УДК 004

Олег РОМАНІВ, Віталій КОБИЦЯ, Ярослав ЛИЗУН

Західноукраїнський національний університет

АВТОМАТИЗОВАНА ПЕРЕВІРКА ПРОГРАМНИХ ЗАСОБІВ ДЛЯ ВИЯВЛЕННЯ ШКІДЛИВОГО ВПЛИВУ НА ОС

Вступ. У сучасному світі програмного забезпечення (ПЗ) якість та надійність є ключовими факторами успішної розробки. Зважаючи на зростаючі вимоги до складності систем, швидкості їх розробки та випуску на ринок, перевірка програмного забезпечення стає невід'ємною складовою процесу розробки. Однак, через велику кількість можливих шляхів виконання програм, ручне тестування є неефективним і витратним.

Автоматизована перевірка ПЗ дозволяє зменшити ці витрати та підвищити якість кінцевого продукту. Одним із інструментів автоматизованої перевірки є пісочниці (sandbox), які створюють ізольоване середовище для безпечного виконання програм.

Пісочниці дозволяють аналізувати поведінку ПЗ без ризику пошкодження основної системи або мережі. Вони відіграють важливу роль у процесі тестування, особливо в умовах необхідності перевірки на шкідливість або перевірки функціональності ПЗ[1] в різних середовищах.

Мета: аналіз існуючих засобів автоматизованої перевірки програмного забезпечення, зокрема за допомогою пісочниць, а також визначення переваг і недоліків їх використання в умовах сучасних загроз для аналізу шкідливого ПЗ.

1. Існуючі засоби автоматизованого аналізу програмного забезпечення

На сьогодні існує велика кількість інструментів, що забезпечують автоматизовану перевірку програмного забезпечення. Серед них виділяються інструменти, орієнтовані на різні аспекти тестування: функціональність, безпека, продуктивність та інші. Найпопулярнішими засобами є:

- Selenium - відкритий інструмент для автоматизації веб-тестування.
- JUnit - фреймворк для автоматизації тестування Java-програм.
- Appium - інструмент для автоматизованого тестування мобільних додатків.

2. Пісочниці як засіб перевірки ПЗ

Пісочниці є ізольованими середовищами, що використовуються для виконання програм або тестів з метою уникнення потенційної шкоди основній системі. Вони дозволяють тестувати програмне забезпечення безпосередньо в середовищі, яке імітує реальні умови експлуатації[2]. Це особливо корисно для виявлення шкідливого ПЗ, тестування безпеки та виконання функціональних тестів у контрольованому середовищі. На рисунку 1 приведена архітектура типової автоматизованої пісочниці для виявлення шкідливого ПЗ.

Одними з основних інструментів для створення пісочниць є:

- VirtualBox - засіб для створення віртуальних машин, які можна використовувати як пісочниці для тестування.

- Docker - контейнеризаційна платформа, яка дозволяє запускати ПЗ в ізольованих контейнерах.
- Sandboxie - специфічний інструмент для ізоляції виконуваних файлів і додатків в середовищі Windows.

Більшість пісочниць мають типову архітектуру та набір компонентів.

Складові компоненти типової пісочниці приведемо на рисунку 1.

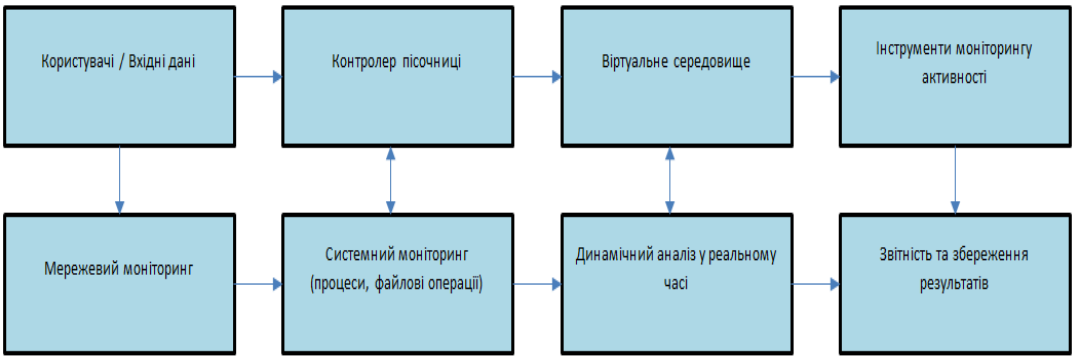

Рисунок 1 - Архітектура пісочниці для тестування ПЗ

Проаналізувавши функціональні можливості пісочниць виділимо переваги їхнього застосування:

1. Безпека: пісочниці дозволяють запускати підозріле програмне забезпечення без ризику для основної системи.
2. Гнучкість: можливість створювати різні середовища для тестування (різні ОС, конфігурації системи тощо).
3. Ефективність: автоматизація процесу тестування дозволяє скоротити час на виявлення проблем.

Серед недоліків є обмеження продуктивності: використання віртуалізації чи контейнеризації може впливати на продуктивність ПЗ, необхідність налаштування: для ефективного використання пісочниць потрібні певні знання і ресурси, неможливість повної імітації реального середовища, хоча пісочниці можуть імітувати багато аспектів системи, вони не завжди можуть точно відтворити всі умови реальної експлуатації.

3. Приклади використання пісочниць в тестуванні ПЗ

Одним з найяскравіших прикладів використання пісочниць є перевірка шкідливого програмного забезпечення в антивірусних компаніях. Компанії, такі як Microsoft та Symantec, активно використовують пісочниці для аналізу підозрілих файлів, визначаючи їхню шкідливість та поведінку в ізольованих середовищах. Часто дослідники використовують пісочниці з відкритим кодом, такі як Cuckoo Sandbox.

Архітектура Cuckoo Sandbox базується на модульному підході, який забезпечує гнучкість, масштабованість і ефективність аналізу підозрілих файлів. Основні компоненти системи розподілені між хостовою системою та віртуальними машинами (VM), що дозволяє ізолювати шкідливе програмне забезпечення для безпечного дослідження.

На рисунку 2 приведена схема послідовності дій для виконання динамічного аналізу шкідливого програмного забезпечення. Хостова система відповідає за управління всім процесом аналізу.

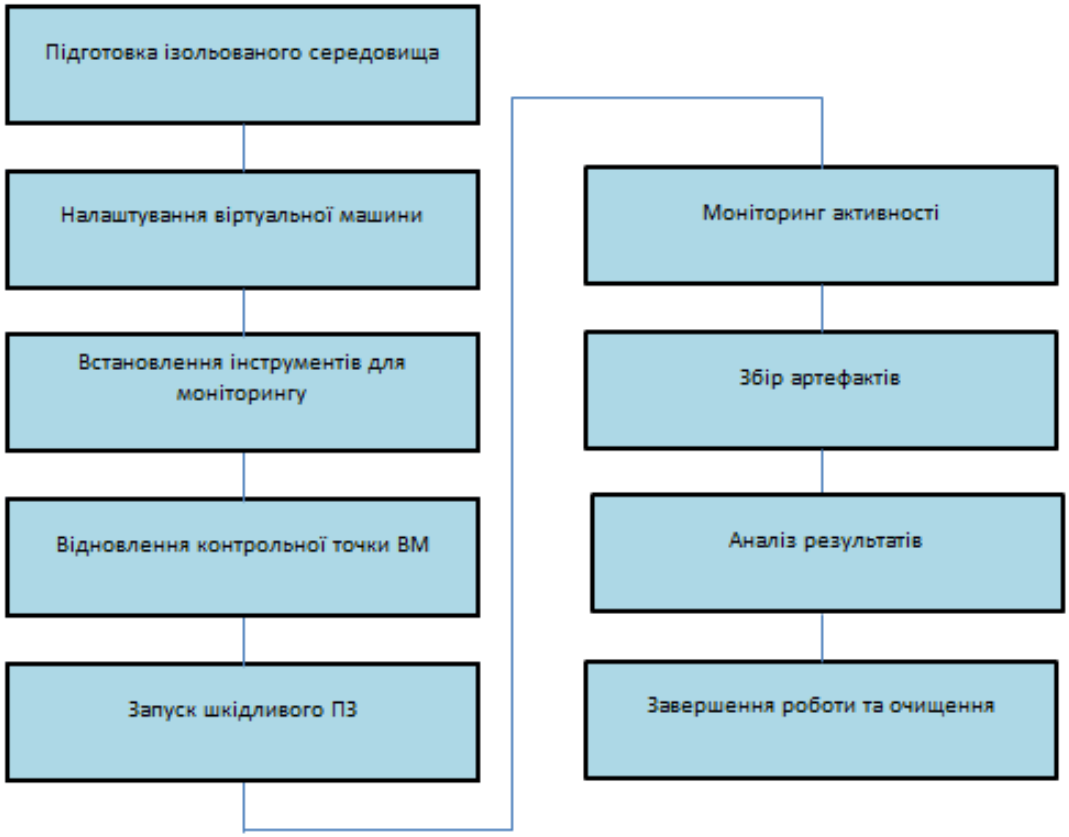

Рисунок 2 - Схема проведення динамічного аналізу шкідливого програмного забезпечення

Гостьові системи (VM) є середовищем, у якому виконується шкідливий файл.

Cuckoo Sandbox - це потужний інструмент для динамічного аналізу підозрілих файлів у безпечному віртуальному середовищі. Налаштування може здатися складним, оскільки вимагає налаштування, як хостової системи, так і віртуальних машин. Проведемо налаштування та встановлення залежностей середовища:

sudo apt update

sudo apt install python3 python3-pip python3-venv git libffi-dev libssl-dev

sudo apt install libjpeg-dev zlib1g-dev swig

sudo apt install mongodb postgresql postgresql-contrib

Настпним етапом буде встановлення hypervisor для віртуалізації. Для virtualbox потрібно виконати команду:

sudo apt install virtualbox virtualbox-ext-pack

Для використання KVM/QEMU потрібно виконати наступну команду:

sudo apt install qemu-kvm libvirt-clients libvirt-daemon-system virt-manager

Важливим етапом є створення Python-оточення. Спочатку потрібно клонувати репозиторій Cuckoo:

git clone https://github.com/cuckoosandbox/cuckoo.git

cd cuckoo

Далі створимо, активуємо та встановимо залежності для віртуального середовища Python:

python3 -m venv venv
source venv/bin/activate
pip install -U pip setuptools wheel
pip install -r requirements.txt

Для нормальної роботи середовища та збереження інформації про події потрібно налаштувати базу даних:

sudo -u postgres psql
CREATE USER cuckoo WITH PASSWORD 'cuckoo';
CREATE DATABASE cuckoo OWNER cuckoo; \q

Cuckoo Sandbox - це проєкт із відкритим вихідним кодом, доступний для будь-кого безкоштовно. Користувачі можуть змінювати код та додавати власні модулі для розширення функціональності. для аналізу Android-зразків. Пісочниця підтримує дослідження виконання різних типів файлів**:** виконувані файли (EXE, DLL), документи (PDF, Word, Excel), скрипти (Python, JavaScript), мобільні програми (APK для Android), **п**ідтримка URL-адрес та мережевого аналізу: можна аналізувати активність вебсайтів і мережевих запитів.

Інструмент дозволяє автоматично виконувати файли та записувати їх дії без втручання людини. Можна використовувати кілька віртуальних машин одночасно для паралельного аналізу великої кількості зразків. Виявлення API-викликів, системних змін, створення/зміни файлів, роботи з реєстром здійснюється окремими інструментами та автоматизовується. Фіксація мережевих запитів (DNS, HTTP, FTP) відбувається за допомогою використання Wireshark. Використання правил YARA для пошуку сигнатур шкідливого ПЗ дозволить ефективніше здійснювати пошук зразків. Пісочниця підтримує ряд операційних систем: Windows, Linux, macOS і Android.

Ще один приклад - тестування нових версій програм у середовищах Docker. Це дозволяє розробникам перевірити програму на різних платформах, не встановлюючи її безпосередньо на основну систему.

Висновок. Засоби автоматизованої перевірки програмного забезпечення, зокрема пісочниці, відіграють важливу роль у сучасній практиці тестування. Пісочниці дозволяють виконувати ПЗ в ізольованих середовищах, що значно знижує ризик пошкодження основної системи та сприяє безпеці процесу перевірки. Завдяки використанню пісочниць можна проводити функціональне тестування, аналіз шкідливих програм, перевірку продуктивності та тестування в різних конфігураціях системи.

Перелік використаних джерел.

1 Michael Sikorski and Andrew Honig. 2012. Practical Malware Analysis: The Hands-On Guide to Dissecting Malicious Software (1st. ed.). No Starch Press, USA.

2 Safa Altaha and Khaled Riad, "Machine Learning in Malware Analysis: Current Trends and Future Directions" International Journal of Advanced Computer Science and Applications(IJACSA), 15(1), 2024. http://dx.doi.org/10.14569/IJACSA.2024.01501124