REIL: A platform-independent intermediate representation of disassembled code for static code analysis

CanSecWest 2009
Thomas Dullien, Sebastian Porst

김영철

2016. 1. 29.

Abstract

- REIL 소개
  ➔ 플랫폼에 구애되지 않는 중간언어
  ➔ 정적 코드 분석을 단순화하고 자동화하기 위함
- REIL 언어와 virtual REIL architecture 소개

Introduction

- 시대가 바뀌면서 다양한 머신 사용
  ➔ Windows : x86
  ➔ 모바일 분야 : ARM
  ➔ 네트워크 분야 : PowerPC
  ➔ 무선 기기 : MIPS
  다양한 플랫폼의 어셈블리 코드에 작동하는 도구 필요

- REIL
  ➔ native assembly를 추상화, 다양한 플랫폼에 적용 가능
  ➔ BinNavi에 탑재

The REIL instruction set

- REIL의 장점
  1. 17개의 instruction을 가짐
     ➔ x86 : 300개 이상, Power PC : 100개 이상
  2. 모든 instruction들은 하나의 역할을 수행
     ➔ x86 : 하나의 instruction이 값 로드, 계산, flag 세팅 수행

The REIL instruction set

- 규칙적인 REIL instruction
  각각의 instruction은 3개의 operand를 가짐
  ➔ 첫 번째, 두 번째 operand는 input
  ➔ 세 번째 operand는 output
    ➔ jump의 경우 jump target
  ➔ NOP operation의 경우, empty 타입의 operand 3개
  ➔ integer, register, subaddress 3가지 타입
    ➔ integer : output으로 사용불가
    ➔ register : $t_{<number>}$ 형태 (t_0, t_1 …)
    ➔ subaddress : jump의 세 번째 operand에서만 보임

The REIL instruction set

- operand의 사이즈 표기
  ➔ 0x17/b_2, t_0/b_4
- meta-data : 정적분석에 중요한 instruction의 key-value 쌍의 맵
  ➔ 최신 버전에는 한 종류의 meta-data
  ➔ subfunction call의 결과로 생성
  jump는 key:*isCall* – value:*true* 쌍으로 표기

The REIL instruction set

- 17개의 instruction의 5개 그룹
  1. the arithmetic instructions
  2. the bitwise instructions
  3. the conditional instructions
  4. the data transfer instructions
  5. other instructions

The REIL instruction set

- The arithmetic instructions
  ADD, SUB, MUL, DIV, MOD, BSH로 구성

  MUL, DIV, MOD
  ➔ unsigned way로 번역
  BSH : logical shift
  ➔ 두 번째 인자의 상태에 따라 left-shift/right-shift

The REIL instruction set

- The arithmetic instructions

```
ADD $t_0/b_4$,    $t_1/b_4$, $t_2/b_8$
SUB $t_7/b_4$,    $t_9/b_4$, $t_{12}/b_8$
MUL $t_8/b_4$,    $4/b_4$,   $t_9/b_8$
DIV $4000/b_4$,   $t_2/b_4$, $t_3/b_4$
MOD $t_8/b_4$,    $8/b_4$,   $t_4/b_4$
BSH $t_1/b_4$,    $2/b_4$,   $t_2/b_8$
```

➔ 잠재적인 오버플로우 처리

The REIL instruction set

- The bitwise instructions
  AND, OR, XOR로 구성

```
AND t0/b4,   t1/b4, t2/b4
OR  t7/b4,   t9/b4, t12/b4
XOR t8/b4,   4/b4,  t9/b4
```

➔ register의 size가 확장될 가능성 X

The REIL instruction set

- The data transfer instructions
  LDM, STM, STR로 구성

  LDM : 메모리로부터 값 로드
  STM : 메모리에 값 저장
  STR : 레지스터에 값 저장

The REIL instruction set

- The data transfer instructions

LDM	$413800/b_4$	,	t_1/b_2
STR	t_1/b_2	,	t_2/b_2
STM	t_2/b_2	,	$415280/b_4$

➔ 413800의 메모리의 값을 t1에 저장
➔ t1의 값을 t2에 저장
➔ t2의 값을 415280의 위치에 저장

The REIL instruction set

- The conditional instructions
  BISZ, JCC로 구성

  BISZ : 값을 0과 비교
  JCC : conditional jump

The REIL instruction set

- The conditional instructions

```
BISZ t0/b4, , t1/b1
JCC  t1/b1, , 401000/b4
```

➔ t_0 레지스터가 0인지 비교하고 결과를 t_1에 저장
➔ t_1 레지스터의 값이 1이면 401000으로 점프

The REIL instruction set

- Other instructions
  UNDEF, UNKN, NOP로 구성

  UNDEF : 아직 정해지지 않은 값으로 정의
  UNKN : 원래의 어셈블리 코드에서 번역하지 못한 것
  NOP : No operation