https://doi.org/10.31891/2307-5732-2026-363-52
УДК 004.04

ІВАСЬЄВ СТЕПАН
Західноукраїнський національний університет
https://orcid.org/0000-0003-2243-5956
e-mail: isv@wunu.edu.ua
БАРАННІК БОГДАН
Західноукраїнський національний університет
https://orcid.org/0009-0003-0408-9807
e-mail: bbo@wunu.edu.ua
ЦАВОЛИК ТАРАС
Західноукраїнський національний університет
https://orcid.org/0000-0002-1136-5705
tth@wunu.edu.ua

ПОБУДОВА ВІРТУАЛЬНОЇ МАШИНИ ДЛЯ ДИНАМІЧНОГО АНАЛІЗУ ШКІДЛИВОГО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Робота присвячена практичним аспектам створення віртуальних машин для аналізу поведінки шкідливого програмного забезпечення. Розглянуто практичні аспекти побудови та автоматизації віртуальних машин для динамічного аналізу шкідливого програмного забезпечення в умовах зростання складності сучасних кіберзагроз. Проаналізовано актуальні підходи до ізольованого виконання та спостереження за поведінкою зразків шкідливого програмного забезпечення з урахуванням сучасних технік обуфускації та протидії віртуалізації. Запропоновано алгоритм побудови контрольованого віртуального середовища для динамічного аналізу зразків програмного забезпечення на базі гіпервізора VirtualBox, який забезпечує можливість автоматизованого розгортання, запуску та завершення аналізу поведінки шкідливого програмного забезпечення. Розроблено PowerShell-скрипт, що реалізує повний цикл динамічного аналізу, включаючи створення віртуальної машини з заданими параметрами, передачу досліджуваних зразків програмного забезпечення та інструментів спостереження, запуск виконуваного коду, збір лог-файлів, скріншотів та результатів аналізу з подальшим копіюванням їх на хостову систему. Проведено тестування розробленого підходу на зразках виконуваного коду, що здійснюють зміни у файловій системі та реєстрі операційної системи Windows. В роботі приведені результати розробки програмного засобу розгортання віртуальної машини для динамічного аналізу шкідливого програмного забезпечення, що спрощує та автоматизує процес дослідження зразків виконуваного коду. Запропоновано алгоритм побудови середовища для динамічного аналізу шкідливого програмного забезпечення.

Ключові слова: *віртуальні машини, virtualbox, динамічний аналіз, шкідливе програмне забезпечення*

IVASIEV STEPAN, BARANNIK BOHDAN, TSAVOLYK TARAS
Western Ukrainian National University

CONSTRUCTION OF A VIRTUAL MACHINE FOR DYNAMIC ANALYSIS OF MALWARE

The paper is devoted to the practical aspects of creating virtual machines for analyzing the behavior of malicious software. The study examines practical issues related to the construction and automation of virtual machines for dynamic malware analysis in the context of the increasing complexity of modern cyber threats. Current approaches to isolated execution and behavioral monitoring of malware samples are analyzed, taking into account modern obfuscation techniques and anti-virtualization mechanisms. An algorithm for constructing a controlled virtual environment for the dynamic analysis of software samples based on the VirtualBox hypervisor is proposed. The solution provides automated deployment, execution, and termination of malware behavior analysis. A PowerShell script has been developed that implements the complete dynamic analysis lifecycle, including the creation of a virtual machine with predefined parameters, transfer of analyzed software samples and monitoring tools, execution of the target code, collection of log files, screenshots, and analysis results, followed by their retrieval to the host system. Attention is paid to the scalability and modifiability of the proposed software solution, which enables the integration of various monitoring tools, modification of sample execution scenarios, and adaptation of the environment to different types of malicious software. The developed approach was tested using executable code samples that perform modifications to the file system and the Windows operating system registry. The paper presents the results of developing a software tool for deploying a virtual machine for dynamic malware analysis, which simplifies and automates the process of investigating executable code samples. An algorithm for constructing an environment for the dynamic analysis of malicious software is proposed.

Keywords*: virtual machines, VirtualBox, dynamic analysis, malicious software*

Стаття надійшла до редакції / Received 11.02.2026
Прийнята до друку / Accepted 28.02.2026
Опубліковано / Published 26.03.2026

Постановка проблеми.

В умовах зростання кіберзагроз та збільшення кількості та різновидів шкідливого програмного забезпечення, зростає необхідність в розвитку систем аналізу та удосконаленню інструментів для дослідження шкідливого програмного забезпечення. Зловмисники все частіше застосовують нові техніки обуфускації та засоби протидії віртуальним машинам, що ускладнює аналіз шкідливого програмного забезпечення та приховує кінцеву мету створених програмних засобів.

Постановка завдання.

Величезна кількість засобів дослідження та спостереження за діяльністю процесів породжує необхідність створення гнучких інструментів автоматизації для динамічного аналізу ШПЗ. Тому існує потреба

в побудові скрипта для автоматизації побудови віртуальної машини та запуску інструментів для дослідження. Алгоритм роботи скрипта буде складатись з кількох етапів. На першому етапі відбувається побудова віртуальної машини з заданими параметрами та передача файлів з інструментами для дослідження на віртуальний диск. Наступним етапом буде запуск засобів спостереження та самого досліджуваного зразка. Завершальним етапом буде збереження результатів, скріншотів та лог файлів на хостову машину.

Аналіз досліджень і публікацій.

Сучасні засоби динамічного аналізу шкідливого програмного забезпечення постійно конкурують з засобами протидії віртуальним машинам. Динаміка розвитку шкідливого програмного забезпечення (ШПЗ) обумовлює необхідність дослідження та розвитку систем динамічного аналізу ШПЗ та побудови нових архітектур систем для ізольованого дослідження діяльності зразків.

В дослідженні [1] проаналізовано мтатичний та динамічний підходи для оцінки діяльності ШПЗ, отримати індикатори компроментації та методики деанонімізації зловмисника. Також в дослідженні приділено увагу аналізу поведінки шкідливого програмного забезпечення.

Дослідження [2] охоплює два підходи: статичний та динамічний. Для статичного аналізу розглянуто найбільш розповсюджені техніки дослідження PE файлів. Динамічний підхід застосовується в парі з використанням Docker контейнера. Наводяться засоби для аналізу оперативної пам’яті та мережевої активності зразка ШПЗ.

Зростання кількості смартфонів призводить до збільшення шкідливого програмного забезпечення для мобільних платформ. В дослідженні [3] запропоновано нову методику, котра поєднує в собі елементи статичного та динамічного аналізу. Проведене порівняння запропонованої методики з уже існуючими.

Аналіз порівняння можливостей динамічного аналізу ШПЗ з допомогою Chuckoo sandbox та статичного аналізу засобами PEFILE показує вищу ефективність застосування саме статичного підходу до дослідження PE файлів приведено в [4]. Проте дане дослідження не бере до уваги можливості обуфускації програмного коду на домішування хаотичного коду(випадкових викликів функцій) для приховування справжніх цілей ШПЗ. В досліджені також наведені недоліки динамічного аналізу ШПЗ.

Розробка автоматизованого інструменту виявлення та запобігання шкідливим програмам для майнінгу криптовалют на базі хоста запропонована в дослідженні [5]. Запропонована автоматизована система поєднує в собі декілька підходів, таких як сигнатурний, для ідентифікації наявних загроз та статичний і динамічний аналіз. Програми майнери, як окремий вид шкідливого програмного забезпечення зазнав значного розвитку завдяки гарантованому прибутку для автора або розповсюджувача програмного засобу.

В дослідженні [6] запропоновано модель на основі Python, котра поєднує статичний та динамічний підходи до аналізу шкідливого програмного забезпечення. В публікації проаналізовано використання комплексу інструментів для різних підходів аналізу на різних видах ШПЗ.

В [7] проаналізовано зразок ШПЗ з використанням методики динамічного аналізу. Проведене дослідження поведінки ШПЗ шляхом його виконання в контрольованому середовищі та проведена оцінка механізмів застосованих в програмному засобі. Поведінковий аналіз є важливим елементом в досліджені та встановлені технік застосованих ШПЗ.

В досліджені [8] пропонується автоматична система виявлення ШПЗ. Система використовує як динамічний підхід до аналізу ШПЗ та статичний. Отримані в результаті аналізу дані про процес класифікуються за допомогою машинного навчання з використанням методу навчання Naive Bayes. Встановлені кількісні характеристики точності виявлення. Для ознак отриманих з використанням статичних ознак валідність виявлення становить 93 відсотки, а для ознак отриманих інструментами для динамічного аналізу становить 85.

З розвитком IoT кількість загроз та ШПЗ для сфери інтернет речей значно зросла. Дослідження [9] присвячене використанню ланцюга Маркова для аналізу API викликів та класифікації ШПЗ. В публікації проаналізовано набори даних з 24 тисяч зразків ШПЗ та 22 тисяч доброякісних додатків. Експериментально підтверджено ефективність використання запропонованого підходу з ймовірністю виявлення ШПЗ 89%, що ефективніше за використання аналізу частоти викликів API.

В дослідженні [10] пропонується вдосконалена методологія класифікації ШПЗ, засновану на аналізі послідовностей системних викликів, що викликаються шкідливим програмним забезпеченням у середовищі динамічного аналізу. Дослідження показує, що додавання механізму уваги до моделі LSTM підвищує точність завдання класифікації ШПЗ. Запропонований підхід може бути частиною системи підтримки та прийняття рішень для експертів з безпеки для завдання класифікації шкідливих програм.

Дослідження [11] спрямоване на виявлення окремого виду ШПЗ, методів майнингу з використанням пісочниць. Дослідження використовує методи статичного, так і динамічного аналізу в цьому процесі. В роботі обґрунтовано поєднання статичного та динамічного аналізу для виявлення цього різновиду ШПЗ.

В дослідженні [12] проаналізовано тенденції щодо розвитку протидії виявлення віртуальних машин. Для пісочниць, реалізованих за допомогою віртуальних машин, засоби протидії виявленню систем віртуалізації працюють шляхом очищення драйверів, процесів, версій BIOS та інших індикаторів, що і ідентифікації віртуальних машин, специфічних для певних постачальників, тоді як більш складні пісочниці відходять від систем на основі емуляції та віртуалізації до хостів на голому залізі. Запропоновано статичні підходи для побудови віртуальних машин для динамічного аналізу ШПЗ.

В [13] досліджено застосування технік імітації взаємодії людини з комп'ютером (HCI), і техніки

застосування розробниками програм для створення розширених моделей загроз. Аналіз шкідливого програмного забезпечення за допомогою пісочниці авторами більше не вважається надійним методом. Стаття має на меті оцінити ефективність методів пісочниці та засобів виявлення систем віртуалізації, що використовуються шкідливими програмами для уникнення їх. Для цього аналізу застосовано Trojan Upclicker, який використовує HCI для свого впровадження та виконання. Комерційна готова пісочниця дала вичерпні детальні результати та виявила ШПЗ. Висновки ґрунтуються на необхідності ефективної імітації подій HCI.

Метою публікації [14] є аналіз еволюції мобільних пісочниць та запропоновано вдосконалений підхід до розширеної архітектури пісочниці, щоб зменшити можливість виявлення систем віртуалізації. Створення інтелектуальних пісочниць стає безперечно важливим для мобільної безпеки. Автори в дослідженні розглядають сучасні методи мобільної пісочниці, визначаються вимоги до пропозиції надійної методології мобільної пісочниці, яка враховує відсутність реальної поведінки користувачів та долає ризик уникнення пісочниці. Завдяки запропонованій архітектурі розумної пісочниці знизиться обізнаність щодо шкідливого програмного забезпечення з боку навколишнього середовища та посилиться захист від атак передових мобільних шкідливих програм.

В [15] досліджено можливості застосування контейнерів для ізоляції шкідливих дій. Автори аналізують використання SecBox, для вилучення системних викликів, показників продуктивності та мережевого трафіку. SecBox реалізує візуальний та простий у використанні інструмент для виконання шкідливих програм з існуючих бірж зразків у режимі реального часу. Проаналізовано експерименти застосування SecBox та проведена оцінка результатів щодо продуктивності, ізоляції, відтворюваності та моніторингу шкідливих програм.

Алгоритм побудови віртуального середовища для динамічного аналізу

Для побудови віртуального середовища для динамічного аналізу ШПЗ обрано віртуальну машину на базі Virtual Box. Обраний гіпервізор дозволяє автоматизувати процес побудови віртуальної машини та організувати обмін даними, такими як файли для аналізу та результати виконання тестового ШПЗ. Алгоритм побудови середовища приведений на рисунку 1.

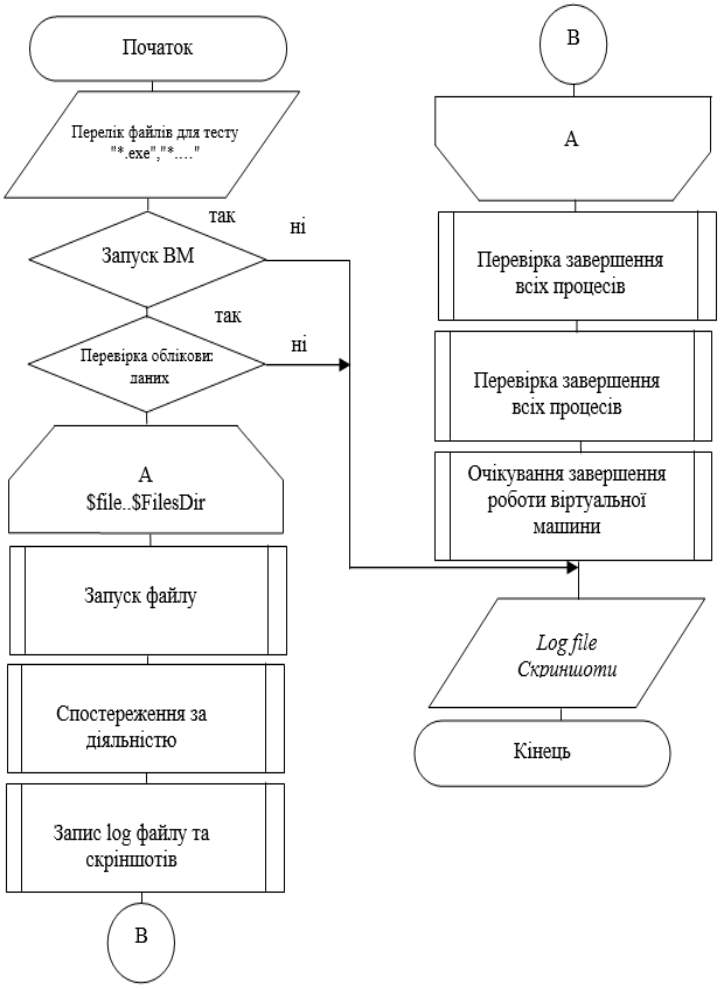

Рис. 1. Алгоритм побудови середовища для динамічного аналізу ШПЗ

Для автоматизованої побудови віртуального середовища створено PowerShell скрипт, що дозволяє запустити віртуальне середовище, скопіювати тестовані файли, запустити засоби спостереження за процесом, зберегти логи файлів та скріншоти виконання, скопіювати результати та завершити виконання середовища.

Такий підхід є досить зручним та дозволяє підключати безліч засобів спостереження за діяльністю процесу.

Для запуску програми що тестується створено окрему функцію Start-VMProgram, роботу якої приведемо на рисунку 2.

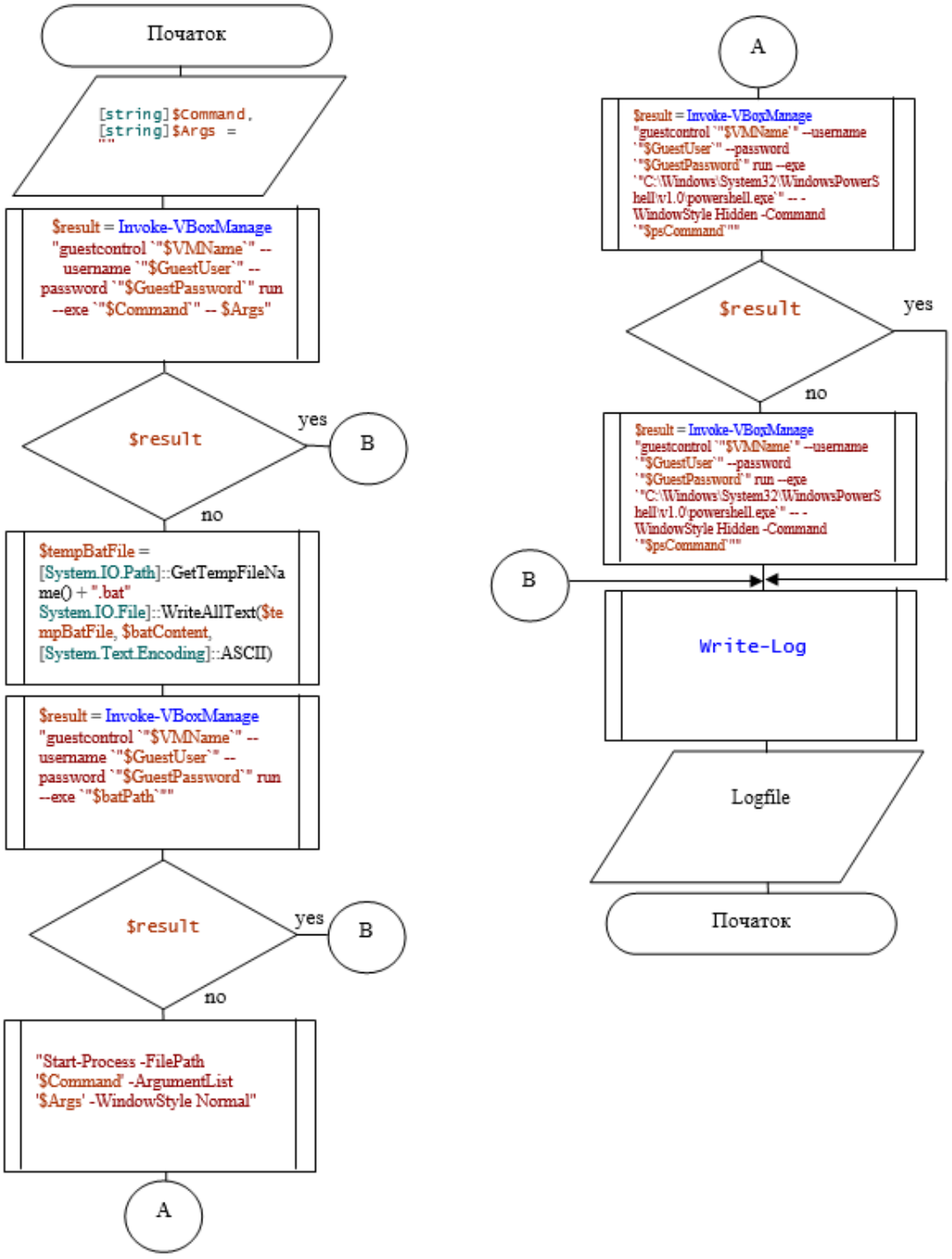

Рис. 2. Схема роботи функції Start-VMProgram

Запропонований алгоритм та розроблене програмне забезпечення легко масштабується та піддається модифікаціям, що дозволяє додати нові модулі та функції. Також скрипт дозволяє додавати параметри запуску зразка та передавати сам зразок на дослідження. Функція логування зберігає результати змін в системі та стан віртуальної машини в лог файл.

Тестування засобів розгортання віртуальної машини

Для тестування автоматизації створення віртуального середовища було передано графічний об’єкт для тестування запуску. На рисунку 3 приведено результати етапу створення віртуального середовища. Для тестування створено тестовий зразок що виконує певні маніпуляції з реєстром та файловою системою windows.

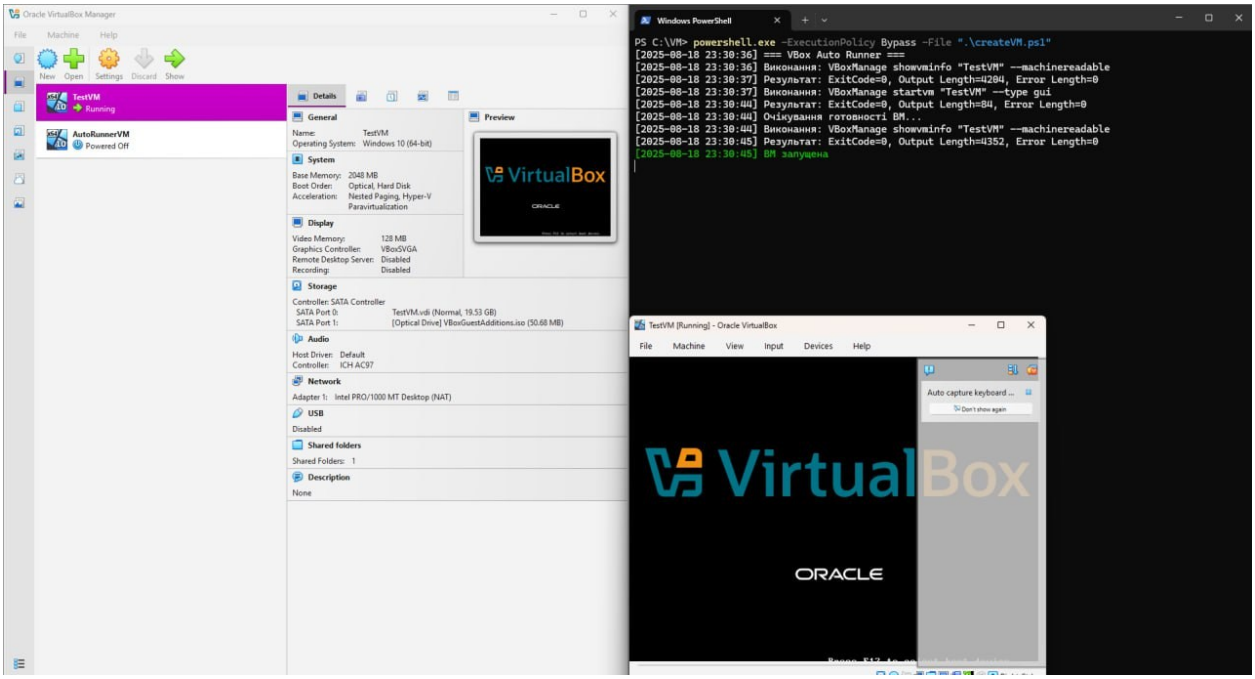

Рис. 3. Запуск віртуальної машини

Тестовий файл копіюється в папку “C:\temp”, після чого він буде відкритий та зафіксовані будь-які зміни в операційній системі. Події операційної системи за котрими ведеться спостереження можна коригувати шляхом модифікації PS скрипта. Результат виконання відображений на рисунку 4.

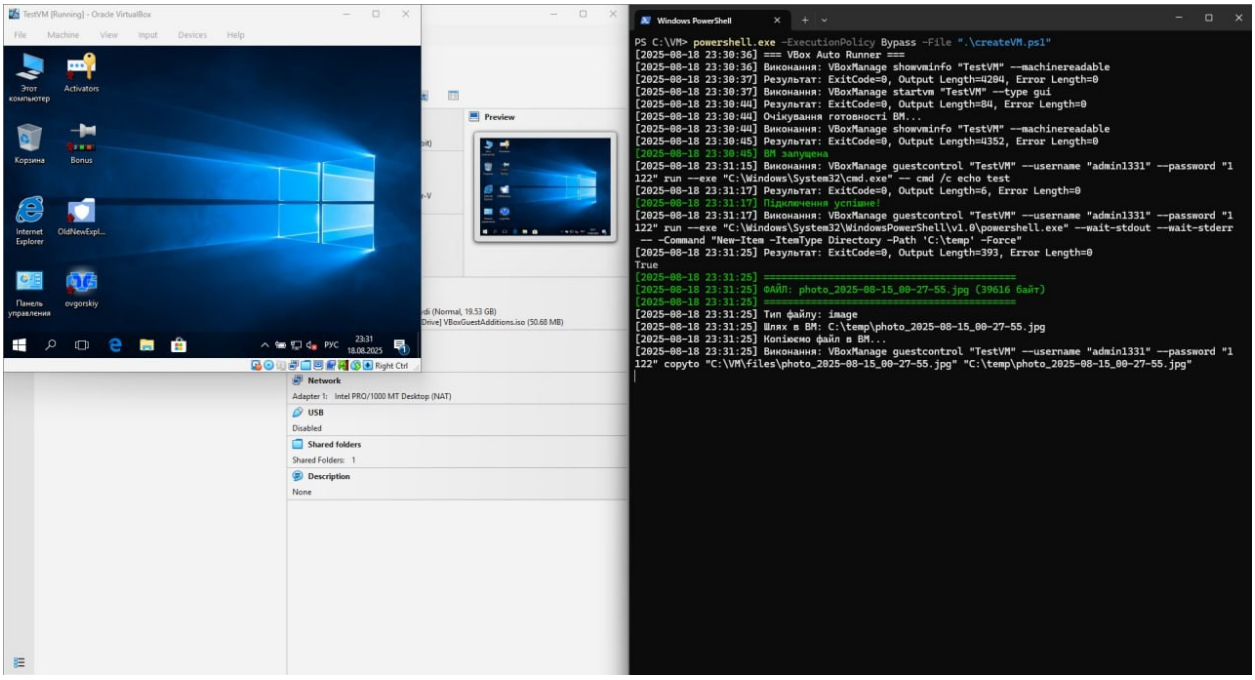

Рис. 4. Копіювання та запуск тестового файлу

Спостереження за діями виконуваного файлу можна організувати за допомогою різних застосунків. Наприклад для логування дій виконуваного файлу можна застосувати команду: «procmon64.exe /testfile.exe C:\logs\reglog.pml /Quiet /Minimized»

Для спостереження за діями лише в реєстрі було протестовано використання «procmon64.exe /LoadConfig C:\filters\registry.pmc /testfile.exe C:\logs\regonly.pml /Quiet». На рисунку 5 приведено результати спостереження за досліджуваним зразком та завершення виконання віртуальної машини. Можна змінювати або додавати утиліти для спостереження за діяльністю зразка та модифіковувати параметри їхнього запуску. Збір результатів відбудеться з усієї папки C:\logs.

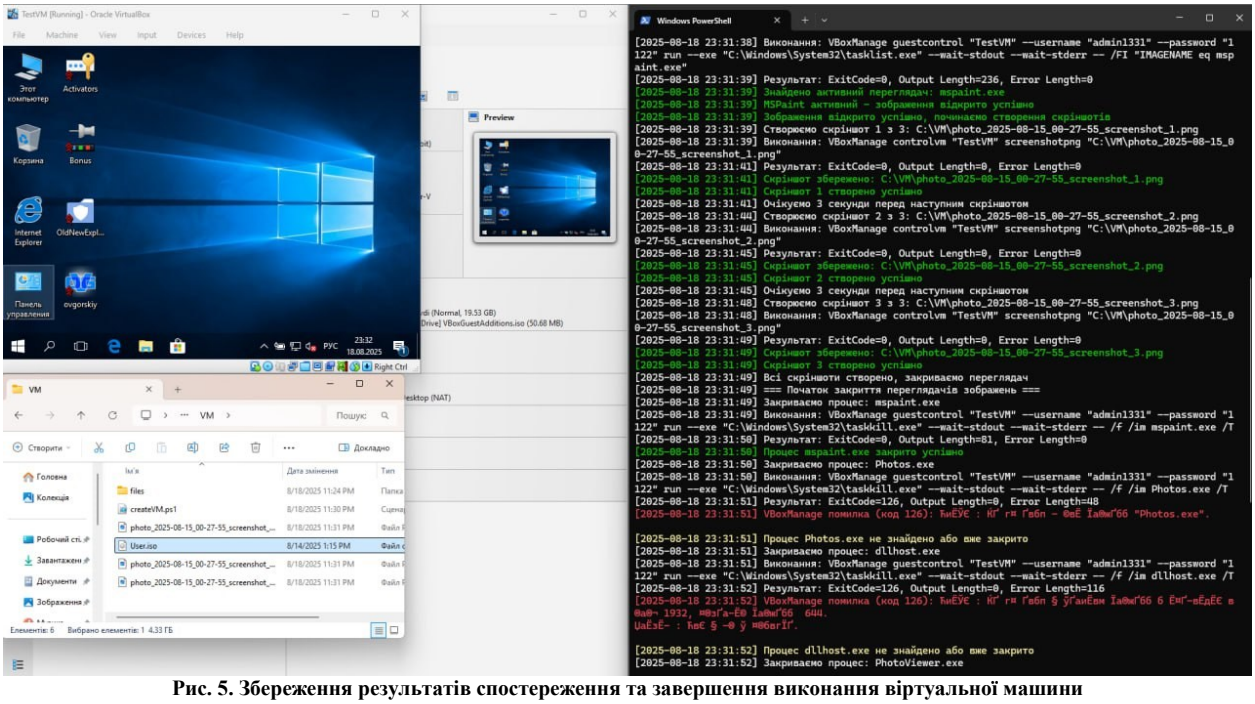

Рис. 5. Збереження результатів спостереження та завершення виконання віртуальної машини

Після завершення дослідження файлу лог-файли та скріншоти результати виконання програми копіюються назад на хостову машину для подальшого вивчення. Тестування показують, що запропонований підхід до автоматизації створення віртуальної машини для динамічного аналізу ШПЗ значно спрощують процес аналізу та дозволяють автоматизувати використання різних інструментів та засобів протидії виявлення віртуальних машин.

Висновки

В роботі проаналізовано системи та підходи до динамічного аналізу ШПЗ. В результаті аналізу можна зробити висновок, що засоби протидії віртуальним машинам та розробники ШПЗ постійно вдосконалюють свої підходи тому дослідження та створення засобів для динамічного аналізу є безумовно актуальною задачею. Створений програмний засіб розгортання віртуальної машини для динамічного аналізу ШПЗ дозволяє запускати окремі інструменти для аналізу поведінки ПЗ, що спрощує та автоматизує процес дослідження. З використанням запропонованого підходу можна будувати веб-орієнтовані засоби дослідження ШПЗ.

Література

1. Saurabh, "Advance Malware Analysis Using Static and Dynamic Methodology," 2018 International Conference on Advanced Computation and Telecommunication (ICACAT), Bhopal, India, 2018, pp. 1-5, doi: 10.1109/ICACAT.2018.8933769.
2. K. Sinha and S. Sai, "Integrated Malware Analysis Sandbox for Static and Dynamic Analysis," 2023 14th International Conference on Computing Communication and Networking Technologies (ICCCNT), Delhi, India, 2023, pp. 1-5, doi: 10.1109/ICCCNT56998.2023.10306805.
3. M. Choudhary and B. Kishore, "HAAMD: Hybrid Analysis for Android Malware Detection," 2018 International Conference on Computer Communication and Informatics (ICCCI), Coimbatore, India, 2018, pp. 1-4, doi: 10.1109/ICCCI.2018.8441295.
4. M. Ijaz, M. H. Durad and M. Ismail, "Static and Dynamic Malware Analysis Using Machine Learning," 2019 16th International Bhurban Conference on Applied Sciences and Technology (IBCAST), Islamabad, Pakistan, 2019, pp. 687-691, doi: 10.1109/IBCAST.2019.8667136.
5. Salam, M. S. Hassim, P. N. Jayawickrama and A. B. Muhandiram, "AntiXcavator: Automated Host-Based Detection and Prevention Tool for Crypto-Mining Malware Using Static and Dynamic Analysis," 2023 5th International Conference on Advancements in Computing (ICAC), Colombo, Sri Lanka, 2023, pp. 191-196, doi: 10.1109/ICAC60630.2023.10417381.
6. D. Das, S. M. Satapathy, A. D and A. Agarwal, "Application of Hybrid Approach towards Multi Aspect Classification and Analysis of Malware," 2023 OITS International Conference on Information Technology (OCIT), Raipur, India, 2023, pp. 284-289, doi: 10.1109/OCIT59427.2023.10430958.
7. M. Bhatia, I. Kumar and N. Mohd, "Dynamic Analysis of a Malware Sample: Recognizing its Behavior using Forensic Application," 2023 4th IEEE Global Conference for Advancement in Technology (GCAT), Bangalore, India, 2023, pp. 1-6, doi: 10.1109/GCAT59970.2023.10353478.

8. Ramadhan, Y. Purwanto and M. F. Ruriawan, "Forensic Malware Identification Using Naive Bayes Method," 2020 International Conference on Information Technology Systems and Innovation (ICITSI), Bandung, Indonesia, 2020, pp. 1-7, doi: 10.1109/ICITSI50517.2020.9264959.

9. M. Ficco, "Detecting IoT Malware by Markov Chain Behavioral Models," 2019 IEEE International Conference on Cloud Engineering (IC2E), Prague, Czech Republic, 2019, pp. 229-234, doi: 10.1109/IC2E.2019.00037.

10. O. Or-Meir, A. Cohen, Y. Elovici, L. Rokach and N. Nissim, "Pay Attention: Improving Classification of PE Malware Using Attention Mechanisms Based on System Call Analysis," 2021 International Joint Conference on Neural Networks (IJCNN), Shenzhen, China, 2021, pp. 1-8, doi: 10.1109/IJCNN52387.2021.9533481.

11. F. H. da Costa, "On the Interplay Between Static and Dynamic Analysis for Mining Sandboxes," 2021 IEEE/ACM 43rd International Conference on Software Engineering: Companion Proceedings (ICSE-Companion), Madrid, ES, 2021, pp. 315-319, doi: 10.1109/ICSE-Companion52605.2021.00135.

12. N. Miramirkhani, M. P. Appini, N. Nikiforakis and M. Polychronakis, "Spotless Sandboxes: Evading Malware Analysis Systems Using Wear-and-Tear Artifacts," 2017 IEEE Symposium on Security and Privacy (SP), San Jose, CA, USA, 2017, pp. 1009-1024, doi: 10.1109/SP.2017.42.

13. M. Mehra and D. Pandey, "Event triggered malware: A new challenge to sandboxing," 2015 Annual IEEE India Conference (INDICON), New Delhi, India, 2015, pp. 1-6, doi: 10.1109/INDICON.2015.7443327.

14. E. Gucuyener and M. A. Guvensan, "Towards Next-Generation Smart Sandboxes: Comprehensive Approach to Mobile Application Security," 2024 12th International Symposium on Digital Forensics and Security (ISDFS), San Antonio, TX, USA, 2024, pp. 1-6, doi: 10.1109/ISDFS60797.2024.10527282.

15. Feng et al., "SecBox: a Lightweight Data Mining Platform for Dynamic and Reproducible Malware Analysis," 2024 11th IEEE Swiss Conference on Data Science (SDS), Zurich, Switzerland, 2024, pp. 62-67, doi: 10.1109/SDS60720.2024.00017.

References

1. Saurabh, "Advance Malware Analysis Using Static and Dynamic Methodology," 2018 International Conference on Advanced Computation and Telecommunication (ICACAT), Bhopal, India, 2018, pp. 1-5, doi: 10.1109/ICACAT.2018.8933769.

2. K. Sinha and S. Sai, "Integrated Malware Analysis Sandbox for Static and Dynamic Analysis," 2023 14th International Conference on Computing Communication and Networking Technologies (ICCCNT), Delhi, India, 2023, pp. 1-5, doi: 10.1109/ICCCNT56998.2023.10306805.

3. M. Choudhary and B. Kishore, "HAAMD: Hybrid Analysis for Android Malware Detection," 2018 International Conference on Computer Communication and Informatics (ICCCI), Coimbatore, India, 2018, pp. 1-4, doi: 10.1109/ICCCI.2018.8441295.

4. M. Ijaz, M. H. Durad and M. Ismail, "Static and Dynamic Malware Analysis Using Machine Learning," 2019 16th International Bhurban Conference on Applied Sciences and Technology (IBCAST), Islamabad, Pakistan, 2019, pp. 687-691, doi: 10.1109/IBCAST.2019.8667136.

5. Salam, M. S. Hassim, P. N. Jayawickrama and A. B. Muhandiram, "AntiXcavator: Automated Host-Based Detection and Prevention Tool for Crypto-Mining Malware Using Static and Dynamic Analysis," 2023 5th International Conference on Advancements in Computing (ICAC), Colombo, Sri Lanka, 2023, pp. 191-196, doi: 10.1109/ICAC60630.2023.10417381.

6. D. Das, S. M. Satapathy, A. D and A. Agarwal, "Application of Hybrid Approach towards Multi Aspect Classification and Analysis of Malware," 2023 OITS International Conference on Information Technology (OCIT), Raipur, India, 2023, pp. 284-289, doi: 10.1109/OCIT59427.2023.10430958.

7. M. Bhatia, I. Kumar and N. Mohd, "Dynamic Analysis of a Malware Sample: Recognizing its Behavior using Forensic Application," 2023 4th IEEE Global Conference for Advancement in Technology (GCAT), Bangalore, India, 2023, pp. 1-6, doi: 10.1109/GCAT59970.2023.10353478.

8. Ramadhan, Y. Purwanto and M. F. Ruriawan, "Forensic Malware Identification Using Naive Bayes Method," 2020 International Conference on Information Technology Systems and Innovation (ICITSI), Bandung, Indonesia, 2020, pp. 1-7, doi: 10.1109/ICITSI50517.2020.9264959.

9. M. Ficco, "Detecting IoT Malware by Markov Chain Behavioral Models," 2019 IEEE International Conference on Cloud Engineering (IC2E), Prague, Czech Republic, 2019, pp. 229-234, doi: 10.1109/IC2E.2019.00037.

10. O. Or-Meir, A. Cohen, Y. Elovici, L. Rokach and N. Nissim, "Pay Attention: Improving Classification of PE Malware Using Attention Mechanisms Based on System Call Analysis," 2021 International Joint Conference on Neural Networks (IJCNN), Shenzhen, China, 2021, pp. 1-8, doi: 10.1109/IJCNN52387.2021.9533481.

11. F. H. da Costa, "On the Interplay Between Static and Dynamic Analysis for Mining Sandboxes," 2021 IEEE/ACM 43rd International Conference on Software Engineering: Companion Proceedings (ICSE-Companion), Madrid, ES, 2021, pp. 315-319, doi: 10.1109/ICSE-Companion52605.2021.00135.

12. N. Miramirkhani, M. P. Appini, N. Nikiforakis and M. Polychronakis, "Spotless Sandboxes: Evading Malware Analysis Systems Using Wear-and-Tear Artifacts," 2017 IEEE Symposium on Security and Privacy (SP), San Jose, CA, USA, 2017, pp. 1009-1024, doi: 10.1109/SP.2017.42.

13. M. Mehra and D. Pandey, "Event triggered malware: A new challenge to sandboxing," 2015 Annual IEEE India Conference (INDICON), New Delhi, India, 2015, pp. 1-6, doi: 10.1109/INDICON.2015.7443327.

14. E. Gucuyener and M. A. Guvensan, "Towards Next-Generation Smart Sandboxes: Comprehensive Approach to Mobile Application Security," 2024 12th International Symposium on Digital Forensics and Security (ISDFS), San Antonio, TX, USA, 2024, pp. 1-6, doi: 10.1109/ISDFS60797.2024.10527282.

15. C. Feng et al., "SecBox: a Lightweight Data Mining Platform for Dynamic and Reproducible Malware Analysis," 2024 11th IEEE Swiss Conference on Data Science (SDS), Zurich, Switzerland, 2024, pp. 62-67, doi: 10.1109/SDS60720.2024.00017.