esil - универсальный il

ESIL - Intermediate Language для radare2

Anton Kochkov (@akochkov)

26 ноября 2015 г.

ZeroNights 11-2015

anton kochkov

- Москва, Россия
- Хобби - реверс инжиниринг, языки и путешествия
- Участник R2 crew и евангелист radare2
- ООО Код Безопасности

краткий обзор intermediate languages

что такое intermediate language

- *Intermediate language is the language of an abstract machine designed to aid in the analysis of computer programs. Intermediate Language - Wikipedia 2015*
- Используется как в теории (и практике) компиляции
- Аналогично незаменим и для декомпиляции
- Огромное количество разных академических и практических воплощений
- Основа для высокоуровневого анализа - SMT, AEG, AEP, etc

reil[1]

- Изобретен компанией Zynamics
- Использовался в продуктах BinNavi, BinDiff
- Поддерживает архитектуры x86, ARM, PowerPC
- Бесконечная память VM
- Бесконечное количество регистров VM
- Без Floating Point
- Оригинальные утилиты написаны на Java

[1] Sebastian Porst Thomas Dullien (2009). *REIL: A platform independent intermediate representation of disassembled code for static code analysis*. в:

reil

- 17 инструкций
- Алиасы для реальных регистров (eax, ebx, r0, . . .)[2]

[2] *REIL description - Zynamics* (2005).

bap

- BAP - Binary Analysis Platform[3]
- Настоящее имя IL - BIL
- Развитый фреймворк
- Интеграция с другими утилитами - TEMU, libVEX, IDA Pro, Qira, ...
- Ориентирован на x86, ARM
- Без Floating Point

[3] Edward J. Schwartz David Brumley Ivan Jager и Spencer Whitman (2014). *The BAP Handbook*. B:

bitblaze (vineil/vex)

- BitBlaze[4] - платформа, аналогичная BAP
- Имеет несколько промежуточных языков
- VEX IL (libVEX из valgrind) - “нижний” уровень
- Vine IL - “верхний” уровень
- Написан на OCaml + C++

[4]Heng Yin Dawn Song David Brumley, Juan Caballero и Ivan Jager (2008). *BitBlaze: A New Approach to Computer Security via Binary Analysis*. B:

vex il

- Явное указание всех side-эффектов для команд
- Ближе всего к ESIL
- Оттестирован и используется в Valgrind
- Хорошо подходит для эмуляции кода
- Избыточен

vine il[5][6]

- Бесконечная память
- Бесконечное количество регистров
- Поддержка типов
- Поддержка “variable scope”

[5] BitBlaze Team (2009). *Vine Installation and User Manual*. в:
[6] David Brumley (2008). *Analysis and Defense of Vulnerabilities in Binary Code*. в:

rreil, openreil, mail

- RREIL[7] - гибкий язык, замена REIL
- RREIL - поддержка типов
- RREIL - интересная концепция “доменов”
- MAIL - IL, созданный для анализа Malware
- MAIL - позволяет программе перезаписывать себя саму
- RREIL и MAIL - опять отсутствие Floating Point

[7] Bigdan Mihaila Alexander Sepp и Axel Simon (2011). *Precise Static Analysis of Binaries by Extracting Relational Information*. B:

rreil, openreil, mail

- OpenREIL[8] - проект, созданный для использования REIL в современных реалиях
- OpenREIL - полноценный фреймворк, как и BAP
- OpenREIL отличается от оригинального REIL
- Использует libVEX и имеет поддержку SMT-solving

[8] Dmytro Oleksiuk (2015). https://github.com/Cr4sh/openreil.

esil - сходства и отличия

краткое описание

- Evaluable Strings Intermediate Language[9]
- Использует обратную польскую нотацию (для скорости)
- Не предназначен для чтения человеком
- По “уровню” приближен к VEX
- Небольшое количество инструкций
- Полный учёт side-эффектов

[9] Radare2 Team (2015b). *ESIL description*.

краткое описание

- Сроектирован для большого количества архитектур
- Бесконечная память
- Бесконечные регистры
- Алиасы (использование “нативных” имен)
- Есть возможность вызывать куски нативного кода (+syscall)
- Возможность добавления “custom ops”
- Нет Floating Point (будет в следующей версии)

операнды esil

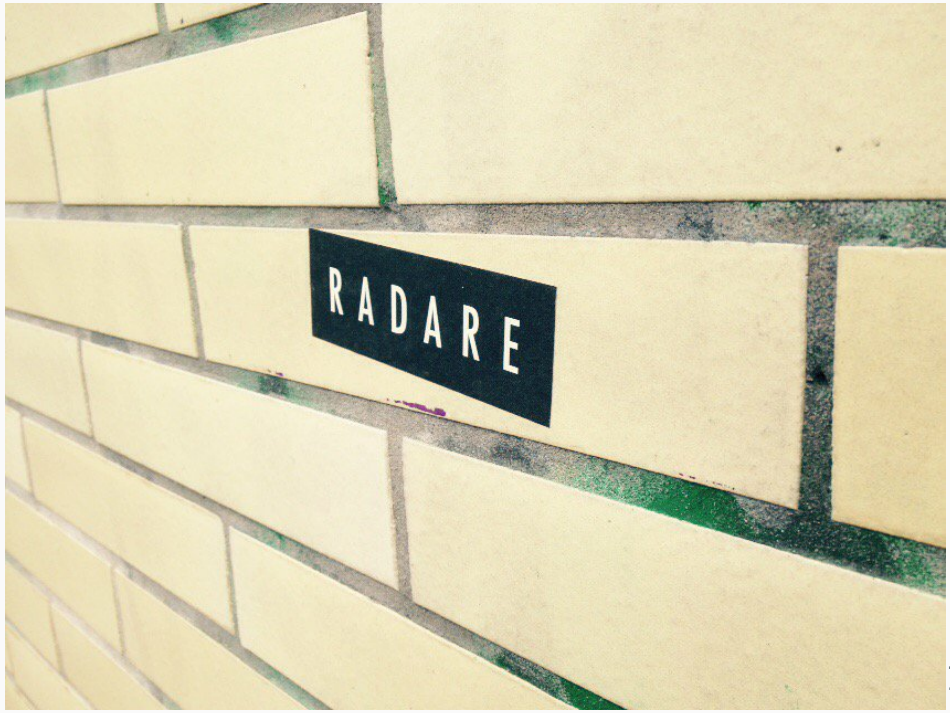
Таблица 1: ESIL Operands[10]

ESIL Opcode	Operands	Name	Desription
$	src	Syscall	syscall
$$	src	Instruction address	Get address of current instruction
==	src,dst	Compare	v = dst - src ; update_eflags(v)
<	src,dst	Smaller	stack = (dst <src)
<=	src,dst	Smaller or Equal	stack = (dst <= src)
>	src,dst	Bigger	stack = (dst >src)
|=	src,reg	OR eq	reg = reg | src

[10]Radare2 Team (2015c). *ESIL Instruction Set*.

практическое применение

radare



[11]

[11] *Radare advertisement in Berlin's U-Bahn* (2015).

radare2 утилиты

- rax2
- rabin2
- rasm2
- radiff2
- rafind2
- rahash2
- radare2
- r2pm
- rarun2/ragg2/ragg2-cc

1 command <—>1 reverse-engineering'notion

1. Каждый символ команды что-то значит (w = write, p = print)
2. Обычно команды - это аббревиатуры действий pdf = p <->print d <->disassemble f <->function
3. Доступна короткая справка для каждой команды **cmd?**, например pdf?,?, ???, ???, ?$?, ?@?

radare2 — основные команды cli-режима

1. **r2 -A** или **r2** + **aaa** : Анализ
2. **s** : Переход по указанному адресу
3. **pdf** : Дизассемблирование функции
4. **af?** : Анализ функции
5. **ax?** : Анализ XREF
6. **/?** : Поиск
7. **ps?** : Напечатать строку (print string)
8. **C?** : Комментарии
9. **w?** : Запись (hex, опкодов, etc)

radare2 — visual mode

radare2 — основные команды визуального режима

1. **V?** или просто **?** : Помощь по командам
2. **p/P** : переключение между разными визуальными представлениями
3. Навигация с помощью стрелок/hjkl
4. **o** : переместиться по адресу
5. **e** : визуальный режим настроек
6. **v** : список функций
7. **_** : HUD
8. **V** : ASCII Graph
9. **0-9** : Прыжок на функцию
10. **u** : Undo

эмуляция участков кода

- **ae*** набор инструкций
- **aei** - инициализация ESIL VM
- **aeim** - инициализация стека/памяти VM
- **aeip** - установка IP (Instruction Pointer)
- **aes** - step в режиме эмуляции ESIL
- **aec[u]** - continue [until]
- **aef** - эмуляция функции

эмуляция участков кода

- DEMO

embedded controller - 8051 - esil vm[12]

- **r2 -a 8051 ite_it8502.rom**
- **. ite_it8502.r2**
- **e io.cache=true** для использования кеширования IO
- запустим **aei**
- запустим **aeim**
- запустим **aeip** для старта с момента указания команды
- **aecu [addr]** для эмуляции, пока не достигнем IP = [addr]

[12] *ESIL emulation in radare2* (2014).

совместная отладка

- Использование “подсказок” ESIL при визуальной отладке
- DEMO

эмуляция vm

- Позволяет выполнить распаковку или выполнение в VM
- Хороший пример - использование ESIL для распаковки Baleful

автоматическое отображение результатов эмуляции в дизассемблере

- Отображает в комментариях значения регистров и пямяти
- Использует тот же механизм эмуляции кода ESIL VM
- Показывает likely/unlikely для условных переходов
- **e asm.emu=true**

автоматическое отображение результатов эмуляции в дизассемблере

- DEMO

конвертация в другие языки - openreil

- OpenREIL - развитый фреймфорк
- Есть возможность использования SMT
- Добавлена возможность конфертации ESIL в OpenREIL
- Команда **aetr**

конвертация в другие языки - openreil

- DEMO

embedded controller - 8051 - esil2reil

- **r2 -a 8051 ite_it8502.rom**
- **. ite_it8502.r2**
- run **pae 36** для показа ESIL представления функции 'set_SMBus_frequency'
- run aetr `pae 36` для конвертации строки ESIL в REIL[13]
- Сохранить вывод в файл и передать управление в OpenREIL
- Можно проделать всё вышеперечисленное с помощью r2pipe скрипта

[13]Dmytro Oleksiuk (2015). https://github.com/Cr4sh/openreil.

radeco il и radeco decompiler

esil -> radeco[14]

- Использует ESIL в качестве входных данных
- Использует другие метаданные из radare2
- Соединяется с radare2 через r2pipe
- Написан на Rust

[14]Radare2 Team (2015a). https://github.com/radare/radeco.

причины появления декомпилятора

- Существующие FOSS декомпиляторы не учитывают последние исследования
- Академические (но интересные) идеи не имеют полноценной реализации
- Radare2 нуждается в декомпиляторе
- Хорошее и интересное задание для Google Summer of Code

описание radeco il

- Графовое представление
- Взяты идеи из RREIL и MAIL
- Использование SSA на этапе лифтинга ESIL -> Radeco IL
- Встроенная поддержка DCE (Dead Code Elimination)
- Базовая возможность вывода типов[15]

[15]Thanassis Avgerinos JongHyup Lee и David Brumley (2011). *TIE: Princpled Reverse Engineering of Types in Binary Programs*. B:

radeco demo

- DEMO

пути будущего развития

поддерживаемые архитектуры

- Сейчас лучше всего поддерживаются x86, ARM, GameBoy, 8051, etc
- Глобальная цель - поддержка ESIL для всех архитектур в radare2
- Поддержка профилей для выбранных модификаций/вариаций процессоров

поддерживаемые наборы инструкций

- Floating point (LLVM/McSema)[16]
- Векторные инструкции (SSE, AVX, Neon, etc)
- VLIW инструкции (для эмуляции кода DSP)

[16] *REIL description - Zynamics* (2014).

визуальная отладка и трассировка

- Улучшение UI
- Возможность визуального сравнения эмуляции и нативного выполнения
- Устранение “мертвого” кода из ASCII графов на лету
- Интеграция в WebUI и Bokken[17]

[17] *Bokken* (2015).

развитие декомпилятора radeco

- Генерация C кода
- Поддержка нативных типов
- Синхронизация с отладкой
- Автовывод типов/распознавание объектов и классов[18][19]

[18] Thanassis Avgerinos JongHyup Lee и David Brumley (2011). *TIE: Princpled Reverse Engineering of Types in Binary Programs*. B:
[19] **il-optimization**.

references

список литературы

- Alexander Sepp, Bigdan Mihaila и Axel Simon (2011). *Precise Static Analysis of Binaries by Extracting Relational Information*. B:
- *Bokken* (2015).
- Brumley, David (2008). *Analysis and Defense of Vulnerabilities in Binary Code*. B:
- David Brumley Ivan Jager, Edward J. Schwartz и Spencer Whitman (2014). *The BAP Handbook*. B:
- Dawn Song David Brumley, Heng Yin, Juan Caballero и Ivan Jager (2008). *BitBlaze: A New Approach to Computer Security via Binary Analysis*. B:
- *ESIL emulation in radare2* (2014).
- *Intermediate Language - Wikipedia* (2015).

a lot of them II

JongHyup Lee, Thanassis Avgerinos и David Brumley (2011). *TIE: Princpled Reverse Engineering of Types in Binary Programs*. B:

Oleksiuk, Dmytro (2015). https://github.com/Cr4sh/openreil.

Radare advertisement in Berlin's U-Bahn (2015).

REIL description - Zynamics (2005).

REIL description - Zynamics (2014).

Team, BitBlaze (2009). *Vine Installation and User Manual*. B:

Team, Radare2 (2015a). https://github.com/radare/radeco.

— (2015b). *ESIL description*.

— (2015c). *ESIL Instruction Set*.

Thomas Dullien, Sebastian Porst (2009). *REIL: A platform independent intermediate representation of disassembled code for static code analysis*. B: