УДК 004.052.32

ПРИМЕНЕНИЕ БОЛЬШИХ ЯЗЫКОВЫХ МОДЕЛЕЙ ДЛЯ ГЕНЕРАЦИИ КОНФИГУРАЦИИ ДЛЯ TAINT АНАЛИЗА

Стукалов Д.О. (ИТМО)
Научный руководитель – кандидат технических наук, доцент Ицыксон В.М. (ИТМО)

Введение. Современное программное обеспечение обычно содержит множество ошибок. Некоторые ошибки приводят к уязвимостям, которые, в свою очередь, проявляются в виде утечек конфиденциальных данных, нарушений целостности данных или в виде отказа в обслуживании. Одним из способов проверки программного обеспечения на наличие ошибок является статический анализ. Одним из самых популярных методов статического анализа является анализ потоков данных (dataflow analysis), а одной из самых простых и производительных его разновидностей является taint-анализ [1]. Taint-анализ используется для контроля некорректного распространения данных по программе. Попадание непроверенных или чувствительных данных в определенные части программы может привести к уязвимости. Современные системы taint-анализа требуют сложного конфигурирования, от качества которого зависит насколько эффективен будет анализ. Данная работа посвящена разработке методов создания конфигураций taint-анализа с привлечением искусственного интеллекта.

Основная часть. Ключевая идея taint-анализа заключается в том, что любая переменная, в которую могут попасть чувствительные данные, представляет собой потенциальную угрозу безопасности, если эти данные при определенном ходе выполнения программы могут попасть за пределы программы. Если потенциально опасная переменная используется в каком-либо выражении, то значение этого выражения тоже становится подозрительным. Существующий алгоритм отслеживает ситуации, когда переменные, отмеченные как подозрительные, используются при выполнении опасных команд, например, при прямых запросах к базе данных, записи в файлы и т.п.

Для taint-анализа требуется конфигурация, в которой пользователь назначает одну из следующих ролей каждому методу в программе. Источник (source) – функция, возвращающая чувствительные данные. Например, это может быть метод чтения пользовательского ввода или метод получения параметра входящего HTTP-запроса. Конфигурация метода определяет метку, которая будет ассоциирована с переменной. Название метки обычно отражает суть чувствительных данных (например, SYSTEM_INFO, PASSWORD и т.п.). Передатчик (pass) – функция, которая помечает возвращаемое значение с учетом анализа имеющихся меток в ее аргументах. В зависимости от реализации метки могут ассоциироваться не только с возвращаемым значением функции, но и с вызывающим объектом или с другими аргументами функции. Очиститель (cleaner) – функция, удаляющая заданный набор меток из переданных аргументов. Чаще всего это – функция, проверяющая корректность входных данных. Сток (sink) – функция, появление в которой помеченных данных свидетельствует о наличии уязвимости. Алгоритм анализирует программы, пытаясь обнаружить пути между источниками и стоками, используя конфигурации источников, передатчиков, очистителей и стоков.

От качества конфигураций напрямую зависит эффективность taint-анализа. При этом создание конфигураций является сложной, рутинной задачей. В данной работе мы используем большие языковые модели для генерации конфигураций для taint-анализа. Такой выбор был сделан в силу положительного опыта применения таких моделей в других задачах генерации данных.

Для создания конфигураций в работе применяются два подхода в формировании запросов: chain-of-thought [2] и few-shot [3]. Первый из подходов заключается в генерировании промежуточных шагов рассуждений, что позволяет значительно улучшить способности большой языковой модели выполнять сложные рассуждения. Второй подход заключается в предоставлении нескольких примеров входных и ожидаемых выходных данных, что позволяет

большой языковой модели лучше находить паттерны. В ходе работы разработан инструмент, который реализует комбинацию обоих методов формирования запросов.

Выводы. В результате выполнения данного проекта был успешно разработан прототип инструмента, который позволяет сериализовать Java-классы в формат JSON для формирования запросов к большой языковой модели. Инструмент обрабатывает полученные от большой языковой модели ответы, преобразовывая их в taint-конфигурации, что позволяет упростить разработку, так как избавляет от необходимости ручного создания файлов конфигурации. Проведенные эксперименты показали применимость использования больших языковых моделей в задаче формирования конфигураций для taint-анализа.

Список использованных источников:

1. Edward J. Schwartz, Thanassis Avgerinos, David Brumley. All You Ever Wanted to Know About Dynamic Taint Analysis and Forward Symbolic Execution (but might have been afraid to ask) [Электронный ресурс] // ieeexplore.ieee.org. 2022. Дата обновления: 08.06.2010. URL: https://ieeexplore.ieee.org/document/5504796 (дата обращения: 01.02.2024).

2. Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, Denny Zhou. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models [Электронный ресурс] // arXiv.org. 2022. Дата обновления: 10.01.2023. URL: https://arxiv.org/abs/2201.11903 (дата обращения: 01.02.2024).

3. Tom B. Brown et al. Language Models are Few-Shot Learners [Электронный ресурс] // arXiv.org. 2020. Дата обновления: 22.07.2020. URL: https://arxiv.org/abs/2005.14165 (дата обращения: 01.02.2024).