“Изучение способов сокрытия вредоносных программ от антивирусных решений на примере сценариев Windows”

Студент : Улеско И.Н., ФИТ 0203

Руководитель: Пищик Б.Н., к.т.н., ст. научный сотрудник КТИ ВТ

Сценарии Windows – что это?

Сценарии Windows - высокоуровневые языки программирования для краткого описания действий, выполняемых системой.

Главное назначение сценариев - автоматизация повторяющихся действий.

Примеры задач, для решения которых удобно применять скрипты:

- Резервное копирование и восстановление;
- Установка ПО на компьютеры и настройка компьютеров;
- Настройка рабочей среды пользователя;
- Рассылка пользователям сообщений;
- Проверка и упорядочивание содержимого на серверах;
- Мониторинг работы служб и приложений, обеспечение немедленного реагирования на возникающие проблемы.

Преимущества использования сценариев Windows для злоумышленника

- ✓ Богатые возможности языков (VBScript, JScript);
- ✓ Лёгкость освоения языков;
- ✓ Простота использования сценариев;
- ✓ Присутствие сервера скриптов в Windows по умолчанию.

Для чего нужна эта работа?

- Основная проблема всех антивирусов:

  Обнаружение ранее неизвестного вредоносного ПО.

- Цель работы:

  Исследование средств и методов сокрытия, применяемых во вредоносных сценариях Windows, и формулировка ключевых принципов их работы.

Методы обнаружения вредоносного ПО

Реактивные технологии:

- Сигнатурный метод обнаружения вредоносного ПО;

Проактивные технологии:

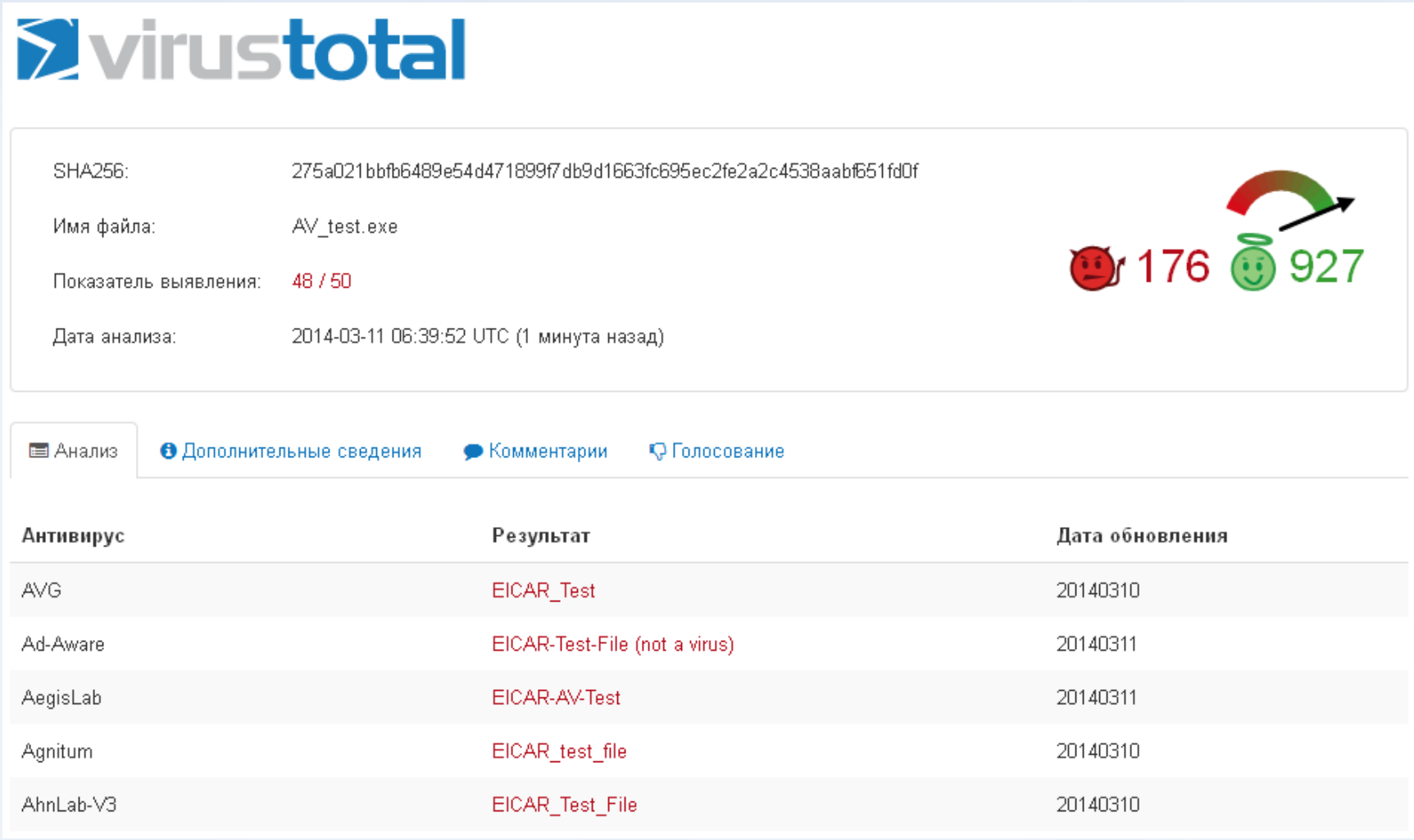
- Эвристический анализ;
- Эмуляция кода;
- Анализ поведения;
- Sandbox (“Песочница”).

Методы сокрытия вредоносного ПО, рассматриваемые в работе

- Обфускация кода сценария;
- Упаковка и шифрование кода сценария;
- Разбиение сценария на комплекс взаимосвязанных модулей, выполняющий функцию оригинального скрипта.

Используемые ресурсы

Сайт: https://www.virustotal.com/



Метод #1: Обфускация кода

- **Обфускация** - приведение исходного текста программы к виду, сохраняющему ее функциональность, но затрудняющему анализ и понимание алгоритмов работы.
- **Цель применения**: Противодействие сигнатурному методу обнаружения вредоносного ПО.

Техники обфускации, применяемые в сценариях:

- Случайные комментарии большого размера;
- Неявные значения строковых переменных;
- Увеличение конструкций;
- Мусорный код.

Метод #1: Обфускация кода

Пример кода с использованием методов обфускации:

```
1  Dim text
2  Set objNetwork = CreateObject("WScript.Network")
3  text = "User name: " & objNetwork.UserName
4  WScript.Echo text
```

После обфускации:

```
1  WScript.Echo StrReverse("resU") & _
2  chr(110)&chr(97)&chr(109)&chr(101)&chr(58)&chr(32) & _
3  CreateObject("WS"&Mid("09ZrCrIpqq(",5,4)&_
4  "t."&chr(80-2)&chr(202/2)&chr(100+16)&chr(119)&"ork").UserName
```

Статистика детектирования при использовании методов обфускации:

Метод обфускации	Детектируемость до применения метода	Детектируемость с применением Метода
Неявные значения строковых переменных	14/50	6/50
Большие случайные комментарии	14/50	8/50
Увеличение конструкций	14/50	13/50
Мусорный код	14/50	14/50

Метод #1: Обфускация кода

Уязвимость метода:

Сохранение семантики кода, несмотря на изменения в его синтаксисе.

Решение проблемы обнаружения:

- Игнорирование комментариев в коде при сигнатурном анализе
- Преобразование переменных и констант, которые являются функциями от зафиксированных переменных, в результат этих самых функций.
- Предварительное вычисление передаваемых параметров в функции.

Метод #2: Упаковка кода

- **Упаковка кода** – преобразование исполнимого файла (например, сжатие) и прикреплении к нему кода, необходимого для распаковывания и исполнения содержимого файла.
- **Цель применения**: Противодействие сигнатурному методу обнаружения вредоносного ПО.

Пример упакованного кода скрипта:

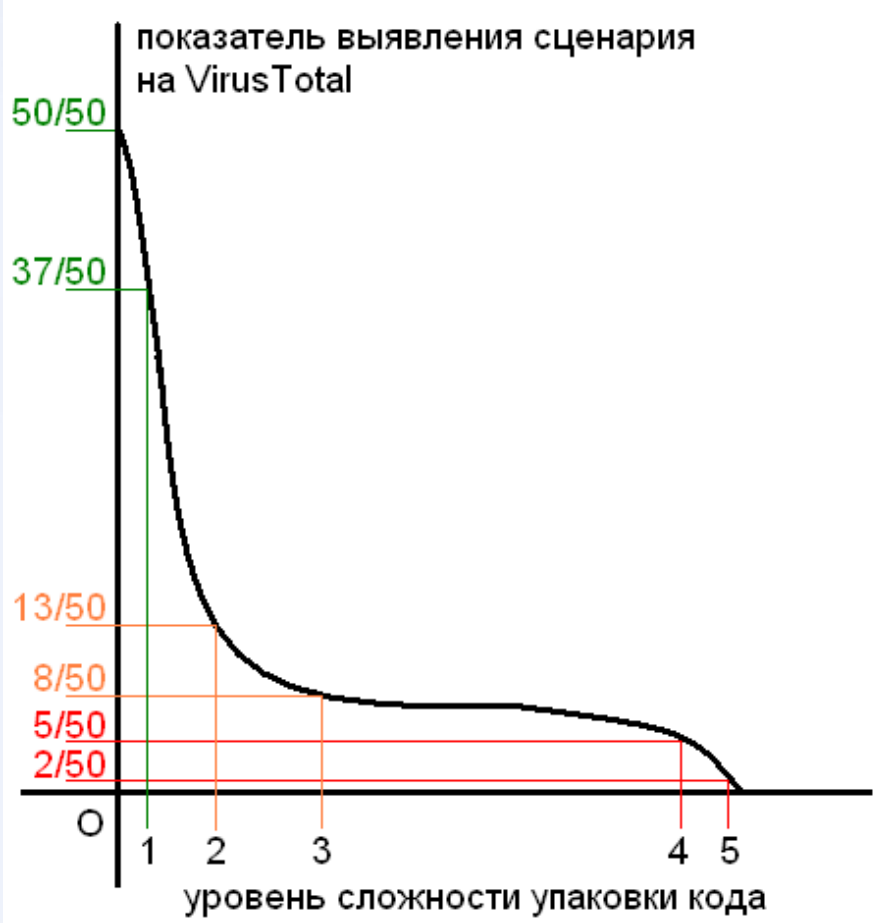
```
1  '    Первоначальный вариант:
2  WScript.Echo "Hello, world!"
```

```
1  '            Преобразованный вариант:
2  ' 1) Зашифрованный код: Wscript.Echo "Hello, world!"
3  Script2 = "ц»«єЎёjжЌ« §икЂ-¤¤§диї§ем¬йк"
4  ' 2) Процесс расшифровки кода скрипта
5  Script1 = ""
6  For i=1 To Len(Script2)
7      Script1 = Script1 & Chr( Asc(Mid(Script2,i,1)) Xor 200)
8  Next
9  ' 3) Запуск расшифрованного кода скрипта
10 Execute Script1
```

Метод #2: Упаковка кода

Эффективность обнаружения упакованного кода:

1. Оригинальный код скрипта
2. Простейшая упаковка всего кода
3. Упаковка кода с использованием более сложных механизмов шифрования
4. **Шифрование каждой операции скрипта отдельно**
5. **Шифрование каждой операции скрипта отдельно и разными методами**



Метод #2: Упаковка кода

Уязвимость метода:

Третий этап – запуск расшифрованного кода. Для выполнения кода в параметр запускающей функции всегда должен передаваться незашифрованный код.

Решение проблемы обнаружения:

- Вычисление параметров функций в коде, которые используются для исполнения команд сценариев, и их последующая проверка антивирусом.

Метод #3: Разбиение сценария на модули

- **Разбиение сценария на модули** – преобразование кода на совокупность примитивных связанных друг с другом модулей, которые в итоге выполняют тот же набор действий, что и первоначальный скрипт.
- **Цель применения**: Противодействие сигнатурным и проактивным технологиям обнаружения вредоносного ПО.

Пример использования разбиения сценария на модули:

(вывод «Hello, world!»)

```
' first.vbs
Arg1 = "world!"
CreateObject("WScript.Shell").Run "second.vbs "&Arg1

' second.vbs
Arg2 = "Hello, " & WScript.Arguments(0)
CreateObject("WScript.Shell").Run "last.vbs "&""""&Arg2&""""

' last.vbs
WScript.Echo WScript.Arguments(0)
```

Метод #3: Разбиение сценария на модули

Статистика обнаружения вредоносного сценария после использования разбиения на модули:

Изменение кода	Показатель выявления на VirusTotal
Исходный код без изменений	36/50
Половина процедур вынесена в отдельный модуль	19/50
Все процедуры в коде вынесены в свой отдельный модуль	10/50
Все процедуры в коде имеют свой модуль, основной код также разделён на несколько модулей	7/50

Метод #3: Разбиение сценария на модули

Уязвимость метода:

При создании цепочки модулей всегда указывается путь к следующему модулю и передаваемые ему параметры.

Решение проблемы обнаружения:

- Сборка всех модулей в единый файл по используемым параметрам функций, отвечающих за связь модулей между собой.
- Проверка получившегося файла антивирусом.

Результаты работы

1. Были изучены образцы вредоносных сценариев, которые имели минимальные показатели выявления в тестировании VirusTotal;
2. Выявлены основные методы, использующиеся для сокрытия сценариев от антивирусов;
3. Исследован функционал языков сценариев, который необходим для реализации методов сокрытия вредоносного ПО и их вариантов усовершенствования;
4. Проведены тестирования методов для изучения их эффективности сокрытия от антивирусов;
5. Для наиболее эффективных методов, которые создавали минимальные показатели выявления, выявлены основные уязвимости, которые позволят в будущем антивирусам успешнее бороться с вредоносными сценариями.

Спасибо за внимание!