ПРИМЕНЕНИЕ ТЕХНОЛОГИЙ СИМВОЛЬНО–КОНКРЕТНОГО ТЕСТИРОВАНИЯ И МАШИННОГО ОБУЧЕНИЯ ДЛЯ АВТОМАТИЗАЦИИ ОБНАРУЖЕНИЯ УЯЗВИМОСТЕЙ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

А.А. Мещеряков

Научный руководитель: доцент, к.т.н. В.С. Горбатов

Национальный исследовательский ядерный университет «МИФИ»,

Россия, г. Москва, Каширское ш., 31, 115409

E-mail: aameshcheryakov@mephi.ru

AUTOMATION VULNERABILITY DISCLOSURE USING CONCOLIC TESTING AND MACHINE LEARNING

A.A. Meshcheryakov

Scientific Supervisor: Associate Professor, Ph.D. V.S. Gorbatov

National research nuclear university "MEPhI",

Russia, Moscow, Kashirskoe shosse, 31, 115409

E-mail: aameshcheryakov@mephi.ru

Abstract. *Security testing is important stage of software development life cycle. However, security testing requires considerable time from highly skilled security experts. The aim of the article is to describe techniques for reducing the number of false positives and false negatives in the automation vulnerability disclosure process. This paper is about an approach for software vulnerabilities discovery using concolic testing and machine learning techniques. Machine learning techniques are used to reduce the number of execution paths during concolic testing. This approach can be used to automate security testing. In this paper, security test cases and traces of previous version of software and similar software are used for training dataset for our models. This scheme of automation vulnerability disclosure will be used to build automation security testing system of software.*

Введение. Безопасность остается одним из главных приоритетов при разработке программного обеспечения (ПО) на протяжении последних лет. Однако, как показывает практика большинство разрабатываемых приложений уязвимо к тем или иным атакам. Так исследование безопасности финансовых приложений [1], проведенное компанией Positive Technologies, показало, что все анализируемые системы дистанционного банковского обслуживания содержали уязвимости по крайней мере среднего уровня риска, при этом почти в каждой из систем (90%) были обнаружены критически опасные уязвимости. Схожую статистику по наличию уязвимостей имеют и приложения, написанные для других отраслей экономики и сфер деятельности.

Для получения приемлемого уровня защищенности разрабатываемых приложений компании внедряют процессы обеспечения безопасности в жизненных цикл разработки ПО (Security Development Lifecycle - SDL) [2]. Одной из обязательных практик SDL является анализ безопасности ПО. Ручной анализ безопасности требует больших затрат времени высококвалифицированных специалистов, что приводит к повышению стоимости разрабатываемого ПО и увеличению сроков выпуска продукта и его

новых версий. В связи с этим для повышения эффективности работы специалистов компаниями применяются различных инструменты автоматизации обнаружения уязвимостей. На текущий момент сформировались три основных подхода: статический, динамический и смешанный анализ. Каждый из указанных подходов имеет свои недостатки, затрудняющие построение на его основе системы автоматизации обнаружения уязвимостей, описанные в работе [3]. Так обнаружение уязвимостей методом статического анализа подвержено ошибкам первого (ложноположительные срабатывания) и второго рода (пропуск ошибок безопасности). В то время как значительным недостатком динамического анализа является значительные временные затраты на обеспечение качественного покрытия кода анализируемой программы.

Материалы и методы исследования. Одним из подходов позволяющий объединить статический и динамический анализ является символьно-конкретное тестирование (concolic testing) [4]. Символьно-конкретное тестирование - это метод поиска дефектов, который на основе символьного выполнения осуществляет генерацию тестовых данных с целью максимизировать процент покрытия кода для увеличения вероятности нахождения уязвимостей ПО.

Одним из ограничений метода символьно-конкретного тестирования является проблема экспоненциального роста количества проходимых путей, что затрудняет его применение для анализа безопасности реальных систем. В данной работе предлагается применение технологии машинного обучения для уменьшения числа символьных переменных и сокращения количества тестовых наборов, что должно ограничить количество проходимых путей и соответствующее время анализа.

Использование технологии машинного обучения активно исследуется для решения проблем функционального тестирования ПО (например, в работе [5]). Однако, применение этой технологии для тестирования безопасности пока мало изучено. В данном работе предлагается использование алгоритмов машинного обучения для повышения качества процесса тестирования безопасности программного обеспечения. Применение подобного подхода позволит повысить уровень безопасности программного обеспечения и сократить затраты эксперта на анализ результатов работы средств тестирования. Ключевая идея состоит в автоматической классификации обработанных трасс выполнения и символьных переменных на основе данных, полученных при тестировании прошлых версий программного обеспечения, а также других приложений схожей функциональности. Таким образом, все наборы тестовых данных делятся на два класса в зависимости от возможности возникновения на конкретном наборе дефекта безопасности.

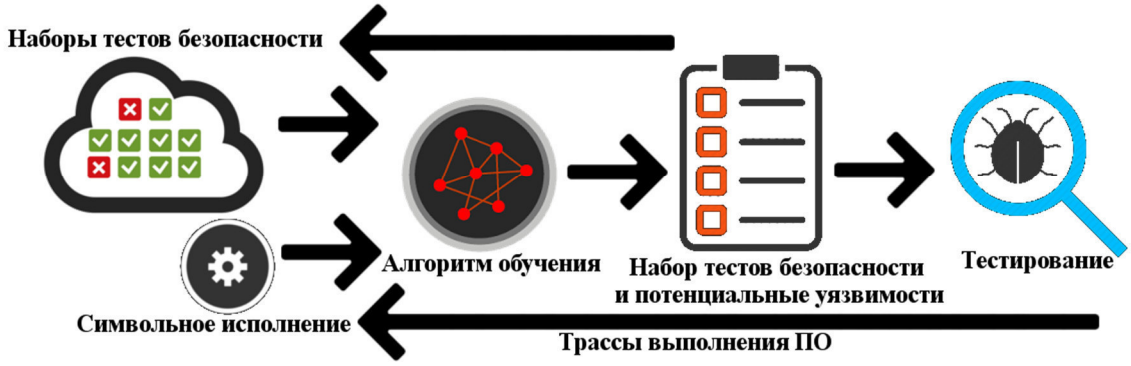

Рис. 1. Общая схема работы системы тестирования.

Для определения уязвимостей ПО используется подход, описанный в [6]. Данный подход основывается на включении в набор данных для обучения трасс выполнения ПО и позволяет определить наиболее вероятные уязвимости. Для классификации путей выполнения ПО используется сверточная нейронная сеть. Трасса исполнения программы разбивается на последовательность лексем, представляющих собой вызовы различных функций. Лексемы объединяются в три-граммы и подаются на вход соответствующих нейронов. Выходных значением нейронной сети является оценка вероятности соответствия данной трассы какой-либо уязвимости.

Результаты. Было проведено экспериментальное исследование работы предложенной схемы. Для анализа были выбраны php-приложения, содержащие заранее известный набор уязвимостей внедрения SQL-кода. В ходе эксперимента система обнаружила 4 уязвимости в программном обеспечении, при 8 ложных срабатываний.

Заключение. Результатом данной работы является описание схемы применения технологий символьно-конкретного выполнения и машинного обучения для тестирования безопасности ПО, применение которой позволит автоматизировать процесс поиска уязвимостей в ПО и сократить время работы эксперта, Полученная схема будет использована при построении системы автоматизации проведения испытаний средств защиты информации, представленной в работе [7].

СПИСОК ЛИТЕРАТУРЫ

1. Positive Technologies. Уязвимости приложений финансовой отрасли (2016) [Электронный ресурс]. – Режим доступа: https://www.ptsecurity.com/upload/ptru/analytics/Financial-Vulnerability-2016-rus.pdf. – 01.02.2016.
2. Microsoft Security Development Lifecycle [Электронный ресурс]. – Режим доступа: https://www.microsoft.com/en-us/sdl/. – 29.01.2016.
3. Anwer F. Security testing / F. Anwer, M. Nazir, K. Mistafa // Trends in security testing. – 2016. – P. 35–66.
4. Chen T. State of the art: Dynamic symbolic execution for automated test generation / T. Chen, X.S. Zhang, S.Z. Guo, H.Y. Li, Y. Wu // Future Generation Computer Systems. – 2013. – P. 1758–1773.
5. Noorian M. Machine Learning-based Software Testing: Towards a Classification Framework / M. Noorian, E. Bagheri, W. Du // Proceedings of the 23rd International Conference on Software Engineering & Knowledge Engineering (SEKE'2011). – USA, 2011. – P. 225–229.
6. Grieco G. Toward large-scale vulnerability discovery using Machine Learning // Proceedings of the Sixth ACM Conference on Data and Application Security and Privacy. – New Orleans, Louisiana, USA, 2016. – P. 85–96.
7. Мещеряков А.А. Построение системы автоматизации проведения испытаний средств защиты информации // Сборник трудов XIII Международной конференции студентов, аспирантов и молодых ученых «Перспективы развития фундаментальных наук». – Томск, 2016. – Т. 7. – С. 90–93.