Єгоров С. В., канд. техн. наук,
доцент кафедри комп'ютеризованих систем захисту інформації
Шкварницька Т. Ю., канд. техн. наук,
доцент кафедри комп'ютеризованих систем захисту інформації
Національний авіаційний університет

АНАЛІЗ ВІРУСНИХ ПРОГРАМ МЕТОДАМИ ЗВОРОТНЬОЇ ІНЖЕНЕРІЇ

На цей час існує багато досліджень в галузі методів аналізу шкідливого коду. Серед них можна відділити [1-3]. Але ці дослідження не містять повної інформації, що пов'язана з виявленням загроз в галузі кібербезпеки. На цей час існує декілька методів аналізу шкідливого програмного забезпечення. Їх можна розділити на дві групи: статичний аналіз та динамічний аналіз. Статичний аналіз, у свою чергу, містить у собі два методи: базовий статичний аналіз та розширений статичний аналіз.

Базовий статичний аналіз використовується для попередньої оцінки функціональності виявленої програми, поведінка якої є підозрілою. Цей вид аналізу базується на аналізі вмісту відкритих текстових строк, що містяться в тілі програми. При цьому базовий статичний аналіз також охоплює і заголовок файлу. Для більш ретельного аналізу файлу використовують розширений статичний аналіз. Такий вид аналізу містить два етапи: дизасемблювання файлу та аналіз отриманих результатів.

Динамічний аналіз дозволяє підтвердити або спростувати результати статичного аналізу. Під час динамічного аналізу на віртуальній машині запускається шкідлива програма, та за допомогою програм моніторінгу активності відстежують поведінку програми.

Ознаками вірусної активності можуть бути наступні ознаки:

– Підвищене споживання оперативної пам'яті;
– Підозрілий мережевий трафік;
– Несподіване закриття програми і при цьому програма може перезапуститися.

Це далеко не повний список ознак вірусної діяльності в операційній системі. Метод протидії загрозі від певного вірусу з'являється в антивірусних програмах тільки після виходу цього вірусу у світ, і не факт, що ліки від вірусу з'являться достатньо швидко, щоб запобігти широкому розповсюдженню загрози.

Тому необхідним, але недостатнім етапом для захисту комп'ютерів від вірусних атак є використання ліцензованого програмного забезпечення та своєчасне оновлення. Але при цьому ніхто не гарантує того, що сам виробник не стане жертвою зловмисників і не буде розповсюджувати вразливі оновлення, що містять, наприклад, бекдор. Однак, антивірусні програми лише частково компенсують несвоєчасні оновлення.

Для виявлення шкідливого файлу перш за все необхідно виявити будь-яку аномальну активність. Наступний крок – локалізація файлу, що спричинив цю аномальну активність. В цьому може допомогти додаток перегляду подій із групи програм «Засоби адміністрування Windows». За необхідністю можна скористатися, наприклад, набором програм Sysinternals Suite.

Після локалізації файлу, перед тим, як переходити до статичного аналізу файлу, необхідно визначити хеш файлу (наприклад MD5) та перевірити цей хеш в мережі. Можливо цю загрозу вже було виявлено та докладно описано.

Після цих кроків, за необхідністю, можна приступати до базового статичного аналізу, який складається з таких кроків:

– Аналіз вмісту заголовку файлу;
– Перевірка слідів обфускації;
– Аналіз вмісту строк файлу.

Якщо з точки зору аналітика було зібрано достатньо відомостей про шкідливий файл, необхідно здійснити динамічний аналіз. При цьому необхідно контролювати стан системи та використання її ресурсів, а також необхідно користуватися засобами моніторингу, які містять набір програм Sysinternals Suite.

Якщо даних, що були зібрані в результаті базового статичного аналізу недостатньо, то необхідно зробити розширений статичний аналіз. Для цього необхідно дизасемблювати файл та проаналізувати отриманий текст.

Динамічний аналіз лише підтверджує або спростовує результати базового статичного та розширеного статичного аналізу. Тому динамічний аналіз має сенс робити лише після того, як був здійснений один з видів або всі види статичного аналізу.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Панов А.С. Реверсинг и защита программ от взлома / А.С.Панов. – СПб.: БХВ-Петербург, 2006. – 256 с.
2. Антонов А. Е. Идентификация типа файла на основе структурного анализа / А. Е. Антонов, А. С. Федулов // Прикладная информатика. – 2013. – № 2 (44). – С. 68–77.
Microsoft Developer Network. – Режим доступу: https://msdn.microsoft.com/uk-ua.