Разработка метода обнаружения вредоносных ПО с помощью графа системных вызовов с использованием машинного обучения

Насер Р.З. Насер

Южный федеральный университет, Ростов-на-Дону

Аннотация: Данная статья посвящена решению задачи исследования и обнаружения вредоносного ПО. Метод, реализованный в работе, позволяет динамически обнаруживать вредоносные программы для Android с помощью графов системных вызовов с использованием графовых нейронных сетей. Задача данной работы заключается в создании компьютерной модели для метода, разработанного с целью обнаружения и исследования вредоносного ПО. Исследования на этой теме имеют важность в математическом и программном моделировании, а также в применении алгоритмов управления системными вызовами на устройствах Android. Оригинальность данного направления заключается в постоянном совершенствовании подходов в борьбе с вредоносным ПО, а также в ограниченной информации об использовании компьютерной симуляции для исследования таких явлений и особенностей в мире.
Ключевые слова: системные вызовы, андроид, вирус, вредоносное ПО, нейронные сети, искусственный интеллект, нечеткая логика.

Введение

В настоящее время количество вредоносных программ для Android [1] резко возросло, создавая критические угрозы безопасности [2].

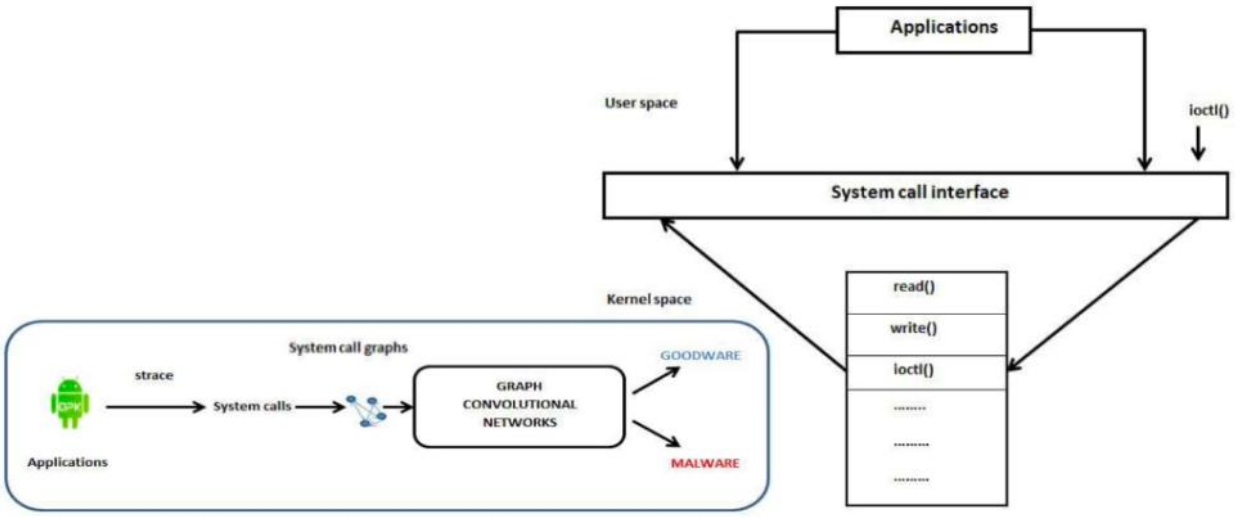

Рис. 1. – Обнаружение вредоносных программ с помощью ГСС, выполнение системного вызова Android

Авторы вредоносных программ [3] теперь используют различные методы запутывания, чтобы избежать их обнаружения. Среди различных

функций системные вызовы [4] являются одной из основных функций, используемых для обнаружения вредоносных программ. Хотя запутанное вредоносное ПО [5] использует различные методы, чтобы скрыть свою вредоносную природу, зависимости между системными вызовами могут выявить их вредоносную природу.

Разработка метода обнаружения вредоносных программ

Предлагается использовать метод машины опорных векторов для обнаружения вредоносных программ. Однако при высоком уровне шума этот метод может допускать ошибки, поэтому был добавлен аппарат нечеткой логики для дополнительной классификации и увеличения точности. Для обучения была использована выборка из 67 программных векторов, в которую были включены 33 измененных вектора программ для усложнения обнаружения вредоносного ПО. Для улучшения классификации были добавлены дополнительные признаки в виде функций принадлежности аппарата нечеткой логики, включая результат классификации методом опорных векторов. Затем была создана база правил для правильной работы аппарата нечеткой логики. Метод обнаружения вредоносных программ включает анализ файлов для извлечения векторных признаков программы [6] и использование программного вектора в качестве входа для классификатора. Рассмотрим набор данных для эксперимента $\{(x_1, y_1),...,\{(x_m, y_m)\}\}$, для каждого x присутствует соответствующее y. Значения для переменной y варьируются в диапазоне от 1 до –1. Весь набор диапазонов для классов имеет определенный вектор численных значений рассматриваемых характеристик $x_i = (x_i^1, x_i^2,..., x_i^q)$. Данные характеристики отображаются в двоичном формате. Например, x_i^j представляет собой численный параметр j-й характеристики в рамках i-го значения x. Для решения проблемы классификации с использованием метода Куна-Таккера [7] необходимо найти точку

минимума-максимума функции Лагранжа [8], что равносильно решению двойственной задачи. Эта проблема может быть определена как проблема квадратичного программирования. Для этого применим следующие переменные:

$$\begin{cases} -L(\lambda) = -\sum_{i=1}^{n}\lambda_i + \frac{1}{2}\sum_{j=1}^{n}\sum_{i=1}^{n}\lambda_i\lambda_j y_i y_j \times \\ \times \exp(- < x_i - x_j, x_i - x_j > /(2\sigma^2) - b)) \\ \sum_{i=1}^{n}\lambda_i y_i = 0 \\ 0 \le \lambda_i \le T, i = 1, n \end{cases}, \quad (1)$$

где λ_i – двойное значение; x_i – элемент из нашего эксперимента; y_i – численное значение из рассмотренного выше диапазона, которое в свою очередь требуется для определения принадлежности x_i к определенному классу из рассмотренного выше диапазона выборки; T – регуляризация элемента выборки; n определяет число x_i в диапазоне выборки $i = 1, n$; $k(x_i, x_j) = \exp(-\prec x_i - x_j, x_i - x_j \succ /(2\sigma^2))$ – радиальная базисная функция информационной системы. При обучении информационной системы опорные векторы выделяются из диапазона выборки при наличии у них характеристик элементов их x. Отбор опорных векторов основан на значениях двойственных переменных λ, которые отличны от нуля (λ ≠ 0). Опорные векторы [9], находящиеся ближе всего к гиперплоскости, содержат всю необходимую информацию о разделении классов. Если квадратичная задача решена, то для классификации произвольного объекта используется соответствующее правило на основе значения переменной λ.

$$\alpha(z) = sign\sum_{i=1}^{m}\lambda_i y_i \exp(-\prec x_i - x_j, x_i - x_j \succ /(2\sigma^2) - b). \quad (2)$$

Результаты классификации и дополнительные признаки, которые представлены в виде функций, выносятся на приемку сопровождения "Системы блочных решений". После анализа результатов на основе правил,

полученный результат отображается в процентах и категориях. Разработанный метод запускает Android-приложения в эмуляторе и генерирует тысячи псевдослучайных событий. После этого извлекаем системные вызовы с помощью утилиты strace [10]. Наконец, уточняем системные вызовы, удаляя из них аргументы, а затем выбираем системные вызовы, которые выполняют управление файлами и доступ к сети, так как вредоносным программам требуются вызовы системы управления файлами для доступа к конфиденциальным ресурсам. Пусть V = {V1,...,Vn} обозначает множество соответствующих системных вызовов. Обозначим через S1, S2, ... последовательность соответствующих системных вызовов, сгенерированных приложением, где Si∈V. Здесь системный вызов Si определяет появление следующего системного вызова Si+1, так как между системным вызовом в последовательности системных вызовов существует управляющая зависимость. Следовательно, можем смоделировать последовательность системных вызовов как орграф системных вызовов G = (V, E), где V = {V1,..., Vn} обозначает множество соответствующих системных вызовов, а E обозначает множество направленных ребер. Ребро существует в E когда системный вызов V_j стоит сразу после V_i в последовательности системных вызовов S1, S2, ... Таким образом, орграф системных вызовов G можно использовать для представления поведения приложения. Затем используем ГСС, чтобы классифицировать, являются ли графы системных вызовов вредоносными или нет.

Литература

1. Менциев А.У., Анализ фишинговых атак как вида социальной инженерии. Наука и молодежь Всероссийская научно-практическая конференция студентов, молодых ученых и аспирантов. 2016. С. 391-394.

2. Инвестиции в будущее: искусственный интеллект. – URL: vc.ru/finance/46776-investicii-v-budushchee-iskusstvennyy-intellekt.

3. Искусственный интеллект - угроза или помощник для человечества? – URL: bbc.com/russian/features-38931070.

4. Килюшева, Е.С., Гнедин, Е.В. (2017). Как хакеры атакуют вебприложения: боты и простые уязвимости. Securitylab.ru. URL: securitylab.ru/analytics/485977.php.

5. Черемных, В.А. (2017). Виды хакерских атак. It-black.ru. URL: itblack.ru/vidy-khakerskikh-atak/.

6. Менциев А.Ю., Джангаров А.И. Угрозы безопасности VoIP // Инженерный вестник Дона, 2019, №1. URL: ivdon.ru/ru/magazine/archive/n1y2019/5636.

7. Менциев А.У., Супаева Х.С. Технологии VoIP // Инженерный вестник Дона, 2019, №1. URL: ivdon.ru/ru/magazine/archive/n1y2019/5609.

8. Пылаева Е.В. Разработка модели управления ИТинфраструктурой кредитной организации на основе архитектурной модели IT4IT // Инженерный вестник Дона, 2018, №2. URL: ivdon.ru/ru/magazine/archive/N2y2018/4927.

9. Попов А.Г., Шикин Е.Е. Администрирование Windows с помощью WMI и WMIC. СПб.: БХВ-Петербург, 2004. 746 с.

10. Сироткин А.В., Брачун Т.А., Бархатов Н.И. Моделирование приоритетного управления информационными потоками с использованием сокетов // Инженерный вестник Дона, 2012, №4. URL: ivdon.ru/ru/magazine/archive/n4p1y2012/1192.

References

1. Menciev A.U. Nauka i molodezh` Vserossijskaya nauchnoprakticheskaya konferenciya studentov, molody`x ucheny`x i aspirantov. 2016. pp. 391-394.

2. Investicii v budushhee: iskusstvenny`j intellekt. [Investment in the future: artificial intelligence]. URL: vc.ru/finance/46776-investicii-v-budushchee-iskusstvennyy-intellekt.

3. Iskusstvenny`j intellekt – ugroza ili pomoshhnik dlya chelovechestva? [Artificial intelligence - a threat or a helper for humanity?]. URL: bbc.com/russian/features-38931070.

4. Kilyusheva, E.S., Gnedin, E.V. (2017). Kak xakery` atakuyut vebprilozheniya: boty` i prosty`e uyazvimosti. [How hackers attack web applications: bots and simple vulnerabilities]. URL: securitylab.ru/analytics/485977.php.

5. Cheremny`x, V.A. (2017). Vidy` xakerskix atak. [Types of hacker attacks]. URL: itblack.ru/vidy-khakerskikh-atak/.

6. Mentsiev A.U., Dzhangarov A.I. Inzhenernyj vestnik Dona, 2019, №1. URL: ivdon.ru/ru/magazine/archive/n1y2019/5636.

7. Mentsiev A.U., Supaeva Kh.S. Inzhenernyj vestnik Dona, 2019, №1. URL: ivdon.ru/ru/magazine/archive/n1y2019/5609.

8. Py`laeva E.V. Inzhenernyj vestnik Dona, 2018, №2. URL: ivdon.ru/ru/magazine/archive/N2y2018/4927.

9. Popov A.G., Shikin E.E. Administrirovanie Windows s pomoshh`yu WMI i WMIC. [Windows Administration with WMI and WMIC]. SPb.: BXV-Peterburg, 2004. 746 p.

10. Sirotkin A.V., Brachun T.A., Barxatov N.I. Inzhenerny`j vestnik Dona, 2012, №4. URL: ivdon.ru/ru/magazine/archive/n4p1y2012/1192.