Верификация дизассемблера x86-64

Тенсин Е. Д.[1] Баклановский М. В.[2]

[1]студент 361 группы [2]старший преподаватель

кафедры системного программирования
математико-механического факультета
Санкт-Петербургского Государственного Университета

30 мая 2012 г.

Постановка задачи

- Популярные дизассемблеры:
  - OllyDbg
  - PEBrowse
  - Objconv
  - ...
  - множество
- + разработка своего дизассемблера
- Громоздкость, а значит ошибки и сложная отладка
- Как проверить заявленный охват инструкций?
- Фреймворк автоматической верификации

Аппаратный декодер

- Процессор декодирует инструкции безошибочно
- Аппаратно, а значит придется судить о результате по косвенным признакам — *исключениям*
- Программная исключений обработка с помощью *Structured Exception Handling*
- Выделяем две страницы с:
  - доступом на чтение и исполнение,
  - запретом любого доступа
- Размещаем тестируемую последовательность смежно со второй страницей
- Тестируем подпоследовательности длиной $i = 1, 2, \ldots, m, m <= n$

Программный декодер

- Есть API для декодирования ровно 1 инструкции
- Не исполняет декодированные инструкции
- Проще обрабатывать исключения (только `access violation`)
- Библиотека, реализующая обертку процедуры декодирования ровно 1 инструкции для каждого тестируемого дизассемблера

Генерация тестируемых инструкций

- Полный перебор
  - 2^{120} итераций
  - Значит, придется оптимизировать
- Генерация случайных последовательностей
  - Экзотические комбинации
  - Пропущенные во время оптимизированного полного перебора

Оптимизация полного перебора

- Только допустимые значения префиксов
- Операнды
  - Целые числа: 0, < 32, намного больше 32, ...
  - Регистры
    - 2 случайных регистра если 1 регистровый операнд
    - 3 случайных регистра если 2 регистровых операнда во всех комбинациях
  - Ячейки в памяти
    - $[reg + disp]$: 2 случайных регистра + случайные смещения разных длин
    - $[reg + k\,reg + disp]$: 3 случайных регистра во всех комбинациях с множителем и смещениями

Результаты

- Разработка фреймворка для автоматической верификации дизассемблера
- Обнаружение и исправление множества ошибок в собственном дизассемблере

Планы

- Верификация популярных дизассемблеров с оформление количественных результатов
- Развитие проектного дизассемблера с системой верификации как отдельного проекта партнерской программы с Intel