АВТОМАТИЗИРОВАННЫЙ АНАЛИЗ ВРЕДОНОСНОГО ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ С ПРИМЕНЕНИЕМ РЕКУРРЕНТНЫХ НЕЙРОННЫХ СЕТЕЙ

Вешторт А. В.
Кафедра интеллектуальных информационных технологий, Белорусский государственный университет информатики и радиоэлектроники
Минск, Республика Беларусь
E-mail: ales.veshtort@gmail.com

Статья описывает метод автоматизированного анализа вредоносного программного обеспечения, позволяющий значительно снизить нагрузку на вирусного аналитика, путем предварительной обработки информации о поведении вредоносного образца внутри изолированной среды исполнения приложений.

Введение

В настоящее время проблема автоматизированного анализа вредоносного программного обеспечения стоит необычайно остро из-за огромных темпов создания новых вредоносных образцов. По данным Virustotal.com [1], только на этот ресурс еженедельно загружается до 400 тыс. образцов вредоносного программного обеспечения.

Очевидно, что ручной анализ такого количества образцов требует больших затрат человеческих и временных ресурсов. Таким образом возникает потребность в автоматизации процесса анализа вредоносного программного обеспечения.

Существующие программные решения лишь частично справляются с задачей автоматизированного анализа вредоносного программного обеспечения, оставляя значительный пласт работы аналитику вредоносных образцов.

Такие решения можно условно разделить на три группы по используемым ими методам анализа вредоносного программного обеспечения: программное обеспечение для сопоставления с образцом; средства статического анализа ПО (интерактивные дизассемблеры); автоматизированные системы динамического анализа ПО («песочницы»).

Сопоставление с образом является наиболее базовым методом автоматизированного анализа вредоносного программного обеспечения, направленным на обнаружение того подмножества вредоносных образцов, которые обладают набором известных программных артефактов. Таким образом, программное обеспечение, основанное на этом методе анализа, фактически бесполезно при исследовании новых, ранее не исследованных, образцов вредоносного ПО, и может быть использовано только в комбинации с другими средствами анализа.

Существующие средства статического анализа (дизассемблеры) являются программными средствами общего назначения, способными эффективно восстанавливать исходный низкоуровневый код и структуру программного образца, но не предназначенными конкретно для анализа вредоносных образцов, поэтому задача поиска вредоносного кода перекладывается на плечи аналитика. Кроме того сама сущность статического анализа не позволяет обнаружить изменения кода программного образца в процессе его исполнения.

Программные средства автоматизированного динамического анализа на данный момент представлены системами автоматизированного анализа вредоносного программного обеспечения на основе изолированной среды («песочницы»). Такие системы позволяют получить лишь общую поведенческую информацию об образце: список созданных, удаленных или модифицированных файлов, список созданных ключей реестра Windows, сетевых соединений и т.д. Сама же задача обнаружения характера изменений и исследования переданных данных делегируется аналитику вредоносного программного обеспечения.

I. Предлагаемое решение

В данной работе предлагается новый метод автоматизированного анализа вредоносного программного обеспечения, основанный на дополнительной обработке данных, извлекаемых системами динамического анализа на основе изолированной среды.

В частности, предлагается рассматривать список вызовов функций WinAPI как последовательность взаимосвязанных событий, среди которых можно выделить те подпоследовательности, которые соответствуют вредоносному поведению.

Таким образом, задача анализа вредоносного образца сводится к задаче маркировки последовательностей, которая может быть решена с помощью рекуррентной нейронной сети.

II. Кодирование данных

Несмотря на то, что категориальные данные в числовом виде могут быть поданы на вход нейронной сети, такой подход может привести к

многочисленным неточностям и ошибкам классификации [2]. Потому предлагается предварительно преобразовать входные данные в one-hot вектора [3].

Для кодирования вызванных функций WinAPI предлагается использовать вектора, составленные из $1 + p$ one-hot векторов, где p - максимальное число параметров функции.

Первый вектор имеет длину t, где t - общее число возвращаемых функцией типов данных; каждому типу данных предварительно присваивается порядковый номер $j, 0 \leq j < t$. При этом если a_i - i-я составляющая вектора, а j - номер возвращаемого функцией типа данных, то $a_i = 1$, если $i = j$, иначе $a_i = 0$.

Все последующие вектора имеют длину $p_t + p_n + 2$ и составлены из двух векторов c и d длиной $p_t + 1$ и $p_n + 1$ соответственно, где p_t - общее число типов данных параметров функции, p_n - общее число имен параметров функции; каждому типу данных параметров функции предварительно присваивается порядковый номер $j, 0 \leq j < p_t$, каждому из имён параметров функции предварительно присваивается порядковый номер $k, 0 \leq k < p_n$.

При этом если c_i - i-я составляющая вектора c, а j - номер типа данных параметра функции, то $c_i = 1$, если $i = j$, иначе $c_i = 0$. Аналогично, если d_i - i-я составляющая вектора d, а k - номер имени параметра функции, то $d_i = 1$, если $i = k$, иначе $d_i = 0$.

Составляющие векторов c и d под номерами p_t и p_n устанавливаются в единицу для «отсутствующих» параметров для функций с общим количеством параметров меньше максимального числа параметров функции. Например, при кодировании функции с p_i параметрами при максимальном числе параметров p_{max}, к функции будет добавлено $p_{max} - p_i$ «отсутствующих» параметров.

III. Архитектура нейронной сети

Для обработки закодированной последовательности вызовов предлагается использовать двунаправленные нейронные сети, т.к. вызов практически любой функции WinAPI может трактоваться по-разному в зависимости как от предыдущего, так и последующего контекста.

Традиционные рекуррентные нейронные сети обладают крайне ограниченным диапазоном запоминаемых событий. Эта проблема связана с высокой степенью влияния новых входных данных на нейроны скрытого слоя, что приводит к экспоненциальному уменьшению влияния контекстных нейронов предыдущих шагов на вывод сети. Этот эффект называется проблемой исчезающего градиента [4].

Для решения этой проблемы применяются LSTM сети. В таких сетях нейроны скрытого слоя имеют особую структуру, изображенную на рисунке 1.

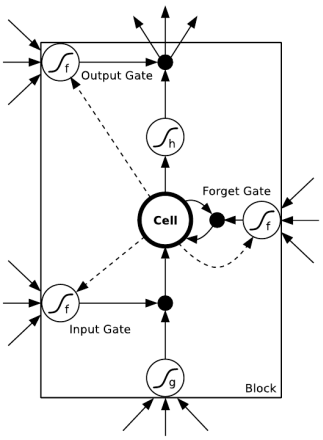

Рис. 1 – Структура LSTM-нейрона

Шлюзы представляют собой нелинейные сумматоры, которые получают сигнал от соседних слоев сети и контролируют активацию нейрона с помощью умножителей (черные точки на рисунке 1). Входной и выходной шлюзы перемножают входные и выходные сигналы нейрона, результат перемножается с предыдущим состоянием нейрона в шлюзе забвения. Функция активации шлюза f, как правило, является сигмоидной. Выходной сигнал нейрона, отправляемый следующему слою сети, формируется умножителем выходного шлюза [4].

Таким образом, в качестве нейронной сети для маркировки последовательностей вызовов была выбрана рекуррентная нейронная сети с одним с скрытым LSTM слоем.

IV. Заключение

В данной статье был представлен метод автоматизированного анализа вредоносного программного обеспечения, позволяющий значительно уменьшить долю человеческого участия в процессе анализа вредоносного программного обеспечения. В настоящий момент базовый прототип системы, основанной на этом методе, используется в антивирусной лаборатории ОДО «ВирусБлокАда».

1. Virustotal.com - Статистика [Электронный ресурс] Режим доступа: https://www.virustotal.com/en/statistics/. – Дата доступа: 31.08.2018.
2. Николаенко, С. И. Глубокое обучение / С. И. Николенко, А. А. Кадурин, Е. О. Архангельская – СПб.: Питер, 2018 - 480 с.
3. Geron, Aurelien. Hands-On Machine Learning with Scikit-Learn and TensorFlow / Aurelien Geron – 2017. – Vol. 1 – 797 p.
4. Graves, Alex. Supervised Sequence Labelling with Recurrent Neural Networks / Alex Graves – 2010. – Vol. 1 – 137 p.