A Platform for Enhancing Security of Binary Code

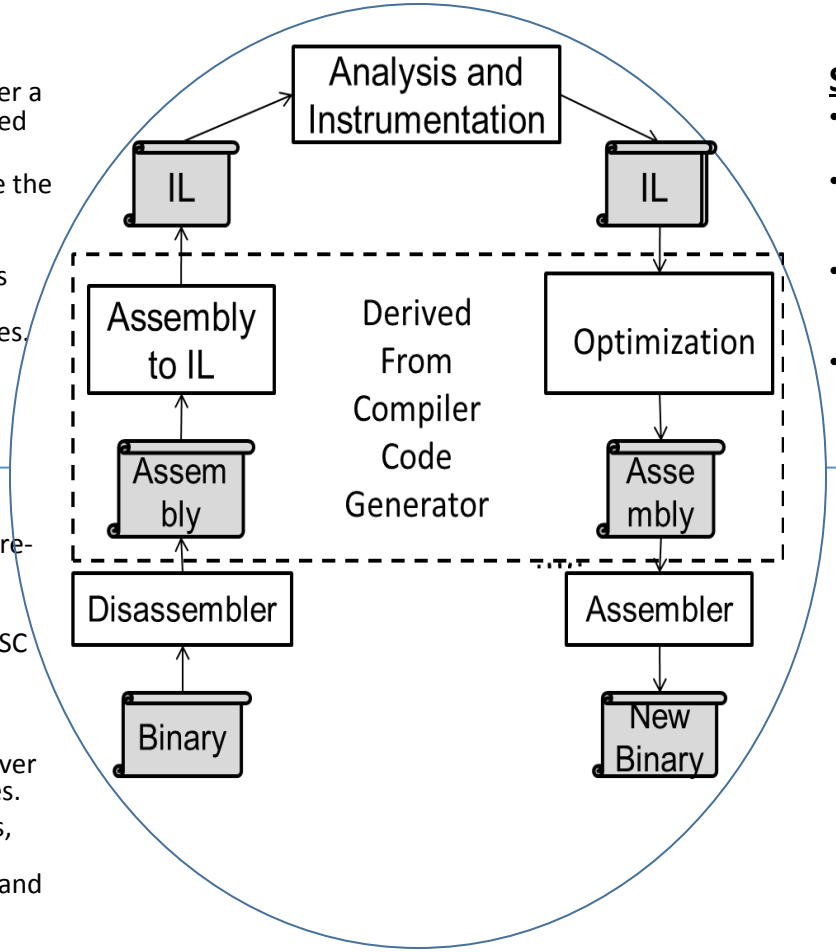
Stony Brook University

Challenge:

- Processor instruction sets are large and complex: both x86 and ARM support over a thousand opcodes, with extensions added frequently.
- IoT and control systems further increase the number of architectures of interest.
- Previous techniques rely on manual instruction semantics modeling, which is error-prone and highly labor-intensive, when considering this many architectures.

Solution:

- Develop a novel approach to extract instruction semantics from compilers, specifically, GCC.
- Lift instructions to compiler's architecture-neutral intermediate language (IL)
- Analyze/Instrument this IL.
- Developed two open-source systems, LISC and EISSEC for lifting binaries to IL.
- Developed an open-source platform for *static* instrumentation (PSI). On system benchmarks, PSI reduces overhead by over 10x, as compared to previous techniques.
- Developed several key instrumentations, including BinCFI (best paper award at USENIX Security), robust shadow stack, and secure code loading.



Scientific Impact:

- Software hardening and sandboxing are often based on instrumentation
- Binary instrumentations are more easily and widely applicable than source-based techniques
- This project develops techniques to greatly ease (the otherwise very hard) task of binary analysis/instrumentation
- Our focus is on architecture neutrality, so that techniques are easily applied to desktop, mobile and embedded systems that use diverse processors.

Broader Impact:

- Software security is an increasingly important and challenging problem facing the society.
- Our techniques make it easier to secure benign software against exploits, while providing means to contain untrusted software.
- Several open-source releases have resulted from this project.
- Several PhD students have been trained on this project. Components of the work have been integrated into grad and undergrad courses.

Award Number: CNS-1319137
Principal Investigator: R. Sekar
sekar@cs.stonybrook.edu
Open source downloads from http://seclab.cs.sunysb.edu/seclab/download.html