Information Flow Tracking for OTBN Assembly

Advisor: **Roderick Bloem**

Motivation

The OpenTitan Big Number coprocessor (OTBN) accelerates cryptographic operations on highly sensitive data. To keep these operations safe against side-channel attacks, the assembly code must obey strict secure coding guidelines: how registers are cleared, which instructions may follow each other, and where transitional leakage can occur.

Whether a piece of OTBN assembly respects these rules depends on where the secret shares actually flow. Today, that flow is reconstructed by hand, which is slow and error-prone. In this project, you make it visible automatically: the simulator itself tells you which registers and memory locations carry secret data at every point of the execution.

Goals and Tasks

- Define how secret shares are annotated in OTBN assembly
- Extend the Python OTBN instruction set simulator with dynamic taint analysis
- Let the simulator print the taints of registers and memory during execution
- Evaluate the tool on OpenTitan implementations (e.g., ECDSA, RSA)

Literature

- B. Gigerl et al.
  Coco: Co-Design and Co-Verification of Masked Software Implementations on CPUs
  USENIX Security 2021
  https://www.usenix.org/conference/usenixsecurity21/presentation/gigerl
- lowRISC
  OpenTitan Big Number Accelerator (OTBN) Technical Specification
  https://opentitan.org/book/hw/ip/otbn/

Courses & Deliverables

- ☑ **Introduction to Scientific Working**
  Short report on background
  Short presentation
- ☑ **Bachelor Project**
  Project code and documentation
- ☑ **Bachelor's Thesis**
  Project code
  Thesis
  Final presentation

Recommended if you're studying

☑ CS ☑ ICE ☑ SEM

Prerequisites

- Solid Python, some assembly
- Interest in side-channel countermeasures (no prior knowledge needed)

Advisor Contact

roderick.bloem@tugraz.at