Анализ кода многопоточных приложений с помощью полносистемного символьного выполнения

Кутовой Егор Арсеньевич[1]
Ниськов Федор Владимирович[2]

1: *Студент, ФИВТ МФТИ, Москва, Россия*
2: *Аспирант, ВМК МГУ, ИСП РАН, Москва, Россия*
E-mail: kutovoi.ea@phystech.edu, fedor.niskov@ispras.ru

Научный руководитель — *Курмангалеев Шамиль Фаимович*

Подавляющее большинство программ, выполняющихся в датацентрах и на персональных компьютерах, являются многопоточными из-за необходимости в полную меру использовать мощности многоядерных процессоров. Но ни для кого не секрет, что писать многопоточные программы значительно сложнее, чем однопоточные, во многом из-за того, насколько сложно находить и исправлять ошибки, связанные с проблемами синхронизации потоков программы.

Одна из самых распространенных ошибок такого типа — «data race», которая заключается в том, что несколько потоков без должной синхронизации обращаются к одной ячейке памяти. Современные анализаторы таких ошибок очень хорошо справляются с их поиском в том случае, когда ошибки относительно легко воспроизводимы и не завязаны на слишком сложных предикатах.

В рамках нашей работы мы решили улучшить обнаружение ошибок в остальных случаях. Для этого мы взяли существующий анализатор кода S2E [1] — платформу полносистемного символьного выполнения — и расширили ее, добавив в нее поддержку эмуляции многоядерных систем и поиска ошибок «data race».

S2E примечательна тем, что она использует мощный движок символьного исполнения, позволяющий эффективно находить ошибки, случающиеся только при очень сложных входных условиях, а также тем, что в ней используется полносистемная эмуляция, т.к. это позволяет анализировать очень сложные программы, работающие только в конкретных окружениях.

Что мы сделали:

- Добавили возможность параллельного выполнения частей программ в S2E, одновременно утилизирующего многие физические ядра процессора
- Расширили применимость символьного выполнения таким образом, чтобы оно работало с многими виртуальными ядрами
- Реализовали гибридный планировщик работы виртуальных ядер, выполняющий код параллельно в конкретном режиме и последовательно по схеме «round-robin» в символьном режиме
- Обновили версию эмулятора QEMU, использующегося в S2E, с 3.0 до 6.1, чтобы добавить возможность тестировать в S2E драйверы для современных устройств
- Добавили в S2E новый анализатор - детектор «data race», реализующий алгоритм «DJIT+» [2]

Наша работа тестировалась на популярной открытой программе для анализа трафика Suricata, а также на модельных примерах. В итоге у нас получилось добиться:

- Ускорения выполнения частей целевых программ в S2E за счет утилизации многих ядер, до 100 ядер включительно
- Символьного выполнения программ на многих виртуальных ядрах
- Упрощения поиска редко возникающих ошибок «data race» благодаря многоядерному символьному выполнению

Мы надеемся, что полученные нами результаты послужат фундаментом для дальнейшего развития многоядерного символьного выполнения, что позволит улучшить возможности обнаружения ошибок синхронизации в программах.

Литература

1. S2E: A Platform for In-Vivo Multi-Path Analysis of Software Systems. Vitaly Chipounov. EPFL PhD Thesis, July 2014.
2. E. Pozniansky and A. Schuster. MultiRace: Efficient on-the-fly data race detection in multithreaded C++ programs. Concurrency and Computation: Practice and Experience, 19(3):327–340, 2007.