**LadderLeak**: Breaking ECDSA with Less than One Bit of Nonsense Leakage

Diego F. Aranha¹, Felipe R. Novais², Akira Takahashi³, Mehdi Tibouchi³, Yuval Yarom⁴

¹DIGIT, Aarhus University, Denmark  ²University of Campinas, Brazil  ³NTT Corporation, Japan  ⁴University of Adelaide and Data61, Australia

---

**Risk of randomness failure in ECDSA-type signatures**

**Algorithm 1: Montgomery ladder**

**Conditions for the attack to work:**
1. Group order is \(2^k\) – 1 with small \(k\).
2. Accumulators \((R_k, A_k)\) are in projective coordinates, but initialized with the base point in affine coordinates.
3. Group bias is non-constant time set handling 2 coordinates \(\leftrightarrow \) Weierstrass model

**Challenges**
- Can we reduce the data complexity of Fourier analysis-based attack?
- Can we attack even less than 1-bit of nonsense leakage (i.e., top-most bit of nonce \(k\) is only leaked with prob. < 1/2)?
- Can we obtain such a leakage from practical ECDSA implementations?

**Our contributions**
- Novel class of cache attacks against the Montgomery ladder scalar multiplication in OpenSSL 1.0.2a and 1.1.0, and RELIC 0.4.0.
- Affected curves: NIST P-192, P-224, P-256 (not by default in OpenSSL). P-384, P-521, B-283, K-283, K-409, B-571, sect163r1, mprime2, meg Montgomery.
- Improved theoretical analysis of the Fourier analysis-based attack on HNP (originally established by Bleichenbacher).
- Significantly reduced the required input data.
- Analysis in the presence of erroneous leakage information.
- Implemented a full secret key recovery attack against OpenSSL ECDSA instantiated over sect163r1 and NIST P-192.

**Comparison with previous HNP records**

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>256-bit</td>
<td>(1)FA38</td>
<td>(1)FA38</td>
<td>(1)FA38</td>
<td>(1)FA38</td>
</tr>
<tr>
<td>192-bit</td>
<td>113h 492GB</td>
<td>113h 492GB</td>
<td>113h 492GB</td>
<td>113h 492GB</td>
</tr>
<tr>
<td>160-bit</td>
<td>68h 10GB</td>
<td>68h 10GB</td>
<td>68h 10GB</td>
<td>68h 10GB</td>
</tr>
</tbody>
</table>

---

**LadderLeak:** Tiny timing leakage from the Montgomery ladder

**Cache-timing attack experiments**

**Experiments** were carried out with Fiat-Hillel cache attack technique \(\rightarrow\) MSE of \(k\) was detected with \(>99.9\%\) accuracy.

**How to quantify the nonce bias**

**Bias function**

The sampled bias of a set of points \(K = \{k_i\}_{i=1}^{\infty}\) in \(\mathbb{Z}_q\) is defined by

\[
\text{Bias}_q(K) = \prod_{i=1}^{\infty} k_i^{(q-2)/3}
\]

Uniform \(k_i \in \mathbb{Z}_q\)

Biased \(k_i \in (0,q/2)\)

---

**Bleichenbacher’s Fourier analysis-based attack**

- **Step 1.** Quantify the modular bias of randomness \(K\) by defining a bias function \(\text{Bias}_q(K)\).
- **Step 2.** Find a candidate secret key which leads to the peak of \(\text{Bias}_q(K)\) (by computing FFT).

**Critical intermediate step:** collision search of integers \(k\).
- Detect the bias peak correctly and efficiently.
- Improvement: 2 Established unified time-memory-data tradeoffs by applying \(K\)-list sum algorithm for the GMP!

**Experimental results on full key recovery**

**Tradeoff graphs for 1-bit bias**

**Main takeaways**
- Securely implementing critical cryptographic algorithms is still hard.
- Don’t underestimate even less than 1-bit of nonsense leakage!
- Interesting connection between the HNP and GMP (from symmetric key crypto)
- Future work:
  - More list sum algorithms and tradeoffs?
  - Improvements to FFT computations?
  - Other sources of small leakage?

More details at [https://te.s.eu/2020/635](https://te.s.eu/2020/635)