

Chapter 2

Mathematical and Cryptological Background

First, we provide the necessary mathematical background information on elliptic curves and bilinear pairings. We assume the reader is familiar with abstract algebra and basic probability theory. Next, we present the AES algorithm and introduce Identity-Based Encryption (IBE). The chapter continues with the explanation of side channel attacks and fault attacks. The necessary terminology is presented and related work is discussed.

2.1 Elliptic Curves and Bilinear Pairings

Some of the definitions in this section can also be given more general. However, we wrote this background as concrete as possible. Therefore, the definitions are often customized to our needs and not as universally valid as they are in most text books.

2.1.1 Elliptic Curves

Definition 2.1 (*Elliptic Curve*) An elliptic curve E over a field K is the set of all points $P = (x_P, y_P)$ which satisfy the Weierstrass equation

$$E : y^2 + a_1x \cdot y + a_3y = x^3 + a_2x^2 + a_4x + a_6, \quad (2.1)$$

together with $\mathcal{O}$, the point at infinity. The coefficients $a_1, a_2, a_3, a_4, a_6 \in K$ are chosen so that the curve is nonsingular, i.e., for all points $P = (x_P, y_P)$ its partial derivatives do not vanish simultaneously.

An additive group structure can be defined on E . The group $(E, \oplus)$, which we refer to as only E , consists of all points satisfying the Weierstrass equation together

with the point at infinity $\mathcal{O}$, which is the identity element in E . We denote the additive inverse of $P \in E$ with $-P$.

For any extension field L of K , we define the set of L -rational points on E as

$$E(L) = \{(x, y) \in L \times L : y^2 + a_1x \cdot y + a_3y - x^3 - a_2x^2 - a_4x - a_6 = 0\} \cup \{\mathcal{O}\}. \quad (2.2)$$

We denote the group order with $\#E$. The main result about the group order is Hasse's theorem.

Theorem 2.1 (Hasse's theorem) *Let E be an elliptic curve defined over a finite field $\mathbb{F}_q$. Then, $\#E(\mathbb{F}_q)$ satisfies*

$$|\#E(\mathbb{F}_q) - q - 1| \leq 2\sqrt{q}. \quad (2.3)$$

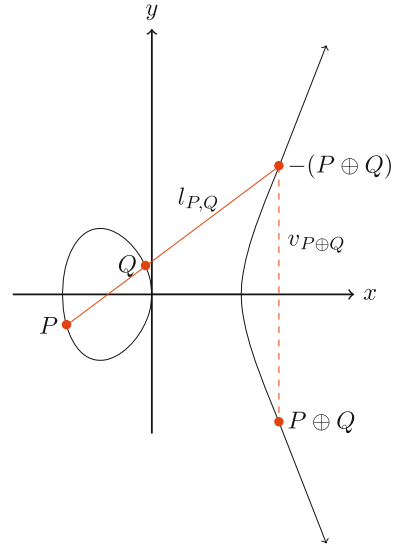
Proof See [164, p. 138]. □

Hasse's theorem only defines boundaries for the cardinality of E . The exact determination of $\#E(\mathbb{F}_q)$, which is also called point counting problem, is of critical importance for many cryptographic applications. Methods to solve this problem are discussed in [30].

The following explanation of addition and scalar multiplication in E is based on the chord and tangent law [73], since the pairing algorithms deployed in this work make use of this construction.

Figure 2.1 shows how the sum $P \oplus Q$ of two distinct points $P, Q \in E$ can be determined. For two points P and Q , the line through P and Q is denoted with

Fig. 2.1 Point addition of two distinct points on the elliptic curve $y^2 = x^3 - 2x$



$l_{P,Q}$. Since the Weierstrass equation is of degree 3 in x , there will always be a third intersection point of $l_{P,Q}$ and the curve [164]. Assuming that $l_{P,Q}$ is neither the tangent line at P or Q nor vertical, we denote the third point of intersection with $-(P \oplus Q)$. If we mirror this point at the x-axis, we obtain $P \oplus Q$, the sum of P and Q . We can identify this point by drawing the vertical line $v_{-(P \oplus Q)}$ through $-(P \oplus Q)$ and then take the intersection of this line with the curve, see Fig. 2.1. This point is the result $P \oplus Q$, the sum of P and Q . Hence, $v_{-(P \oplus Q)} = v_{P \oplus Q}$. In the case that $l_{P,Q}$ is the tangent of P or Q , the third point of intersection will be P or Q , respectively. In the case that $l_{P,Q}$ is vertical, i.e., Q is the reflection of P at the horizontal axis, the third point of intersection is $\mathcal{O}$.

To define scalar multiplication, we start with the definition of point doubling. Point doubling is the addition of the point to itself and thus, the procedure is analogous to point addition for two distinct points. To double P and compute $[2]P$, we draw the tangent line g_P through P at E . Since this tangent line intersects E with multiplicity 2 at P , there is only one further intersection point of g_P at E . In the case that the tangent line is not vertical, we denote this third point with $-(P \oplus P)$ and again reflect it at the x-axis to determine the point $P \oplus P$. The line connecting $-(P \oplus P)$ and $P \oplus P$ is again denoted with $v_{P \oplus P} = v_{-(P \oplus P)}$. In the case that the tangent line at P is vertical, the third point of intersection at E is $\mathcal{O}$, the point at infinity. Hence, we have $[2]P = \mathcal{O}$ in this situation. For the curve which is shown in Fig. 2.1, there are three points with a vertical tangent line. These are $(0, 0)$, $(\sqrt{2}, 0)$, and $(-\sqrt{2}, 0)$.

Scalar multiplication of elliptic curve points can now be calculated with successive point additions and point doublings. With $[n]P$, we denote scalar multiplication of P with $n \in \mathbb{Z}$. For $P \in E$ and $n \in \mathbb{N}$, we have

$$[n]P = \overbrace{P \oplus P \dots \oplus P}^{n \text{ times}}. \quad (2.4)$$

Consequently, we have

$$[-n]P = \overbrace{-P \oplus -P \dots \oplus -P}^{n \text{ times}} \quad (2.5)$$

and

$$[0]P = \mathcal{O}. \quad (2.6)$$

The point representation that is used in this work is referred to as affine coordinate system and affine coordinates. To speed up group operations and to mitigate side channel attacks, points on elliptic curves can also be represented in different coordinate systems such as projective coordinates [89] and Jacobian coordinates [124].

Definition 2.2 (*Torsion Point*) Let E be an elliptic curve. For $r \in \mathbb{Z}$ we define the set

$$E[r] = \{P \in E : [r]P = \mathcal{O}\}. \quad (2.7)$$

These points of finite order are called torsion points and $E[r]$ is the group of r -torsion points.

Definition 2.3 (*Supersingular Curve*) Let E be an elliptic curve defined over a field with characteristic p . If $E[p^n] = \{\mathcal{O}\}$ for all $n \geq 1$, E is called supersingular. Otherwise, E is called ordinary.

Definition 2.4 (*Distortion Map*) Let E be a supersingular elliptic curve and $r \in \mathbb{N}$ with $r \geq 2$. Let $P \in E$ so that $P \in E[r]$ and P is of exact order r . We call a homomorphism $\psi : E \rightarrow E$ distortion map, if $\{P, \psi(P)\}$ is a basis for $E[r]$ and hence, $\psi(P)$ is not in the cyclic subgroup generated by P .

Distortion maps were introduced in [183].

Definition 2.5 (*Twist*) Let E and E' be elliptic curves over $\mathbb{F}_q$. If there is an isomorphism $\phi_d : E' \rightarrow E$ defined over $\mathbb{F}_{q^d}$ with minimal d , then E' is a degree- d twist of E .

We refer the reader to [30, 31, 54, 104, 164] for a thorough treatment of elliptic curves. Information about elliptic curve cryptography can be found in [30, 31, 54, 89].

2.1.1.1 Elliptic Curve Discrete Logarithm

The Discrete Logarithm Problem (DLP) is a well-known mathematical problem on which cryptographic algorithms such as ElGamal encryption, the ElGamal signature scheme, and the Diffie-Hellman key establishment are based [119]. The analog problem exists for elliptic curves. It is called Elliptic Curve Discrete Logarithm Problem (ECDLP).

Definition 2.6 (*Elliptic Curve Discrete Logarithm Problem*) Let E be an elliptic curve over a finite field $\mathbb{F}_q$ and $P, Q \in E$ so that there is $n \in \mathbb{Z}$ with $Q = [n]P$. The Elliptic Curve Discrete Logarithm Problem (ECDLP) is, given P and Q , to find n .

As it is the case for the DLP on natural numbers, there are simple instances of the ECDLP, while the problem is not efficiently solvable for other groups and points [30]. For details on elliptic curve discrete logarithms and their application, we refer the reader to [73].

2.1.2 Bilinear Pairings

Bilinear pairings are bilinear maps defined over groups on elliptic curves. Originally, they have been used for cryptanalytic techniques [120]. In 2001, however, they gained the research community's attention when they were used to realize IBE [39], see Sect. 2.2.2. Today, a wide range of different pairings is used [11] and several cryptographic protocols such as attribute-based encryption [149], identity-based signatures [91], and key agreement protocols [96] are based on pairings. Moreover, pairings help to secure useful technologies such as WSNs [133]. Ever since pairings were proposed to be used for IBE, cryptanalysis of pairings and pairing-based schemes became an active field of research, e.g., [7, 74, 189].

For the definition of pairings, we first have to define the embedding degree.

Definition 2.7 (*Embedding Degree*) Let E be an elliptic curve defined over the finite field $\mathbb{F}_q$ and let r be an integer coprime to q with $r \mid \#E(\mathbb{F}_q)$. The embedding degree (with respect to r) is the smallest natural number k such that $r \mid (q^k - 1)$. It satisfies $\mathbb{F}_{q^k} = \mathbb{F}_q(\mu_r)$, where μ_r denotes the group of r th roots of unity in $\overline{\mathbb{F}_q}$.

Thus, the embedding degree is the smallest natural number so that $\mathbb{F}_{q^k}^*$ has a subgroup of order r , i.e., $r \mid (q^k - 1)$. Since k is the order of q modulo r , i.e., k is the order of q in the unit group $\mathbb{Z}_r^*$, it follows that k divides Euler's totient function $\phi(r)$. Hence, if r is prime, then k divides $(r - 1)$.

Definition 2.8 (*Pairing*) Let E be an elliptic curve defined over a finite field $\mathbb{F}_q$. We define three finite abelian groups $\mathbb{G}_1$, $\mathbb{G}_2$, and $\mathbb{G}_T$. The groups $\mathbb{G}_1$ and $\mathbb{G}_2$ are subgroups of E , while $\mathbb{G}_T$ is a subgroup of $\mathbb{F}_{q^k}^*$, with k the embedding degree. A pairing is an efficiently computable, non-degenerate bilinear map

$$e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T. \quad (2.8)$$

We refer the reader to [54] for a thorough treatment of bilinear pairings and their application. For details on the selection of pairing-friendly curves we refer to [72].

Most pairings $e(P, Q)$ on elliptic curves are computed by first computing the so-called Miller function $f_n, P(Q)$ [125] followed by an exponentiation to the power $z = (q^k - 1)/r$. To understand this Miller function and to study pairings in more detail, we have to address divisors and some of their characteristics.

Definition 2.9 (*Divisor*) For an elliptic curve E , a divisor D is a formal sum

$$D = \sum_{P \in E} n_P \cdot \langle P \rangle. \quad (2.9)$$

Only finitely many of the coefficients $n_P \in \mathbb{Z}$ are nonzero. The divisor associated to the point $P \in E$ is denoted with $\langle P \rangle$.

The divisors generated by the points of E form a free abelian group. We denote the set of all divisors generated by the points of E with $\text{Div}(E)$.

We want to define the divisor associated to a rational function. For this definition, we first need to define the order of a rational function at a point of an elliptic curve [164].

Definition 2.10 (*Order of a function at a point*) Let E be an elliptic curve defined over a finite field $\mathbb{F}_q$, $P \in E$, and $f \in \overline{\mathbb{F}_q}[E]_P$. The order of f at P is

$$\text{ord}_P(f) = \sup\{d \in \mathbb{Z} : f \in M_P^d\}, \quad (2.10)$$

where M_P denotes the maximal ideal of the local ring $\overline{\mathbb{F}_q}[E]_P$.

For a nonzero rational function $f = g/h \in \overline{\mathbb{F}_q}(E)$ with $g, h \in \overline{\mathbb{F}_q}[E]_P$, we define

$$\text{ord}_P(f) = \text{ord}_P(g) - \text{ord}_P(h). \quad (2.11)$$

If $\text{ord}_P(f) > 0$, we say that f has a zero at P while we say that f has a pole at P if $\text{ord}_P(f) < 0$. If $\text{ord}_P(f) \geq 0$, then f is defined at P and $f(P)$ can be evaluated. Otherwise f has a pole at P and we write $f(P) = \infty$.

Definition 2.11 (*Divisor associated to a function*) Let E be an elliptic curve defined over a finite field $\mathbb{F}_q$, and $f \in \overline{\mathbb{F}_q}(E)$ a nonzero rational function. The divisor associated to f is

$$\text{div}(f) = \sum_{P \in E} \text{ord}_P(f) \cdot \langle P \rangle. \quad (2.12)$$

A divisor is called principal if it is equal to a divisor that is associated to a function. Thus, $D \in \text{Div}(E)$ is principal if $D = \text{div}(f)$ for some nonzero rational function $f \in \overline{\mathbb{F}_q}(E)$. We say that two divisors $D_1, D_2 \in \text{Div}(E)$ are linearly equivalent if $D_1 - D_2$ is principal. To denote that two divisors are linearly equivalent, we write $D_1 \sim D_2$.

Definition 2.12 (*Support of a Divisor*) Let E be an elliptic curve and $D \in \text{Div}(E)$ with $D = \sum_{P \in E} n_P \cdot \langle P \rangle$. The support of D is the set of all points $P \in E$ with $n_P \neq 0$. Two divisors are coprime when their supports are disjoint.

For a divisor $D \in \text{Div}(E)$ and a principal divisor $\text{div}(f)$ for some nonzero rational function $f \in \overline{\mathbb{F}_q}(E)$, we can define the evaluation of f at D if D and $\text{div}(f)$ are coprime. Then, we define $f(D) = \prod_{P \in E} f(P)^{n_P}$.

With the help of divisors, we can now define the Miller function, which is the heart of most pairings.

Definition 2.13 (*Miller function*) Let E be an elliptic curve defined over a finite field $\mathbb{F}_q$. A rational function $f_{n,P} \in \overline{\mathbb{F}_q}(E)$, with $P \in E$ and $n \in \mathbb{N}$, is a Miller function if its divisor satisfies

$$\text{div}(f_{n,P}) = n \cdot \langle P \rangle - \langle [n]P \rangle - (n-1)\langle \mathcal{O} \rangle. \quad (2.13)$$

The Miller function can also be defined recursively as follows [36, 125]:

$$f_{n+1,P} := \begin{cases} 1 & \text{if } n = 0 \\ f_{n,P} \cdot l_{P,[n]P}/v_{[n+1]P} & \text{else.} \end{cases}$$

As stated above, most pairings on elliptic curves are computed by first evaluating the Miller function, followed by an exponentiation. The evaluation of the Miller function can be efficiently computed with the Miller Algorithm, see Algorithm 2.1.

Algorithm 2.1 Miller Algorithm and final exponentiation.

Require: $P, Q \in E, n = \sum_{j=0}^{t-1} n_j 2^j$ with $n_j \in \{0, 1\}$ and $n_{t-1} = 1$

Ensure: $f_{n,P}(Q)^z$

```

1:  $T \leftarrow P, f \leftarrow 1$ 
2: for  $j = t - 2 \dots 0$  do
3:    $f \leftarrow f^2 \cdot g_T(Q)/v_{[2]T}(Q)$ 
4:    $T \leftarrow [2]T$ 
5:   if  $n_j = 1$  then
6:      $f \leftarrow f \cdot l_{T,P}(Q)/v_{T \oplus P}(Q)$ 
7:      $T \leftarrow T \oplus P$ 
8:   end if
9: end for
10:  $f \leftarrow f^z$  ▷ final exponentiation
11: return  $f$ 
```

The Miller Algorithm successively computes the required function in $\lfloor \log_2(n) \rfloor$ iterations. Since this algorithm utilizes a **for** loop, see Lines 2–9, this evaluation is called Miller loop [46]. The value n , which determines the number of loop iterations, is often called Miller bound. The variable f , which is updated during the computation of the **for** loop until it stores the value of the evaluation of the Miller function, is called Miller variable.

Independent of the concrete design of a pairing, there are three different types of pairings. Following [75], we distinguish between pairings of Type 1, Type 2, and Type 3. These three basic types differ in the similarity of their domains $\mathbb{G}_1$ and $\mathbb{G}_2$.

- In a Type 1 pairing, both groups are equal, i.e., $\mathbb{G}_1 = \mathbb{G}_2$.
- In a Type 2 pairing, the groups are different, i.e., $\mathbb{G}_1 \neq \mathbb{G}_2$, but it exists an efficiently computable homomorphism $\phi : \mathbb{G}_2 \rightarrow \mathbb{G}_1$.
- In a Type 3 pairing, the groups are different, i.e., $\mathbb{G}_1 \neq \mathbb{G}_2$, and there are no efficiently computable homomorphisms between $\mathbb{G}_1$ and $\mathbb{G}_2$.

Type 1 pairings are also called symmetric pairings, while Type 2 and Type 3 pairings are called asymmetric pairings. The choice of the type of the pairing is related to the elliptic curve that is used, e.g., Type 1 pairings are generally implemented using supersingular curves over $\mathbb{F}_p$, $\mathbb{F}_{2^m}$, and $\mathbb{F}_{3^m}$ [158].

Those pairings which are used in cryptography require that their inversion is not efficiently computable. There are four mathematical problems connected to pairing inversion.

Definition 2.14 (*FAPI-1*) Given a point $P \in \mathbb{G}_1$ and a value $\alpha \in \mathbb{G}_T$, both chosen at random, the fixed argument pairing inversion problem (FAPI-1) is to find $Q \in \mathbb{G}_2$ such that $e(P, Q) = \alpha$ [74].

The problem can also be defined with P unknown and $Q \in \mathbb{G}_2$ chosen at random. The problem is then called FAPI-2.

Both FAPI-1 and FAPI-2 treat the pairing as a black box and only consider the inputs and the output. However, analogous to the two steps of the pairing calculation, i.e., Miller Algorithm and final exponentiation, the pairing inversion can also be treated as a two-step process [100]. Hence, FAPI-1 is usually split into two parts in the literature: the exponentiation inversion [46] and the Miller inversion [74].

Definition 2.15 (*Exponentiation Inversion*) Given the output of the pairing as well as $P \in \mathbb{G}_1$ and the final exponent z , the exponentiation inversion problem is to find the correct preimage of the final exponentiation, i.e., the field element $f_{n,P}(Q)$.

Definition 2.16 (*Miller Inversion*) The Miller inversion problem is, given n , $P \in \mathbb{G}_1$, and a field element $f_{n,P}(Q)$, to find the correct input $Q \in \mathbb{G}_2$.

Recently, a novel theoretic idea for pairing inversion was presented [100]. Kanayama and Okamoto show that the Miller inversion does not have to be solved in several cases provided that a generic algorithm for solving the exponentiation inversion exists. Thus, given such an algorithm, pairing inversion can be reduced to exponentiation inversion for a special family of pairings, the Ate_i pairings. This work was later revised and extended [46].

Related to the inversion of pairings, especially when it comes to fault attacks, is the Hidden Root Problem (HRP), which was introduced only in 2008 [181]. We define the HRP exactly as it was defined in the original publication.

Definition 2.17 (*Hidden Root Problem*) Let $\mathbb{F}_q$ be a finite field with $q = p^n$ elements, where p is prime and let e be a positive integer with $e \mid (q - 1)$. Let $f_x(\cdot) : \mathcal{D}(\mathbb{F}_q) \rightarrow \mathbb{F}_q$ be a specified map, i.e., the precise description of which is given, depending on x from a domain $\mathcal{D}$ to $\mathbb{F}_q$. Let $\mathcal{O}_x(\cdot)$ denote an oracle that on input $\alpha \in \mathcal{D}$ returns

$$\xi_\alpha = f_x(\alpha)^e \quad (2.14)$$

for a fixed secret $x \in \mathbb{F}_q$. The Hidden Root Problem is to recover x in expected polynomial time in $\log q$ by querying the oracle repeatedly for chosen $\alpha_i \in \mathcal{D}(\mathbb{F}_q)$.

2.1.2.1 The Tate Pairing

We describe the Tate pairing and its reduced variant following [182].

Definition 2.18 (*Tate Pairing*) Let E be an elliptic curve defined over $\mathbb{F}_q$. Let $r \mid \#E(\mathbb{F}_q)$ so that r and q are coprime. Let k be the embedding degree. The Tate pairing $\hat{t}$ is defined as

$$\begin{aligned} \hat{t} : E(\mathbb{F}_{q^k})[r] \times E(\mathbb{F}_{q^k})/rE(\mathbb{F}_{q^k}) &\rightarrow \mathbb{F}_{q^k}^* / (\mathbb{F}_{q^k}^*)^r \\ (P, Q) &\mapsto \hat{t}(P, Q) = f_{r,P}(D_Q). \end{aligned} \quad (2.15)$$

Here, $\text{div}(f_{r,P}) = r \cdot \langle P \rangle - r \cdot \langle \mathcal{O} \rangle$ and $D_Q \sim \langle Q \rangle - \langle \mathcal{O} \rangle$ is coprime to $\text{div}(f_{r,P})$.

To yield a unique result instead of a representative of an equivalence class, we define the reduced Tate pairing t .

Definition 2.19 (*Reduced Tate Pairing*) Let E be an elliptic curve defined over $\mathbb{F}_q$. Let $r \mid \#E(\mathbb{F}_q)$ so that r and q are coprime. Let k be the embedding degree. The reduced Tate pairing t is defined as

$$\begin{aligned} t : E(\mathbb{F}_{q^k})[r] \times E(\mathbb{F}_{q^k})/rE(\mathbb{F}_{q^k}) &\rightarrow \mu_r \subset \mathbb{F}_{q^k}^* \\ (P, Q) &\mapsto t(P, Q) = \hat{t}(P, Q)^{(q^k-1)/r}. \end{aligned} \quad (2.16)$$

Thus, the reduced Tate pairing consists of two stages: first the Miller function that is parameterized by P is evaluated at D_Q , then a final exponentiation follows to ensure that the algorithm outputs a unique value.

2.1.2.2 The Eta Pairing

The eta pairing can be regarded as an optimized version of the reduced Tate pairing [19, 182]. The optimization consists in a shortened Miller loop. To define the eta pairing, we first need to introduce the Frobenius endomorphism.

Definition 2.20 (*Frobenius Endomorphism*) Let E be an elliptic curve defined over a finite field $\mathbb{F}_q$. The endomorphism

$$\begin{aligned} \phi_q : E &\rightarrow E, \\ (x, y) &\mapsto (x^q, y^q) \end{aligned} \quad (2.17)$$

is called Frobenius endomorphism. The group $E(\mathbb{F}_q)$ of $\mathbb{F}_q$ -rational points on E is fixed by ϕ_q since the q th power is the identity on $\mathbb{F}_q$.

Definition 2.21 (*eta Pairing*) Let E be a supersingular elliptic curve defined over $\mathbb{F}_q$. Let $r \mid \#E(\mathbb{F}_q)$ so that r and q are coprime and let k be the embedding degree. Let n be any integer with $n \equiv q \pmod{r}$ so that $(n^k - 1)/r$ and r are coprime. Let $\mathbb{G}_1, \mathbb{G}_2 \subseteq E$

be restricted to the Eigenspaces of Frobenius with $\mathbb{G}_1 = E[r] \cap \ker(\phi_q - [1])$ and $\mathbb{G}_2 = E[r] \cap \ker(\phi_q - [q])$. The eta pairing η_n is defined as

$$\begin{aligned} \eta_n : \mathbb{G}_1 \times \mathbb{G}_2 &\rightarrow \mu_r \subset \mathbb{F}_{q^k}^* \\ (P, Q) &\mapsto \eta_n(P, Q) = f_{n, P}(Q)^{(q^k-1)/r}. \end{aligned} \quad (2.18)$$

2.2 Cryptographic Algorithms and Protocols

This section first presents the AES algorithm, which is the target of the photonic side channel attacks from Chap. 4. Then, background information on IBC is given.

2.2.1 The Advanced Encryption Standard

The analyses in Chap. 4 focus on the Advanced Encryption Standard (AES). AES is a symmetric encryption algorithm based on the Rijndael cipher [60]. It is ratified as a standard by the National Institute of Standards and Technology of the USA. AES has a fixed block size of 128 input bits and operates on a 4×4 matrix of bytes, named the state. Depending on the length of the key, which is 128, 192, or 256 bits, the cipher is termed AES-128, AES-192, or AES-256. The algorithm is specified as a number of rounds that transform the input plaintext into the ciphertext. AES consists of 10, 12, and 14 rounds for 128-, 192-, and 256-bit keys, respectively. Each round consists of four operations SubBytes, ShiftRows, MixColumns, AddRoundKey, except for the final round, which skips the MixColumns operation. Additionally, there is an initial AddRoundKey operation before the first round. Algorithm 2.2 shows the sequence of the rounds for AES-128.

Algorithm 2.2 AES-128 Algorithm.

Require: plaintext $m \in \{0, 1\}^{128}$, key $k \in \{0, 1\}^{128}$

Ensure: ciphertext $c \in \{0, 1\}^{128}$

```

1: state  $s \leftarrow m$ 
2:  $s \leftarrow \text{AddRoundKey}(s, k)$ 
3: for  $i = 1 \dots 9$  do
4:    $s \leftarrow \text{SubBytes}(s)$ 
5:    $s \leftarrow \text{ShiftRows}(s)$ 
6:    $s \leftarrow \text{MixColumns}(s)$ 
7:    $s \leftarrow \text{AddRoundKey}(s, k_i)$ 
8: end for
9:  $s \leftarrow \text{SubBytes}(s)$ 
10:  $s \leftarrow \text{ShiftRows}(s)$ 
11:  $s \leftarrow \text{AddRoundKey}(s, k_{10})$ 
12:  $c \leftarrow s$ 
13: return  $c$ 

```

Regarding AES-128, the initial AddRoundKey operation uses the complete secret 128-bit key. Then, each 128-bit round key k_i is derived deterministically from this original secret key with Rijndael's key schedule [60]. During each round of AES-192 and AES-256, also a 128-bit round key is used. The key for the initial AddRoundKey operation consists of the first 128 bits of the secret 192- and 256-bit secret key, respectively. The round key of the first round consists of the remaining bits of the secret key. Regarding AES-192, the second half of this round key is derived with the key schedule, while for AES-256, the key schedule is used only from the second round.

In the AddRoundKey step, each byte of the state is combined with a byte of the round key using the exclusive or operation ($\oplus$). In the SubBytes step, each byte of the state is substituted by an affine transformation of its multiplicative inverse over the Galois field $\mathbb{F}_{2^8}$. This is the only operation that provides non-linearity in the algorithm. Since this deterministic operation is very costly, a precomputed 8-bit lookup table is often used. This so-called S-Box is shown in Table 2.1. The substitution value of a byte $b = b_7|b_6|b_5|b_4|b_3|b_2|b_1|b_0$ with $b_i \in \{0, 1\}$ for all $i \in \{0, \dots, 7\}$ is calculated by deriving the row and column numbers from the byte value. The row number consists of the value of the four Most Significant Bits (msb), i.e., b_7 to b_4 , and the column number consists of the value of the four Least Significant Bits (lsb), i.e., b_3 to b_0 . In the ShiftRows step, each row of the state matrix is shifted to the left by 0, 1, 2 or 3 bytes. In the MixColumns step, the four bytes of each column of the state matrix are combined using an invertible linear transformation, resulting in another four bytes.

Table 2.1 The AES S-Box, used during the SubBytes operation, in hexadecimal representation

	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
1	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
2	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
3	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
4	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
5	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
6	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
7	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
8	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
9	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
a	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
b	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
c	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
d	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
e	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
f	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

The 4 msb of the input byte determine the row and the 4 lsb determine the column. The element accessed by the input is the substitution value and hence, the next state value

The memory access patterns of AES are particularly susceptible to cryptanalysis [23, 85, 134]. Optical emissions related to AES S-Box accesses are also exploited in the photonic side channel attacks in Chap. 4. The algorithm running on our Device Under Attack (DUA) consists of a software AES implementation. To increase the frequency of the execution, only the first AddRoundKey and SubBytes operations were computed on the chip after which the input was reset and the measurement restarted. Table 4.8 shows the assembly code of our SubBytes implementation.

2.2.2 Identity-Based Cryptography from Pairings

IBC was invented by Shamir, who presented the idea of this public-key system in 1984 [162]. However, he could not explain back then how to realize such encryption schemes mathematically. Nearly two decades later, in 2001, Boneh and Franklin showed that elliptic curves and pairings can be used to realize IBE [39].

IBE is a form of asymmetric cryptography where the identity of a user is at once his public key. In any form of communication and cryptography, the sender of a message has to know an information about the receiver which is uniquely assigned to him, such as his email address. The idea of IBE is that this information is sufficient to use cryptography and that no further information such as a dedicated public key is necessary. Therefore, IBE facilitates especially those systems which have to manage large numbers of key material.

A system which implements IBE needs a trusted third party or trusted authority, the so-called Private Key Generator (PKG). Following Boneh and Franklin's initial scheme, four probabilistic polynomial time algorithms are used:

- **Setup:** This algorithm generates all system parameters and the secret master key. Only the PKG knows this master key. The public system parameters include descriptions of the plaintext space $\mathcal{P}$ and the ciphertext space $\mathcal{C}$ and of the set of possible identities.
- **Extract:** The private key of a user is extracted from the secret master key and his identity, i.e., his public key.
- **Encrypt:** The randomized encryption of a message $m \in \mathcal{P}$ is computed based on the public system parameters and the identity of the receiver, i.e., his public key.
- **Decrypt:** The deterministic decryption function outputs a plaintext for given ciphertext $c \in \mathcal{C}$ and the receiver's private key.

In the FullIdent scheme proposed by Boneh and Franklin, both in the Encrypt and Decrypt step a bilinear pairing is computed [39]. During the Decrypt step, the input to the pairing consists of a part of the ciphertext and the secret key of the receiver of that ciphertext. Hence, it has to be ensured that an attacker cannot gain any information about the secret input to a pairing. Attacks on pairings have therefore become a prominent strand within the research on pairings.

An advantage of IBE is the facilitated key management. Certificates are no longer necessary. Furthermore, the sender can send an encrypted message to the receiver

even if the receiver does not know his private key yet. It is likewise possible to provide public keys with a validity period which results in a simple method for key revocation. On the other hand, key escrow is immanent in IBE systems since each private key can be derived from the master key at any time. This is an advantage in the case that each user really trusts the PKG. Otherwise, it is a drawback of such systems. In the case of identity-based signatures, the nonexisting non-repudiation is another disadvantage for the same reason.

Identity-Based Cryptography for Wireless Sensor Networks IBE is especially well suited for those devices with a constrained power supply, such as smartcards and WSNs. On the one hand, WSNs benefit from the non-necessary costly verification of certificates. On the other hand, it is easy to add an additional node since there are no certificates of that node to be verified [50]. Hence, IBE can be used to solve the key distribution problem in WSNs [131]. Another decided advantage is that the existence of a PKG is immanent to WSNs by means of the base station which connects the WSN to other networks. In addition to these advantages, the computation of pairings necessary for IBE is for a given security level more efficient than classical public key cryptography [133]. However, since the PKG can derive all private keys from the master key, in general it is considered a caveat that all users of an IBE system have to trust the PKG.

2.3 Side Channel Attacks

In a cryptographic side channel attack, the attacker captures physical characteristics of the DUA while it performs computations with secret data. She analyzes these characteristics to reveal secret information, such as the secret key of an encryption algorithm. Side channel attacks have been a significant research area since the seminal papers of Kocher in 1996 and 1999, which introduced the timing [106] and the power side channel [107]. Since then, a plethora of other side channels, applications, and analysis methods have been presented. In this section, we provide the necessary background information about side channel attacks and present a selection of relevant work. Side channel attacks are also called passive or non-invasive attacks [35].

The attacker collects a number of so-called traces to reveal the secret information. In the case of power consumption, “a trace refers to a set of power consumption measurements taken across a cryptographic operation” [107]. Thus, each trace consists of a set of side channel leakage values, each related to a distinct point in time. Side channel attacks are divided into univariate and multivariate attacks. Univariate attacks exploit the leakage of only a single point in time [62], while multivariate attacks exploit multiple aspects of the measurements jointly. Side channel attacks combining multiple points of leakage are also called higher-order attacks [58, 107]. Since protected implementations can generally not be successfully attacked with a univariate attack, higher-order attacks are also a means to break countermeasures [97, 115].

Since in real-world applications access to the DUA is often restricted, an attacker cannot get an unlimited number of traces. Template attacks compensate for a smaller number of traces during the attack by adding an additional attack phase in which the attacker has unlimited access to an identical experimental device [47]. During this phase, she can execute arbitrary code on this device and thereby derive templates which model both the signal and the noise. These templates improve the efficiency of the attack in terms of the required number of traces and let the attacker fully utilize the leakage information from each trace.

For the analysis of side channel information, many different distinguishers exist [62, 92]. Distinguishers are the statistical methods which are applied to side channel measurements. However, often the choice of the distinguisher is of minor importance [117].

If a secret key is the target of a side channel attack, it is often not revealed as a whole, but in separate chunks. These separate chunks are called subkeys. If the AES algorithm is the target of the attack, for instance, each key byte is generally revealed independently.

The success rate of an attack can be measured based on its Global Success Rate (GSR) and its Partial Success Rate (PSR). The Global Success Rate is defined as the probability that the complete key is ranked first, i.e., it is the probability of getting the correct value for all key bytes simultaneously [87]. The Partial Success Rate is defined as the probability that the correct subkey is ranked first among all possible subkeys, i.e., the Partial Success Rate (PSR) is the probability of obtaining the correct value, computed independently for each key byte [92]. Since each key byte has its own PSR, often the minimal PSR (min PSR) is used to describe the attack efficiency [170].

Example 2.1 Assume that we have a set of traces and want to attack a 16-byte key. We randomly choose ten subsets of the traces and try to reveal the secret key for each of these subsets. In five cases, we reveal the secret key completely, while in the remaining 5 cases, we do not correctly identify the Least Significant Byte (LSB). Then, the Global Success Rate (GSR) is $\frac{5}{10} = 0.5$, while the PSR is 1 for the first 15 key bytes and 0.5 for the LSB. Thus, min PSR is 0.5. Assume now that we never reveal the secret key completely, but in experiment i , $i \in \{1, \dots, 10\}$, all key bytes except for key byte i are revealed. Then, we have min PSR = 0.9, but GSR = 0.

2.3.1 Timing Attacks

In 1996, a timing attack against several cryptographic algorithms was the first published side channel attack [106]. In a timing attack, the attacker analyzes the time required to perform operations which involve secret key material to extract information about that key material. Often, timing attacks target algorithms which employ an exponentiation with a secret exponent, since the timing strongly depends on the

value of the processed exponent bits in unprotected implementations [151]. Timing attacks can also be conducted remotely, e.g., against network servers [44].

A variant of timing attacks, both local and remote, are cache attacks. In a cache attack, the timing information provides information about cache hits and misses and hence, about the cache entries. Therefore, cache attacks often target algorithms using table lookups such as AES. In 2004, Bernstein conducted a known-plaintext cache timing attack on the OpenSSL AES implementation that uses precomputed tables [23]. He extracted a complete 128-bit AES key. The mathematical analysis of our attack presented in Sect. 4.1 is similar to that analysis. In 2005, Percival revealed an OpenSSL 1024-bit RSA private key by exploiting simultaneous multithreading [139]. This OpenSSL implementation used RSA-CRT for private-key operations. During the attack, 310 out of 512 bits per exponent could be extracted, which is enough for factorizing the modulus.

2.3.2 Power Analysis

The second side channel that was exploited is power analysis. In 1999, Kocher et al. presented Simple Power Analysis (SPA) and Differential Power Analysis (DPA). They define SPA as “a technique that involves directly interpreting power consumption measurements collected during cryptographic operations”. In contrast, DPA does not interpret the traces directly, but uses “statistical functions tailored to the target algorithm” [107]. In this seminal work, the Data Encryption Standard (DES) was attacked. After the publication of these results, countermeasures against power analysis have been developed, e.g., masking and hiding [116]. Masking means to make the processed value uncorrelated to the algorithmic value, while hiding means to make the power consumption uncorrelated to the processed value. However, the analysis methods also evolved, and higher-order attacks against DPA-resistant software and hardware have been published [53, 121]. A few years ago it was even shown that successful SPAs are still possible, despite several implemented countermeasures like message padding [57, 110].

Power analysis attacks also target novel algorithms like PBC. In 2006, both SPA and DPA on the eta pairing over binary fields were presented [103]. The authors suggest randomization and blinding as defense against such attacks, both of which are already used in other algorithms such as RSA. In 2009, a DPA against pairing computations was simulated [66]. It was shown how the secret input point to the Miller Algorithm can be revealed by analyzing a modular multiplication and an addition. In 2013, these results were improved and it was theoretically described how the secret input to the Miller Algorithm can be revealed only by a DPA of a modular multiplication [35].

2.3.3 *Electromagnetic Analysis*

The analysis of electromagnetic radiation as a side channel was already mentioned in the seminal work on power analysis [107]. Electromagnetic emanation is a three-dimensional vector field which changes over time. Instead of observing the near-field emanation of the whole Integrated Circuit (IC), the observation can be restricted to a certain location such as a specific component of the IC. Such localized measurements allow for side channel attacks which exploit location-dependent information leakage, though to a lesser extent than is the case for Photonic Emission Analysis. The level of localization is related to the diameter of the magnetic coil usually used in electromagnetic side channel attacks to acquire the measurements. We refer to Heyszl's PhD thesis for more information on the strengths and limitations of high-resolution measurements for electromagnetic side channel attacks [94].

In 2001, two publications on the electromagnetic (EM) side channel appeared [77, 143]. Both publications stress the advantage that EM analysis has over power and timing analysis: exploitation of locally resolved data leakage. A technical description of EM attacks on smartcards is given in [143]. The authors explain the physics of EM radiation and describe how attacks can be practically realized. However, EM radiation is not analyzed for cryptanalytic purposes in this work. Gandolfi et al. showed by example of a Simple Electromagnetic Analysis (SEMA) against RSA and a Differential Electromagnetic Analysis (DEMA) against the DES how EM attacks can be practically conducted [77]. Both implementations were unprotected against side channel attacks. The authors state that in terms of their experiments, EM analysis outperforms power analysis. In 2002, Agrawal et al. presented a systematic investigation of the EM side channel for Complementary Metal-Oxide-Semiconductor (CMOS) devices [8]. They present concrete results for two different smartcards. They show that EM analysis can even be successful in the presence of power analysis countermeasures. An extended version of their work is also available [9]. In this work, especially higher order attacks and assessment methodologies for these are examined in more detail.

In 2012, location-dependent electromagnetic leakage was successfully exploited in an attack on an elliptic curve scalar multiplication implementation on a Field Programmable Gate Array (FPGA) using a near-field EM probe [95]. The authors scanned the die surface and collected EM traces at every point. They demonstrated that location-dependent leakage can be used in a template attack and countermeasures against system-wide leakage thus can be circumvented.

2.3.4 *Other Side Channels*

In addition to these classical side channels, there are also other sources of side channel leakage which can be exploited.

In 2008, a cold boot attack was presented [88]. The authors show how disc encryption keys which are stored in Dynamic Random-Access Memory (DRAM) can be easily stolen if the attacker has physical access to the computer. Contrary to many other side channel attacks, this work assumes a very realistic attack scenario and this attack is a serious security vulnerability. The attack becomes possible since data stored in DRAM does not vanish instantaneously, but fades away gradually when the power is turned off. By cooling the memory, this process can even be slowed down. This attack is considered to be a side channel attack even though the attackers do not capture physical characteristics of the DUA while it performs computations with secret data. They do, however, exploit the physical implementation of a cryptosystem.

In 2013, the acoustic side channel was presented [79]. The authors show how a 4096-bit RSA key can be extracted by analyzing the acoustic frequency spectrum during decryption. The decryption was performed by GnuPG running on a laptop, while the acoustic emanation was captured only with a plain mobile phone. Thus, this attack was launched with low-cost equipment. Although the acoustic side channel has a very low bandwidth which makes such attacks more difficult to conduct, specially crafted chosen messages lead to more discernible leakage in the presented attack.

2.4 Fault Attacks

The principle of fault attacks is to disturb the computation of cryptographic operations by induction of one or more faults. From the faulty result, the attacker learns information about the internal sensitive data such as the secret key. The first fault attack against a cryptographic algorithm was presented in 1997 [38]. Since then, fault attacks have been applied against various cryptographic algorithms [180] and became a standard tool to facilitate cryptanalysis. The attack assumptions can be described in detailed fault models, which include the location and the timing of the fault, and the number and kind of faults. Nowadays, many techniques exist to induce faults, e.g., clock glitching, power glitching, and laser beams [17]. To thwart countermeasures against fault attacks, even two faults within one computation have been performed [102, 177]. These attacks are often called second-order attacks [63]. More generally, we call a fault attack with more than one fault a higher-order attack.

2.4.1 RSA

The first fault attack against a cryptographic algorithm, known as the Bellcore attack, was presented in 1997 by Boneh, DeMillo and Lipton against the RSA signature scheme [38]. They showed that the RSA modulus can be factorized if an attacker induces a fault into the computation of one of the two parts of the signature generation in case the RSA CRT version is used. With a single faulty signature and a correct signature under the same secret key, an attacker can reveal the secret key.

Since then, RSA has been a popular target for fault attacks. A hardware-based fault attack against the RSA verification process was presented in 2005 [160] and generalized one year later [126]. The attack consists of forcing the attacked cryptographic device to use a slightly modified modulus instead of the original one by inducing transient hardware faults. Since the factorization of the altered modulus is known, the attacker can calculate a new private key so that the device accepts the signature of any arbitrary message signed with this new key. A similar attack was presented against the signature process in 2006 [43]. The authors even show how the full RSA private key can be recovered by corrupting the modulus. Hence, it was concluded that “RSA public key elements also have to be protected against fault attacks”. In the same year, it was even questioned if it is wise to publish public key elements [84]. The authors present an attack against the RSA verification process where they induce not transient, but permanent faults. This allows the forgery of any signature at any time.

Another work that targets the RSA modulus was published in 2012 [123]. The authors developed a purely software-based fault attack against the verification process. They showed that the modulus can be completely replaced when the structures which manage the public key material are attacked. The new modulus can be easily factorized with a high probability. The practicability was demonstrated on a widely deployed conditional access device.

2.4.2 Elliptic Curve Cryptography

Fault attacks have also been presented against Elliptic Curve Cryptography (ECC). In 2000, the first fault attack on elliptic curve cryptosystems was presented [25]. The authors present three ideas for faults attacks, all of which are based on the same idea: when the coordinates of a point are modified by a fault, the new point will not be on the original curve. The curve on which the modified point lies is potentially cryptographically less secure. Hence, the ECDLP might be easier to solve on that curve. The authors stress that this attack might even be possible without any fault induction: if the DUA does not explicitly check whether or not the input point is on the specified curve, a malicious user can just input a point with the desired properties. Later, this work was refined to a more relaxed fault model [52]. Another idea for fault attacks against ECC are sign change attacks [32], in which the signs of intermediate points are changed to facilitate the computation of the secret scalar. It is more difficult to mitigate these attacks, since the modified points still lie on the original curve, so that integrity checks would not detect the fault after such attacks. Sign change fault attacks were also described against PBC [176]. In 2008, an attack tailored to the Montgomery ladder was presented [71]. The authors state that they can reveal the secret scalar with only one or two faults, even in the presence of countermeasures which aim at preventing fault attacks. The attacks also consist in performing operations with the modified point on another curve. In this scenario, the modified point lies on the twist of the original curve.

2.4.3 *Symmetric Cryptography*

Fault attacks were also conducted against symmetric cryptography. Still in 1997, Biham and Shamir described the idea of these kinds of attacks against symmetric cryptographic algorithms [28]. They applied them against the DES and Triple-DES ciphers. Such an attack is called Differential Fault Analysis (DFA), or Differential Fault Attack [140]. The attacker induces faults on the cryptographic primitive level. The assumed fault model gives her partial information about the differences between certain states of the correct and the faulty computations, although she will not know the concrete value of the fault in most scenarios. Since the attacker also knows the correct and faulty ciphertext, and thereby their difference, she can deduce information about the secret key. Small differences in the fault models might crucially affect the capabilities and the complexity of the attacks [28]. Today, there is a wide range of literature on DFA, e.g., [16, 111, 150].

The mathematical security of block ciphers increases with the number of rounds. Hence, another line of research on fault attacks aims at reducing the number of rounds which are actually computed. In [49], the authors reduced the number of rounds of an AES computation to one by attacking several instructions by means of power glitches.

Why Cryptography Should Not Rely on Physical Attack
Complexity

Krämer, J.

2015, XXI, 122 p. 26 illus., 15 illus. in color., Hardcover

ISBN: 978-981-287-786-4