DOI 10.31891/2307-5732-2020-283-2-101-107
УДК 621.391 160164

С.М. ЛИСЕНКО, Р.В. ЩУКА
Хмельницький національний університет

АНАЛІЗ МЕТОДІВ ВИЯВЛЕННЯ ШКІДЛИВОГО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ В КОМП'ЮТЕРНИХ СИСТЕМАХ

У даній статті здійснено аналіз сучасного стану шкідливого програмного забезпечення (ШПЗ). Для цього розв'язано три часткові задачі: класифіковано і описано основні типи ШПЗ, прийоми і методи боротьби з його окремими різновидами; розглянуто ряд сучасних підходів до виявлення загроз; з'ясовано основні недоліки поширених методів викриття згубних програм. Виконані дослідження дозволили обґрунтувати необхідність пошуку нових шляхів боротьби з програмними небезпеками. В якості концептуальної основи для такого випадку запропоновано обрати методи штучного інтелекту. На нашу думку, це дозволило б виявляти ШПЗ, яке до відомих хакерських атаках ще не залучалось.

Ключові слова: *шкідливе програмне забезпечення, OpCode, N-грами, комп'ютерна система.*

S. LYSENKO, R. SCHUKA
Khmelnytskyi National University

ANALYSIS OF MALWARE DETECTION METHODS IN COMPUTER SYSTEMS

Malware (malicious software or malware) are programs that are designed to make harm and use the resources of the targeted computer. They are often masked in legal programs, imitate them or just hide in different folders and files in the computer. Moreover, they can get an access to the operating system that allows malware to encrypt files and steal personal information. In some cases malware are distributed by themselves, by e-mail from one computer to another, or through infected files and disks. Fast growing amount of malware makes the computer security researchers invent new methods to protect computers and networks. There are three main methods that are using for malware detection – signature based, behavioural based and heuristic. Signature based malware detection is the most common method used by commercial antiviruses and used in the cases which are completely known and documented. Behaviour-based malware detection evaluates an object based on its intended actions before it can actually execute that behaviour. This malware detection method used to cover disadvantages of signature based method. However, this approaches cannot normally detect harmful software, since such new signatures are not available for newly created malware. On another hand, heuristic methods for detecting harmful software are considered the most effective because they use advanced algorithms based on machine learning technologies. In this paper, we provide the analysis of current state of malicious software. Firstly, we described and classified main types of malware. Then we provide common malware detection approaches and their disadvantages. After that we focused on heuristic malware detection approaches based on artificial intelligence and briefly overview various features of this methods such as API Calls, OpCodes, N-Grams etc.

Keywords: *Malware detection, N-gram, API, Neural networks, computer system.*

Вступ

Стрімкий розвиток інформаційних технологій обумовив помітні зміни у способах та засобах комунікації між людьми із застосуванням мережевих технологій. Створення, зберігання, розповсюдження та спільне використання інформації стає дедалі ще більш простим і доступним. Однак розширення асортименту носіїв інформації, збільшення способів її поширення, подальше удосконалення операційних систем і поява їх нових версій стимулювали еволюцію шкідливого програмного забезпечення та його урізноманітнення. Це призвело до збільшення вразливості інформаційних пристроїв та несанкціонованого доступу до них [1].

Динаміка розвитку ШПЗ

Згідно з даними компанії «McAfee Labs», яка вивчає кіберзагрози та займається дослідженням питань кібербезпеки, протягом останніх років зростання чисельності нового ШПЗ невпинно прискорюється. Так, у першому кварталі 2018 року в середньому реєструвалось 5 нових шкідливих програм (ШП) за секунду. Крім того, спостерігались суттєві технологічні зміни нових ШП, внаслідок яких підвищувалася успішність прийомів зламу. Щодня сервіс McAfee Global Threat Intelligence аналізував 2 500 000 URL-адрес й понад 700 000 файлів. Така обставина обумовила наступну статистику [2]:

- за день в середньому виконувалось 51 000 000 000 запитів;
- у першому кварталі 2018 року захист від ШПЗ спрацьовував 79 000 000 разів на добу, порівняно з 45 000 000 у четвертому кварталі 2017 року;
- у першому кварталі 2018 року захист від ризикованих URL-адрес спрацьовував 49 000 000 разів, що є на 12 000 000 більше, ніж за попередній квартал;
- у першому кварталі 2018 року захист від ризикованих IP-адрес спрацьовував 36 000 000 разів, в порівнянні з 26 000 000 за четвертий квартал 2017.

У другому кварталі 2018 McAfee GTI в середньому отримувало 49 000 000 000 запитів щодоби. У цей час також спостерігався сплеск чисельності нового ШПЗ для мобільних пристроїв – кількість програм збільшилась на 27% порівняно з першим кварталом [3].

Динаміку появи нових ШП та їхню загальну кількість можна побачити на рис. 1, рис. 2, відповідно.

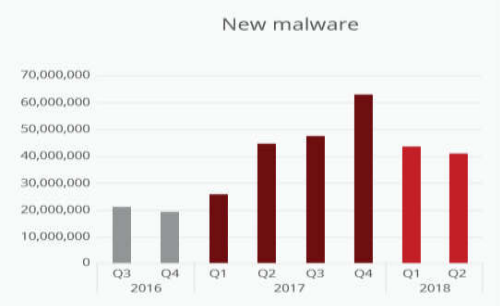

Рис. 1. Кількість нових ШПЗ [3]

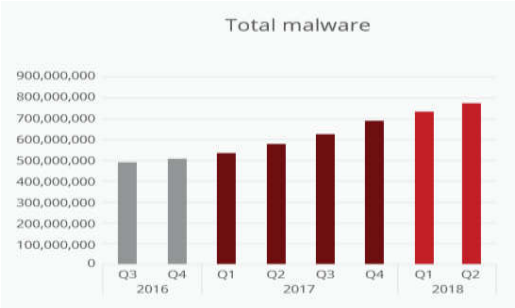

Рис. 2. Загальна кількість ШПЗ [3]

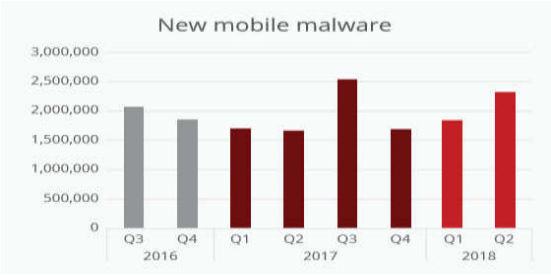

Рис. 3. Кількість нових мобільних ШП [3]

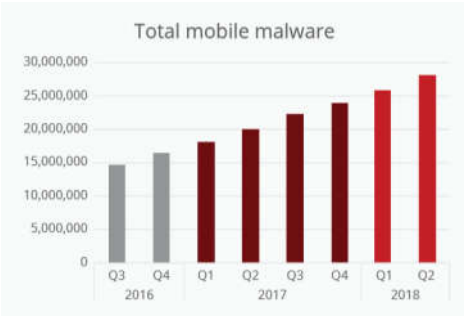

Рис. 4. Загальна кількість мобільних ШП [3]

Як зазначалось у [4], в 2018 році значно виросла кількість нових ШП, пов'язаних з добуванням криптовалют, тоді як ріст інших видів атак поступово спадає. Крім того, виявлення криптомайнінгового ШПЗ зросло на 27 відсотків за останній квартал 2019 року, як наслідок – за кількістю атак цей тип ШПЗ поступається лише рекламним ШП.

На ринку Android ситуація ще більш вразлива. Кількість нового криптомайнінгового ШПЗ за перший квартал 2019 зросла у 40 разів в порівнянні з попереднім періодом, тобто темп склав 4000%. На рис. 3, рис. 4 показано динаміку нових ШП та їх загальну кількість.

Сучасне ШПЗ та тип кібер-атак

Як відомо, вперше визначення поняттю "комп'ютерний вірус" дав Ф. Коен у своїй дисертації, присвяченій самовідтворюваним програмам [5]. З того часу завдання сегрегації комп'ютерних вірусів стали частиною царини задач, пов'язаних з виявленням шкідливого програмного забезпечення.

ШПЗ може бути класифіковане як віруси, хробаки, трояни, бекдори, шпигунські програми та інші типи й підтипи. Вони часто перекривають один одного, тому між ними іноді важко провести межу [6]. У таблиці 1 наведено основні види атак на пристрої-жертви.

Методи виявлення ШПЗ

Виявлення ШПЗ може відбуватися як на стороні мережі, так і на стороні хосту. У першому випадку присутність і дія програми-шкідника фіксується під час використання мережевого трафіку, у другому це відбувається на тлі застосування внутрішніх даних. Подібна обставина зумовлює появу двох типів аналізу шкідливих програм:

- статичного (код програми перевіряється без її фактичного запуску на виконання);
- динамічного (програма виконується у реальному чи віртуальному середовищі).

І ще одна диференціація базується на виокремленні стратегій виявлення шкідливого ПЗ:

- аномалії виконання (полягає у пошуку відхилень від нормальної роботи програми);
- неправомірне виконання (зосереджується на конкретних неправомірних діях й поведінці).

Усе викладене вище дозволяє виділити три основні методи, що використовуються для виявлення шкідливого програмного забезпечення. Це сигнатурні, поведінкові та евристичні методи, які ми зараз розглянемо більш детально.

Сигнатурні методи

Сигнатурні методи описують кожну атаку власною моделлю — сигнатурою. В останньої може застосовуватися:

Таблиця 1

Основні види атак, ініційовані шкідливим ПЗ

Атака	Опис
Фішинг (phishing)	Особисті дані користувачів — облікові записи та номери кредитних карт — збираються за допомогою додатків, електронних листів або SMS, які видаються за справжні.
Шпигунська (Spyware)	Особиста інформація користувачів витягується або виводиться в процесі моніторингу заражених пристроїв. У порівняні зі спостережними атаками, шпигунські не націлюються на окрему жертву.
Спостережна (surveillance)	Конкретний користувач знаходиться під спостереженням власного зараженого пристрою, у якому для цієї мети використовуються вбудовані датчики.
Дайлер (diallware)	Гроші користувачів викрадаються через ШПЗ, яке робить приховані дзвінки на преміальні номери або операторам SMS-послуг.
Хробакова (worm-based)	Черв'як/хробак — це шкідлива програма, яка дублює себе, поширюючись через існуючу мережу (без втручання користувача) від одного пристрою до іншого завдяки використанню різних засобів.
Ботнет (botnet)	Ботнет — це набір пристроїв, заражених шкідливими програмами, які дозволяють хакеру дистанційно керувати ними.
Riskware	Легальне програмне забезпечення, що не несе прямої загрози на встановленому пристрої, але дозволяє зловмисникам отримувати доступ до інформації користувачів
Криптомайнінг – (Cryptomining / cryptojacking)	Криптомайнінг – це відносно новий термін, що стосується шкідливих програм, розроблених з метою залучення ресурсів комп'ютера та використання їх для добування криптовалюти без явного дозволу користувача.
Рекламна (adware)	Спеціальний тип ШПЗ, який орієнтований на показ реклами на пристрої-жертви й збір пошукових запитів, які в майбутньому можуть використовуватись у маркетингових цілях

- рядок символів;
- семантичні вирази на спеціальній мові;
- формальна математична модель тощо.

Виділенням сигнатур займаються експерти в області комп'ютерної вірусології. Вони здатні виділити код вірусу з коду програми і сформулювати його характерні властивості в найбільш зручній для пошуку формі. Практично в кожній компанії, яка займається розробкою антивірусних програм, є своя група фахівців, що аналізує нові віруси і поповнює антивірусну базу новими сигнатурами.

Алгоритм роботи сигнатурного методу заснований на пошуку сигнатур-атак у вихідних даних, зібраних мережевими і хостовими датчиками СВА (інакше, системи виявлення атак). При виявленні сигнатури, СВА фіксує факт інформаційної атаки, яка відповідає знайденому підпису.

Сутність стандартної стратегії виявлення вірусних програм сигнатурними методами полягає у підтримці бази даних (БД) шкідливих сигнатур (інакше, підписів). Кожний вхідний файл перевіряється на наявність у ньому вірусного “підпису”, який збігається з записом у базі даних відомих сигнатур (БДС). При цьому система захисту передбачає постійне оновлення БДС і покладається на нього.

Виявлення ШПЗ на основі сигнатур є найпоширенішим методом, що використовується комерційними антивірусами. Але його можна використовувати у випадках, які є повністю відомими і задокументованими. Із зазначеного стає очевидним суттєвий недолік сигнатурних алгоритмів. Вони не здатні розпізнати атаку нового ШПЗ [7]. Внаслідок цього виникає необхідність постійного оновлення, вдосконалення та впровадження нових евристичних підходів для боротьби зі ШПЗ.

Поведінкові методи

Методи виявлення ШПЗ на основі поведінки полягають в спостереженні за роботою програм та подальшому прийнятті рішення щодо її шкідливості чи безпечності. Оскільки поведінкові методи ґрунтуються на спостеріганні за діями виконуваного файлу, вони здатні нівелювати недоліки сигнатурних методів. Простіше кажучи, детектор на основі поведінки робить висновок про те, чи програма шкідлива, перевіряючи, що вона робить, а не те, що вона “говорить”.

Поведінкові методи виявлення ШПЗ здатні класифікувати програми з однаковою поведінкою. Таким чином, одна поведінкова сигнатура може ідентифікувати різні зразки шкідливих програм. Детектор на основі поведінки в основному складається з наступних компонентів [10]:

- Data Collector: ця компонента збирає динамічну або статичну інформацію про виконуваний файл;
- Interpreter: перетворює необроблену інформацію, зібрану модулем збору даних, на проміжні представлення, придатні для подальшого аналізу;
- Matcher: використовується для порівняння цього представлення з підписами поведінки.

Одним із прикладів поведінкових методів є технологія виявлення зловмисного коду на основі гістограми, що його запатентовано компанією Symantec [10]. Основною перевагою методів виявлення шкідливих програм на основі поведінки є можливість викриття такого типу ШПЗ, який не здатні детектувати сигнатурні методи. Це, наприклад, невідомі та поліморфні варіанти шкідливих програм. З іншого боку, основними недоліками поведінкових методів виявлення ШПЗ є відсутність перспективного відношення хибних спрацювань (False Positive Ratio, FPR) при великій кількості порівнянь а також високий час сканування.

Евристичні методи

Як ми вже згадували, методи виявлення ШПЗ на основі сигнатур і поведінкові методи мають деякі недоліки. Для їх подолання уявляється за доцільне застосувати евристичні методи виявлення шкідливих програм (ЕМШП). Переваги такому кроку забезпечує інтеграція інтелектуального аналізу даних і техніки машинного навчання для вивчення поведінки виконуваного файлу.

Для виявлення ШПЗ евристичними методами застосовуються підходи, показані на рис. 5.

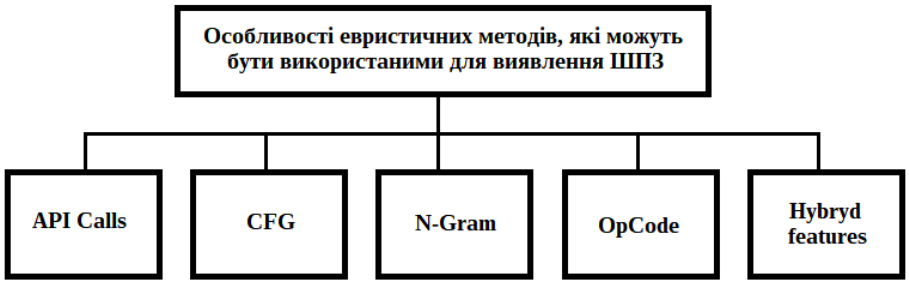

Рис. 5. Особливості використання евристичного методу

Уявляється за доцільне детальніше розглянути основні з них — виклики API, N-грами та Op-коди.

Виклики API

Для відправлення своїх запитів до операційної системи майже всі додатки використовують виклики інтерфейсу прикладного програмування (API) [10]. У цьому сенсі облік послідовності викликів API здатний стати одним з найкращих способів, який відображає поведінку коду.

Одним з перших, хто розглядав послідовності викликів API як функцію шкідливого програмного забезпечення, став Хофмайер [11]. У своїх роботах він запровадив метод виявлення аномалій на основі послідовностей системних викликів. Профайли звичайної поведінки були виконані з використанням коротких послідовностей системних викликів. Для узгодження послідовностей використовувалась відстань Хеммінга. Для встановлення аномалій встановлювався певний поріг — звичайною вважається велика відстань Хеммінга [11].

У 2007 році, на основі аналізу послідовностей виконання Windows API, які викликались файлами Portable Executable (PE), дослідник Ye разом з іншими у [12] запропонували інтелектуальну систему виявлення шкідливих програм — Intelligent Malware Detection System (IMDS), яка використовувала класифікацію на основі майнінга об'єктно-орієнтованої асоціації (ООА). Для створення ефективних правил ООА з метою класифікації ШПЗ було запропоновано адаптувати алгоритм ООА-Fast-FP Growth. Проте, як з'ясувалося, незважаючи на прийнятну результативність виявленні шкідливих програм, IMDS притаманні дві проблеми. Ними виявились:

1) обробка велетенського набору згенерованих правил для побудови класифікатора;

2) знаходження ефективних правил класифікації нових зразків файлів.

Для вирішення зазначених задач Джонг і Лі [13] використовували послідовності системних викликів як для шкідливих, так і для "доброякісних" виконуваних файлів. На основі таких послідовностей будувався топологічний граф, інакше — граф коду. Для кожного двійкового файлу цей граф видовжується та порівнюється з кодовими графами як шкідливих, такі безпечних програм. Відповідно до результатів подібного порівняння програма класифікується як небезпечна або безпечна.

Головним недоліком топологічного графу є його велика довжина. Для подолання цієї вади Lee та ін. у [18] запропонували запровадити виклики API до 128 груп. В результаті графи коду зменшувались і ставали більш зручними для використання.

Згодом цю ідею Ye та ін. у [14] втілили у так званий "класифікатор, що інтерпретує". Він дозволяв на основі аналізу викликів API за допомогою PE-файлу виявляти шкідливі програми з великого та незбалансованого "сірого" списку.

Дослідження групи Ye було засноване на 8 млн ШП шкідливих, такій же кількості безпечних програм і на 100 тис. зразках із сірого списку. Останні були зібрані з антивірусної лабораторії корпорації King soft. Ye з колегами побудували ефективний асоціативний класифікатор, заснований на декількох різних технологіях обробки, включаючи обрізання правил і їх переупорядкування (в оригіналі — rule pruning and rule reordering). Для зменшення чутливості класифікатора до даних дисбалансу і поліпшення його продуктивності, вони розробили ієрархічний асоціативний класифікатор HAC (Hierarchical Associative Classifier).

Операційні коди — OpCode

Варто зауважити, що OpCode (операційний код) — це частина інструкції на машинній мові, яка однозначно ідентифікує операцію, що підлягатиме виконанню. У цьому сенсі будь-яка програма визначається як серія впорядкованих інструкцій, що в остаточному підсумку веде до обчислення конкретного результату. Інструкція — це своєрідний тандем (пара) операційного коду й операнда, іноді — списку операндів.

Ґрунтовне дослідження опкодів [15] зробив D. Bilar. Він припустив, що окремі OpCodes можна використовувати як функцію для виявлення шкідливих програм. Прагнучи довести припущення, він статистично проаналізував можливості окремих OpCodes і продемонстрував їх високу надійність для визначення шкідливості виконуваного файлу. D. Bilar довів, що операційний код може використовуватися в якості потужного представлення виконуваних файлів.

Методам виявлення ШП на основі послідовностей OpCode присвячено ряд праць I. Santos, P. G. Bringas та ін. Наприклад, у [16] вони представили підхід, орієнтований на виявлення заплутаних (обфуркованих) варіантів шкідливих програм (в оригіналі — obfuscated malware variants) за допомогою розрахунків частоти появи послідовностей OpCode у ШПЗ. Частота появи кожного з OpCode визначалась для обох наборів даних — набору ШПЗ та безпечних програм. В якості детектора, що дозволяв розв'язувати заплутані варіанти шкідливих програм використовувалась так звана "зважена частота термінів" (Weighted Term Frequency, WTF). Власне "вагою" слугувала міра подібності нового вектору ознак і вектору властивостей та варіантів шкідливих програм. Вона визначалась, як косинус кута між цими двома згаданими векторами ознак.

Продовжуючи дослідження послідовностей опкодів у напрямі, визначеному в [16], I. Santos підготував декілька класифікаторів машинного навчання. Як відомо, такі класифікатори на основі машинного навчання потребують великої кількості зразків шкідливих і безпечних програм для кожного концептуального класу, який вони намагаються виявити. На практиці отримати і дослідити велику кількість помічених і (або) опрацьованих даних — клопітка і не завжди успішна праця. Для вирішення цієї проблеми у наступних роботах Сантос та ін. запропонували три методи:

1) колективну класифікацію [17];

2) однокласове навчання [18];

3) навчання з напівконтролем [19].

Runwal et al. [20] запропонував новий підхід на основі OpCodes і використовував цей метод для виявлення невідомих, а також метаморфічних сімейств шкідливих програм на основі простого оцінювання подібності графів (a simple graph similarity measurement). Дослідники вилучали опкоди з обох типів файлів (тобто і зі зловмисних, і з доброякісних програм), підраховували кількість пар, які щоразу з'являлися, і на основі цього створювали граф OpCodes. Обчислений таким чином граф давав їм можливість прогнозувати ймовірність небезпечності (шкідливості) нового виконуваного файлу. У подальшому за подібністю графу виконуваного файлу до тестових графів робився висновок щодо приналежності файлу до доброякісних програм чи ШПЗ.

N-грами

T. Abou-assaleh, N. Cercone та ін. у [21] дали визначення терміну "N-грами", як "всі підрядки конкретного рядка з довжиною N". Наприклад, нехай рядок складається з послідовності символів "VIRUS". Його можна розділити на три 3-грами: "VIR", "IRU" та "RUS". Подібно до цієї операції так можна поступити з будь-яким рядком. Двійкові рядки — послідовності одиниць та нулів — не виключення. Протягом останнього десятиліття було проведено цілий ряд досліджень щодо виявлення невідомого шкідливого програмного забезпечення на основі змісту двійкового коду у N-грамах.

Першими, хто запровадив ідею застосування методів виявлення різноманітних ШП на основі власних двійкових кодів, були M. Schultz, E. Eskin, S. Stolfo та ін. У [22] були задіяні три різні способи сегрегації функцій (features), що їх несе виконуваний код:

- функції, вилучені з виконуваних переносних файлів (Portable Executable, PE, PE-section);
- рядки звичайного тексту (expressive plain-text strings), що кодуються у виконуваному файлі;
- функції послідовності байтів (byte sequence features).

Відомо, що чи не найнебезпечнішим є ШПЗ, яке вражає завантажувальні сектори зовнішніх носіїв інформації — DOS Boot Sector або Master Boot Record (MBR). Після ураження системи MBR зазвичай руйнується і модифікується для докорінної зміни порядку завантаження комп'ютерної системи. Першими, хто спромігся використати N-Gram як функції для виявлення подібного ШПЗ, були J. Tesauro, Jeffrey O. Kephart та ін. У [23] вони використовували N-грами для виявлення вірусів, які вражали заватажувальний сектор (Boot Sector Viruses). В якості основного інструментарію дослідники застосували штучні нейронні мережі (Artificial Neural Networks – ANN). N-Grams вибиралися з найбільш уживаних ділянок як шкідливих, так і безпечних ("доброякісних") виконуваних файлів. Учені використовували спеціальний алгоритм зменшення специфічних можливостей функцій (feature reduction algorithm). Принцип роботи останнього ґрунтувався на припущенні про те, що кожна зловмисна програма повинна складатися щонайменше з чотирьох N-грам, які входять довідомого набору N-грам.

У подальших дослідженнях, описаних в [24], J. Tesauro та ін. використовували N-Grams для

побудови декількох класифікаторів на основі ANN та спеціальної стратегії голосування (voting strategy) для досягнення кінцевих результатів (and also used a specific voting strategy to achieve final results.).

Цікаві результати у плані виявлення шкідливих програм дала спроба поєднати фреймворк, який використовує метод загальних N-Gram (Common N-Gram), з класифікатором K-Nearest-Neighbour (KNN), описана Abou-Assaleh зі співаторами в [25]. Для обох класів, тобто зловмисних і доброякісних програм, будувався профіль "делегатів". Кожний новий екземпляр (нова програма) порівнювався з профілями обох класів і класифікувався за подібністю до того чи іншого класу.

Kotler і Maloof [26] використовували байт N-грами для виявлення невідомих шкідливих програм. Хоча вектор функцій N-Gram був двійковим, проте все одно давав можливість детектувати наявність або констатувати відсутність у препарованому файлі шуканих функцій. У своєму попередньому дослідженні Kolter і Maloof [27] вже класифікували ШПЗ на кілька сімейств, долучаючи до цього функції відповідного корисного навантаження. При цьому дослідники намагались максимізувати здатність виявляти шкідливі коди, спираючись на дати створення та призначення файлів.

T. J. Cai DM, M. Gokhale [28] провели кілька експериментів, в яких вони оцінювали суміші семи методів вибору ознак, трьох класифікаторів і розміру байтів N-Gram. Кожний з розглянутих вище евристичних підходів та методів показав свої переваги та недоліки. Для наочності зберемо їх до таблиці 2.

Таблиця 2

Порівняння евристичних методів класифікації ШПЗ [10]

Метод	Переваги	Недоліки
N-грами	Висока точність виявлення ШПЗ. Низький коефіцієнт помилкового розпізнавання (Low false positive ratio).	Великий час виконання
Опкоди (OpCode)	Виявляє заплутані варіанти шкідливих програм. Виявляє метаморфічні ШПЗ. Виявляє невідомі ШПЗ.	Потребує великої кількості виконуваних файлів для кожного з класів (ШПЗ чи доброякісне ПЗ).
API	Виявляє поліморфні та невідомі шкідливі програми. Перевершує інші підходи класифікації як у відношенні виявлення, так і в точності. Виявляє ШПЗ перед його виконанням.	Великий набір створених правил для побудови класифікатора. Великі об'єми даних для порівняння

Для підвищення ефективності виявлення ШПЗ можна використовувати інші методи. Найбільшу цікавість, являють собою графи контролю потоків (CFG) та можливі комбінації розглянутих тут підходів. Всупереч тому, що вони залишилися поза межами нашого аналізу, вважатимемо їх дослідження напрямком для подальшої роботи. У цьому сенсі перспективним вважаємо і застосування можливостей штучного інтелекту.

Висновки

Таким чином, у даній роботі нами розглянуто динаміку розвитку ШПЗ, а також здійснено огляд ряду методів виявлення програм, які можуть становити загрозу для комп'ютерних систем. Нами коротко проаналізовано сигнатурні та поведінкові підходи. Окреслено недоліки існуючого методологічного апарату. Основну увагу приділено евристичним методам на основі викликів API, N-грам та опкодів. Визначено шляхи подальших досліджень у напрямі комплексування досліджених методів з графами контролю потоків (CFG) та використання методів штучного інтелекту.

References

1. Al-khatib A. A., Hammood W. A. Mobile Malware and Defending Systems, Comparison Study. Journal of Electronics and Information Engineering, 2017, Vol. 6, No. 2. pp. 116–123.

2. Beek C., Dunton T. Advanced Data-Stealing Implants GhostSecret and Bankshot Have Global Reach and Implications. McAfee Labs Threats Report, June 2018. pp. 1–27.

3. Beek C., Castillo C. McAfee Global Threat Intelligence analyzed, on average, 1,800,000 URLs, 800,000 files, and another 200,000 files in a sandbox each day in Q2. McAfee Labs Threats Report, September. 2018. pp. 1–21.

4. Kujawa A., A. Kujawa Cybercrime tactics and techniques: Q1 2018 [Electronic resource]. Cybercrime tactics and technique, 2018. Available at: https://www.malwarebytes.com/pdf/white-papers/CTNT-Q1-2018.pdf (last access: 26.02.2020).

5. Cohen F., Cohen F. Computer Viruses, dissertation [Electronic resource]. A Dissertation Presented to the FACULTY OF THE GRADUATE SCHOOL UNIVERSITY OF SOUTHERN CALIFORNIA In Partial Fulfillment of the Requirements for the Degree DOCTOR OF PHILOSOPHY Electrical Engineering January 1986. Available at: http://all.net/books/Dissertation.pdf (last access: 26.02.2020).

6. Szor P., Szor P. The Art of Computer: Virus Research and Defence. [Electronic resource]. Addison Wesley for Symantec Press, New Jersey, vol. 2005. pp. 283–290 Available at: http://index-

of.es/Viruses/T/The%20Art%20of%20Computer%20Virus%20Research%20and%20Defense.pdf (last access: 26.02.2020).

7. Saurabh R., Wilson N., Vaderia S., Panigrakhi R. Decentralised rewall for malware detection [Electronic resource], 2016. Available at: https://ieeexplore.ieee.org/document/8318755 (last access: 26.02.2020).

8. Siddiqui M., Wang M. C., Lee J., A Survey of Data Mining Techniques for Malware Detection using File Features [Electronic resource], 2008. Available at: https://dl.acm.org/doi/10.1145/1593105.1593239 (last access: 26.02.2020).

9. You I., Yim K. Malware Obfuscation Techniques: A Brief Survey[Electronic resource]. In: International Conference on Broadband, Wireless Computing, Communication and Applications (BWCCA), Fukuoka, Japan, 2010. Available at: https://www.researchgate.net/publication/221420990_Malware_Obfuscation_Techniques_A_Brief_Survey (last access: 26.02.2020).

10. Bazrafshan Z., Hashemi H., Hazrati Fard S. M., Hamzeh A. A Survey on Heuristic Malware Detection Techniques [Electronic resource], 2013. Available at: https://www.researchgate.net/publication/260729684_A_survey_on_heuristic_malware_detection_techniques (last access: 26.02.2020).

11. Hofmeyr S., Forrest S., Somayaji A. Intrusion detection using sequences of system calls. Journal of Computer Security, 1998, pp. 151–180.

12. Ye Y., Wang D., Li T., Ye D. IMDS: Intelligent malware detection system. Proc. ACM Int. Conf. Knowl. Discovery Data Mining, 2007. pp. 1043–1047.

13. Jeong K., Lee H. Code graph for malware detection. In Information Networking. ICOIN. In: International Conference on, Jan 2008, p. 679

14. Ye Y., Li T., Huang K., Jiang Q. and Chen Y. Hierarchical associative classifier (HAC) for malware detection from the large and imbalanced gray list. Journal of Intelligent Information Systems, 2008, vol. 3, pp. 1–20.

15. Bilar D. OpCodes as predictor for malware. International Journal of Electronic Security and Digital Forensics, 2007, vol. 1, No 2, p. 156.

16. Santos I., Brezo F., Ugarte-Pedrero X., Bringas P. G. OpCode sequences as representation of executables for data-mining-based unknown malware detection. Information Sciences, Aug. 2011.

17. Santos I., Laorden C., and Bringas P. Collective classification for unknown malware detection. Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security, 2011.

18. Santos I., Brezo F., Sanz B., Laorden C., Bringas P. G. Using opCode sequences in single-class learning to detect unknown malware. IET Information Security, 2011, vol. 5, No 4, p. 220.

19. Santos I., Sanz B., Laorden C. OpCode-sequence-based semisupervised unknown malware detection. Computational Intelligence in Security for Information Systems, 2011.

20. Runwal N., Low R. M., Stamp M. OpCode graph similarity and metamorphic detection. Journal in Computer Virology, Apr. 2012, vol. 8, No 1–2, pp. 37–52.

21. Cercone N. T., Keß N., Sweidan R. Abou-assaleh. N-gram-based Detection of New Malicious Code, 2004, No. 1.

22. Bazrafshan Z., Hashemi H., Hazrati Fard S. M., Hamzeh A. A Survey on Heuristic Malware Detection Techniques [Electronic resource]. Available at: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.530.8584&rep=rep1&type=pdf (last access: 26.02.2020).

23. Schultz M., Eskin E., Zadok E., Stolfo S. Data mining methods for detection of new malicious executables. In IEEE Symposium on Security and Privacy, 2001. IEEE COMPUTER SOCIETY, pp. 38–49.

24. Gerald G. B. S., Tesauro J., Kephart J. O.Neural Network for Computer Virus Recognition. IEEE Expert, 1996.

25. Tesauro W. A., Tesauro G. Automatically Generated Win32 Heuristic Virus Detection. Virus Bulletin Conference, 2000.

26. Santos I., Brezo F., Sanz B., Laorden C., Bringas P. G. Using opCode sequences in single-class learning to detect unknown malware. IET Information Security, 2011, vol. 5, No. 4, p. 220.

27. Kolter M. M. Learning to detect malicious executables in the wild. JZ In roc of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2006.

28. Wild. J. Z., Kolter, M. A. Maloof Learning to Detect and Classify Malicious Executables in the, 2006, vol. 7, pp. 2721–2744.

29. Cai T. J., DM, Gokhale M. Comparison of feature selection and classification algorithms in identifying malicious executables. In Computational Statistics and Data Analysis, 2007.

Рецензія/Peer review : 11.5.2020 р. Надрукована/Printed : 16.6.2020 р.
Стаття рецензована редакційною колегією