Chair of Decentralized Systems Engineering
Department of Informatics

Performance Analysis for Run-time Verification

General information	**Advisor**	Dr. Myoung Jin Nam
	Email	Myoungjin.nam “at” tum email address
	Date	25.01.2022
Type	Master / Bachelor / Guided Research	
Description	To ensure system integrity, safety, and security, it is inevitable to run a program and check errors due to the limitation of static analysis, which verifies and validates a system without execution. Despite advances in run-time verification mechanisms (dynamic analysis), performance degradation still remains an open problem. Most dynamic analysis solutions manage and utilise supplementary data (metadata) to check errors, and this metadata management creates cache impact; thus causes severe overhead during execution. This cause has been one of the main bottlenecks for practical use of dynamic analysis solutions, along with increased executed instructions. Another crucial point in the aspect of performance is that overhead is not only heavy but also *unpredictable*. Run-time verification approaches, especially inserting run-time checks into a program (*inline* approaches), suffer from more unpredictable overhead than *concurrent* approaches. Some portion of overhead can be largely resolved with optimisation or hardware support, or some kind of overhead cannot be resolved with support. Hence it is important to predict overhead of each approach that is built on a different design and metadata management.	

Chair of Decentralized Systems Engineering
Department of Informatics

	This research aims to *shape* performance degradation of run-time verification mechanisms, and show how much of overhead can be resolved with optimisation or other ways of acceleration. This work is important not just to improve certain run-time verification mechanisms, but to have an insight for a new design of run-time verification in general, or a new safe programming language.
Keywords	Software Security, Compiler, LLVM, Program Transformation, High performance, Profiling
Goals	**Concrete outcomes** 1. Program analysis and transformation using LLVM to measure overhead 2. Profiling performance degradation caused by software debugger or security solutions. **Bonus points** 3. Improve performance of existing run-time verification tools.
Prerequisites	**Compulsory** ● Familiar with C and C++ ● Taken courses in Compiler covering LLVM **Preferred** ● Experience in performance profiling tools (Perf, PCM, etc) ● Experience in debuggers (GDB, Valgrind, etc) ● Some knowledge in FPGA
References	1. Inline and sideline approaches for low-cost memory safety in C 2. AddressSanitizer: A Fast Address Sanity Checker 3. SoK: Eternal War in Memory
Application process	Please send an email to the advisor including the following: ● Email subject: “Thesis application (DSE)” ● CV ● A copy of your transcript(s) ● A **motivation statement**, please include samples of your work that you are proud of (e.g., major projects, open-source contributions, Github page, etc.) and/or writing samples (e.g., your technical blog, project reports, etc.)

Chair of Decentralized Systems Engineering
Department of Informatics

Prof. Pramod Bhatotia