УДК 004.49

В. А. Гарнага
Д. А. Сокол

МЕТОД ТА ЗАСІБ ВИЯВЛЕННЯ ПРИХОВАНИХ ПРОЦЕСІВ В ОПЕРАЦІЙНІЙ СИСТЕМІ WINDOWS

Вінницький національний технічний університет

Анотація

Розглянуто методику та засоби виявлення прихованих процесів в операційній системі. Запропонований програмний засіб забезпечує виявлення прихованих процесів шляхом безпосереднього зчитування даних із фізичної пам'яті, уникаючи залежності від стандартних API-викликів, які можуть бути скомпрометовані руткітами. На основі аналізу існуючих методів було встановлено їх обмеження, зокрема використання лише користувацького режиму чи опору на стандартні механізми ядра, що не забезпечує достовірного моніторингу у випадку маніпуляцій. У результаті було розроблено архітектуру програмного засобу, що включає модульний підхід, алгоритми глибокого аналізу системних структур і перехресну перевірку даних, отриманих із різних джерел. Розроблений підхід дозволяє виявляти приховані процеси навіть за умов застосування сучасних методів технік маскування. Проведене тестування підтвердило ефективність запропонованого засобу, продемонструвавши його перевагу над аналогами в умовах реальних загроз.

Ключові слова: приховані процеси, аналіз фізичної пам'яті, руткіт, операційна система, безпека ядра, системні структури.

Abstract

The paper describes the methodology and tools for detecting hidden processes in an operating system. The proposed software tool provides detection of hidden processes by directly reading data from physical memory, avoiding dependence on standard API calls that can be compromised by rootkits. Based on the analysis of existing methods, their limitations were identified, including the use of only user mode or reliance on standard kernel mechanisms, which does not provide reliable monitoring in case of manipulation. As a result, we developed a software architecture that includes a modular approach, algorithms for in-depth analysis of system structures, and cross-checking data from various sources. The developed approach allows detecting hidden processes even when using modern methods of masking techniques. The conducted testing has confirmed the effectiveness of the proposed tool, demonstrating its superiority over analogues in the face of real threats.

Keywords: hidden processes, physical memory analysis, rootkit, operating system, kernel security, system structures.

Вступ

Зі зростанням складності методів приховування шкідливих процесів та підвищенням активності руткітів, операційні системи стикаються з необхідністю удосконалення механізмів виявлення загроз. Традиційні засоби моніторингу, які базуються на системних API-викликах, не завжди є ефективними, оскільки шкідливий код може маніпулювати результатами їх виконання, що дозволяє приховувати активні процеси від стандартних інструментів контролю. Це створює значний ризик для інформаційної безпеки системи, роблячи необхідним розробку більш надійних методів аналізу. Запропонована робота спрямована на вирішення цієї проблеми шляхом розробки програмного засобу, який здійснює безпосередній доступ до фізичної пам'яті операційної системи для отримання достовірних даних про активні процеси.

Метою даної роботи є розробка і реалізація ефективного засобу для виявлення прихованих процесів, що усуває недоліки існуючих аналогів та підвищує стійкість системи до сучасних кіберзагроз.

Результати дослідження

Сучасні методи виявлення прихованих процесів здебільшого ґрунтуються на інструментах моніторингу операційної системи, що взаємодіють із системними функціями API для отримання даних про активні процеси. Одним із найпоширеніших підходів є використання виклику NtQuerySystemInformation для отримання списку активних процесів у системі[1]. Однак цей метод є ненадійним, оскільки сучасні руткіти та шкідливі програми можуть маніпулювати функціями API, підміняючи результати їхнього виконання[2]. Шляхом модифікації системних структур, таких як EPROCESS, а також таблиць імпорту та експорту бібліотек (IAT, EAT), шкідливі програми виключають себе зі списків, що повертаються системними функціями. Це унеможливлює їх виявлення традиційними методами моніторингу.

Альтернативна методика, що застосовується деякими аналогами, полягає в аналізі даних користувацького режиму. У цьому випадку перевірка процесів здійснюється через інтерфейси реєстру або диспетчера завдань. Проте цей підхід також є ненадійним, оскільки користувацький режим системи легко піддається маніпуляціям шкідливого коду. Навіть засоби, які працюють на рівні ядра, часто спираються на стандартні API виклики, що робить їх вразливими до технік Rootkit Hooking, таких як DKOM (Direct Kernel Object Manipulation) та підміна SSDT (System Service Dispatch Table) [2].

Запропонований програмний засіб для виявлення прихованих процесів принципово відрізняється від аналогів завдяки використанню методики прямого доступу до фізичної пам’яті операційної системи. Головною перевагою розробленого засобу є те, що він не покладається на виклики API, які можуть бути скомпрометовані, а напряму зчитує дані з фізичної пам'яті[3]. Це дозволяє отримати нефільтровану та достовірну інформацію про системні структури, обійшовши будь-які маніпуляції зі сторони руткітів. У результаті програма здатна ідентифікувати приховані процеси, які не відображаються у стандартних списках системи, навіть якщо їх приховування реалізоване через складні техніки маніпуляції ядром.

Методика роботи запропонованого засобу базується на багаторівневому аналізі системи та інтеграції результатів, отриманих із фізичної пам’яті, структур ядра та функцій API. Програмний засіб було структуровано у вигляді модульної архітектури, що забезпечує чіткий поділ функціоналу та ефективну взаємодію між компонентами.

Модуль MemoryDriver забезпечує низькорівневий доступ до фізичної пам'яті системи. Він дозволяє зчитувати необхідні блоки пам'яті, в яких розташовані критично важливі структури ядра, такі як EPROCESS, ETHREAD та інші об'єкти. Цей підхід дозволяє отримати повний список процесів, включаючи ті, які були приховані за допомогою маніпуляцій у ядрі.

Модуль Structures відповідає за створення та обробку структур даних, які необхідні для аналізу системних об'єктів. До них належать атрибути процесів, ресурси, шляхи до виконуваних файлів та права доступу. Використання точних і деталізованих структур дозволяє ефективно аналізувати кожен процес у системі.

Модуль ProcessParser аналізує дані, отримані з фізичної пам'яті, та порівнює їх із інформацією, отриманою за допомогою API. У процесі порівняння програма виявляє розбіжності між результатами обох методів. Наприклад, якщо активний процес відсутній у списку, отриманому через NtQuerySystemInformation, але присутній у фізичній пам’яті, він позначається як прихований.

Модуль HiddenProcessAnalyzer проводить комплексну перевірку процесів, що були ідентифіковані як приховані або підозрілі. У ході аналізу перевіряються їхні атрибути, споживання ресурсів і шляхи до виконуваних файлів. Активні процеси з аномально низьким використанням пам'яті або невідповідністю критичних атрибутів позначаються як потенційно шкідливі.

Модуль Demon здійснює глибокий аналіз характеристик процесів, включаючи права доступу та інші атрибути. Цей етап дозволяє виявити процеси, які застосовують приховані техніки для уникнення моніторингу.

Модуль SystemMonitor підсумовує результати роботи всіх попередніх модулів і формує детальний звіт про виявлені приховані та підозрілі процеси. Користувач отримує структуровану інформацію про потенційні загрози, що дозволяє оперативно реагувати на виявлені аномалії.

Таким чином, запропонований програмний засіб демонструє значні переваги над традиційними аналогами. Завдяки безпосередньому доступу до фізичної пам’яті та багаторівневому аналізу він дозволяє ефективно виявляти приховані процеси, уникаючи обмежень, пов'язаних із використанням стандартних API та маніпульованих інтерфейсів користувацького режиму. У ході тестування було підтверджено здатність програми виявляти приховані процеси навіть у випадках застосування

сучасних методів маскування, що робить її ефективним інструментом для аналізу та моніторингу системної безпеки.

Висновки

Розробка програмного засобу для виявлення прихованих процесів у операційній системі має значний потенціал для покращення механізмів забезпечення інформаційної безпеки. Запропонований підхід дозволяє ефективно вирішувати проблему приховування шкідливих процесів, що є однією з основних загроз для сучасних операційних систем. Розроблена методика базується на прямому доступі до фізичної пам'яті системи, що дозволяє обійти традиційні методи аналізу на основі стандартних API-викликів, які можуть бути скомпрометовані руткітами та іншими методами маніпуляції ядром.

Таким чином, розроблений програмний засіб та запропонована методика дозволяють суттєво підвищити рівень інформаційної безпеки операційних систем. За рахунок інтеграції глибокого аналізу фізичної пам’яті та перехресної перевірки даних було досягнуто високої ефективності в ідентифікації прихованих процесів, що забезпечує перевагу над існуючими аналогами.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. NtQuerySystemInformation function (winternl.h) - Win32 apps. Microsoft Learn: Build skills that open doors in your career. URL: https://learn.microsoft.com/en-us/windows/win32/api/winternl/nf-winternl-ntquerysysteminformation (дата звернення: 10.12.2024)
2. The Art Of Hiding In Windows – HADESS. URL: https://hadess.io/the-art-of-hiding-in-windows/ (дата звернення: 10.12.2024)
3. Bypassing User-Mode Hooks and Direct Invocation of System Calls for Red Teams - MDSec. URL: https://www.mdsec.co.uk/2020/12/bypassing-user-mode-hooks-and-direct-invocation-of-system-calls-for-red-teams/?utm_source=chatgpt.com (дата звернення: 10.11.2024).

Науковий керівник: ***Гарнага Володимир Анатолійович*** — канд. техн. наук, доцент кафедри захисту інформації, Вінницький національний технічний університет, м. Вінниця,
e-mail: garnaga.volodymyr@vntu.edu.ua
Сокол Дмитро Анатолійович — студент групи 1БС-23М, факультет інформаційних технологій та комп'ютерної інженерії, Вінницький національний технічний університет, Вінниця,
e-mail: dimafolkman@gmail.com.

Supervisor: ***Garnaga Volodymyr Anatoliyovych*** — Cand. Sc., Associate Professor of Information Protection, Vinnytsia National Technical University, Vinnytsia, e-mail: garnaga.volodymyr@vntu.edu.ua
Sokol Dmytro Anatoliyovych — student of group 1BS-23M, faculty of information technologies and computer engineering, Vinnytsia National Technical University, email: dimafolkman@gmail.com.