Microarchitectures Undo Software Security Measures

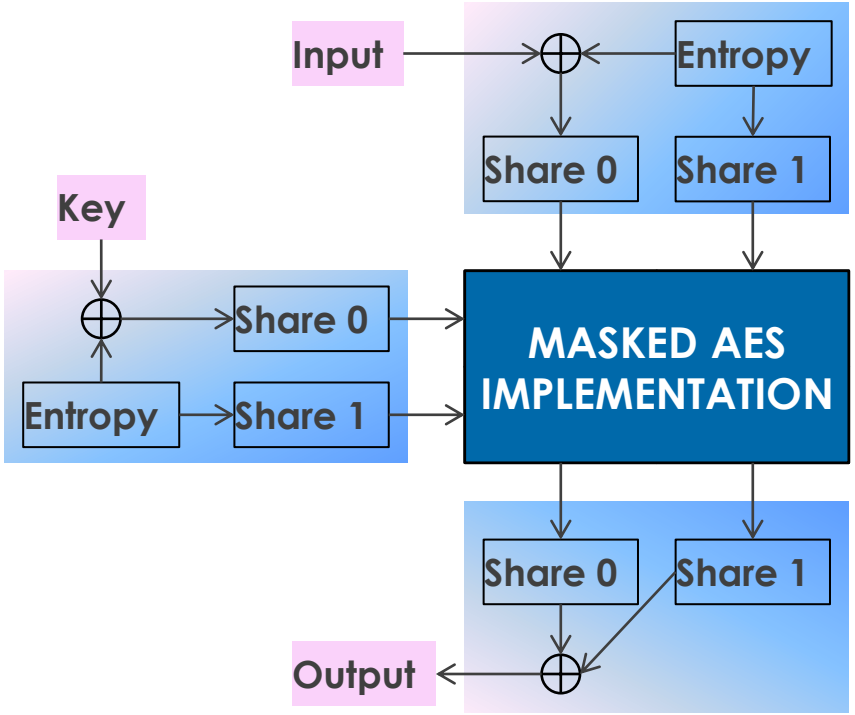
- To implement secure algorithms, software based cryptography utilizes the ISA through instructions or cryptographic extensions.
- Security measures include masking to hide sensitive data.
- Microarchitecture's sophisticated efficiency logic can neutralize intended masking.
- Due to unknown microarchitecture implementations, ensuring effective masking reduces to a game of trial and error, guess work and luck.

Example of a Security Measure: Boolean Masking

- Popular form of masking due to its efficiency.
- Secret key and inputs are masked and split into shares.

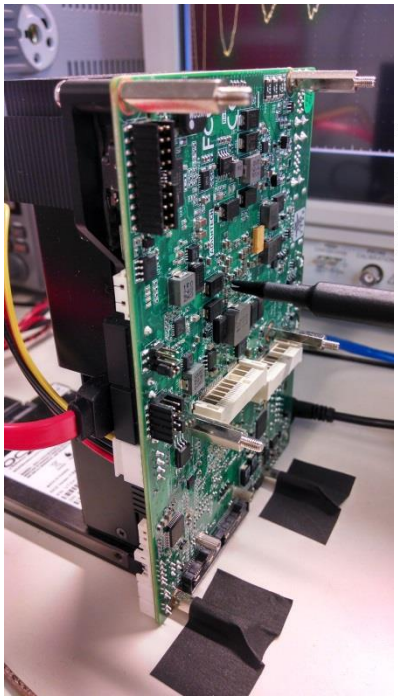
$$K_{i_0} = K \oplus R_{2i}, \qquad K_{i_1} = R_{2i}$$
$$D_{i_0} = D_i \oplus R_{2i+1}, \qquad D_{i_1} = R_{2i+1}$$

  - K is the secret key, D_i is the i^{th} input, and R_i are uniformly distributed random numbers.
  - K_{i_0} and K_{i_1} are key shares.
  - D_{i_0} and D_{i_1} are input, or state shares.
- Each share, alone, **does not** represent sensitive information.
- The XOR, or Hamming Distance, of shares **does** represent sensitive information and must be avoided.



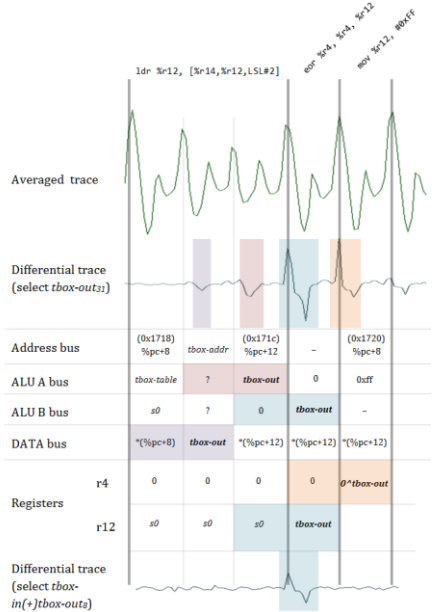
Examples of Microarchitecture Leaks

Intel AES-NI



- Cache reads reveal relationship between inputs and secret keys!
- State transitions going into or output of AES reveal secret keys!

ARM



- Sequential operands through ALU leak Hamming distance.
- Accessed value in register bank leaks Hamming distance with other values in register bank.
- Sequential bus addresses leak Hamming distance.
- Data used in LDRB instruction leak.
- Combinations of seemingly unrelated data and instructions leak.

Call for Collaboration

- Side Channel Analysis is a real problem, even for large and complicated microprocessors.
  - Large and complicated microprocessors make it harder to implement software based countermeasures against DPA.
    - Especially when microarchitecture is not known.
- Software based countermeasures for cryptographic ISA extensions can only reduce, not remove leaks.
- Effective software based masking incurs large efficiency hits due to extra work to avoid implicit unmasking in microarchitecture.
- Suggest possible additions, or modifications to microarchitecture to allow for a DPA based trusted execution environment.
- Looking to the microarchitecture community for advice, suggestions and ideas.