Єгоров С. В., к.т.н., доцент,
Шкварницька Т. Ю., к.т.н., доцент,
Яремич Т. І., ст. викладач
Національний авіаційний університет

МЕТОД ВИЯВЛЕННЯ ШКІДЛИВОГО КОДУ У ПРОГРАМНОМУ ЗАБЕЗПЕЧЕННЯ

Одним із самих розповсюджених та легких методів боротьби з хакерами це своєчасне оновлення програмного забезпечення, вірусних баз, інсталяція та налаштування брандмауера.

Все що пов'язано з оновленням програмного забезпечення – це реакція на вже виявлені загрози. Тому оновлення програмного забезпечення на надає захист від щойно виявлених загроз.

Єдиним ефективним методом виявлення нових загроз та створення ефективних локальних та мережевих сигнатур це аналіз щойно виявленого шкідливого програмного забезпечення.

Тому аналіз шкідливого програмного забезпечення є задачею актуальною та визначає напрям дослідження.

В [1] розроблено метод оцінки зрілості екосистем але не розглянуто оцінку шкідливого програмного забезпечення, яке здатне проникнути в цю екосистему. В результаті заражена вірусною програмою екосистема буде працювати неналежним чином, або на користь зловмисника.

В [2] побудована математична модель кіберзахисних дій, що дозволяє оцінити вартість кіберзахисту але не розглянуті методи забезпечення кіберзахисту.

В [3] були розглянуті технології для поліпшення безпеки польотної зони. Але автори [3] не врахували той факт, що програмні системи керування можуть контролюватися зловмисником. Для ефективного усунення зламу необхідно буде аналізувати код програми, за допомогою якої зламник контролює систему.

Метод виявлення шкідливого коду складається з таких етапів:

1. Аналіз та пошук строк;
2. Аналіз структури файлів;
3. Аналіз дизасембльованого тексту.

Аналіз та пошук строк. Строка в програмі являє собою деяку послідовність символів. Наприклад send. Якщо програма копіює будь-яку інформацію, звертається за url або IP адресою, виводить сповіщення, створює файл це ознаки того, що програма містить строки.

Здійснюючи пошук строк в програмі можна скласти уявлення про загальний функціонал програми. Припустимо, якщо програма звертається за url або IP адресою, то в програмі можна знайти відповідні строки, що містять url, IP адресу, номер порта. Строки зберігаються в програмі в форматі ASCII, або Unicode та закінчуються нульовим символом, що вказує на кінець строки.

Існує безліч програм, які здатні шукати строки в програмах. Наприклад, Strings з комплекту програм Марка Русиновича. Ця програма дозволяє шукати строки в файлах буд-якого типу ігноруючи при цьому контекст та форматування. Але при цьому може виявити строки, що такими не є.

На рис. 1 показано уривок результату пошуку строк в шкідливій програмі, що була виявлена на комп'ютері.

```
p@@
%\0@
[Window: %s - at %s](1)
1337                    (2)
127.0.0.1               (3)
[BACKSPACE]             (4)
[ENTER]                 (5)
[SPACE]                 (6)
[TAB]                   (7)
[SHIFT]                 (8)
[CONTROL]               (9)
SetWindowsHookExA       (10)
CallNextHookEx          (11)
GetKeyState             (12)
USER32.dll              (13)
WS2_32.dll              (14)
```

Рис. 1. Уривок результату пошуку строк в шкідливій програмі

Строка (1) являє собою формат для текстового сповіщення. Строки (2), (3) є портом та IP адресою відповідно. IP адреса була змінена навмисно для безпечного запуску програми. Інформація, що міститься в строках (4) – (9) наштовхує на думку, що цей файл є кейлоггером, що передає зібрану інформацію за зазначеною IP адресою. Це припущення підтверджується інформацією, що міститься в строках (10) – (14).

SetWindowsHookExA, що займає строку (10), встановлює визначену програмою процедуру перехоплення сповіщень в ланцюжок перехоплень. Може використовуватись для моніторингу системи щодо певних типів подій. Ці події пов'язані або з певним потоком, або з усіма потоками на одному робочому столі, з якого був викликаний потік. Буква «A» у кінці API функції означає, що ця функція працює зі строками ASCII.

CallNextHookEx, що займає позицію (11) передає інформацію про перехоплення до наступної процедури перехоплення в поточному ланцюжку перехоплень.

Процедура перехоплення може викликати цю функцію до або після обробки інформації про перехоплення. GetKeyState, що розташована в позиції (12) отримує статус вказаного віртуального ключа. Статус визначає те, що клавіша натиснута, не натиснута або перемикається (по черзі при кожному натисканні клавіші натискається і відпускається). Бібліотека USER32.dll містить елементи графічного інтерфейсу користувача та WS2_32.dll містить функції роботи з мережею.

Під час аналізу структури файлів необхідно вивчити вміст заголовків файлів та секцій.

Під час аналізу дизасембльованого тексту за допомогою дизасемблера вивчається вміст ключових фрагментів коду отриманого вихідного тексту.

Список використаних джерел

1. Svitlana Popereshnyak, Sergiy Grinenko, Olena Grinenko, Oleksandr Kovalenko, Tamara Radivilova. «Methods for Assessing the Maturity Levels of Software Ecosystems», International Workshop on Cyber Hygiene (CybHyg-2019) November 30, 2019, Kyiv, p. 251–261.
2. Aleksandr Litvinenko, Boris Maslovsky, Oleksiy Glazok, Anton Petrov, «Method of Optimal Planning of Cyberprotection Actions for a Corporate Information System», International Workshop on Cyber Hygiene (CybHyg-2019) November 30, 2019, Kyiv, p. 60–71.
3. Olena Kozhokhina, Olga Shcherbyna, Oleksii Chuzha, Serhii Yehorov, Maksim Iavich, Nikolay Churkin, «Informational Technology for the Improvement of Flight Zone Security», International Workshop on Cyber Hygiene (CybHyg-2019) November 30, 2019, Kyiv, p. 262-275.