Computing devices play an increasing role in many aspects of life, including commerce, communication and transport. Traditional concerns such as efficiency, physical size and power consumption remain key design influencers, but security is often paramount; the devices themselves play an important role in preventing passive side-channel and active fault injection attacks on the cryptography they execute.

**Motivation**

To prevent attacks, or at least make them harder, one approach is to alter the underlying processor design. This is attractive as a generic solution: typically, all programs that run on the processor are afforded some level of protection.

1. The **NONDET** processor randomises some aspects of execution. For example, it “shuffles” the order in which instructions are executed: the following examples compute the same result, but the ordering of the instructions differs.

   ```
   l d [ %r0 + 0 ] , %r1
   l d [ %r0 + 4 ] , %r2
   l d [ %r0 + 8 ] , %r3
   x o r %r1 , %r2 , %r1
   x o r %r1 , %r3 , %r1
   ⇝
   l d [ %r0 + 8 ] , %r3
   l d [ %r0 + 0 ] , %r1
   x o r %r1 , %r3 , %r1
   l d [ %r0 + 4 ] , %r2
   ```

   This makes targeting leakage from the red instruction harder because it changes position each time the program is executed.

2. The **Power-Trust** processor takes a different approach, executing some instructions in a protected region; since this “secure zone” is implemented using a secure logic style, less leakage from instructions occurs than with standard CMOS.

---

**Funding, Collaboration and Impact**

- EPSRC via grants GR/R13654/1, EP/F039638/1, EP/H001689/1 and EP/E001556/1.
- European Commission via ECRYPT II.
- **Academic Collaborators:**
  - Graz University of Technology.
- **Industrial Collaborators:**
  - Infineon Technologies.

---

**Micro-architectural Side-Channels**

The concept of “sandboxing”—isolating processes to provide a secure execution environment—is attractive in software, but difficult to realise in hardware without impacting on performance. Consequently, processes may leak information through shared micro-architectural resources. The trend towards virtualised, cloud computing platforms makes this very problematic. Our work has produced results in two main areas:

1. The latency of early terminating integer multiplication on **ARM7TDMI** processors depends on the multiplier operand; this can be exploited to attack primitives such as AES and RSA.

2. The data-dependent behaviour of cache memory has been viewed as a potential covert- and side-channel for several decades, with recent advances yielding cryptographic attacks, including our own attacks on the DSA implementation in OpenSSL when executed on Intel Pentium 4 and Atom processors.