Project in advanced programming - Outline

Edo Cohen	Tzafrir Rehan	Gai Shaked
039374814	039811880	036567055
sedoc@t2	tzafrir@cs	gai@tx

November 1, 2010

We intend to implement a static Buffer Overrun Analyzer (boa). Boa will receive a C program as input, and determine whether a buffer overrun is possible during execution of the code. This must be done with care - while most static analysis methods result in some false positives, extra care should be taken to ensure no false negatives (measures not taken in [1, 2]). We refer to the lack of false negatives as the *soundness* of the analysis.

Boa will be implemented in two stages, the first will be context and flow insensitive analysis, which will be sound under the assumption of no pointer manipulation regarding buffers. However, context and flow insensitivity is prone to false alarms. Therefore, we intend to test limited flow and context sensitivity, which we hope will be able to reduce the amount of false positives without leading to false negative results. For this second phase we will also require that the input C code not include -

- concurrency
- goto statements

The implementation will be based on Clang[4] as the static analysis front end. Clang API will be used to generate integer linear programming constraints for each buffer and integer in the code. These constriants will model the max and min used (and allocated) index for each buffer. Finally we will use GLPK[5, 6] to solve the integer linear programming problem designated by the constriants, and the solution will determine whether buffer overrun is possible.

If time allows, we will also explore the idea introduced in [3]. By performing symbolic execution on the sites of the possible buffer overflows, we can further differentiate between false positives and true vulnerabilities - improving the overall quality of the analysis.

References

[1] Buffer Overrun Detection using Linear Programming and Static Analysis

[2] A First Step Towards Automated Detection of Buffer Overrun Vulnerabilities

[3] Filtering False Alarms of Buffer Overflow Analysis Using SMT Solvers

[4] http://clang.llvm.org/

[5] http://en.wikipedia.org/wiki/GNU_Linear_Programming_Kit

[6] http://www.gnu.org/software/glpk/