SCALE2 활용 프로그램 분석

이명륜(2015124147)

지도교수: 안준선 교수

요 약

본 논문에서는 SCALE2를 활용하여 프로그램을 분석하는 데 도움을 주기 위해 SCALE2를 발전시키고, 분석기간의 성능을 비교하고 이 결과를 시각적으로 나타낸 뒤, SCALE2에 업로드하는 구현을 보인다. 여러 분석기의 정보의 공통점과 차이점을 살펴 보면 분석기별 특징을 알 수 있고 false positive 문제를 해결하는데 도움을 줄 수 있다.

Keywords : 정적분석 , secure coding , SCALE2

Ⅰ. 서 론

정적 프로그램 분석은 프로그램을 실행하지 않고 에러를 탐지하는 것이다. 정적 분석 기술을 사용하면 프로그램 취약점을 사전에 발견할 수 있지만 탐지오류(false positive,false negative) 문제가 있다. false positive는 오류가 아닌데 오류라고 보고한 것을 말하고, false negative는 오류인데 오류가 아니라고 보고한 것을 말한다.

분석 결과에서 false positive 비율을 줄이는 방법으로 여러 분석기로 분석을 수행한 뒤, 분석기들이 동시에 찾은 오류는 false positive가 아닐 가능성이 크고 그렇지 않으면 false negative일 가능성이 크다는 점을 이용할 수 있다.

SCALE2는 여러 정적분석 도구의 결과를 통합해서 보여주는 도구이다. 오픈 소스이며 분석에 도움을 줄 수 있는 다양한 기능(filter에 따른 검색, source code viewer, 분석 결과 export)을 제공해서 auditor가 오류를 최종적으로 판정하는데 도움을 준다.

본 논문에서는 동일한 소스코드 베이스에 대해서 cppcheck과 codemind라는 분석기로 분석을 진행했으며 각 분석기의 특징을 살펴보고, 공통적으로 발견한 오류와 베타적으로 발견한 오류를 분류하여 표로 나타냈다. 그리고 나서 CWEs를 기준으로 오류를 분류하여 그래프로 나타내었고 SCALE2에 그래프 업로드 기능을 추가하여 SCALE2에 업로드하여 통합했다.

Ⅱ. 본 론

1. SCALE2 시스템

그림 1에 SCALE2 시스템 동작이 설명되어있다. SCALE2에서 소스 코드들은 다양한 분석기에 의해 분석된다. 분석 결과를 SCALE에 업로드하면 발견된 오류에 대응되는 CWE 규칙이 없으면 Secure Coding Filters에 의해 걸러지고 결과에 표시되지 않는다. Secure Coding Filter를 통과한 오류들은 데이터베이스에 저장되어 나중에 오류를 평가할 때 사용된다.

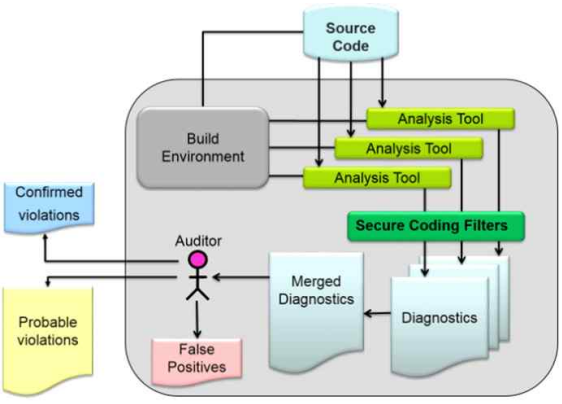


그림 1. SCALE2 process

2. 구현

가. codemind 분석기 추가

SCALE에 등록되어 있지 않은 codemind라는 분석기를 추가했다. 그러기 위해서 codemind 분석 결과를 파싱하여 org파일 형식으로 만드는 파이썬 스크립트를 작성했고, codemind rule code를 CWE/CERT 규칙에 매핑시키는 규칙표를 작성했다. 이렇게 두 가지를 작성하면 SCALE2에서 digest_diagnostics.py가 이를 이용하여 SCALE2에 업로드할 데이터베이스를 만들어주기 때문에 SCALE2에서 codemind 분석기를 이용할 수 있게 된다.

나. export한 파일 자동 분석 스크립트 작성

SCALE2에서 분석 결과는 csv 또는 db 형식으로 export할 수 있다. 본 논문에서는 csv로 export해서 분석을 진행했다. “파일명+라인넘버 “를 key로 하여 {key,분석기명},{key,CWE규칙},{CWE규칙,발견 횟수} 등과 같은 dictionary 자료형 여러 가지를 만들어 분석을 했다. cppcheck과 codemind에 분석기가 발견한 오류가 몇개씩 발견되었는지, 또 어떤 CWE/CERT 오류가 몇개씩 발견되었는지를 그래프로 분석해주는 스크립트를 작성했다. 입력으로 파일을 주면 출력으로 분석 결과 그래프를 얻게 된다. 이 스크립트를 SCALE2에 통합하면 더 자동화된 시스템을 만들 수 있을 것이라 예상되지만 아직은 별개의 스크립트로 구성되어 있다.

다. SCALE2에 분석 그래프 업로드 기능 추가

SCALE2에 분석그래프를 업로드하기 위한 시스템을 추가하였다. SCALE2는 ruby on rails framework로 이루어져 있어서 MVC(Model View Controller) 모델을 따른다. View에서 보여지는 부분을, Controller 부분에서 backend 로직을 작성하여 업로드 시스템을 구현했다.

Ⅲ. 실험

총 107개, 207932 라인의 소스코드에 대해 cppcheck, codemind로 분석을 진행했다. 분석 결과는 표 2에 나타나 있다.

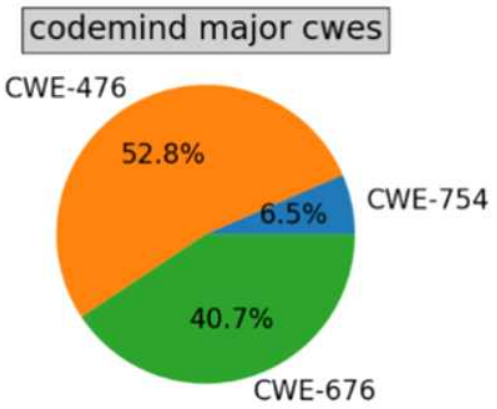
항목	내용
프로젝트명	sql3.21new
소스코드 파일 수	107개
소스코드 라인 수	207932 라인

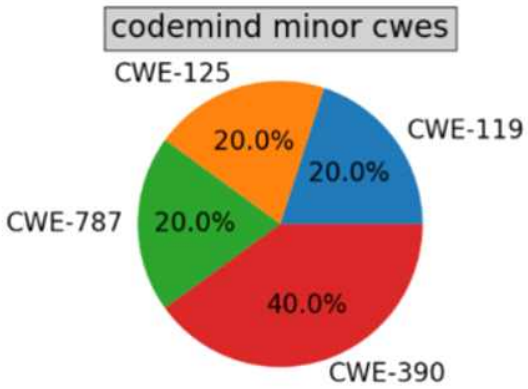
표1. 분석 대상

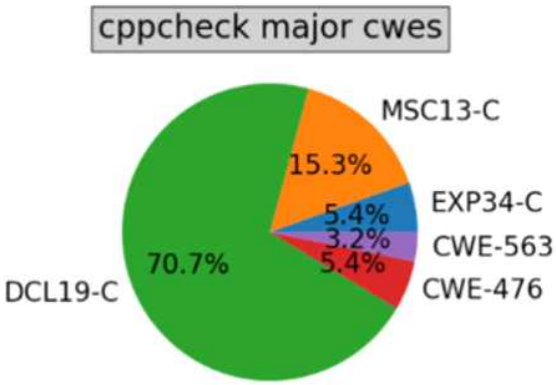
분석기	오류 종류	오류 개수
cppcheck	26	487
codemind	7	128

표2. 분석 결과

각 분석기마다 발견한 주요 오류가 무엇인지 파악하기 위해 오류들을 major 또는 minor 오류로 나누었다. cppcheck이 더 많은 오류를 보고 했고, codemind는 비교적 적은 오류를 보고해서 cppcheck에서는 10회 이상 발견시 major 오류, codemind에서는 5회 이상 발견시 major 오류로 분류했다.







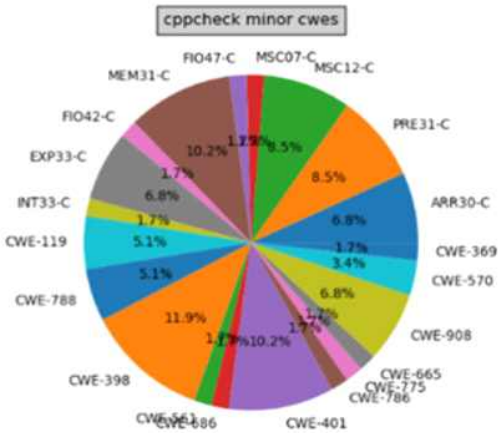


그림2.분석 결과

주요 오류	codemind	cppcheck
1	CWE-476	DCL19-C
2	CWE-676	MSC13-C
3	CWE-754	CWE-476

표 3. 분석기별 주요 오류

codemind에서는 CWE-476(Null Pointer Dereference),CWE-676(Use of Potentially Dangerous Function),CWE-754 (Improper Check for Unusual or Exceptional Conditions) 순으로 많이 탐지되었다.

cppcheck에서는DCL19-C(minimize the scope of variables and functions), MSC13-C(Detect and remove unused values), CWE-476(NULL Pointer Dereference) 순으로 많이 탐지되었다. cppcheck에서 많이 발견된 오류인 DCL19-C ,MSC13-C는 최적화와 관련된 오류들이었고 codemind에서는 이러한 오류가 전혀 탐지되지 않은 것이 codemind와 cppcheck 간에 오류 보고 개수 차이에 기인했다.

Ⅳ. 결 론

본 논문에서는 분석정보를 통합하여 보여주는 SCALE2에 기능을 추가한 시스템을 제시했다. 분석기별 특성 파악 및 여러 분석기 정보를 이용해 false positive,false negative를 판단하는데 도움을 줄 수 있도록 SCALE에 tool을 추가하기, 분석 결과

파싱 스크립트,통합 분석 결과로부터 분석 그래프를 얻는 스크립트, SCALE2 wep app 코드를 구현했고 자세히 설명했다. 최종적으로 그래프로 각 분석기가 탐지한 CWE 오류 개수를 보여주었고, 그로부터 각 분석기의 특징을 도출했다.

참고문헌

[1] Robert C. Seacord, Will Dormann, James McCurley, Philip Miller, Robert W. Stoddard, David Svoboda, Jefferson Welch ,*Source Code Analysis Laboratory (SCALe)* ,Software Engineering Institute,1,2012