Development of the Next Generation of Side-Channel Analysis Resistant Processors

Stefan Tillich
Outline

- Motivation
- Securing processors against SCA
  - Randomized execution (NONDET)
  - HW-protected execution (secure zone)
- The secure zone concept
- Future work & synergies
Motivation

• Widespread use of powerful (32-bit) embedded microprocessors
• Problem of SCA security
• SW countermeasures weak and/or costly
• SCA-secure coprocessors inflexible
• Replace custom-built HW countermeasures with more “natural” approach
Securing Processors

- Randomized execution (NONDET)
  - Exploit unused degrees of freedom in program execution
  - E.g. Instruction scheduling, register allocation
- HW-protected execution (secure zone)
  - Actual computation in secure logic
  - “Data movement” protected by masking
Why Bother?

- Put whole processor in secure logic?!
- Problems:
  - No good solution to protect internal memory blocks (regfile, caches) readily available
  - External memory vulnerable (table lookups, register spills)
  - Speed penalties of secure logic for whole processor
Secure Zone - Concept

- Draws from three basic ideas
  - Confine “critical” operations (crypto ISE)
  - Mask “raw data movement”
  - Protect operations and masks with secure logic

- Advantages
  - Flexible
  - Good performance
  - Only a part of processor in secure logic
RISC Datapath (with ISE)

- Standard and custom crypto instructions supported
Adding a “Secure Zone”

- Secure zone
  - Similar to an FU
  - Protected by secure logic
- Rest of processor
  - Remains largely unchanged
  - Ordinary CMOS: Protected by mask
Inside the Secure Zone

Input operands (masked)

Unmask operands

Perform operation

Mask result

Operand and result addresses

Mask storage: Hold several 32-bit masks

Mask generator: Produce one fresh mask per cycle

Result (masked)
• Reflect processor pipeline structure
• Execute/commit driven by pipeline control logic
Secure Zone Details

- HW module with simple interface
- Logically distributed over several pipeline stages
- Various tradeoffs of flexibility and speed against hardware size
Design Options

• Mask generator
  • TRNG, PRNG: stream-cipher based, LFSR

• Association of masked values to masks
  • Register address, custom

• Mask storage
  • “Fully associative”, only fixed registers
Design Options (cont’d)

- Support parallel use
  - Task switching, process isolation
- Error handling
  - None, assert, trap support
Task Switching

• Instruction in secure zone context-dependent
• Masks *must never* leave secure zone
• Goal: Multiple processes can use the secure zone in parallel
• Provide mechanism to save and restore secure zone context securely
Freebie: More Masks

• Task switching support can be used to extend capacity of mask storage
• Use mechanisms for a single process
Process Isolation

• Uphold/enhance process isolation, e.g. ARM TrustZone
• Goal: No access to secure zone context of another process
• Provide mechanism for OS to clear secure zone context prior to task switch
Task Switching/Process Isolation

• Alternate mask representation
  • LFSR state + “distance” from that state
• Mask generator/storage keep track of “distance”
• OS can
  • Save LFSR state and “distances” of old process
  • Clear mask generator and mask storage
  • Restore LFSR state and masks of new process
Architecture Revisited

- Mask storage can be seen as cache
  - Large set of masks in alternate form in memory
  - Cached masks readily available
  - Flush masks on “cache conflict”
  - Restore masks on “cache miss”
Error Handling

- Traps signal exceptional states
  - Mask storage full ("cache conflict")
  - “Distance” counter overflow
  - Mask missing ("cache miss")
- OS trap handling transparent to process
Building Secure Systems

- Select desired crypto algorithms/protocols
- Add required instructions to secure zone
- Derive secure processor + tool chain from processor template
- Implement crypto securely on processor
Cost Estimation

• Prototype implementation based on SPARC V8 Leon3

• Small secure zone
  • 8.4 kGates (CMOS), ~ 30 kGates (WDDL)

• Full secure zone
  • 16.9 + 5.5 kGates (CMOS), ~ 65 kGates (WDDL + CMOS)
Preliminary SCA Evaluation: Setup

- AES-128 encryption
- No secure logic (!)
- Xilinx ML410 (Virtex-4 FX)
- Unprotected ISE vs. secure zone
- DEMA attack, 250,000 traces
Preliminary SCA Evaluation: Results

Unprotected ISE

• Successful attack
• $\rho \approx 0.02$
• -> $\sim 70,000$ traces required

Secure Zone

• Attack
• Correl: hypoth
• 10 traces or all
Future Work

• ASIC measurements: POWER-TRUST chip
• Sort out required functionality to reduce cost
• Investigate drawbacks of LFSRs for mask generation
  • Potential vulnerability against higher-order attacks
Other Issues

• More generic crypto support
• Support for table lookups
  • Should not be required (cache-timing side channel)
• General issues for secure processors
  • Protection of memory from direct readout
  • Program code integrity
• Compatible with NONDET concept
  • Increase security
  • Fend off higher-order attacks
• Compatible with a range of software countermeasures
Thanks for your attention!