Improving software security with Infer Quandary via confusion matrix perturbation analysis

ICT Risk Assessment final project

Giovanni Bartolomeo and Laura Bussi

1 Introduction

When dealing with security issues, we are provided a pletora of tools for the analysis of possible vulnerabilities in software. Many of them analyse the behaviour of the software at run time, but static analysis can also be performed: several tools can analyse source code and find possible flaws before the program is running.
Of course, both this kind of approaches cannot be 100% accurate. Most likely they will provide as result a set of possible vulnerabilities which intersect the set of the actual ones, i.e. for each pointed out vulnerability we will have four possible cases:

- **True Positive** Tool correctly identifies a real vulnerability
- **False Negative** Tool fails to identify a real vulnerability
- **True Negative** Tool correctly ignores a false alarm
- **False Positive** Tool fails to ignore a false alarm

From this classification, we obtain a 2 × 2 matrix, namely a confusion matrix. Confusion matrix can be a useful tool to benchmark a security analysis tool capabilities.

1.1 OWASP Benchmark

The OWASP benchmark project is an executable web application, provided as a Maven project, written in Java using the javax framework. It contains several Java files (slightly less than 3000): each one of this files is a Java servlet wich can contain either a true vulnerability or a false positive. As the project is provided as a public repository on GitHub, one can run both dynamic application security testing tool or static vulnerability analysis tools against it.
Of course, we are also provided with a csv containing the expected results for each test case, meaning that for each test we have:

- The name of the test case;
- The vulnerability area;
- A boolean flag indicating if the vulnerability is a true or a false positive;
- The CWE number of the vulnerability.

Once the analysis is complete, we are provided with a score, called the Benchmark Accuracy Score, in the range 0-100. This score is a Youden index, computed as:

$$J = sensitivity + specificity - 1$$

where in turn we have:

$$sensitivity = TP/P = TP/(TP + FN)$$

and

$$specificity = TN/N = TN/(TN + FP)$$

Thus, we have that sensitivity represents the ability of recognizing a true positive, while the specificity is the ability of correctly identifying true negative. Looking at the formulas, it is clear that a tool which label each line of code as a vulnerability has a very low sensitivity, as the number of False Negatives is very high. At the very same way, a tool which does not recognize any vulnerability has sensitivity 0, as the TP factor is nullified. A similar consideration can be done for specificity.

1.2 Infer Quandary

Facebook Infer is an open source static code analyser written in OCaml. Based on abstract interpretation, it is not primarily intended to be used to discover security issues but, more generally, possible errors in the source code. This obviously means that it can shows several limitations for this kind of purpose: however, it seems interesting to test it against the Owasp benchmark, in order to see how can it be used and improved for security.
Infer is provided with a plugin, namely Quandary, devoted to the taint analysis of the source code. Quandary can be configured by providing a JSON file, where one can define:

- a list of sources for the input data;
- a list of sink procedures, which may use tainted inputs;
- a list of sanitizer procedures;
- a list of endpoints.

As the OWASP benchmark is a web application, tainted inputs come from servlet procedures, which are not included in the Quandary configuration by default. This means that, at a first run, Quandary got a score 0: in the next sections we are providing some configurations which can improve Infer Quandary's performances in a web environment.

2 Project structure

Prova

3 Benchmark results

Prova

4 Conclusions

Prova