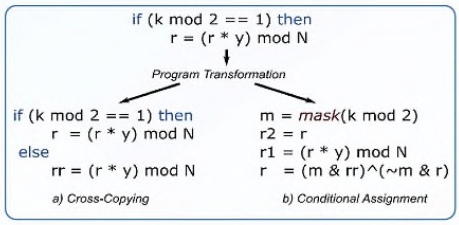
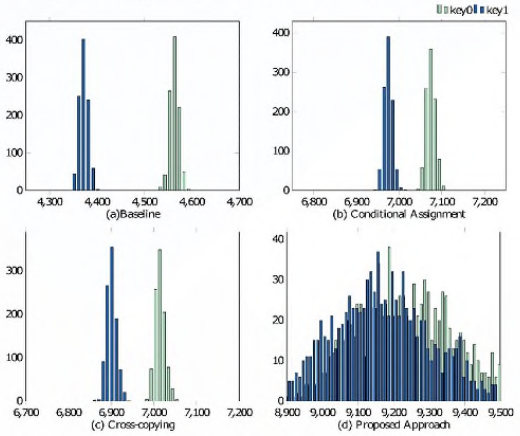
Custom Instructions for Dynamic Program Diversification to Mitigate Timing Leakage

Motivation

Embedded systems have become an integral part of our lives, and hence it is essential that they are secure. Timing side channel attacks pose a major threat to embedded systems due to their ease of accessibility. Existing countermeasures that rely on program transformation cannot guarantee timing leakage reduction after compilation.

```
//Input: data y, private key k which has d bits, integer N
//Output: encrypted data r; r = yk mod N
1: Procedure Mod_Exp(y, k, N)
2:     r = 1
3:     for i = 0 to d do
4:         if (k mod 2 == 1) then        } Security Critical
5:             r = (r * y) mod N         } Segment
6:         y = (y * y) mod N
7:         k = k >> 1
8:     return r mod N
```



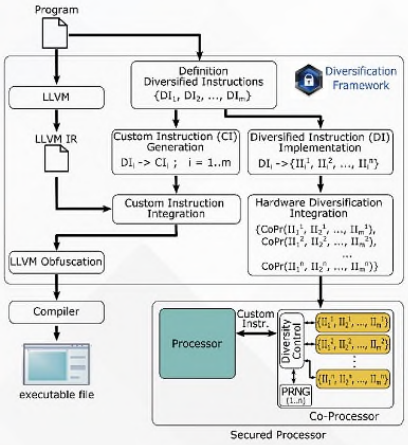


We propose a framework that relies on dynamic program diversification to protect against such timing attacks. The proposed framework is built upon the widely-used LLVM compiler infrastructure and generates custom instructions on the RISC-V soft-processor to minimize the timing leakage of the program.

Description

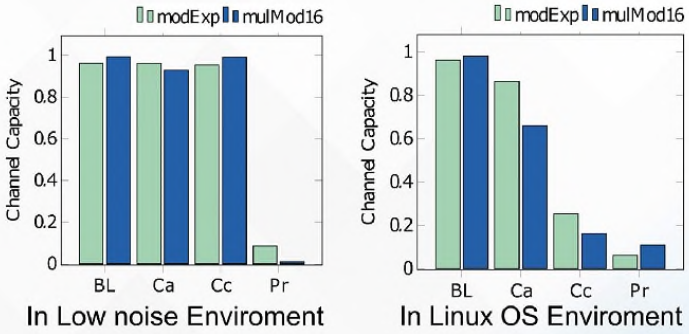
The LLVM compiler is used to automatically generate custom instructions for the security critical segments. At runtime, these instructions are executed on a custom co-processor to produce random (diversified) timing characteristics on each execution instance.

The RISC-V based platform is implemented on the Xilinx Zynq-7000 FPGA device.



Experiment Results

Our solution achieves 91% to 99% timing channel capacity reduction compared to existing solutions. This is achieved with moderate performance reduction and acceptable hardware area cost.



	RocketTile	CPU	FPU	RoCC
Slices	6603	1446	3545	73
DSPs	34	4	20	10
Area (mm^2)	3.77	0.08	0.17	0.03
Power (mW)	70.96	1.32	2.14	0.39

Pham Hung Thinh, Alexander Fell, Nandeesha Veeranna, Lam Siew Kei

Hardware and Embedded Systems Lab, SCSE, NTU

siewkei_lam@pmail.ntu.edu.sg