EÖTVÖS LORÁND UNIVERSITY

FACULTY OF INFORMATICS

Thesis Registration Form

Student's Data:

Student's Name: Cristea Andrei

Student's Neptun code: W61RAB

Course Data:

Student's Major: Computer Science BSc

I have an internal supervisor

Internal Supervisor's Name: *Gera Zoltan*

Supervisor's Home Institution: ***Department of Programming Languages and Compilers***

Address of Supervisor's Home Institution: ***1117, Budapest, Pázmány Péter sétány 1/C.***

Supervisor's Position and Degree: *docent*

Thesis Title: Static security code analysis

Topic of the Thesis:

(Upon consulting with your supervisor, give a 150-300-word-long synopsis os your planned thesis.)

C++ is a powerful and quite helpful language in terms of writing complex, performance-oriented applications. It has a lot of advantages and gives to the programmer a vast toolset for different purposes, but it is still really hard to write good, qualitative code in C++. Moreover, even after the release of the C++20, there is still a big chunk of legacy code written in the other versions of this language. Some of the most dangerous bugs which could happen in C++ (since it has an access to system-level resources) would be undefined or implementation-dependent behaviour. This could be caused by hidden mistakes like division by zero, which could happen on one of the methods inside a big project codebase leading to hours, maybe even days of lost time, nerves, and money. To avoid it we need to secure a programmers which are using C++ from making such mistakes. This is where static analysis becomes useful.

Static analysis of the methodology is testing software without actually running it. The static analysis tool is providing a unique code analysis to detect bugs and focuses on finding undefined behaviour and dangerous coding constructs.

For making rules for this tool, we will probably use SEI CERT C++ coding rules. Moreover, the plan will be to implement it by adding a new patch for a Clang compiler (maybe even using the Tidy framework underneath).

Therefore, the overall goal of this thesis is to extend standard compiler error and warning detections to help avoid code security vulnerable projects from successful compiling, this way provisioning the end-user with safe software and programmer with nice API interaction.

Budapest, 2022. 06. 15.