Root-cause Analysis of the Side Channel Leakage from ASCON Implementations

Zhenyuan Liu (zliu12@wpi.edu) and Patrick Schaumont

Worcester Polytechnic Institute, Worcester, Massachusetts

Introduction and Motivation

Others work

Side-channel leakage assessment

1. T-test, Chi2
2. Leakage and Countermeasure Testing (yes/no)

Our focus

Root-case analysis of Side-channel leakage

1. Identify Origin of Leakage
2. Test Implementation Properties
3. Leakage and Countermeasure Debugging (how, where)
4. Starting point to design the Countermeasures
Target (Device Under Test)

Target 1: a hardware coprocessor with an iterated implementation of ASCON-128 (open-source implementation)

Target 2: a software implementation of ASCON-128 on RISC-V (RV32IMC) (open-source implementation)

We are able to perform both simulation (pre-silicon) and measurement (post-silicon) side-channel analysis on the target.
Root-cause Analysis Methodology

1. Non-specific root-cause analysis: two groups of test vectors (random vs fixed key).

2. Simulation setup: chip frequency 4 MHz, 4 samples per clock cycle (hardware ASCON), and 1 sample per clock cycle (software ascon).

Tools: Synopsys, Modelsim, Cadence Joules

Architecture Correlation Analysis (ACA)

Our own tool (ACA)

1. Identify leaky time interval (LTI) (as leaky time points).

2. Rank the logic gates of the design according to the amount of contributed leakage per gate by computing a leakage impact factor (LIF) for each gate.

3. Identify where these leaky gates coming from and what instructions cause it.

A list of ranked leaky gates

Leaky gates are top-ranked ACA gates with a leakage-model correlation higher than a chosen threshold.

Gates correspond to standard cells in this paper. There are 57,671 gates in the PicoChip Design.
Root-cause Analysis on ASCON Co-processor (KeyLoad)

Gates correspond to standard cells in this paper. There are 57,671 gates in the PicoChip Design.
Root-cause Analysis on ASCON Co-processor (Initialization)

Gates correspond to standard cells in this paper. There are 57,671 gates in the PicoChip Design.
Root-cause Analysis on Software ASCON (NonceLoad + KeyLoad + 1st round of INIT)

Sequence

```
2a72: addi $a0, $sp, 88
2a6e: sw $s7, 112(sp) # store nonce[0]
2a6e: sw $s3, 116(sp) # store nonce[1]
2a6e: sw $s11, 124(sp) # store nonce[2]
2a70: sw $a1, 120(sp) # store nonce[3]
2a72 sw $s4, 96(sp) # store key[0]
2a74 sw $s6, 100(sp) # store key[1]
2a76 sw $s10, 104(sp) # store key[2]
2a78 sw $s6, 108(sp) # store key[3]
2a7a: jal ra,$fe4 <P12>
```

PC Counter

```
fe6  lw $s1, 8($a0)  # load key[0]
ff0  lw $ra, 12($a0)  # load key[1]
...
1004 lw $a5, 16($a0)  # load key[2]
1006 lw $t4, 20($a0)  # load key[3]
...
1026 xori $a4, $a5, 240  # round constant: state_reg[2] XOR with 0x0F
1036 not $t7, $t4  # not key[3]
1038 xori $a5, $a5, -241  # XOR1 key[2]
...
1246: jal $ra, 19c
```

Gates correspond to standard cells in this paper. There are 57,671 gates in the PicoChip Design.
Root-cause Analysis on Software ASCON (12\textsuperscript{th} round of INIT + KeyXor)

There are 57,671 gates in the PicoChip Design.
Validating Simulation against Measurement (Software ASCON)

Measurement Setup:


2. Chip frequency 4 MHz, 4 samples per clock cycle.
Thank you!
Please ask me questions for bonus points :)