SEMA and MESD Leakage of TinyECC 2.0 on a LOTUS Sensor Node

Jacek Samotyja[1], Kerstin Lemke-Rust[1], and Markus Ullmann[1,2]

[1] Bonn-Rhein-Sieg University of Applied Sciences
D-53757 Sankt Augustin, Germany
{Jacek.Samotyja, Kerstin.Lemke-Rust, Markus.Ullmann}@h-brs.de

[2] Federal Office for Information Security
D-53175 Bonn, Germany
Markus.Ullmann@bsi.bund.de

Abstract. TinyECC 2.0 is an open source library for Elliptic Curve Cryptography (ECC) in wireless sensor networks. This paper analyzes the side channel susceptibility of TinyECC 2.0 on a LOTUS sensor node platform. In our work we measured the electromagnetic (EM) emanation during computation of the scalar multiplication using 56 different configurations of TinyECC 2.0. All of them were found to be vulnerable, but to a different degree. The different degrees of leakage include adversary success using (i) Simple EM Analysis (SEMA) with a single measurement, (ii) SEMA using averaging, and (iii) Multiple-Exponent Single-Data (MESD) with a single measurement of the secret scalar. It is extremely critical that in 30 TinyECC 2.0 configurations a single EM measurement of an ECC private key operation is sufficient to simply read out the secret scalar. MESD requires additional adversary capabilities and it affects all TinyECC 2.0 configurations, again with only a single measurement of the ECC private key operation. These findings give evidence that in security applications a configuration of TinyECC 2.0 should be chosen that withstands SEMA with a single measurement and, beyond that, an addition of appropriate randomizing countermeasures is necessary.

Keywords: TinyECC 2.0, Side Channel Analysis, SEMA, MESD, LOTUS Sensor Node, Wireless Sensor Network.

1 Introduction

Wireless sensor nodes are the building blocks of many ubiquitous applications in the Internet of Things (IoT), Ambient Intelligence, and Cyber Physical Systems. Such a Wireless Sensor Network (WSN) consists of many sensor nodes. Sensor nodes gather physical properties of an environment such as temperature or sound and communicate with a gateway sensor node, possibly via other sensor nodes. Each sensor node is equipped with a battery and communicates with other nodes through a wireless channel. The communication channels of a WSN

are therefore susceptible to sniffing attacks. To reach confidentiality on the wireless communication channel as well as to establish secure routing or enable node authentication, the sensor nodes need to be equipped with cryptographic implementations. For this, symmetric or public-key cryptographic primitives can be used. However, public-key cryptography is of advantage if there is a need of key establishment, message authentication or authentication of broadcast messages.

For public-key cryptography, a performance advantage of Elliptic Curve Cryptosystems (ECC) over the RSA algorithm is well proven, e.g., it was demonstrated in [7] for small microprocessor platforms. TinyECC [16, 19] is a configurable cryptographic library for ECC that has been developed by North Carolina State University for the usage with the operating system TinyOS [1] on sensor nodes. It offers implementations of EC Diffie-Hellman key exchange (ECDH), EC digital signature algorithm (ECDSA), and EC public-key encryption (ECIES). Further, it includes various configuration options for optimizing the computation on an elliptic curve. TinyECC will guide real IoT ECC implementations. Therefore it is very important that especially open source implementations are not severely vulnerable on known attacks.

Since 1996, a new class of implementation attacks on cryptographic algorithms appeared that exploits the observation of side channels such as the consumed time, power, or the emitted electro-magnetic radiation during computation in order to determine secret keys of an implementation. This class of passive implementation attacks is called Side Channel Cryptanalysis (SCC). While the first published attack was a timing attack [14], the most relevant publication in this field is about Simple Power Analysis (SPA) and Differential Power Analysis (DPA) [15]. Later on, it was shown that also the emitted electromagnetic (EM) radiation during computation can be successfully exploited [6]. The same methods used in [15] are referenced as Simple Electromagnetic Analysis (SEMA) and Differential Electromagnetic Analysis (DEMA) if the EM emanation is used as side channel. SPA/SEMA uses a direct visible interpretation of measurements. It may be successful with one single measurement and it does not require the knowledge of input and output data. DPA/DEMA computes statistics between predictable key-dependent internal states of the computation and the physical leakage. For this, DPA/DEMA needs the knowledge of input or output data of the computation and requires a sufficient number of measurements.

In this paper we present practical EM side channel analysis results of a TinyECC 2.0 implementation on a LOTUS sensor node. A LOTUS node is an advanced 32-bit wireless sensor node platform [10, 17]. In our work we tested 56 configurations of TinyECC 2.0 regarding side channel leakage on this platform. We tested the vulnerability of these configurations on SEMA and, additionally, on MESD [18] attacks which was rarely studied on ECC implementations before. It it is shown that SEMA compromises the secret scalar of 30 tested TinyECC 2.0 configurations after just one single EM measurement of an ECC private key operation. Moreover, all tested configurations turned out to be vulnerable on MESD using one single EM measurement of the secret scalar in an ECC private key operation.

2 Elliptic Curve Cryptography

Elliptic curve cryptography (ECC) is a very efficient technology to realize public key cryptosystems and public key infrastructures (PKI). A comprehensive introduction to ECC is given in [9].

An elliptic curve E is an algebraic structure. E is defined by an equation over an arbitrary field M:

$$E: \quad y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6 \tag{1}$$

where $a_1, a_2, a_3, a_4, a_6 \in M$. Equation (1) is called a *Weierstrass equation* over *Affine Coordinates*.

Fields are abstractions of number systems such as real numbers $\mathbb{R}$. Fields consists of a set of elements and two operations, usually denoted addition and multiplication. Due to implementation and performance issues *finite fields* are regarded called *Galois Field GF()*, too. The *order* of a Galois field is the number of elements of the field GF(). A Galois Field $GF(q)$ with order q exists only if $q = p^m$ where p is a prime number and m a positive integer. p is called *characteristic* of the Galois Field in this case. If $m = 1$, $GF(p)$ is called a *prime field*, if $m \geq 1$, $GF(p^m)$ is called an *extension field* and if $p = 2$, $GF(2^m)$ is called a *binary field*. In TinyECC only elliptic curve groups over prime fields $GF(p)$ are regarded. So we restrict to elliptic curves over prime field in this paper. A prime field $GF(p)$ consists of the following elements: {0, 1, 2, ..., p - 1}.

If the characteristic of p is different from 2 or 3 equation (1) is transformed to

$$y^2 = x^3 + ax + b \ \ mod\, p \tag{2}$$

where $a, b \in GF(p)$ and $4a^3 + 27b^2 \not\equiv 0$.

Equation (2) describes an *elliptic curve* $\mathbf{E}$ over $GF(p)$ which consists of all pairs (x_i, y_i) where $x_i, y_i \in GF(p)$ which solve equation (2). The pairs (x_i, y_i) are called *ECC curve points*. Additionally, the *point at infinity* ∞ is also said to be on the elliptic curve. Given an elliptic curve E, there is a defined operation for adding two curve points $\mathbf{G} : (x_1,\ y_1)$ and $\mathbf{Q} : (x_2,\ y_2)$ to produce a third point $\mathbf{T} : (x_3,\ y_3)$ with $\mathbf{G} + \mathbf{Q} = \mathbf{T}$. $\mathbf{T}$ is element of the given elliptic curve E, too. With this addition rule + the set of points of E ($GF(p)$) forms an (additive) Abelian group with ∞ serving as identity element. The addition rule contains the case that two identical points $\mathbf{G} : (x_1,\ y_1)$ are added: $\mathbf{G} + \mathbf{G} = \mathbf{2\,G}$ (point doubling) with $\mathbf{2\,G}$ on the elliptic curve E. This idea is used to define a cyclic subgroup:

$$\langle G \rangle = \{\infty, \mathbf{G}, \mathbf{2\,G}, \mathbf{3\,G}, ..., \mathbf{(n-1)\,G}\} \tag{3}$$

$\mathbf{G}$ is the *base point* of the cyclic group $\langle G \rangle$ and n is the order of $\mathbf{G}$. The calculation of $k\,\mathbf{G}$, where k is an integer, is called a *scalar multiplication*. The problem of finding k given the ECC points $\mathbf{G}$ and $k\,\mathbf{G}$ is called *elliptic curve discrete logarithm problem (ECDLP)*. It is computational infeasable to solve ECDLP for

appropriate domain parameters. The prime p of $GF(p)$, the coefficients a, b of equation (2), $\mathbf{G}$ and its order n form the domain parameter of a cyclic subgroup.

The ECDLP is often exploited in elliptic curve cryptography, e.g., to construct elliptic curve keys. A private key is an integer d that is selected uniformly at random from the interval {1, ..., n - 1}. The result $\mathbf{S} = d\,\mathbf{G}$ is the according public key. $(d, \mathbf{S})$ forms an elliptic key pair. The scalar multiplication is computed by a sequence of point doublings and conditional point additions on the elliptic curve whereat the concrete formulas for point doublings and point additions depend on the choice of coordinates and the underlying modular arithmetics. The ECDLP is utilized in ECC schemes such as *Diffie-Hellman Key Agreement (ECDH)*, *Elliptic Curve Digital Signature Algorithm (ECDSA)* or the *Elliptic Curve Integrated Encryption Scheme (ECIES)*.

3 TinyECC 2.0 and the LOTUS Sensor Node

TinyECC, a Configurable Library for Elliptic Curve Cryptography in Wireless Sensor Networks (Version 2.0) is released on 2/3/2011 [19]. It was developed at the Cyber Defense Laboratory, a Department of Computer Science at the North Carolina State University with support by the National Science Foundation (NSF) and the US Army Research Office (ARO).

TinyECC 2.0 is implemented in the programming language nesC. It is developed for use in TinyOS-2.x operating systems. It offers implementations of an EC Diffie-Hellman key exchange (ECDH), EC digital signature algorithm (ECDSA) and a public-key encryption (ECIES). It includes the following variants of optimizations for computing on an elliptic curve [16]:

- Barrett Reduction (see also [2]),
- Curve-Specific Optimization,
- Inline Assembly and Hybrid Multiplication,
- Sliding Window Method (see also [2]),
- Projective Coordinate Systems,
- Shamir's Trick.

Barrett Reduction and Curve-Specific Optimization optimize the modular reduction. Inline Assembly and Hybrid Multiplication uses Inline-Assembly code for speeding up *double* and *add* operations in particular. This Inline-Assembly optimization is developed for the sensor nodes MICAz, TelosB/Tmote Sky and Imote2 motes. The Sliding Window Method reduces the number of additions by computing a window of some bits of the scalar. Elliptic curves can be defined over different coordinate systems. Point addition in affine coordinates require the calculation of a multiplicative inverse. If inversion is significant more expensive than multiplication it is advantageous to represent ECC points using projective or Jacobian representation. So, the scalar multiplication is also speeded-up by using a weighted Jacobian representation in contrast to affine coordinates. Shamir's Trick reduces the signature verification time.

TinyECC 2.0 uses ECC curves and parameters that are recommended by Standards for Efficient Cryptography Group (SECG) [22, 19]. Concretely, the curves secp128r1, secp128r2, secp160k1, secp160r1, secp160r2, secp192k1, and secp192r1 are used. The numbers 128, 160, and 192 are the bits of key length. The addition 'k' denotes that the parameters are associated with the Koblitz curve. The addition 'r' marks that the parameter are chosen randomly. Note, that there is an ongoing trend towards higher key lengths of elliptic curves, e.g., according to [21] the minimum key length is increased to 192 bit. However, for a medium security level ECC key lengths of 128 and 160 bit should still be acceptable.

3.1 LOTUS Sensor Node

For our analysis of TinyECC 2.0 we chose the sensor node LOTUS because it belongs to the most advanced generation of sensor nodes [10, 17] that is also intended for security applications. It is equipped with a low-power ARM Cortex M3 32-bit processor. The current version of TinyOS Version 2.1.2 does not support this sensor node. The manufacturer MEMSIC Inc of the Lotus sensor node offers a development environment MoteWorks3.4. MoteWorks comprises TinyOS in a version 1.x and further product specific optimizations [11]. For the use of TinyECC 2.0 in this development environment, TinyECC 2.0 and MoteWorks 3.4 need to be integrated. For this, we did very small adjustments that are mainly renamings of functions, an insertion of a hardware specific macro, and an extension of the interfaces random.nc and RandomLFSR.nc with a function rand32().

3.2 Configurations of TinyECC 2.0 on the LOTUS Sensor Node

As described in Section 3, TinyECC 2.0 supports six optimizations. Among them, we cannot use the Inline Assembly and Hybrid Multiplication because it is an inline Assembly code for the sensor nodes MICAz, TelosB/Tmote Sky and Imote2 mote that cannot be used with the LOTUS sensor node. This is certainly a restriction when using TinyECC 2.0 on a modern platform. Further, we do not use the optimization "Shamir's Trick" in our tests.

TinyECC 2.0 provides seven elliptic curve parameters they are recommended by SECG. For the calculation on each elliptic curve we used both affine coordinates and projective coordinates. Each combination of elliptic curve and coordinates is tested with the optimizations "Simply", "Barrett Reduction", "Curve-Specific Optimization" and "Sliding Window Method". The term "Simply" indicates that no optimization is applied. Altogether, this results in 56 different configurations of TinyECC 2.0 on the LOTUS sensor node that are analyzed on side channel leakage in Section 5.

4 Related Work on Side Channel Analysis

Computation of the scalar multiplication is efficiently done using the double-and-add algorithm shown in Algorithm 1 that is a left-to-right binary multiplication algorithm [2, 4]. The double-and-add algorithm is analogous to the common square-and-multiply algorithm for computing the modular exponentiation.

Algorithm 1 Double-and-Add Algorithm

Require: elliptic curve point $\mathbf{P}$ and a positive integer $d = (1, d_{t-2}, ..., d_1, d_0)_2$
Ensure: $\mathbf{S} = d\,\mathbf{P}$
1: $\mathbf{Q} \leftarrow \mathbf{P}$;
2: **for** i from $t-2$ to 0 **do**
3: $\quad \mathbf{Q} \leftarrow 2\,\mathbf{Q}$;
4: $\quad$ If $d_i = 1$ then $\mathbf{Q} \leftarrow \mathbf{Q} + \mathbf{P}$;
5: **end for**
6: Return $\mathbf{Q}$;

Coron [4] has pointed out that a naive implementation of Algorithm 1 is vulnerable against SPA if the point doubling (aka *double* operation) and point addition (aka *add* operation) can be visually distinguished in the power measurements of a scalar multiplication. Experimental results for distinguishing point doubling and point addition are provided by Örs et al. [20] on an FPGA based ECC implementation.

In [18] a powerful technique called "Multiple-Exponent, Single-Data" (MESD) was introduced for an exponentiation algorithm using the 'square-multiply' algorithm. For MESD, the adversary is assumed to have access to a special cryptographic device that exponentiates a chosen value using chosen exponents and further that the implementation of this special cryptographic device is identical to the device under attack that computes the exponentiation with the secret exponent. The attack starts from the first secret bit of the exponent that is processed. To attack the i-th bit of the secret exponent the adversary asks the special cryptographic device to exponentiate the key guess that the i-th bit is one. Afterwards the adversary computes the difference signal between the measured trace with the secret exponent and the measurement trace with the chosen exponent. If the difference remains zero for more than one subsequent operation the guess is correct. Otherwise, if the difference remains zero only for the next squaring operation the guess is wrong, i.e., the next secret exponent bit is zero. The amount of leakage determines whether one trace of the secret exponent is sufficient for analyzing the difference signal or whether multiple traces are needed. We are aware of only one paper [8] that has applied MESD to ECC implementations before.

Principle solutions for countermeasures are already given by Coron in [4]. For SPA resistance a double-and-add resistant algorithm is proposed that avoids conditional branching [4]. Further directions for preventing simple side channel

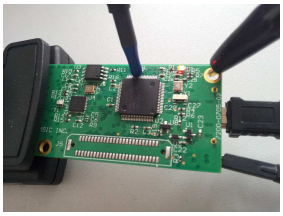
analysis are the use of unified formulae for point doubling and addition [3] and the use of inherently protected algorithms such as the Montgomery Ladder [12]. These countermeasures may secure against SPA/SEMA, but not against MESD and DPA/DEMA as the processed data are not randomized. For DPA resistance Coron [4] proposes three directions for countermeasures based on randomizing intermediate data: (i) randomization of the private exponent, (ii) blinding the point $\mathbf{G}$, and (iii) the use of randomized projective coordinates. The last two countermeasures are also effective to prevent an MESD attack. A valuable overview on the state-of-the-art of countermeasures for ECC can be found in [5].

5 Experimental Results

5.1 Measurement Set-Up

The measurement setup consists of a Lotus node and two digital storage Oscilloscopes, a Picoscope 5203 by Picotech and a LeCroy WaveRunner 640Zi. We used the near-field probes LF-B-3 and LF-U-2,5 from Langer EMV. As a trigger we used the green LED of the Lotus node (see Fig. 1(a)). The node was powered by an USB cable with 5 Volt and the CPU frequency was 16 MHz. The near-field probe LF-B-3 was positioned near the V_{CC} pin of the Lotus node (see Fig. 1(b)). The positioning of near-field probe LF-U-2,5 was on the resistor "R18" or on the surface of the microcontroller shown in Fig. 1(c).

(a) Lotus Node with trigger on LED D2



(b) LF-B-3 positioned near the VCC pin.

(c) LF-U-2,5 positioned on the surface of the microcontroller.

Fig. 1. Measurement setup.

5.2 SEMA and MESD Results

Experimental tests were applied at the computation of the scalar multiplication $d\,\mathbf{G}$ where d is the private key and $\mathbf{G}$ is the base point of the elliptic curve. This indeed is the computation of the public key $\mathbf{S} = d\,\mathbf{G}$, e.g., at key generation. To simplify testing we generally used a small bit sized value of $d = (101011011)_2$. This does not affect the generality of our attacks.

We followed a two-step approach:

1. First, we recorded a few measurements of the beginning of the scalar multiplication. For SEMA, we performed an optical inspection of the measurements curve whether we can distinguish the *double* and *add* operation. This step is repeated using different EM probe settings.
2. Second, we applied the MESD attack for all configurations that were not compromised using a single measurement in the first step, Again, we repeated this step using different EM probe settings.

The results of the analysis are shown in Fig. 2. For assessment of our results we define the following classifications:

- 's': SEMA leakage using a single measurement.
- 'a': SEMA leakage using an average curve of twenty of measurements.
- 'd': MESD leakage using a single measurement.
- 'wd': MESD leakage of the sliding window method using a single measurement.

These classifications are ordered towards attacks of increasing difficulty. We ensured that all configurations that are classified with 'a' can also be broken with MESD using only one single measurement of the secret scalar. The additional character '+' indicates that the operation *double* and *add* are clearly visible and the character '-' indicates that it is more difficult to separate the operations by optical inspection, but still feasible.

In the context of ECC protocol schemes, averaging of measurements of a secret scalar is a threat for ECIES and static ECDH schemes. In such a scheme SEMA using an average curve is the easier attack compared to an MESD attack. If a single measurement using SEMA or MESD is sufficient to extract the secret scalar this jeopardizes every ECC scheme.

Summarizing Fig. 2, there are fourteen extremely vulnerable configurations categorized as 's+' and sixteen vulnerable configurations categorized as 's-'. A single measurement of each of these thirty configurations is sufficient to expose the secret scalar. Using averaging of twenty single measurements, the secret scalar can be read out from ten configurations that are classified as 'a'. There are two configurations that withstand SEMA using averaging but can be broken with MESD, therefore they are classified as 'd'. All fourteen configurations using the sliding window method cannot be compromised using SEMA. This is obvious because there are multiple possibilities for each addend in the *add* operation so that distinguishing the *double* and *add* operations is not sufficient in these configurations. However, our results show that all fourteen configurations using

Affine Coordinates	secp128r1	secp128r2	secp160k1	secp160r1	secp160r2	secp192k1	secp192r1
Simply	s-	s-	s+	s+	s+	s+	s+
Barrett Reduction	s-	s-	s-	s-	s-	s-	s-
Curve-Specific Optimizations	s-	s-	s-	s-	s-	s-	s-
Sliding Window Method	wd	wd	wd	wd	wd	wd	wd

Projective Coordinates	secp128r1	secp128r2	secp160k1	secp160r1	secp160r2	secp192k1	secp192r1
Simply	a	a	a	a	a	s+	s+
Barrett Reduction	d	d	s+	s+	s+	s+	s+
Curve-Specific Optimizations	s+	s+	a	a	a	a	a
Sliding Window Method	wd	wd	wd	wd	wd	wd	wd

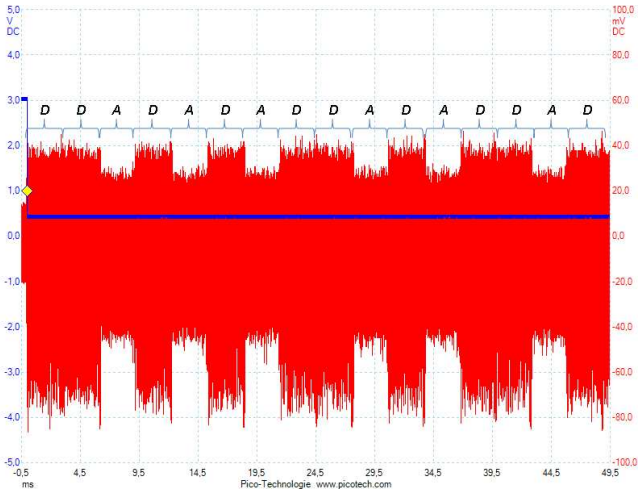
Fig. 2. Summary of Classification Results.

a sliding window method can be broken with MESD. It is important to note that with a good positioning of the EM probe it is possible to successfully apply MESD with only one single measurement of the secret scalar. A further result is that all configurations using affine coordinates – besides those using sliding window method – belong to the categorizations 's+' and 's-'. The results using projective coordinates are manifold, there are eight configurations classified as 's+', ten classifications classified as 'a' and two classifications 'd'. The results depend both on the elliptic curve and on the optimization, there is no clear advise to use a certain optimization. Below we provide some exemplary experimental results.

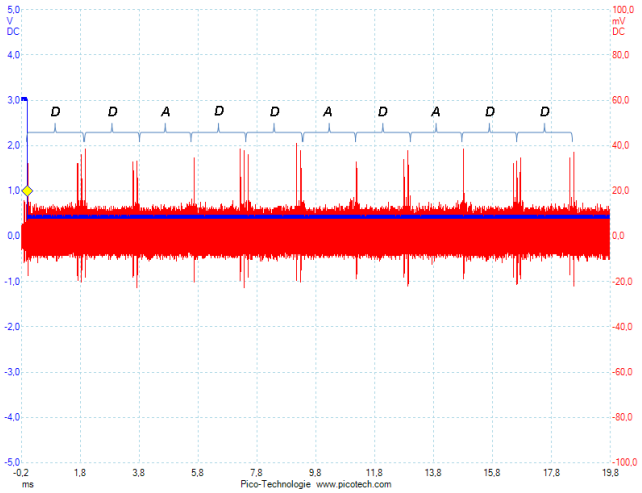
The EM leakage of an 's+' classified measurement is shown in Fig. 3(a). The operation *double* and *add* are clearly distinguishable. Therefore an adversary can easily identify the secret key by a single measurement. The classification 's-' must be viewed in more detail. Fig. 3(b) shows the EM measurement, the time range 1.8 to 3.8 ms shows a *double* operation and the time range 3.8 - 5.7 ms shows the *add* operation. The difference of the two operations is the following: the *double* operation start with a peak and ends with two peaks. The *add* operation doesn't have a peak. This difference is not the same for all classifications with 's-' but the difficulty is similar.

First, all configurations were tested with the Picoscope. Configurations that cannot be broken with a single-measurement were further analyzed using the WaveRunner 640Zi.

For averaging of EM measurements, twenty traces were found to be sufficient to show visible differences of the *double* and *add* operation. At the configuration "Curve-Specific Optimization" using projective coordinates with the two elliptic



(a) Classification 's+': Configuration "Simply" with affine coordinates and the secp160r1 elliptic curve parameter.

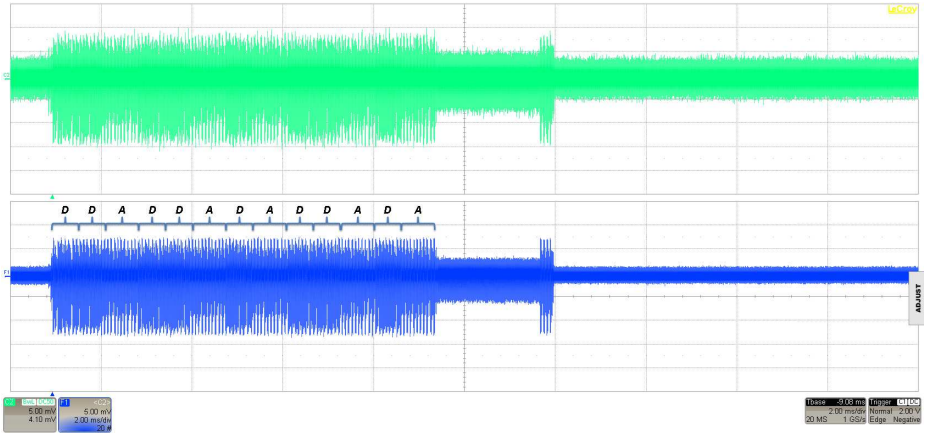


(b) Classification 's-': Configuration "Curve-Specific Optimization" with affine coordinates and the secp128r1 elliptic curve parameter.

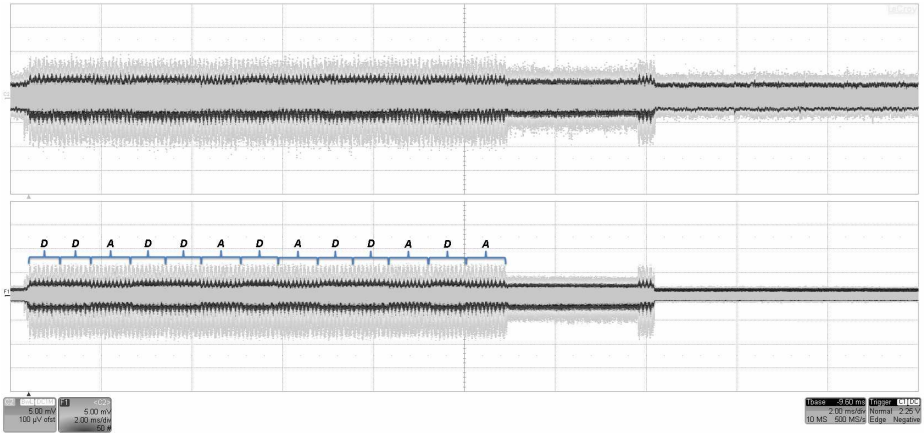
Fig. 3. Exemplary Results of the Classification 's'.

curve parameters secp192k1 and secp192r1 a difference could be found which was visible on the measuring instrument by switching the mode intensity.

For the classification 'd' we used the technique "Multiple-Exponent, Single-Data" (MESD) [18]. For simplification of our tests we used the same LOTUS sensor node for both the measurement of the secret scalar and the hypothetical



(a) Classification 'a': Configuration "Curve-Specific Optimization" using projective coordinates and secp160k1 elliptic curve parameter.

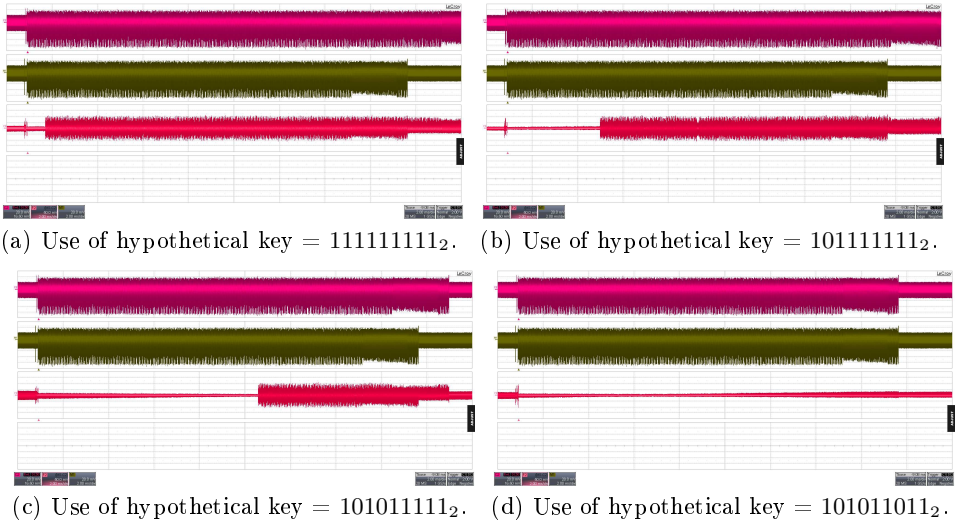


(b) Classification 'a': Configuration "Curve-Specific Optimization" using projective coordinates and secp192r1 elliptic curve parameter.

Fig. 4. Exemplary Results of the Classification 'a'.

scalar. If two LOTUS sensor nodes have to be used because the hypothetical key cannot be loaded in the target device, the measurement condition of the second device needs to be adjusted to the target device which demands for some additional post-processing in order to align the measurement traces. The first hypothetical key value was always $(111...1)_2$ in dual representation. The length depends on the length of the chosen elliptic curve parameter. We also measured a single trace and computed the difference of the traces. If the starting bit sequence from the secret scalar and the hypothetical key is equal, then the difference is remarkably reduced up to a certain time. If the starting bit sequence is not equal the difference persists. Next we built a new hypothesis by changing the bit on

the i-th position of the guessed scalar to 0 and measured the EM trace again. This is repeated until the last difference of the traces has disappeared. The last hypothetical key is then expected to be the correct secret scalar. If l is the length of the secret scalar, one expects l iterations of this analysis in the worst case.



(a) Use of hypothetical key = 111111111_2. (b) Use of hypothetical key = 101111111_2.

(c) Use of hypothetical key = 101011111_2. (d) Use of hypothetical key = 101011011_2.

Fig. 5. Exemplary Results of the classification 'd' using secp128r1 elliptic curve parameter. In each figure, the top curve is the single measurement of the guessed scalar and the second curve is the single of the secret scalar. If the bit sequences are made more equal, the differences approaches zero in the third curve.

In these experiments we found out that a delay must be considered if the secret scalar and the key hypothesis have a different bit length, see Fig. 6. An adversary can use this information to build the key hypothesis with the same length.

The classification 'wd' is exploitable by the technique MESD on the sliding window method. Usually, the EM leakage of the sliding window method makes the block length visible and one can distinguish between the *double* and the *add* operation. Without knowledge of the value of the additional point a direct extraction of the secret scalar is not possible. The sliding window method used with TinyECC 2.0 divides the bit string in blocks of 4 bits. The addend is chosen according to the value of a 4-bit block. If the 4-bit block is zero there is no *add* operation, i.e., in this case the 4-bit block can usually be directly identified as all-zero block. The MESD technique operates similar as described before at the 'd' classification. The difference is that the window size has to be determined first. For each block, 2^4-1 guessed exponents are loaded and analyzed. If the first block of scalar bits is correct the difference approaches zero for the *add* operation and at least for the next four subsequent *double* operations. The worst case effort

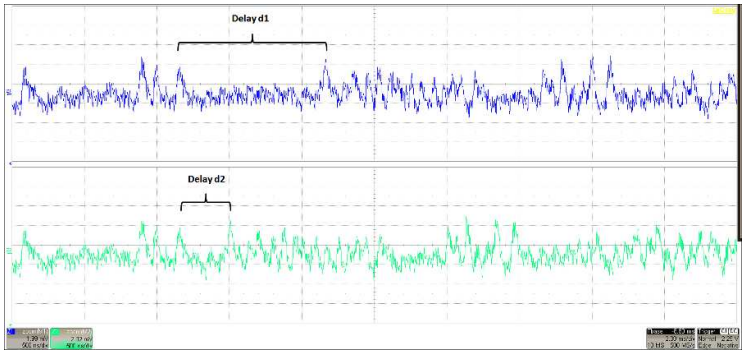
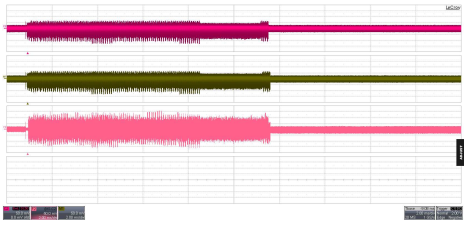
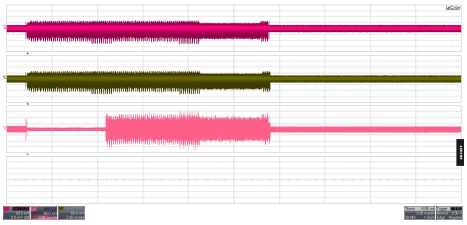


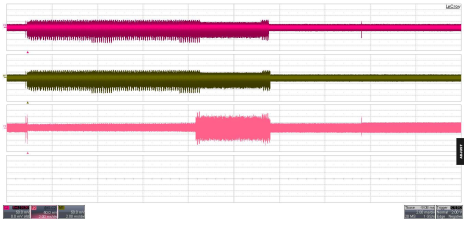
Fig. 6. Delay caused by the bit length of the scalar.



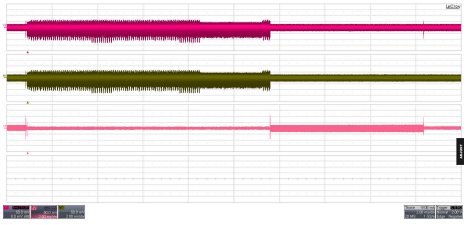
(a) Use of hypothetical key = 111111111111_2.



(b) Use of hypothetical key = 111111111_2.



(c) Use of hypothetical key = 101011111_2.



(d) Use of hypotheticalc key = 1010111011_2.

Fig. 7. Exemplary Results of the classification 'wd' using the sliding window exponentiation on elliptic curve secp160r1 and projective coordinates. In each figure, the top curve is the single measurement of the guessed scalar and the second curve is the single measurement of the secret scalar. If the bit sequences are made more equal, the differences approaches zero in the third curve.

needed by this classification is given by $b \cdot (2^w - 1)$ wherein b is the number of blocks of w-bit windows.

Note that the quality of EM analysis results depends strongly on the concrete positioning of the EM probe. It is therefore a difficult matter. There were three areas found, on the VCC-Pin, on the resistor "R18" and on the surface of the microcontroller shown in Fig. 1(c). The best positioning depends on the studied configuration of TinyECC 2.0 and has to be analyzed manually. Besides the

position of the EM probe the polar and azimuthal angle between the head of the EM probe and the chip surface are of importance. Some experience is needed to find a good positioning of the EM probe that enables that the analysis is successful. The voltage of the measurements was between 10 mV and 100 mV.

Full Key Recovery It is worth noting that the SEMA characteristics of the measurements do not change if a full-bit sized scalar is used. Full key recovery boils then down to a question of the capabilities of the oscilloscope, i.e., it needs to be checked whether a single measurement can be completely recorded with a sampling rate that is sufficient for a successful SEMA attack. In order to analyze full key recovery with one single measurement we exemplarily investigated the configurations "Simply" ('s+') and "Barrett Reduction" ('s-') on the curve secp192r1 using affine coordinates which lead to the largest execution times of the scalar multiplication. Both take approximately 1.1 seconds with a randomly chosen scalar d. Using the Picoscope 5203 there is a buffer size of 32 MS that limits the maximum sampling rate to 20 MS/s with one channel and to 10 MS/s with two channels. 10 MS/s turned out to be sufficient to break the 's+' characterization, here an undersampling can be easily tolerated. For the 's-' characterization it is significantly more difficult. However, it was verified that a sampling rate of 10 MS/s also allows to break this configuration with one single measurement.

6 Conclusion

It is revealed that the susceptibility of the analyzed 56 TinyECC 2.0 configurations towards SEMA is manifold. Though in the end the secret scalar of all configurations can be compromised, they are susceptible to different classes of attacks. This paper leads to a cautionary note that 30 configurations are vulnerable to a complete SEMA leak with one single measurement. Adding of randomizing countermeasures is assumed to fail to protect these configurations. Though the leakage characteristics also depend on the used hardware platform, LOTUS in our case, this is a strong indication that exploitable vulnerabilities of TinyECC 2.0 exist also on other platforms.

These findings are important for users of TinyECC 2.0. For practical applications with TinyECC 2.0 we recommend to use the sliding window method that has intrinsically a high resistance against SEMA attacks. Moreover, all configurations turned out to be vulnerable on MESD using a single measurement of the secret scalar. To counteract MESD attacks it is required that each run of the implementation uses internally a different representation of the processed data [18]. This can be achieved by blinding of the base point or the use of randomized projective coordinates [4]. Additionally, randomization of the secret scalar may be useful to counteract possible SEMD attacks [18].

References

1. TinyOS. Available from: `http://www.tinyos.net/`.
2. Paul C. van Oorschot Alfred J. Menezes and Scott A. Vanstone. *Handbook of Applied Cryptography*. CRC Press, 2001.
3. Eric Brier and Marc Joye. Weierstraß Elliptic Curves and Side-Channel Attacks. In *Public Key Cryptography, PKC 2002*, pages 335–345, 2002.
4. Jean-Sébastien Coron. Resistance against Differential Power Analysis for Elliptic Curve Cryptosystems. In Koç and Paar [13], pages 292–302.
5. Junfeng Fan, Xu Guo, Elke De Mulder, Patrick Schaumont, Bart Preneel, and Ingrid Verbauwhede. State-of-the-art of Secure ECC Implementations: A Survey on Known Side-channel Attacks and Countermeasures. In *HOST 2010, Proceedings of the 2010 IEEE International Symposium on Hardware-Oriented Security and Trust (HOST)*, pages 76–87, 2010.
6. Karine Gandolfi, Christophe Mourtel, and Francis Olivier. Electromagnetic Analysis: Concrete Results. In Ç Koç, D. Naccache, and C. Paar, editors, *Cryptographic Hardware and Embedded Systems - CHES 2001*, volume 2162 of *LNCS*, pages 251–261. Springer-Verlag, 2001.
7. Nils Gura, Arun Patel, Arvinderpal Wander, Hans Eberle, and Sheueling Chang Shantz. Comparing Elliptic Curve Cryptography and RSA on 8-bit CPUs. In Marc Joye and Jean-Jacques Quisquater, editors, *Cryptographic Hardware and Embedded Systems - CHES 2004*, volume 3156 of *Lecture Notes in Computer Science*, pages 119–132. Springer, 2004.
8. JaeCheol Ha and Sang-Jae Moon. Randomized signed-scalar multiplication of ECC to resist power attacks. In *Cryptographic Hardware and Embedded Systems - CHES 2002, pages = 551–563, year = 2002, crossref = DBLP:conf/ches/2002,*.
9. Darrel Hankerson, Scott Vanstone, and Alfred J Menezes. *Guide to elliptic curve cryptography*. Springer, 2004.
10. MEMSIC Inc. LOTUS, High-Performance Wireless Sensor Network Platform, Datasheet. Available from: `http://www.memsic.com/wireless-sensor-networks/`.
11. MEMSIC Inc. MoteWorks Getting Started Guide, 2013.
12. Marc Joye and Sung-Ming Yen. The Montgomery Powering Ladder. In *Cryptographic Hardware and Embedded Systems - CHES 2002*, pages 291–302, 2002.
13. Çetin Kaya Koç and Christof Paar, editors. *Cryptographic Hardware and Embedded Systems, First International Workshop, CHES'99*, volume 1717 of *Lecture Notes in Computer Science*. Springer, 1999.
14. Paul C. Kocher. Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems. In Neal Koblitz, editor, *Advances in Cryptology - CRYPTO '96*, volume 1109 of *Lecture Notes in Computer Science*, pages 104–113. Springer, 1996.
15. Paul C. Kocher, Joshua Jaffe, and Benjamin Jun. Differential Power Analysis. In Michael J. Wiener, editor, *Advances in Cryptology - CRYPTO '99*, volume 1666 of *Lecture Notes in Computer Science*, pages 388–397. Springer, 1999.
16. An Liu and Peng Ning. TinyECC: A configurable library for elliptic curve cryptography in wireless sensor networks. In *Information Processing in Sensor Networks, 2008. IPSN'08. International Conference on*, pages 245–256. IEEE, 2008.
17. V. Veena Madhuri, Syed Umar, and P. Veeraveni. A Study on Smart Dust (MOTE) Technology. *IJCSET, March 2013, Vol 3, Issue 3, 124-128*, 2013.

18. Thomas S. Messerges, Ezzy A. Dabbish, and Robert H. Sloan. Power Analysis Attacks of Modular Exponentiation in Smartcards. In Koç and Paar [13], pages 144–157.
19. Department of Computer Science NCSU College of Engineering. TinyECC: A Configurable Library for Elliptic Curve Cryptography in Wirelss Sensor Networks (Version 2.0). Available from: `http://discovery.csc.ncsu.edu/software/TinyECC/`.
20. Siddika Berna Örs, Elisabeth Oswald, and Bart Preneel. Power-Analysis Attacks on an FPGA - First Experimental Results. In Colin D. Walter, Çetin Kaya Koç, and Christof Paar, editors, *Cryptographic Hardware and Embedded Systems - CHES 2003*, volume 2779 of *Lecture Notes in Computer Science*, pages 35–50. Springer, 2003.
21. Certicom Research. SEC 2: Recommended Elliptic Curve Domain Parameters, January 27, 2010, Version 2.0. Available from: `http://www.secg.org/sec2-v2.pdf`.
22. Certicom Research. SEC 2: Recommended Elliptic Curve Domain Parameters, September 20, 2000, Version 1.0.