Московский государственный университет имени М.В. Ломоносова
Факультет вычислительной математики и кибернетики
Кафедра информационной безопаснсти
Лаборатория безопасности информационных систем

Николайчук Артём Константинович

Исследование методов фаззинга сложных програма

Курсовая работа

Научный руководитель:
м.н.с
А.А.Петухов

Москва, 2022

Содержание

1 Введение

1.1 Аннотация

В настоящее время разработчики всё больше беспокоятся о безопасности создаваемых приложений. Цена ошибки или бага может очень высокой. Тестирование стало неотъемлемой частью жизненного цикла разработки программного обеспечения. Известно, что даже 100%-ое покрытие исходного кода тестами не гарантирует отсутствие ошибок. Более того, разработчики при написании тестов руководствуются тем, как должна вести себя программа. Из-за этого многие "неочевидные"частные случаи могут быть пропущены. Некоторые програмы, такие как парсеры, интерпретаторы, компиляторы, обработывают данные, которые имеют сложную структуру. Для них просто невозможно перебрать все варианты входных данных, чтобы удостовериться, что всё работает правильно. Для таких программ разумно применять методы фаззинга.

1.2 Фаззинг

Фаззинг - способ автоматического тестирования программного обеспечения. Фаззер генерирует случайные входные данные и улучшает или изменяет их, затем анализирует работу программы на этих данных и пытается обнаружить потенциальные дефекты или уязвимости программного обеспечения. Фаззеры принято классифицировать по принципу генерации данных:

- Мутационные фаззеры обрабатывают заранее подготовленное множество входных данных. Наиболее популярными изменениями являются заимствованные из биологии мутации и скрещивания. Мутации - это изменение какой-то части входных данных на случайную. При скрещивании выбираются два примера, которые обмениваются друг с другом частью данных.

- Генерационные фаззеры создают новые примеры, основываясь на информации о требуемой структуре входных данных.

- Смешанные фаззеры объединяют в себе два предыдущих подхода. Например, при мутации данные могут меняться не на случайные, а на сгенерированные. Или фаззер может сначала создать пул тестовых данных и к нему применять мутационный метод.

1.3 Представление входных данных

На практике оказалось очень удобно задавать структуру входных данных с помощью грамматик. Если мы знаем грамматику, то все возможные инпуты можно представить абстрактным синтаксическим деревом. Это позволяет избегать синтаксических ошибок на этапе запуска программы. В дальнейшем мы покажем, что такое представление полезно при генерации и мутации данных.