안전한 프로그램을 위한 이론, 도구, 설계

보안약점분석기

SIGPL Workshop, KCC 2013

한양대학교 김현하

2013.6.28.

차례

- 보안약점분석기 동향
  - 국외 보안약점분석기
  - 국내 보안약점분석기
- 보안약점분석기 개발사례
  - 보안약점 분석
  - 보안약점을 기술하는 언어
  - 보안약점 분석도구

PLASSE RESEARCH

국내외 보안약점진단도구 동향

국외 보안약점도구

- Fortify
  - HP : http://www.hpenterprisesecurity.com
- AppScan
  - IBM : http://www-03.ibm.com/software/products/us/en/appscan/
- Coverity
  - Coverity : http://www.coverity.com
- CODESONAR
  - GrammaTech : http://www.grammatech.com/codesonar

PLASSE RESEARCH

Fortify

- HP Fortify Software Security Center
  - 2003년 California 주에서 설립, 2010년 HP에서 인수
  - 전세계 점유율 1위
  - 정적분석과 동적인 침투시험을 복합적으로 수행
  - 다양한 지원대상언어
    - ASP.NET, C, C++, C#, other .NET, COBOL, Java, JavaScript/AJAX, JSP, PHP, PL/SQL, Python, T-SQL, XML, ...

AppScan

- IBM Security AppScan Enterprise
  - 1998년 이스라엘 Sanctum Ltd. 에서 개발
  - 2004년 Watchfire로 인수
  - 2007년 IBM으로 인수, Rational 제품군에 포함
  - 2009년 Ounce Labs 인수
  - 2011-2013 Andromeda (Patrick Cousot 등 참여) 도구 포함
  - 지원대상언어
    - C, C++, Java, JSP, ASP.NET, VB.NET, C#

PLASSE RESEARCH

Coverity

- Coverity Security Advisor
  - 2002년 San Francisco에서 창립, Stanford Checker를 상용화
  - 적정분석과 동적분석을 모두 수행
    - Coverity Static Analyzer - C/C++, C#, Java 프로그램의 정적분석도구
    - 2008 Coverity Dynamic Analyzer - Java 프로그램의 Race condition, Deadlock 등을 검출
  - 지원대상언어
    - C, C++, C#, Java

PLASSE RESEARCH

CodeSonar

- GrammaTech
  - 1998년 Cornell 대학 연구팀에서 spin-off
  - 분석도구
    - CodeSurfer - 1999년, C/C++ 의 Program Slicing 도구
    - CodeSonar - 2005년, C/C++ 정적분석도구
    - Intel x86 기계어를 분석하는 버전 존재
  - 공동창설자 : Wisconsin 대학 Thomas Reps
  - 지원대상언어
    - C, C++

국내 보안약점도구

- BigLook Wass
  - 이븐스타 http://www.evenstar.co.kr
- Code-Ray
  - 트리니티소프트 http://www.trinitysoft.co.kr
- NEXCORE Code Inspector
  - SK C&C http://www.skcc.co.kr
- SecurityPrism
  - 지티원 http://www.gtone.co.kr
- Sparrow
  - 파수닷컴 http://www.fasoo.com

PLASSE RESEARCH

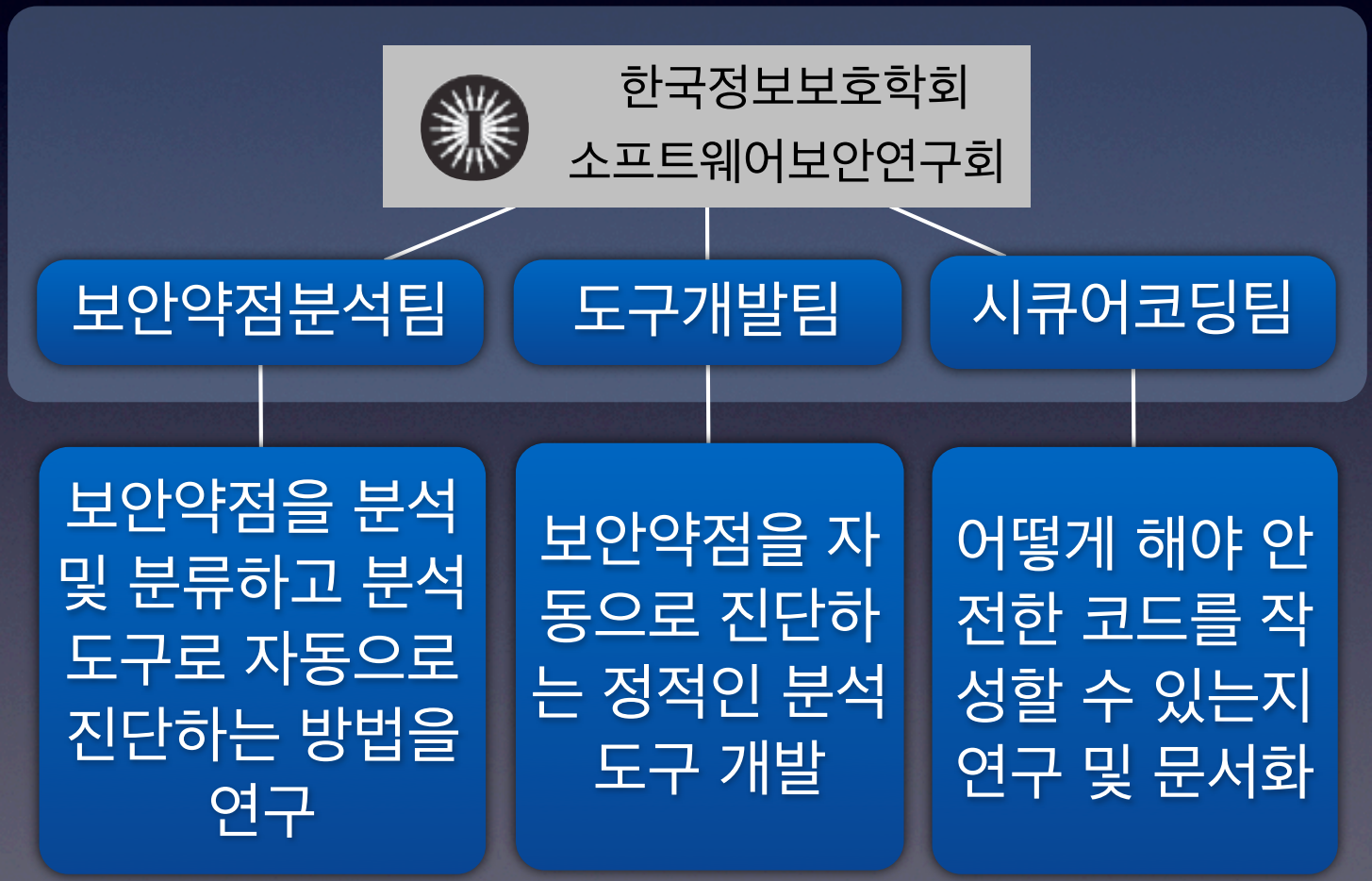
보안약점분석기 개발사례

PLASSE RESEARCH

밑그림

1. 소스코드 오류 및 보안약점 자동분석 국내기술배양
2. 해외 종속성 탈피 및 지속 발전을 위한 국가 차원의 사업
3. 해당 분야 최고의 전문가 그룹 협력 체제 필요

소프트웨어보안연구회 세부 팀별 역할



보안약점 분석

- 보안약점 분석단계
  - 정의: 어떤 보안약점인가
  - 원인: 무엇 때문에 발생하는가
  - 탐지: 어떻게 찾을 수 있는가
  - 방어: 어떻게 막을 것인가

PLASSE RESEARCH

보안약점 분석
예제 I

- SQL 삽입
  - 정의: 사용자의 입력값 등 외부 입력값이 SQL 질의문에 삽입되어 공격자가 질의문을 조작해 공격할 수 있는 보안약점
  - 원인: 외부 입력값이 (여과 없이) 질의문을 생성하는데 사용
  - 탐지: 외부 입력값의 자료흐름이 (여과 함수를 거치지 않고) 질의문을 생성하는 함수로 이어지는지 확인
    source : 외부입력함수의 결과 (예: inputStream()함수의 결과)
    sink : SQL 질의문 수행 함수의 입력(예: executeQuery() 함수의 입력)
  - 방어
    - 원인을 제거 : executeQuery 대신 preparedStatement 등을 사용
    - 문제를 방어 : 외부입력을 사용하기 전에 여과함수 장착

PLASSE RESEARCH

보안약점 분석
예제 2

- 크로스 사이트 스크립트(XSS)
  - 정의: 검증되지 않은 외부입력값에 의해 브라우저에서 악의적인 코드가 실행되는 보안약점
  - 원인: 외부 입력값이 (여과 없이) 웹문서를 생성하는데 사용
  - 탐지: 외부 입력값의 자료흐름이 (여과 함수를 거치지 않고) 웹문서를 출력하는 함수로 이어지는지 확인
    source : 외부입력함수의 결과 (예: inputStream()함수의 결과)
    sink : 웹문서 출력 함수의 입력(예: JspWriter() 함수의 입력)
  - 방어
    - 문제를 방어 : 외부입력을 사용하기 전에 여과함수 장착

PLASSE RESEARCH

유사한 보안약점

	SQL 삽입	크로스 사이트 스크립트
source	외부입력함수	
sink	SQL 질의문 실행함수	웹문서 출력함수
방어방법	외부입력함수의 결과가 여과함수를 거쳐서 사용되는지 확인	

유사한 보안약점

- 입력데이터 검증 및 표현 (보안약점 유형 1)
  - 신뢰할 수 없는 외부 입력이 검증없이 사용하는 취약점
- 에러처리 (보안약점 유형 5)
  - 에러를 처리하지 않거나 불충분하게 처리하여 발생하는 시스템의 불안정이나 정보누출과 관련한 약점

PLASSE RESEARCH

보안약점을 기술하는 언어

- [source] - [sink] 형식으로 보안약점을 기술
  - [inputStream] –> [executeQuery(rtn)]
  - [inputStream] –> [jspWriter(rtn)]
  - [exception] –> [println(rtn)]
  - ...

PLASSE RESEARCH

보안약점을 기술하는 언어

- [source] - [sink] 형식으로 보안약점을 기술
  - [inputStream] –> [executeQuery(rtn)]
- [source] - [filter] - [sink] 형식으로 보안약점의 방어방법을 기술
  - [inputStream] –> [filter(rtn)] –> [executeQuery(rtn)]

PLASSE RESEARCH

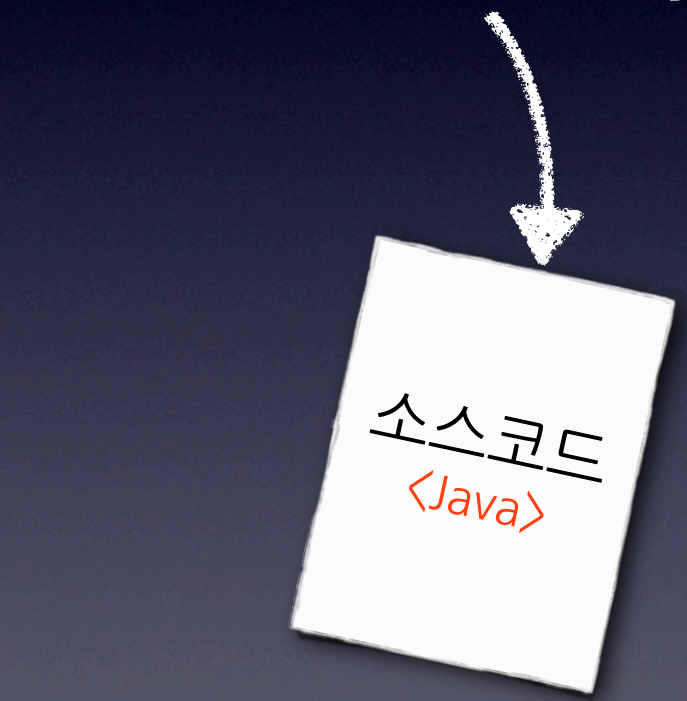
정적프로그램분석

- 프로그램을 실행해보지 않고 소스코드나 실행 파일을 분석해서 원하는 성질을 알아내는 기술
- 난이도에 따라 3단계로 구분
  - 토큰 수준 분석
  - AST 수준 분석
  - 의미분석

PLASSE RESEARCH

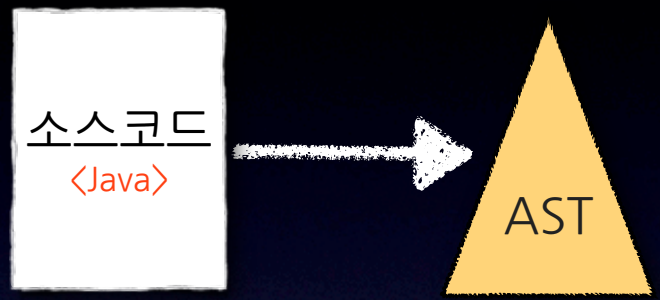
토큰 수준 분석

- 소스코드를 하나의 문자열로 취급
- 정규표현식 등의 패턴이 소스코드에 존재하는지 여부로 검사
- 예: 소스코드에 executeQuery 함수가 존재하는가

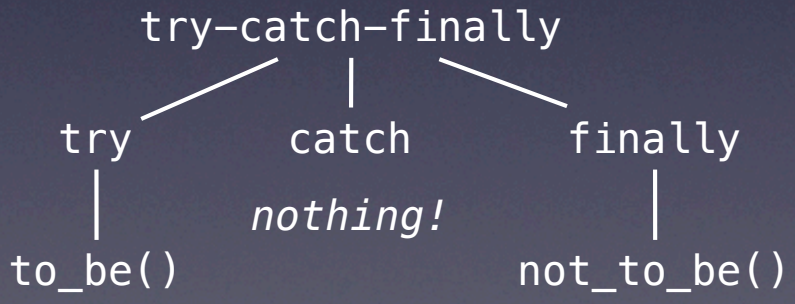


AST 수준 분석

- 소스코드를 파싱해서 얻은 AST수준에서 분석
- AST를 구조적으로 파악하는 방법으로 분석, 특정 구문요소를 잡을 수 있음
- 예: catch 구문 안이 비어 있는가

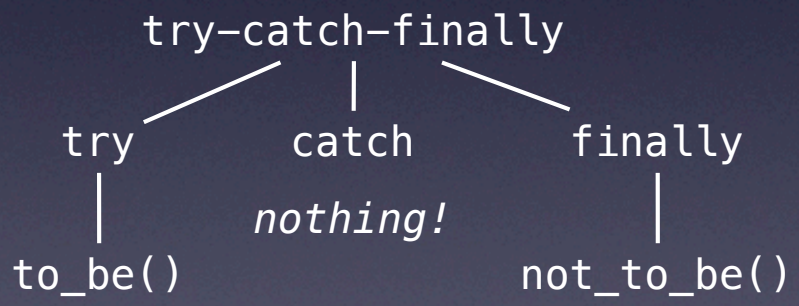



```
try {
 to_be();
} catch (Exception e) {
  /* do nothing */
} finally {
 not_to_be();
}
```



보안약점 기술 언어로 프로그램의 구조 탐지

[empty] in Catch



PLASSE RESEARCH

의미 수준 분석

- 흐름그래프 등을 생성해서 정적분석기술로 자세한 의미를 해석
- 단순한 패턴이 아닌 실제 실행과정을 고려한 분석결과로 검사
- 예: to_be 함수 이후에 not_to_be 함수가 호출되는가


```
try {
 to_be();
} catch (Exception e) {
  /* do nothing */
} finally {
 not_to_be();
}
```

보안약점 기술 언어로 제어흐름 탐지

[to_be()] -> [not_to_be()]

의미 수준 분석

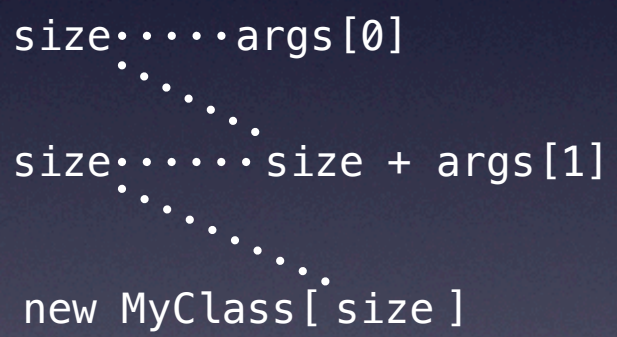
- 흐름그래프 등을 생성해서 정적분석기술로 자세한 의미를 해석
- 단순한 패턴이 아닌 실제 실행과정을 고려한 분석결과로 검사
- 예: to_be 함수 이후에 not_to_be 함수가 호출되는가


```
public static void main(String[] args) {

 int size = new Integer(args[0]).intValue();
 size += new Integer(args[1]).intValue();

 MyClass[] data = new MyClass[size];
 ...
```

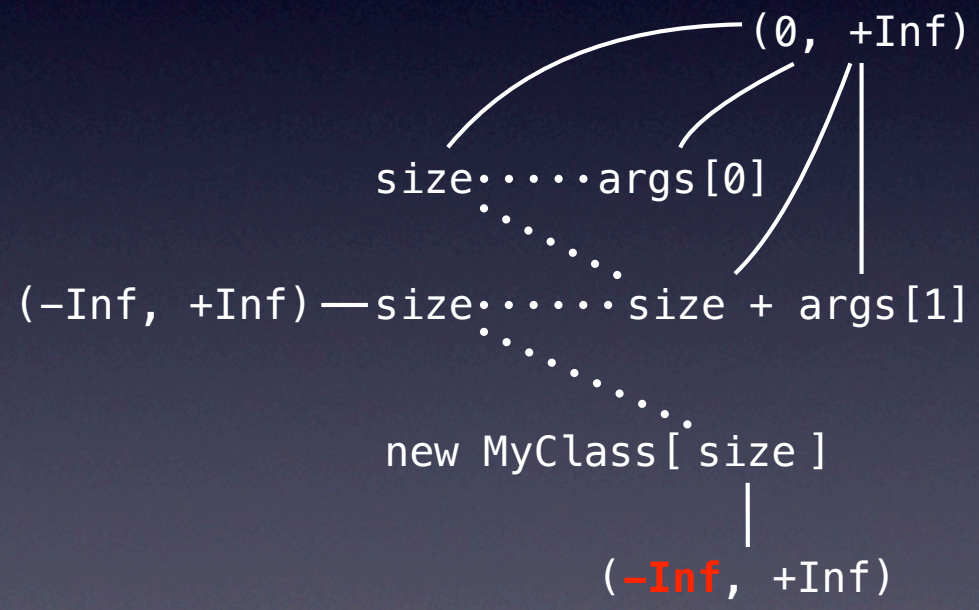
```
                                    args[0]= ?
 size·····args[0]                   size= ?
      ·····
 size·····size + args[1]            args[1]= ?
      ·····                         size= ?
      ·····
 new MyClass[ size ]                size >= 0 ?
```

보안약점 기술 언어로 자료흐름 탐지

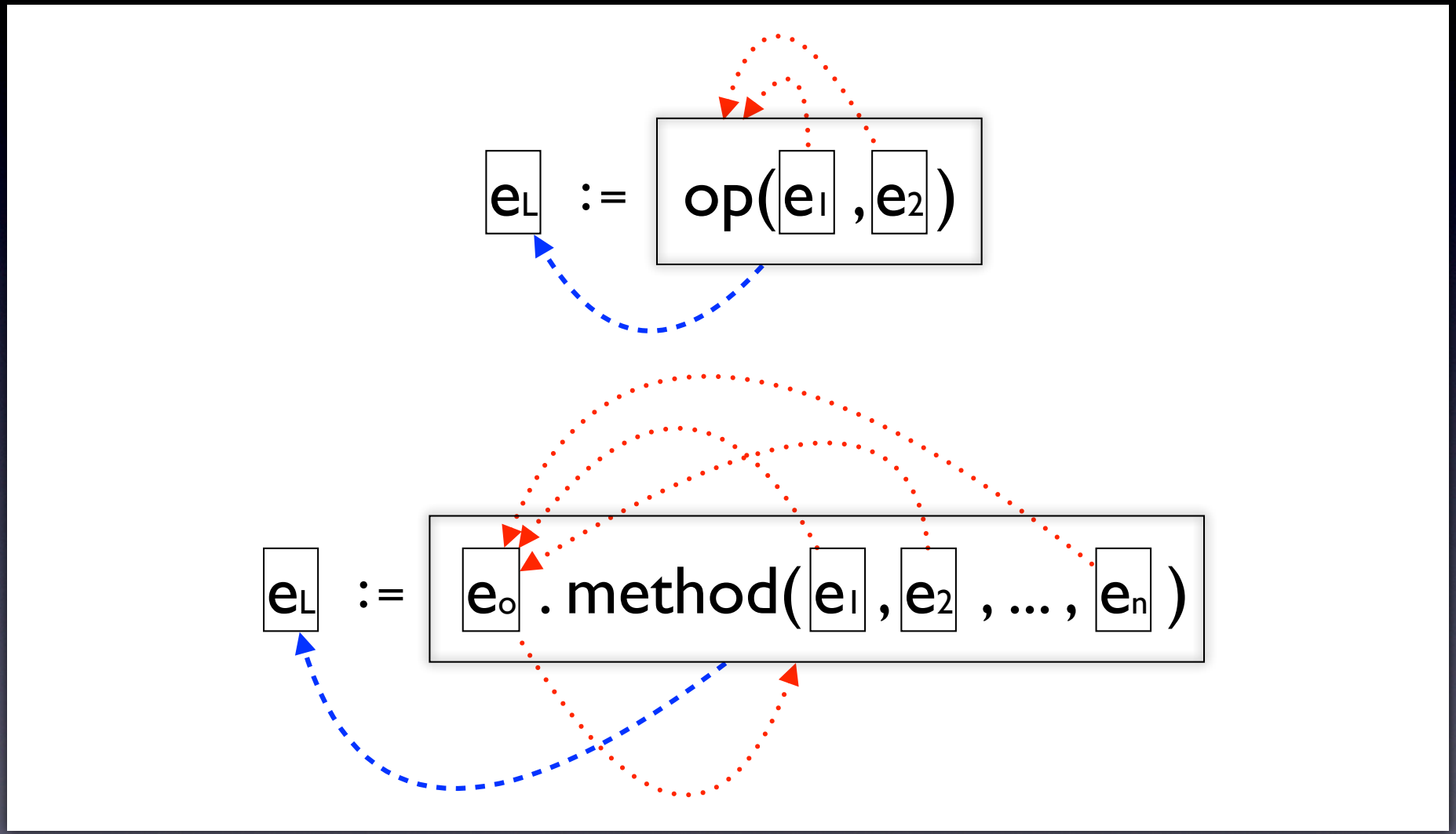
[extern] -> [arrayindex]



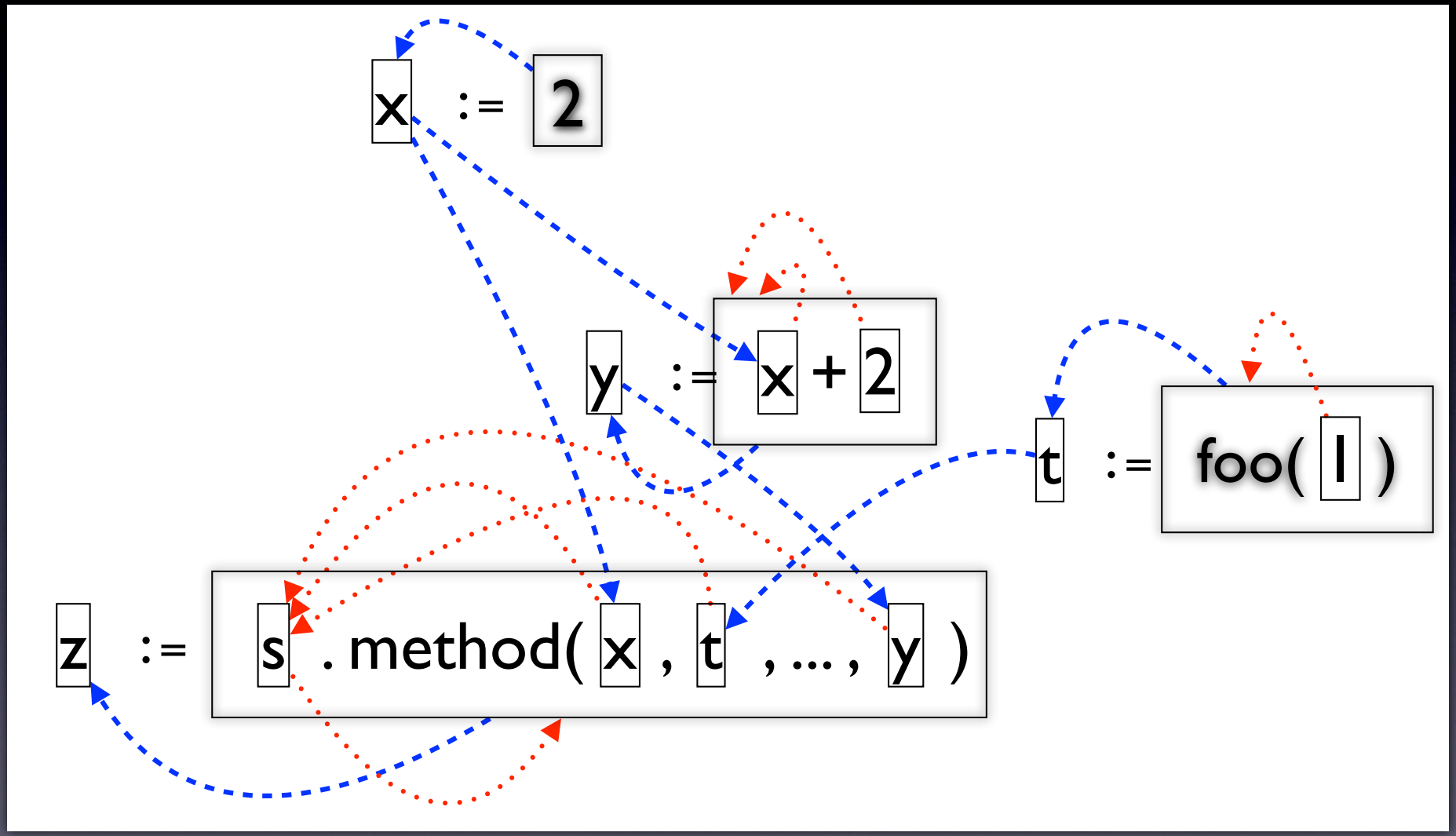
자세한 의미분석으로 보안약점 탐지



제어/자료 흐름그래프



제어/자료 흐름그래프



RDL Pattern Matcher 1.0beta-r572

test.cl

Code

Show position keys

```
1
public .Test extends java.lang.Object { }

6
package .Test(s:string, t:string) {
5 4     return;
}

10
package void .Test.foo(s:string) {
9 8     return;
}

15
package string .Test.bar(s:string) {
14 13     return 12s;
}

40
public static void .Test.main(args:(string:arr)) {
39 17     (.Test:class) test;
20      18test := 19new .Test();
21      string s;
24      22s := 23"a string";
27      (26test:.Test).foo(string:25s);
28      string t;
33      29t := 32(31test:.Test).bar(30s:string);
37      (36test:.Test).zoo(string:34t, string:35s);
38      return;
}
```

Matched position: 27

TestRule-00:[foo($1)]	
f	Test.foo
$1	25

Matched position: 37

TestRule-00:[zoo($0, $1)]	
f	Test.zoo
$0	34
$1	35

test.xml

3 Matched Locations

TestRule-00 TestRule-00 TestRule-00

3 Matched patterns

[bar($1)]
[foo($1)]
[zoo($0, $1)]

Rules containing matched patterns

RULEID TestRule-00

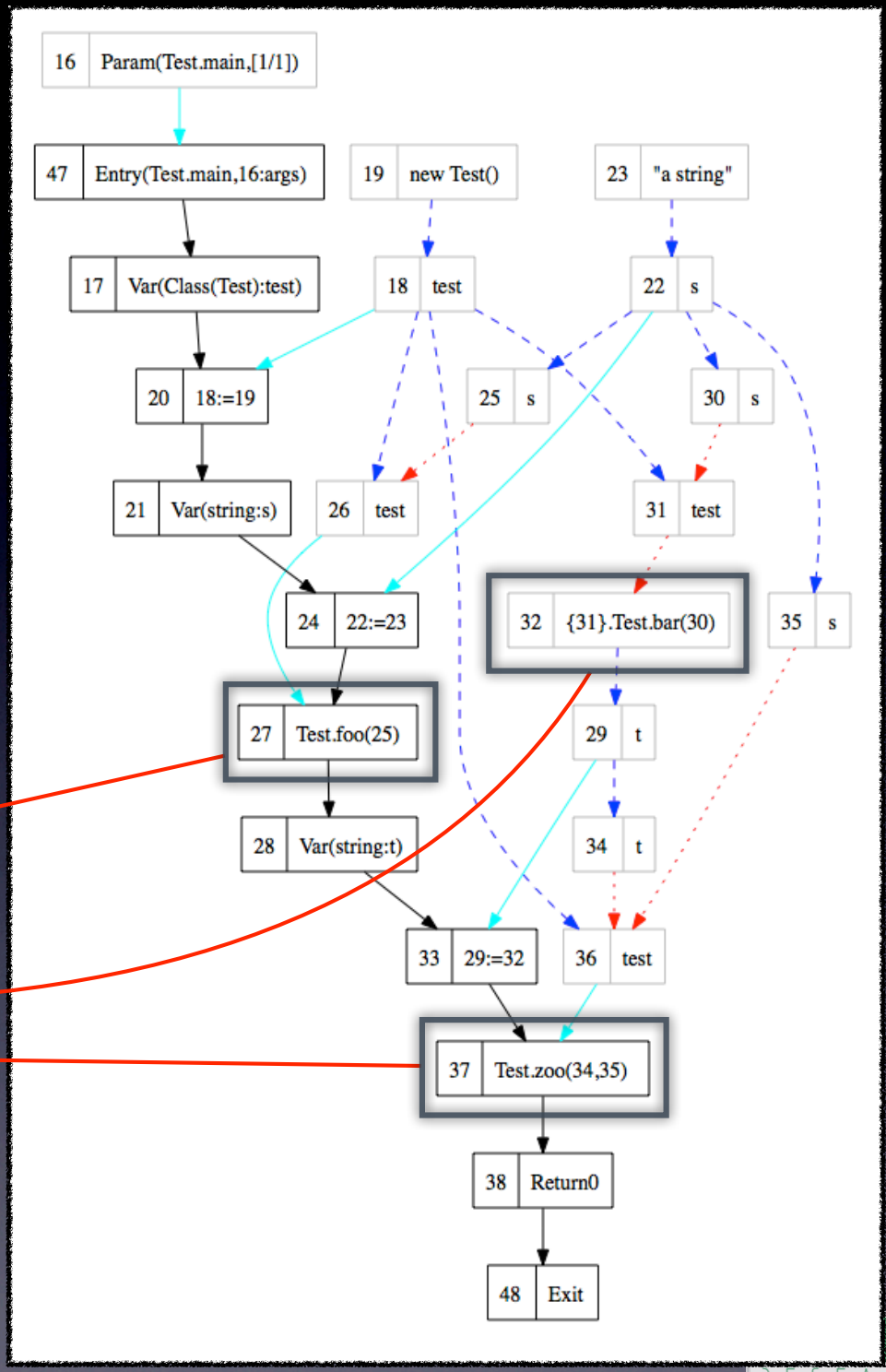
UNSAFE [foo($1)] -->df [bar($1)] -->df [zoo($0, $1)]

Other rules

Groups

흐름그래프 추적

```
40
public static void .Test.main(args:(string:arr)) {
39 17     (.Test:class) test;
20      18 test := 19 new .Test();
21      string s;
24      22 s := 23 "a string";
27      (26 test:.Test).foo(string:25 s);
28      string t;
33      29 t := 32 (31 test:.Test).bar(30 s:string);
37      (36 test:.Test).zoo(string:34 t, string:35 s);
38      return;
}
```



감사합니다