[WhitePaper]

Cross-function 분석 여부가 정적분석 도구의 차이

Soft 4 Soft

RESORT Code Analysis

DNV

DNV GL BUSINESS ASSURANCE

CWE COMPATIBLE

2024. 7

㈜소프트4소프트

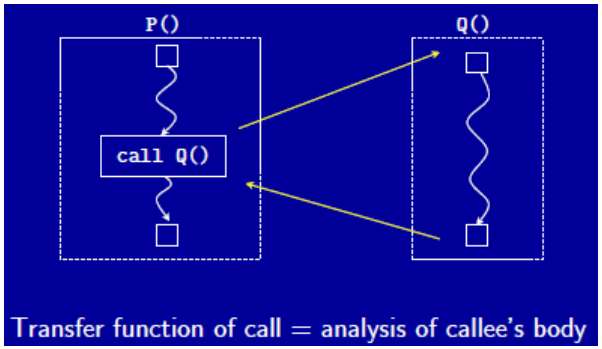
www.soft4soft.com

02-553-9464

목 차

◆ Static Program Analysis

➢ 프로그램을 실행하지 않고 프로그램의 run-time 행위 등을 분석하는 방법으로, 프로그램의 가능한 모든 실행에 대해 코드 결함(code bug)과 보안 허점(security hole)을 찾는 코드 분석을 수행



➢ Program Analysis Approach

✓ Inter-procedural analysis

- 프로그램 전체인

  Cross-function의 프로그램 흐름 분석

  ❖ P( ) -> Q( ) -> P( ) // 올바른 실행 순서로 정확한 코드 분석

✓ Intra-procedural analysis

- 한 함수 내 또는 한 파일에서만 코드 흐름 분석
- Cross-function간 코드 흐름 분석 불가
  - ❖ P( ) // 피호출 Q() 인터페이스가 없는 부정확한 코드 분석
  - ❖ Q( ) // 호출 P() 인터페이스가 없는 부정확한 코드 분석

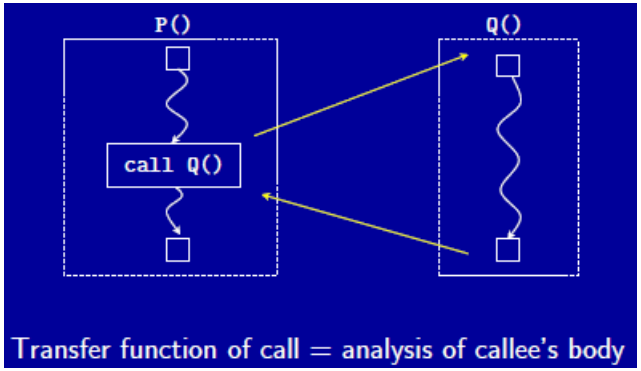
2.1 Inter-Procedural Analysis

◆ Inter-procedural analysis

➢ 프로그램 전체인 함수간 분석

✓ Cross-function의 데이터 값의 흐름 분석

- P( ) -> Q( ) -> P( )



➢ Good

✓ Call, Return의 정확한 프로그램 분석

- (도구 관점) 오탐 및 미탐의 비율 거의 없음

✓ 코드 점검 도메인 확장성

- Run-time 분석 및 보안 결함의 식별에 최적화된 데이터 흐름 분석 기법

✓ 사이버 보안 포함 정보 보호 서비스 관련 도메인의 확장 점검

- 암호, 인증 등 안전한 보안 기능의 정책 준수와 보안 정보의 안전한 사용까지 검증

➢ Bad

✓ Intra-procedural analysis의 정적분석도구 보다 코드 흐름 분석 시간 소요

➢ **관련 도구: RESORT(소프트4소프트), 외산 최상위 상용 도구(CodeSonar, Coverity, PolySpace 등)**

2.2 Intra-Procedural Analysis

◆ Intra-procedural analysis

- **한 함수내 또는 한 파일내 분석**
  - ✓ Cross-function의 데이터 흐름 분석을 위해 인터페이스의 코딩 등이 필요
- **Good**
  - ✓ 구조적/방어적 코딩 점검에 최적화된 정적 분석 기법

- **Bad**
  - ✓ Cross-function의 인터페이스 검증 불가로 부정확한 코드 점검
    - (코딩 관점) Call, Return 인터페이스의 검증 필요로 검증 조건절(true, false) 코드추가
    - (Rule 관점) 함수 인터페이스의 가정을 설정하기 위한 Rule Option 필요
    - (도구 관점) 오탐(false positive)과 미탐(false negative)의 발생 비율 높음
  - ✓ 코드 점검 도메인 제약성
    - Run-time 분석 및 보안 결함의 식별에 대한 구조적 프로그램 관점으로 점검 가능
  - ✓ 사이버 보안 포함 정보 보호 서비스 관련 도메인 점검의 한계
    - 암호, 인증 등 안전한 보안 기능의 정책 준수 점검

- **관련 도구: Open Source 도구, Scanner 및 보안 취약점 점검 상용 도구, 코딩 표준 가이드 점검 상용 도구 (국산, QAC, LDRA 등)**

2.2 Intra-Procedural Analysis - Cross Function 분석 불가

◆ Intra-procedural analysis의 정적분석도구 문제점

- ➢ (Juliet Test Case for Java) CWE476_NULL_Pointer_Dereference
  - ✓ String_22a.java -> String_22b.java의 Cross Function 분석 불가
    - String_22a.java의 “badSink(data)”에서 오탐 제공
    - String_22b.java의 “data.lenth()”에서 미탐 발생

```java
/*
 * @description
 * CWE: 476 Null Pointer Dereference
 * BadSource:  Set data to null
 * GoodSource: Set data to a non-null value
 * */

package testcases.CWE476_NULL_Pointer_Dereference;
import testcasesupport.*;

public class CWE476_NULL_Pointer_Dereference_String_22a
{
  public static boolean badPublicStatic = false;

  public void bad() throws Throwable
  {
    String data = null;
    data = null;          /* POTENTIAL FLAW: data is null */
    badPublicStatic = true;
    (new CWE476_NULL_Pointer_Dereference_String_22b()).badSink(data);
  }
}
```

Call

Intra tool의 data 오탐

- Caller String_22a.java -

```java
/*
 * @description
 * CWE: 476 Null Pointer Dereference
 * BadSource:  Set data to null
 * GoodSource: Set data to a non-null value
 * */

package testcases.CWE476_NULL_Pointer_Dereference;
import testcasesupport.*;

public class CWE476_NULL_Pointer_Dereference_String_22b
{
  public void badSink(String data) throws Throwable
  {
    if(CWE476_NULL_Pointer_Dereference_String_22a.badPublicStatic)
    {
      /*FLAW: null dereference will occur if data is null*/
      IO.writeLine("" + data.length());
    }
    else
    {
      data = null;
    }
  }
}
```

Inter tool의 data 정탐
Intra tool의 data 미탐

- Callee String_22b.Java -

- Cross Function 분석 불가로 오탐/미탐 -

3. Static Analysis Tool Comparison

◆ (고급/상위 도구) 코드 점검의 노력/시간 절감 (SW 개발 비용 절감)

➢ (Inter- vs. Intra Analysis) Cross-function 프로그램 분석 차이가 정적 도구의 검증 차이

✓ (Inter-procedural Analysis: True Path) Cross Function의 올바른 실행 순서로 정확한 흐름 분석

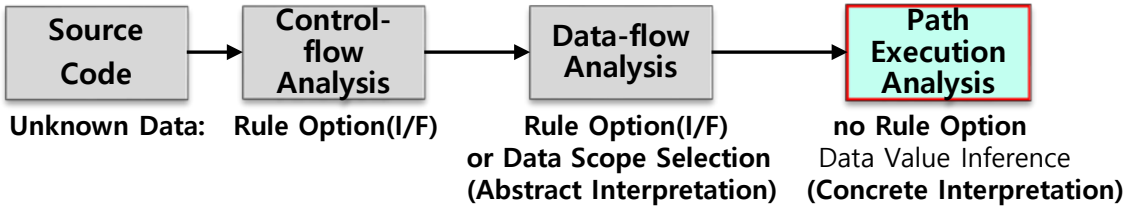
✓ (Intra-procedural Analysis: False Path) Cross Function의 인터페이스의 부정확한 코드 분석

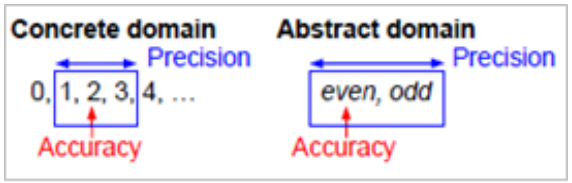
기능	Inter-procedural Path Analysis (RESORT)	Intra-procedural Analysis	코드 분석의 기능 차이점
코딩 가이드	O	O	(1) 코드 분석/검증 기법 Inter: 올바른 함수간 호출 분석과 조건 경로 흐름의 상태 값 분석으로 옵션이 필요하지 않으며, 오탐도 없음 Intra: 함수/파일 내의 제한된 코드 분석으로 외부 인터페이스 위한 옵션 설정 필요. 따라서, 점검마다 결함 결과 차이 발생; (오탐 증가 원인) 전역변수 등 외부 정의 등을 참조하는 프로그램의 경우, 오탐율 10% 이상 발생
런-타임	O	O	
보안 취약점	O	O	
코드 메트릭	O	O	
규칙 옵션 설정	X	O	
코드 점검 기술 (차이점)	데이터의 상태 범위/값, 경로 정보로 상태 값 재생산 (Path-Sensitive Analysis)	데이터의 상태 타입/범위 값, 경로 정보가 없어 상태 값 분석 불가 (Flow-Sensitive Analysis)	(2) 경로 별 정보 (Path-specific Information)
	True Path: 오탐 없음	False Path: 오탐 5~10%	(3) 결함 메시지 유형 Path Analysis:결함원인 및 라인 위치 제공 no Path Analysis:결함원인 추적으로 경로확인
	반복 점검 결과 동일	반복 점검 결과 다름	
	결함메시지:결함결과/원인	결함메시지:결함결과	

```
if (x == 0)
  ++x;          // x == ?
else
  x = 2;        // x == 2
y = x;          // x == ?, y == ?
```

```
// x == 1

// x == 2
// (x == 1, y == 1) or (x == 2, y == 2)
```





㈜소프트4소프트

본사: 서울 서초구 서운로1길 34 한국산업기술보호협회 3층
R&D: 대전광역시 유성구 가정북로 96 대전일자리경제진흥원 308호

Tel : 02-553-9464

기술 및 일반 문의 : info@soft4soft.com

www.soft4soft.com