УДК 004.023

[0000-0002-2180-0969] **Р. Л. Пташкін,**
[0000-0001-5320-3432] **О. О. Гожий,**
[0000-0002-7451-8772] **Ю. Ю. Обруч,**
[0000-0002-4937-747X] **В. В. Павлов,**
[0000-0003-0008-741X] **Р. А. Калініченко**
e-mail: ndekc.ck@gmail.com
Черкаський науково-дослідний експертно-криміналістичний центр МВС України
вул. Пастерівська, 104, м. Черкаси, Україна

АНАЛІЗ ОПЕРАТИВНОЇ ПАМ'ЯТІ ЯК ОДИН ІЗ МЕТОДІВ ПРОВЕДЕННЯ КОМП'ЮТЕРНО-ТЕХНІЧНИХ КРИМІНАЛІСТИЧНИХ ДОСЛІДЖЕНЬ

Підходи та методи сучасної комп'ютерної криміналістики зазвичай оминають такий напрям досліджень, як аналіз оперативної пам'яті. Варто зауважити, що інформаційний вміст оперативної пам'яті працюючої операційної системи є досить цікавим об'єктом дослідження, оскільки може містити не тільки інформацію про роботу системи чи окремих процесів, а й відомості про авторизаційні дані чи ключі шифрування інформації.

Аналіз системної пам'яті як окремий метод криміналістики загалом вимагає вирішення деяких проміжних питань, зокрема вибору інструменту для отримання її вмісту та засобу для дослідження створеного знімка. Ця робота має на меті вирішення зазначених питань. Аби аргументовано здійснити вибір того чи іншого засобу, в роботі проведено деяку апробацію програмного забезпечення та аналіз отриманих результатів.

Ключові слова: *віртуалізація, VirtualBox, дамп пам'яті, RAM, volatility.*

Вступ. У сучасному світі невпинно зростає вплив цифрових технологій на повсякденне життя – наразі під керуванням операційних систем знаходяться не тільки персональні комп'ютери чи спеціалізоване обладнання, а й пристрої звичного нам побуту.

Наприклад, таке звичне нам обладнання, як телевізор за допомогою персонального комп'ютера базового рівня, операційної системи сімейства Linux та програмного засобу на кшталт Kodi можна перетворити на повноцінний медіацентр. Також як окремий приклад можна розглянути використання в системах розумного будинку одноплатних комп'ютерів на кшталт Raspberry Pi.

В обох випадках навіть мінімальні апаратні засоби знаходяться під керуванням повноцінної операційної системи сімейства Linux, яка зазвичай не має кардинальних відмінностей від систем, що застосовуються на більш «серйозному» обладнанні.

Разом зі значним розповсюдженням як операційних систем загалом, так і окремих програмних засобів значно збільшилася кількість випадків неправомірного втручання в їх роботу [1], [2]. Дедалі частіше судовим експертам за напрямом комп'ютерно-технічних досліджень потрібно проводити дослідження специфічних програмних засобів, що зазвичай класифікуються як шкідливі чи небезпечні.

Деякі програмні засоби в ході своєї роботи можуть породжувати певні процеси, створювати файли чи записи реєстру, встановлювати додаткові бібліотеки чи навіть окреме програмне забезпечення [3]. Зазвичай така інформація не зберігається в енергонезалежну пам'ять, а натомість записується в тимчасову. Іноді після виконання необхідних операцій навіть безпосередні програмні засоби самознищуються чи кардинально модифікують себе [4]. Такі дії виконуються з єдиною метою – значно ускладнити проведення аналізу роботи певних засобів, приховати комп'ютерну програму загалом чи певні її функціональні можливості [5].

Проведення дослідження програмного забезпечення, яке або самовидаляється, або розміщується в тимчасовій пам'яті, вимагає від судового експерта застосування кардинально нових методів дослідження. Адже класичні методи, як-от дослідження образу носія цифрової інформації, зазвичай не містять інформації про роботу певних програмних засобів. Прямим дослідженням інформаційного вмісту носія інформації іноді можливо встановити

DOI: 10.24025/2306-4412.4.2020.214993

лише факт наявності програмних засобів, але не факт їхнього використання.

Криміналістичні методи дослідження передбачають спочатку виконання процедур з отримання знімків даних, а потім їх опрацювання в автономному режимі. Зазвичай знімки енергонезалежних даних називають образами носіїв цифрової операції, а знімки системної пам'яті – дампами оперативної пам'яті.

Під автономним режимом варто розуміти можливість проведення аналізу та опрацювання інформації в тому чи іншому вигляді на будь-якому іншому обладнанні без наявності джерела цієї інформації.

Дослідження оперативної пам'яті дає можливість не тільки отримати доступ до даних, які не зберігаються на енергонезалежних носіях інформації, а й провести дослідження «в динаміці». Тобто, почергово створюючи дампи пам'яті, можна проаналізувати, як змінюється та чи інша інформація.

Однак створення дампа оперативної пам'яті не є тривіальним завданням. Варто взяти до уваги той факт, що досліджуване програмне забезпечення зазвичай має функції захисту від аналізу – якщо в системі буде виявлено активний програмний засіб для аналізу (ProcMon, Wireshark тощо), то процес може як зупинити своє виконання, так і зупинити операційну систему загалом. Тобто застосування спеціалізованого програмного забезпечення, яке створює дамп оперативної пам'яті системи, виконуючись у цій же системі, фактично є недоцільним.

Доцільніше розміщувати образ досліджуваного носія інформації у віртуальному середовищі та запускати систему в рамках віртуальної машини. Використання окремої віртуальної машини дасть можливість здійснити дослідження оперативної пам'яті системи без інсталяції додаткових інструментів аналізу та нівелювати майже будь-який захист від здійснення аналізу [7].

У процесі підготовки до проведення аналізу дампа пам'яті варто враховувати той факт, що деякі дані можуть знаходитися в області віртуальної пам'яті – свопі. Адже менеджер віртуальної пам'яті працює таким чином, що на диск потрапляє та інформація, котра не вимагає негайної обробки. Тобто аналіз лише дампа оперативної пам'яті не гарантує повноти й об'єктивності такого дослідження, оскільки недослідженою залишиться інформація про процеси, для яких було встановлено низький пріоритет [8]. Однак ця проблема загалом вирішується в процесі налаштування віртуального середовища. Щоб система не використовувала своп, достатньо виділити віртуальній машині достатню кількість оперативної пам'яті [10].

Наступним питанням, що виникне після отримання дампа оперативної пам'яті, є питання вибору інструменту для проведення аналізу та узагальнення отриманих даних. Вирішення цього питання може бути ключовим для всього дослідження, адже вибір інструменту аналізу повністю залежить від досліджуваної системи та поставленого завдання.

Аналіз останніх досліджень. Аналізуючи останні дослідження за напрямом отримання знімків та безпосереднього дослідження оперативної пам'яті, варто виділити роботи Abhijit Mohanta [4], Reginald Wong [7], Bruce Dang [9], Майкла Сікорські [10], Монаппи [11] та Michael Hale Ligh [15] як найбільш інформативні та різносторонні. Науковцями запропоновано різні програмні засоби як для прямого отримання дампів оперативної пам'яті, так і за допомогою середовища віртуалізації. Також розглянуто деякі програмні рішення для здійснення аналізу отриманих знімків.

У роботі Abhijit Mohanta [4] детально розглянуто програмні рішення для отримання доступу до оперативної пам'яті з гостьової операційної системи. Натомість у роботах Reginald Wong [7] та Майкла Сікорські [10] надано важливі зауваження щодо можливого спотворення дампа оперативної пам'яті при прямому дослідженні, адже деяке програмне забезпечення може фіксувати роботу засобів аналізу та маскуватися чи припиняти свою роботу. Дослідження, описані в роботі Michael Hale Ligh [15], дають змогу повною мірою ознайомитися з деякими засобами аналізу оперативної пам'яті.

Спираючись на роботи [4], [7] та [15], можна стверджувати, що оперативна пам'ять містить структури даних та відомості про процеси, які працюють у системі, – віртуальну пам'ять усіх процесів, віртуальну пам'ять ядра, дескриптори, мережеві з'єднання та інші ресурси, які нині використовуються активними процесами.

З роботи [6] відомо, що для здійснення повного та об'єктивного криміналістичного аналізу судовому експертові необхідно мати

DOI: 10.24025/2306-4412.4.2020.214993

доступ як до енергонезалежної пам'яті, так і до тимчасової. Тут важливо зазначити, що під тимчасовою пам'яттю варто розуміти не тільки тимчасові файлові системи, що зазвичай розміщуються в оперативній пам'яті, а оперативну пам'ять загалом, як це зазначено в [4] та [8].

Загалом інформація, отримана з вищезазначених робіт, вимагає узагальнення та формування на її основі певного єдиного алгоритму чи методики дослідження оперативної пам'яті для їх застосування в комп'ютерній криміналістиці.

Метою дослідження є висвітлення питання криміналістичного аналізу оперативної пам'яті як одного з методів пошуку та виявлення відомостей, цінних у комп'ютерній криміналістиці, оскільки саме системна пам'ять іноді містить критично важливі відомості, як-от ключі шифрування енергонезалежних носіїв інформації чи виконувані фрагменти коду тощо. Опрацювання алгоритму фіксування пам'яті системи та аналізу її вмісту може стати безцінним ресурсом серед можливостей цифрової криміналістики.

До основних завдань роботи варто віднести опрацювання питання створення дампів оперативної пам'яті та питання вибору й застосування програмних засобів аналізу отриманої інформації.

Виклад основного матеріалу необхідно розпочати з вирішення питання створення дампа оперативної пам'яті. Як вже раніше зазначалось у вступній частині, аби мінімізувати вплив внутрішніх процесів операційної системи чи її програмного забезпечення, доцільно розмістити образ досліджуваного носія інформації у віртуальному середовищі та здійснювати завантаження у рамках віртуальної машини.

Загалом поняття віртуалізації – це надання набору обчислювальних ресурсів або їх логічного об'єднання, абстрагованих від апаратної реалізації, та водночас таких, що забезпечують логічну ізоляцію обчислювальних процесів, які виконуються на одному фізичному ресурсі.

Також визначимо поняття гіпервізора – комп'ютерної програми, що забезпечує одночасне і паралельне виконання декількох віртуальних машин на одному фізичному комп'ютері [6]. Саме гіпервізор забезпечує взаємну ізоляцію операційних систем, що виконуються на віртуальних машинах, шляхом розділення фізичних та логічних пристроїв між декількома віртуальними машинами.

Враховуючи вищезазначені поняття, можна визначити, що віртуальною машиною є модель обчислювальної машини, створеної шляхом віртуалізації обчислювальних ресурсів: процесора, оперативної пам'яті, пристроїв зберігання та вводу і виводу інформації. Віртуальна машина, на відміну від програми емуляції певного пристрою, забезпечує повну емуляцію фізичної машини.

Обираючи гіпервізор, важливо мати уявлення про різні наявні платформи та орієнтуватися в їхніх функціональних можливостях. Така необхідність продиктована, насамперед, тим, що деякі середовища віртуалізації перед здійсненням запуску системи вимагають проведення спеціальних підготовчих процедур або зберігають знімки пам'яті віртуальних машин у власних форматах тощо.

Спираючись на роботи Сікорські [10], Монаппи [11] та приклад практичного застосування засобів віртуалізації [12], авторами в цій роботі буде використано середовище віртуалізації Oracle VM VirtualBox – програмний продукт є безкоштовним та відкритим. Його функціональні можливості гарантують стабільну роботу на найбільш поширених архітектурах – Intel x86 та AMD64. Також серед своїх функціональних можливостей Oracle VM VirtualBox містить засоби зупинки виконання віртуальної машини, збереження її стану, повернення до раніше збереженого стану та створення дампа оперативної пам'яті в некодованому форматі raw.

Перед створенням і запуском віртуальної машини варто зауважити, що образу носія інформації в форматі raw не достатньо для того, щоб приєднати його до віртуальної системи. Програмний засіб Oracle VM VirtualBox не використовує формат даних raw як носія інформації, натомість застосовуються деякі спеціалізовані формати: VMDK, VDI, VHD тощо [13], [14]. Проте зазначене середовище віртуалізації комплектується утилітами для конвертування образів носіїв інформації з одного формату в інший, що дає можливість перетворити образ з формату raw у формат VMDK. Для цього необхідно виконати команду «VBoxManage convertdd», передаючи їй такі параметри: шлях до файлу з raw-образом, шлях, де необхідно розміщувати перекодований образ, формат даних, в який необхідно

DOI: 10.24025/2306-4412.4.2020.214993

здійснити конвертацію. Після завершення процедури конвертації у вихідний файл буде збережено перекодований образ, який можна приєднати до віртуальної машини.

Створюючи віртуальну машину, варто пам'ятати про те, що замалий об'єм системної пам'яті призведе до запису даних у своп (за його наявності), а завеликий об'єм може значно збільшити кількість часу, яка необхідна для здійснення аналізу.

В разі необхідності дослідження різних ситуаційних властивостей певного процесу, перед створенням дампа варто спочатку виконати необхідні дії, наприклад запуск певної команди чи програми, підключення стороннього пристрою, ініціювання процедур копіювання чи створення файлів тощо. Так, для прикладу, авторами в рамках середовища операційної системи, що виконується на віртуальній машині, активовано повноекранний додаток, який в подальшому буде досліджено (рисунок 1).

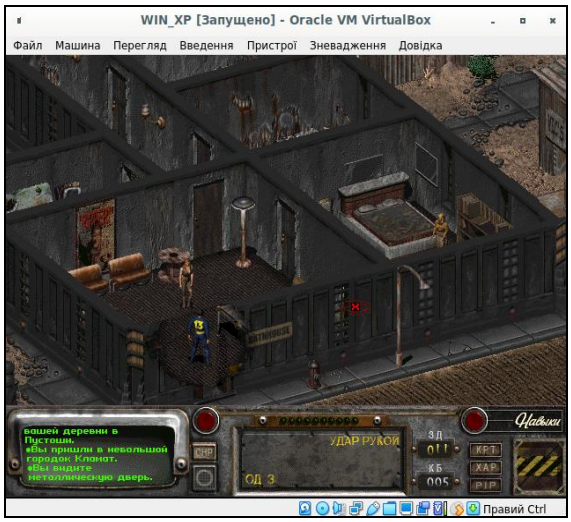

Рисунок 1 – Віртуальна машина з активованим повноекранним процесом

Після створення та запуску віртуальної машини, виконання необхідних підготовчих дій з активації певних процесів можна безпосередньо виконати процедуру створення дампа оперативної пам'яті. Це можливо здійснити через командний рядок середовища віртуалізації (VirtualBox Debugger), виконавши команду «.pgmphystofile» [7]. Зазначеній команді варто передати один параметр – адресу файлу, в який буде записано дамп оперативної пам'яті (рисунок 2).

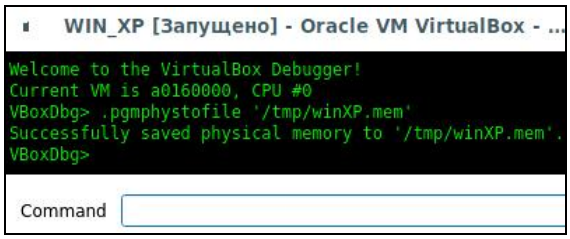

Рисунок 2 – Приклад створення дампа оперативної пам'яті віртуальної машини

За необхідності проведення дослідження в динаміці, варто створити декілька дампів оперативної пам'яті віртуальної машини.

Безпосереднє дослідження створеного дампа чи декількох дампів можливе лише після вибору інструменту для аналізу й узагальнення отриманих даних.

Аналізуючи сучасні напрацювання в галузі аналізу оперативної пам'яті, особливо книг таких авторів, як Монаппа [11] та Michael Hale Ligh [15], встановлено, що найбільш актуальним інструментом для дослідження вмісту оперативної пам'яті операційних систем є програмний засіб «volatility».

Фактично, «volatility» – це безкоштовна інфраструктура програмних рішень (фреймворк) для криміналістичного аналізу інформаційного вмісту оперативної пам'яті. Проаналізувавши його можливості, встановлено, що основна потужність засобу базується на його модульності – фреймворк «volatility» дає змогу підключати величезну кількість плагінів, кожен з яких вирішує певне мінімальне завдання.

Програмний засіб написаний мовою програмування «Python» і підтримує структуру пам'яті, притаманну операційним системам сімейств Windows, Mac OS та Linux. Відкритість вихідних кодів дає можливість аналізувати алгоритм роботи наявних плагінів та, за необхідності, створювати свої, адаптуючи їх під вирішення тих чи інших завдань.

Дослідженням деяких можливостей фреймворку «volatility» встановлено, що найбільш функціональними можливостями можна скористатися лише для аналізу оперативної пам'яті операційних систем сімейства Windows. Можливості для аналізу системної пам'яті операційних систем OS X чи Linux є значно меншими.

Насамперед, це продиктовано тим фактом, що «volatility» не має повного каталогу всіх версій Linux-подібних операційних систем [15]. Це обумовлено тим, що для створен-

DOI: 10.24025/2306-4412.4.2020.214993

ня профілів розпізнавання операційних систем використовується специфічний файл, який містить символьну таблицю адресних функцій та процедур, що використовуються ядром операційної системи Linux. Тобто надзвичайно велика кількість різних дистрибутивів операційних систем разом з надзвичайно великою кількістю версій кожної системи значно ускладнює створення каталогу профілів Linux-подібних систем для фреймворку «volatility». На відміну від Linux-систем, операційні системи сімейства Windows достатньою мірою узагальнені.

Загалом керування програмним засобом «volatility» здійснюється шляхом його запуску з командного рядка з певними параметрами. В обов'язковому випадку через ключ «-f» (наприклад «-f /tmp/dump.mem») варто передавати шлях до файлу з дампом пам'яті. Ще одним параметром, який не зазначається лише при початковому аналізі, а в інших випадках є обов'язковим є профіль досліджуваної операційної системи. Передається він через ключ «--profile», наприклад «--profile=Win7SP1x86». Після зазначених параметрів вказується назва модуля, котрий варто використати, та керуючі ключі цього модуля.

З метою початкового аналізу створеного раніше дампа оперативної пам'яті застосуємо засіб «volatility» і визначимо інформацію про досліджувану операційну систему та назву її профілю в середовищі фреймворку. Для виконання цього завдання необхідно, як зазначено вище, передати ключі зі шляхом до файлу з дампом та зазначити назву модуля «imageinfo» (рисунок 3).

```
mrgauss@ck:/$ volatility -f /tmp/mem.raw imageinfo
Volatility Foundation Volatility Framework 2.6
 INFO: volatility.debug    : Determining profile based on KDBG se
      Suggested Profile(s) : WinXPSP2x86
                 AS Layer1 : IA32PagedMemoryPae (Kernel AS)
                 AS Layer2 : VirtualBoxCoreDumpElf64 (Unnamed AS)
                 AS Layer3 : FileAddressSpace (/tmp/mem.raw)
                  PAE type : PAE
                       DTB : 0x33b000L
                      KDBG : 0x8054d2e0L
      Number of Processors : 1
 Image Type (Service Pack) : 3
            KPCR for CPU 0 : 0xffdff000L
         KUSER_SHARED_DATA : 0xffdf0000L
       Image date and time : 2020-10-02 12:43:43 UTC+0000
 Image local date and time : 2020-10-02 15:43:43 +0300
mrgauss@ck:/$
```

Рисунок 3 – Дослідження дампа оперативної пам'яті засобом «volatility»

В результаті виконання програмний засіб проаналізує дамп пам'яті, визначить та виведе на екран інформацію про досліджувану операційну систему, де буде зазначено її назву та версію. Та найголовнішою буде інформація стосовно назви профілю.

В подальшому спробуємо дослідити процес, який було запущено на етапі підготовки до створення дампа пам'яті. Для цього виконаємо засіб «volatility» з керуючими ключами, які визначатимуть шлях до файлу з дампом, профіль системи та назву модуля – «pslist» чи «pstree». Кожен з цих модулів виведе на екран інформацію про активні процеси, різниця полягає лише у форматуванні виводу – «pstree» сформує результат у вигляді дерева, що корисно для аналізу дочірніх процесів певного процесу (рисунок 4).

```
root@ck:~# volatility -f /tmp/mem.raw --profile=WinXPSP2x86 pstree
Volatility Foundation Volatility Framework 2.6
Name                                 Pid   PPid   Thds   Hnds Time
-------------------------------------------------------------------------------
 0x819ca7f8:System                     4      0     56    257 1970-01-01 00:00:00 UTC+0000
. 0x81709020:smss.exe                368      4      3     19 2020-10-02 12:01:36 UTC+0000
.. 0x817b87e8:winlogon.exe           616    368     19    531 2020-10-02 12:01:37 UTC+0000
... 0x817bbda0:lsass.exe             672    616     25    359 2020-10-02 12:01:37 UTC+0000
... 0x816fbda0:services.exe          660    616     14    262 2020-10-02 12:01:37 UTC+0000
.... 0x8184b020:VBoxService.exe      848    660      9    115 2020-10-02 12:01:37 UTC+0000
.... 0x817533d0:svchost.exe          896    660     20    219 2020-10-02 12:01:37 UTC+0000
.... 0x816d8c18:alg.exe             1024    660      5    103 2020-10-02 12:01:52 UTC+0000
0x81423828:explorer.exe             1560   1500      9    319 2020-10-02 12:01:38 UTC+0000
. 0x817c45d0:VBoxTray.exe           1104   1560     12    117 2020-10-02 12:01:53 UTC+0000
. 0x81758da0:fallout2.exe           1468   1560      5    175 2020-10-02 12:02:14 UTC+0000
. 0x817bc4d8:ctfmon.exe             1144   1560      1     84 2020-10-02 12:01:53 UTC+0000
root@ck:~#
```

Рисунок 4 – Частина списку процесів, виявлених програмним засобом «volatility»

В отриманих даних особливу цінність має ідентифікатор процесу – Pid – числовий номер у порядку запуску. Саме ідентифікатор процесу необхідний для його подальшого дослідження, оскільки є обов'язковим параметром при використанні інших модулів фреймворку «volatility».

Отримавши ідентифікатор необхідного процесу, можливо здійснити його безпосереднє дослідження. Наприклад, використовуючи модуль «dlllist» фреймворку «volatility», можна отримати інформацію про динамічно приєднувані бібліотеки, що були завантаженими й використаними певним процесом (рисунок 5).

```
root@ck:~# volatility -f /tmp/mem.raw --profile=WinXPSP2x86 dlllist -p 1468
Volatility Foundation Volatility Framework 2.6.1
************************************************************************
fallout2.exe pid:   1468
Command line : "C:\Program Files\1C\Fallout2\fallout2.exe"
Service Pack 3

Base             Size  LoadCount  Path
---------- ---------- ----------  ----
0x00400000   0x2f0000     0xffff  C:\Program Files\1C\Fallout2\fallout2.exe
0x7c900000    0xb4000     0xffff  C:\WINDOWS\system32\ntdll.dll
0x7c800000    0xf7000     0xffff  C:\WINDOWS\system32\kernel32.dll
0x77f10000    0x4a000     0xffff  C:\WINDOWS\system32\GDI32.dll
0x7e410000    0x91000     0xffff  C:\WINDOWS\system32\USER32.dll
0x76b40000    0x2d000     0xffff  C:\WINDOWS\system32\WINMM.dll
0x77dd0000    0x9b000     0xffff  C:\WINDOWS\system32\ADVAPI32.dll
0x77e70000    0x93000     0xffff  C:\WINDOWS\system32\RPCRT4.dll
0x77fe0000    0x11000     0xffff  C:\WINDOWS\system32\Secur32.dll
root@ck:~#
```

Рисунок 5 – Список динамічних бібліотек, виявлених програмним засобом «volatility»

DOI: 10.24025/2306-4412.4.2020.214993

Оскільки динамічні бібліотеки можуть містити програмний код, дані чи ресурси в будь-яких комбінаціях, інформація про них може бути досить цінною, аби мати загальне розуміння про можливу функціональність досліджуваного процесу.

Проте іноді програмним засобам вдається приховати деякі з завантажених бібліотек, в такому випадку варто застосувати плагін «ldrmodules» фреймворку «volatility», який проаналізує дамп пам'яті та спробує відобразити всі модулі, що фактично були завантажені процесом, незалежно від методу завантаження [4].

Також, маючи в своєму розпорядженні ідентифікатор процесу, програмним засобом «volatility» можливо виконати експорт процесу в файл. Для цього, за аналогією з попередніми командами, необхідно звернутися до фреймворку «volatility», передаючи ключі з розташуванням дампа пам'яті, профілю системи, ідентифікатором процесу та модулем «procdump» (рисунок 6).

```
root@ck:~# \
> volatility -f /tmp/mem.raw \
> --profile=WinXPSP2x86 \
> procdump -p 1468 -D /tmp
Volatility Foundation Volatility Framework 2.6.1
Process(V) ImageBase  Name         Result
---------- ---------- ------------ ------
0x81758da0 0x00400000 fallout2.exe OK: executable.1468.exe
root@ck:~#
```

Рисунок 6 – Приклад експорту процесу в файл

Отримання файлу з дампом процесу дає можливість його подальшого дослідження вже іншими методами – дизасемблюванням та аналізом програмного коду. Такі методи дають змогу не тільки детально вивчити алгоритм роботи програмного засобу, а й дослідити його на предмет наявності ключів шифрування чи доступу до інформації.

Однак процедури дизасемблювання, аналізу програмного коду та його можлива подальша декомпіляція є завданнями, які потребують проведення подальших досліджень, зокрема аналізу й опрацювання наявних як наукових, так і практичних рішень і напрацювань.

Результати проведеного дослідження повною мірою задовольняють поставлену мету, що полягала в опрацюванні питання криміналістичного аналізу оперативної пам'яті, а саме: в завданнях вибору програмних інструментів і апробації алгоритмів для створення дампів оперативної пам'яті та подальшого її аналізу.

В результаті дослідження встановлено, що достатнім засобом для віртуалізації досліджуваної системи та створення дампа оперативної пам'яті є середовище віртуалізації Oracle VM VirtualBox. Його функціональні можливості дозволяють як створювати віртуальну машину, так і зберігати в файл її системну пам'ять.

Окрім того, програмний засіб «volatility», який було апробовано в рамках цієї роботи, також є достатнім та перспективним засобом для здійснення аналізу дампів оперативної пам'яті.

Результати досліджень та їх аналіз. Обговорюючи результати дослідження, спочатку варто виділити той факт, що запропоновані та застосовані в роботі програмні засоби є безкоштовними й розповсюджуються з відкритим вихідним кодом, тобто є вільними у використанні. Цей факт є важливим у перспективі, оскільки подальші навчання й підготовка фахівців не будуть потребувати придбання додаткових копій програмних засобів чи його ліцензування.

Запропоноване середовище віртуалізації Oracle VM VirtualBox загалом є достатньо універсальним засобом і може бути використане для більш широкого кола завдань при проведенні комп'ютерно-технічних досліджень. Функціональна можливість створення дампів оперативної пам'яті є достатньою для проведення її аналізу.

Стосовно ж фреймворку «volatility», цей програмний засіб є досить перспективним у вирішенні питань аналізу оперативної пам'яті. Суттєвий потенціал програмного рішення ґрунтується на його модульності, а отже, можливості адаптації функціональності під нестандартні завдання.

Досліджуючи ефективність засобу «volatility», було проаналізовано дампи оперативної пам'яті таких операційних систем сімейства Windows, як: XP, Server 2003, Vista, 7, Server 2008 R2 та 8.1. В середовищі кожної операційної системи було запущено один і той же повноекранний додаток. В кожному дослідженому випадку «volatility» правильно розпізнав операційну систему та дав змогу здійснити експорт виконуваного файлу.

DOI: 10.24025/2306-4412.4.2020.214993

Як окремі випадки позитивного досвіду застосування тандему «Oracle VM VirtualBox» та фреймворку «volatility» можна навести практику працівників Черкаського науково-дослідного експертно-криміналістичного центру МВС України. Функціональні можливості «volatility» дали змогу відстежити приховані процеси (malware) в операційних системах сімейства Windows, експортувати виконувані файли для їх подальшого аналізу та дизасемблювання.

Загалом розглянута послідовність виконання процедур дає можливість успішно створити знімок оперативної пам'яті віртуальної машини і здійснити його дослідження на предмет активних процесів, завантажених динамічних бібліотек тощо. Цей факт дає змогу розглядати аналіз оперативної пам'яті як один із методів цифрової криміналістики за експертною спеціальністю 10.9 «Дослідження комп'ютерної техніки та програмних продуктів».

Висновки. В роботі розглянуто та висвітлено питання аналізу оперативної пам'яті як одного з методів пошуку та виявлення відомостей, що мають цінність у розрізі комп'ютерної криміналістики.

Дослідженням запропоновано для застосування середовище віртуалізації та інструмент для аналізу вмісту оперативної пам'яті. Середовище віртуалізації, насамперед, необхідне для реалізації можливості створення знімків (дампів) системної пам'яті віртуальних машин, оскільки дає змогу здійснювати запуск операційних систем у контрольованому середовищі. Як такий засіб запропоновано програмне забезпечення Oracle VM VirtualBox. Окрім того, застосування цього засобу дає можливість здійснювати не тільки статичний аналіз, а й динамічний – почергове створення знімків пам'яті дає змогу відстежувати зміну її вмісту та проаналізувати роботу певних програмних засобів.

Безпосередньо ж для проведення аналізу вмісту знімків оперативної пам'яті запропоновано використання програмного засобу «volatility», який фактично є модульною платформою для виконання базових модулів. Модульна архітектура засобу дає змогу розділити глобальне завдання аналізу на дрібніші проміжні завдання, що, в свою чергу, робить алгоритм загального дослідження більш повторюваним і стійким до нестандартних ситуацій.

Тандем запропонованих засобів може дати змогу судовим експертам за напрямом комп'ютерно-технічних досліджень здійснювати аналіз оперативної пам'яті та отримувати відомості, які є недоступними при застосуванні класичних методів дослідження.

Вищезазначені засоби та підходи відкривають шлях для проведення подальших досліджень. Обумовлено це, насамперед, універсальністю та модульністю програмного забезпечення й можливістю реалізації власних програмних модулів для вирішення специфічних завдань комп'ютерної криміналістики. Загалом же метод аналізу оперативної пам'яті є досить широким поняттям і вимагає подальшого вивчення та розділення на більш конкретизовані напрями.

Список використаних джерел

[1] T. Holt, A. Bossler, and K. Seigfried-Spellar, *Cybercrime and Digital Forensics*. NY, USA: Routledge, 2018. ISBN: 978-1-138-23872-5.

[2] Рада національної безпеки і оборони України (2016, січ. 27). *Рішення "Про Стратегію кібербезпеки України"*. [Електронний ресурс]. Режим доступу: https://zakon.rada.gov.ua/laws/show /96/2016#Text. Дата звернення Жовт. 23, 2020.

[3] Ю. Диогенес, и Э. Озкайя, *Кибербезопасность: стратегии атак и обороны*, пер. с анг. Д. Беликова. Москва, Россия: ДМК Пресс, 2020.

[4] A. Mohanta, and A. Saldanha, *Malware Analysis and Detection Engineering. A Comprehensive Approach to Detect and Analyze Modern Malware*. California, USA: Apress Media, 2020. ISBN-13: 978-1-4842-6192-7.

[5] D. Yurichev, "Reverse Engineering for Beginners". [Online]. Available: https://beginners.re/RE4B-EN.pdf. Accessed on: Oct. 23, 2020.

[6] X. Zhang, *Digital Forensic Education. An Experiential Learning Approach.* Switzerland: Springer, 2020.

[7] R. Wong, *Mastering Reverse Engineering.* Birmingham, UK: Packt Publishing Ltd, 2018. ISBN 978-1-78883-884-9.

[8] A. Arnes, *Digital Forensics.* Hoboken, NJ: Wiley, 2018.

DOI: 10.24025/2306-4412.4.2020.214993

[9] B. Dang, A. Gazet, and E. Bachaalany, *Practical Reverse Engineering: x86, x64, ARM, Windows® Kernel, Reversing Tools, and Obfuscation.* Indianapolis, IN: Wiley, 2014.

[10] М. Сикорски, и Э. Хониг, *Вскрытие покажет! Практический анализ вредоносного ПО.* Санкт-Петербург: Питер, 2018. ISBN 978-5-4461-0641-7.

[11] К. Монаппа, *Анализ вредоносных программ*, пер. с анг. Д. Беликова. Москва, Россия: ДМК Пресс, 2019. ISBN 978-5-97060-700-8.

[12] В. В. Павлов, "Практика завантаження операційної системи, що міститься на цифровому носії інформації в середовищі віртуальної машини", *Вісник Черкаського державного технологічного університету*, № 1, с. 27-33, 2020.

[13] Oracle VM VirtualBox. User Manual. [Online]. Available: https://www.virtualbox.org/manual/UserManual.html. Accessed on: Oct. 23, 2020.

[14] В. Столлингс, *Операционные системы: внутренняя структура и принципы проектирования,* 9-е изд., пер. с англ. Санкт-Петербург: Диалектика, 2020.

[15] M. Hale Ligh, A. Case, J. Levy, and A. Walters, *The Art of Memory Forensics: Detecting Malware and Threats in Windows, Linux, and Mac Memory*. Hoboken, NJ: Wiley, 2018. ISBN: 978-1-118-82509-9.

References

[1] T. Holt, A. Bossler, and K. Seigfried-Spellar, *Cybercrime and Digital Forensics.* NY, USA: Routledge, 2018. ISBN: 978-1-138-23872-5.

[2] National Security and Defense Council of Ukraine (2016, Jan. 27). The decision "On the Cyber Security Strategy of Ukraine". [Online]. Available: https://zakon.rada.gov.ua/laws/show/96/2016#Text. Accessed on: Oct. 23, 2020.

[3] Yu. Diogenes, and E. Ozkaya, *Cybersecurity: Attack and Defense Strategies.* Birmingham, UK: Packt Publishing Ltd., 2018. ISBN 978-1-78847-529-7. [in Russian].

[4] A. Mohanta, and A. Saldanha, *Malware Analysis and Detection Engineering. A Comprehensive Approach to Detect and Analyze Modern Malware*. California, USA: Apress Media, 2020. ISBN-13: 978-1-4842-6192-7.

[5] D. Yurichev, "Reverse Engineering for Beginners". [Online]. Available: https://beginners.re/RE4B-EN.pdf. Accessed on: Oct. 23, 2020.

[6] X. Zhang, *Digital Forensic Education. An Experiential Learning Approac*h. Switzerland: Springer, 2020.

[7] R. Wong, *Mastering Reverse Engineering.* Birmingham, UK: Packt Publishing Ltd, 2018. ISBN 978-1-78883-884-9.

[8] A. Arnes, *Digital Forensics.* Hoboken, NJ: Wiley, 2018.

[9] B. Dang, A. Gazet, and E. Bachaalany, *Practical Reverse Engineering: x86, x64, ARM, Windows® Kernel, Reversing Tools, and Obfuscation*. Indianapolis, IN: Wiley, 2014.

[10] M. Sikorski, and A. Honig, *Practical Malware Analysis*. California, USA: No Starch Press, 2018. ISBN 978-1-59327-290-6.

[11] K. Monappa, *Learning Malware Analysis*. Birmingham, UK: Packt Publishing Ltd., 2018. ISBN: 978-1-78839-250-1.

[12] V. V. Pavlov, "Practice of downloading the operating system on digital media in virtual machine environment", *Visnyk Cherkaskogo derzhavnogo tekhnologichnogo universytetu*, no. 1, pp. 27-33, 2020. [in Ukrainian].

[13] Oracle VM VirtualBox. User Manual. [Online]. Available: https://www.virtualbox.org/manual/UserManual.html. Accessed on: Oct. 23, 2020.

[14] W. Stallings, Operating Systems: Internals and Design Principles, 9th ed. Pearson, 2018.

[15] M. Hale Ligh, A. Case, J. Levy, and A. Walters, *The Art of Memory Forensics: Detecting Malware and Threats in Windows, Linux, and Mac Memory*. Hoboken, NJ: Wiley, 2018. ISBN: 978-1-118-82509-9.

DOI: 10.24025/2306-4412.4.2020.214993

R. L. Ptashkin,
O. O. Hozhyi,
Yu. Yu. Obruch,
V. V. Pavlov,
R. A. Kalinichenko
e-mail: ndekc.ck@gmail.com
Cherkasy scientific research forensic centre MIA of Ukraine
Pasterivska st., 104, Cherkasy, 18000, Ukraine

RAM ANALYSIS AS ONE OF THE METHODS FOR COMPUTER FORENSICS

Modern methods of computer forensics usually do not use RAM analysis. This is due to the significant complexity of this task. But at the same time, RAM is an interesting object of research. RAM contains the data structures, the data related to processes running on the system, and the kernel. This includes virtual memory of all processes, virtual memory of the kernel, handles, mutexes, network connections, and other resources that are currently being used by all processes and the kernel. All these data and data structures are available in the memory dump. Other than that, RAM can contain a data about decryption keys. Such information will be valuable for computer forensics.

RAM analysis, as a separate method of computer forensics, generally requires to find solutions for some intermediate questions. The task of memory acquisition involves capturing the contents of RAM using a memory capture tool, which creates a memory dump file. The second step, memory analysis or forensics, involves an analysis of this memory dump file.

The base aim of this article is finding and testing tools for RAM forensics. We have analyzed the most respected forensics books and latest scientific papers to perform these tasks. After this we have made own research and tests of some software.

Forensics starts with data acquisition. Since we want to look at the memory, we need to use memory acquisition tools such as virtual environment. Using tools of virtual environment, we can take a complete dump of RAM, which includes both the user-mode memory space for all the processes and the kernel-mode memory as well. The best tool, according to the authors, is the Oracle VM VirtualBox, because it's useful, simple and free tool.

Analyzing the latest scientific works in the field of computer forensics, for the memory analysis the authors choose a tool, such as "volatility". "Volatility" is one of the popular pieces of open source software used. It is able to read suspended states of virtual machines. The advantage of such tools is that malware, such as rootkits, that try to hide themselves from user domains, can be extracted using memory forensic tools.

The proposed tools allow experts to make forensics analysis of RAM and obtain information that is not available from classical research methods. In this research we have tested functionality of virtual environment and various volatility plugins, that allow to get an idea of the events taking place in the operating system and the activities of some software. These methods and algorithms will be useful for the computer forensics in government forensic centers or for cybersecurity workers.

Keywords: *virtualization, VirtualBox, memory dump, RAM, volatility.*

Стаття надійшла 26.10.2020
Прийнято 02.12.2020

DOI: 10.24025/2306-4412.4.2020.214993