TWC: Medium: Automating Countermeasures and Security Evaluation against Software Side-channel Attacks

PIs: Yunsi Fei, Thomas Wahl, and Adam A. Ding, Northeastern University

http://tescase.coe.neu.edu, {y.fei, a.ding, t.wahl}@northeastern.edu

The objective of this project is to minimize side-channel leakage of software automatically. We shift the paradigm of side-channel attack research from a manual process to automatic implementation, from specific attacks and systems to common general methodologies, from ad hoc subjective measures to rigorous and objective security validation and evaluation.

Countermeasures against side-channel attacks

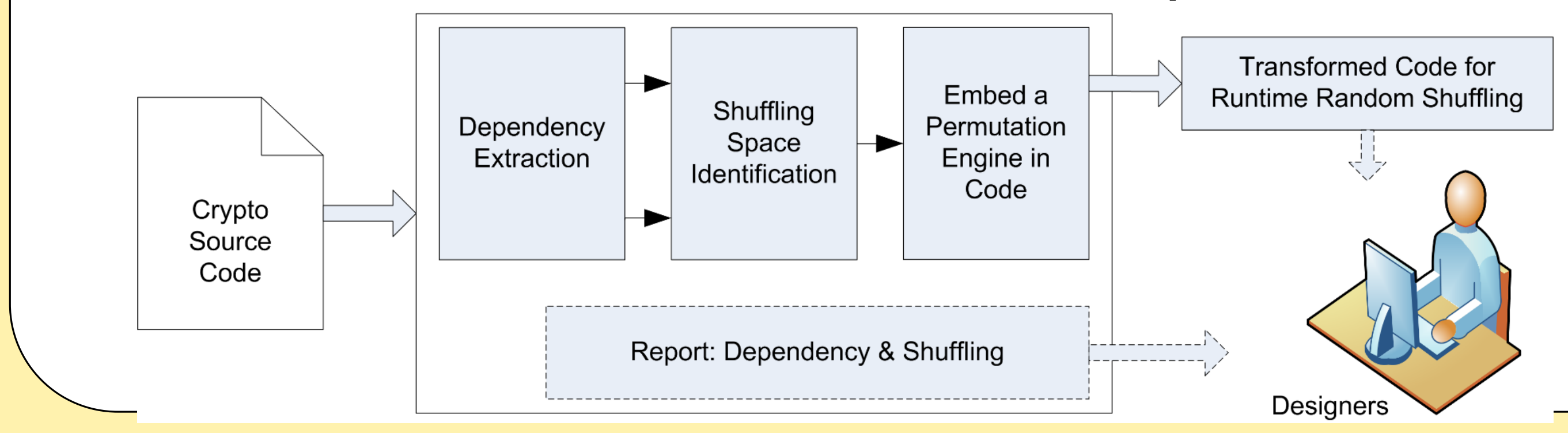
- Masking (secret sharing): randomized addition
- Operation shuffling: temporal variation of leakage
- Hiding: level power consumption with power balance circuits and ISA

Challenges

- Early side-channel leakage detection
- Automatic countermeasure protections at compile-rime and run-time
- Quantify the side-channel resilience and provide security guarantee

Preliminary Results

- Source code transformation for operation shuffling



Crypto Source Code

Dependency Extraction

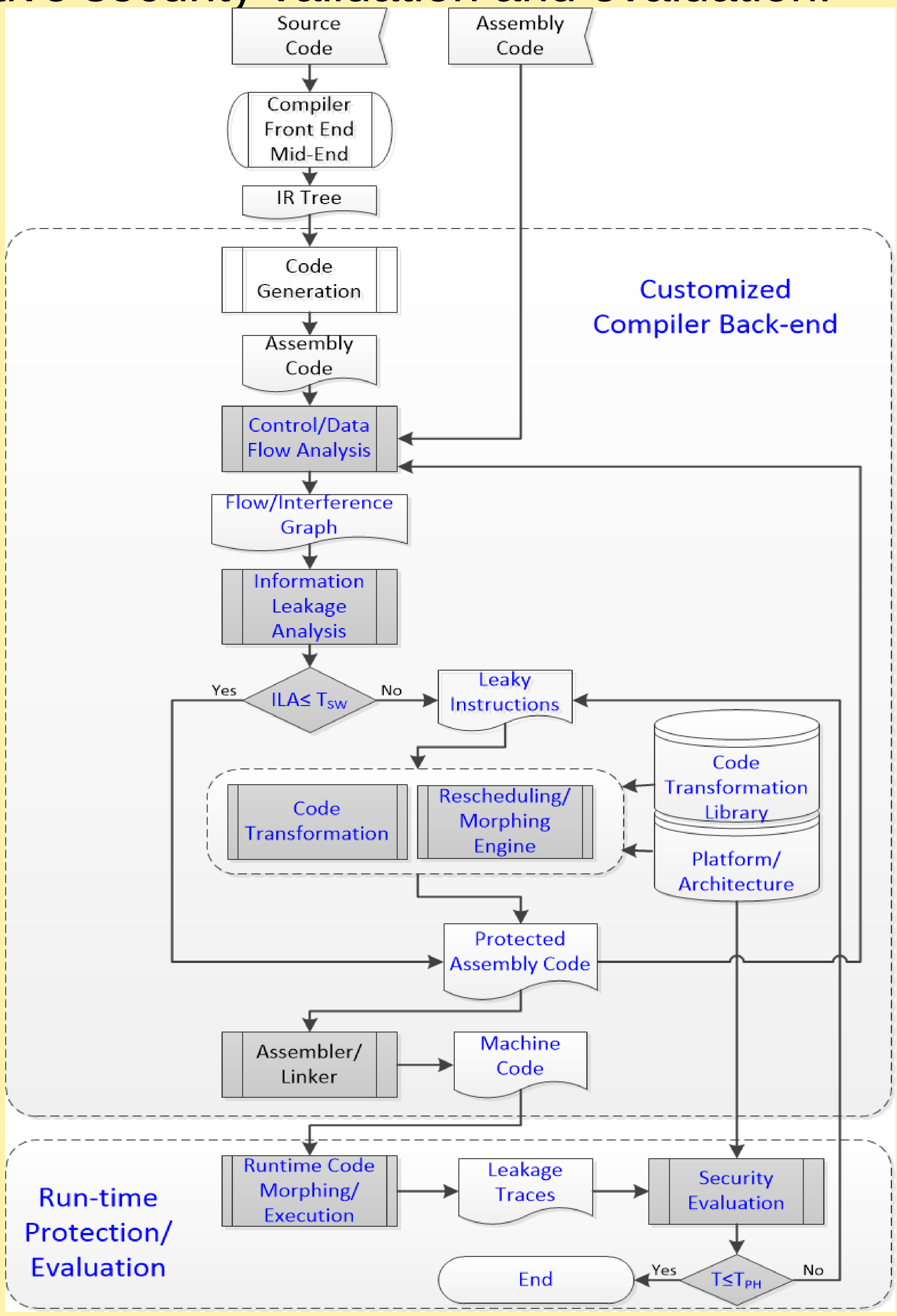
Shuffling Space Identification

Embed a Permutation Engine in Code

Transformed Code for Runtime Random Shuffling

Report: Dependency & Shuffling

Designers



Source Code

Assembly Code

Compiler Front End Mid-End

IR Tree

Customized Compiler Back-end

Code Generation

Assembly Code

Control/Data Flow Analysis

Flow/Interference Graph

Information Leakage Analysis

Yes

$ILA \le T_{SW}$

No

Leaky Instructions

Code Transformation Library

Code Transformation

Rescheduling/ Morphing Engine

Platform/ Architecture

Protected Assembly Code

Assembler/ Linker

Machine Code

Run-time Protection/ Evaluation

Runtime Code Morphing/ Execution

Leakage Traces

Security Evaluation

End

Yes

$T \le T_{PH}$

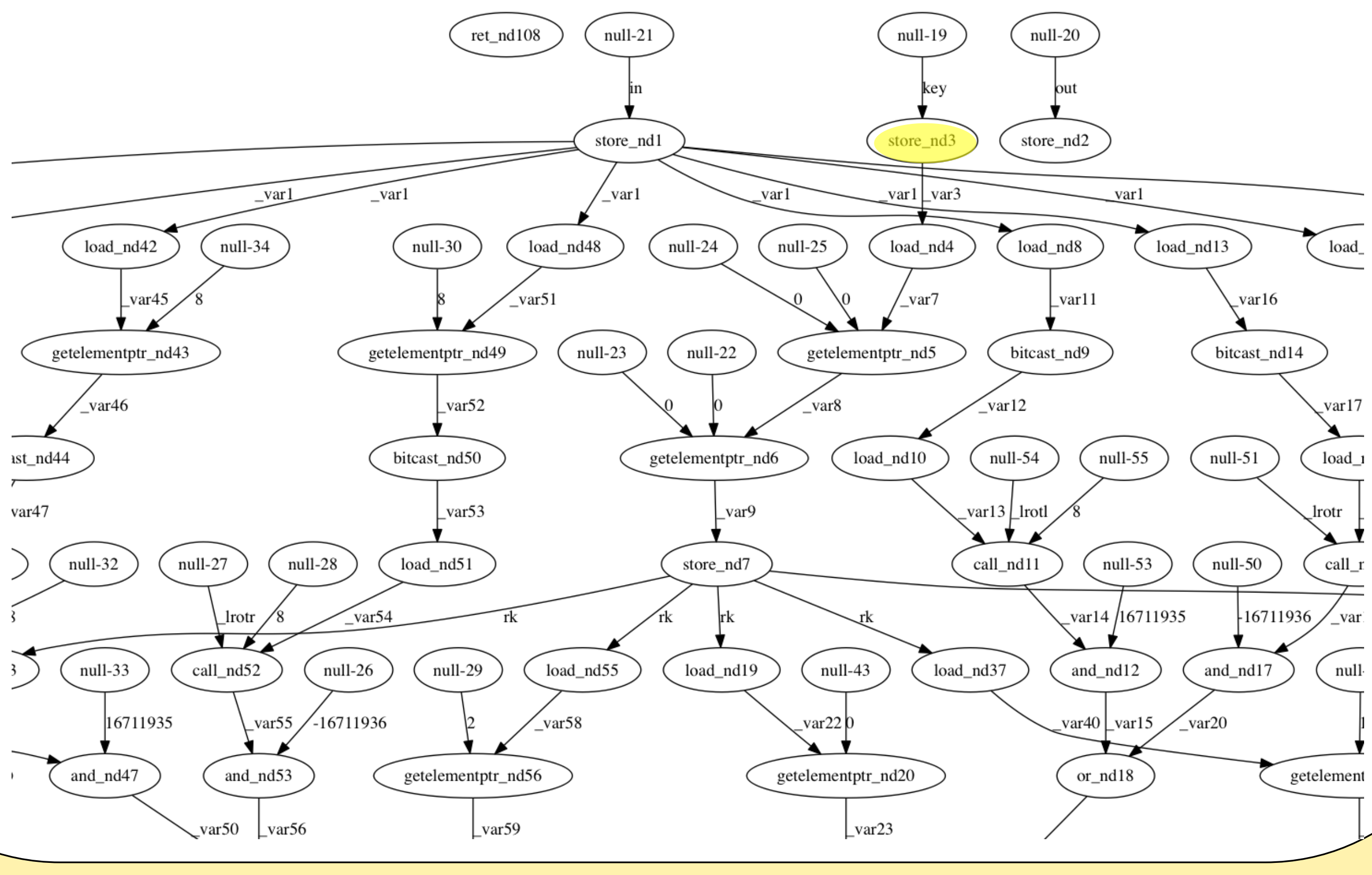
No

Approach

We will build an automation framework for information leakage analysis, multi-level countermeasure application, and formal security evaluation against software side-channel attacks.

Project Progress – 6 months

- Start with LLVM the open compilation framework
- Generate data flow graphs
- Information leakage analysis (for power leakage)



Future Work

- Perform code morphing and embed rescheduling engine in compiler back-end
- Implement run-time code morphing and randomization against side-channel attacks
- Refine metrics and models for early information leakage quantification and run-time physical leakage evaluation
- Develop information leakage analysis for cache timing attacks and higher-order attacks

Interested in meeting the PIs? Attach post-it note below!

National Science Foundation
WHERE DISCOVERIES BEGIN

The 3rd NSF Secure and Trustworthy Cyberspace Principal Investigator Meeting
January 9-11, 2017
Arlington, Virginia

Northeastern