Tracelet-Based Code Search in Executables

PLDI 2014

Yaniv David, Eran Yahav

김영철

2016. 4. 8.

Introduction

- Tracelet-based matching

  (1) Tracelet decomposition
  함수의 CFG를 tracelet으로 분해
  고정된 길이 k의 tracelet 사용
  Basic block(BB)의 수에 따라 tracelet의 길이를 결정
  하나의 tracelet은 제어흐름 명령내에서 시작하고 끝을 맺음
  ➔ tracelet decomposition은 흐름을 포착

Introduction

- Tracelet-based matching

  (2) Tracelet similarity by rewriting
  tracelet 사이의 유사도 측정을 위해 rewrite rule을 정의
  ➔ tracelet을 다른 tracet으로 바꾸기 위한 rewrite rule
  ➔ constraint-solving problem으로 encode
  ➔ 목적 tracelet match를 위해 위반되는 제약의 수 측정

Introduction

- Main contributions
  실행파일 내에서 탐색을 위한 프레임워크
  tracelet 기반으로 한 유사도
  rewriting engine “TRACY” 툴 개발

Overview

- Motivating Example
  소스코드 레벨에서는 유사하지만 어셈블리 코드는 차이가 큼

  original 프로그램의 BB에는 n을,
  patched 프로그램의 BB에는 n'을,
  patched 프로그램에서 새롭게 나타나는 BB에는 m*로 넘버링

Overview

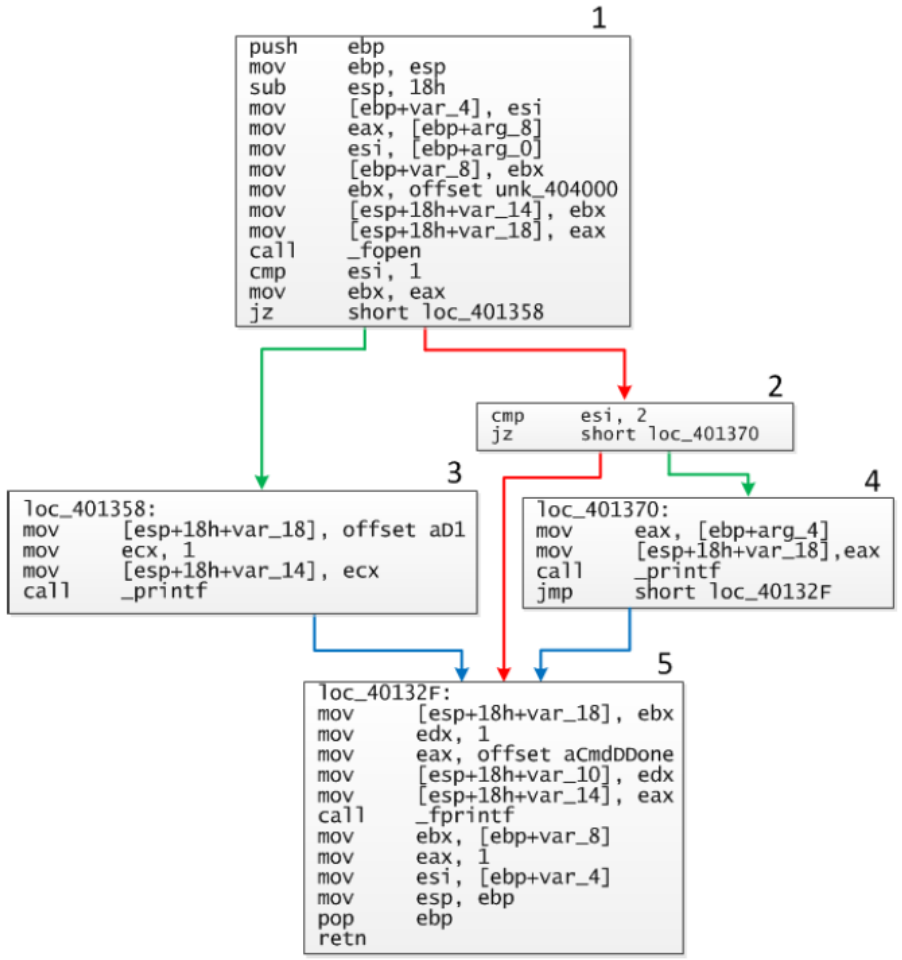
- Motivating Example
  바이너리 레벨에서의 코드는 차이가 많이 남.
  (1) CFG가 다르다. (매칭이 안되는 블록 존재)
  (2) 스택에 있는 지역변수의 오프셋이 다르다.
  (3) 같은 operation에서 다른 레지스터가 사용된다.
  (4) 점프 주소가 바뀐다.

Overview

- Motivating Example
  Tracelets – 부분적인 실행 trace
  ➔ 점프 주소에 대한 안정성
  ➔ 변화에 대한 안정성
  ➔ 의미 비교 – semantic equivalence 체크

Overview

```
1   int doCommand1(int cmd,char * optionalMsg,
2           char * logPath) {
3       int counter =1;
4       FILE *f = fopen(logPath,"w");
5       if (cmd == 1) {
6           printf("(%d) HELLO",counter);
7       } else if (cmd == 2) {
8           printf(optionalMsg);
9       }
10      fprintf(f,"Cmd %d DONE",counter);
11      return counter;
12  }
```

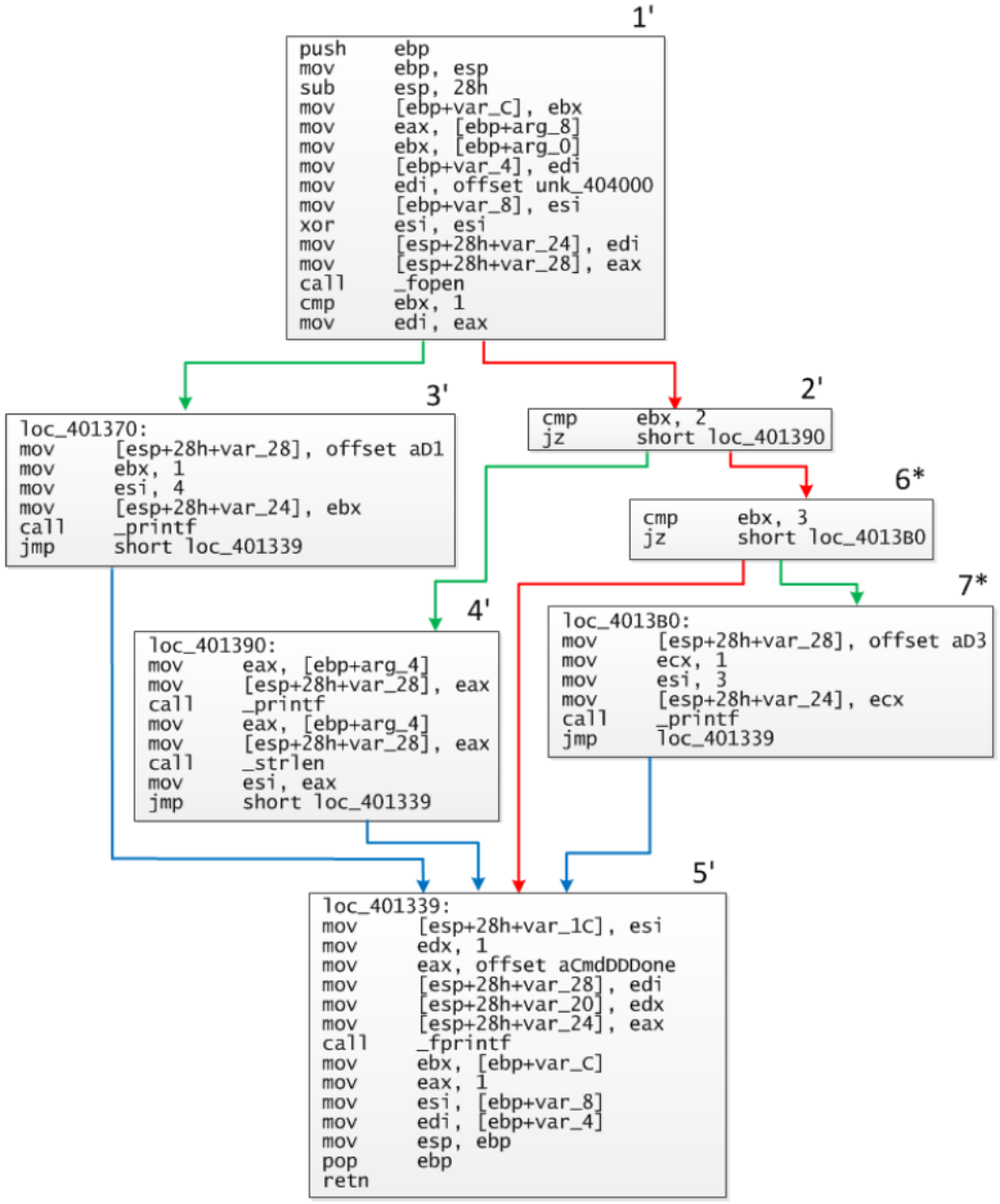


(b) G_1

Figure 1. `doCommand1` and its corresponding CFG G_1.

Overview

```
1   int doCommand2(int cmd,char *optionalMsg,char *logPath){
2       int counter = 1; int bytes = 0; // New variable
3       FILE *f = fopen(logPath,"w");
4       if (cmd == 1) {
5           printf("(%d) HELLO",counter); bytes += 4;
6       } else if (cmd == 2) {
7           printf(optionalMsg); bytes+= strlen(optionalMsg);
8       /* This option is new: */
9       } else if (cmd == 3) {
10          printf("(%d) BYE",counter); bytes += 3;
11      }
12      fprintf(f,"Cmd %d\\%d DONE",counter,bytes);
13      return counter;
14  }
```



(b) G_2

Figure 2. `doCommand2` and its corresponding CFG G_2.

Overview

- Similarity using Tracelets
  G1을 3-tracelets으로 분해
  ➔ (1,2,4) (1,2,5) (1,3,5) (2,4,5)

  G2를 3-tracelets으로 분해
  ➔ (1′,2′,4′) (1′,2′,6′) (1′,3′,5′) (2′,4′,5′) (2′,6*,7*) (6*,7*,5′)

  G1,G2의 tracelets의 유사도를 측정

Overview

- Similarity using Tracelets
  Tracelet comparsion as a rewriting problem
    rewrite rules
    (1) instruction 삭제 [instDelete]
    (2) instruction 추가 [instAdd]
    (3) 같은 타입의 instruction 으로 대체 [Opr-for-Opr]
    (4) 다른 타입의 instruction 으로 대체 [Opr-for-DiffOpr]