Is There Value in Reasoning about Security at the Architectural Level: a Comparative Evaluation

Ebrahim Khalaj  
Radu Vanciu  
Marwan Abi-Antoun  

{mehalaj, radu, mabiantoun}@wayne.edu

Method

We want to compare tools that find security vulnerabilities. We propose a benchmark with hand-selected testcases.

Different resources for testcases used to build ScoriaBench:
- DroidBench(DB)  
  - SAMATE Reference Dataset (SRD)  
  - CERT rules examples  
  - Designed by us (US)

Comparison Metrics

We compare approaches in terms of:
- True Positives (TP): a real vulnerability reported by the tool. Higher is better.
- False Positives (FP): a vulnerability that does not exist but is reported by the tool. Lower is better.
- False Negatives (FN): a real vulnerability that is missed by the tool. Lower is better.

Then we use them to calculate comparison metrics:

Precision = (TP)/(TP+FP)
Recall = (TP)/(TP+FN)

Results

Reasons about information flow at the level of variables. FlowDroid creates an information flow graph: nodes represent variables and edges represent assignments.

Constraint: FlowDroid looks for transitive information flow from a source to a sink.

(\text{C1}, \text{md1}, \text{Property1}) \rightarrow (\text{C2}, \text{md2}, \text{Property2})

FlowDroid Constraint for AChat:

(\text{ContactsProvider}, \text{getNumbers}, \text{IsConfidential}) \rightarrow (\text{PrintStream}, \text{println}, \text{Untrusted})

ScoriaBench

A benchmark with testcases that are grouped in different equivalence classes. Some testcases focus on architectural flaws.

Architectural flaw  
e.g., missing authentication

number of testcases from different resources grouped in equivalence classes

Example

Our goal is to find security vulnerabilities such as information disclosure.

AChat: A chat application that discloses confidential information of its users to a malicious client. This example represents an exploitable service.

*Inspired by SnapChat data exploit on Dec 2013.*