Научная статья
Original article
УДК 004.424

ЭВОЛЮЦИОННЫЙ ФАЗЗИНГ СИСТЕМ

EVOLUTIONARY FUZZING OF SYSTEMS

Куренная Валерия Олеговна, студент бакалавр, Донской государственный технический университет, г. Ростов-на-Дону (344003 Россия г. Ростов-на-Дону, Гагарина 1), kurinaaa@rambler.ru

Kurennaya Valeria Olegovna, bachelor student, Don State Technical University, Rostov-on-Don (344003 Russia, Rostov-on-Don, Gagarina 1), kurinaaa@rambler.ru

Аннотация. В программировании увеличилась поребность в нечетком программировании. Фаззинг прогаммы основываются на генетических методах, однако в случае с интерпретатором фаззинг представляет собой сложную задачу, поскольку входными данными являются фрагменты кода, которые должен быть синтаксически/семантически корректным, чтобы пройти элементарные проверки интерпретатора. С другой стороны, нечеткий ввод также должен быть достаточно необычный, чтобы вызвать исключительное поведение в интерпретаторе, такие как сбои, утечки памяти и ошибочные утверждения.

Annotation. In programming, the need for fuzzy programming has increased. Fuzzing programs are based on genetic methods, however, in the case of an interpreter, fuzzing is a difficult task, since the input data is code fragments that must be syntactically / semantically correct in order to pass elementary checks of the

interpreter. On the other hand, fuzzy input must also be unusual enough to cause exceptional behavior in the interpreter, such as crashes, memory leaks, and false assertions.

Ключевые слова: тесирование, нечеткая логика, решение задач, программирование.

Key words: testing, fuzzy logic, problem solving, programming.

Что такое фаззинг (фазз-тестирование)?

Фаззинг — это метод обеспечения качества, используемый для обнаружения ошибок кодирования и уязвимостей безопасности в программном обеспечении, операционных системах или сетях. Он работает, пытаясь вывести систему из строя или вызвать ошибки, предоставляя большой объем случайных входных данных. Если обнаружена уязвимость, платформа фазз-тестирования (также называемая фаззером) может помочь определить основную причину.

Системы фаззинга очень хорошо находят определенные типы уязвимостей, включая переполнение буфера, отказ в обслуживании (DoS), межсайтовый скриптинг и внедрение кода. Однако они менее эффективны при борьбе с скрытыми угрозами безопасности, которые действительно приводят к сбоям или видимым ошибкам, таким как программы- шпионы, черви, трояны и руткиты.

Хотя фаззинг — это простой метод, он экономически эффективен и легко масштабируется. Он часто может выявить серьезные недостатки, которые остаются незамеченными при создании и отладке программ. Однако он не дает полной картины безопасности, качества или эффективности программного продукта. Поэтому его обычно сочетают с другими методами, такими как тестирование черного ящика, бета-тестирование и модульное тестирование.

Типы фазз-тестирования

Фаззинг приложений

Этот метод фаззинга проверяет функции пользовательского интерфейса, такие как кнопки, поля ввода в формах или параметры в программах командной строки. Его также можно использовать для тестирования команд API. Он работает, обращаясь к функциям с необычно высокой частотой, предоставляя недопустимый контент, такой как слишком много текста в полях ввода, и пробуя различные случайные входные данные.

Фаззинг протокола

Такие протоколы, как протокол передачи гипертекста (HTTP), используются для обмена данными через Интернет. Фаззинг протокола используется для проверки поведения сервера при отправке нежелательного контента по заданному протоколу. Основная цель состоит в том, чтобы предотвратить ошибочное истолкование запросов протокола как команд и их выполнение на сервере.

Фаззинг формата файла

Фаззинг формата файла создает поврежденный файл и представляет его целевому программному обеспечению для обработки. Это актуально как для установленного ПО, так и для веб-приложений, принимающих файлы на вход. Файлы обычно имеют стандартные форматы, такие как .jpg, .docx или .xml. Фаззер может протестировать приложение, предоставив файлы, которые не соответствуют ожидаемому формату или содержат неожиданное содержимое. Усовершенствованная версия фаззинга форматов файлов может тестировать функции для определенных форматов файлов, например методы сжатия изображений или видео.

Инструменты безопасности фаззинга

Генеративные фаззеры

Генеративный фаззер может быть чем угодно: от полностью случайных данных до слегка измененных данных. Например, при нечетком тестировании HTTP-трафика можно фаззить весь пакет, что означает, что он, скорее всего, не достигнет пункта назначения. В качестве альтернативы генеративный фаззер может разбить пакет на отдельные компоненты и попытаться фаззить

каждый из них отдельно, сохраняя структуру пакета нетронутой. Это позволяет тестировать недопустимое содержимое пакета, сохраняя при этом заголовки TCP/IP и HTTP.

Фаззеры мутаций

Фаззеры на основе мутаций берут набор допустимых входных данных и выполняют над ними мутации, чтобы вызвать ошибки или сбои в тестируемом программном обеспечении. Вот несколько примеров методов мутации:

Переключение наименее значимого бита (LSB) — изменение бита в конце каждого двоичного целого числа.

Обфускация HTTP-запросов — добавление случайного значения к каждому значению HTTP-заголовка. Это может быть очень эффективным для обнаружения уязвимостей и обеспечивает высокое покрытие кода, поскольку входные данные достаточно похожи на исходные допустимые входные данные.

Шаблоны — использование допустимой структуры данных или формата для повышения вероятности того, что нечеткий ввод будет принят целевой системой. Это может сократить время и ресурсы фазз-теста, гарантируя, что входные данные правильно сформированы.

Фаззеры эволюции

Эволюционный фаззинг основан на использовании генетического программирования, предназначенного для сходимости к вводу, который приведет к ошибке. Генетические алгоритмы используют концепции мутации, кроссовера и отбора для поиска решений сложных проблем. Эволюционный подход к генерации кода, направляя процесс эволюции для создания необычных фрагментов кода. Как было сказано ранее, есть несколько параметров, доступных для точной настройки фаззинг для повышения производительности. Например, выбор использования подмножества (мощности равной размер совокупности) исходного набора тестов, а не всего набора, поскольку первое поколение должно эффективно использовать имеющиеся ресурсы. Остальные входные данные из набора тестов могут быть

использованы в более позднем поколении, когда фаззинг зависнет в некоторых локальных минимумах, что является известным препятствием в эволюционных алгоритмах. Поколение, в котором идентифицируется ошибка, зависит от различных факторов, в том числе размер входной тестовой выборки, размер рассматриваемого фрагмента для генетической операции и размера индуцированного нового фрагмента и т. д., как обсуждалось, чем выше сложность входных данных, тем выше вероятность найти ошибку.

Литература

1. Ван Хайтао Исследование системы осведомленности об информационной безопасности на основе технологий больших данных и искусственного интеллекта, Технологии и приложения сетевой безопасности, 2018 (3): 60-63.
2. Чжэн Яньфан Исследования и практика применения искусственного интеллекта и технологий анализа в системе ситуационной осведомленности информационной безопасности Мир цифровых коммуникаций, 2018, № 160 (4): 229.
3. Чжэн Фан Информационная безопасность в эпоху искусственного интеллекта, Исследования в области информационной безопасности, 2017, 3 (11): 966-967.
4. Дэн Вэньбинь Проблемы и меры противодействия надзору за информационной безопасностью в эпоху искусственного интеллекта Информационная безопасность Китая, 2018 г., 106 (10): 106-108.

References

1. Wang Haitao Research on Information Security Situation Awareness System Based on Big Data and Artificial Intelligence Technology, Network Security Technology and Application, 2018 (3): 60-63.
2. Zheng Yanfang Research and Practice of Artificial Intelligence Application and Analysis Technology in Information Security Situational Awareness System Digital Communication World, 2018, No.160 (4):229.

3. Zheng Fang Information Security in the Age of Artificial Intelligence, Information Security Research, 2017, 3 (11): 966-967.
4. Deng Wenbing Challenges and Countermeasures for Information Security Supervision in the Age of Artificial Intelligence China Information Security, 2018, 106 (10): 106-108.

© Куренная В.О., 2022 Научно-образовательный журнал для студентов и преподавателей «StudNet» №6/2022.

Для цитирования: Куренная В.О. ЭВОЛЮЦИОННЫЙ ФАЗЗИНГ СИСТЕМ// Научно-образовательный журнал для студентов и преподавателей «StudNet» №6/2022.