Template Attacks on ECDSA on a 32-bit ARM

**Theory**

**ECDSA Signature Generation**
1. $e = \text{HASH}(m)$
2. Random integer $k$ from $[1, n - 1]$.
3. $r = x \pmod{n}$, where $(x, y) = kG$.
4. Calculate $s = k^{-1}(e + rd_A) \pmod{n}$.
5. The signature is the pair $(r, s)$.

**Template Based SPA Attack**
- Superior over standard SPA, succeeds even for SPA resistant algorithms
- Describes power consumption with a probability distribution
- Allows classification of all kinds of instructions, operations or features
- Single shot attack

**Attack on ECDSA**
- ECDSA is well suited for template attacks
- Basepoint $G$ is known and intermediate curve points can be precomputed and classified for a guessed key $k$
- Successful attacks on the ephemeral key $k$ reveal the secret key $d_A$

**ECC Implementation**
- Optimized for 32-bit ARM7 platform, which is used in many modern embedded systems
- NIST curve P192
- Binary and windowed double and always add method (both attacked successfully)
- Standard and constant runtime GF(p) arithmetic (both attacked successfully)

**Practical Attack**

**Used Leakage**
- Standard GF(p) implementations leak dramatically due to key-dependent reductions
- Constant runtime GF(p) arithmetic has DPA leakage

**Templates for Several Curve Points**
- Allows optimal template point selectoin
- Best DPA point per intermediate
- Use only intermediates with large Hamming distance
- Leads to a success rate beyond 99.99%

---

All implementations have been attacked on this ARM7 based LPC2124 32-bit microcontroller

For the P192 curve a single group operation provides over 100 intermediate values, larger curves even more