрозу усугубляет и то, что американские вооружённые силы также слепо полагаются на интернет. Американская армия не способна обойтись без интернета в той же мере, как и какая-нибудь торговая сеть, работающая в режиме он-лайн!». А американский журнал Military Review описывает результаты смоделированной с привлечением более 200 специалистов кибератаки на системы жизнеобеспечения восточной части США. «Участникам эксперимента с лёгкостью удалось нарушить работу энергосистемы и телефонной сети страны. В ходе атаки вредоносное мобильное ПО начинало рассылать вирусные программы на другие телефоны, которые также активно заражали друг друга.

Моделирование кибервойны показало, что грамотная атака способна оставить 40 миллионов жителей США, проживающих на востоке страны, без электричества всего через полчаса после её начала. Ещё через час 60 миллионов абонентов сотовой связи с удивлением обнаружат, что их мобильные телефоны превратились в обычные пластмассовые игрушки. А еще через пару часов паралич доберётся и до центра финансового мира – Уолл-стрит». Главной особенностью такой кибервойны является то, что вражеским армиям не надо пересекать границы государства и вести огонь на поражение. Тех же результатов, но гораздо дешевле, быстрее и эффективнее, можно достичь при помощи одного персонального компьютера, распространяющего в сети вредоносное ПО.

Одним из путей предупреждения ситуации, когда вредоносная программа внедряется в информационную систему и захватывает контроль над ее частью или над ее работой в целом, является выявление авторов вредоносного кода с целью пресечения их незаконной деятельности. На сегодняшний день хорошо разработанными могут считаться методы определения авторства текстов на естественных языках. Попытки применения этих методов для искусственных языков, которыми являются языки программирования, не позволяют получить достаточно достоверные результаты. Это связано с тем, что искусственные языки имеют на порядки меньшее количество слов и гораздо более жесткие синтаксические правила конструирования предложений (операторов). Поэтому разработка методов и информационной технологии распознавания авторства (атрибуции) программного обеспечения по исходному коду программ является актуальной.

1. Современное состояние проблемы

Методы определения авторства текстов на естественных языках начали развиваться с тридцатых годов прошлого столетия и в настоящее время считаются хорошо исследованными. Наиболее распространенным является метод атрибуции на основе так называемых «N-грамм», то есть, последовательности из N элементов, которую могут составлять рядом расположенные буквы, слоги или другие лексические единицы изучаемого текста [3]. Сравнивая количество появлений различных N-грамм в авторских и спорных текстах, создатели метода смогли добиться убедительных результатов, когда точность определения авторства составляет порядка 90% и больше. Однако, этот метод имеет один существенный недостаток, заключающийся в том, что основные элементы данного метода открыты для субъективных манипуляций, то есть зная частоту встречаемости N-грамм какого либо автора, легко подделать его стиль. Такая подделка не будет распознана методом.

Для определения авторства текстов на искусственных языках чаще используются методы, основанные на подсчете определенных метрик. Так было предложено использовать метрики, основанные на «хорошем» стиле программирования на языке С [4]. Данные метрики применимы в большинстве своем только для данного языка, что не позволяет его использовать для других популярных языков. Другой метод для языка программирования С++ использует цикломатическую сложность и похожие метрики, в основе которых лежат графы, также некоторое внимание уделяется комментариям [5, 6]. Один из недавно разработанных методов [7] включает в себя 56 метрик, из которых 48 были определены как помогающие идентификации авторства. Метод работает для программ на Java, и его точность составляет порядка 63%. Несмотря на сравнительно низкую точность распознавания, исследование [7] показывает, что метрики исходного кода могут помочь в установлении авторства.

Однако основной проблемой таких метрик является их зависимость от языка программирования. Поэтому целью настоящей работы является построение метрик, максимально адаптированных под особенности любых языков программирования, и сосредоточенных, в основном, на индивидуальной манере написания программ разными авторами.

2. Формирование набора метрик исходного кода

Для достижения поставленной цели было выбрано несколько основных категорий метрик, ориентированных на отражение индивидуальных особенностей написания программ разными программистами. Этому предшествовал подробных анализ особенностей «почерка» пяти различных авторов автора программного обеспечения по разработан-

ным ими 25 программным проектам, написанным на языке Java. В результате были выделены следующие категории и определены составляющие их метрики. Всего выделено пять различных категорий, включающих 100 отдельных метрик.

1. **Ключевые слова**. Данная категория включает в себя 50 метрик по количеству ключевых слов в разных языках программирования. Рассчитывается частота использования различных ключевых слов программистом. С помощью данной метрики можно выявить служебные слова, чаще всего используемые программистом. Так, например, существуют взаимозаменяемые ключевые слова и составленные с их помощью операторы, использование которых не меняет логику работы программы (if/else могут заменяться на switch/case или оператор for на оператор while). Это открывает возможности отслеживания особенностей использования отдельными программистами ключевых слов и операторов языка.

2. **Знаки операций**. Категория вмещает 36 метрик, по количеству знаков операций в наиболее часто используемых языках программирования. Также как и в случае с ключевыми словами, рассчитывается частота использования тех или иных операций.

3. **Разделители**. Данная категория содержит 10 метрик, которые показывают частоту и особенности использования программистом разделителей. Во многих языках программирования такие разделители, как пробел, знак табуляции или перевод на новую строку не являются обязательными. Однако именно разделители формируют «внешний вид» листинга программы и отражают устоявшиеся и трудно меняемые особенности почерка. Эта группа метрик показывает частоту использования программистом отдельных разделителей.

4. **Разделители до и после знаков операций**. Данная категория содержит 4 метрики, показывающих необязательное обрамление знаков операций пробелами или другими разделителями. Рассчитывается частота использования разделителей до и после каждого знака операций. Категория показывает использование разработчиком так называемого «хорошего» стиля программирования.

5. **Имена идентификаторов**. В данную категорию входят метрики, показывающие среднюю длину используемых программистом идентификаторов глобальных и локальных переменных, названий классов и методов. Учитываются также некоторые стилистические особенности использования программистом слов для названий переменных, например, употребляет ли программист русские названия, написанные английскими буквами или нет, использует ли сокращения или полных названий и так далее.

3. Процедура установления авторства

Для автоматизации процедуры установления авторства была разработана специальная программная система, рассчитывающая метрики требуемого исходного кода. Программа может обрабатывать тексты на языках Java, C, C++ и C#. Набор языков может быть неограниченно расширен путем добавления таблиц служебных слов и разделителей любого языка программирования. Для исключения влияния длины исходного кода на результаты, используются относительные значения метрик.

Как уже говорилось, для проверки работоспособности созданных метрик использованы по пять проектов пяти различных авторов. Причем авторство всех пяти проектов было известно абсолютно точно. Все проекты разрабатывались авторами независимо, причем в разработке каждого проекта принимал участие только один автор. Исследование работоспособности предложенных метрик и их способность выявить особенности почерка автора проводилось в три этапа.

На первом этапе группа из пяти проектов каждого автора была разбита на две подгруппы. Три проекта, выбранные случайным образом, образовали подгруппу проектов с известным автором. Эта подгруппа предназначена для построения «профиля автора» по выбранным метрикам. Вторая подгруппа (по два проекта) считалась исследуемой подгруппой для выявления авторства текста программ.

На втором этапе для каждого из авторов составлен «профиль автора» на основе расчета средних значений всех метрик на основе трех проектов автора. Для построения профиля использовалась описанная выше программная система. На втором этапе определялось авторство двух оставшихся проектов в исследуемой группе. Для каждого из этих проектов рассчитывались значения всех метрик, и определялся процент отклонения каждой метрики исследуемого проекта (профиля исследуемого проекта) от соответствующего значения метрики профилей всех пяти авторов. При этом естественно предположить, что средний процент всех отклонений будет минимален только для одного из профилей исследуемых проектов – того, который наиболее близок к профилю одного из авторов. Именно этот минимальный процент отклонений и позволит установить авторство исследуемого проекта.

Таким образом, было проведено 50 сравнений профилей пяти авторов с десятью профилями проектов. Отклонения, полученные для каждого из двух проектов исследуемой подгруппы, усреднялись. Средние проценты отклонений профилей исследуемых проектов каждого из авторов от метрик профилей всех авторов приведены в таблице 1.

В каждой строке этой таблицы приведены средние отклонения по двум известным проектам соответствующего автора от профилей всех авторов. Как видно из таблицы 1, отклонения, стоящие на главной диагонали (отклонения проектов автора от профиля того же автора) всегда являются минимальными, что и позволяет достоверно определить автора каждого из исследуемых проектов.

Таблица 1

Результаты исследований профилей

Профили авторов	Профили исследуемых проектов				
	Авт. 1	Авт. 2	Авт. 3	Авт. 4	Авт. 5
Автор 1	2,01	3,05	4,73	2,64	2,98
Автор 2	2,29	1,97	4,23	2,43	2,5
Автор 3	2,84	2,61	2,16	2,57	2,72
Автор 4	2,83	3,30	4,61	1,97	3,45
Автор 5	3,18	2,78	4,09	3,21	2,69

Таким образом, авторство всех проектов исследуемой подгруппы было установлено абсолютно точно. Поэтому можно утверждать, что предложенная система метрик позволяет установить авторство программного обеспечения по исходному коду программ.

Выводы

В данной работе представлены первые обнадеживающие результаты установления авторства программного обеспечения по исходному коду. Показано, что система метрик исходного кода может учесть особенности «почерка», однозначно указывающие на автора программы. Эти особенности образуют так называемый «профиль» автора, который может быть выражен в числовом виде.

Особенностью предложенной методики является возможность регулярного уточнения профиля автора на основе подсчета метрик тех программных продуктов, для которых его авторство установлено достоверно. Таким образом, профиль автора не остается статичным. постоянно корректируется, то есть, методика предусматривает «обучение».

Перспективным направлением дальнейших исследований является разработка аналогичной методики для так называемых «байт-кодов» («р-кодов») программ, написанных на Java, Java-script и других подобных языках. Ведь структура такого кода практически полностью повторяет структуру исходного кода, за исключением структуры разделителей.

Интересной и важной проблемой является также и установление авторства не по исходному, а по исполняемому, двоичному программному коду – exe-модулей и модулей динамически подключаемых библиотек. Традиционные методы атрибуции здесь работать не могут, так как структура таких двоичных файлов значительно отличается от структуры исходного кода – здесь нет уже не только разделителей, но и идентификаторов, служебных слов и операторов языка высокого уровня. В связи с этим перспективным представляется подход, при котором исследуемый двоичный файл интерпретируется как некоторое черно-белое или цветное изображение, например, как файл, представленный в формате bmp. Профиль автора в этом случае также должен быть представлен в том же формате, что позволит сравнивать двоичные файлы хорошо разработанными методами обработки изображений.

Литература

1. Statutes of the Nobel Foundation [Electronic resource]. – Access mode: http://www.nobelprize.org/nobel_organizations/nobelfoundation/statutes.html. – 3.02.2014.

2. Тарасов, А. М. Кибершпионы: Duqu, Stuxnet, Flame, Gauss: что дальше? [Текст] / А. М. Тарасов // Право и кибербезопасность. – 2012. – № 1. – С. 23–26.

3. Identifying Authorship by Byte-Level N-Grams: The Source Code Author Profile (SCAP) Method [Electronic resource] / G. Frantzeskou, S.G. MacDonell, E. Stamatatos, S. Georgiou, S. Gritzalis // International Journal of Digital Evidence. – Trier, Germany, 2007. – Vol. 6, № 1. — P. 139–148. – Access mode: http://www.utica.edu/academic/institutes/ecii/publications/articles/B41158D1-C829-0387-009D214D2170C321.pdf. – 3.02.2014

4. Krsul, I. Authorship Analysis: Identifying The Author of a Program. [Electronic resource] / I. Krsul, E.H. Spafford // CiteSeerX. – Access mode: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.89.6233&rep=rep1&type=pdf. – 3.02.2014.

5. Gray, A. IDENTIFIED (integrated dictionary-based extraction of non-language-dependent token information for forensic identification, examination, and discrimination): A dictionary-based system for extracting source code metrics for software forensics[Electronic resource] / A. Gray, P. Sallis, S. MacDonell // IEEE Computer Society Press. – Dunedin, New Zealand: 1998. – P. 252 – 259. – Access mode: http://aut.researchgateway.ac.nz/bitstream/handle/10292/3472/Gray%2c%20Sallis%20and%20MacDonell%20%281998%29%20SEEP.pdf?sequence=2. – 3.02.2014.

6. Salis, P. Software Forensics: Old Methods for a New Science. [Electronic resource] / P. Sallis, A. Aakjaer, S. G. MacDonell // IEEE Computer Society Press. – Dundin, New Zealand, 1998. – P. 367-371. – Access mode: http://aut.researchgateway.ac.nz/bitstream/handle/10292/4083/Sallis%2c%20Aakjaer%20and%20MacDonell%20%281996%29%20SEEP.pdf?sequence=2. – 3.02.2014.

7. Ding, H. Extraction of Java program fingerprints for software authorship identification [Electronic resource] / H. Ding, M. H. Samadzadeh // The Journal of Systems and Software. – 2004. – Vol. 72, № 1. – P. 49-57. – Access mode: http://www.sciencedirect.com/science/article/pii/S0164121203000499. – 3.02.2014.

Поступила в редакцию 3.02.2014, рассмотрена на редколлегии 25.03.2014

Рецензент: д-р техн. наук, проф. И. Б. Туркин, Национальный аэрокосмический университет «ХАИ», Харьков, Украина.

ВИЗНАЧЕННЯ АВТОРСТВА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ЗА ПОЧАТКОВИМ КОДОМ ПРОГРАМ

Д. А. Маєвський, Ю. П. Чербаджи

У роботі запропоновано спосіб визначення авторства програмного забезпечення за початковим кодом програм. У основі визначення авторства лежить спеціально розроблена система, що складається з 100 метрик, які відбивають «почерк творця» програмного забезпечення. Розроблено програмне забезпечення, що дозволяє автоматизувати підрахунок метрик для початкових кодів програм. На основі метрик складено «профіль почерку» п'яти різних програмістів по текстах трьох розроблених ними програмних систем і перевірена відповідність цьому профілю інших програм, написаних як цими ж, так і іншими програмістами. Показано, що за розробленим профілем можна достовірно здійснювати ідентифікацію автора програмного забезпечення.

Ключові слова: програмне забезпечення, атрибуція тексту, почерк автора, метрики авторства, метрики початкового коду.

AUTHORSHIP ATTRIBUTION OF SOFTWARE THROUGH SOURCE CODE

D. A. Maevsky, J. P. Cherbadzhi

This paper is connected with a method for determining authorship of software source code programs. At the heart of authorship is specially designed system consisting of 100 metric reflecting "handwriting of creator" of software. Software has been developed to automate the calculation of metrics for program source code in Java. Based metrics compiled "profile handwriting" of five different programmers texts three software systems they have developed and tested for compliance with this profile other programs written data as well as other programmers. It is shown that the developed profile can be reliably carry out the identification of the software author.

Keywords: software, authorship, metrics of code, attribution text, handwriting author`s metrics authorship, source code metrics.

Маевский Дмитрий Андреевич – д-р техн. наук, доцент, заведующий кафедрой теоретических основ и общей электротехники Одесского национального политехнического университета, г. Одесса, Украина, e-mail: Dmitry.A.Maevsky@gmail.com.

Чербаджи Юлия Павловна – студент Одесского национального политехнического университета, г. Одесса, Украина, e-mail: yulkachirba@gmail.com.