Dytan

ISSTA 2007

James Clause, Wanchun Li, Alessandro Orso

김영철

2016. 4. 22.

Introduction

- Dynamic taint analysis
  - 프로그램 실행 시에 특정한 데이터를 표시하고 추적
  - 다양한 분야에서 사용
  - 특정한 목적에 대해서만 정의
    ➔ 확장이나 다른 상황에 적용하는 데 어려움
  - data-flow based tainting only

- Framework in this paper
  (1) flexible and customizable
  (2) data-flow & control-flow based taint 수행
  (3) binary 에 대해서도 작동

Background and motivation

- dynamic tainting
  (1) 데이터를 marking
  (2) 실행하면서 marking들을 전파
  ➔ 프로그램 내부의 정보 흐름을 추적

- explicit information flow
  tainted data가 직접적으로 다른 data에 영향을 줌
  data dependency와 관련

- implicit information flow
  tainted data각 간접적으로 다른 data에 영향을 줌
  control dependency와 관련

Background and motivation

```
1   int a, b, w, x, y, z;
2   a = 11;
3   b = 5;
4   w = a * 2;
5   x = b + 1;
6   y = w + 1;
7   z = x + y;
```

a -> w -> y

explicit information flow

```
1   void foo(int a) {
2     int x, y;
3     if (a > 10) {
4       x = 1;
5     }
6     else {
7       x = 2;
8     }
9     y = 10;
10    print(x);
11    print(y);
12  }
```

implicit information flow

Our approach

- General Framework
  ➔ taint sources, propagation policy, taint sinks
  : 세 가지 설정에 따라 다양한 taint 분석이 가능

Our approach

- General Framework

  Taint sources
  : taint marking 으로 초기 설정되는 data

  (1) Variables and memory offsets

  (2) Data returned from a specific functions

  (3) Data from a type of I/O stream

  (4) Data from a specific I/O stream

Our approach

- General Framework

  Propagation policies
  : taint marking을 전파시키는 방법

  (1) Identifying affecting data
  data-flow only
  data- and control-flow

  (2) Defining a mapping function
  여러 marking을 하나의 set으로 통합
  ➔ 각각을 구분하거나, 하나로 만듦

Our approach

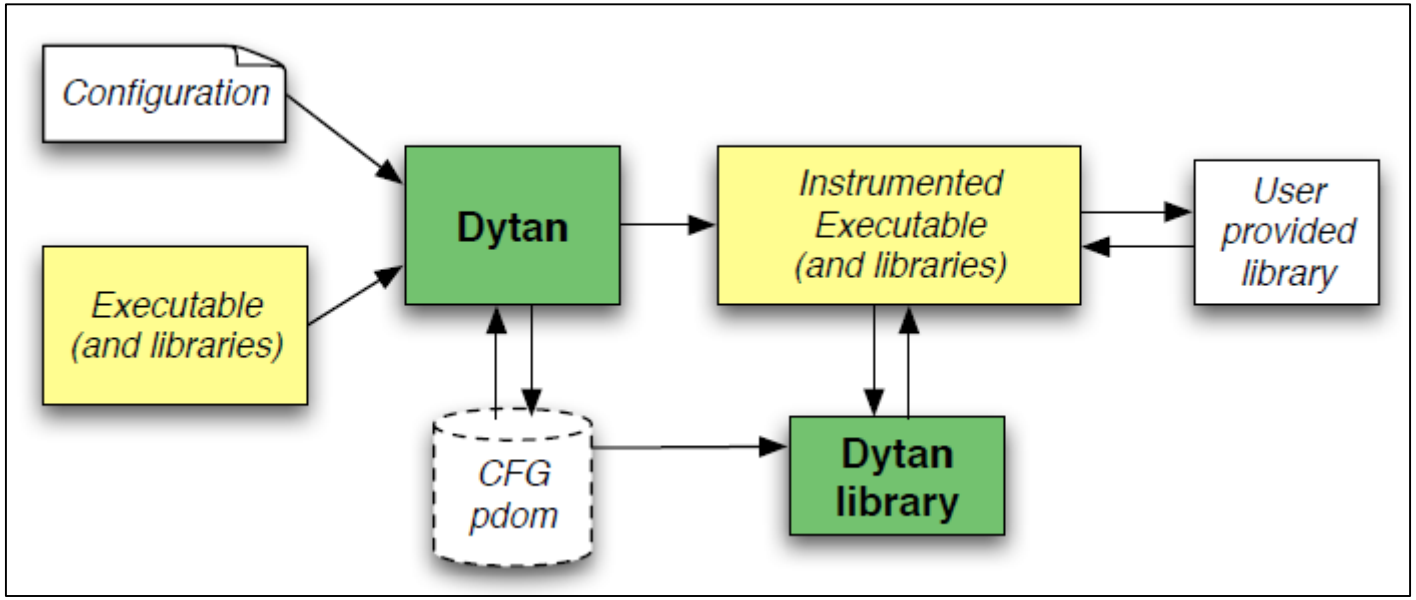
- General Framework

  Taint sinks
  : 사용자가 원하는 marking을 확인하는 위치

  (1) ID
  (2) memory location
  (3) code location
  (4) checking operation

Our approach

- The Tool: DYTAN



Empirical evaluation

- Research Question 1
  - Prevention of overwrite attacks
  - Detection of SQL injection

➔ 빠르게 구현 가능

```
<dytan-config>
<sources>
  <source type=''network''>
    <host>*</host>
    <port>*</port>
  </source>
</sources>
<propagation>
  <dataflow>true</dataflow>
  <controlflow>false</controlflow>
</propagation>
<sinks>
  <sink>
    <id>36</id>
    <location type=''instruction''>
      <instruction=''ret''/>
       ...
      <instruction=''jmp''/>
    </location>
    <action=''validate-absence''/>
  </sink>
</sinks>
</dytan-config>
```