Practical Attack on Bilinear Pairings to Disclose the Secrets of Embedded Devices

Thomas Unterluggauer and Erich Wenger
Graz University of Technology
Institute for Applied Information Processing and Communications
Inffeldgasse 16a, 8010 Graz, Austria
Email: {Thomas.Unterluggauer, Erich.Wenger}@iaik.tugraz.at

***Abstract*—Identity-based encryption constitutes a promising alternative to traditional cryptography that works without symmetric keys or public key infrastructures. Such schemes generally depend on the computation of bilinear pairings. The latest developments in efficient pairing algorithms made identity-based encryption available to embedded devices as well. However, those devices are inherently exposed to side-channel attacks. In this paper, we present a correlation power analysis attack to extract the private key in the popular identity-based encryption scheme by Boneh and Boyen. On an ARM Cortex-M0 we exploit the leakage of a finite field multiplication within the highly practical optimal-Ate pairing defined over the elliptic curves by Barreto and Naehrig. As a secondary contribution, we practically verified the feasibility of our attack on an FPGA, an ASIC, and using power simulations. For future work our research intends to raise awareness of the importance of the randomization countermeasure in pairing computations.**

Keywords*—*Optimal-Ate Pairing; BN Curves; Side-Channel Attack; CPA; FPGA; ASIC; Power Simulation;

I. Introduction

In the last decade, much attention was drawn to the idea of identity-based encryption. The concept, which was proposed by Shamir [21] in 1984, allows the secure transmission of confidential data by just using the recipient's identity string as the key. As a huge benefit compared to traditional cryptography, identity-based encryption avoids both public key infrastructures and the distribution of symmetric keys. One thing most identity-based encryption schemes have in common is the computation of bilinear pairings.

A bilinear pairing is a cryptographic primitive that may be built upon elliptic-curve cryptography. BN curves by Barreto and Naehrig [4], best suitable for the 128-bit security level, are one of the most promising proposals for secure bilinear pairings. BN curves allow such a level of performance that they are even suitable for embedded devices (cf. Unterluggauer and Wenger [22]). In these embedded environments we expect identity-based encryption to play an important role in providing secure applications in the future. However, side-channel leakage constitutes an omnipresent threat to the security of such devices. Susceptibility of pairing implementations to side-channel attacks were investigated early with respect to such in small characteristics by Kim et al. [15] and Page and Vercauteren [18]. However, the practical examination of a full implementation of an identity-based encryption scheme based on the optimal-Ate pairings over BN curves with large characteristic still remains.

In this paper we present the results of a practical Correlation Power Analysis (CPA) attack that leaks a user's private key in the identity-based encryption scheme by Boneh and Boyen [7]. In contrast to the work in [11], which recovers the user's private key from an exemplary 8-bit hardware circuit that only performs the operations that leak the sensitive information, we successfully attack a full implementation on a 32-bit architecture. Therefore, the secret input point of the highly practical optimal-Ate pairing defined over BN curves is revealed as opposed to [15, 18], who focused on pairings over fields of small characteristics. For this purpose, this work exploits the leakage of a finite field multiplication [14] within the pairing computation. On the contrary, the attack on Tate pairings over BN curves in [12] exploits the leakage of a finite field addition. Besides the CPA attack, we provide future work with evidence on how power analysis attacks perform relatively to each other on an FPGA, an ASIC, and using power simulations. Moreover, we emphasize that the projective point randomization technique [9] is a countermeasure that is applied to Ate pairings on BN curves almost without effort.

The paper is structured as follows. In Section II, we investigate related work and further highlight how our work complements the related work. Besides the background of identity-based encryption and pairings, a high-level view of the attack setting is given in Section III. A general description of the attack is part of Section IV. Section V discusses the practical results of the attack. Following possible countermeasures in Section VI, a conclusion is drawn in Section VII.

II. Related Work

The first to investigate side-channel attacks in the context of pairing computations were Page and Vercauteren [18]. They focused on pairings over ternary fields, pointed out the possibility of timing and Simple Power Analysis (SPA) attacks of improperly implemented finite field multiplications, and proposed a Differential Power Analysis (DPA) attack that sequentially extracts one bit after another using the technique by Messerges [16]. Similarly, Kim et al. [15] showed each a timing, an SPA and a DPA attack that potentially extract a secret value involved in the computation of the Eta pairing over hyperelliptic curves using binary fields. This paper in contrast focuses on a Correlation Power Analysis (CPA) attack on optimal-Ate pairings using large prime fields, whose arithmetic

differs enormously to that in binary or ternary fields used in, e.g., [18].

Whelan and Scott [25] investigated the side-channel vulnerability of the Tate, the Ate, and the Eta pairing more generally. They concluded that the computation of a bilinear pairing $e(P, Q)$ of the two elliptic curve points P and Q is inherently more secure if its first parameter P is the secret as it seemed impossible to build the hypothesis for a DPA attack. However, for the Tate pairing not using elliptic curve twists, Blömer et al. [6] concluded theoretically that schemes using bilinear pairings with its first argument P being secret are not less vulnerable to side-channel attacks than otherwise. We complement their work by presenting results of a practical attack on the secret first argument of an Ate pairing computation over BN curves that uses elliptic curve twists.

An attack similar to the one presented in this work was done by Ghosh and Roychowdhury [12]. In their attack, the secret parameter Q of the Tate pairing $e(P, Q)$ over BN curves was revealed. In more detail, a finite field addition during the evaluation of the line function in the Miller loop was targeted. The operation involves the secret input Q as well as the x-coordinate of the intermediate elliptic curve point that derives from the public input point P. Starting from the Least Significant Bit (LSB), they recover the secret x-coordinate successively by performing a difference-of-means for each bit. They gather the necessary power measurements from their own FPGA-based pairing cryptoprocessor. Contrary to attacking a finite field addition within the pairing computation, we exploit the leakage of a finite field multiplication. Thereby we utilize the technique of Hutter et al. [14], who efficiently attack a multi-precision integer multiplication within ECDSA-enabled RFID devices.

Private keys in identity-based encryption were shown to be vulnerable to side-channel attacks in [11]. In a DPA attack, they demonstrated the feasibility of extracting the secret input of a prime-field based pairing computation from a hardware circuit which has an 8-bit datapath and which merely performs the operations leaking the secret information. In contrast, we extract the private key from a full and practical implementation of identity-based encryption on a 32-bit architecture, which is significantly harder to be performed successfully due to the exponentially larger number of possible values for each word of the secret. Besides, our results are based on three different measurement setups, while [11] use power simulations only.

Several countermeasures to inhibit attacks on pairing computations were shown in the past. Page and Vercauteren [18] proposed two variants of point blinding mechanisms to counteract DPA attacks. In addition to that, Whelan and Scott [25] proposed multiplying the Miller variable in each iteration with a different random value. Unluckily, all of the mentioned countermeasures offer rather bad performance. Point blinding requires the computation of a second pairing at least, while the multiplication of the Miller variable involves an additional finite field multiplication in each iteration of the Miller loop. However, Kim et al. [15] adopted the fast and effective randomization countermeasure by Coron [9] to the Eta pairing. They provided modified formulas to deal with the randomized projective coordinates of one of the two input points. In this paper we intend to raise awareness of the randomization countermeasure in the context of optimal-Ate pairings over BN curves. In this case, it is not even necessary to modify the formulas to deal with the randomized coordinates.

III. Background

A. Identity-based Encryption

In 1984, Shamir [21] proposed the concept of identity-based encryption for secure communication in company networks and mailing systems without the necessity of public key infrastructures. The concept uses identity strings instead of public keys for encryption, e.g., someone's e-mail address in a mailing system, which inherently allows sending encrypted e-mails. In order to achieve that, a trusted third party is responsible for providing public parameters and for generating the users' private keys.

One fast identity-based encryption scheme that is already in practical use is the BB_1 scheme that was presented by Boneh and Boyen [7]. Besides, they proposed a very practical BB_1-based Key Encapsulation Mechanism (KEM) for the future IEEE standard on identity-based encryption. The KEM variant of the scheme specifies the four algorithms *Setup*, *Derive*, *Encapsulate* and *Decapsulate*. The *Setup* algorithm is run at the trusted third party and creates a master secret and the public parameters. Also the *Derive* algorithm is run at the trusted third party in order to generate each user's private key. The two algorithms *Encapsulate* and *Decapsulate* are run by the respective users, who may use embedded devices. The *Encapsulate* algorithm provides both a session key and a ciphertext that is decryptable by the intended recipient only. The recipient recovers the session key from the received ciphertext by invoking the *Decapsulate* algorithm with their private key as a parameter.

The scheme uses three cyclic order-n groups $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ that allow the definition of a bilinear pairing $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$. The produced ciphertext $C = (C_0, C_1)$ consists of two elements in $\mathbb{G}_1$ and the respective private keys $D_{id} = (D_{0,id}, D_{1,id})$ are comprised of two elements in $\mathbb{G}_2$. The scheme's *Decapsulate* algorithm recovers the session key K from a ciphertext C with the aid of the user's private key D_{id}, the properties of the bilinear pairing e, and a hash function H:

$$K = H(e(C_0, D_{0,id})/e(C_1, D_{1,id})).$$

In this algorithm, the session key is obtained from bilinear pairing computations involving both a public and a secret operand. The secret operand to the bilinear pairing—the user's private key in this particular case—is the target of adversaries.

Note that we focus our analysis on the BB_1 scheme, but the subsequent attack is applicable to all schemes that involve pairing computations with a secret and a public operand.

B. Bilinear Pairings

The aforementioned BB_1 scheme requires the computation of bilinear pairings. A bilinear pairing $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$ accepts an element of the two additive groups $\mathbb{G}_1$, $\mathbb{G}_2$, respectively, maps them to the multiplicative group $\mathbb{G}_T$, and hereby fulfills several properties:

1) Bilinearity: $e(aP, bQ) = e(P, Q)^{ab}$ $\forall P \in \mathbb{G}_1$, $Q \in \mathbb{G}_2, a, b \in \mathbb{Z}$.

Algorithm 1 Ate pairing over BN curves.

Input: $P \in E(\mathbb{F}_p), Q \in E'(\mathbb{F}_{p^2})$
Output: $a(Q, P)$
1: $T \leftarrow Q, f \leftarrow 1$
2: **for** $i = \lfloor ld(s) \rfloor - 2$ downto 0 **do**
3: $f \leftarrow f^2 \cdot \ell_{T,T}(P)$ ▷ subject to our attack
4: $T \leftarrow [2]T$
5: **if** $s_i = 1$ **then**
6: $f \leftarrow f^2 \cdot \ell_{T,Q}(P)$
7: $T \leftarrow T + Q$
8: **end if**
9: **end for**
10: $f \leftarrow f^{(p^{12}-1)/n}$
11: **return** f

2) Non-degeneracy: $\forall P \in \mathbb{G}_1 \setminus \{\mathcal{O}\}\ \exists\, Q \in \mathbb{G}_2 : e(P, Q) \neq 1$.
3) Computable: $e(P, Q)$ can be computed efficiently.

The groups $\mathbb{G}_1$, $\mathbb{G}_2$ are typically groups over elliptic curves and $\mathbb{G}_T$ is the subgroup of a large extension field. However, only certain elliptic curves allow the definition of $\mathbb{G}_1$, $\mathbb{G}_2$, $\mathbb{G}_T$ with an admissible bilinear pairing. In this paper, we use the pairing-friendly elliptic curves by Barreto and Naehrig [4] of the form $E : y^2 = x^3 + b$ with $b \neq 0$. Ate pairings $a(Q, P)$ based on these curves are defined as

$$a\colon\ \mathbb{G}_2 \times \mathbb{G}_1 \rightarrow \mathbb{G}_T\ :\ E(\mathbb{F}_{p^{12}}) \times E(\mathbb{F}_p) \rightarrow \mathbb{F}_{p^{12}}. \quad (1)$$

Since there exists an efficiently computable group homomorphism that exploits the curve's sextic twist E', elements in $\mathbb{G}_2$ can be compressed, which leads to the more efficient definition of the Ate pairing

$$a\colon\ \mathbb{G}_2 \times \mathbb{G}_1 \rightarrow \mathbb{G}_T\ :\ E'(\mathbb{F}_{p^2}) \times E(\mathbb{F}_p) \rightarrow \mathbb{F}_{p^{12}}. \quad (2)$$

Optimal-Ate pairings by Vercauteren [23] constitute particularly fast variants of the Ate pairing and were used for the practical evaluation. However, the succeeding elaborations are valid for Ate pairings in general.

C. Vulnerability

The Ate pairings in the identity-based encryption scheme are computed according to Algorithm 1. Since the attack aims to recover the pairing's secret input, a more detailed investigation of the algorithm is necessary.

The algorithm to compute the Ate pairing $a(Q, P)$ basically consists of the Miller loop in Lines 1-9 and the final exponentiation step in Line 10. The evaluation of the tangent line $\ell_{T,T}(P)$ in Line 3 and the point doubling in Line 4 of Algorithm 1 can be interleaved using the fast formulas by Costello et al. [10]. The respective sequence of operations at the beginning of the first iteration of the Miller loop is shown in Algorithm 2.

This sequence of operations is vulnerable to a side-channel attack and may be exploited to extract either of the pairing's two parameters P and Q. In the *Decapsulate* routine of the aforementioned BB_1 identity-based encryption scheme, the pairings $a(D_{0,id}, C_0)$ and $a(D_{1,id}, -C_1)$ are computed. In both cases, the input parameter Q of the pairing $a(Q, P)$ is the secret to be extracted.

Algorithm 2 Initial sequence of Ate pairing computations.

Input: $\mathrm{P} \in \mathrm{E}(\mathbb{F}_\mathrm{p}), \mathrm{Q} \in \mathrm{E}'(\mathbb{F}_{\mathrm{p}^2})$
1: $(\mathrm{X_T}, \mathrm{Y_T}, \mathrm{Z_T}) \leftarrow (\mathrm{x_Q}, \mathrm{y_Q}, 1)$
2: $\mathrm{L_{1,0}} \leftarrow \mathrm{X_T^2}$
3: $\mathrm{L_{1,0}} \leftarrow 3 \cdot \mathrm{L_{1,0}}$
4: $\mathrm{L_{1,0}} \leftarrow \mathrm{L_{1,0}} \cdot \mathrm{x_P}$
5: ...

In the following, we assume the input point Q of $a(Q, P)$ to be secret and P to be public. In Line 4 of Algorithm 2, the x-coordinate of the publicly known input $x_P \in \mathbb{F}_p$ is multiplied with the unknown intermediate value $L_{1,0} \in \mathbb{F}_{p^2}$. This finite field multiplication consists of two separate prime field multiplications of x_P with the two $\mathbb{F}_p$-elements of $L_{1,0}$. A prime field multiplication is often partitioned into a multiplication and a reduction step. The multiplication step within those two prime field multiplications allows the extraction of the two $\mathbb{F}_p$-elements of the unknown intermediate $L_{1,0}$ using a Correlation Power Analysis (CPA) attack. The original secret input Q is then easily computed from $L_{1,0}$ using Tonelli-Shanks square root computation in $\mathbb{F}_{p^2}$ and the elliptic curve equation. Accordingly, the two pairing computations $a(D_{0,id}, C_0)$ and $a(D_{1,id}, -C_1)$ in the identity-based encryption scheme allow the recovery of the two parts of the user's private key $D_{0,id}$ and $D_{1,id}$.

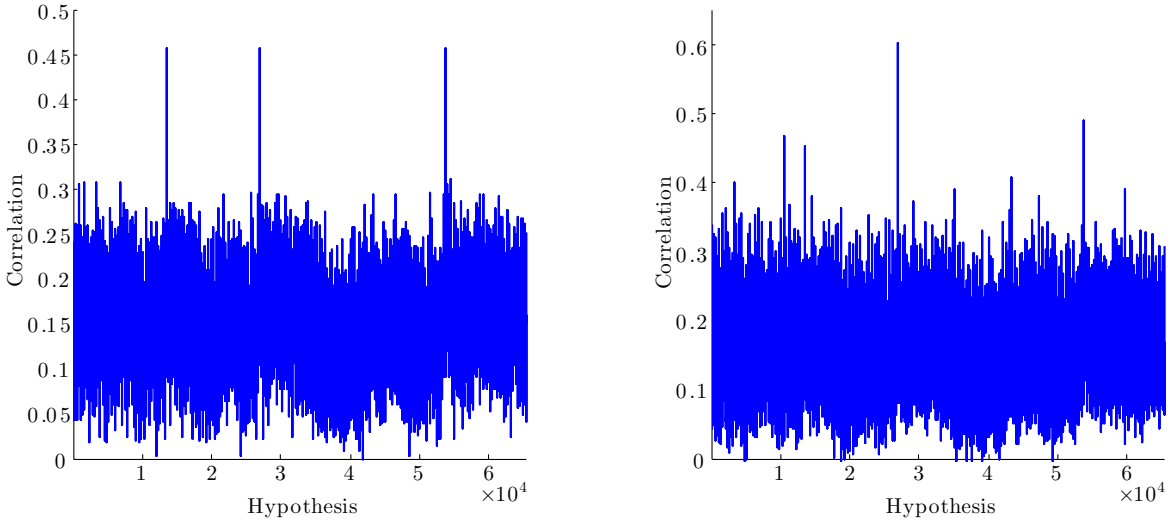
To counteract the attack, an idea may be to design protocols such that P is secret and Q is public. However, in this setup the same prime field multiplication $L_{1,0} \cdot x_P$ can be attacked to reveal the secret P since we are able to compute $L_{1,0}$ for any public input.

Other implementation formulas than the ones by Costello et al. [10] may also be vulnerable to such type of attack. In particular, the same type of attack can be performed on the revised formulas for point doubling and tangent line evaluation by Aranha et al. [1]. With a slightly modified hypothesis, the same attack is feasible on the formulas using Jacobian coordinates by Hankerson et al. [13], Beuchat et al. [5], and Aranha et al. [1]. Moreover, other protocols and schemes using pairing computations are exposed as well if these involve one both constant and secret parameter.

IV. General Attack

As indicated before, the attack to extract the secret parameter used in the optimal-Ate pairing $a_{opt}(Q, P)$ is performed on a prime-field multiplication. A prime-field multiplication on an embedded processor usually consists of a multi-precision integer multiplication of the two input operands a and b that is succeeded by a modular reduction. In order to attack the multi-precision integer multiplication, we followed the ideas presented by Hutter et al. [14].

The public operand a and the both constant and secret operand b of the multi-precision integer multiplication consist of N words of w bits, where w denotes the architecture's word size. The i-th word of a is labeled $a[i]$. A multi-precision integer multiplication basically consists of the addition of word



(a) Hamming weight model.

(b) Hamming distance model.

Fig. 1: Correlation of multiplication result for 16-bit hypotheses.

multiplication products, i.e.,

$$c = a \cdot b = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} a[i]b[j]2^{(i+j)w}.$$

Its implementation may use, for example, operand scanning or product scanning [8]. We focus on product scanning, but the attack can easily be adapted for other implementation variants as well. As secret multi-precision integers span a large space of possible values, the attack is split into two basic steps:

1) The **word multiplications** $a[i] \cdot b[j]$ are attacked to reduce the number of candidates for each of the N words of b. The k most probable candidates for each word are chosen to be used in the second step.
2) The **accumulated intermediate sums of products** resulting from the respective word multiplications are attacked using solely the remaining k candidates for each word of b.

Generally, each word $b[j] \forall\, j = 0, ..., N-1$ can be any value between 0 and $2^w - 1$. In the first step, we try to extract the most probable k candidates of the 2^w possible values for each word of b. This is done by attacking the products of each word of the secret b with the i-th word of the public input $a[i]$. All of the N words of the public input a are equally suitable for this. Depending on the details known about the implementation, a Hamming weight or a Hamming distance model may be used to construct a matrix that reflects the hypothetical power consumption of the respective multiplications. Assuming that the algorithm is executed t times, the hypothesis matrix using a Hamming weight model is computed as follows, where $a_l[i]$ denotes the i-th word of the input used in the l-th execution of the algorithm:

$$\begin{array}{c} \text{Hypothesis} \rightarrow \\ \text{Execution} \downarrow \end{array} \begin{pmatrix} \mathrm{HW}(a_0[i] \cdot 0) & \dots & \mathrm{HW}(a_0[i] \cdot (2^w - 1)) \\ \vdots & \ddots & \vdots \\ \mathrm{HW}(a_{t-1}[i] \cdot 0) & \dots & \mathrm{HW}(a_{t-1}[i] \cdot (2^w - 1)) \end{pmatrix}$$

A second matrix is built from the power traces measured for each of the t executions of the algorithm. Correlation of the hypothesis matrix with the matrix of measured power traces results in a correlation matrix that shows how each hypothesis correlates for every sample in the power traces. The correlation matrix allows the detection of the regions in the power traces where each of the multiplications $a[i] \cdot b[j] \forall\, j = 0, ..., N-1$ take place. Fig. 2d, for example, shows eight regions of high correlation that correspond to the respective multiplications $a[i] \cdot b[j]$. Evaluating each of these regions over all hypotheses makes possible the extraction of the most likely candidates for each multiplication and hence for each word $b[j]$ of the secret.

As pointed out by Hutter et al. [14], shifted variants of the correct hypothesis also lead to high correlation since multiplication is a linear operation. In the best case, each word can be identified uniquely, but in the worst case w equally likely hypotheses remain. An evaluation of all possible values for the secret input of a word multiplication is depicted in Fig. 1. In the Hamming weight model three equally likely candidates remain. Their respective values are bit-shifted versions of the correct secret-under-attack. In this case, the second part of the attack is necessary to uniquely determine the word from the remaining k candidates.

In Fig. 1b we attacked the same word using a Hamming distance model. The correct value of the secret word becomes clearly visible, but other hypotheses also yield high correlations. In this instance, the second part of the attack helps

to gain certainty about the correctness of the most likely candidate found.

Based on the k most probable candidates that were determined for each word $b[j]$ in the first part, the second step of the attack aims to uniquely determine the full secret value b. In this iterative process, one word after another is revealed by consecutively attacking the single words of the final result c. Initially, the first two words of the secret value b are determined. For this purpose all combinations of the candidates found for the first two words of b and all different inputs of a are used to create a suitable hypothesis matrix that models the second word of the result, $c[1]$. It is computed from the second partial sum and the part of the first partial sum the propagates into the second word of the result, i.e.,

$$c[1] = a[0]b[1] + a[1]b[0] + (a[0]b[0] \gg w).$$

The modeled power consumptions of the hypothetical values for $c[1]$ are then correlated with the recorded power traces. The resulting correlation matrix uniquely determines the first two words of the secret b. These revealed parts of the secret, namely $b[0]$ and $b[1]$, are then used together with the candidates for $b[2]$ to create a new hypothesis for the third word of the result, $c[2]$. In general, the hypothesis that attacks $c[l]$ to uniquely determine $b[l]$ is build as

$$c[l] = \left(\sum_{i\geq 0, j\geq 0}^{i+j=l} a[i]b[j] + \left(\sum_{m=0}^{l-1} \left(\sum_{i\geq 0, j\geq 0}^{i+j=m} a[i]b[j] \right) \gg (l-m)w \right) \right) \mod 2^w.$$

In this manner, the candidates found in the first step are used to successively determine the complete secret value b.

Note that the attack is not limited to implementations that separate the multiplication and the reduction step, but may also be applied to a Finely Integrated Product Scanning (FIPS) implementation of the Montgomery multiplication. In this case, the reduction with the public modulus needs to be considered in the hypothesis of the single words of the final result in the second step of the attack.

V. Practical Setup and Results

The attack presented in the previous sections was conducted in practice. An embedded software implementation of the BB_1-KEM identity-based encryption scheme suitable for both the ARM Cortex-M0 [2] and the Cortex-M0+ [3] was chosen as a target. The software implements optimal-Ate pairings over 254-bit BN curves and uses an assembler-optimized variant of the Separate Product Scanning (SPS) method of the Montgomery multiplication [8] for prime field multiplications. The Finely Integrated Product Scanning (FIPS) method was faster, but since the implementation incorporated the optimized multiplication in $\mathbb{F}_{p^2}$ that was presented in [5, 20], it became necessary to separate the multiplication and the reduction step in order to keep the size of the program memory low.

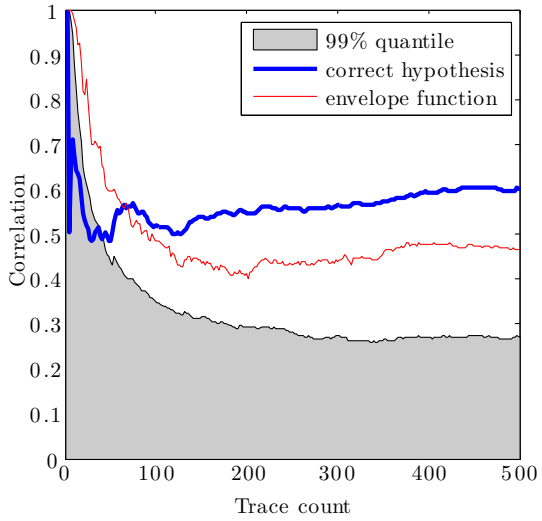
Both the ARM Cortex-M0 and the ARM Cortex-M0+ work on 32-bit operands, but merely support a $32 \times 32 \rightarrow 32$ bit multiplication that discards half of the product. Therefore, each of the N^2 word multiplications in the multiplication step of the SPS multiplication method is split into four $16 \times 16 \rightarrow 32$ bit multiplications that are aligned and accumulated appropriately. A suitable multiplication routine that simultaneously does the accumulation necessary for product scanning was presented by Wenger et al. [24] and is shown in Algorithm 3.

Algorithm 3 Multiply-Accumulate routine for Cortex-M0 and Cortex-M0+ processors.

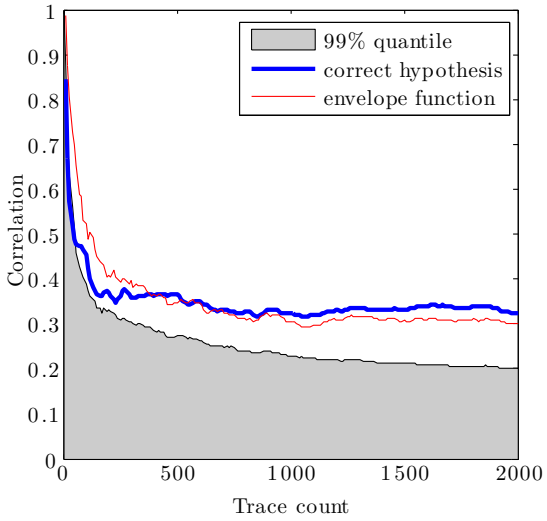
Input: `r1, r2` are 32-bit operands
Input: `r8, r9` are pointers to the operands
Output: {`r5, r4, r3`} is the accumulator

```
 1: mov  r1, r8
 2: ldr  r1, [r1, #offset1]
 3: mov  r2, r9
 4: ldr  r2, [r2, #offset2]

 5: uxth r6, r1
 6: uxth r7, r2
 7: lsr  r1, r1, #16
 8: lsr  r2, r2, #16

 9: mov  r0, r6
10: mul  r0, r0, r7  ▷ low × low
11: mul  r6, r6, r2  ▷ low × high
12: mul  r2, r2, r1  ▷ high × high
13: mul  r1, r1, r7  ▷ high × low
14: mov  r7, #0
15: add  r5, r5, r0  ▷ low × low
16: adc  r4, r4, r2  ▷ high × high
17: adc  r3, r3, r7

18: lsl  r0, r6, #16
19: lsr  r2, r6, #16
20: add  r5, r5, r0  ▷ low × high
21: adc  r4, r4, r2
22: adc  r3, r3, r7

23: lsl  r0, r1, #16
24: lsr  r2, r1, #16
25: add  r5, r5, r0  ▷ high × low
26: adc  r4, r4, r2
27: adc  r3, r3, r7
```

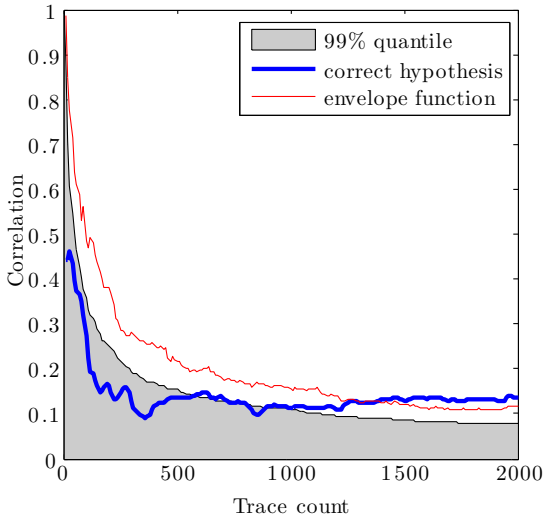
The attack described in Section IV is rather hard to perform on a 32-bit platform as each of the words of the secret operand can attain any value between 0 and $2^{32} - 1$. This leads to extremely large hypothesis matrices and requires high computational effort. Therefore, the attack was modified to better suit the targeted platform. Since each 32-bit multiplication is split into four 16-bit multiplications, the first step of the practical attack targets the 16-bit half-words of the secret operand. The respective hypothesis matrix is built from the multiplication results of the least significant half-word of the public input with all values possible for a secret half-word (2^{16} possibilities). This matrix targets the multiplications in Line 10 and 11 of Algorithm 3. The first of these multiplications reveals the lower half and the latter the upper half of each word of the secret operand. As one of the operands is overwritten by the multiplication result, a Hamming distance model is used to reflect the hypothetical power consumption of the changing



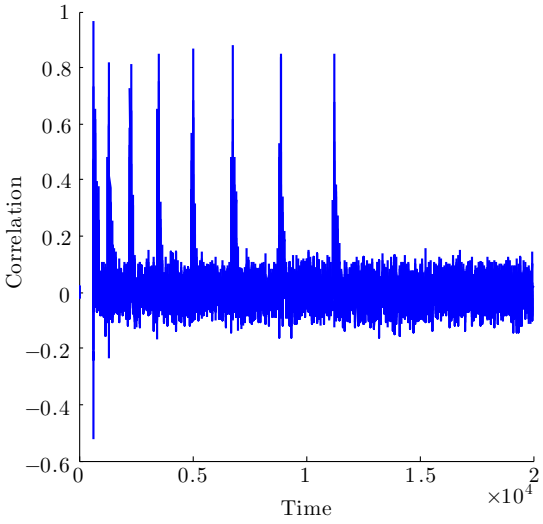
(a) Toggle counts.



(b) FPGA.



(c) ASIC.



(d) Correlation over time.

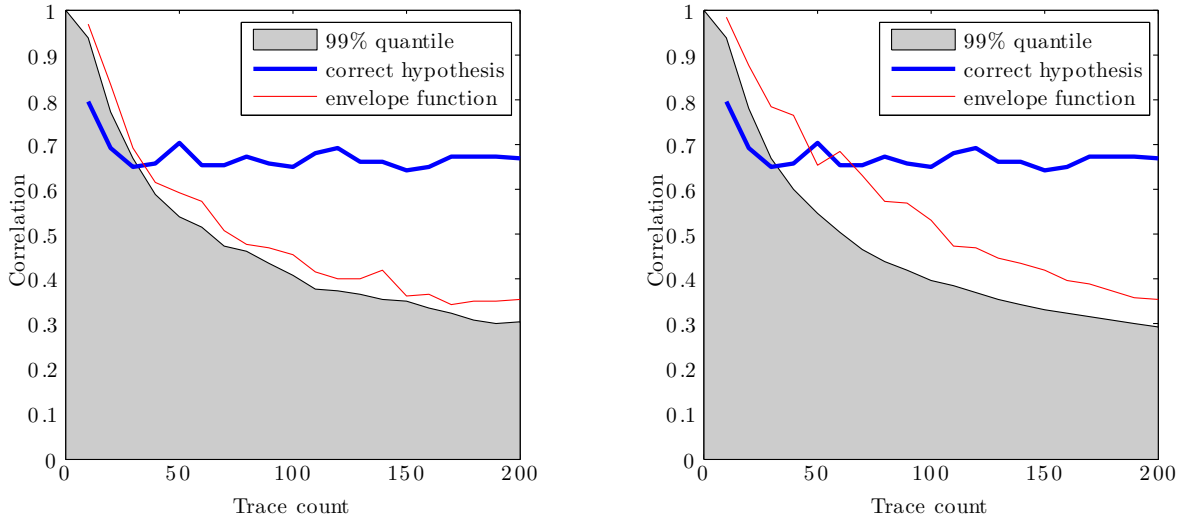
Fig. 2: Correlation of word multiplication results.

registers.

The second step of the attack was adapted accordingly. The candidates for the 16-bit half-words of the unknown operand are used to compute the hypothetical outcome for each word of the final result. The respective words are contained by the accumulator registers at various times. A simple Hamming weight model was preferred to describe the actual power consumption as the changes of the accumulator registers are rather complex to model.

Three different setups were used to collect the power traces necessary to practically perform the attack. In the first setup, a self-built processor functionally equivalent to the ARM Cortex-M0+ and its respective software implementation were deployed to the Xilinx Virtex-II Pro xc2vp30 FPGA [26] on a Sasebo G board [19]. In the second setup, the same hardware platform was synthesized for a UMC 130 nm process and power simulations were run to obtain the count of bit toggles in each clock cycle. In the third setup, the same software implementation was deployed to an ARM Cortex-M0 MCU by NXP (LPC1114FN28 [17]). For all three setups the same set of input data was used, which allows comparison of the quality of side-channel leakage. Mixing results of the Cortex-M0 and the Cortex-M0+ seems acceptable as the two processors differ only slightly. The Cortex-M0+ comes with two pipeline stages while the Cortex-M0 is in possession of three, which mainly affects branching and only marginally influences the attack.

For the power measurements on the FPGA and the ARM Cortex-M0 a MATLAB Side-Channel Analysis toolbox was utilized to communicate with the cryptographic device using



(a) k=5.

(b) k=10.

Fig. 3: Correlation of accumulation register.

its serial interface. It was further used to retrieve the power traces from the oscilloscope. The FPGA and the Cortex-M0 were operated at a clock frequency of 25 MHz and 10 MHz, respectively. To attain good measurements, both clock frequencies were chosen such that the sampling rate of the oscilloscope is an integer multiple of the device clock frequency. A trigger signal was used to align the power traces, which were measured on an 1 Ω resistor on the line from the device to VCC using a differential probe.

The effort to successfully perform the presented attack was evaluated for the three different setups. For the first part of the attack, which targets the multiplication of half-words, Fig. 2a-2c show the number of traces required to distinguish the correct hypothesis and its shifted variants from the others. Apart from the correct hypothesis' correlation, these figures show the envelope function and the 99% quantile of all hypotheses, *i.e.*, the range of correlations of all hypotheses but the highest 1%. The envelope function represents the highest correlation of any hypothesis but the correct one in each of the experiments with different trace counts.

When using the noiseless toggle counts instead of power measurements, the attack is already possible with data from less than 100 different traces. The rather old Virtex-II FPGA has quite high leakage, which results in successful attacks with merely 800 traces. When attacking the ARM Cortex-M0 by NXP that is built with modern process technologies, the attack succeeds with approximately 1,500 traces. Contrary to the other two experiments, the correct hypothesis' correlation is much lower. Further, it takes significantly more traces for the correct hypothesis to elevate from the hypotheses in the 99% quantile.

The results for the second part of the attack are similar. Fig. 3 shows the correlation of the second result word $a[0]b[1] + a[1]b[0] + (a[0]b[0] \gg w)$ depending on the number of traces when using toggle counts. The experiment was done with different numbers of candidates k learned for each half-word of the secret b in the first part of the attack. These were determined as the top k correlating hypotheses. The respective first part of the attack was conducted using 100 power traces. Since half-word candidates are found in the first part, there remain k^4 candidates to build the hypothesis matrix for the second partial sum. Using the $k = 5$ most likely candidates for each half-word resulted in a sooner success than when using the $k = 10$ most likely candidates. For higher numbers of candidates, tested with $k = 15$ and $k = 20$, no difference could be observed compared to $k = 10$. The results from Fig. 2a and Fig. 3b allow the conclusion that the complete attack succeeds with the same number of traces as required in the first part of the attack.

VI. Countermeasures

The presented CPA attack on a multi-precision integer multiplication leads to the successful extraction of the secret input point of a bilinear pairing. To mitigate such kind of attacks, several general countermeasures have been presented before, e.g., point blinding [18] and randomization of the Miller variable [25]. Point blinding techniques leave the pairing algorithm untouched and solve the problem on a higher level, i.e., instead of computing $e(P, Q)$ directly, one could either compute $e(P, Q) = e(aP, bP)^{1/ab}$ with a and b being random values or $e(P, Q) = e(P, Q + R)/e(P, R)$ with R being a random point. However, in the first case two additional point multiplications in $\mathbb{G}_1$ and $\mathbb{G}_2$ and an exponentiation in $\mathbb{G}_T$ are required, and in the second case the computation of a second pairing is necessary. Since either of those two approaches degrades performance massively, both can hardly be applied to embedded scenarios. Less expensive and hence better suitable for embedded devices is the randomization of the Miller variable as in [25]. This countermeasure requires that in each iteration of the Miller loop all intermediate variables contributing to f (cf. Algorithm 1) are multiplied

with a random value. Due to the final exponentiation, this does not affect the final result of the pairing algorithm. Still, this kind of countermeasure is not very efficient.

Therefore, we propose a more suitable method to counteract side-channel attacks on pairings over BN curves in embedded devices. Following the idea of Randomized Projective Coordinates (RPC) in [9], resistance against the presented type of attack is achieved by randomizing the intermediate point T in the computation of $a_{opt}(Q, P)$ in Algorithm 1. Instead of initializing T trivially with $(X_T, Y_T, Z_T) = (x_Q, y_Q, 1)$, one chooses a random value λ and assigns $(X_T, Y_T, Z_T) = (\lambda x_Q, \lambda y_Q, \lambda)$ to the homogeneous projective point T. Independently of which of the two input points Q and P is secret, one is not able to build a suitable hypothesis for the presented attack any more. Apart from this single initialization step, the countermeasure does not incur any overhead. Moreover, the randomization can easily be adapted to other sets of implementation formulas and different variants of projective coordinates.

VII. Conclusion

This paper featured a CPA attack on bilinear pairings that poses a significant threat to pairing-based protocols. In this respect, we pointed out how the pairing computation can leak a user's private key in the popular identity-based encryption scheme BB_1 by Boneh and Boyen [7]. We thereby illustrated that many implementation formulas of the widely used Ate pairings $a(Q, P)$ over BN curves are vulnerable to power analysis attacks. In this regard, we were able to elaborate that the presented attack is viable independently of which of the two input parameters P and Q is secret.

Contrary to previous results, the attack targeted a finite field multiplication in the computation of the practically relevant optimal-Ate pairings. The feasibility of the attack was evaluated using three different setups. For the attack to succeed, the ASIC implementation turned out to require twice as many traces as the FPGA implementation, which on the other hand required eight times more traces than when using power simulations. However, it remains an open question, whether this observation can be generalized for future side-channel evaluations. Finally, we want to emphasize that Coron's projective point randomization techniques are equally important for pairing implementations as they are for elliptic curve cryptography. Therefore it must be mandatory to utilize randomized projective coordinates in all future side-channel-secured pairing implementations.

Acknowledgements.

This work has been supported in part by the Austrian Government through the research program FIT-IT under the project number 835917 (project NewP@ss) and by the European Commission through the FP7 program under project number 610436 (project MATTHEW).

References

[1] D. F. Aranha, K. Karabina, P. Longa, C. Gebotys, and J. López. Faster Explicit Formulas for Computing Pairings over Ordinary Curves. In K. Paterson, editor, *Advances in Cryptology EUROCRYPT 2011*, volume 6632 of *Lecture Notes in Computer Science*, pages 48–68. Springer Berlin Heidelberg, 2011.

[2] ARM Ltd. Cortex-M0 Processor, Jun 2014. URL http://www.arm.com/products/processors/cortex-m/cortex-m0.php.

[3] ARM Ltd. Cortex-M0+ Processor, Jun 2014. URL http://www.arm.com/products/processors/cortex-m/cortex-m0plus.php.

[4] P. Barreto and M. Naehrig. Pairing-Friendly Elliptic Curves of Prime Order. In B. Preneel and S. Tavares, editors, *Selected Areas in Cryptography*, volume 3897 of *Lecture Notes in Computer Science*, pages 319–331. Springer Berlin Heidelberg, 2006.

[5] J.-L. Beuchat, J. González-Díaz, S. Mitsunari, E. Okamoto, F. Rodríguez-Henríquez, and T. Teruya. High-Speed Software Implementation of the Optimal Ate Pairing over Barreto-Naehrig Curves. In M. Joye, A. Miyaji, and A. Otsuka, editors, *Pairing-Based Cryptography - Pairing 2010*, volume 6487 of *Lecture Notes in Computer Science*, pages 21–39. Springer Berlin Heidelberg, 2010.

[6] J. Blömer, P. Günther, and G. Liske. Improved Side Channel Attacks on Pairing Based Cryptography. In E. Prouff, editor, *Constructive Side-Channel Analysis and Secure Design*, volume 7864 of *Lecture Notes in Computer Science*, pages 154–168. Springer Berlin Heidelberg, 2013.

[7] D. Boneh and X. Boyen. Secure Identity Based Encryption Without Random Oracles. In M. Franklin, editor, *Advances in Cryptology CRYPTO 2004*, volume 3152 of *Lecture Notes in Computer Science*, pages 443–459. Springer Berlin Heidelberg, 2004.

[8] Ç.K. Koç, T. Acar and B.S. Kaliski, Jr. Analyzing and Comparing Montgomery Multiplication Algorithms. *IEEE Micro*, 16(3):26–33, June 1996.

[9] J.-S. Coron. Resistance against Differential Power Analysis for Elliptic Curve Cryptosystems. In Ç. K. Koç and C. Paar, editors, *Cryptographic Hardware and Embedded Systems – CHES'99, First International Workshop, Worcester, MA, USA, August 12-13, 1999, Proceedings*, volume 1717 of *Lecture Notes in Computer Science*, pages 292–302. Springer, 1999.

[10] C. Costello, T. Lange, and M. Naehrig. Faster Pairing Computations on Curves with High-Degree Twists. In P. Nguyen and D. Pointcheval, editors, *Public Key Cryptography PKC 2010*, volume 6056 of *Lecture Notes in Computer Science*, pages 224–242. Springer Berlin Heidelberg, 2010.

[11] N. El Mrabet, M.-L. Flottes, and G. Di Natale. A practical Differential Power Analysis attack against the Miller algorithm. In *Research in Microelectronics and Electronics, 2009. PRIME 2009. Ph.D.*, pages 308–311, July 2009.

[12] S. Ghosh and D. Roychowdhury. Security of Prime Field Pairing Cryptoprocessor against Differential Power Attack. In M. Joye, D. Mukhopadhyay, and M. Tunstall, editors, *Security Aspects in Information Technology*, volume 7011 of *Lecture Notes in Computer Science*, pages 16–29. Springer Berlin Heidelberg, 2011.

[13] D. Hankerson, A. Menezes, and M. Scott. *Software Implementation of Pairings*, chapter 12, pages 188–206. M. Joye and G. Neven, 2008.

[14] M. Hutter, M. Medwed, D. Hein, and J. Wolkerstorfer. Attacking ECDSA-Enabled RFID Devices. In M. Abdalla, D. Pointcheval, P.-A. Fouque, and D. Vergnaud, editors, *Applied Cryptography and Network Security – ACNS 2009, 7th International Conference, Paris-Rocquencourt, France, June 2-5, 2009, Proceedings*, volume 5536 of *Lecture Notes in Computer Science*, pages 519–534. Springer, May 2009.

[15] T. Kim, T. Takagi, D.-G. Han, H. Kim, and J. Lim. Side Channel Attacks and Countermeasures on Pairing Based Cryptosystems over Binary Fields. In D. Pointcheval, Y. Mu, and K. Chen, editors, *Cryptology and Network Security*, volume 4301 of *Lecture Notes in Computer Science*, pages 168–181. Springer Berlin Heidelberg, 2006.

[16] T. S. Messerges. *Power Analysis Attacks and Countermeasures for Cryptographic Algorithms*. PhD thesis, University of Illinois, 2002.

[17] NXP Semiconductors. LPC1114FN28 MCU Product Information, Jun 2014. URL http://www.nxp.com/products/microcontrollers/cortex_m0_m0/lpc1100/LPC1114FN28.html.

[18] D. Page and F. Vercauteren. Fault and Side-Channel Attacks on Pairing Based Cryptography. Cryptology ePrint Archive (http://eprint.iacr.org/), Report 2004/283, 2004.

[19] RISEC, AIST. Side-Channel Attack Standard Evaluation Board, Jun 2014. URL http://www.risec.aist.go.jp/project/sasebo/.

[20] A. H. Sánchez and F. Rodríguez-Henríquez. NEON Implementation of an Attribute-Based Encryption Scheme. In M. Jacobson, M. Locasto, P. Mohassel, and R. Safavi-Naini, editors, *Applied Cryptography and Network Security*, volume 7954 of *Lecture Notes in Computer Science*, pages 322–338. Springer Berlin Heidelberg, 2013.

[21] A. Shamir. Identity-Based Cryptosystems and Signature Schemes. In G. Blakley and D. Chaum, editors, *Advances in Cryptology*, volume 196 of *Lecture Notes in Computer Science*, pages 47–53. Springer Berlin Heidelberg, 1985.

[22] T. Unterluggauer and E. Wenger. Efficient Pairings and ECC for Embedded Systems. In L. Batina and M. Robshaw, editors, *Cryptographic Hardware and Embedded Systems - CHES 2014*, Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2014.

[23] F. Vercauteren. Optimal Pairings. *Information Theory, IEEE Transactions on*, 56(1):455–461, 2010.

[24] E. Wenger, T. Unterluggauer, and M. Werner. 8/16/32 Shades of Elliptic Curve Cryptography on Embedded Processors. In G. Paul and S. Vaudenay, editors, *Progress in Cryptology INDOCRYPT 2013*, volume 8250 of *Lecture Notes in Computer Science*, pages 244–261. Springer International Publishing, 2013.

[25] C. Whelan and M. Scott. Side Channel Analysis of Practical Pairing Implementations: Which Path Is More Secure? In P. Nguyen, editor, *Progress in Cryptology - VIETCRYPT 2006*, volume 4341 of *Lecture Notes in Computer Science*, pages 99–114. Springer Berlin Heidelberg, 2006.

[26] Xilinx, Inc. Xilinx Virtex-II Pro Data Sheet, Jun 2014. URL http://www.xilinx.com/support/documentation/data_sheets/ds083.pdf.