BRNO UNIVERSITY OF TECHNOLOGY

Faculty of Electrical Engineering
and Communication

HABILITATION THESIS

Brno, 2019

Ing. LUKÁŠ MALINA, Ph.D.

BRNO UNIVERSITY OF TECHNOLOGY

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

DEPARTMENT OF TELECOMMUNICATIONS

ÚSTAV TELEKOMUNIKACÍ

APPLIED MODERN CRYPTOGRAPHY ON CONSTRAINED DEVICES

APLIKOVANÁ MODERNÍ KRYPTOGRAFIE NA OMEZENÝCH ZAŘÍZENÍCH

HABILITATION THESIS

HABILITAČNÍ PRÁCE

AUTHOR AUTOR PRÁCE	Ing. Lukáš Malina, Ph.D.

BRNO 2019

ABSTRACT

The habilitation thesis deals with modern public key cryptographic protocols and their effectiveness. The thesis focuses on the design and deployment of advanced asymmetric cryptographic schemes that are suitable for heterogeneous networks with constrained and small devices. The thesis consists of three main parts. The first part contains a description of conventional digital signatures and public key cryptographic schemes with enhanced security features. The second part presents a comprehensive practical assessment of cryptographic schemes implemented on various devices and platforms used in heterogeneous networks. The third part presents three author's security proposals based on advanced cryptographic protocols. The first security system deals with access control and secure authentication based on smart cards. The second proposal provides efficient and secure data transfer with privacy protection between constrained and small devices such as smartphones and small embedded computers. The third proposal is based on a lightweight privacy-preserving ring signature scheme. The third method is suitable for anonymous transactions and e-voting services that run in an environment with constrained devices such as small devices and nodes in Internet of Things.

KEYWORDS

Authentication, Constrained Devices, Cryptography, Digital Signatures, Privacy-preserving protocols, Public key cryptography

ABSTRAKT

Habilitační práce pojednává o moderních asymetrických kryptografických protokolech a jejich efektivitě. Práce se zaměřuje na návrh a nasazení pokročilých asymetrických kryptografických schémat, která jsou vhodná pro heterogenní sítě s omezenými a malými zařízeními. Práce se skládá ze tří hlavních částí. První část obsahuje popis konvenčních digitálních podpisů a asymetrických kryptografických schémat s rozšířenými bezpečnostními vlastnostmi. Druhá část představuje komplexní praktické zhodnocení kryptografických schémat implementovaných na různých zařízeních a platformách používaných v heterogenních sítích. Třetí část uvádí tři autorovy nové návrhy metod zabezpečení založených na pokročilých kryptografických schématech. První navržený systém se zabývá řízením přístupu a bezpečnou autentizací pomocí smart karet. Druhý návrh poskytuje efektivní, bezpečný přenos dat s ochranou soukromí mezi omezenými a malými zařízeními jako jsou chytré telefony a malé vestavěné počítače. Třetí návrh je založen na lehkých kruhových digitálních podpisech poskytující ochranu soukromí. Třetí metoda je vhodná pro služby anonymních transakcí a elektronických voleb, které běží v prostředí s omezenými zařízeními, jako jsou malá zařízení a uzly Internetu věcí.

KLÍČOVÁ SLOVA

Autentizace, omezená zařízení, kryptografie, digitální podpisy, protokoly s ochranou soukromí, kryptografie s veřejným klíčem

MALINA, Lukáš. *Applied Modern Cryptography on Constrained Devices*. Brno, 2019, 164 p. Habilitation thesis. Brno University of Technology, Faculty of Electrical Engineering and Communication, Department of Telecommunications.

Used package `thesis` version 3.03; `http://latex.feec.vutbr.cz`

DECLARATION

I declare that I have written the Habilitation Thesis titled "Applied Modern Cryptography on Constrained Devices" independentlyand using exclusively the technical references and other sources of information cited in the thesis and listed in the comprehensive bibliography at the end of the thesis.

As the author I furthermore declare that, with respect to the creation of this Habilitation Thesis, I have not infringed any copyright or violated anyone's personal and/or ownership rights. In this context, I am fully aware of the consequences of breaking Regulation § 11 of the Copyright Act No. 121/2000 Coll. of the Czech Republic, as amended, and of any breach of rights related to intellectual property or introduced within amendments to relevant Acts such as the Intellectual Property Act or the Criminal Code, Act No. 40/2009 Coll., Section 2, Head VI, Part 4.

Brno . . . .

author's signature

Contents

Introduction

This thesis deals with modern asymmetric cryptographic protocols and methods and their deployment on constrained (resource-limited) and small devices that are often used in heterogeneous networks and Internet of Things (IoT). Internet of Things (IoT) can be defined as a highly interconnected network of various entities such as tags, sensors, smart cards, small devices (i.e., embedded devices, hand-held devices, smart phones), personal computers and servers. Heterogeneous networks and IoT enable us to establish and provide many useful applications, services and systems such as smart homes, access control systems, ID systems, transport applications (e.g., Vehicular Ad hoc Networks), smart metering, smart grid, etc. Nevertheless, a lot of constrained devices and platforms have usually performance and memory issues with public key cryptography and advanced cryptographic constructions that provide enhanced security properties such as privacy protection, non-linkability, zero-knowledge etc. Many advanced cryptographic solutions and protocols based on computationally expensive cryptographic primitives are originally designed for powerful computers and communication nodes. Despite the fact that the application of cryptographic protocols with enhanced security features increases the level of cybersecurity in heterogeneous and IoT networks, it is still a challenge to design secure and efficient advanced cryptographic protocols and methods protecting data and services that are performed on constrained and limited devices.

The purpose of this thesis is to provide the basic overview about modern asymmetric cryptography protocols with focus on advanced digital signature schemes that can be deployed on constrained and small devices. This thesis has three main goals. The first goal is to present a theoretical background which focuses on the description of various digital signature schemes and advanced public cryptographic schemes such as privacy-preserving cryptographic protocols and post-quantum cryptographic protocols. The second goal is to provide the assessment of chosen cryptographic schemes on various entities included constrained and small devices. This part is based on author's results published in impact factor journals and international conferences. The third goal is to present author's results achieved in the field of the design of advanced cryptographic protocols for constrained devices. This part contains three novel conceptions that are published in two impact factor journals and one international conference dedicated to cryptography and security.

The thesis is divided into three parts that deal with described goals. Chapter 2 focuses on the pedagogical goal and contains the theoretical background and state of the art schemes. Chapter 3 contains practical results and a detailed assessment of cryptographic schemes. Chapter 4 is dedicated to a scientific part and presents author's scientific work. The overview of this thesis is described in the Chapter 1.

1 Thesis Overview

This chapter provides an overview of this thesis. Firstly, Section 1.1 presents the scope of the thesis and research motivation. Secondly, the thesis objectives are described in Sec. 1.2. Section 1.3 presents the contribution of the thesis and relation to author's publications. Finally, the organization of this thesis is outlined in Sec.1.4.

1.1 Scope and Research Motivation

The scope of this work involves advanced public key cryptographic schemes based on modern constructions such as zero-knowledge and sigma protocols. The thesis focuses on digital signature schemes, privacy-preserving schemes such as group signatures and post-quantum cryptographic schemes. On one hand, these schemes provide beneficial and unique security properties such as privacy protection, data pseudonymity, one group key and zero-knowledge properties. On the other hand, the constructions of these schemes consist of several math and cryptographic operations. These schemes are usually more computationally expensive and memory demanding than conventional cryptographic schemes with fewer security properties. Therefore, the implementation and usage of advanced cryptographic protocols with more security properties on constrained platforms and devices could be a challenge. This thesis deals with the design of new efficient security conceptions based on advanced public key cryptographic protocols that can be suitable for constrained platforms. The thesis also provides the assessment of various cryptographic schemes and protocols on chosen constrained platforms and devices that often appear in heterogeneous networks such as IoT.

1.2 Thesis Objectives

The objectives of the thesis are as follows:

- The first goal of the thesis is to provide the basic theory regarding public key cryptography assumptions and primitives, digital signature schemes, advanced digital signatures providing privacy protection, and post-quantum public key cryptographic schemes.
- The second objective of the thesis is to present the detailed assessment of the conventional and advanced cryptographic schemes on constrained and small devices such as microcontrollers, smart cards, mobile devices and small computers.
- The third goal of the thesis is to propose novel cryptographic methods and systems that are more efficient in comparing with related work and could be

deployed in various IoT and heterogeneous networks using constrained and small devices.

1.3 Relation to Author's Publications and Contribution

This thesis presents author's scientific work and results published since 2015. The main results and proposals presented in the thesis have been published in various international journals with impact factors (e.g., [12, 14, 16, 17, 9]) and international journals and conferences dedicated to the cryptography, security and computer networks (e.g., [27], [2], [15, 10, 20, 11, 22, 8, 13, 7]).

In addition, the author is also a main author or co-author of various publications related to other cryptography and security topics such as attribute authentication, security in optical networks and side channel cryptanalysis. These results are published in international journals with impact factors [19, 6, 4, 26, 25] and various conferences [21, 18, 99, 108, 98, 5, 4, 1, 3, 24].

None of the results or proposed solutions presented in this thesis were published in the author's Ph.D. thesis or any past author's theses. Nevertheless, author's significant results (e.g., [23, 143, 145, 142, 146, 147, 148, 141, 144, 104, 101, 97, 100]) from his Ph.D. study are used as foundation for new results and proposals used in this work. For example, the sections 2.3.1 and 4.2 devoted to group signatures are related to author's Ph.D. thesis [139]. The Ph.D. thesis focused solely on pairing-based group signatures protocols. In this habilitation thesis, the section 4.2 presents an enhanced system built on the results and the theory background of the Ph.D. thesis but which provides a new contribution. This new system is also based on group signatures providing the natural revocation but without pairing operations in the signing phase. The former proposed protocol presented in [139] performs 2 expensive pairing operations in the signing phase. Therefore, the novel design presented in this thesis could be more suitable for constrained devices. In addition, the performance and security analysis of the system is provided in this thesis. Moreover, the habilitation thesis presents other two novel proposals based on advanced cryptographic constructions. Both proposals are also suitable for constrained and small devices.

The contribution of this thesis is threefold:

- **Pedagogical contribution**: Chapter 2 describes various digital signature schemes with underlying cryptographic methods and primitives. Further, privacy-preserving cryptographic schemes including anonymous digital signatures such as group signatures and ring signatures are introduced and explained. Moreover, the brief overview of Post-Quantum Cryptography (PQC) is provided. This theoretical part of the thesis offers foundation and concrete examples regarding conventional and advanced public cryptography with focus on digital signatures and authentication.

- **Practical contribution**: Chapter 3 contains practical results and the detailed assessment of cryptographic schemes on various small and constrained devices often used in IoT and heterogeneous networks. The obtained results and lessons learned can help with future research and the practical deployment of classical and advanced cryptographic schemes on small and constrained devices.
- **Scientific contribution**: Chapter 4 presents author's original scientific work. Three novel proposed systems and methods are based on modern cryptographic constructions that are presented in the theoretical part. The common goal of all proposals described in sections 4.1, 4.2 and 4.3 is to provide efficient and practical solutions with advanced security properties in environment using constrained and small devices. The proposals have been presented in journals with an impact factor [12], [145] and the international conference dedicated to cryptography [13].

1.4 Thesis Organization

This thesis contains 4 chapters and is organized as follows:

- Chapter 1 presents the scope of the thesis in Sec. 1.1, the thesis objectives in Sec. 1.2, the contribution of the thesis with relation to author's publications in Sec. 1.3, and thesis organization in Sec. 1.4.
- Chapter 2 contains the theoretical background that presents a state of the art cryptographic schemes. The chapter introduces underlying cryptographic methods and primitives in Sec. 2.1, conventional signature schemes in Sec. 2.2, and privacy-preserving cryptographic schemes including anonymous digital signatures such as group signatures and ring signatures in Sec. 2.3. The theoretical evaluation of chosen digital schemes is presented in Sec. 2.4. Finally, Section 2.5 introduces the brief overview of PQC.
- Chapter 3 contains the assessment of chosen cryptographic primitives and schemes on various constrained devices. Section 3.1 briefly introduces the security and privacy requirements and devices in IoT and heterogeneous networks. Further, the chapter discusses the feasibility of common, privacy-preserving and post-quantum cryptography on constrained devices in sections 3.2, 3.3 and 3.4. Section 3.5 concludes the chapter and presented results.

- Chapter 4 introduces author's novel proposals of security systems based on advanced public key protocols that are suitable for constrained devices. The chapter presents three proposals. Section 4.1 presents an authentication system based on a zero-knowledge protocol implemented on smart cards. Section 4.2 presents a novel privacy-preserving cryptographic conception based on group signatures that is suitable for small devices. Section 4.3 introduces a novel security conception for anonymous transactions based on a lightweight privacy-preserving ring signature scheme.
- Chapter 5 concludes this thesis.

2 Cryptographic Background

This chapter presents the cryptographic schemes and constructions that are used in this thesis. The first section introduces basic notation, general assumptions and an introduction into underlying cryptographic methods, elliptic curves and bilinear pairings. The second section presents conventional digital signature schemes. The third section focuses on privacy-preserving cryptographic schemes including anonymous digital signatures such as group signatures and ring signatures. Finally, Post-Quantum Cryptography (PQC) is introduced.

The performance assessment of described cryptographic schemes is presented in the following Chapter 3. Then, Chapter 4 outlines three author's proposals that are based on some constructions presented in this background.

2.1 Background and Preliminaries

This section introduces basic notation, general assumptions and introduction into elliptic curves and bilinear pairings. Further, the section presents the basic description of underlying cryptographic methods and primitives such as commitment schemes and Zero-Knowledge (ZK) proofs that are used in modern cryptography.

2.1.1 Common Symbols and Basic Notation

In the following text, the common symbols and basic notation that are used in conventional cryptography and also in this thesis are defined as follows:

- E - elliptic curve.
- $E(\mathbb{F}_q)$ - elliptic curve over finite field $\mathbb{F}_q$.
- $e(,)$ - pairing operation.
- $\mathbb{F}_q$ - finite field.
- g - generator of a finite cyclic group.
- $\mathbb{G}$ - finite cyclic group.
- $\gcd()$ - greatest common divisor.
- gpk - group public key.
- $H()$ - hash function.
- l - security level in bits.
- m - message.
- M - message.
- n - modulus.
- p - prime number.
- PK - public key.

- SK - secret key.
- t - runtime of an operation.
- q - prime number.
- $\underline{x}$ - bit encoding of element x.
- $\mathbb{Z}$ - ring of integers.
- σ - signature.
- ψ - computable isomorphism.
- $\times$ - elliptic curve scalar (point) multiplication.
- $\in_R$ - randomly chosen in ... (e.g., in finate field).

Special symbols and enhanced notation for concrete cryptographic schemes are then defined within the descriptions of schemes in the next sections and chapters.

2.1.2 General Assumptions and Problems

Many asymmetric cryptographic schemes such as authentication schemes and digital schemes are based on computational hardness assumptions (problems). The following text introduces the definition of basic problems that are used in cryptography:

- **Integer Factorization Problem** (IFP): Given a large integer n, it is hard to compute the prime factorization of this number that is defined as $n = {p_1}^{e_1}{p_2}^{e_2}...{p_k}^{e_k}$ where p_i is a distinct prime and e_i is integer ≥ 1.
- **RSA Problem** (RSAP): Given a modulus n of unknown factorization, an integer c and a positive integer e such that $\gcd(e, (p-1), (q-1)) = 1$ where primes p, q define the modulus ($n = pq$), it is hard to find an integer m such that $m^e \equiv c(\mod n)$.
- **Strong RSA Problem** (SRSAP): Given a modulus n of unknown factorization and an integer c, it is hard to find integers (M, e) such that $M^e \equiv c(\mod n)$.
- **Discrete Logarithm Problem** (DLP): Given a finite cyclic group $\mathbb{G}$ of order q, a generator g and an element $c \in \mathbb{G}$, it is hard to compute x such that $g^x = c$.
- **Elliptic Curve Discrete Logarithm Problem** (ECDLP): Let E be an elliptic curve over finite field $\mathbb{F}_q$. Given $P, Q \in E(\mathbb{F}_q)$, it is hard to compute a such that $Q = a \times P$.
- **Decisional Diffie-Hellman Problem** (DDHP): Given a finite cyclic group $\mathbb{G}$ of order q, a generator g, and elements $a = g^x$, $b = g^y$, $c = g^z$ with uniformly and independently chosen $x, y, z \in \mathbb{Z}_q$, it is hard to decide if $xy \equiv z(\mod q)$.
- **Computational Diffie-Hellman Problem** (CDHP): Given a finite cyclic group $\mathbb{G}$ of order q, a generator g, and elements $a = g^x$, $b = g^y$ with uniformly and independently chosen $x, y \in \mathbb{Z}_q$, it is hard to compute $c = g^{xy} \mod q$.

There are more variations of assumptions, e.g., Decisional Linear (DLIN) assumption, external Diffie–Hellman (XDH) assumption, Inverse Computational Diffie-Hellman Problem (Inv-CDHP), and Bilinear Diffie-Hellman Problem (BDHP). These assumptions can be used in advanced cryptographic schemes.

2.1.3 Elliptic Curves

Elliptic Curves (EC) are employed in many public key cryptography schemes such as Elliptic Curve Digital Signature Algorithm (ECDSA), Edwards-Curve Digital Signature Algorithm (EdDSA) and Elliptic Curve Diffie-Hellman (ECDH) protocol. An elliptic curve E over a finite field $\mathbb{F}_q$ is an algebraic curve and any elliptic curve can be defined by the generalized Weierstrass equation:

$$y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6, \tag{2.1}$$

where $a_1, a_2, a_3, a_4, a_6 \in \mathbb{F}_q$ are the coefficients of the curve and are constants. $E(\mathbb{F}_q)$ denotes the set of values (x, y) with $x, y \in \mathbb{F}_q$ which satisfies Equation 2.1, along with a "point at infinity" denoted O. Points (x, y) are known as affine coordinates and there are only finitely many pairs (x, y) with $x, y \in \mathbb{F}$ where $\mathbb{F}$ is a finite field. The points of $E(\mathbb{F}_q)$ have a group structure under an explicitly defined additive group law. An elliptic curve offers operations such as adding points and doubling (or tripling) one. The results are the points that belong to the curve itself. Adding two points on an elliptic curve in the Weierstrass form requires 2 multiplications, 1 squaring, and 1 inversion in the field. The time needed for finding inversion is estimated between 9 and 40 times slower than multiplication [68]. Squaring takes about 0.8 the time of multiplication.

Elliptic curves can be expressed by different forms (shapes) such as the short Weierstrass curve, the Doche-Icart-Kohel curve, the Koblitz curve, the Edwards curve, the twisted Edwards curve, the Montgomery curve, the Barreto-Naehrig curve, the Jacobian curve, the Jacobi curve and the Hessian curve.

For example, the short Weierstrass curve [202] is an elliptic curve defined as:

$$y^2 = x^3 + ax + b, \tag{2.2}$$

where $a_1 = a_2 = a_3 = 0, a_4 = a$ and $a_6 = b$ in Equation 2.1 and with $a, b \in \mathbb{F}_q$. This form can be used only in a field with $p \neq 2, 3$ and, normally, it is used over the prime field $\mathbb{F}_p$. The short Weierstrass curve is probably the most deployed shape and defines curves such as NIST curves (p-224, p-256, p-384), BN(2,254) curves, the secp256k1 curve, the brainpoolP256t1 curve, the brainpoolP384t1 curve, the ANSSI FRP256v1 curve. ECDH and ECDSA schemes use these curves.

The twisted Edwards curve with coefficients a and d is defined as:

$$ax^2 + y^2 = 1 + dx^2y^2, \tag{2.3}$$

where $a, d \in \mathbb{F}_{p^m} \setminus \{0\}$ with $p \neq 2$. The twisted Edwards curve is employed in EdDSA.

The basic operations (addition, doubling) on various curves differ in the number of underlying arithmetic operations in the field (multiplication, squaring, inversion). The most expensive operation is elliptic curve scalar multiplication (denoted as ×) that is carried out by point addition and point doubling operations and its efficiency is related to the efficiency of these operations. Thus, various elliptic curves may offer different performance characteristics. Furthermore, the affine representation of EC is often replaced with the form in projective coordinates in order to be more efficient during finding inversions. More information about ECC can be found in [68], [202] and [2].

2.1.4 Bilinear Pairing

A pairing operation is defined by mapping between two elements of cryptographic groups $\mathbb{G}_1$ and $\mathbb{G}_2$ and its output is an element of the third cryptographic group. The pairing operation can be also called as a bilinear map. The bilinear pairing schemes may use two types of notation: additive and multiplicative. The multiplicative notation is also used in the descriptions of schemes presented in this thesis. The bilinear pairing operation is defined as follows:

- $\mathbb{G}_1$ and $\mathbb{G}_2$ are two multiplicative cyclic groups.
- g_1 is a generator of $\mathbb{G}_1$ and g_2 is a generator of $\mathbb{G}_2$.
- $\mathbb{G}_T$ is a multiplicative cyclic group of order p.
- ψ is a computable isomorphism from $\mathbb{G}_2$ to $\mathbb{G}_1$.
- e is a computable bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ with the following properties:
  - Bilinearity: for all $g_1 \in \mathbb{G}_1, g_2 \in \mathbb{G}_2$ and $a, b \in \mathbb{Z}, e(g_1^a, g_2^b) = e(g_1, g_2)^{ab}$.
  - Non-degeneracy: $e(g_1, g_2)$ is a generator of the group $\mathbb{G}_T$.
- e is distributive $e(g_1, g_2)^{a+b} = e(g_1, g_2)^a e(g_1, g_2)^b$.

If $\mathbb{G}_1 = \mathbb{G}_2$ then the pairing is symmetric. If $\mathbb{G}_1 \neq \mathbb{G}_2$ then the pairing is asymmetric. Pairing operations are usually implemented by the Weil pairing or the Tate pairing but there are more variants such as Ate, Eta, O-Ate pairings. Pairing-based schemes are based on pairing-friendly elliptic curves such as supersingular curves ($y^2 = x^3 + x$), the Barreto and Naehrig curve construction, the Freeman curve construction and other specialized curves. The pairing operations use the Miller algorithm or the Miller loop [151] that is the straightforward double-and-add algorithm for elliptic curve point multiplication.

Bilinear pairing operations are used in many cryptographic protocols such as short signatures, group signatures, ring signatures, attribute authentication, signcryption or three-party one-round key agreement protocols. Pairing-based constructions may reduce the hardness of one problem in one group to an easier problem in another group (the so-called gap group). The security of these schemes is usually based on another problem which still remains hard. The benefits of Pairing-based Cryptography (PBC) are new constructions (e.g., three-party one-round key agreements) and short lengths of signatures and other cryptographic parameters. PBC schemes are based on assumptions such as Bilinear Diffie-Hellman Problem (BDHP) and its variants. Nevertheless, the pairing operation is computationally more expensive than the modular exponentiation of big integers on many platforms (approximately 10 times more, see the paper [173]). Therefore, computationally expensive PBC schemes are not attractive for constrained devices such as sensors and smart cards. More details about the pairing operation efficiency, pairing types and their security can be found in [59].

2.1.5 Commitment Schemes

Commitment schemes allow one to commit and hide a chosen value and are the parts of various signature and authentication cryptographic schemes. In real life, commitment schemes are similar to locked boxes with secret documents inside. The document remains secret until the key of the box is provided to an opener. Commitment schemes consist of the Commit phase and the Reveal phase. In the Commit phase, a sender chooses the input value and computes the output (a commitment) value that is sent to a receiver. The input value is hidden to others including the receiver and it cannot be changed after the commitment phase. In the Reveal phase, the input value is revealed and checked that it has been used in the Commit phase.

Commitment schemes provide hiding and binding properties and can be interactive or non-interactive. In practice, commitment schemes could be perfectly binding and computationally hiding or computationally binding and perfectly hiding. For example, the Pedersen commitment scheme [163] is perfectly (unconditionally) hiding and computationally binding and consists of these steps:

- **Setup**: A receiver chooses large primes p and q such that q divides $p-1$, and the generator g of the order-q subgroup of $\mathbb{Z}_p^*$. Then, a random secret value a is chosen from $\mathbb{Z}_q$ and a public value is computed as

  $$h = g^a \mod p. \tag{2.4}$$

  The values p, q, g, h are public and a is secret.

- **Commit**: A sender chooses a random value $r \in \mathbb{Z}_q$ and commits $x \in \mathbb{Z}_q$ by

$$c = g^x h^r \mod p. \tag{2.5}$$

  Then, the sender sends c to the receiver.
- **Reveal**: To open the commitment, the sender reveals x and r, and the receiver verifies that

$$c = g^x h^r \mod p. \tag{2.6}$$

The hiding and binding properties of the Pedersen commitment scheme are defined as follows:

- The **hiding** property: for a receiver it is statistically indistinguishable to get x due to the random r that randomizes the commitment c.
- The **binding** property: for a sender it is difficult to find such (x', r') that makes $c = g^{x'} h^{r'} \bmod p$ equal to c computed by Equation 2.5. The sender has to solve the discrete logarithm problem of h to compute (x', r') which is computationally infeasible.

There are various versions of commitment schemes (e.g., Discrete Logarithm Commitment, ElGamal Commitment, and Pedersen Commitment), and more details are in [75]. Commitment schemes can be deployed in zero-knowledge proofs, signature schemes, secret sharing, coin flipping and other cryptographic protocols.

2.1.6 Zero Knowledge Proofs

A Zero-Knowledge (ZK) proof (or a ZK protocol) is a cryptographic method by which a prover can prove a verifier that he/she knows a value without disclosing any additional information besides the fact that he/she knows the value. Only the prover with the secret information is able to correctly complete the ZK protocol (construct the valid proof). The proof does not leak any information besides boolean information (a true/false statement). Further, the verifier without the knowledge of the prover's secret is not able to prove the statement to another party and impersonates the prover. The basic three properties of ZK protocols are defined as follows:

- The **zero-knowledge** property: the proof does not leak any information leading to obtain/restore the secret value by verifiers and observers.
- The **completeness** property: an honest verifier who follows the protocol properly is always convinced by an honest and prover who provides the valid proof.
- The **soundness** property: no one who does not know the secret value is able to convince an honest verifier with non-negligible probability. A dishonest prover is not able to successfully prove a false statement.

There are several variants of ZK protocols such as Honest Verifier Zero-Knowledge protocols (HVZK), Computational Zero-Knowledge protocols (CZK), Statistical Zero-Knowledge protocols (SZK), Perfect Zero-Knowledge protocols (PZK), Interactive Zero-Knowledge protocols (IZK), Non-Interactive Zero-Knowledge protocols (NIZK). These protocols have different constructions, parameters and are based on various assumptions and hard problems. More details about variants of ZK protocols can be found in [75], [92] and [169].

ZK protocols are useful as fundamental blocks for advanced authentication and signature schemes, e.g., attribute authentication, group signatures and other privacy-enhancing cryptographic protocols that are described in Section 2.3.

2.1.7 Fiat-Shamir Heuristic

The Fiat–Shamir heuristic (transformation) [84] is a cryptographic mechanism that transforms an interactive proof of the knowledge construction (e.g., an identification scheme) into a non-interactive proof of the knowledge construction (e.g., a digital signature scheme). This transformation is often used in group signature schemes.

2.2 Conventional Digital Signature Schemes

This section describes chosen conventional digital signature schemes such as Schnorr signatures, Elliptic Curve Digital Signature Algorithm (ECDSA), Edwards-Curve Digital Signature Algorithm (EdDSA), the Rivest, Shamir, Adleman scheme (RSA) and Rabin scheme. These schemes are used in this thesis as basic underlying primitives in the proposals or are evaluated on some constrained devices in the following chapters.

2.2.1 Schnorr Signature Scheme

The Schnorr identification scheme proposed in 1991 [172] provides a zero knowledge proof method. A prover can convince a verifier that he/she knows a secret value without disclosing this secret (the verifier does not need this value). The math description of the Schnorr identification scheme is depicted in Figure 2.1 where p is a large prime, g is the generator of group $\mathbb{Z}_q$, and $r \in_R \mathbb{Z}_q$ denotes a value r randomly chosen in the group $\mathbb{Z}_q$ (an integer less than q). The Schnorr identification scheme is the part of ISO/IEC 9798-5:2009 that specifies entity authentication mechanisms using zero-knowledge techniques. The Schnorr identification scheme is a Honest Verifier Zero-Knowledge protocol.

Prover **Verifier**

Setup phase

g, p, q

Private key: $w \in_R \mathbb{Z}_q$
Public key: $c = g^w \bmod p$

$$\xrightarrow[\text{Identification phase}]{\quad c \quad}$$

$r \in_R \mathbb{Z}_q$, $\bar{c} = g^r \bmod p$

$$\xrightarrow{\quad \bar{c} \quad}$$

$$\xleftarrow{\quad e \in_R \mathbb{Z}_q \quad}$$

$z = (r - ew) \bmod q$

$$\xrightarrow{\quad z \quad}$$

Check: $\bar{c} \overset{?}{\equiv} g^z c^e \pmod{p}$

Fig. 2.1: Schnorr identification scheme.

The Schnorr signature scheme is constructed by applying the Fiat-Shamir transformation on the Schnorr identification scheme. The Schnorr signature scheme is the part of ISO/IEC 14888-3:2016 that specifies digital signatures with appendix - discrete logarithm based mechanisms. The Schnorr signature scheme is depicted in Figure 2.2 where H is a secure hash function. The usual size of the modulus q is 256 bits and the size of p is 2048 bits or higher. Assuming the hardness of the Discrete Logarithm Problem (DLP), the scheme is secured in the random oracle model. The depicted versions are based on Digital Signature Algorithm (DSA) parameters. The provers and signers have to compute one modular exponentiation and one multiplication. The Schnorr signature and identification schemes can be also based on elliptic curves and use the hardness of the Elliptic Curve Discrete Logarithm Problem (ECDLP).

There are several variants and extensions of Schnorr signature schemes such as Elliptic Curve-Schnorr scheme, EdDSA, Elliptic Curve-Schnorr Digital Signature Algorithm (EC-SDSA), Elliptic Curve-Full Schnorr Digital Signature Algorithm (EC-FSDSA) and others that differ in components, the signature size and arithmetic operations. The perspective signature scheme EdDSA is described in the following subsection.

Signer **Verifier**

Setup phase

g, p, q

Private key: $w \in_R \mathbb{Z}_q$
Public key: $c = g^w \bmod p$

$$\xrightarrow{\qquad c \qquad}$$

Signature phase

$r \in_R \mathbb{Z}_q$, $\bar{c} = g^r \bmod p$
$e = H(c, \bar{c}, M)$, $z = (r - ew) \bmod q$

$$\xrightarrow{\qquad z, e, M \qquad}$$

Restore: $\bar{c}_v = g^z c^e \pmod p$
$e_v = H(c, \bar{c}_v, M)$
If $e_v = e$ then the signature is valid.

Fig. 2.2: Schnorr signature scheme.

2.2.2 Edwards-Curve Digital Signature Algorithm

Edwards-curve Digital Signature Algorithm (EdDSA) is a Schnorr-type signature scheme based on elliptic curves. EdDSA was designed as a fast digital signature scheme by Daniel J. Bernstein *et al.* [41] in 2012 and described in the RFC 8032 document in 2017.

J. Bernstein *et al.* recommend to use Curve25519 that is bi-rationally equivalent to the twisted Edwards curve. Nonetheless, the RFC 8032 document introduces more variants. Curve25519 is the Montgomery curve with the quadratic extension of the prime field defined by prime $q = 2^{255} - 19$. The curve offers 128-bit security and serves as an alternative to the NIST's P-256 curve. The EdDSA signature scheme with SHA-512/256 and Curve25519 is named as Ed25519. The scheme employs methods for encoding and parsing integers and curve points. For example, b-bit encoding of each element (x, y) on a curve E as the b-bit string $(\underline{x},\underline{y})$.

EdDSA consists of the Setup, Signing, and Verification phases that are defined as follows:

- **Setup**: EdDSA uses eleven parameters such as an integer $b \geq 10$, a cryptographic hash function H producing $2b$-bit output, a prime power q congruent to 1 modulo 4, a $(b1)$-bit encoding of elements of the finite field $\mathbb{F}_q$, an odd prime l defined as $lB = 0$ and $2^c l = \#E$ where c is an integer $\{2,3\}$, B

is a base point and other values that are specified in [42]. An EdDSA private key is a b-bit string k that is uniformly chosen in random. The hash $H(k) = (h_0, h_1, ..., h_{(2b-1)})$ determines an integer $s = 2^n + \sum_{c \leq i \leq n} 2^i h_i$. The s value then determines the multiple $A = [s] \times B$. The EdDSA public key $\underline{A}$ is a curve point encoded in b bits.

- **Signing**: a signature generation on the message M using the private key k consists of three steps: (1) compute $r = H(h_b, ..., h_{2b-1}, M)$; (2) compute $R = [r] \times B$; (3) compute $S = (r + H(\underline{R}, \underline{A}, M)s) \mod l$. The signature of M is the $2b$-bit string $(\underline{R}, \underline{S})$.
- **Verification**: The signed message M is verified by using the public key as follows. The verifier parses the parameters and checks $2^c SB = 2^c R + H(\underline{R}, \underline{A}, M)A$ in E. The verifier rejects the signature if the parsing fails or the previous equation does not hold.

EdDSA chooses the nonce r deterministically as the hash of the private key and the message in order to enhance the security of the scheme. Therefore, the signing phase of EdDSA does not need a random number generator. Hence, there is no danger of revealing the private key by a broken random number generator that is used to make a signature as can be in ECDSA.

The Ed25519 scheme is included in many current cryptographic libraries such as OpenSSH, wolfSSL, NaCl, Boran and others. Deterministic EdDSA/Ed25519 schemes are considered as more secure than ECDSA (described in the following subsection) and due to their computational efficiency and small signatures and keys can be attractive for deployment on constrained devices.

2.2.3 Digital Signature Algorithm and Elliptic Curve Digital Signature Algorithm

Digital Signature Algorithm (DSA) standardized in FIPS-186-4 and Elliptic Curve Digital Signature Algorithm (ECDSA) [115] standardized in X9.62 are two digital signature schemes that provide data integrity and verifiable authenticity. ECDSA is the variant of DSA that uses elliptic curves. Both DSA and ECDSA need to produce a fresh random value during each signature generation. The need of a secure source of randomness can be an obstacle for deployment of these schemes on various constrained devices such as smart cards, sensors and embedded systems.

ECDSA consists of the Setup, Signing, and Verification phases that are defined as follows:

- **Setup**: ECDSA uses parameters such as an elliptic curve base point G, an integer n order of $\mathbb{G}$, and a cryptographic hash function H. An ECDSA private key d_A is randomly selected in the interval $[1, n-1]$. The ECDSA public key is a public key curve point Q_A that is computed as $Q_A = d_A \times G$.

- **Signing**: a signature generation on the message M using the private key d_A consists of the following steps: (1) compute $e = H(M)$; (2) Let z be the L_n leftmost bits of e, where L_n is the bit length of the group order n; (3) Select a secure random integer k from the interval $[1, n-1]$; (4) The signer computes the curve point as $(x_1, y_1) = k \times G$; (5) The signer computes $r = x_1 \mod b$ (if $r = 0$ then back to Step 3); (6) The signer computes $s = k^{-1}(z + rd_A) \mod n$ (if $s = 0$ then back to Step 3). The signature is the pair (r, s).
- **Verification**: The signed message M with the signature (r, s) is verified by using the public key Q_A as follows. The verifier firstly checks whether Q_A is a valid curve point by checking that Q_A is not equal to the identity element O, lies on the curve and that $n \times Q_A = O$. Then, the verifier checks that r and s are integers in the interval $[1, n-1]$, otherwise rejects the signature. Further, the verifier calculates $e = H(M)$ and defines z that is the L_n leftmost bits of e. Then, the verifier computes $w = s^{-1} \mod n$, $u_1 = zw \mod n$ and $u_2 = rw \mod n$ and restores $(x_1, y_1) = u_1 \times G + u_2 \times Q_A$. If $(x_1, y_1) = O$ then the signature is rejected. Finally, the verifier checks whether $r \equiv x_1(\mod n)$, otherwise, the signature is rejected.

The size of an ECDSA public key should be chosen $2l$ where l is the security level, e.g., if $l = 80$ bits then the ECDSA public key should be 160 bits. Nonetheless, the DSA public key should be long at least 1024 bits for the same security level. Therefore, the ECDSA scheme is considered as more efficient in memory than DSA. The signature size is the same for both DSA and ECDSA and is defined as $4l$ bits, i.e., 320 bits for the 80-bits security level. Both ECDSA and DSA rely on a random generator during the signing phase. This could cause a security flaw in case of using a weak random generator. Nevertheless, the RFC 6979 document defines a deterministic digital signature generation. Further, EdDSA solves this problem by avoiding the random generator during signing.

2.2.4 RSA Signature Scheme

The cryptosystem RSA (named by initials of surnames of Ron Rivest, Adi Shamir, and Leonard Adleman) provides the encryption and signing of messages. Both variants are based on public and private keys. The encryption (verification) key is public and it is different from the decryption (signing) key that is secret (private). A private key size and a modulus size are chosen from 1024 bits to 4096 bits. Nevertheless, the public key is usually chosen small in order to improve the performance of encryption and verification, most commonly $e = 2^{16} + 1 = 65537$ or $e = 3$. The security of the RSA scheme is based on the Integer Factorization Problem and also on the RSA problem.

The variant of the RSA signature algorithm consists of the Setup, Signing, and Verification phases that are defined (without padding) as follows:

- **Setup**: Two distinct prime numbers p and q are chosen at random. The modulus n is computed as $n = pq$ and is the part of both keys. Then, the Euler's function $\phi(n) = (p-1)(q-1)$ is computed and the public exponent is chosen as $1 < e < \phi(n)$ and as $\gcd(e, \phi(n)) = 1$. The private exponent is derived as $d \equiv e^{-1} (\mod \phi(n))$. The private key is d and $p, q, \phi(n)$ must remain secret. The public key is (n, e).
- **Signing**: Given the hash of a message $h(M)$, the signature is computed as $s = h(M)^d (\mod n)$.
- **Verification**: Given the public key (e, n), the message M and the signature s, the verifier checks whether $s' = h(M)^e (\mod n)$ is equal to the origin signature value s.

Using RSA encryption and signatures without padding in a plaintext may suffer by several attacks. Therefore, RSA adds a form of structured and randomized padding into the value m before processing it. The RSA verification by $e = 3$ could be very fast but the signing phase remains less efficient than EdDSA or ECDSA. RSA is implemented on most smart card platforms and several constrained devices but often only in 1024-bit and 2048-bit versions that are not recommended by NIST.

2.2.5 Rabin Signature Scheme

The Rabin cryptosystem [164] is essentially a special version of RSA with the encryption key $e = 2$, and it is secure under the integer factorization problem. The Rabin cryptosystem is based on factoring $n = pq$, where n is a public key and p and q are private keys. A message M is encrypted as $C = M^2 \text{mod } n$. The decryption process produces four possible roots of C computed by using the Chinese Remainder Theorem (CRT) and $\sqrt{C} \bmod p$ and $\sqrt{C} \bmod q$. The integer C is called a quadratic residue. A variant of the signature algorithm consists of three phases: Key generation, Signing, and Verification that are defined as follows:

- **Setup**: A signer (S) chooses primes p, q each of size approximately $k/2$ bits, and computes the modulo $n = p \cdot q$. Then, S chooses a random $b \in \mathbb{Z}_n$. The public key is (n, b) and the private key is (p, q).
- **Signing**: To sign a message M, the signer chooses random padding U and calculates $H(M, U)$. In case that $H(M, U)$ is not a square modulo n, the signer chooses a value U until he/she finds the square modulo n. Then, the signer computes the value x which solves the equation $x^2 = H(M, U) \mod n$. The

signature on M is the pair (U, x).

- **Verification**: Given a signature (U, x) on the message M, the verifier calculates $h = x^2 \mod n$ and $h' = H(M, U)$ and checks whether $h = h'$. If the values are equal, the signature on the message is valid, otherwise, the signature is rejected.

The main benefit of the Rabin cryptosystem is that encryption (verification) computes only one squaring in mod n. The decryption (signing) is more expensive because of computing the quadratic residue. This operation is as expensive as one exponentiation. As the verification is very efficient, this scheme can be attractive for constrained devices. The Rabin cryptosystem is also used in our proposal of an efficient ring signature scheme, see Section 4.3.

2.3 Privacy-Preserving Cryptographic Schemes

Privacy-preserving cryptographic schemes are usually constructed with some advanced security properties (e.g., anonymity, unlinkability, untracebility) in order to protect user privacy (i.e., user's identity, user's vital data). This section introduces the basic privacy-preserving cryptographic methods that can be defined as follows:

- **Anonymous Digital Signatures (ADS)** - ADS enable signers to sing the message without revealing user's identity. Common digital signature schemes such as RSA, DSA, ECDSA are usually linkable and traceable by user public keys. In common signature schemes, a verifier needs a public key that is usually bound to the user identity in order to verify the signed message from the concrete signer. Digital signature schemes, which do not use a user identity/user public key in a verification procedure, provide the user privacy, authentication and unlinkability. These privacy-preserving signature schemes are called Anonymous Digital Signatures (ADS). ADS schemes are usually based on zero knowledge proofs and commitment schemes. ADS can be solved as Group Signatures (GS) and Ring Signatures (RS) that are presented in the following subsections.
- **Attribute Based Signatures (ABS) and Attribute Based Encryption (ABE)** - ABS represents an advanced digital signature scheme that enables users to prove the possession of their attributes (e.g., age, membership) without revealing user's identity. ABE enables users to encrypt and decrypt data based on their attributes.
- **Homomorphic Encryption** - this technique represents an advanced encryption scheme that enables to work with ciphertexts without the need of their decryption. The privacy protection is enhanced because user's data remain in secret at the side of the third parties and service providers.

Besides mentioned cryptographic schemes, there are more privacy-enhancing approaches and communication protocols such as using anonymous and onion routing (e.g., TOR protocol [79]), mixers and mixed networks, anonymizer proxies, data blurring and minimization. These protocols are sometimes used in cooperation with cryptographic schemes in order to enhance privacy, especially at network and transport layers.

2.3.1 Group Signatures

Group signature (GS) schemes allow any group member (user) to anonymously sign a message on behalf of the group. Users can also authenticate themselves on behalf of a group without using certificates or user identities. The signature on the message is created by using a group secret member key. The signed message is verified by one public group key that is spread in the group of users. The basic principle of group signatures is depicted in Figure 2.3. The group signature schemes usually employ 4 basic parties:

- **Group Manager** (GM) - a semi-trusted party that adds group members into a group. GM also generates public parameters including a group public key *gpk* and issues the group member secret keys *gsk* of group members.
- **Revocation Manager** - a trusted party that can disclosure the identity of a dishonest member.
- **User** - a group member who owns the group member secret key *gsk*. The user can sign a message on behalf of the group. Users can also verify incoming signatures.
- **Verifier** - a party that verifies a signature using the group public key *gpk*.

In certain circumstances, e.g., breaking the rules, authorities can trace the identity of the signer by a revocation phase. The revocation phase can be done by the group manager, the revocation manager or by the cooperation of both parties. GM or RM can use group manager's secret key *gmsk* to reveal the user identity that is mapped in the signature and in the manager database of members. The group signatures are therefore pseudonymous because of the potential revocation of the user identity.

Group signatures were introduced by Chaum and Heyst [60] in 1991. Nowadays, many variants of group signature schemes have been proposed with various properties and different revocation methods. There are two basic types of group signatures:

- **Static group signatures** - the number of group members is fixed in the setup and group members secret keys are computed for each member in this phase. Static group signature schemes usually have 4 basic phases: Setup (Key

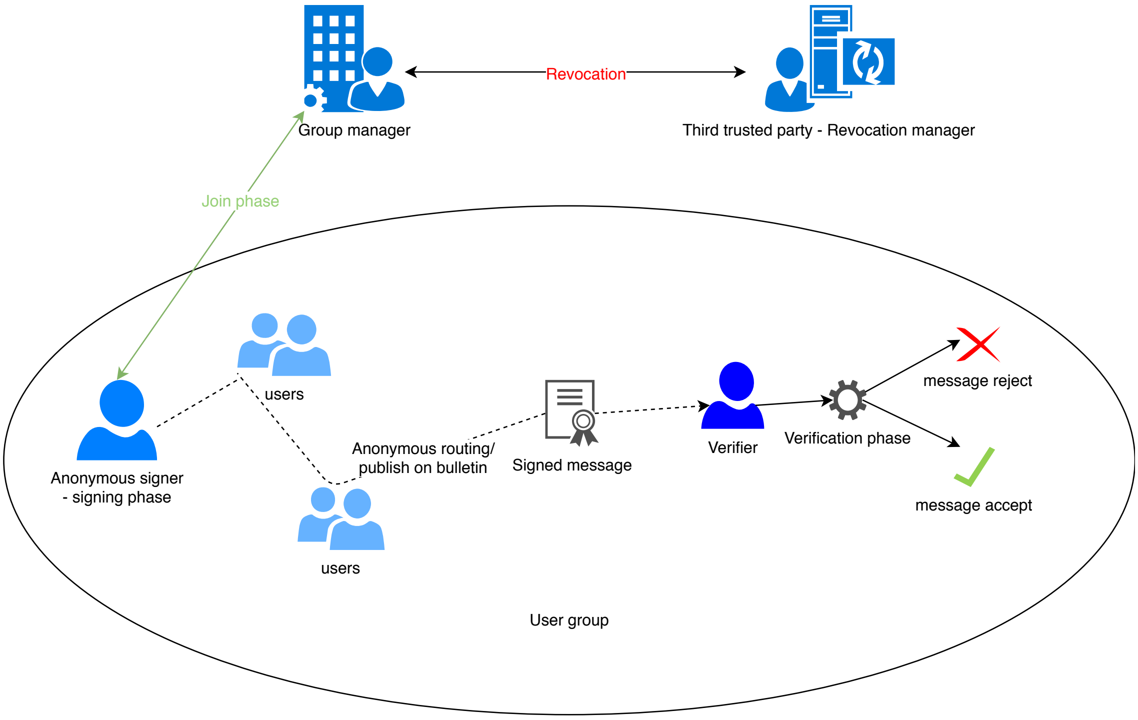


Fig. 2.3: Basic principle of group signatures.

generation), Signing, Verification and Open. Static group signature schemes do not offer a join phase where new members are added into existed systems. Therefore, static GS are not suitable for dynamic systems and ad hoc systems where the number of group members is unpredictable. On the other hand, static GS schemes are simpler than dynamic group signatures.

- **Dynamic group signatures** - these schemes provide a join phase that enables the group manager to add new members (users) into a group. Dynamic GS consist of 5 phases: Setup (Key generation), Join, Signing, Verification and Open. Furthermore, some GS schemes provide more algorithms such as membership revocation and update procedures. These phases can prevent former group members and malicious members who have been excluded from the group from generating valid signatures and re-joining the group.

Group signature schemes may provide the following properties:

- **Correctness** (soundness and completeness)- every correct signature produced by a valid user has to be always accepted and every incorrect signature has to be always rejected during the verification phase.
- **Unforgeability** - only a valid user is able to create a valid signature on behalf of the group.
- **Anonymity** - a verifier is not able to determine the identity of a user.
- **Complete** (full) **anonymity** - an adversary with a valid signature, *gpk* and

all keys of group members' $gsk[i]$ is not able to determine the identity of a user.

- **Traceability** - every valid user can be tracked by the group manager or the revocation manager by his/her signed messages.
- **Untraceability** - no one can be tracked by a verifier and other group members by his/her signed messages.
- **Unlinkability** - a verifier and other users are not able to link two signatures that are signed by a single member of the group.
- **Coalition-resistance** - it is not possible to create a valid signature by a subgroup of users.
- **Non-frameability** - all participating group members and the managers cannot forge a signature for a non-participating group member.
- **Exculpability** - the group manager is not capable to construct a valid signature on behalf of a certain group member (the actual signer of a group signature is able to claim that the signature is not signed by him/her even after revealing the private key).
- **Revocation** - a revoked user is not able to create valid signatures on behalf of the group.
- **Differentiation of group members** - all group members must have a different $gsk[i]$.
- **Immediate-revocation** - if a group member is revoked, his capability of creating group signatures is immediately disabled.

The standard ISO/IEC 20008-2:2013 [29] provides the general description of anonymous digital signature mechanisms that use a group public key. These schemes are usually based on zero knowledge and the proof of knowledge protocols and provide advanced security properties such as soundness, completeness, anonymity, unforgeability, traceability, coalition resistance, non-frameability and unlinkability (more detailed definitions can be found in [60], [37]).

The following subsections introduce 7 ADS schemes that are 3 pairing based schemes: the BBS scheme [43], the DP scheme [76] and the HLCCN scheme [111], and 4 non-pairing based schemes: the ACJT scheme [33], the CG scheme [53], the IMSTY scheme [112] and the HM scheme [101]. The following description of ADS schemes is an amended part of the theoretical background in the author's paper published in the international journal indexed in Scopus [11]. The following description focuses solely on the most used phases (signing, verification) of the anonymous digital signature schemes. In addition, it is shown which equations and parameters can be pre-computed and cached. Nonetheless, more details about these schemes can be found in their original papers.

2.3.2 Pairing based Group Signature Schemes

Pairing based group signature schemes are based on pairing-based cryptography. Three chosen schemes are the BBS scheme [43], the DP scheme [76] and the HLCCN scheme [111]. The sign and verification phases of these schemes usually contain pairing operations e that compute elements over pairing-friendly elliptic curves. More details can be found in [85].

Boneh Boyen Schaham (BBS) scheme

The security of the BBS scheme [43] is based on the Strong Diffie-Hellman and the Decisional Linear assumptions. The scheme uses the standard generalization of the Schnorr's protocol [172] for proving the knowledge of the discrete logarithm. This group signature scheme consists of these phases: Key generation, Sign, Verify and Open.

The sign phase generates a signature σ on a message $M \in \{0,1\}^*$ by using a member secret key $gsk = (A, x)$ and a group public key $gpk = (g_1, g_2, h, u, v, w = g_2^\gamma)$, where γ is a secret issuer key. The signature is computed by the zero-knowledge protocol of the Strong Diffie-Hellman assumption. A signer randomly selects exponents $\alpha, \beta \in \mathbb{Z}_p$ and computes the linear encryption of A represented by values T_1, T_2, T_3 and variable parameters δ_1, δ_2:

$$\begin{aligned} T_1 &= u^\alpha, T_2 = v^\beta, T_3 = Ah^{\alpha+\beta}, \\ \delta_1 &= \alpha x, \delta_2 = \beta x. \end{aligned} \tag{2.7}$$

The blinding values $r_\alpha, r_\beta, r_x, r_{\delta_1}, r_{\delta_2}$ are randomly picked from $\mathbb{Z}_p$, and values R_1, R_2, R_3, R_4, R_5 are computed:

$$\begin{aligned} & R_1 = u^{r_\alpha}, R_2 = v^{r_\beta}, \\ & R_3 = e(T_3, g_2)^{r_x} \cdot e(h, w)^{-r_\alpha - r_\beta} \cdot e(h, g_2)^{-r_{\delta_1} - r_{\delta_2}}, \\ & R_4 = T_1^{r_x} u^{-r_{\delta_1}}, \\ & R_5 = T_2^{r_x} v^{-r_{\delta_2}}. \end{aligned} \tag{2.8}$$

The signer computes a challenge $c \in \mathbb{Z}_p$ using the hash function $\mathcal{H}$ and values $s_\alpha, s_\beta, s_x, s_{\delta_1}, s_{\delta_2}$ to seal the proof of the knowledge of $(\alpha, \beta, x, \delta_1, \delta_2)$:

$$\begin{aligned} & c = \mathcal{H}(M, T_1, T_2, T_3, R_1, R_2, R_3, R_4, R_5), \\ & s_\alpha = r_\alpha + c\alpha, s_\beta = r_\beta + c\beta, \\ & s_x = r_x + cx, \\ & s_{\delta_1} = r_{\delta_1} + c\delta_1, s_{\delta_2} = r_{\delta_2} + c\delta_2. \end{aligned} \tag{2.9}$$

The signer outputs the signature $\sigma = (T_1, T_2, T_3, c, s_\alpha,\ s_\beta, s_x, s_{\delta_1}, s_{\delta_2})$. All pairing operations can be precomputed and cached (pairing precomputation) because the input parameters are static. If the signer uses the full precomputation then all values are generated and the parameters in Equations 2.7 and 2.8 are computed in advance. The signer computes only the parameters in Equation 2.9.

In the verification phase, a verifier checks the validity of the signature σ generated on the message M by using the group public key $gpk = (g_1, g_2, h, u, v, w)$. All the values $R_1, R_2, R_3, R_4, R_5, c$ are restored:

$$\begin{aligned}
&R_1^{'} = u^{s_\alpha} T_1^{-c},\\
&R_2^{'} = v^{s_\beta} T_2^{-c},\\
&R_3^{'} = e(T_3, g_2)^{s_x} e(h, w)^{(-s_\alpha - s_\beta)} e(h, g_2)^{(-s_{\delta_1} - s_{\delta_2})} (e(T_3, w) e(g_1, g_2)^{-1})^c,\\
&R_4^{'} = u^{-s_{\delta_1}} T_1^{s_x},\\
&R_5^{'} = v^{-s_{\delta_2}} T_2^{s_x},\\
&c^{'} = \mathcal{H}(M, T_1, T_2, T_3, R_1^{'}, R_2^{'}, R_3^{'}, R_4^{'}, R_5^{'}).
\end{aligned} \tag{2.10}$$

If c is equal with restored c', then the verifier accepts the signature and rejects otherwise. All pairing operations with static values $e(h, w), e(h, g_2), e(g_1, g_2)$ can be precomputed in advance. The pairings $e(T_3, g_2)$ and $e(T_3, w)$ can be collapsed into one pairing operation.

Delerablee and Pointcheval (DP) scheme

The security of the DP scheme [76] holds under the q-SDH and the XDH assumptions (also DLIN assumption can be used), in the random oracle model. The scheme improves the security (anonymity and non-frameability) of the BBS scheme [43] by involving an extra parameter into a membership certificate during the join phase. The scheme consists of these phases: Key generation, Join, Sign, Verify, Open and Judge.

The sign phase generates a signature σ on a message $M \in \{0,1\}^*$ by using a member secret key $gsk = (A, x, y)$ and a group public key $gpk = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, \psi,$ $g_1, k, h = k^{\xi_1}, g = k^{\xi_2}, g_2, w = g_2^{\gamma})$. A signer randomly selects values $\alpha, \beta, r_\alpha,$ $r_\beta, r_x, r_z \in \mathbb{Z}_p^*$ and computes values $T_1, T_2, T_3, T_4, z,$ R_1, R_2, R_3, R_4:

$$\begin{aligned}
&T_1 = k^\alpha, T_2 = A h^\alpha, T_3 = k^\beta, T_4 = A g^\beta, z = \alpha x + y,\\
&R_1 = k^{r_\alpha}, R_2 = e(T_2, g_2)^{r_x} e(h, w)^{-r_\alpha} e(h, g_2)^{-r_z},\\
&R_3 = k^{r_\beta}, R_4 = h^{r_\alpha} g^{-r_\beta}.
\end{aligned} \tag{2.11}$$

The signer computes a challenge $c \in \mathbb{Z}_p^*$ using the hash function $\mathcal{H}$ and values $s_\alpha, s_\beta, s_x, s_z$ to seal the proof of knowledge of (α, β, x, z):

$$\begin{aligned}
&c = \mathcal{H}(M, T_1, T_2, T_3, T_4, R_1, R_2, R_3, R_4),\\
&s_\alpha = r_\alpha + c\alpha, s_\beta = r_\beta + c\beta, s_x = r_x + cx, s_z = r_z + cz.
\end{aligned} \tag{2.12}$$

The signer outputs the signature $\sigma = (T_1, T_2, T_3, T_4, c,\ s_\alpha, s_\beta, s_x, s_z)$. All pairing operations can be precomputed and cached like in the BBS scheme. If the signer uses the full precomputation then all values are generated and the parameters in Equation 2.11 are computed in advance. The signer computes only the parameters in Equation 2.12.

In the verification phase, a verifier checks the validity of the signature σ generated on the message M by the following:

$$\begin{aligned}
R_1^{'} &= k^{s_\alpha} T_1^{-c},\\
R_2^{'} &= e(T_2, g_2)^{s_x} e(h, w)^{-s_\alpha} e(h, g_2)^{-s_z} (e(g_1, g_2)/e(T_2, w))^{-c},\\
R_3^{'} &= k^{s_\beta} T_3^{-c},\\
R_4^{'} &= h^{s_\alpha} g^{-s_\beta} (T_2/T_4)^{-c},\\
c^{'} &= \mathcal{H}(M, T_1, T_2, T_3, T_4, R_1^{'}, R_2^{'}, R_3^{'}, R_4^{'}).
\end{aligned} \tag{2.13}$$

If c is equal with restored c', then the verifier accepts the signature and rejects otherwise. All pairings with static values $e(h, w), e(h, g_2), e(g_1, g_2)$ can be precomputed in advance. The pairings $e(T_2, g_2)$ and $e(T_2, w)$ can be collapsed into one pairing operation.

Hwang *et al.* (HLCCN) scheme

The short dynamic group signature scheme [111] is included in the IS020008-2 standard [29]. The security of the scheme holds under the modified q-SDH and the XDH assumptions [111] in the random oracle model. The scheme consists of these phases: Setup, Join/Issue, Sign, Verify, Open, Judge and Link.

The sign phase generates a signature σ on a message $M \in \{0,1\}^*$ by using a member secret key $gsk = (A = (g_1 g_2^{-y} w^{-z})^{1/(\theta+x)}, x, y, z)$ and a group public key $gpk = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g, g_1, g_2, h_1, h_\theta = h_1^\theta, u, w = u^\eta, d = u^\xi)$. A signer randomly selects values $\alpha, r_\alpha, r_\gamma,\ r_x, r_y \in \mathbb{Z}_p^*$ and computes values $T_1, T_2, T_3, \gamma, R_1, R_2, R_3$:

$$\begin{aligned}
T_1 &= u^\alpha, T_2 = A w^\alpha, T_3 = g^y d^\alpha, \gamma = \alpha x - z \bmod p,\\
R_1 &= u^{r_\alpha}, R_2 = e(T_2, h_1)^{r_x} e(w, h_\theta)^{-r_\alpha} e(w, h_1)^{-r_\gamma} e(g_2, h_1)^{r_y}, R_3 = g^{r_y} d^{r_\alpha}.
\end{aligned} \tag{2.14}$$

The signer computes a challenge $c \in \mathbb{Z}_p^*$ using the hash function $\mathcal{H}$ and values $s_\alpha, s_\gamma, s_x, s_y$ to seal the proof of knowledge of (α, γ, x, y):

$$\begin{aligned}
&c = \mathcal{H}(M, T_1, T_2, T_3, R_1, R_2, R_3),\\
&s_\alpha = r_\alpha + c\alpha, s_\gamma = r_\gamma + c\gamma, s_x = r_x + cx, s_y = r_y + cy.
\end{aligned} \tag{2.15}$$

The signer outputs the signature $\sigma = (T_1, T_2, T_3, c, s_\alpha,\ s_\gamma, s_x, s_y)$. All pairings can be precomputed and cached like in the BBS scheme. If the signer uses the full precomputation then all values are generated and the parameters in Equation 2.14

are computed in advance. The signer computes only the parameters in Equation 2.15.

In the verification phase, a verifier checks the validity of the signature σ generated on the message M by the following:

$$\begin{aligned}
&R_2^{'} = e(T_2, h_1)^{s_x} e(w, h_\theta)^{-s_\alpha} e(w, h_1)^{-s_\gamma} e(g_2, h_1)^{s_y} (e(T_2, h_\theta)/e(g_1, h_1))^c,\\
&R_1^{'} \;=\; u^{s_\alpha} T_1^{-c}, R_3^{'} \;=\; g^{s_y} d^{s_\alpha} T_3^{-c},\\
&c^{'} = \mathcal{H}(M, T_1, T_2, T_3, R_1^{'}, R_2^{'}, R_3^{'}).
\end{aligned} \tag{2.16}$$

If c is equal with restored c', then the verifier accepts the signature and rejects otherwise. All pairings with static values $e(w, h_\theta), e(w, h_1), e(g_2, h_1), e(g_1, h_1)$ can be precomputed in advance. The pairings $e(T_2, h_1)$ and $e(T_2, h_\theta)$ can be collapsed into one pairing operation.

2.3.3 Non-Pairing based Group Signature Schemes

This subsection briefly describes non-pairing anonymous signature schemes such as the ACJT scheme [33], the CG scheme [53], the IMSTY scheme [112] and the HM scheme [101].

Ateniese *et al.* (ACJT) scheme

This group signature scheme [33] is secure under the strong RSA problem and the decisional Diffie-Hellman problem. The scheme relies on the Fiat-Shamir heuristic (the random oracle model). The scheme consists of five phases: Setup, Join, Sign, Verify and Open.

The sign phase creates a signature σ on a message $M \in \{0{,}1\}^*$ by using a member secret key $gsk = (x)$, a membership certificate $[A, e]$ and public parameters. A signer randomly selects w, r_1, r_2, r_3, r_4 and computes:

$$\begin{aligned}
&T_1 \;=\; Ay^w \bmod n, T_2 \;=\; g^w \bmod n, T_3 \;=\; g^e h^w \bmod n\\
&d_1 = T_1^{r_1}/(a^{r_2} y^{r_3}) \bmod n, d_2 = T_2^{r_1}/g^{r_3} \bmod n,\\
&d_3 = g^{r_4} \bmod n, d_4 = g^{r_1} h^{r_4} \bmod n,
\end{aligned} \tag{2.17}$$

The signer computes a challenge c using the hash function $\mathcal{H}$ and seals the proof of knowledge:

$$\begin{aligned}
&c = \mathcal{H}(g, h, y, a_0, a, T_1, T_2, T_3, d_1, d_2, d_3, d_4, M),\\
&s_1 = r_1 - c(e - 2^{\gamma_1}), s_2 = r_2 - c(x - 2^{\lambda_1}),\\
&s_3 = r_3 - cew, s_4 = r_4 - cw.
\end{aligned} \tag{2.18}$$

The signer outputs the signature $\sigma = (c, s_1, s_2, s_3, s_4,\, T_1, T_2, T_3)$. If the full precomputation is used then all values are generated and the parameters in Equation 2.17

are computed in advance. The signer computes only the parameters in Equation 2.18.

In the verification phase, a verifier checks the validity of the signature σ generated on the message M by the following:

$$\begin{aligned}
&d_1' = a_0^c T_1^{s_1 - c2^{\gamma_1}} / (a^{s_2 - c2^{\lambda_1}} y^{s_3}) \bmod n,\\
&d_2' = T_2^{s_1 - c2^{\gamma_1}} / g^{s_3} \bmod n,\\
&d_3' = T_2^c g^{s_4} \bmod n, d_4' = T_3^c g^{s_1 - c2^{\gamma_1}} h^{s_4} \bmod n,\\
&c' = \mathcal{H}(g, h, y, a_0, a, T_1, T_2, T_3, d_1', d_2', d_3', d_4', M).
\end{aligned} \tag{2.19}$$

If c is equal with restored c', then the verifier accepts the signature and rejects otherwise.

Camenisch and Groth (CG) scheme

This group signature scheme [53] is related to the ACJT scheme [33]. The security of the scheme holds under the strong RSA assumption and the decisional Diffie-Hellman assumption. The scheme consists of six phases: Key generation, Join, Sign, Verify, Open proof and Revoke.

The sign phase creates a signature σ on a message $M \in \{0,1\}^*$ by using a member secret key $gsk_i = (vk, w_i, x_i, r_i = r_i' + r_i'', y_i, e_i)$, a group public key $gpk = (n, a, g, h, Q, P, F, G, H, w)$. A signer randomly selects $r, r_x, r_r, r_e, R_R, R \in \mathbb{Z}_q$ and computes:

$$\begin{aligned}
&u = h^r y_i w_i \bmod n, U_1 = F^R \bmod P, U_2 = G^{R + x_i} \bmod P,\\
&U_3 = H^{R + e_i} \bmod P, v = u^{r_e} g^{-r_x} h^{r_r} \bmod n,\\
&V_1 = F^{R_R} \bmod P, V_2 = G^{R_R + r_x} \bmod P, V_3 = H^{R_R + r_e} \bmod P.
\end{aligned} \tag{2.20}$$

The signer computes a challenge c using the hash function $\mathcal{H}$ and seals the proof of knowledge:

$$\begin{aligned}
&c = \mathcal{H}(vk, u, v, U_1, U_2, U_3, V_1, V_2, V_3, M), z_x = r_x + cx_i,\\
&z_r = r_r + c(-r_i - rE_i), z_e = r_e + ce_i, Z_R = R_R + cR \bmod Q.
\end{aligned} \tag{2.21}$$

The signer outputs the signature $\sigma = (c, u, U_1, U_2, U_3,\ z_x, z_r, z_e, Z_R)$. If the full precomputation is used then all values are generated and the parameters in Equation 2.20 are computed in advance. The signer computes only the parameters in Equation 2.21.

In the verification phase, a verifier checks the validity of the signature σ generated on the message M by the following:

$$\begin{aligned}
&v = (aw)^{-c} g^{-z_x} h^{z_r} u^{c2^{l_E} + z_e} \bmod n, V_1' = U_1^{-c} F^{Z_R} \bmod P,\\
&V_2' = U_2^{-c} G^{Z_R + z_x} \bmod P, V_3' = U_3^{-c} H^{Z_R + z_e} \bmod P,\\
&c' = \mathcal{H}(vk, u, v, U_1, U_2, U_3, V_1', V_2', V_3', M).
\end{aligned} \tag{2.22}$$

If c is equal with restored c', then the verifier accepts the signature and rejects otherwise.

Isshiki *et al.* (IMSTY) scheme

This group signature scheme [112] with membership revocation is included in the IS020008-2 standard [29]. The security of this scheme holds under the strong RSA assumption and the decisional Diffie-Hellman assumption on an elliptic curve group. The scheme consists of nine phases: Setup for issuing manager, Setup for user revocation manager, Setup for opening manager, Join, Sign, Verify, Open, User revocation and Update.

The sign phase creates a signature σ on a message $M \in \{0,1\}^*$ by using a member secret key $gsk = (x)$, a member public key $gpk = (A = a^x \mathrm{mod} n, e', h, B)$, an issuer public key $ipk = (n = p_1p_2, a_0, a_1, a_2)$, a revocation public key $rpk = (l = l_1l_2, b, w)$ and an opening public key $opk = (q, G, H_1 = [y_1]G, H_2 = [y_2]G)$, where parameters $isk = (p_1, p_2), rsk = (l_1, l_2), osk = (y_1, y_2)$ are randomly generated and stored as the secret keys of the scheme's parties. A signer randomly selects $\rho_E, \rho_m, \rho_r, \mu_x, \mu_s, \mu_e, \mu_t$, and $\mu_E \in \mathbb{Z}_q$ and computes values $E_0, E_1, E_2, A_{COM}, B_{COM}, V_{ComCipher}$, V_{ComMPK}, V_{ComRev}:

$$\begin{aligned}
&E_0 = [\rho_E]G, E_1 = h_i + [\rho_E]H_1, E_2 = h_i + [\rho_E]H_2,\\
&A_{COM} = Aa_2^{\rho_m} \bmod n, e = 2^{K_e} + e', s = e\rho_m,\\
&B_{COM} = Bw^{\rho_r} \bmod l, t = e'\rho_r,\\
&V_{ComCipher} = (V_{ComCipher0} = [\mu_E]G,\\
&V_{ComCipher1} = [\mu_x]G + [\mu_E]H_1, V_{ComCipher2} = [\mu_x]G + [\mu_E]H_2),\\
&V_{ComMPK} = a_1^{\mu_x} a_2^{\mu_s} A_{COM}^{-\mu_{e'}} \bmod n, V_{ComRev} = w^{\mu_t} B_{COM}^{-\mu_{e'}} \bmod l.
\end{aligned} \tag{2.23}$$

The signer computes a challenge c using the hash function $\mathcal{H}$ and values τ_x, τ_s, τ_t, $\tau_{e'}$, τ_E to seal the proof of knowledge of (x, s, t, e', E):

$$\begin{aligned}
&c = \mathcal{H}(\text{lengths of parameters}, ipk, opk, rpk, E_0, E_1, E_2, A_{COM},\\
&B_{COM}, V_{ComCipher}, V_{ComMPK}, V_{ComRev}, M), \tau_x = cx + \mu_x,\\
&\tau_s = cs + \mu_s, \tau_t = ct + \mu_t, \tau_{e'} = ce' + \mu_{e'}, \tau_E = c\rho_E + \mu_E \bmod q).
\end{aligned} \tag{2.24}$$

The signer outputs the signature $\sigma = (E_0, E_1, E_2, A_{COM},\ B_{COM}, c, \tau_x, \tau_s, \tau_t, \tau_{e'}, \tau_E)$. If the full precomputation is used then all values are generated and the parameters in Equation 2.23 are computed in advance. The signer computes only the parameters in Equation 2.24.

In the verification phase, a verifier checks the validity of the signature σ generated on the message M by the following:

$$\begin{aligned}
&\tau_e \ = c2^{K_e} + \tau_{e'}, V'_{ComCipher} = (V'_{ComCipher0} = [\tau_E]G - [c]E_0, \\
&V'_{ComCipher1} = [\tau_x]G + [\tau_E]H_1 - [c]E_1, \\
&V'_{ComCipher2} = [\tau_x]G + [\tau_E]H_2 - [c]E_2), \\
&V'_{ComMPK} = a_0^c a_1^{\tau_x} a_2^{\tau_s} A_{COM}^{-\tau_e} \bmod n, \\
&V'_{ComRev} = b^c w^{\tau_t} B_{COM}^{-\tau_{e'}} \bmod l, \\
&c' = \mathcal{H}(\text{lengths of parameters}, ipk, opk, rpk, E_0, E_1, E_2, A_{COM}, B_{COM}, \\
&V'_{ComCipher}, V'_{ComMPK}, V'_{ComRev}, M).
\end{aligned} \tag{2.25}$$

If c is equal with restored c', then the verifier accepts the signature and rejects otherwise.

Hajny and Malina (HM) scheme

The attribute-based authentication scheme [101] can be used as a group signature scheme [102]. This scheme is secure under the generalized discrete logarithm assumption and the integer factorization hardness in Okamoto-Uchiyama trapdoor one-way function [158]. The scheme consists of five phases: Setup, Register, Sign, Verify and Revoke.

The sign phase creates a signature σ on a message $M \in \{0{,}1\}^*$ by using a member secret key $gsk = (w_1, w_2, w_M)$ and public system parameters $params = (q, p, h_1, h_2, n, g_1, g_2, g_3, G_{PK} = g_1^{w_1} g_2^{w_2} g_3^{w_M} \bmod n)$. A signer randomly selects K_S, r_1, r_2, r_3, r_S and computes:

$$\begin{aligned}
&A = G_{PK}^{K_S} \bmod n, C_1 = g_3^{K_S w_M} \bmod n, C_2 = g_3^{K_S} \bmod n, \\
&\bar{A} = G_{PK}^{r_S} \bmod n, \bar{G}_{PK} = g_1^{r_1} g_2^{r_2} g_3^{r_3} \bmod n, \\
&\bar{C_1} = g_3^{r_3} \bmod n, \bar{C_2} = g_3^{r_S} \bmod n.
\end{aligned} \tag{2.26}$$

Then, the signer computes a challenge c using the hash function $\mathcal{H}$ and seals the proof of knowledge:

$$\begin{aligned}
&c = \mathcal{H}(params, M, A, \bar{A}, \bar{G}_{PK}, C_1, C_2, \bar{C_1}, \bar{C_2}), \\
&z_1 = r_1 - cK_S w_1, z_2 = r_2 - cK_S w_2, \\
&z_3 = r_3 - cK_S w_M, z_S = r_S - cK_S.
\end{aligned} \tag{2.27}$$

The signer outputs the signature $\sigma = (A, C_1, C_2, c, z_1,\ z_2, z_3, z_S)$. If the full precomputation is used then all values are generated and the parameters in Equation 2.26 are computed in advance. The signer computes only the parameters in Equation 2.27.

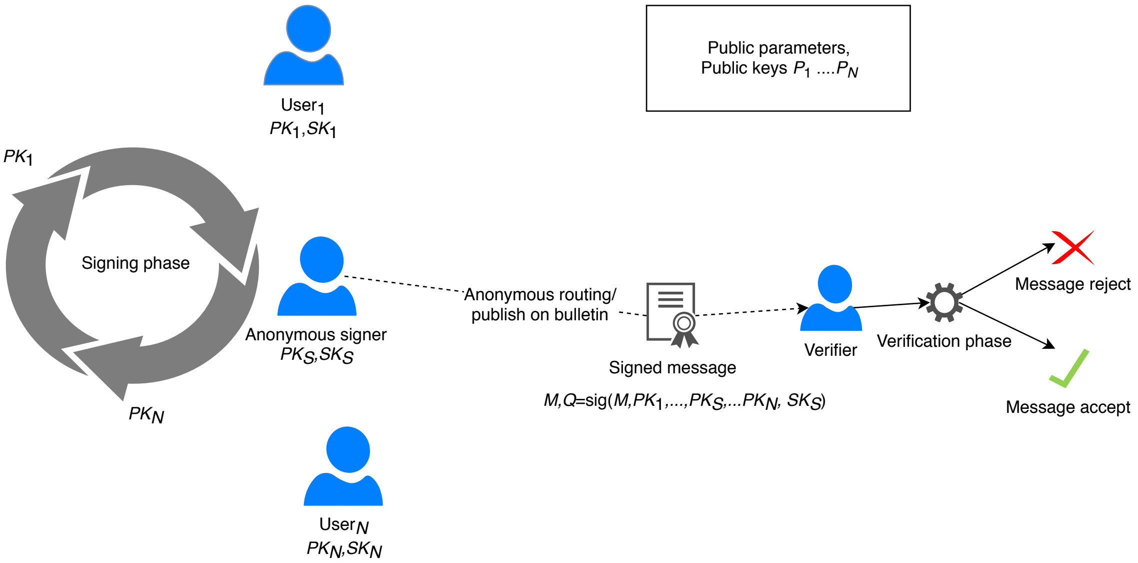


Fig. 2.4: Basic principle of ring signatures.

In the verification phase, a verifier checks the validity of the signature σ generated on the message M by the following:

$$\begin{aligned}
&G\bar{_{PK}} = A^c g_1^{z1} g_2^{z2} g_3^{z3} \bmod n,\\
&\bar{A} = A^c G_{PK}^{z_S} \bmod n, \bar{C_1} = C_1^c g_3^{z3} \bmod n,\\
&\bar{C_2} = C_2^c g_3^{z_S} \bmod n, c' = \mathcal{H}(params, M, A, \bar{A}, G\bar{_{PK}}, C_1, C_2, \bar{C_1}, \bar{C_2}).
\end{aligned} \tag{2.28}$$

If c is equal with restored c', then the verifier accepts the signature and rejects otherwise.

2.3.4 Ring Signatures

Ring signatures (RS) are very similar to group signatures. A member of a group (a ring) can anonymously sign a message on behalf of a group (a ring). In ring signatures firstly defined in [167], a signer signs a message with his/her private key and then he/she publishes a set of public keys together with his/her public key. RS remove the centralization point of a group manager and are often called as ad hoc group signatures. RS usually provide a perfect privacy (untracebility) because there are no authority that can trace the signers and their signatures. The basic principle of ring signatures is depicted in Figure 2.4.

The ring signature schemes have similar properties like group signature schemes. Because a signer is not able to prove his/her signature (non-repudation) and RS are untraceable, the ring signature schemes also provide the following security properties:

- **Claimability** - a signer is able to prove that he/she generated a given signature. This property can be reached by adding the proof of knowledge of some secret.
- **Culpability** - any party with given a message-signature pair and the private key of the group member can determine if this group member is the actual signer.
- **Deniability** - a signer signs a message on behalf of an ad hoc group and he/she convinces a verifier that this message is correct. Then, the authority is able to prove that the specified user is not the signer of the signature. The proof can be done without revealing the actual signer. This property guarantees full anonymity.
- **Non-frameability** - if a signer did not sign a signature, he/she can prove this claim using a disavowal protocol.
- **Linkability** - two signatures by the same signer can be linked. This property prevents double voting.
- **Signer ambiguity** - no party can identify who signs the message. This property depends on the number of members or used public keys in the ring.
- **Spontaneity** - RS scheme does not need a group manager, TTP or an interactive setup.

Ring signature schemes are usually implemented in anonymous membership authentication for ad hoc groups, e-voting protocols and in anonymous transactions such as the CryptoNote protocol [195] . The section 4.3 provides more information about ring signatures and also present author's proposed solution based on lightweight ring signatures.

2.3.5 Attribute Based Signatures and Attribute Based Encryption

Attribute-based schemes are cryptographic schemes that are designed to enhance user privacy. These schemes use defined attributes (e.g., personal attributes, policies) in signing/verification or in encryption/decryption phases.

Attribute Based Signatures

Attribute based signature (ABS) schemes allow users to generate signatures with attributes which are satisfying a policy without leaking more personal information. The users who request some data or services have to generate signatures by using the attributes. These signers remain anonymous and are indistinguishable among all users. The signers are not able to forge signatures with attributes that they do not own. More details about the attribute based signature and its application can be found in [129]. The attribute based signatures are often based on the attribute based credentials schemes, such as Idemix [54], U-Prove [160] and the HM scheme [101] that are used in authentication systems where users are proving the possession of the attributes. Several proposed ABS schemes exist, e.g., the Su *et al.* scheme [185] and the Alcaide *et al.* scheme [30]. Nevertheless, these schemes are usually not suitable for constrained devices due to the many expensive operations (bilinear pairing and exponentiation).

Attribute Based Encryption

Attribute-Based Encryption (ABE) has enhanced the Identity-Based Encryption (IBE) that defines public keys as arbitrary strings, e.g., the email address, names etc. ABE does not use an identity as a public key but defines a set of attributes (e.g., roles) that are needed for encryption or decryption. ABE schemes can be based on keys, i.e., Key-Policy ABE (KP-ABE) where the message can be decrypted only by a user that holds the set of the attributes. Ciphertext-Policy ABE (CP-ABE) schemes use policies that are defined over the set of attributes with using conjunctions, disjunctions and threshold gates. ABE schemes are usually computationally expensive due to many pairings but this approach can be useful in some cloud storage applications [95]. Several proposed ABE schemes exist, for example, Key-Policy ABE (GPSW) scheme [93] and Water's CP-ABE scheme [203].

2.3.6 Homomorphic Encryption

Homomorphic encryption allows the users to encrypt sensitive data and enables to process these encrypted data without their decryption. These encrypted data can be processed by another party without revealing what information is inside. There are two basic types of homomorphic encryption schemes: Partially Homomorphic Encryption (PHE) and Full Homomorphic Encryption (FHE). There are several partially homomorphic encryption systems such as Paillier [159] or Benaloh [38]. Nevertheless, some works such as [90], [49] and [69] show that fully homomorphic encryption (FHE) schemes can be very computationally and memory expensive. According to the paper [174], homomorphic encryption can be also a part of a secure multi-party computation that creates the new opportunities in the area of development privacy-preserving ubiquitous applications. Further, Sun *et al.* [186] propose a multiplication homomorphism method that is used as a privacy protection solution in IoT services.

Generally, homomorphic encryption can provide data privacy during data aggregation services (smart grid services [137], WSN services [181], healthcare monitoring with an IoT platform [188], IoT data collection services [206]). These solutions are usually based on the Pailler's homomorphic encryption scheme [159]. This PHE scheme provides the additive property. The product of two ciphertexts is equal to the sum of two corresponding plaintexts after the decryption of the product. This encryption enables to sum encrypted data without a private key. In addition, the Pailler's homomorphic encryption scheme enables the addition and multiplication of a plaintext by a constant value. These properties are useful in privacy-preserving data aggregation services. The Pailler's scheme with several modular arithmetic operations in encryption (two exponentiation and one multiplication) and in decryption (one exponentiation, two multiplication, one division) is more expensive than the RSA scheme (one exponentiation). The equations for encryption and decryption are defined as follows:

$$\begin{aligned} c &= g^m \cdot r^n \mod n^2, \\ m &= (c^\lambda \mod n^2 - 1)/n \cdot \mu \mod n, \end{aligned} \tag{2.29}$$

where c is a ciphertext, m is a message in a plaintext, r is a random number $r \in \mathbb{Z}_n^*$, (n, g) is a public key and $(\lambda,\ \mu)$ is a private key.

Homomorphic encryption schemes can be the useful tools for the applications using the high-performed devices or cloud storage solutions but currently, the application of these schemes on resource-constrained nodes is not practical due to expensive operations and large sizes of keys, parameters and ciphertexts.

2.4 Theoretical Evaluation of Digital Signature Schemes

This section presents the theoretical evaluation of conventional digital schemes such as RSA signature scheme, Rabin signature scheme [164], Schnorr signature scheme [172], EdDSA scheme [41], DSA and ECDSA schemes [115] and chosen anonymous digital signature schemes such as group signatures (the BBS scheme [43], the ACJT scheme [33], the DP scheme [76], the HLCCN scheme [111], the CG scheme [53], the IMSTY scheme [112], the HM GS scheme [102], the HCCN scheme [110], the EH scheme [82]) and the ring signature scheme proposed by Liu *et al.* [134] .

Table 2.1 shows the number of operations in sign and verification phases, the length of signatures, revocation mechanisms and complexity that is presented as the number of algorithms/phases used in each scheme. On one hand, pre-caching operations such as bilinear pairing operations of two constant values are considered. These operations can be precomputed once and their results are not variable if different messages are processed. On the other hand, it is not considered the full pre-computation of dynamic parameters (random values) and precomputed coupon techniques. Further, the sign and verification phases are evaluated without the influence of a number of revoked users in the systems (i.e., a revocation list is empty) and some private key/credential revocation approaches do not influence the verification.

Table 2.1 denotes a pairing operation as P, exponentiation as E, inversion as INV, squaring as SQ, multiplication as M, addition (subtraction) as A and a hash function as H. N denotes the number of users in a ring/ad hoc group. In the pairing-based schemes, the time execution of exponentiation and multiplication operations can depend on the lengths of elements that are in different groups and fields (p, $\mathbb{G}_1$, $\mathbb{G}_2$, $\mathbb{G}_T$). The length l_{G_1} describes the length of a group element $\in \mathbb{G}_1$ (e.g., 175 bits). The length l_{G_2} describes the length of a group element $\in \mathbb{G}_2$ (e.g., 175 bits). The length of a group element $\in \mathbb{G}_T$ l_{G_T} is computed as $k \cdot l_{G_1}$, e.g., 1050 bits, where k is an embedded degree (e.g., k=6). l_p denotes the length of an element in modulo p (e.g., 170 bits). In the non-pairing based schemes, l_n denotes the length of the RSA modulo n (e.g., 1024 bits). l_z denotes the length of the scalars of various lengths less than n (e.g., $<$ 1024 bits). l_c denotes the length of the hash used (e.g., 160 bits). l_{ec} denotes the length of an elliptic curve element (e.g., 163 bits). The total lengths of signatures depend on the security level chosen.

Conventional digital signature schemes need less operations than anonymous digital signature schemes due to their low complexity. On the other hand, ADS schemes provide privacy-preserving properties for their users (signers), hence, the number of operations are higher during signing and verification phases in comparison with phases in conventional signatures. The most efficient conventional scheme in the signing phase is EdDSA that needs 1 scalar multiplication as the most expensive operation. The Rabin signature scheme is the most efficient in the verification phase where the one squaring is performed. All conventional signature schemes are evaluated without additional optimization procedures and tricks (e.g., using CRT, Shamir's trick, batch verification).

Among chosen ADS schemes, the most efficient sign phase is provided by the HM scheme [102] which needs to perform only 9 exponentiation, 10 modular multiplication, 4 addition and 1 hash function. The most efficient verification phase is provided by the HM scheme [102] that takes 10 exponentiation, 6 modular multiplication and 1 hash function. The shortest signature is offered by the HLCCN scheme [111]. The HCCN scheme [110] extends the HLCCN scheme [111] and proves its security features (anonymity, traceability, nonframeability, linkability) under a random oracle model. However, the HCCN scheme [110] provides the same number of operations and the length of the signature as the HLCCN scheme [111]. Recently, Emura and Hayashi [82] have proposed a Group Signatures with Time-token Dependent Linking scheme that supports verifier-local revocation and backward unlinkability. The

Tab. 2.1: Theoretical Evaluation of Digital Signature Schemes.

Scheme	Pairing-based	Sign phase (number of operations)	Verification phase (number of operations)	Communication / memory cost - Signature length [b]	Revocation	Complexity
Conventional Digital Signature Schemes						
RSA scheme	no	1E	1E	l_n (> 2048 b)	revocation list	3 phases: Setup, Sign, Verify
RABIN [164]	no	1E	1SQ	$2l_n$ (> 4096 b)	revocation list	3 phases: Setup, Sign, Verify
Schnorr signature [172]	no	1E + 1M + 1A+ 1H	2E +1M + 1H	$2l_q + 1l_p$ (> 2048 b)	revocation list	3 phases: Setup, Sign, Verify
EdDSA signature [41]	no	1SM+ 1M + 1A+2H	1SM+ 1M + 1H+ 1A	$2l_q$ (> 512 b)	revocation list	3 phases: Setup, Sign, Verify
DSA signature	no	1E in l_n + (1INV + 1M +1A +1H) in l_q	2E+3M+1H in l_q	$2l_q$ (> 512 b)	revocation list	3 phases: Setup, Sign, Verify
ECDSA signature [115]	no	1SM+ 1INV+2M+ 1A+1H	2SM+ 1INV + 2M + 1H+ 1A	$2l_q$ (> 512 b)	revocation list	3 phases: Setup, Sign, Verify
Anonymous Digital Signature Schemes						
BBS [43]	yes	0P+($9l_{G_1}$+$3l_{G_T}$)E + ($8l_p$ +$3l_{G_1}$+$2l_{G_T}$)M +9A+1H	1P+($8l_{G_1}$+$2l_{G_2}$+$3l_{G_T}$)E + ($4l_{G_1}$+$1l_{G_2}$+ $3l_{G_T}$)M +2A+1H	$3l_{G_1} + 6l_p$ (> 1545 b)	private key update	4 phases: Key generation, Sign, Verify, Open
DP [76]	yes	0P+($8l_{G_1}$+$3l_{G_T}$)E + ($6l_p$+$3l_{G_1}$+$2l_{G_T}$)M +6A+1H	1P+($7l_{G_1}$+$2l_{G_2}$+$3l_{G_T}$)E + ($5l_{G_1}$+$1l_{G_2}$+ $3l_{G_T}$)M +0A+1H	$4l_{G_1} + 5l_p$ (> 1559 b)	private key update	6 phases: Key generation, Join, Sign, Verify, Open, Judge
HLCCN [111] and HCCN [110]	yes	0P+($7l_{G_1}$+$5l_{G_T}$)E + ($6l_p$+$3l_{G_1}$+$4l_{G_T}$)M +5A+1H	1P+($5l_{G_1}$+$2l_{G_2}$+ $4l_{G_T}$)E + ($3l_{G_1}$+$1l_{G_2}$+ $4l_{G_T}$)M+0A+1H	$3l_{G_1} + 5l_p$ (> 1375 b)	private key update	7 phases: Setup, Join, Sign, Verify, Open, Judge, Link
EH [82]	yes	0P+($2l_{G_1}$+$4l_{G_T}$)E + ($3l_p$+$2l_{G_1}$+$4l_{G_T}$)M +7A+1H + RSAverify	4P+($6l_{G_T}$)E + ($6l_{G_T}$)M+0A+1H + RSAverify	$2l_{G_1} + 4l_p$ + *token* (> 1020 b + 3072b RSA signature)	revocation list	9 phases: Setup, Group key generation, Token key generation, Join, Token generation, Sign, Revoke, Verify, Link
ACJT [33]	no	12E+11M+6A+1H	10E+10M+4A+1H	$3l_n + 4l_z + 1l_c$ (> 7328 b)	none (credential update / revocation list defined in [34])	5 phases: Setup, Join, Sign, Verify, Open
CG [53]	no	10E+9M+9A+1H	10E+8M+3A+1H	$4l_n + 4l_z + 1l_c$ (> 8352 b)	credential update / revocation list	6 phases: Key generation, Join, Sign, Verify, Open, Revoke
IMSTY [112]	no	7E+($8l_{ec}$+$11l_n$)M +10A+1H	7E+($8l_{ec}$+$6l_n$)M +6A+1H	$2l_n + 3l_z + 5l_{ec} + 1l_c$ (> 6095 b)	credential update	9 phases: Setup issuing manager, Setup revocation manager, Setup opening manager, Join, Sign, Verify, Open, User revocation, Update
HM GS [102]	no	9E+10M+4A+1H	10E+6M+0A+1H	$3l_n + 4l_z + 1l_c$ (> 7328 b)	revocation list	5 phases: Setup, Register, Sign, Verify, Revoke
Ring signature Liu *et al.* [134]	no	$(5+N)$E+$(4+N)$M	$(4+N)$E+$(3+N)$M	$O(N)$, $N + 3$ $3l_n + 4l_z + 1l_c$ (> 7328 b)	revocation list	5 phases: Setup, Register, Sign, Verify, Revoke

group signature consists of only 6 elements but it is combined with a token that is signed by using a public cryptography scheme, e.g., 3072-bit RSA. Thus, a signer and a verifier have to firstly verify the public cryptography signature of the token during the group signing and verification phases. Further, the group signatures become publicly linkable if signers sign more than once per time period. The EH scheme [82] can be useful for scenarios that require the linkability of signatures in periods, e.g., VANET applications supporting the short-term linkability.

The most complex schemes are the EH scheme [82] and the IMSTY scheme [112] that consist of 9 algorithms. In contrast, the BBS scheme [43] has only 4 basic phases. More complex schemes usually provide more security capabilities, e.g., signatures linking and open signer identity. In general, pairing-based ADS schemes have shorter signatures and requires less communication/memory cost than non-pairing-based ADS schemes. Nevertheless, the above results are only theoretical. The practical evaluation which is based on real implementations and the experimental measurement is presented in Section 3.3.1.

2.5 Post Quantum Public Key Cryptographic Schemes

Conventional and anonymous digital signature schemes are often based on the hardness of the Integer Factoring Problem (IFP) or the Discrete Logarithm Problem (DLP). Further, many those schemes are based on the variants of the IFP or DLP problems such as RSAP, DDHP, CDHP, ECDLP, DLIN or BDHP. Nevertheless, quantum computers using the Shor's algorithm are able to effectively solve all these problems. Hence, if functional quantum computers will be invented then the security of schemes based on mentioned problems could be broken.

Post-Quantum Cryptography (PQC) introduced in [39] describes the schemes that should resist to the quantum computers. Many designed PQC schemes have been already implemented and tested, e.g., the New Hope key exchange scheme has been tested in the Google Chrome Canary web browser. The quantum-resistant cryptographic library called liboqs has been integrated into the openssl library. Finally, National Institute of Standards and Technology (NIST) has released a call for proposals in order to solicit, evaluate, and standardize post-quantum schemes. The successful submissions of quantum-resistant public key encryption algorithms, key agreement mechanisms, and digital signature schemes will offer quantum-safe alternatives to currently used conventional cryptosystems such as RSA or ECDSA. NIST estimates that a draft standard with finally chosen schemes will be available between 2023 - 2025.

In general, post-quantum public key cryptosystems can be divided into five main categories: hash-based cryptography, code-based cryptography, multivariate cryptography, lattice-based cryptography and isogeny-based cryptography. These categories are defined as follows:

- **Hash-based cryptography** - these schemes are based on the security of hash functions (as a one-way function) and require less security assumptions than number-theoretic signature schemes (e.g., RSA, DSA). Ralph Merkle in 1979 introduced the Merkle Signature Scheme (MSS) [150] that is based on a one-time signatures (e.g., the Lamport signature scheme [125]) and uses a binary hash tree (a Merkle tree). MSS is resistant against quantum computer algorithms. More details can be found in a survey on hash-based schemes [52].
- **Code-based cryptography** - these cryptosystems are based on error correcting codes to construct a one-way function. The security is based on the hardness of decoding a message which contains random errors and recovering the code structure. The McEliece public key encryption scheme [149] is based on binary Goppa codes with high error correction capability and works with matrices. A receiver secretly chooses a private key that is a binary Goppa code. The corresponding public key is generator matrix $\mathbf{G}$ that describes a scrambled and randomly permuted variant of the Goppa code. A sender first encodes the plain text using $\mathbf{G}$ and adds t random errors during the encryption. Then, the receiver who knows the private key (the hidden algebraic structure of the Goppa code) is able to correct the errors and recover the message. The McEliece scheme [149] is still considered as secure for 40 years. The Niederreiter cryptosystem [156] as a McEliece variant provides both encryption and signature schemes. Nevertheless, many McEliece variants require large public keys. The work [175] presents the introduction of code-based cryptography and its perspectives.
- **Multivariate cryptography** - these schemes are based on systems of multivariate polynomial equations over a finite field $\mathbb{F}$.There are several variants of multivariate cryptography schemes based on Hidden Field Equations (HFE) trapdoor functions [161] such as the Unbalanced Oil and Vinegar Cryptosystems (UOV) [119]. UOV are used for signatures. Other examples of multivariate public-key cryptosystems (MPKC) are the Rainbow scheme[78] and Tame Transformation Signatures [63]. More about current state of the multivariate cryptography schemes can be found in the paper of [77].
- **Lattice-based cryptography** - these schemes are based on lattice-based computational problems, e.g., the Shortest Vector Problem (SVP) and the Ring Learning With Errors (RLWE) problem. A lattice $L \subset R^n$ is defined as the set of all integer linear combinations of basis vectors. Lattice-based public key schemes are used for public key encryption, key exchange, signature and hash functions. Well known cryptosystems are the Frodo scheme [46] and Ring-

Learning with Errors (Ring-LWE) schemes such as NTRU [107], New Hope [32] or Kyber [47]. The paper [154] analyzes lattice-based schemes in more details, investigates their properties and surveys existed implementations.

- **Isogeny-based Cryptography** - these schemes are based on supersingular elliptic curve isogenies that are secure against quantum adversaries. These schemes are secured under the problem of constructing an isogeny between two supersingular curves with the same number of points. Isogeny-based schemes may serve as digital signatures or key exchange such as Supersingular Isogeny Diffie-Hellman (SIDH) scheme [114]. More about schemes based on supersingular isogeny problems can be found in [87].

In the next chapter, Section 3.4 presents the performance assessment and feasibility of various PQC schemes on smartphones and small devices.

3 Assessment of Cryptographic Schemes on Constrained Devices

The goal of the chapter is to evaluate the performance of chosen cryptographic primitives and schemes on various constrained devices. The chapter also discusses the feasibility of common, privacy-preserving and post-quantum cryptography on various devices.

This chapter contains various results from selected author's publications such as [14], [27], [20], [2], [11] and [22] that are focused on the performance assessment of state of the art and conventional cryptographic schemes on constrained and small devices.

The structure of the chapter is as follows: In the first section, various constrained devices are defined. This section also discusses the security requirements of applications such as IoT that employ constrained devices. The second section describes the study of conventional cryptographic schemes on chosen constrained devices. The amended results have been published in the journal with impact factor [14] and the conference paper [27]. Further, this section presents the investigation of cryptographic primitives and schemes on smart cards. These results have been published in the conference papers [20] and [2]. The third section focuses on privacy-preserving schemes on chosen constrained devices and anonymous digital signature schemes on handheld devices. The amended results have been published in the journal with impact factor [14] and international journal indexed in Scopus [11]. The final section of this chapter provides the evaluation of post-quantum public key cryptographic schemes on small devices. The results have been published in the international journal indexed in Scopus [22].

3.1 Constrained Devices in the Internet of Things

Current heterogeneous communication environment such as IoT and modern communication networks consists of various devices and communication protocols. Interconnected devices usually have different hardware specifications with various computational and memory abilities. Further, various software and communication protocol characteristics (e.g., bandwidth, delay, message size) could restrict the deployment of some cryptographic protocols and schemes on end-point devices in various applications and systems. The end-point and communication devices can be characterized as follows:

- **Microcontrollers** - Microcontrolers in ICT could work as actuators and data collectors. Microcontrollers as sensor nodes are usually employed in wireless sensor networks. These devices collect, process and forward data that are sensed in a certain environment. Smart sensor nodes could have various performance characteristics. The CPU frequency reaches up to several hundreds of MHz. Nevertheless, the memory capabilities are usually reduced due to minimizing the power consumption. Some sensor devices usually offer cryptographic coprocessors for accelerating symmetric ciphers such as AES and DES. Microcontrollers are usually considered as very constrained devices.

- **Smart cards** - Smart cards can be used as small authentication items in access control systems, e-ticketing and e-payments. Further, smart cards in the SIM size can securely store the cryptographic material and perform various cryptographic operations in the embedded devices and microcontrollers. Such chip cards are often called as SAM modules or TPM modules. The CPU frequency of smart cards usually reaches up to several tens of MHz. Smart cards also offer a sufficient memory storage for various services and applications. Many platforms of smart cards provide Application Programming Interfaces (APIs) of conventional cryptographic schemes such as AES, DES, RSA, SHA, MD5, PRNG. In addition, there are programmable smart cards (Basic card, JAVA card, MultOS card, .NET card) that can be used for the implementation of advanced cryptographic schemes based on modular arithmetic. Smart cards are considered as constrained devices.
- **Single-boards, small computers and embedded devices** - Single-boards, small PCs and embedded devices are widely used in industrial networks (SCADA systems) and IoT. The performance characteristics of these devices are usually very various. The CPU frequency could be from several hundreds MHz to units of GHz. The advanced cryptographic schemes can be easily implemented on these devices due to advanced operating systems that support various cryptographic libraries. These devices are considered as constrained and middle-performed devices.
- **Small handheld devices** - Smartphones, tablets and mobiles represent small handheld devices that can be used as authentication items and/or can host many secure services such as e-payment, geo-localization and others. Nowadays, these devices provide strong performance and can offer a secure storage of data in the secure element. The CPU frequency reaches up to units of GHz. The operating systems on these devices such as Android, iOS, Windows enable developers to use various conventional cryptographic methods, libraries and cryptographic schemes. Due to the support of plenty cryptographic and math functions, the handheld devices can host advanced cryptographic schemes, such as anonymous digital signatures. Handheld devices are considered as middle-performed devices.

- **Powerful nodes** - These devices are represented by personal computers and servers with powerful computation parameters. Their CPU frequencies reach up to units of GHz and more cores are used. These devices can manage expensive computational procedures and tasks such as generating cryptographic parameters, revocation procedures, bilinear pairing operations etc. These devices are considered as high-performed devices.

3.1.1 Security and Privacy in the Internet of Things

The Internet of Things paradigm has been described in many papers, e.g., [35], [96], [36]. IoT can be defined as a highly interconnected network of heterogeneous entities. Figure 3.1 depicts an example of an IoT environment and shows some technologies and appliances that can be used in IoT.

The machine-to-machine and machine-to-human communications are usually based on the IP protocol which can cause that billions of IoT objects become part of the Internet. Therefore, the security in IoT has to be addressed due to the high possibility of security risks such as eavesdropping, unauthorized access, data modification, data forgery and unauthorized remote tampering with devices. For example, attackers can turn on smart devices and heating systems to trigger a collapse of the power grid. Furthermore, attacks against routing protocols can be performed in IoT infrastructure and applications, e.g., Sybil attacks [218], the sinkhole attack [56].

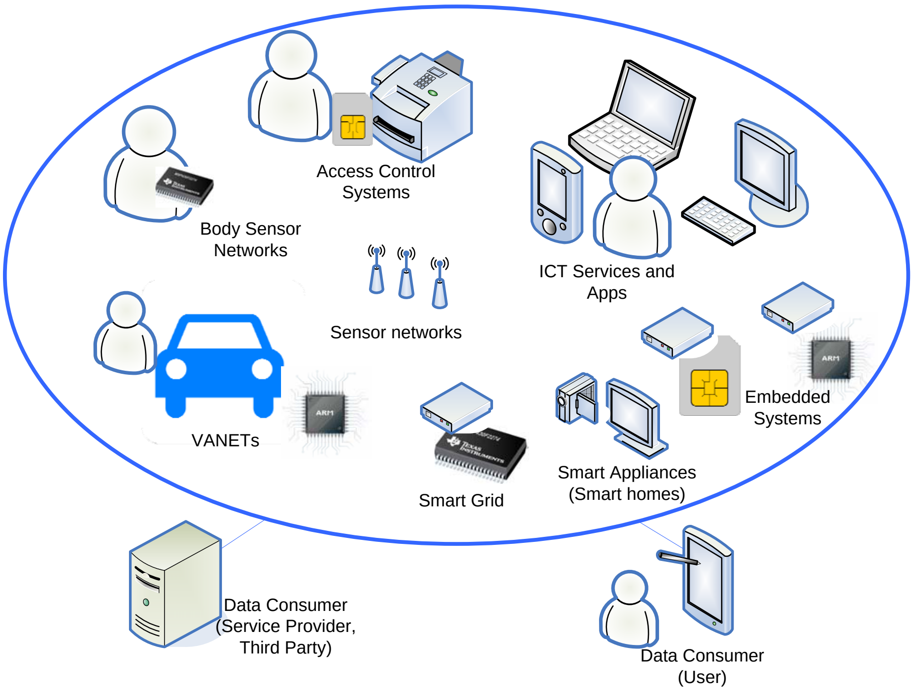


Fig. 3.1: Technologies and applications in the Internet of Things environment.

Security solutions designed for IoT environments have to deal with heterogeneous IoT entities with various hardware specifications. In IoT, the most spread devices are usually resource-constrained devices because of their low cost. These devices usually employ Constrained Application Protocol (CoAP) [179] at the application layer. The security solutions in IoT have to provide the authentication and authorization of IoT nodes (things, users, servers, objects) and data authenticity, confidentiality, integrity and freshness. The security solutions are usually implemented at network, transport and application layers in IoT. Figure 3.2 depicts the IoT layers and the security protocols that can be used in IoT, e.g., IPSec, Host Identity Protocol (HIP), Transport Layer Security (TLS) protocol, Datagram Transport Layer Security (DTLS) protocol and Slim Extensible Authentication Protocol Over Local Area Networks (SEAPOL). For example, Extensible Authentication Protocol (EAP) messages that ensure Point-to-Point authentication at the link layer can be transfered over SEAPOL or Trust Extension Protocol for Authentication of New deployed Objects and sensors through the Manufacturer (TEPANOM) [162].

Besides the basic security properties, privacy has to be addressed in IoT as well. Many IoT services and applications provide sensitive and personal information that are exposed, and can be misused by an attacker. Unsecured sensitive data can leak to third parties. The concept of privacy may differ but it should protect user's personally identifiable information and keep a certain degree of anonymity, unlinkability and data secrecy.

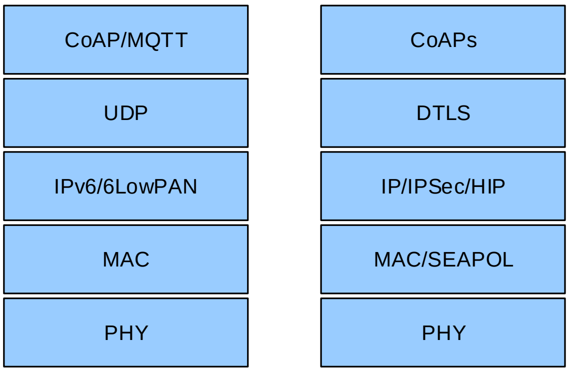


Fig. 3.2: The Internet of Things layers connected with the security protocols.

A lot of privacy-preserving solutions are designed for powerful computers and nodes in the Internet. The privacy-preserving solutions are usually based on computationally expensive cryptographic primitives, such as bilinear pairing, exponentiation of big numbers. Due to this fact, it is still an open challenge to design a secure, efficient and privacy-preserving solution for the Internet of Things that works mostly with the constrained devices.

The next sections present the performance assessment of the conventional cryptographic schemes and primitives that are considered in security solutions designed for the Internet of Things applications which use constrained devices. Then, the privacy-preserving cryptographic protocols and anonymous digital signature schemes are evaluated on various devices. Finally, the feasibility of post-quantum public key cryptography on handheld and single-board devices is analyzed.

3.2 Conventional Cryptography on Constrained Devices

This section presents the performance assessment of widely used cryptographic primitives, schemes and some important modular arithmetic operations which are used in security solutions implemented in IoT. Besides the performance of the operations, the section also discusses their memory requirements.

The operations are implemented and measured on various platforms such as microcontrollers, smart cards, smart phones, single-boards that are used in the IoT environment. The most used cryptographic primitives and schemes are implemented, for example, Advanced Encryption Standard (AES), Secure Hash Algorithms (SHA-1 and SHA-2), asymmetric encryption and signature scheme RSA, random number generator functions and elliptic curve point multiplication used in ECDH and ECDSA schemes. These cryptographic primitives and operations are employed in many IoT security solutions. For example, AES is used for data encryption in the DTLS protocol [166], the Lithe-DTLS protocol [165] and the OSCAR solution [197]. Random number generator and hash functions are used for secret key derivation functions and are employed in many authentication and key establishment protocols. The RSA encryption and modular arithmetic operations are used in strong authentication schemes, key establishment protocols (DH, ECDH) and privacy preserving protocols such as [109], [192] and [185]. The tested cryptographic primitives and schemes are set as follows:

- AES 128b - an encryption of a 128-bit plaintext by the symmetric cipher AES with a 128-bit key in the ECB mode.
- SHA1 4256b - a hash function SHA1 with a 4256-bit plaintext.
- SHA2 8448b - a hash function SHA2 with a 8448-bit plaintext.
- RSA ver/enc 1024b - a RSA exponentiation operation with a 1024-bit modulo, a 1024-bit base (data) and a small public exponent ($e = 65537$). This operation is used for data encryption or for the verification of a RSA signature.
- RSA sig/dec 1024b - a RSA exponentiation operation with a 1024-bit modulo, a 1024-bit base (data) and a private exponent (1024 bits). This operation is

used for data decryption or for RSA signing.

- RSA ver/enc 2048b - a RSA exponentiation operation with a 2048-bit modulo, a 2048-bit base (data) and a small public exponent ($e = 65537$). This operation is used for data encryption or for the verification of a RSA signature.
- RSA sig/dec 2048b - a RSA exponentiation operation with a 2048-bit modulo, a 2048-bit base (data) and a private exponent (2048 bits). This operation is used for data decryption or for RSA signing.
- RND 160b - a random number generator function producing a 160-bit random number.
- RND 560b - a random number generator function producing a 560-bit random number.
- ECPM 128b Fp - an elliptic curve point multiplication operation with 128-bit elliptic curves.

The performance of these primitives is measured on various devices, namely, microcontrollers of family MSP430fX, programmable chip cards (JAVA and MultOS) and devices with ARM processors to get an overview of how the cryptographic primitives affect the IoT applications and services. For example, MSP430fX microcontrollers are widely used in devices employed in the smart grid systems (smart meters), home automation systems (smart thermostats, smart air condition controllers), industrial embedded systems and sensor networks. The smart cards are used in access control systems or as SAM (Secure Access Module) in embedded devices or in ICT devices which need a secure module. The ARM platform is often used, e.g., in industrial embedded systems and vehicular ad hoc network systems. Also many handheld devices (smartphones, tablets etc.) contain ARM controllers. The devices have different technical specifications which are summarized in Table 3.1.

Tab. 3.1: Technical Specifications of Devices Used.

Designation	Device	Processor	RAM size (RAM)	Storage size (ROM / EEPROM / Flash)
8 MHz Microcontroller	ultra-low-power microcontroller MSP430F149	16-bit CPU with 8 MHz	60 kB	60 kB
20 MHz Microcontroller	microcontroller MSP430F6638	16-bit CPU with 20 MHz	18 kB	256 kB
30 MHz JAVA card	smart card NXP JCOP CJ3A080v24	16-bit CPU with 30 MHz	6 kB	200 + 80 kB
33 MHz MultOS card	smart card ML3-36k-R1	16-bit CPU with 33 MHz	1088 + 960 B	280 + 60 kB
700 MHz ARM	single-board computer Raspberry Pi model B+	32-bit ARM11 Single-core with 700 MHz	512 MB	8 GB
2260 MHz ARM	smart phone Nexus 5 LG	32-bit ARMv7 Quad-core with 2260 MHz	2 GB	16 GB

A delay caused by processing the cryptographic overhead can negatively affect the quality of services in some IoT applications. Therefore, it is defined a threshold T = 300 ms that is considered as a maximum latency for cryptographic operations and schemes tolerated by real-time IoT applications. Hence, IoT applications can be divided into two groups:

- Real-time IoT applications (*time* =<T), e.g., patient monitoring applications in body sensor networks that must send vital data in real time, some applications in Vehicular Ad hoc Networks (VANET) that send real-time notifications (break alerts, proximity alerts) to drivers on the road.
- Non-real-time IoT applications (*time* >T), e.g., power consumption monitoring in smart grid, traffic jam monitoring in VANET.

3.2.1 Performance Assessment of Cryptographic Primitives on Resource-Constrained Devices

Resource-constrained devices are considered as the most employed devices in the IoT infrastructure. This subsection presents the performance results of cryptographic primitives that are implemented on resource-constrained devices, namely, 8-MHz microcontroller, 20-MHz microcontroller, 30-MHz JAVA card and 33-MHz MultOS card. The technical specifications of these devices are in Table 3.1.

The implementations of cryptographic functions on the microcontrollers are written in the C programming language (C). It is wrapped some existed libraries such as LibTomCrypt (libtom.net), OpenSSL (openssl.org) and PolarSSL (polarssl.org). External crypto-accelerator hardware modules are not used. Our test application for the JAVA card is written in the JAVA language (Java Card Open Platform v2). The chosen JAVA card provides some cryptographic functions but it does not contain any modular arithmetic functions with big integers. On the other hand, the MultOS card with the MultOS card operating system provides both crypto APIs and some modular arithmetic functions. Our test application for the MultOS card is written in the C programming language.

The methodology of the measurement is different for microcontrollers and for smartcards. The performance of the cryptographic operations on the microcontrollers is measured as the number of cycles and this number is recomputed on the execution time (1 cycle takes 1μs on a 1-MHz processor). The execution times of the operations on smartcards are average values computed from 10 iterations.

In Figure 3.3, the execution times of the AES-128b, RND-160b and RND-560b operations on the resource-constrained devices are depicted. These operation takes only few milliseconds on these devices. The AES encryption operation of one 128-bit

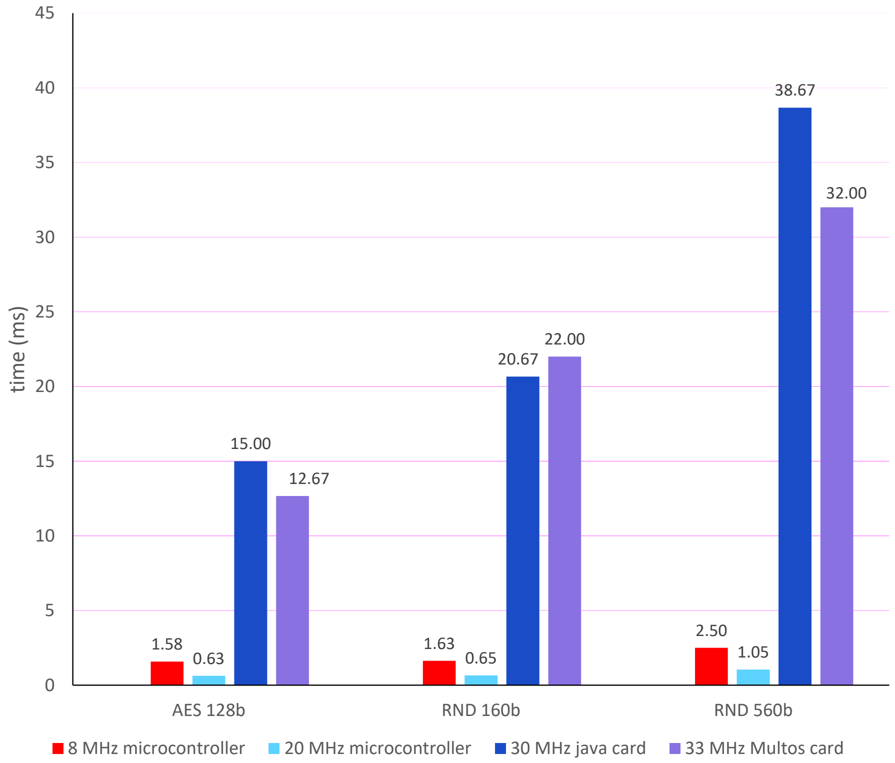


Fig. 3.3: The execution times of AES 128b, RND 160b and RND 560b operations on resource-constrained devices.

data block and the random number generation operations of 160-bit or 560-bit numbers are more efficient on microcontrollers than on smart cards. The initialization of card's operating system and APDU communication between a chip card and a reader device cause time delay. Therefore, chip cards with higher CPUs frequencies than microcontrollers need more execution time for some cryptographic operations in comparison with microcontrollers. Nevertheless, the measurement shows that AES and random number generator functions are efficient and can be implemented into the IoT security solutions that are run on constrained devices.

Figure 3.4 depicts the execution times of hash functions SHA1-4256b and SHA2-8448b on microcontrollers and smart cards. The SHA-1 and SHA-2 functions take from tens to hundreds milliseconds on these constrained devices. Many authentication and cryptographic schemes are based on hash functions. The use of such schemes that are performing many hash functions or hashing the large data structures (several kB) can be difficult and problematic in the IoT infrastructure that employs constrained devices.

Figure 3.5 shows the execution times of RSAsig/dec and RSAver/enc operations on the low-performance devices. The RSA scheme uses modular exponentiation operations. The RSA operations with public keys take hundreds milliseconds on

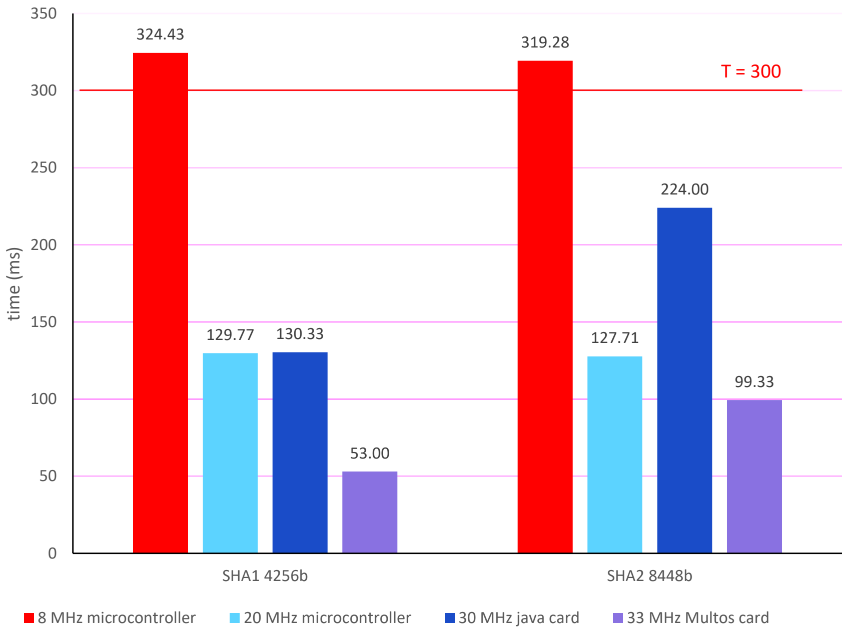


Fig. 3.4: The execution times of SHA1 4256b and SHA2 8448b operations on resource-constrained devices.

microcontrollers. The RSA operations with private keys (e.g., a 1024-bit exponent) take several seconds on microcontrollers. The JAVA card and MultOS card provide direct RSA APIs that are optimized. Due to this fact, the RSA operations on these smart cards take from tens to hundreds milliseconds.

Modular arithmetic operations such as modular multiplication or modular exponentiation take from several tens to hundreds milliseconds on common smart cards. For example, modular multiplication with 1024-bit numbers takes about 546 ms and modular multiplication with 2048-bit numbers takes about 998 ms on the java card. Nevertheless, some smart cards offer co-processors and direct functions to enhance the performance of some modular arithmetic operations and asymmetric cryptographic operations such as RSA and ECC operations. For example, one ECDH operation with a 128-bit $\mathbb{F}_p$ elliptic curve takes about 104 ms on the JAVA smart card. Further, the MultOS card which supports the big number operations needs only 28 ms to compute one 1024-bit modular multiplication and the MultOS card ML2-80K-65 needs only 58 ms to compute one modular exponentiation with a 1024-bit modulo, a 1024-bit base and a 160-bit exponent.

Many symmetric ciphers, e.g., AES, are fast enough to be implemented into security solutions that run on constrained devices in the IoT infrastructure. On the other hand, security solutions based on asymmetric cryptographic operations, e.g., RSA, ECDH, ECDSA, and big integer modular arithmetic operations (multiplication, exponentiation) need more time to execute, e.g., from tens to hundreds milliseconds

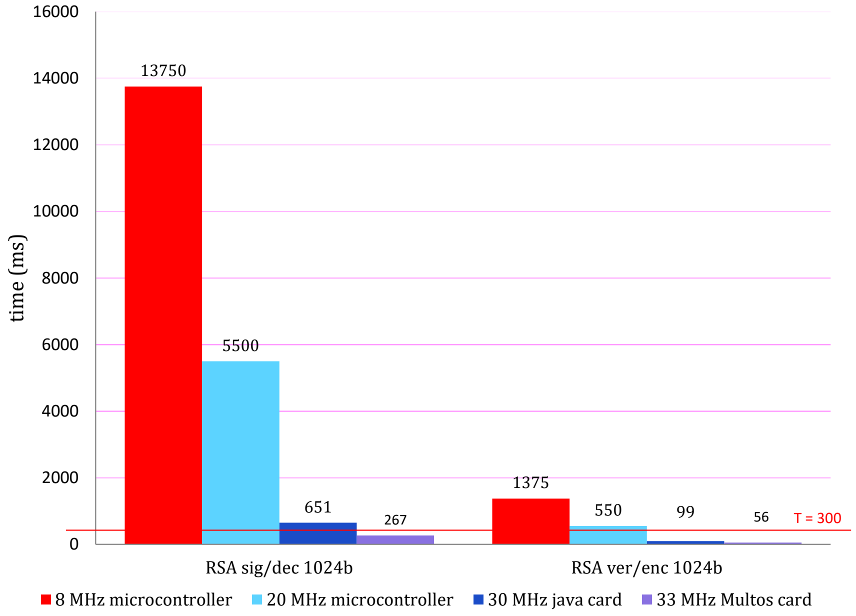


Fig. 3.5: The execution times of RSAsig/dec and RSAver/enc operations on resource-constrained devices.

on constrained devices. For example, RSA signature scheme that takes several seconds on microcontrollers is not suitable for real-time IoT applications (e.g., patient monitoring). On the other hand, smart meters with constrained microcontrollers that usually send power consumption data can use RSA signing because the data are sent only few times per day.

3.2.2 Performance Assessment of Cryptographic Primitives on Programmable Smart Cards

This subsection focuses only on programmable smart card platforms such as JAVA Cards, MultOS Cards, .NET Cards and Basic Cards that support various basic security and cryptographic features. Table 3.2 shows the support of cryptographic functions on these chosen smart card platforms that also including MIFARE SAM AV2 Cards that provide basic cryptographic functions.

The performance assessment of basic cryptographic functions on main smart card platforms (JAVA Cards, MultOS Cards, Basic Cards, .NET Cards) is presented in Table 3.3. These experimental results have been published in our paper [140]. The values are averaged from 10 measurements (unexpected peaks are omitted). The smart card platforms do not enable us measurements in cycles, hence, the results are in ms. The operation runtimes also include delays caused by card communication and OS initialization. The communication delay depends on the data length, Tab. 3.2: Cryptography and Math Support of Smart Card Platforms.

Platform	JAVA Cards	Basic Cards	MultOS Cards	.NET Cards	MIFARE SAM AV2 Cards
Symmetric Crypto API	DES, TDES, AES (keys up to 256b), SEED, CBC/ECB modes, CMAC, HMAC	DES, TDES, AES (keys up to 256b), CBC/CFB/OF-B/EAX modes, OMAC	DES, TDES, AES (keys up to 256b), SEED, CBC/ECB modes	DES, TDES, AES (keys up to 256b), ECB/CBC modes	DES, TDES, AES
Asymmetric Crypto API	RSA (up to 4096b), DSA (up to 1024b), ECDH, ECDSA (up to 512b)	RSA (up to 4096b), ECDSA, ECDH, ECNR signature (up to 521b)	RSA (up to 2048b), ECDH, ECDSA, ECIES (up to 512 b)	RSA (up to 2048b)	RSA (up to 2048b)
Hash functions API	MD5, RIPEND160, SHA-1, SHA-2, SHA-3 (JC 3.0.5)	SHA-1, SHA-2 (up to 512b)	SHA-1, SHA-2 (up to 256b)	MD5, SHA-1, SHA-2 (up to 256b)	SHA-1, SHA-2 (up to 256b)
Random number generator API	Pseudo RND, TRNG (JC 3.0.5)	4B RND function, TRNG	TRNG	Pseudo RNG, TRNG	TRNG
Modular arithmetic API	not supported, exponentiation (JC 3.0.5), ECDH point multiplication (JC 3.0.5)	supported (up to 16 kB)	supported (up to 2048 bits)	not supported	not supported

communication interface and other parameters (e.g., conductance/RF field strength, modulation, signal gain, threshold level). If the cryptographic operation is quite fast, then the delay is more significant in total time. The time for sending 128-bit data by APDU to/from a card is 6.6 ms on a contactless smart card and 9.4 ms on a contact smart card.

Further, the performance analysis and comparison of different elliptic curves on smart cards are investigated. The results of ECDH, ECDSA (Sign) and ECDSA (Verify) operations are depicted in 3.6, 3.7 and 3.8. More details and results about basic EC arithmetic are published in our conference paper [2].

3.2.3 Performance Assessment of Cryptographic Primitives on Middle and High-Performed Devices

This subsection presents the performance results of cryptographic primitives and schemes that run on devices with ARM chips having several hundreds MHz which are used in embedded devices, mobiles or control devices. It is assumed that devices based on the ARM platform are also widely used in the IoT infrastructure. These devices are much more computational powerful and enable us to use more cryptographic libraries and functions than microcontrollers or smart cards. Our measurement includes the devices with 700 MHz and 2260 MHz ARM chips. Their

Tab. 3.3: Performance Assessment of Cryptographic Functions on Smart Card Platforms.

Platform (card type):	JAVA Card (JAVA Card J3D081)	Basic Card (Basic Card ZC 7.6 Rev D)	MultOS Card (MultOS Card ML3)	.NET Card (.NET Smart Card V2+)
Average time of one operation [ms] (including communication and OS initialization)				
Symmetric Crypto API:				
AES - 128-bit key and CBC mode on 128-bit message	15	12	13	32
Asymmetric Crypto API:				
RSA - sign - 1024-bit	192	72	69	179
RSA - sign - 2048-bit	725	397	267	625
ECDSA - sign - 256-bit	104	58	174	not-supported
ECDSA - verify - 256-bit	112	334	228	not-supported
Hash functions API:				
SHA-256 on 128-bit message	14	13	11	13
Random number gen. API:				
Random of 160-bit	21	12	32	33
Modular arithmetic API:				
Multiplication with 2048-bit bigints	998	19	29	697
Exponentiation with 2048-bit bigints	478	1180	385	820

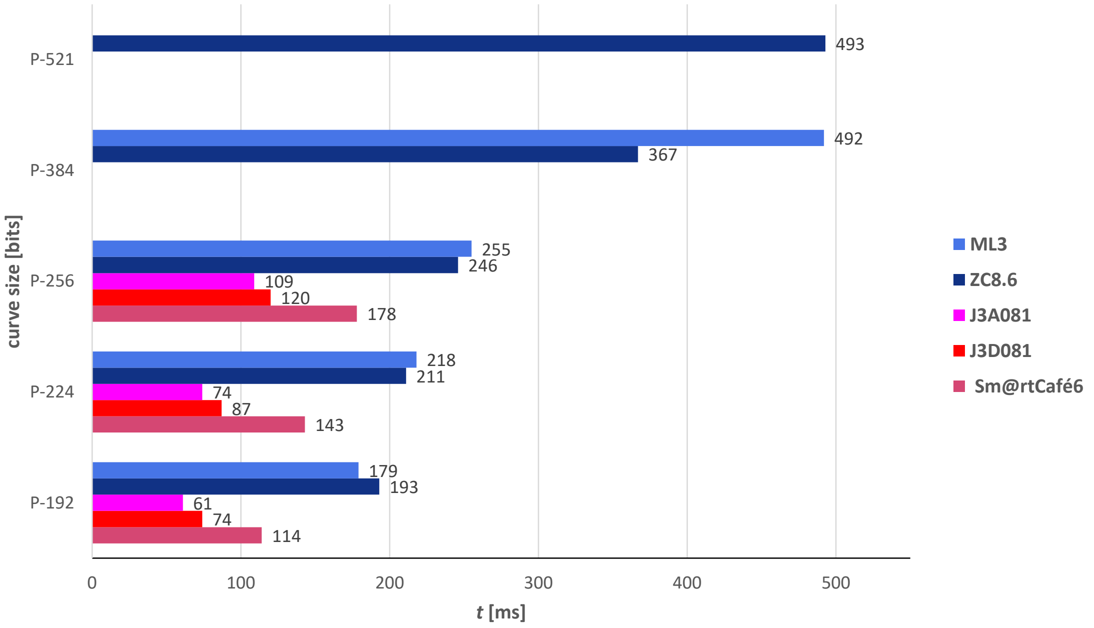


Fig. 3.6: Efficiency of ECDH protocol on different smart card platforms.

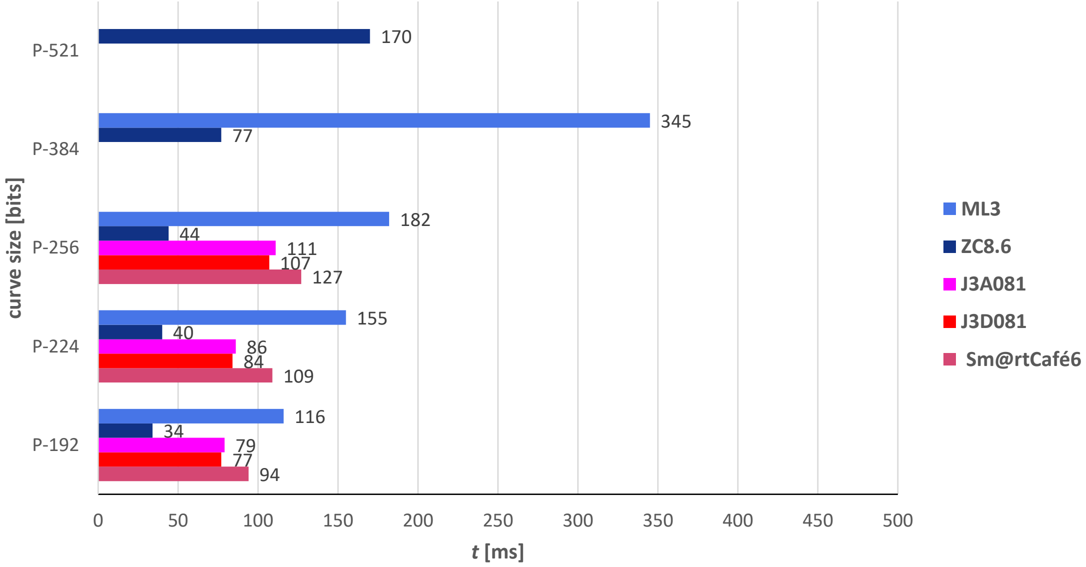


Fig. 3.7: Efficiency of ECDSA (sign) on different smart card platforms.

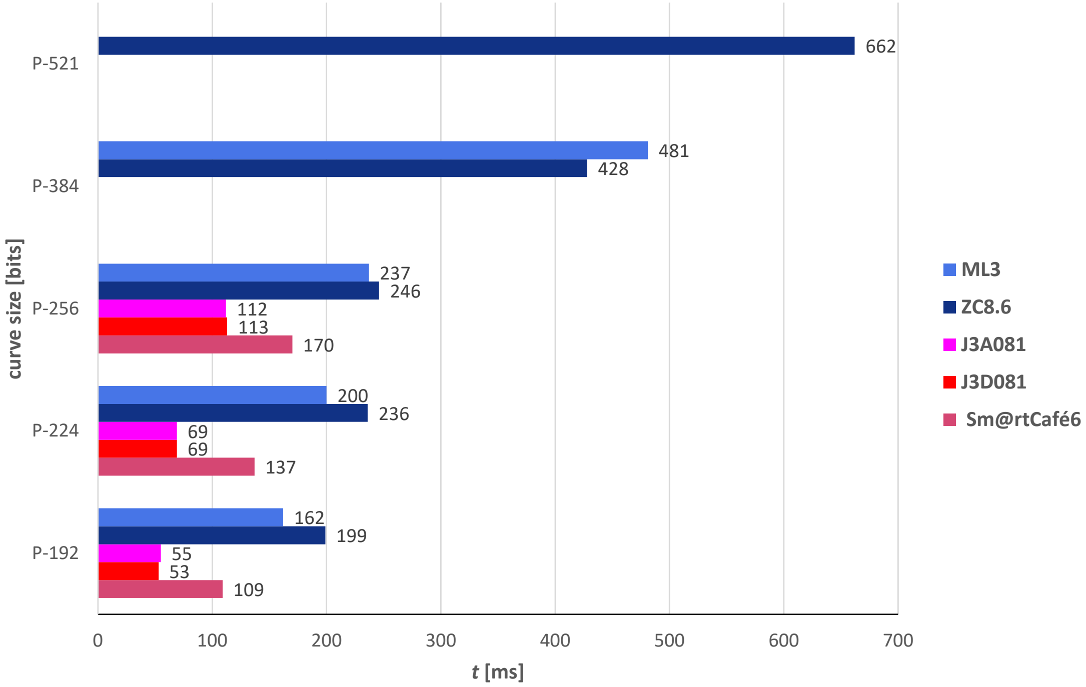


Fig. 3.8: Efficiency of ECDSA (verify) on different smart card platforms.

specifications are described in Table 3.1. The 700 MHz ARM device is a single-board computer with a Linux OS (Raspbian) where is run our test application written in the JAVA programmable language. To obtain the fast modular arithmetic operations, it is called a C math library, namely GNU Multiple Precision Arithmetic Library (GMP) via JAVA Native Interface (JNI). The 2260 MHz ARM device is a smartphone with Android OS (Android 4.4 KitKat). Our test application is written in the JAVA programming language by using the Android Software Development Kit (SDK). The platforms of both ARM devices provide many crypto APIs and modular arithmetic operations via big integer APIs.

Figure 3.9 depicts the execution times of AES-128b encryption, hash functions SHA1-4256b and SHA2-8448b on the ARM devices. These operations take few milliseconds on the single-board computer with 700 MHz ARM. On the smartphone with 2260 MHz ARM (Quad-core), these operations take about one hundred μs.

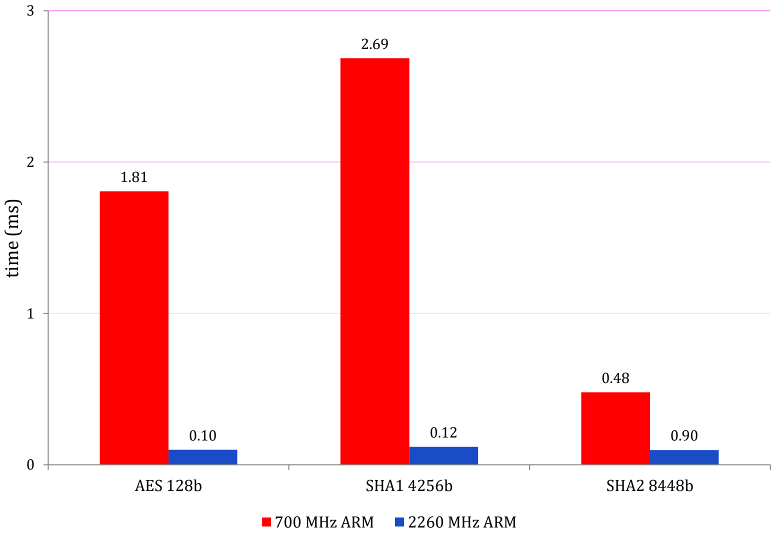


Fig. 3.9: The execution times of AES 128b, SHA1 4256b and SHA2 8448b operations on middle and high-performed devices.

Figure 3.10 shows the execution times of RND, RSAsig/dec, RSAver/enc and ECDH operations on the ARM devices. The smartphone is able to compute the RSA operations within hundreds μs. The single-board computer needs several tens milliseconds for computing the RSA operations. The random number generator functions generate secret random numbers within few milliseconds. The execution time depends on a generation method which is used in a device.

On the ARM devices, modular arithmetic operations such as modular multiplication or modular exponentiation take from hundreds μs up to tens milliseconds. The times depend on the sizes of inputs, a programing language and a cryptographic/math library. For example, modular multiplication with 2048-bit numbers takes

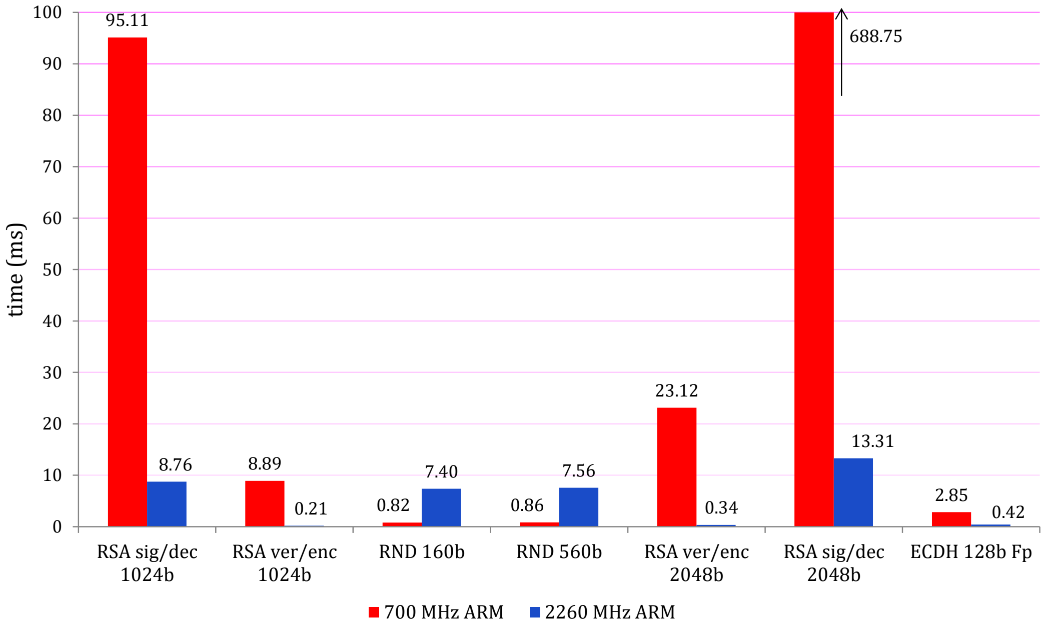


Fig. 3.10: The execution times of RND 160b, RND 560b, RSA sig/dec 1024b, RSA ver/enc 1024b, RSA sig/dec 2048b, RSA ver/enc 2048b and ECPM 128b $\mathbb{F}_p$ operations on middle and high-performed devices.

about 0.1 ms and modular exponentiation with a 2048-bit modulus and a 160-bit exponent takes 17 ms on the 700 MHz ARM employing the GMP library written in the C programmable language.

Our measurement proves that robust security solutions which include asymmetric cryptography and big integer modular arithmetic operations can be implemented on many ARM devices deployed as the IoT nodes. These operations takes several ms and are easy to implement. Nevertheless, some cryptographic operations such as bilinear pairing operations that are widely used in many privacy-preserving, identity based or group signature schemes are computationally expensive. The bilinear pairing operations take from tens milliseconds to few seconds on ARM devices. The runtimes of pairing operations mainly depend on libraries that support pairings, e.g., JPBC, PBC, MIRACL, MCL, RELIC. For example, one asymmetric bilinear pairing operation (175-bit curves) performed by the JPBC library (written in the JAVA programmable language) takes about 2.4 s on the smartphone (2260 MHz ARM) with Android OS. On the other hand, 64-bit ARM (Cortex-A53) microcomputer with the PBClib library (written in the C programmable language) computes one pairing operation (256-bit BN curves) in 247 ms and the TEPLA library (written in the C programmable language) needs only 20 ms to compute one pairing operation on the same device. Thus, pairing based cryptographic schemes are not suitable for the Android devices that use JAVA libraries, but optimized libraries written in

C enable developers to deploy pairing-based schemes also on ARM platforms. The optimization techniques such as the batch verification and the pairing precomputation can reduce the total number of pairing operations and decrease the overall runtimes of signing and authentication phases of pairing-based schemes. Our paper [7] presents more results and the performance analysis of pairing-based elliptic curve cryptography on constrained devices.

3.2.4 Memory Requirements of Conventional Cryptographic Primitives

Besides long execution times of some cryptographic operations, the security solutions have to deal with memory constrains of devices in IoT. RAM memory and code size requirements of the cryptographic schemes and primitives are various. Obviously, ARM devices and many smart cards provide enough RAM and a storage memory for cryptographic primitives. These platforms usually have larger RAM and a storage memory than most of microcontrollers. Nevertheless, the microcontrollers with small RAM and a storage memory are usually cheaper and therefore, these devices are widespread in the systems with a large number of nodes, e.g., home automation controllers, sensors, smart meters, etc. However, a small RAM memory could be an issue for the deployment of several cryptographic schemes on such devices. Figure 3.11 depicts the RAM consumption of our AES, RSA, ECDH and SHA2 implementations on MSP microcontrollers.

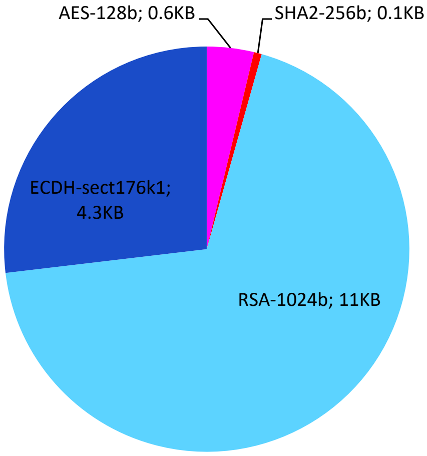


Fig. 3.11: The RAM consumption of cryptographic primitives on MSP microcontrollers.

The following text describes the RAM and storage memory requirements of our test implementations written in C on the MSP microcontrollers:

- Asymmetric cryptographic schemes - the RSA implementation takes almost 249 kB in a storage memory due to many functions wrapped from libraries (LibTomCrypt, GMP) and uses approximately 11 kB of a RAM memory due to large byte array structures for input parameters (1024 - 2048 bits per one) and variables. Therefore, the complete implementation of the RSA scheme is not feasible for microcontrollers with a small RAM memory ($<$ 10 kB). Further, we implement big integer modular arithmetic and elliptic curve (EC) operations by wrapping the OpenSSL library. The total code size of our implementation is 12412 bytes. RAM memory usage depends on the size of elliptic curves and the types of the arithmetic methods. Nevertheless, the solutions that use ECC and modular arithmetic operations need microcontrollers with a RAM memory at least 4 kB.
- Symmetric ciphers - the ciphers usually take from hundreds bytes to few kB in a storage memory and use approximately from tens to hundreds B of a RAM memory. For example, AES wrapped from the LibTomCrypt library takes 550 B in RAM. There are AES implementations that are optimized for RAM memory usage but the total number of cycles for one 128-bit encryption can be higher. Nevertheless, many ciphers such AES, XTEA, Noekeon can be implemented into microcontrollers with a small RAM memory, e.g., 1 kB RAM, because the operations of symmetric ciphers are repeated in rounds and work with smaller keys (e.g., 128 bits) than asymmetric ciphers (e.g., 1024 bits).
- Hash functions - these functions usually take few kB in a storage memory and use approximately tens to hundred B of a RAM memory, e.g., 107 B in RAM by using the SHA-256 function wrapped from the LibTomCrypt library. A small RAM consumption of the SHA2 function can be explained by using small variables (32-bits) and simple operations that are repeated in rounds. Many hash functions can be implemented into microcontrollers with a small RAM memory, e.g., 1 kB RAM.

3.3 Privacy-Preserving Cryptographic Schemes on Constrained Devices

A lot of the Internet of Things services sense and collect sensitive data such as an actual user location, personal data, vital data, medical data and so on. Therefore, privacy-preserving cryptographic schemes should be deployed in order to prevent privacy leakage. In the following subsections, the privacy-preserving techniques and their use in IoT infrastructures using constrained devices are analyzed and evaluated.

This analysis is based on the evaluation of cryptographic primitives on various devices which is presented in the previous section. The analysis focuses especially on cryptographic solutions and schemes that provide privacy. These cryptographic solutions are usually based on standard cryptographic operations (data encryption, hash functions) and modular arithmetic (multiplication, exponentiation). In addition, selected privacy-preserving solutions are implemented and their time execution on various devices are measured.

The following text provides the comparison of advanced privacy-preserving schemes such as Homomorphic Encryption (HE), Group Signatures (GS), Ring Signatures (RS) and Attribute-Based Signatures (ABS). These chosen cryptographic schemes are compared: the Paillier's homomorphic encryption scheme [159], the BBS group signature scheme [43], the DP group signature scheme [76], the ring signcryption (Li *et al.*) scheme [127], the attribute based signature (HM) scheme [101] and the attribute based encryption (GPSW) scheme [93].

In the comparison, three types of IoT devices are assumed: a resource-constrained IoT device (Chip card with the OS MultOS), a medium-performed IoT device (Mobile ARM device with Android 4.2) and a high-performed IoT device (PC with Windows 7). All these devices provide the sufficient space of RAM and storage memory for privacy-preserving schemes. Furthermore, the devices offer programmable platforms for loading own applications and libraries. Chosen privacy-preserving schemes are implemented and loaded on these devices. The technical specifications of the devices are listed in Table 3.4.

Tab. 3.4: Technical Specifications of Devices Used in the Comparison of Privacy-Preserving Techniques.

Type	Device	Processor	RAM size (RAM)	Storage size (ROM / EEPROM / Flash)
Chip card	smartcard ML3-36k-R1	16-bit CPU with 33 MHz	1088 + 960 B	280 + 60 kB
Mobile	smartphone Nexus i9250	32-bit ARM 2x 1200 MHz	1024 MB	8 GB
PC	personal computer Lenovo Think station E20	64-bit Intel Xeon CPU 8x 2.53 MHz	8 GB	16 GB

The selected schemes are implemented in three programing languages that depend on the device. The Android platform is used for the implementation and the tests on the mobile device. JAVA is used for the implementation and the tests on the PC device and the programming language C is used for the implementation and the tests on the chip card device.

Table 3.5 presents the experimental results for selected privacy-preserving techniques. The performance overhead on the devices is measured and the memory/communication overhead is estimated. The measurement focuses on the time of main operations/phases such as the encryption time (Enc), the decryption time (Dec), the signing time (Sig) and the verification time (Ver), the signcryption time (Signcrypt) and the unsigncryption time (Unsigncrypt). All time values are computed as the mean values from 10 iterations. Due to the complexity of the library that provides the bilinear pairing operations, pairing-based solutions are not implemented on the chip card device. The techniques provides at least the 80-bit security level. The length l_{G_1} describes the length of the group element $\in \mathbb{G}_1$ (e.g., 171 bits). l_{G_T} describes the length of the group element $\in \mathbb{G}_T$ (e.g., 1026 bits). l_p denotes the length of scalars in modulo p (e.g., 170 bits). l_n denotes the length of the RSA modulo n (e.g., 1024 bits). l_z denotes the length of the scalars of various lengths less than n (e.g., < 1024 bits). l_c denotes the length of the hash used (e.g., 160 bits). l_{ec} denotes the length of the elliptic curve element (e.g., 169 bits). AT denotes the size of an attribute set.

Tab. 3.5: Comparison of Privacy-Preserving Techniques.

Type of technique/Scheme	Performance overhead on Chip card	Performance overhead on Mobile	Performance overhead on PC	Memory and communication overhead
HE/Paillier's scheme [159] (1024 b parameters)	Enc = 488 ms; Dec = 746 ms	Enc = 48 ms; Dec = 90 ms	Enc = 20 ms; Dec = 45 ms	ciphertext size = $2l_z$ (e.g., 2048 b)
HE/Paillier's scheme [159] (2048 b parameters)	Enc = 799 ms; Dec = 1213 ms	Enc = 368 ms; Dec = 686 ms	Enc = 146 ms; Dec= 303 ms	ciphertext size = $2l_z$ (e.g., 4096 b)
GS/BBS scheme [43]	-	Sig = 11226 ms; Ver = 33153 ms	Sig = 215 ms; Ver = 518 ms	$3l_{G_1} + 6l_p$ (e.g., 1533 b)
GS/DP scheme [76]	-	Sig = 15778 ms; Ver = 32016 ms	Sig = 208 ms; Ver = 516 ms	$4l_{G_1} + 5l_p$ (e.g., 1444 b)
RS/Li scheme [127] (100 identities)	-	Signcrypt = 13362 ms; Unsigncrypt = 20294 ms	Signcrypt = 510 ms; Unsigncrypt = 580 ms	ciphertext size = $\|m\| + (n+2)l_{G_1}$ (hundreds kB)
ABS/HM scheme [101] (1 attribute)	Sig = 2509 ms; Ver = 2515 ms	Sig = 45 ms; Ver = 44 ms	Sig = 16 ms; Ver = 17 ms	$3l_n + 4l_z + 1l_c$ (e.g., 4265 b)
ABE/GPSW scheme [93] (AT=10)	-	Enc = 1572 ms; Dec = 38590 ms	Enc = 60 ms; Dec = 500 ms	ciphertext size = $(AT + 1)l_{G_1} + l_{G_T}$ (e.g., 2907 b)

Figure 3.12 depicts the performance overhead of selected privacy-preserving techniques on the chip card, the mobile device and PC.

The practical results of the performance overhead show that many of advanced schemes are not suitable for IoT real-time applications with constrained IoT nodes. Especially, the execution times of schemes with many expensive operations such as pairing operations (e.g., 3.5 s on the mobile device used), point multiplication or exponentiation (e.g., 131 ms on the mobile device used) take hundreds milliseconds or

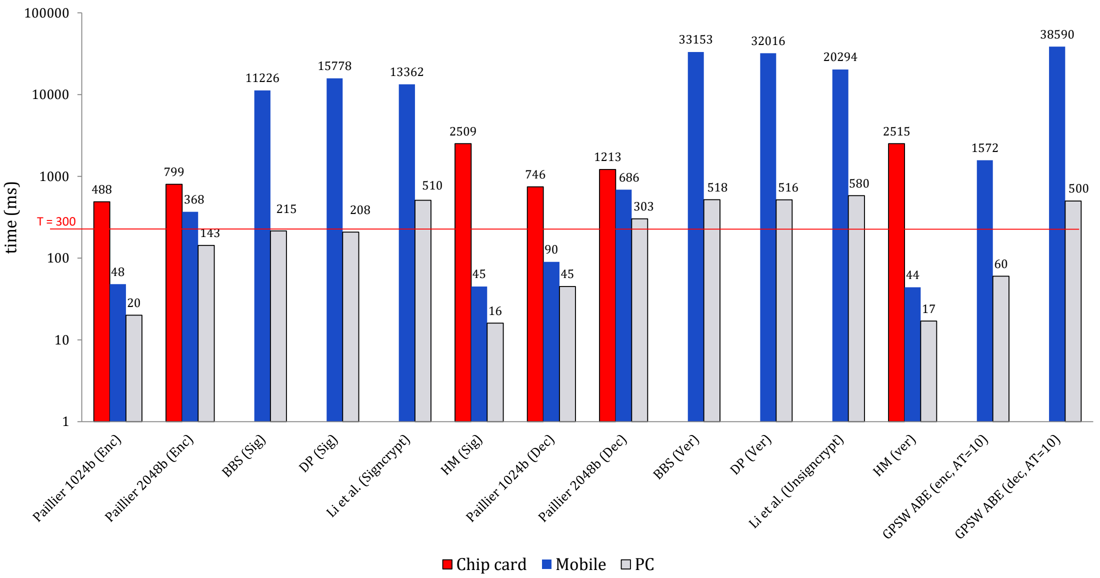


Fig. 3.12: The execution times of privacy-preserving techniques on the devices.

seconds on IoT devices such as smartphones or single-board computer units. Therefore, the privacy protection solutions need more computationally powerful nodes that can perform the expensive cryptographic operations. Furthermore, several privacy-preserving schemes need special cryptographic libraries in order to compute operations such as pairings and so on. Thus, the code size and memory demands are higher than with basic cryptographic methods (AES, RSA, SHA-1,...).

Current trends in IoT such as employing wearable devices and low-cost sensors cause that the IoT environment will consist of more restricted devices than powerful devices (laptops, smartphones). Hence, the privacy-preserving techniques must be efficient and easy-to-deploy at the side where the restricted devices are used. For example, the signing data by some anonymous digital signature schemes (group or attribute signature schemes) should be as efficient as possible in some data collection services. The signing phase should not contain bilinear pairings and only the minimal number of expensive operations. The next subsection presents the assessment of anonymous digital signatures in more details.

3.3.1 Performance Assessment of Anonymous Digital Signatures on Handheld devices

This subsection focuses on the performance evaluation of Anonymous Digital Signatures (ADS) schemes on handheld devices. The following text and results are published in the international journal indexed in Scopus [11].

Seven ADS schemes that are described in Subsection 2.3.1 are implemented and evaluated. Firstly, our implementation and measurement setup are introduced. Then, the experimental results on selected devices are presented, and finally, the deployment of ADS schemes in several use cases are discussed.

Implementation and Measurement Setup

All ADS schemes are implemented in JAVA (Android platform and PC). The java.math.BigInteger class which provides modular arithmetic operations is used. Then, the implementation uses the Bouncy Castle library for advanced cryptographic APIs and the Java Pairing-Based Cryptography (jPBC) library that is a wrapper of PBC (Pairing-Based Cryptography Library) which enables pairing-based and elliptic curve operations. The Android version uses the Spongy Castle library (wrapped version of the Bouncy Castle library). The experimental applications employ basic software optimization tricks but do not use experimental hardware optimization methods, e.g., using drop-in modules [194]. The goal of our implementation is to compare all seven schemes by using the standard cryptographic libraries on standard devices without additional crypto-coprocessors.

In the comparison, the basic parameters of the implemented anonymous digital signature schemes are set to a 80-bit security level, i.e., $\geq$ 1024-bit modulus or $\geq$ 160-bit curves:

- BBS [43], DP [76], HLCCN [111] - these schemes use the MNT curve parameters (type D) with an embedded degree k=6 and the 175-bit order of curves in the implementation. The lengths of the basic parameters are: l_{G_1} is 175 b, l_p is 170 b, l_{G_2} is 175 b and l_{G_T} is 1050 b.
- ACJT [33] - this scheme uses big integers (the strong RSA problem, the decisional DH problem). The lengths of the basic parameters are: l_p is 512 b, l_n is 1024 b and l_c is 160 b.
- CG [53] - the scheme uses big integers (the strong RSA problem, the decisional DH problem). The lengths of the basic parameters are: l_Q is 382 b, l_P is 1024 b, l_n is 2048 b and l_c is 160 b.
- IMSTY [112] - the scheme uses big integers and elliptic curves (the strong RSA problem, the decisional DH problem on elliptic curves). The lengths of the basic parameters are: l_n is 1024 b, l_l is 1024 b, l_{ecc} is 169 b and l_c is 160 b.
- HM GS [102] - the scheme uses big integers (the DL problem). The lengths of the basic parameters are: l_n is 1024 b, l_q is 160 b, l_r is 360 b and l_c is 160 b.

The lengths of other parameters are set in accordance with the recommended lengths that are described in the original papers of the schemes. Devices used in the experimental measurement are specified in Table 3.6. Mobile I. represents less powerful smartphones (Nexus i9250). Mobile II. represents more powerful smartphones

(Nexus 5). PC represents service providers' servers. It is assumed that these servers can be more powerful in practice.

Tab. 3.6: Technical Specifications of Devices Used.

Device	Processor	RAM size	Storage size	Operating System
mobile I (Nexus i9250)	2x 1200 MHz 32-bit ARM	1024 MB	16 GB	Android 4.2
mobile II (Nexus 5 LG)	4x 2260 MHz 32-bit ARM	2 GB	16 GB	Android 5.1
PC	8x 2.53 GHz 64-bit Intel Xeon	8 GB	500 GB	Windows 7

All implemented schemes work with a short 80-bit message. The lengths of produced signatures depend on the schemes. It is measured the execution times of sign and verification phases that are computed on mobiles and PC (specifications are in Table 3.6). The execution times are computed as mean values from 10 and 100 measurements (10 - mobiles, 100 - PC).

Experimental Results

In this subsection, the experimental results of the implemented schemes are presented. Figure 3.13 depicts the performance of the pairing-based ADS schemes on mobile I. The performance of these schemes on mobile II is shown in Fig. 3.14. The most efficient pairing-based scheme is the DP scheme. Its signing with the pairing precomputation takes around 8 s on mobile II. Its verification of the one signature takes about 24 s on mobile II and the verification with the pairing precomputation takes about 9.8 s. The HLCCN scheme that provides the shortest signature needs more time for both phases than the schemes BBS and DP, i.e., signing with the pairing precomputation takes 11.748 s and verification takes 29.395 s (11.7 s with the pairing precomputation) on mobile II. The signing with the pairing precomputation improves the efficiency of signing and the runtime is reduced by ca 50%. The full precomputation in signing reduces the runtimes for all schemes. The signer needs only several ms (BBS: 3 ms, HLCCN: 3 ms, DP: 4 ms) because he/she computes one hash function and few modular multiplications and additions. The pairing precomputation and pairing collapsing in the verification can improve runtimes by ca 60%.

Figure 3.15 shows the performance of the non-pairing-based ADS schemes on mobile I. The performance of these schemes on mobile II is depicted in Fig. 3.16. The most efficient non-pairing-based scheme is the HM GS scheme. The signing in this scheme needs only 15 ms on mobile II and 45 ms on mobile I. The verification of one signature in this scheme needs only 12 ms on mobile II and 44 ms on mobile I. Nevertheless, the sign (verification) phase of the IMSTY scheme needs 8894 ms (9872 ms) on mobile II due to the inefficiency of point multiplication operations in

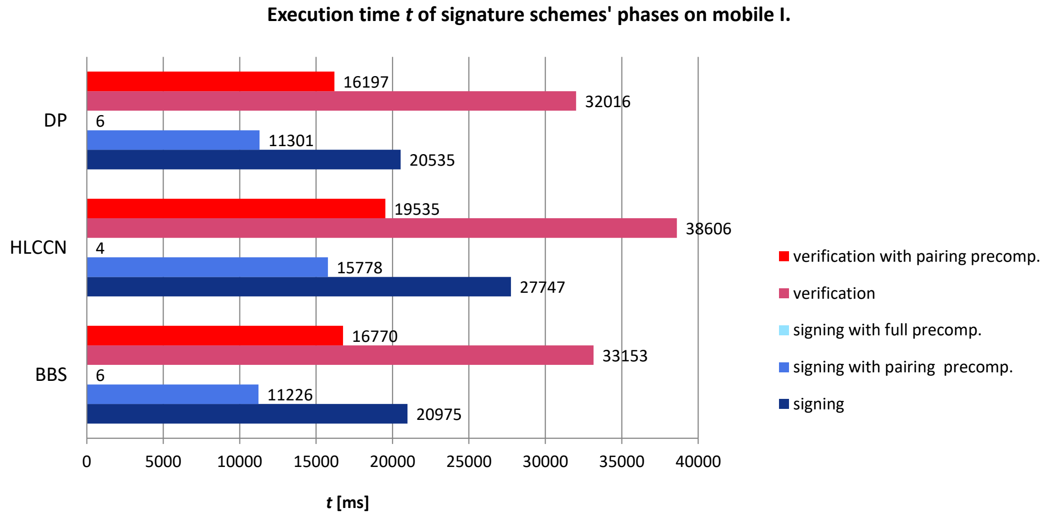


Fig. 3.13: Performance evaluation of pairing-based ADS schemes on mobile I.

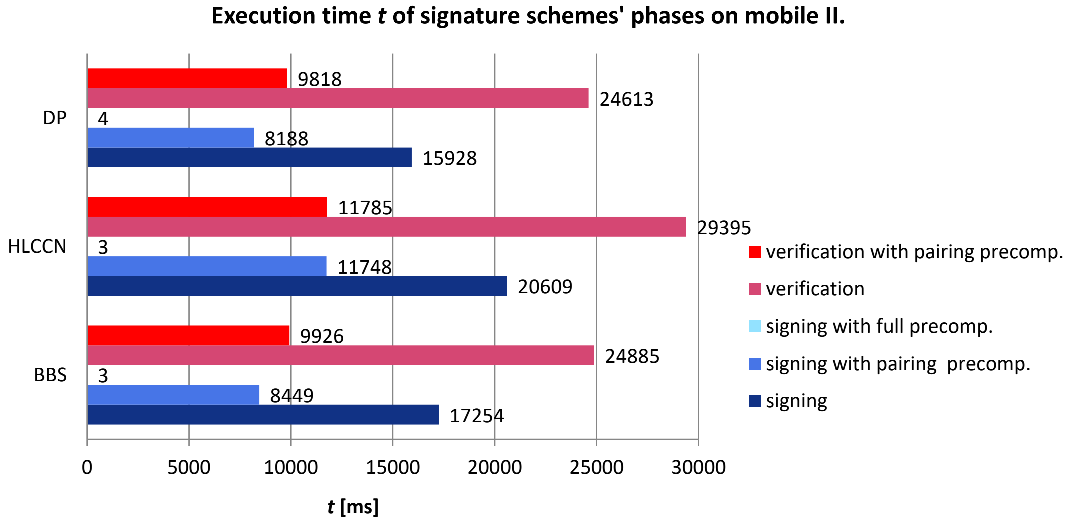


Fig. 3.14: Performance evaluation of pairing-based ADS schemes on mobile II.

the Android platform. All schemes can apply signing with full precomputation and the runtimes of that signing mode take from several hundreds microseconds to few ms.

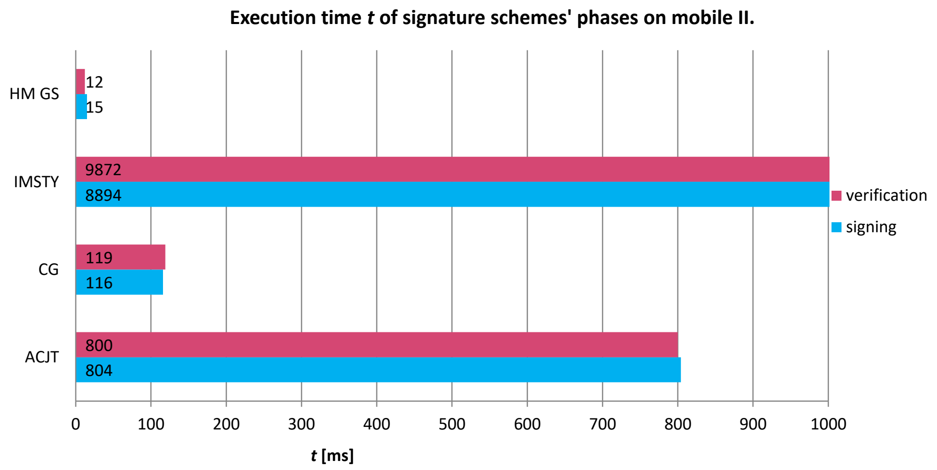


Fig. 3.15: Performance evaluation of non-pairing-based ADS schemes on mobile I.

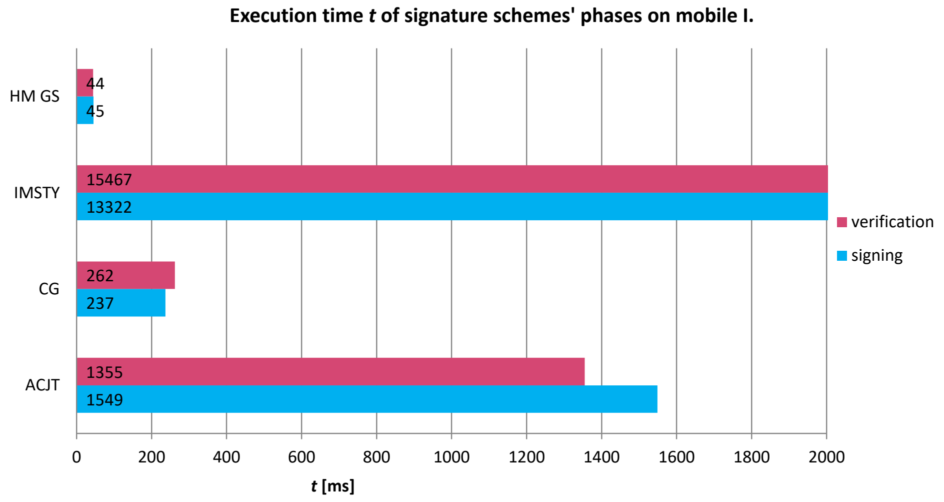


Fig. 3.16: Performance evaluation of non-pairing-based ADS schemes on mobile II.

Further, it is investigated the influence of schemes when the numbers of messages and signatures increase to get insights about the schemes working in systems with several messages/signatures in real time. These results are measured on PC that can represent a server deployed in a privacy-preserving system. Figure 3.17 shows the runtime of the sign phase on PC when the number of messages increases. The pairing precomputation is used in pairing-based schemes. The concrete values in ms can be found in Table 3.7. On the PC device, the schemes BBS, DP, HLCCN, CG, IMSTY and HM GS are able to produce 20 signatures within 2 seconds. The scheme ACJT needs approx. 9 s to sign 20 messages.

Figure 3.18 presents the runtime of the verification phase on PC when the number of signatures increases and Table 3.8 shows the concrete values in ms. The schemes BBS, HLCCN, DP, IMSTY, CG and HM GS are able to verify 100 signatures within 10 s. Nevertheless, the pairing-based scheme BBS, HLCCN, DP must use the pairing precomputation trick. Without this optimization these schemes need from 50 s to 60 s to verify 100 signatures. The non-pairing-based scheme ACJT needs 38.302 s for the verification of 100 signatures.

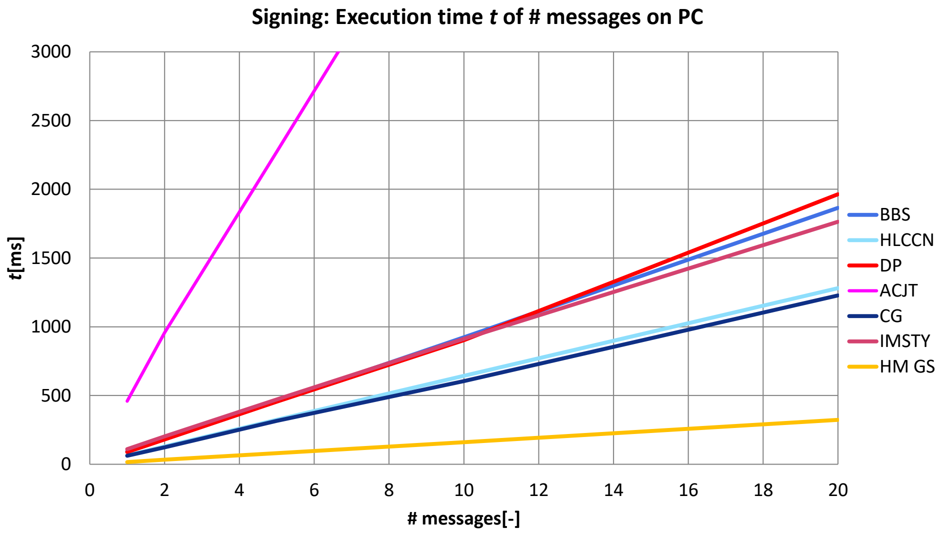


Fig. 3.17: Performance evaluation of ADS: signing on PC.

Tab. 3.7: Time [ms] of signing for various number of messages M on PC.

# M	BBS	HLCCN	DP	ACJT	CG	IMSTY	HM GS
1	93	64	89	459	62	113	16
2	187	129	180	957	123	203	33
5	459	323	455	2274	316	472	81
10	924	644	903	4485	605	913	160
20	1865	1281	1964	8932	1228	1764	323

The practical implementation shows that the used Android platform has difficulties with large structures of the group elements and curves and with their computations. This can be caused by restrictions on the Android platforms (32-bit CPU architectures, less RAM and cache memories, etc.) and by the Android garbage collector. Therefore, the non-pairing-based ADS schemes (tens to hundreds ms) are more efficient than pairing-based schemes (several seconds) on smartphones with the Android platform. The non-pairing-based ADS schemes are also more efficient

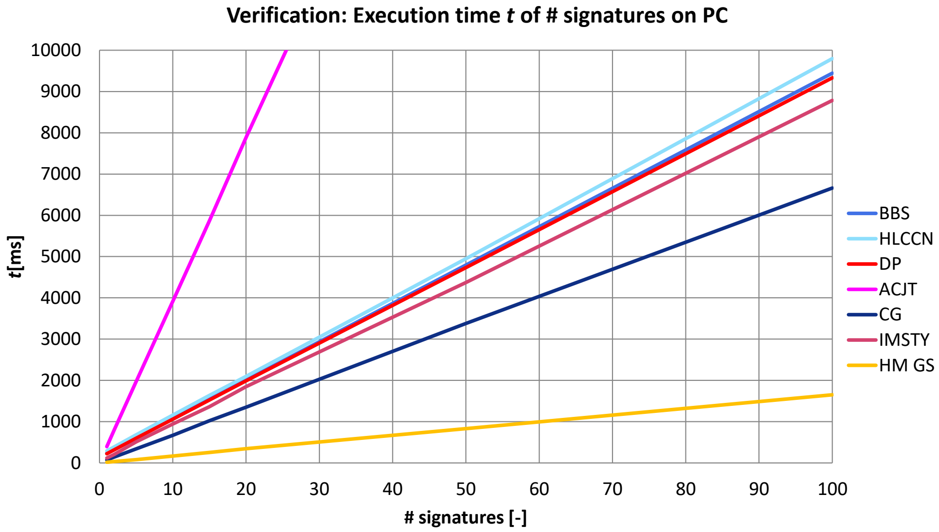


Fig. 3.18: Performance evaluation of ADS: verification on PC.

Tab. 3.8: Time [ms] of verification for various number of signatures σ on PC.

# σ	BBS	HLCCN	DP	ACJT	CG	IMSTY	HM GS
1	237	293	223	396	73	117	17
2	330	389	317	795	141	204	36
5	609	681	599	1974	343	513	82
10	1074	1157	1060	3914	670	944	167
15	1539	1637	1525	5867	1021	1363	255
20	2004	2098	1990	7882	1350	1848	344
50	4794	4948	4731	19440	3379	4373	832
100	9444	9797	9332	38302	6661	8784	1651

than pairing-based schemes on the PC platforms but the performance gap is smaller than on the Android platform. On the other hand, the pairing-based ADS schemes enable us to precompute expensive pairing operations with static parameters that causes the decrease of the runtimes during the signing and verification phases. All schemes enable signing with the full precomputation of parameters that takes only few milliseconds per one signature. Finally, the runtimes of signing and verification can be affected by various random values generated during each signature.

3.3.2 Feasibility of Anonymous Digital Signatures on Constrained Devices

In this subsection, the feasibility of ADS on constrained devices is discussed. The subsection focuses on three scenarios: data collection, access control and data notification systems. It is highlighted the basic security and practical requirements (i.e., performance, storage and communication cost) for every system, and it is recom-

Tab. 3.9: Suitability of Anonymous Digital Signatures in Application Scenarios (☆ - not suitable; ☆ ☆ - conditionally suitable; ☆ ☆ ☆ - suitable).

Scheme	Online Data Collection Systems	Offline Data Collection Systems	Access Control Systems	Data Notification Systems
BBS [43]	☆	☆ ☆	☆ ☆	☆
DP [76]	☆	☆ ☆	☆ ☆	☆
HLCCN [111]	☆	☆ ☆ ☆	☆ ☆	☆
EH [82]	☆	☆ ☆ ☆	☆	☆
ACJT [33]	☆	☆	☆ ☆ ☆	☆
CG [53]	☆ ☆ ☆	☆	☆ ☆ ☆	☆ ☆ ☆
IMSTY [112]	☆	☆	☆ ☆	☆
HM GS [102]	☆ ☆ ☆	☆	☆ ☆ ☆	☆ ☆ ☆

mended the suitable schemes for every system. The analysis also takes into account the cost of revocation process. The recommendation of ADS suitability is based on the practical and theoretical results and is summarized in Table 3.9.

Perspective of ADS in Online and Offline Data Collection Systems

Data collection systems represent the many-to-one communication model. A general data collection system is depicted in Figure 3.19. Users create signatures on their messages that are sent to a server managed by a service provider. If some unsecured services work with sensitive and personal user data, then there is the risk that the data may leak to some third parties without user's permission. Furthermore, the attackers can tamper with messages and users can deny their creation without the deployment of digital signature or digest mechanisms. However, ADS should provide security to the provider and privacy to users in this scenario. Thus, users' messages are signed, protected against tampering and an unauthorized party is not able to link the messages from the same user.

These systems do not usually require some advanced revocation mechanisms or revocation lists. In some services, the online systems collect messages in short time periods (e.g., real-time monitoring services, healthcare monitoring, ...). Therefore, the node has to create the signature in a short time (e.g., $\leq$ 150 ms). For this scenario, the suitable schemes are the HM GS and CG schemes that satisfy the online monitoring by fast sign and verification phases on recent smartphones.

Further, there are systems (e.g., smart grid consumption data collection) which collect data in longer periods, i.e., hours, days. In these offline (non-real-time) systems with a huge number of nodes (signers), the signed messages are sent to a central server which has to verify many messages. The servers are usually more powerful than signer devices but the length of the signature should be as short as possible in order to mitigate traffic congestion in communication, and save storage space. Therefore, the suitable scheme should produce short signatures and be efficient. The pairing-based schemes, which provide shorter signatures ($\approx 1300 - 2000$ bits) than non-pairing-based schemes ($\approx 6000 - 8000$ bits), can be more suitable in this scenario. Nevertheless, the optimization techniques and tricks such as pairing collapsing and precomputation must be employed to make signing and verification efficient. Moreover, some pairing-based schemes usually enable to use the batch verification that reduces the number of pairing operations to a constant number. Existed advanced software and hardware optimizations, which significantly reduce the time of pairing and point multiplications, make pairing based schemes (e.g., the HLCCN scheme [111] and the EH scheme [82]) useful for offline data collection systems that requires short signatures to reduce communication and storage cost.

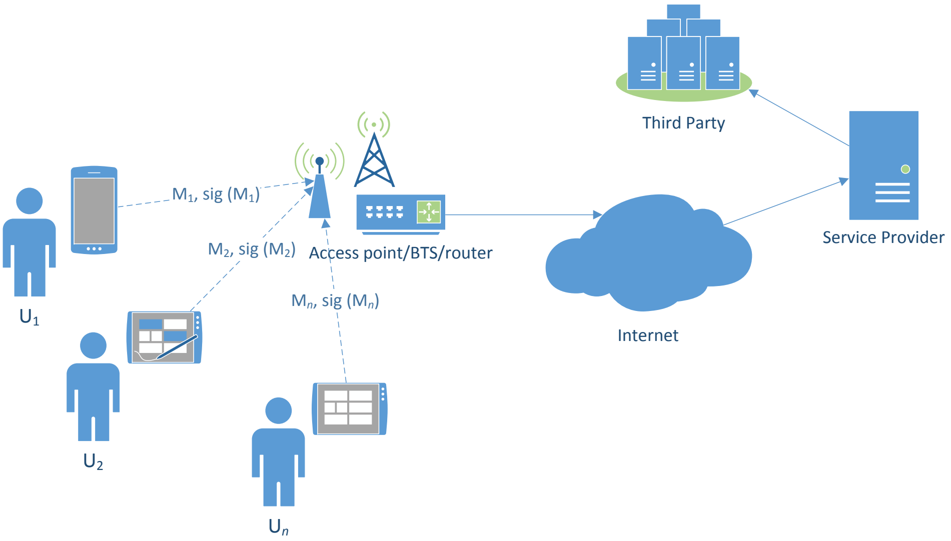


Fig. 3.19: A privacy-preserving data collection system.

Perspective of ADS in Access Control Systems

Access control systems can protect physical or digital assets such as room/building entrances, online services, subscription memberships, data storages and clouds. A general scheme is depicted in Figure 3.20. Users can use their smartphones, smartwatches and other handheld devices instead of chip cards and RFID tags. If smartphones are able to communicate with a reader or a verifier then a user can authenticate themselves and lock/unlock protected services. In the authentication phase, users sign challenge messages obtained from the verifier by the anonymous digital signature scheme. Smartphones are more powerful than chip cards and enable to employ more advanced cryptographic protections. Therefore, it can be expected that smartphones will be used more and more in these services in future.

On one hand, access control systems do not have high communication and storage cost. On the other hand, the clients usually require a fast authentication process. The signature creation on the client side with a smartphone and the verification on a server together with message communication must take a practical time (up to 2 s). Moreover, ADS must offer advanced security properties such as an immediate user revocation (i.e., a revocation list), the non-collusions of malicious users and so on. Hence, suitable schemes from the evaluation are the ACJT scheme, the CG scheme and the HM GS scheme that also allow to check revoked users on the verifier side. Nevertheless, the process of checking the revoked users requires some additional operations on the verifier side. For example, the HM GS scheme requires one exponentiation operation in modulo n per one revoked user in the revocation list. If the revocation list increases by new revoked users then the time of authentication process increases too. In order to prevent the slow authentication process, the credentials/private keys should be updated which enable to erase the revocation list.

The pairing-based schemes and the IMSTY scheme can be suitable only if the full precomputation signing mode is applied and the verifier uses a strong device such as PC. Moreover, pairing-based schemes must be enhanced by a revocation phase that also requires some additional operations on the verifier side.

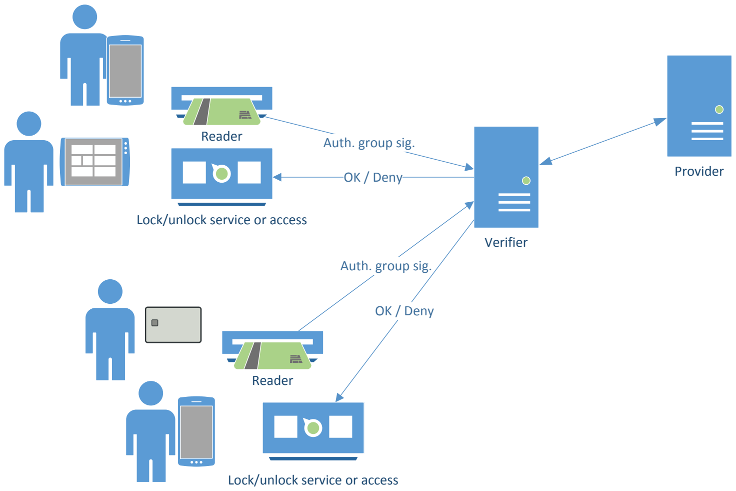


Fig. 3.20: A privacy-preserving access control system.

Perspective of ADS in Data Notification Systems

The data notification systems represent the many-to-many communication model. Figure 3.21) depicts the general scheme where users' messages and notification data are broadcasted from users to other users. In this communication pattern, ADS are usually used to secure data that are broadcasted or uploaded. This scenario offers anonymous data uploading, notification and sharing. Users with their smartphones and handheld devices directly broadcast data via an access point or a router or send data to servers that forward data to other users. Other users then check data signatures and accept messages if the signatures are valid.

Some notification systems require a very short computation overhead (e.g., $\leq$ 300 ms) for fast reactions in some safety applications, e.g., break alerts in Vehicular Ad hoc Networks (VANETs). The verification and signature creation must be as fast as possible but the revocation property is not so important in these services.

In this scenario, the suitable schemes are the HM GS scheme and the CG scheme. These schemes are very efficient in signing and verification. For example, the verifier (PC) with the HM GS scheme can verify about 20 messages in real time if the runtime of cryptographic overhead must be $\leq$ 300 ms. The schemes also provide revocation mechanisms (i.e., revocation lists, credential updates) that are required in some privacy-preserving data notification systems.

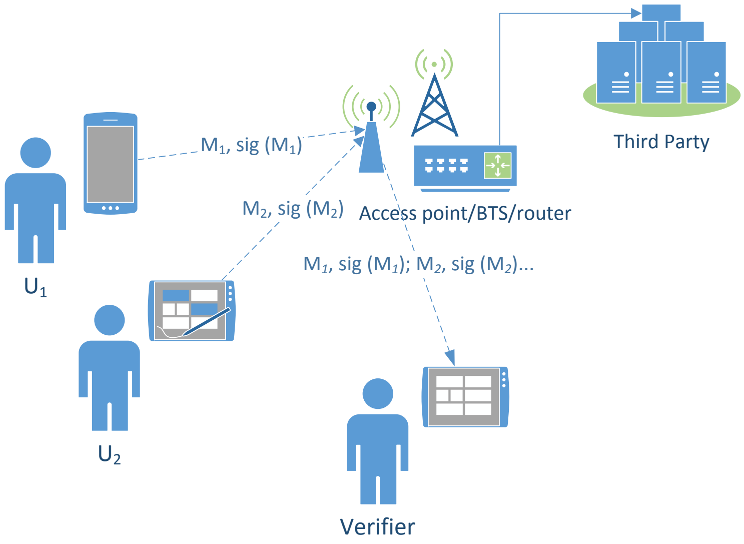


Fig. 3.21: A privacy-preserving data notification system.

3.4 Post-Quantum Cryptography on Constrained Devices

This section investigates the performance and memory assessment of post-quantum public key schemes on constrained devices. The section contains the practical runtimes of chosen PQC schemes measured on single-boards and smartphones, and a discussion about PQC feasibility on constrained devices such as microcontrollers and smart cards.

Implementation of PQC cryptosystems

The following text introduces the setup and software implementation details and issues of PQC cryptosystems on 32-bit CPU single-board devices and mobile devices. PQC schemes can be implemented from scratch by using the descriptions in research papers in various programming language such as C/C++, JAVA, Python and others. Nevertheless, developers have to implement and prepare many math, cryptographic and arithmetic modules, e.g., Gaussian sampler and matrix/polynomial multiplication and so on. This from-scratch implementations of the schemes may cause many security flaws and performance issues. However, several open source projects and libraries that implement PQC schemes and math functions already exist, e.g., Codecrypt - the post-quantum cryptography tool, Java Lattice Based Cryptography Library (jLBC), libPQP - Python post-quantum library, the LatticeCrypto Library [136] and liboqs library (the Open Quantum Safe project [184]). Therefore, employing the libraries with PQC primitives could be more efficient and stable. In this investigation, the liboqs library (the Open Quantum Safe project [184]) is used. The following text describes the Android platform and its suitability for PQC libraries and schemes, and the setup steps on a single board 32-bit ARM device with Linux OS.

PQC on Android Systems

Many post-quantum libraries such as liboqs are implemented in C/C++. In order to wrap PQC C-written libraries and schemes in the Android platform, the developers can use Android Native Development Kit (NDK) and write code in C/C+ and Java Native Interface (JNI) using for calling native functions from the C libraries. Then, PQC implementations written in C could be modified and be accessible by JAVA via JNI. There are also few PQC implementations already written in JAVA but these implementations are usually inefficient, see the experimental results in the following subsection.

As an alternative option, the developers and security engineers may use the ARM big.LITTLE heterogeneous computing architecture to embedded C/C++ software development on ARM processors. The PQC implementations written directly in assembly can be faster than C-written implementations, see results [113].

PQC on 32-bit Single Boards

The liboqs library can be easily installed and built on 64-bit platforms with various Linux OS. Nevertheless, a single-board device that has only 32-bit ARM CPU is used. Hence, some steps have to be made prior to run, benchmark and modify liboqs with PQC schemes on the 32-bit platform. Firstly, header files and methods

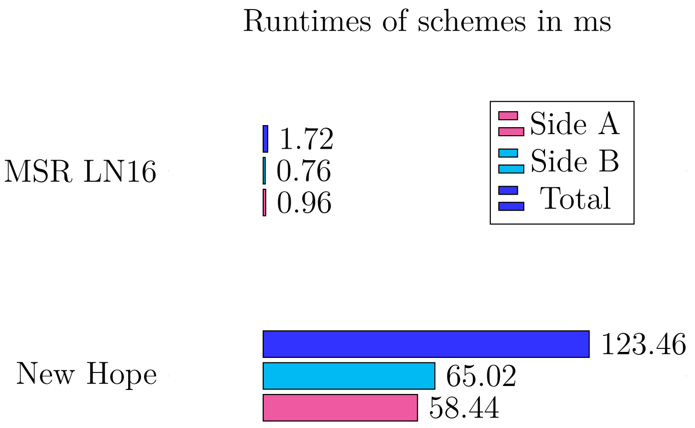


Fig. 3.22: The performance of PQC key exchange schemes on mobile devices.

have to define ARM as the platform. Further, it is necessary to retype all 64-bit types that 32-bit CPU cannot handle. The implementation must be adapted to the 32-bit architecture.

The next subsection outlines the experimental results from both platforms described above.

3.4.1 Performance Assessment of PQC key exchange schemes

This subsection presents the experimental results of post-quantum key exchange cryptographic schemes that are implemented on small ARM devices.

PQC key exchange schemes on Android mobile devices

This experimental measurement uses a mobile device with 32-bit CPU Qualcomm Snapdragon 801, 4 cores, 2.5 GHz, 2 GB RAM with Android 6. Two PQC key establishment schemes are tested with the configuration defined as follows:

- New Hope - lattice-based scheme with 206-bit post quantum security level. The implementation is written in JAVA. The details of the scheme can be found in [32].
- MSR LN16 - lattice-based scheme with 128-bit post quantum security level. The implementation is written in C and uses JNI. The details of the scheme can be found in [136].

Figure 3.22 depicts the performance results of two PQC key exchange schemes measured on the mobile ARM devices. The depicted results are the runtimes of the schemes on side A, side B and the total runtimes on both sides. The values are averaged from 30 measurements. The results indicate that the C/JNI application of the lattice-based key exchange scheme (MSR LN16) is more efficient than the JAVA application of the lattice-based key exchange scheme scheme (New Hope). However, New Hope provides higher security level (206-bit) than MSR LN16 with 128-bit.

PQC key exchange schemes on single-board devices

The second experimental measurement uses a single-board ARM device with 32-bit CPU ARMv7l 1.2 GHz, 1 GB RAM with Linux OS (Raspbian Stretch Lite). This measurement employs six PQC key establishment schemes with the configuration defined as follows:

- New Hope - lattice-based scheme with 206-bit post quantum security level. The full specification can be found in [32].
- NTRU - lattice-based scheme with 128-bit post quantum security level. The full specification can be found in [107].
- BCNS - lattice-based scheme with 78-bit post quantum security level. The full specification can be found in [48].
- Frodo - lattice-based scheme with 130-bit post quantum security level. The full specification can be found in [46].
- McBits - code-based scheme with 120-bit post quantum security level. The full specification can be found in [40].
- SIDH - isogeny-based Supersingular Isogeny Diffie-Hellman (SIDH) scheme with 128-bit post quantum security level. The full specification can be found in [71].

Figure 3.23 depicts the performance results of six chosen PQC key exchange schemes measured on the single-board ARM devices with a single core. The runtimes on side A, side B and in total are averaged from 10 measurements. The lattice-based New Hope scheme is the most efficient scheme from 6 measured schemes. New Hope, NTRU, BCNS schemes take less than 35 ms. On the other hand, the SIDH scheme takes about 4414 ms.

Figure 3.24 shows the message lengths of chosen PQC schemes. The scheme with the least number of exchanged bytes during the key exchange protocol is SIDH with 1152 B. To be noted, that this scheme is less efficient than 6 measured schemes. The New Hope and NTRU schemes exchange less than 4 kB during the protocol and due to their efficiency these schemes offer tradeoff between efficiency and message size parameters.

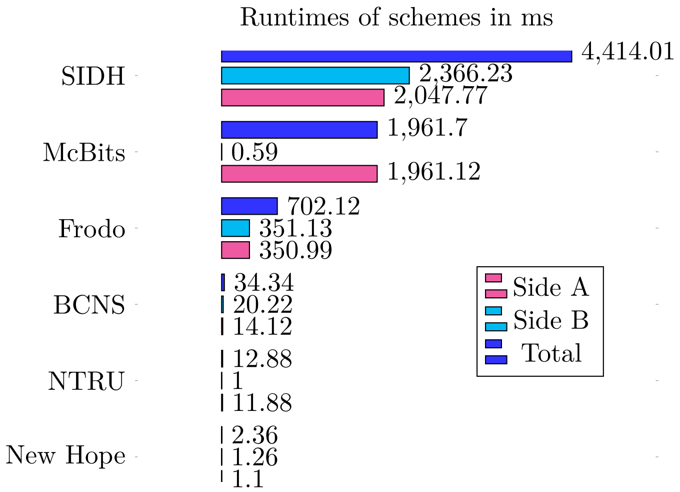


Fig. 3.23: The performance of PQC key exchange schemes on single-board devices.

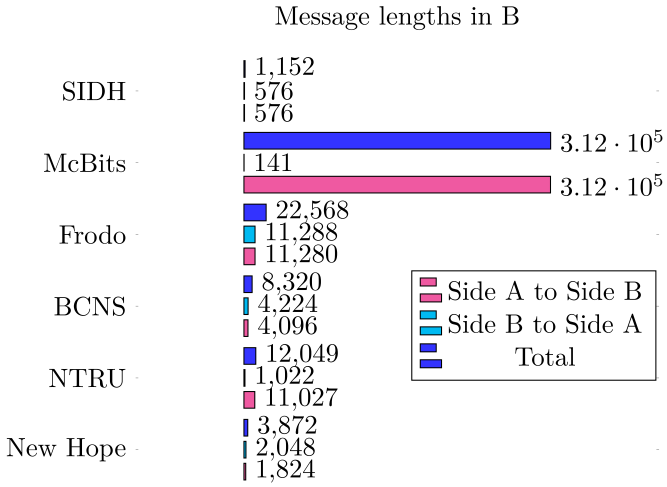


Fig. 3.24: The total message lengths of the PQC key exchange schemes.

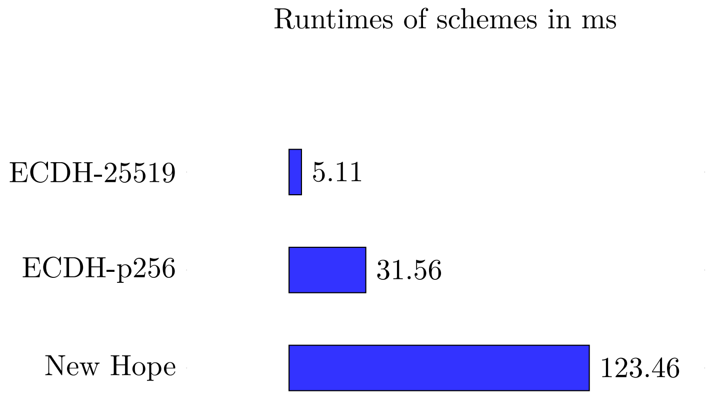


Fig. 3.25: Performance of PQC New Hope and ECDH on the mobile devices.

Comparison of New Hope and ECDH schemes

This part compares classic ECDH key establishment scheme with a PQC key exchange scheme, namely, New Hope.

Figure 3.25 compares New Hope and ECDH key exchange schemes on the mobile device. The ECDH-25519 takes only 5.11 ms, ECDH-p256b takes 31.56 ms and the post-quantum New Hope scheme takes 123.46 ms.

Figure 3.26 compares New Hope and ECDH key exchange schemes on the single-board device. New Hope takes 2.36 ms and is slightly less efficient than ECDH-p256 that takes 2.11 ms. Both schemes are written in C. Nevertheless, the ECDH scheme written in JAVA takes more time than in C (68.69 ms for ECDH-p256, 29.15 ms for ECDH-25519) on the single-board devices. The results prove that post-quantum cryptographic schemes such as lattice-based key exchange schemes could be competitive to classic asymmetric cryptosystems for key establishment.

3.4.2 Feasibility of PQC Schemes on Constrained Devices

This subsection discusses the feasibility of PQC schemes that can suite in IoT systems employing computational constrained nodes, memory constrained nodes and message sized restricted communication protocols. Table 3.10 summarizes the categories of PQC cryptosystems and their suitability for IoT systems with various restrictions. The following text presents some perspective PQC schemes for IoT based on the experimental results and general knowledge.

- **PQC for systems with performance restrictions** - these systems employ nodes with several MHz. The cryptographic and PQC schemes and their op-

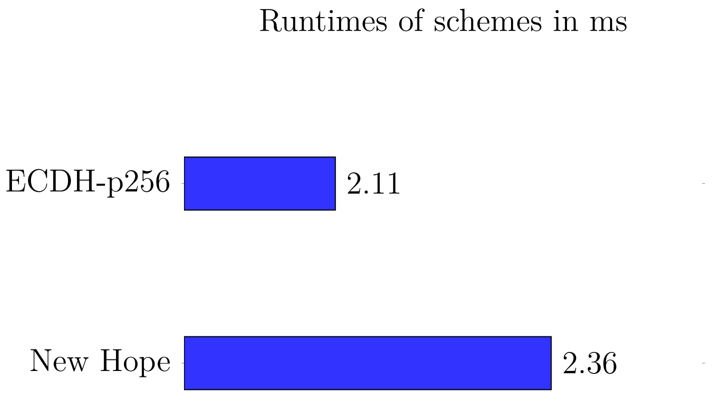


Fig. 3.26: Performance of PQC New Hope and ECDH on the single board devices.

erations/phases should have minimal cycles. The key exchange PQC schemes that take up to several tens of ms in total can be suitable. As the most efficient PQC schemes are considered lattice-based schemes. Also the experimental results indicate that schemes such as New Hope can be quite fast and could be as efficient as a not-quantum resistant ECDH key exchange scheme.

- **PQC for systems with memory restrictions** - these systems employ memory constrained nodes, e.g., microcontrollers with several kB, smart cards as secure elements etc. All parameters and keys used in a PQC scheme should have moderate sizes and take a reasonable-portion of memory (RAM, EEPROM) in nodes. Isogeny-based schemes such as isogeny-based Supersingular Isogeny Diffie-Hellman (SIDH) that employ supersingular elliptic curves offer short lengths of parameters and keys. Employing compression algorithms and techniques [70] offers SIDH with 330-bytes public keys at a 128-bit quantum security level. On other hand, Multivariate schemes such as Rainbow use secret and public keys having several tens of kB. The long keys are also used in code-based cryptography (e.g., the McEliece scheme).
- **PQC for systems with restricted communication protocols** - several IoT and wireless communication protocols and technologies have restricted lengths of messages such as LORAWAN, SigFox (more about limitations see [3]), Application Protocol Data Unit (APDU) used by smart cards and other protocols. This restriction requires that exchanged messages must keep minimal sizes during the key establishment. In the measurement, SIDH with 1152 B seems as the most promising scheme. The low performance of SIDH could be solved by utilizing a high performance cryptographic co-processor such as

Tab. 3.10: The feasibility of PQC schemes in IoT (☆ - not suitable; ☆ ☆ - conditionally suitable; ☆ ☆ ☆ - suitable).

PQC categories	Systems with performance restrictions	Systems with memory restrictions	Systems with restricted communication
Hash-based	☆ ☆	☆ ☆	☆
Code-based	☆ ☆	☆	☆
Lattice-based	☆ ☆ ☆	☆ ☆	☆ ☆
Multivariate-based	☆ ☆	☆	☆ ☆
Isogeny-based	☆	☆ ☆ ☆	☆ ☆ ☆

in [176] or by accelerating this algorithm on FPGA cards. Code-based schemes that send hundreds kB long messages are not suitable in this scenario.

3.5 Summary of Chapter

This chapter presents the performance and memory assessment of modern and state of the art cryptographic schemes on various types of devices. Nowadays, symmetric ciphers and hash functions can be easily implemented into the IoT services that use constrained devices. These functions take only few milliseconds and can be run on memory restricted microcontrollers with RAM less than 1kB. Asymmetric cryptographic schemes and modular arithmetic operations can be used in the IoT services and applications as well. However, the devices should provide at least middle-sized RAM (e.g., > 4kB) and storage memories (e.g., > 10kB). Further, the time execution of some asymmetric cryptography functions and operations, e.g., RSA signing by a 2048-bit private key, can cause a latency more than hundreds milliseconds on computationally constrained devices such as MSP microcontrollers, smart cards, etc. For example, applications that must sign and send data in real-time cannot employ such computational expensive operations on computationally constrained devices.

Further, privacy-preserving techniques and their perspective in IoT are analyzed. Many strong privacy-preserving solutions are based on proof of knowledge schemes, bilinear pairing operations and employ public key cryptography schemes. Constrained devices with a small RAM memory may have difficulties to employ these privacy-preserving solutions. Cryptographic operations such as bilinear pairings, modular exponentiation and point multiplication are computationally expensive and have usually high memory and RAM demands due to the large input/output structures and the need of the external cryptographic libraries. The most computationally expensive cryptographic operation on ARM devices is the pairing operation.

For example, one pairing operation with 175-bit curves (performed by the JAVA implementation) takes about 2.3 s on a 2.26 GHz 32-bit ARM device with Android OS.

The privacy-preserving techniques such as homomorphic encryption, group signature schemes, attribute based signature schemes, attribute based encryption or signcryption schemes need from tens milliseconds to several seconds for their phases, i.e., sign, encrypt, verify, decrypt, on devices such as chip cards and ARM devices. The secure and privacy-preserving IoT applications need a solution that is not based on expensive bilinear pairing operations, produces short signatures and is easy to deploy in memory-restricted devices. The non-pairing attribute based signature schemes seem to be perspective privacy-preserving techniques in some IoT applications. These schemes can be implemented on chip cards, SAM modules and other constrained devices.

Finally, the last section in this chapter provides the assessment of post-quantum cryptography on constrained and small devices. The results show that some PQC schemes can be implemented into applications on hand-held and single-board devices. Nevertheless, many PQC schemes are not suitable for the deployment on constrained devices such as smart cards and microcontrollers due to the large sizes of parameters.

4 Novel Systems Based on Advanced Public Key Cryptographic Protocols for Constrained Devices

This chapter presents three author's proposals of security systems based on advanced public key cryptographic protocols. Each security proposal is designed for a different application, i.e., secure authentication and access control system, secure and privacy-preserving data transfer, and anonymous transactions. The main goal of these proposed systems and methods is suitability for deployment on constrained devices. Proposals are presented in the following sections as follows:

- Section 4.1 presents a secure and efficient two-factor zero-knowledge authentication system based on smart cards.
- Section 4.2 presents a secure and privacy-preserving data transfer system based on light-weight group signatures with a time-bound membership.
- Section 4.3 presents a proposal of secure decentralized privacy-preserving transactions based on lightweight ring signatures.

4.1 Secure and Efficient Two-factor Zero-knowledge Authentication System Based on Smart Cards

This section presents a secure and efficient two-factor authentication protocol for fast access control systems and user-things identification schemes based on programmable smart cards. The proposed solution is based on a zero-knowledge approach, and it is protected against common attacks. Further, the proposed authentication protocol is implemented on current off-the-shelf programmable smart cards in order to demonstrate its efficiency and practicality. Finally, the solution is compared with related works and the improvement of the proposed solution in computation and communication perspectives is shown. The amended version of the text below is a part of the author's paper in the journal with an impact factor, namely, Computers & Security, [12].

4.1.1 Introduction, State of the Art and Contribution

Authentication protocols based on smart cards are very popular and are used in many access control systems where users are proving the possession of smart card items. Using contactless smart cards or active radio-frequency identification (RFID) tags can be very practical in various scenarios where using contact smart cards can be obstacle. The suitable scenarios with contactless smart cards are access control systems and identification systems in road tolls, gates, doors and entrances of facilities (workplaces, libraries, hospitals, universities, parking tolls), where comfort and simple usage are required properties. Access control systems usually check users' credentials (secret keys, private keys, passwords, authentication codes, user IDs) and then they decide about users' access to services and assets. These credentials stored on contactless cards can be transcribed via a radio interface by the standards ISO/IEC 14443 (proximity cards with read distance up to 10 cm) and ISO/IEC 15693 (vicinity cards with read distance up to 1.5 m). Nevertheless, contactless interfaces allow attackers to eavesdrop and copy these credentials. Hence, the authentication process should provide user comfort, simple usage, fast verification and the security properties such as non-forgeability, soundness, completeness, low false acceptance rate and privacy protection.

Many access control systems based on chip cards still use older chip cards, e.g., Mifare Classic cards that can be easily copied, forged, emulated or can be broken by attacks such as replay and other attacks [89]. For example, the paper [106] shows that some features of the Mifare DESFire card scheme can be insecure. The paper [106] also shows possible attacks on the integrity of encrypted data on the Mifare DESFire platform. Nevertheless, some current programmable smart cards based on platforms such as JAVA Card OS, .NET Card OS, MultOS, Basic Card OS offer several basic cryptographic methods (e.g., DES, AES, SHA-1, SHA-2, ECDH, 1024/2048 bit RSA) and enable us to employ more secure authentication protocols. Furthermore, Basic cards and MultOS cards also offer modular arithmetic operations and elliptic curve operations. These platforms can host advanced authentication protocols based on modern cryptography such as zero-knowledge protocols, sigma protocols and attribute authentication schemes. However, programmable cards are constrained devices with limited computational power (1 core, tens MHz) and restricted memory (tens of kB), thus such devices are not appropriate for computationally and memory expensive authentication schemes (e.g., schemes using expensive bilinear pairing operations).

In this section, a novel smart-card-based authentication system is proposed. The system deploys zero-knowledge authentication (i.e., a modified Schnorr signature scheme [171]) and elliptic curves. The goal is to propose a solution that offers secure and efficient user authentication in access control systems and user/things identification schemes deployed in protected environments such as critical infrastructure. Moreover, the proposed authentication solution is implemented on off-the-shelf programmable smart cards. Then, its performance is measured and communication delay with contact and contactless smart cards is analyzed. In order to prove the efficiency of the solution, the comparison with related works is provided. Furthermore, the security analysis shows that the solution protects against various common attacks.

State of the Art

There are many authentication protocols and solutions based on smart cards that can be used in access control systems, identification systems, ePassports [182], remote control applications and secure access modules in smart grid.

User authentication schemes based on smart cards and shared passwords are studied in many papers, e.g., [64], [126], [199], [131], [130] and [209]. There are many simple and efficient authentication schemes based on user passwords, hash functions and simple modular arithmetic operations that secure the authentication process of users with smartcards. Madhusudhan and Mittal in [138] review the security features of dynamic ID based password authentication schemes that are based only on hash functions and XOR operations. Their analysis shows that it is hard to achieve all security goals. The similar analysis is provided by Wang and Ma in [199]. They mention several defects of the authentication schemes that do not use public-key techniques. Furthermore, Wang and Wang [200], [201] map the history of smartcard-based password authentication schemes from the Chang's scheme proposed in 1991 [58] to recent schemes, e.g., the Jiang *et al.*'s scheme proposed in 2014 [211]. These schemes can be efficient due to using the symmetric cryptography, hash functions and simple math operations, but these schemes are usually developed in a break-fix-break-fix cycle [201]. These schemes are usually successfully attacked after few years from their publishing. For example, Song proposes an advanced smart card based password authentication protocol in [183], and later, the paper [62] presents several vulnerabilities (an internal offline guessing attack) of this scheme. Therefore, these password-based authentication schemes with non-tamper-resistant smart cards are not suitable for secure access control systems employed in critical infrastructures requiring a strong security level [31].

Recently, Wang and Xu [198] have analyzed 3 password-based remote user authentication schemes with non-tamper-resistant smart cards. They show that many schemes do not achieve some critical security goals and suffer from offline dictionary attacks and impersonation attacks. They propose a reference model for deploying the public key algorithms correctly.

There are also several authentication schemes with smart cards that use public cryptographic primitives and these schemes serve to authenticate users/nodes in various scenarios such as secure communication via Session Initiation Protocol (SIP), secure communication in Wireless Sensor Networks (WSN), remote server access and access control systems. For example, Mishra *et al.* [152] propose a secure and efficient mutual authentication scheme with key agreement for SIP based on elliptic curve cryptography (ECC). Their login and authentication phase consists of 11 hash functions and 3 point multiplication operations. Their proposed scheme is efficient in the terms of computational and communication overheads comparing to other authenticated key agreement (AKA) schemes for SIP. Yeh *et al.* [215] present their ECC-based authentication mechanism that should enhance security during the authentication in the WSN environment. The verification and mutual authentication phases of their protocol require 11 hash functions, 4 point additions, 6 point multiplications, and 2 exponentiation operations in total. Nevertheless, the smart card side computes 2 point multiplications, 1 random number generation, 1 point addition and 4 hash functions. Another similar protocol proposed by Choi *et al.* [65] needs only 3 point multiplications and 7 hash functions on a smart card side. Recently, Wu *et al.* [207] have noticed several weaknesses in Yeh *et al.*'s and Choi *et al.*'s schemes. Their proposed authentication scheme needs 2 scalar multiplications on F_p, 1 symmetric encryption and 11 hash functions on a smart card side. However, Xie *et al.* [210] demonstrate that Wu scheme cannot resist an impersonation attack.

Further, there are several strongly secure authentication schemes with smart cards that provide advanced security features, e.g., user privacy, attribute authentication, mutual authentication and authenticated key agreement, forward secrecy. These authentication schemes such as [4], [103], [54] are usually computationally expensive due to using several asymmetric operations, e.g., the exponentiation of big integers, multiplication, bilinear pairings. Their authentication processes usually take often more than 500 milliseconds on current programmable smart cards. For example, the 1024-bit scheme [103] needs about 2.9 s on MultOS ML3 smartcards. The authenticated key establishment protocols such as Password Authenticated Connection Establishment (PACE) [28, 105], the three-party authenticated key agreement (3PAKA) protocol [212], the OPACITY protocol [74] and the TP-AMP protocol [124] can be used for the authentication of users with smart cards but also these protocols are more complex and usually contain more expensive cryptographic and math operations. For example, Ullmann *et al.* [193] present the real life implementation of two cryptographic password-based protocols (PACE and TP-AMP) on smart cards. Their implementation of the PACE protocol takes about 945 ms and their implementation of the TP-AMP protocol takes about 978 ms. The authors use a contactless java card with the NXP's high security SmartMX chip. Nevertheless, the goal of our work is to design the authentication solution with smart cards that is secure and more efficient (up to 500 ms) on current programmable smart cards.

The recent work [189] presents a phone-based authentication solution that utilizes a zero knowledge protocol. The authentication process is based on identity-

Tab. 4.1: Summary of Authentication Approaches/examples based on Smart Cards.

Approach / example scheme	Efficiency	Security level / advanced properties
Password-based authentication - e.g., Song scheme[183]	High efficiency (<100 ms)	Low security (Internal offline guessing attack [62])
ECC-based authentication - e.g., Wu scheme *et al.* [207]	Moderate efficiency (100 - 500 ms)	Medium security (impersonation attack [210])
Attribute-based authentication - e.g., HM scheme [103]	Low efficiency (> 500 ms)	Strong security / Privacy-preserving properties
Password Authenticated Connection Establishment (PACE) [28]	Low efficiency (> 500 ms)	Strong security / Key establishment
Identity-based authentication - e.g., Teh *et al.* scheme [189]	Low efficiency (pairing operations on a verifier side)	Strong security
Zero knowledge authentication - this scheme	Moderate efficiency (100 - 500 ms)	Strong security

based identification schemes that is proposed by Kurosawa and Heng [123]. A prover (i.e., a mobile device) computes 1 modular addition and 2 modular exponentiation operations. A verifier computes 2 bilinear pairings, 1 exponentiation and 1 multiplication during the authentication phase. Due to expensive pairing operations, the implemented authentication phase in the 1024-bit version takes about 2744 ms on the Lenovo K900 mobile and a server. Employing mobile devices as provers' devices in authentication solutions can increase the efficiency but storing user private keys in mobiles can be a security risk due to various attacks. Usually, mobile-based solutions should employ some secure elements, e.g., microSD cards, SIM cards (i.e., smart cards). Table 4.1 summarizes discussed authentication approaches and their security and efficiency properties. The following text introduces an efficient and secure user authentication solution based on the off-the-shelf smart cards, i.e., Basic cards and MultOS cards. The proposed authentication protocol protects against common attacks and it provides some security features such as linkability and traceability in order to monitor user IDs in an access control system. This property allows to link user movements in highly secure areas.

Contribution

The proposed authentication protocol is based on strong cryptographic primitives that can be implemented on current programmable off-the-shelf smart cards. The system uses a zero-knowledge protocol (the Schnorr's scheme) with elliptic curves. The user must prove the knowledge of the private key that is stored only in his/her smart card. The private key is mapped to the public user ID and never leaves a secure storage (a smart card) in the solution. Only user's smart card knows his/her private key for authentication. Hence, cloning the user credentials is hard under assumptions that smart cards provide basic countermeasure methods that prevent basic side channel attacks and that the leakage of the private key is negligible. Nevertheless, if an attacker is able to break into the smart card storage and get user's private key, then a system manager is able to link all public IDs and revoke compromised IDs (e.g., IDs from stolen cards) in the access control system.

Furthermore, it is presented a proof-of-concept implementation of the proposed authentication solution on two smart card platforms, i.e., Basic Cards and MultOS Cards. Four implementations are produced, namely, two applications for the smart card side (Basic/MultOS) and two applications for the verifier side (with/without Secure Access Module - SAM). The runtimes of the authentication and communication processes on smart cards confirm that the solution is competitive to the related work. Further, the performance evaluation proves the high efficiency of the proposed authentication scheme.

Moreover, the security analysis demonstrates that the solution prevents from current attacks. The solution provides the strong security levels (224-bit/256-bit/384-bit for elliptic curves) in concordance with the NIST recommendations for 2017 - 2030 and beyond. The proposed system offers strong security and the practical runtime of the authentication process in order to be attractive for access control systems based on current contactless smart cards.

4.1.2 Background

This subsection presents a used notation, primitives and a system model used in a proposed solution.

Used Notation and Primitives

The proposal employs the interactive Proof of Knowledge (PK) of discrete logarithm, i.e., the Schnorr's identification scheme [171]. The Schnorr's identification scheme is also the part of ISO/IEC 9798-5:2009 that specifies entity authentication mechanisms using zero-knowledge techniques. The authentication protocol is based on the modified Schnorr's scheme, and uses elliptic curves. Employing elliptic curves helps to keep the sizes of parameters on a moderate level which increases the efficiency in terms of communication and computational overheads. Moreover, two random values are used, and the interactive proof of knowledge (a random challenge from a verifier) and the non-interactive proof of knowledge (a random value on the prover side) are combined in order to increase security in the proposed system.

Tab. 4.2: Notation and Parameters used.

E	an elliptic curve over a finite field $\mathbb{F}_p$
p	a prime number which defines finite field $\mathbb{F}_p$
q	an integer order of the point $G : q * G = 0$, where 0 is a point at infinity
a	an elliptic curve parameter
b	an elliptic curve parameter
G	a base point of E, $G \in_R E(\mathbb{F}_p)$
ID_i	a point of E, serves as a public key (user ID)
ID_{i8B}	an abbreviated user ID of ID_i, this abbreviated ID (64 bits) stored in a white list
a_i	a random number in $\mathbb{Z}_q$, serves as a user's private key
k	a random number in $\mathbb{Z}_q$
R	a point of E, the commitment of PK
d	a random number (the first challenge) from the verifier in $\mathbb{Z}_q$
e	a data fingerprint (the second challenge) from the user
z	a response in $\mathbb{Z}_q$ based on the knowledge of the private key a_i and values e, k
R'	a restored commitment on the verifier side
e'	a restored data fingerprint on the verifier side
$H()$	a secure hash function
$Attribute$	a user's attribute
$version$	a used version of the protocol

The proposal uses points of an elliptic curve E over a finite field $\mathbb{F}_p$, where G is a base point of E. The symbol "$\cdot$" denotes modular multiplication, "$\bullet$" denotes scalar EC point multiplication, ":" means "such that", "|" means "divides","||" denotes "concatenation" of values, "$|x|$" is the bitlength of x, and "$x \in_R \{0,1\}^l$" is a randomly chosen bitstring of given length l. The notations and parameters used in the proposed solution are defined in Table 4.2.

System Model

The system model of the proposed authentication system is depicted in Fig 4.1. The main entities in the system are defined as follows:

- **User (U)** - a user with a contactless smart card requires an access into protected areas/services. Users use the authentication protocol to prove their right for access into protected areas/services.
- **Verifier (V)** - a verifier is a device that contains a card reader for contactless communication ISO 14443 with users' smartcards. **V** can contain a Secure Access Module (SAM) for managing the authentication protocol or the verifier application can be run on common PC with a USB card reader. **V** decides about a user access or denial according to the result of the authentication protocol and user's presence in the whitelist. **V** is securely connected to a manager who maintains the whitelist of users.

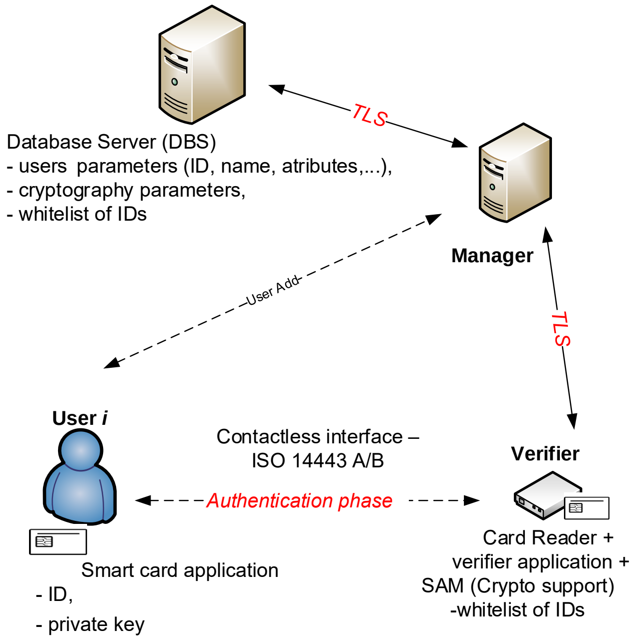


Fig. 4.1: System model.

- **Manager (M)** - a manager is a trusted server which generates cryptographic parameters. **M** also adds and revokes users. **M** is connected to other system entities via secured connections (e.g., Transport Layer Security - TLS).
- **Database Server (DBS)** - **DBS** serves as a central storage unit that maintains the user parameters, the whitelist with user IDs, cryptography parameters and other attributes. It is assumed that this entity is placed in a secured location inside a protected area. **DBS** is connected to the manager and verifiers via secured connections (e.g., TLS).

4.1.3 Proposed System

The proposed authentication system consists of 4 basic phases (`Setup, Join, Authentication, Revocation`). These phases are defined as follows:

`Setup` phase

$(sysparam) \leftarrow \texttt{Setup}(t, l)$ – The `setup` phase provides the generation of cryptographic keys and parameters. Cryptographic parameters are securely generated by the manager and are securely stored in the database server. The database server also stores revocation lists, certificates (used by TLS) and the whitelists of identities. The input parameters t, l define the security level of the authentication scheme, where t presents the length of the hash function and l is related to the length of U's private keys. The manager chooses an EC over a finite field $E(\mathbb{F}_p)$ with the domain

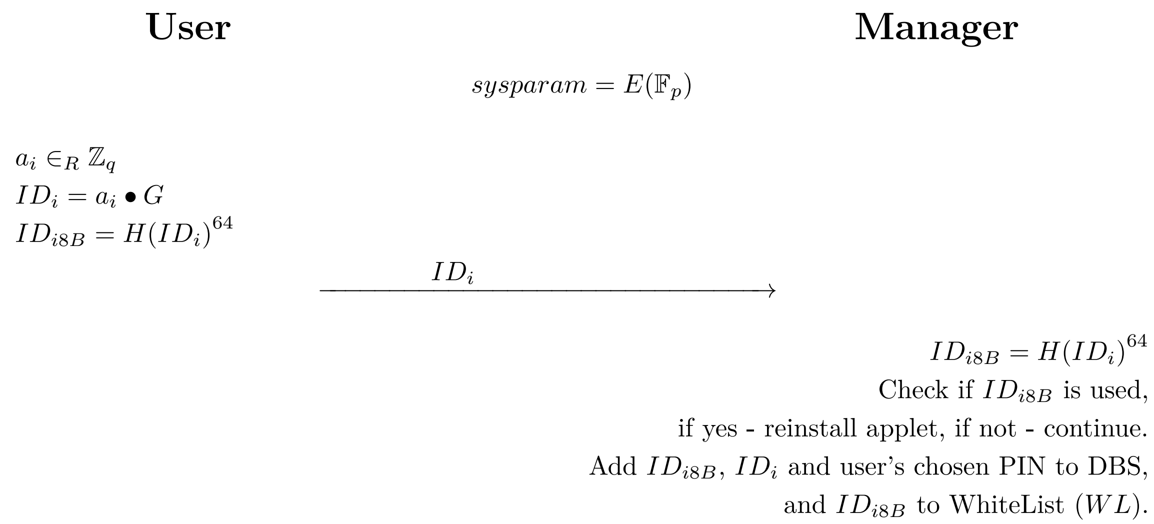

User
Manager
$sysparam = E(\mathbb{F}_p)$
$a_i \in_R \mathbb{Z}_q$
$ID_i = a_i \bullet G$
$ID_{i8B} = H(ID_i)^{64}$
ID_i
$ID_{i8B} = H(ID_i)^{64}$
Check if ID_{i8B} is used,
if yes - reinstall applet, if not - continue.
Add ID_{i8B}, ID_i and user's chosen PIN to DBS,
and ID_{i8B} to WhiteList (WL).

Fig. 4.2: `Join` phase of proposed system.

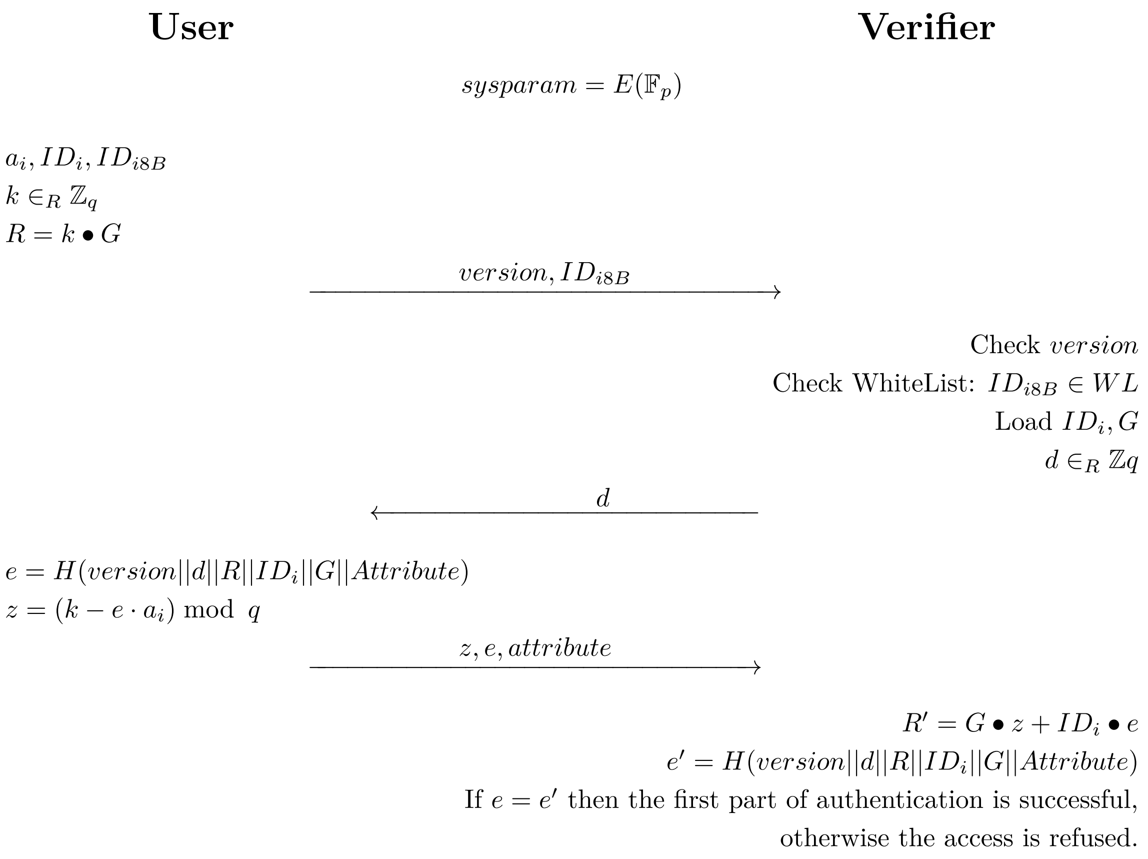
parameters (a, b, p, q, G, h), where p is an big prime number specifying the field $\mathbb{F}_p$, $a, b \in \mathbb{F}_p$ are coefficients of the EC, G is an EC point generator $G = (x_G, y_G)$ of order q, and h is the cofactor defined as $h = \#E(\mathbb{F}_p)/q$.

The system parameters $sysparam = E(\mathbb{F}_p)$ are made public. The lengths of all parameters meet NIST recommendations, i.e., 224 bits for elliptic curves.

`Join` phase

$(Q_i, a_i, ID_{i8B} \in WL) \leftarrow \texttt{Join}(sysparam)$ – This phase runs between User and Manager in a trusted environment. The protocol is depicted in Figure 4.2. Each User downloads the application for his/her smart card from Manager. During the installation of the application on the smart card, the user private key a_i is randomly generated and the smart card also computes unique public value $ID_i = a_i \bullet G$. The private key a_i is securely stored only in the user's smart card and it is generated only during the `join` phase.

If the manager adds the new user into the system, the public key ID_i is computed and stored in the user's smart card, and it is also exported from the smart card into the database server. The new user also provides his/her PIN to the manager in order to provide the second level of the authentication (the user knowledge). Finally, the manager sets user's rights in the system, adds the abbreviated user ID ID_{i8B} into the WhiteList (WL) for the suitable terminals and user's PIN into the secured database.



User **Verifier**

$sysparam = E(\mathbb{F}_p)$

a_i, ID_i, ID_{i8B}

$k \in_R \mathbb{Z}_q$

$R = k \bullet G$

$\xrightarrow{version, ID_{i8B}}$

Check $version$

Check WhiteList: $ID_{i8B} \in WL$

Load ID_i, G

$d \in_R \mathbb{Z}q$

$\xleftarrow{d}$

$e = H(version||d||R||ID_i||G||Attribute)$

$z = (k - e \cdot a_i) \bmod q$

$\xrightarrow{z, e, attribute}$

$R' = G \bullet z + ID_i \bullet e$

$e' = H(version||d||R||ID_i||G||Attribute)$

If $e = e'$ then the first part of authentication is successful, otherwise the access is refused.

Fig. 4.3: `Authentication` phase of proposed system.

`Authentication` phase

$(AuthOK/NOT) \leftarrow$ `Auth`$(sysparam, ID_i, a_i, WL)$ – This phase runs between User and Verifier in an untrusted environment. The phase is depicted in Figure 4.3.

The `authentication` phase is based on a zero knowledge authentication protocol over $E(\mathbb{F}_p)$ as follows:

- The user initiates the authentication protocol by calculating a commitment $R = k \bullet G$ where k is a random number in $\mathbb{Z}_q$. The user sends his/her short public key ID_{i8B} and $version$ to the verifier (**V**).
- **V** checks $version$ and selects the suitable version of the cryptographic suite (e.g., both sides must support same elliptic curves). Then, **V** checks the presence of ID_{i8B} in the whitelist for the used version. If WL contains ID_{i8B}, the phase continues. **V** loads ID_i, G, q and generates a random number d (the first challenge) in $\mathbb{Z}_q$ and sends this number to the user.
- The user verifies that d is in Z_q and calculates the data fingerprint e (the second challenge) by using the hash function $H(version||d||R||ID_i||G||\ Attribute)$. The attribute field is optional, e.g., day/night access, area restriction, group

restriction etc. Further, the user calculates the response $z = k - e \cdot a_i$ mod$q$. The parameters $z, e, Attribute$ are sent to the verifier.

- The verifier restores a cryptographic commitment $R' = G \bullet z + ID_i \bullet e$ and the data fingerprint $e' = H(version||d||R'||ID_i||G||Attribute)$. Then, the verifier compares the e' value with the data fingerprint (e) from the user. If $e = e'$, the first authentication part is evaluated as successful, otherwise, the verifier rejects the user access and sends a log report to the manager. This part can work also if a verifier is offline.
- The second part of the user authentication extends the `authentication` phase in order to check user's knowledge (the second level of authentication). This phase requires online verifiers equipped with a keyboard where users input their PINs. The verifier securely sends ID_{i8B}, *Attribute* and PIN by TLS to the manager. The manager with the actual **DBS** that contains ID_8B and user's PIN checks the user ID and compares the received user PIN with the PIN stored in the database (**DBS**). If PINs are equal and the short ID is in the whitelist, then the manager decides about user's access. Checking the PIN helps to prevent from attacks with the stolen cards.

`Revocation` phase

$(ID_{i8B} \notin WL) \leftarrow$ `Revoke`$(sysparam, ID_i/ID_{i8B}, WL)$ – The manager is able to immediately revoke a user (e.g., a user with a stolen/lost smart card, a leaving user). If a user is revoked, his/her ID is removed from the whitelist by the manager. The manager must refresh the whitelist used on the verifier.

4.1.4 Security Analysis

In this subsection, the security analysis of the proposed authentication system is outlined. The proposed system provides the following security properties:

- The system provides **completeness** - If a valid user and a valid verifier follow the protocol, then the valid verifier always accepts user's proof of identity (ID_i, z, e). The valid user knows the private key a_i, therefore he/she can always compute a correct response $z = k - e \cdot a_i \bmod q$ which is always accepted by the verifier in the final checks $R' = G \bullet z + ID_i \bullet e = G \bullet (k - e \cdot a_i) + (a_i \bullet G) \bullet e = k \bullet G - G \bullet e \cdot a_i + a_i \cdot e \bullet G = k \bullet G = R$ where restored R' is used in $e' = H(version||d||R'||ID_i||G||\ Attribute)$ that must be equal to received e from the user.
- The system provides **soundness** - The soundness property requires that no user is able to make a verifier accept a wrong statement $(ID_{i8B}/ID_i \notin WL, z, e)$ except with some small probability $P = \frac{1}{2^t}$ where t is the length of the output

(e) of the hash function used. The `authentication` phase uses the combination of the interactive proof of knowledge (random challenge d from a verifier) and the Fiat-Shamir heuristic (non-interactive proof of knowledge) which produces a digital signature by one-way function (a hash function). The user and the verifier firstly generate randomness (values d, k) where k is the part of the commitment R. Both values R, d are used as inputs for the hash function that produces e (a challenge) that is then used as the input for the response z. The user is unable to predict e before the selection of the random values. The more detailed proof of the soundness property regarding the non-interactive proof of knowledge can be found in [98].

- The system provides **linkability and traceability** - Every system entity is able to link the users access by the user ID (ID_{i8B}) that is sent during every `authentication` phase. The users can be easily tracked by the manager in order to control user's access to various areas or to various services. On one hand, these properties are opposite to privacy-preserving features. For example attribute authentication schemes protect user anonymity and provide unlinkability and untraceability. On the other hand, the authentication solution tends to be more secure for services/locations where traceability is more important. Hence, users should be aware that they can be monitored and traced by the access control system.
- Protection against **the verifier impersonation** - The verifier, the manager and database servers do not store the user private keys a_i. Therefore, even if an adversary accesses these entities and servers, he/she does not obtain necessary authentication information of users a_i and is not able to successfully authenticate in the later attempts. The private key is used during the computation of the user public key $ID_i = a_i \bullet G$. If the user public key ID_i leaks to the adversary, then in order to get the private key, he/she has to resolve the Elliptic Curve Discrete Logarithm Problem (ECDLP). ECDLP is defined as follows: given a public key $ID_i = a_i \bullet G$ and base point G, it is hard to compute private key a_i.
- Protection against **the user impersonation** - If a valid user or an adversary attempts to impersonate another valid user by stealing his/her ID, he/she is not able to successfully authenticate on the verifier side (i.e., the final step of Fig. 4.3). Even if the adversary eavesdrops parameters (i.e., ID_{i8B}, d, z, e) and forges these values, he/she is not able to provide the valid proof or recompute valid response z without the knowledge of the user private key a_i on a refreshed input d from the verifier. It is assumed that the user private key is securely stored on user's smart card and its extraction from this storage by an adversary is hard.

- Protection against **the private key extraction** - The user uses his/her private key a_i during computing the response $z = k - e \cdot a_i$ mod$q$ and during computing his/her public key $ID_i = a_i \bullet G$. To get a_i from z is hard due to k is an ephemeral secret value that never leaves the user's smart card. Getting the private key from the public key is hard and based on ECDLP that is defined above. Due to avoiding RSA scheme, the proposed solution is also resistant to new Coppersmith's Attack (ROCA) published in [155].
- Protection against **the replay attack** - The `authentication` phase uses a nonce-based authentication mechanism that protects against the replay attacks. It is assumed that an adversary is able to eavesdrop all parameters. Further, a valid response z is computed from randomly generated nonce d from a verifier and the output e of the hash function where is inserted the parameter R computed from randomly generated parameter k on the user side. Then, replaying the parameters z and e and making the authentication process successful is negligible and can be done only if a verifier would send the same d. There is a very low probability that the attacker will be able to use d from the previous sessions for successful authentication in the future. This attack depends on the size of d and the quality of the TRNG function on the verifier side. It is worth noting that the proposed solution does not require a clock synchronization. Moreover, the current smart cards usually do not provide APIs for obtaining the actual time or date.
- Protection against **the data modification** - An adversary who attempts to modify the authentication parameters (i.e., ID_{i8B}, *version*, *Attribute*) in the protocol will be detected by a verifier. The system uses a one-way hash function to ensure that the parameters cannot be modified. An adversary cannot obtain the unique parameter R further before that is used to generate the legitimate hash value e if it is assumed that the hash function is a secure one-way function where getting x from given $y = H(x)$ is hard. R can be restored (e.g., by the verifier) only after all parameters were computed and sent.
- Protection against **the man-in-the-middle attack** - An adversary as the man-in-the-middle is able to eavesdrop and modify communication in order to try to steal login credentials, masquerade or impersonate one of the parties. The attacker can change nonce d from the verifier but he/she does not learn any private key from the user. Then, he/she is not able to recompute the response z without the knowledge of the valid user private key a_i in order to impersonate the user. In the zero knowledge protocol all parameters are bounded together and only the usage of the private key and the corresponding user ID (ID_{i8B}) returns a successful authentication. The protection properties

against the user impersonation and verifier impersonation are discussed above.

- Protection against **the off-line private key guess attack** - If an adversary tries to get private key a_i, he/she must try to compute such $z = (k - e \cdot a_i) \bmod q$ that is valid in the protocol. The attack is negligible without knowledge k that is the secret random value which never leaves a smart card. The success of the brute force attack on the private key depends on the size of k that is usually $\in_R \mathbb{Z}_q$.
- Protection against **the leakage of ephemeral secret values** - This attack is described in the paper [122]. The protocol generates the ephemeral secret value (k) directly on the smart card. Nevertheless, it is assumed that k and a_i never leave the secure storage of the smart card.
- Protection against **the side channel attacks** - Side-channel attacks analyze physical characteristics (called leakages) of cryptographic devices related to the execution of the implementation of a cryptographic algorithm. The physical analysis tries to extract a secret value, i.e., a user private key. There is a relationship between the manipulated data, the executed operations and the physical properties observed during the execution of the cryptographic device. The physical properties that can be extracted are, for example, the execution time of a cryptographic algorithm [121], the electromagnetic emanation [88] or the power consumption of the device [120]. Considering the ECC algorithms, the execution of scalar multiplication can leak the information of private key in many ways. Therefore, a variety of side channel attacks can be utilized in order to reveal secret information. In the system, an adversary aims at the `join` phase when the public key is computed by point multiplication $ID_i = a_i \bullet G$, and at the `authentication` phase when the response is computed as $z = k - e \cdot a_i \bmod q$. Naturally, if these crucial operations are not implemented in respect of countermeasures techniques, the adversary can reveal the private key without significant troubles. Nevertheless, in general, the realization of the side channel attack is not an easy task and it requires a big amount of effort depending on a concrete scenario. In order to counteract these side-channel attacks, the countermeasure has to be added by implementers of cryptographic protocols. Generally, countermeasure techniques are divided into two basic groups: masking [170], [205] and hiding [219]. In the masking approach, each intermediate value is concealed by a random mask. By contrast, hiding tries to break the link between the power consumption and the actual processed data values. Every countermeasure technique can be implemented in software, hardware or in the protocol level [45]. Our implementation is protected by software countermeasure techniques. The intermediate values are randomized during the calculation of scalar multiplication.

4.1.5 Performance Evaluation

This subsection presents the experimental implementation and the performance evaluation of the proposed system.

Experimental Implementation of Proposed System

The proposed authentication solution is implemented on current smart cards. The Basic Cards and MultOS Cards platforms are chosen due to the support of all cryptographic primitives such as modular arithmetic operations with big integers and operations over elliptic curves. JAVA Cards provide standard cryptographic operations but some modular arithmetic operations are offered only in JAVA Card OS version 3.0.5. Nevertheless, JAVA cards with JC OS 3.0.5 are usually not available at common smart card vendors.

Both sides a user with a smart card and a verifier with/without a SAM module are implemented. The system uses two communication smart card standards: ISO/IEC 14443 and ISO/IEC 7816. The standard ISO/IEC 14443 defines transmission protocols for communicating between Proximity Integrated Circuit Cards (PICC), i.e., contactless smart cards used for identification/authentication, and Proximity Coupling Device (PCD), i.e., a card reader with an interface for contactless cards. The ISO/IEC 7816 defines transmission protocols for communicating between a reader and a contact smart card (SAM modules, ID-000 smart cards). The communication model of the proposed solution is depicted in Fig. 4.4.

The verifier side is implemented in the JAVA programming language and can be run on an embedded machine and on classic PC with the installed JAVA runtime. The main JAVA implementation uses several cryptographic libraries such as Bouncy Castle library and javax.crypto. Therefore, the main verifier application with support these libraries on PC is able to verify the user without a SAM module. Nonetheless, some embedded verifier machines without possibility to call cryptographic APIs used on the verifier side (e.g., TRNG, hash function, ECC operations) must integrate SAM modules in order to call these APIs from SAM. In the proposed system, SAM is represented by a contact Basic card in the ID-000 format.

The communication between the main JAVA application and the contact smart card (SAM module) or contactless user smart cards is provided by the javax.smartcardio library that manages API for processing Application Protocol Data Units (APDU). Further, two applications for a prover side that uses a contactless smart card are implemented. The first application runs on MultOS Cards and the second application runs on Basic Cards.

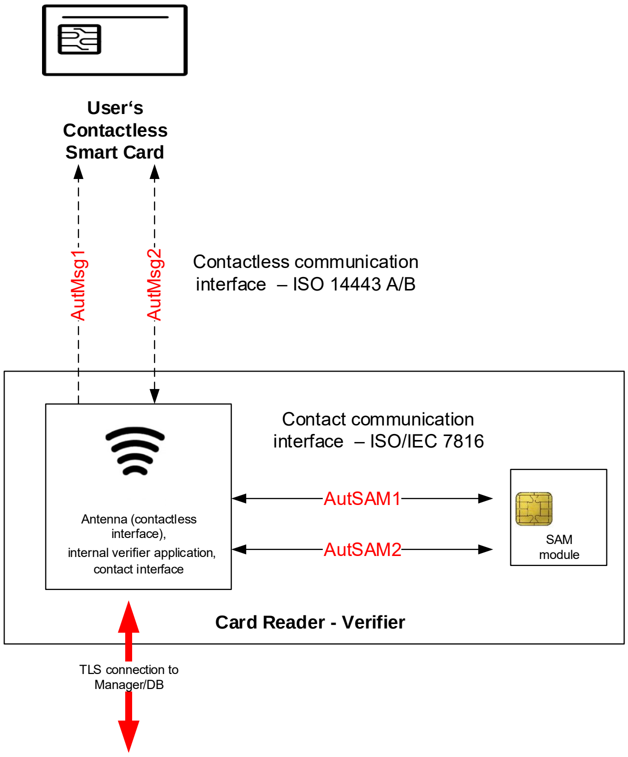

User's Contactless Smart Card
AutMsg1
AutMsg2
Contactless communication interface – ISO 14443 A/B
Contact communication interface – ISO/IEC 7816
Antenna (contactless interface), internal verifier application, contact interface
AutSAM1
AutSAM2
SAM module
Card Reader - Verifier
TLS connection to Manager/DB

Fig. 4.4: Communication model of proposed system.

Parameter Setup

The authentication system provides three versions that use elliptic curves (NIST curves P-224 - Version 1, P-256 - Version 2, P-384 - Version 3). All curves are supported on MultOS and Basic Cards. Table 4.3 presents the chosen lengths of parameters for all three versions that meet NIST requirements for 2017 - 2030.

Proposed APDU messages

During the authentication protocol, 5 APDU messages are sent. 4 APDU messages are employed in the `authentication` phase and one APDU message is used in the `join` phase.

The `authentication` phase is divided into these APDU messages with the following lengths:

- APDU AutMsg1 between the reader and the contactless user smart card (sent data length: 0 B for Version 1, 0 B for Version 2, 0 B for Version 3; data response length: 9 B for Version 1, 9 B for Version 2, 9 B for Version 3).
- APDU AutSAM1 between the reader and the SAM module (sent data length: 57 B for Version 1 , 65 B for Version 2, 97 B for Version 3; data response length: 28 B for Version 1, 32 B for Version 2, 48 B for Version 3).

Tab. 4.3: Chosen Lengths of Parameters for Version 1, 2 and 3.

Version:	Version 1	Version 2	Version 3
Parameter:	[bits/Bytes]		
p	224/28	256/32	384/48
a	224/28	256/32	384/48
b	224/28	256/32	384/48
Gx	224/28	256/32	384/48
Gy	224/28	256/32	384/48
q	224/28	256/32	384/48
h	8/1	8/1	8/1
a_i	224/28	256/32	384/48
ID_i	456/57	520 /65	776/97
ID_{i8B}	64/8	64/8	64/8
k	224/28	256/32	384/48
d	224/28	256/32	384/48
$H()$	224/28	256/32	384/48
R	456/57	520/65	776/97
e	224/28	256/32	384/48
z	224/28	256/32	384/48
R'	456/57	520/65	776/97
e'	224/28	256/32	384/48
$Attribute$	256 /32	256 /32	256 /32
$Version$	8/1	8/1	8/1

- APDU AutMsg2 between the reader and the contactless user smart card (sent data length: 28 B for Version 1, 32 B for Version 2, 48 B for Version 3; data response length: 88 B for Version 1, 96 B for Version 2, 128 B for Version 3).
- APDU AutSAM2 between the reader and the SAM module (sent data length: 88 B for Version 1, 96 B for Version 2, 128 B for Version 3; data response length: 1 B for Version 1 , 1 B for Version 2, 1 B for Version 3).

Measurement of Communication Delay

The time of communication between a card reader and a smart card by APDU messages over contact (ISO/IEC 7816) and contactless (ISO/IEC 14443) interfaces is measured. Then, the real bit rate speed for these interfaces is calculated. Communication delay is an important factor in practical deployment. This implementation utilizes 4 APDU messages during the authentication process that are shown in Fig. 4.4.

In the experimental measurement, three cards are used: contactless Basic ZC7.5 rev.A card (SC1-CL), contact Basic ZC7.6 rev.D card (SC2-C), and contact Java NXP J3D081 card (SC3-C). The processing and communication times of APDUs with different amounts of data (0, 8, 16, 32, 64, 128 and 256 bytes) are measured for two scenarios (reading data from card, sending data to card). Table 4.4 shows the results for the first scenario where data are sent to a card, i.e., data transmitting (TX). Table 4.5 presents the results for the second scenario where data are read from

Tab. 4.4: Times in ms for Send Data by APDU to Smart Cards (Tx).

Message Size [B]	0	8	16	32	64	128	256
Time [ms] with SC1-CL	3.3	5	5.1	6.6	9	13.1	23.2
Time [ms] with SC2-C	4.2	5.2	7	9.4	14.4	25	49.8
Time [ms] with SC3-C	11.9	14	15.3	18.4	25.4	40.4	72.6

Tab. 4.5: Times in ms for Read Data by APDU from Smart Cards (Rx).

Message Size [B]	0	8	16	32	64	128	256
Time [ms] with SC1-CL	4.4	4.9	6.1	7.2	9.5	14	21.3
Time [ms] with SC2-C	4.4	5.2	5.8	6.4	9.8	14.6	26.4
Time [ms] with SC3-C	12.8	13.1	15.8	18.9	27.1	40.5	73

a card, i.e., data receiving (RX). All results are averaged over 10 measurements. To be noted that the smart cards do not compute any functions in this concrete experiment and only read or send data.

The results of send or read of 0 B data in APDU indicate times needed for the communication of empty APDU (only APDU header - 4 bytes) and the internal initialization (smart card OS, applications, codes) on the card. There is a difference between the JAVA Card platform ($\approx$ 12 ms) and the Basic Card platform ($\approx$ 4 ms). Basic Card platform needs less time for initialization than Java Cards due to their simplicity. The data transmission speed usually depends on many parameters such as conductance/RF field strength (from 0 to 63), modulation (from 0 to 63), signal gain (from 1 to 3), threshold level (by a PN512 datasheet), and RF level (by a PN512 datasheet). It is strictly prohibited to change these parameters. Nevertheless, a developer can restrict maximum speed for receiving (Rx) or transmitting (Tx) directions, e.g., 0 (106 kbps), 1 (212 kbps), 2 (424 kbps), 3 (848 kbps) for contactlesss cards.

Contact cards are usually limited to 9600 Bd transmission rate but some newer cards can reach 38400 Bd. According to the results in Tables 4.4, 4.5, the measured times for Tx and Rx directions and for same amount of data are similar on SC1-CL and SC3-C. Nonetheless, the SC2-C card obviously chooses two different modulation rates during data receiving and transmitting, therefore the results for Tx and Rx directions are different.

Based on the measurements, the real bit rate speed of APDU communication and processing (RS_{exp}) on the card are computed by this equation:

$$RS_{exp} = \frac{256}{A_{256} - A_0} \cdot 1000 \cdot 8[bps], \tag{4.1}$$

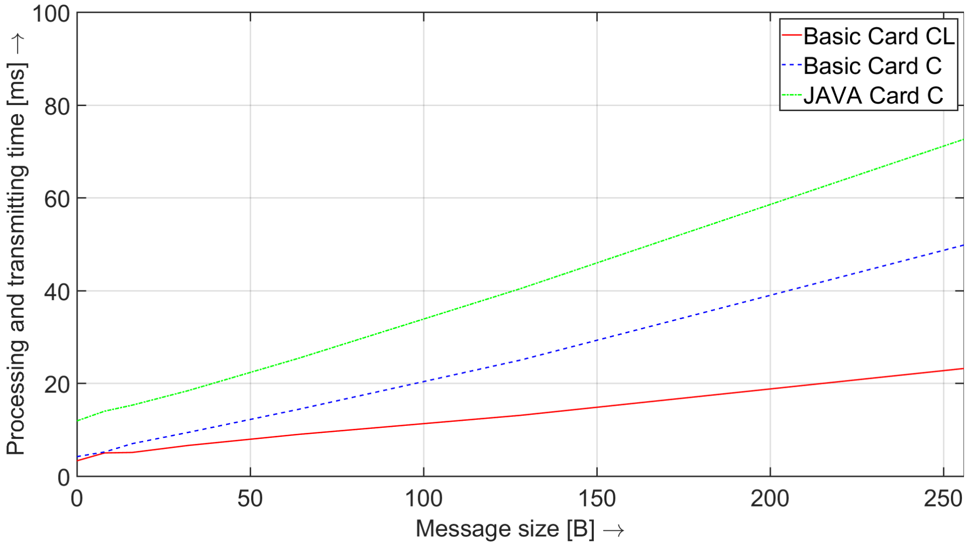


Fig. 4.5: The processing and transmitting time (Tx) for various APDU data sizes on Basic Cards and Java Cards.

where A_{256} denotes the transmission and processing time of APDU with 256 B data, A_0 denotes the transmission and processing time of APDU without data (0 B).

The contactless card SC1-CL (Basic card ZC7.5 rev.A) reaches this bit rate speed:

$$RS_{exp} = \frac{256}{23.2 - 3.3} \cdot 1000 \cdot 8 \cong 102915[bps]. \tag{4.2}$$

The computed value (RS_{exp} = 102915bps) is close to a minimal bit rate speed (106 kbps) defined by the ISO/IEC 14443 standard.

The contact card SC2-C (Basic card ZC7.6 rev. D) reaches this bit rate speed:

$$RS_{exp} = \frac{256}{49.8 - 4.2} \cdot 1000 \cdot 8 \cong 44912[bps]. \tag{4.3}$$

The bit rate speed for the Tx direction is higher (93090 bps) than the bit rate speed for Rx direction (44912 bps) on the SC2-C card. Nevertheless, the manufacture claims the maximal modulation rate is 38400 Bd.

The contact card SC3-C (Java card NXP J3D081) provides this bit rate speed:

$$RS_{exp} = \frac{256}{72.6.2 - 11.9} \cdot 1000 \cdot 8 \cong 33739[bps]. \tag{4.4}$$

It is close to the speed claimed by the manufacture which is 38400 bps (i.e., 9600 bauds, $M = 16$, with the bit rate $N = \log_2(16) = 4$).

Fig. 4.5 depicts how the processing and transmitting (Tx) time grows with the increase of APDU data size. Therefore, using the APDU messages with reasonable lengths is essential for the authentication protocol efficiency. The next subsection presents the performance results of the authentication protocol that includes communication delay, Smart Card OS initialization and the times needed for cryptographic operations.

Measurement of Authentication

The following text presents the performance of the `authentication` phase between a smart card and a verifier (a reader) with SAM and a verifier without SAM. The `join` phase is not measured due to the fact that this phase is done only once when a user is added in a system.

The runtimes of the `authentication` phase for 3 versions are measured. All results are averaged over 10 measurements. The SAM module is represented by contact ID-00 Basic Card ZC 7.6 rev. D and contact MultOS card with OS ML4.3.1. As the user smart card, Basic Card ZC 7.6 rev. D with the dual interfaces (contact and contactless) and the contactless MultOS card OS ML4.3.1 are chosen. The verifier without SAM runs on PC with CPU Intel i5-2450M (2.5 GHz, 2 Cores) and 6 GB RAM (1333 MHz).

Table 4.6 presents the results of the `authentication` phase for all 3 versions with 2 card platforms (Basic Card and MultOS Card). The times needed for communication and processing of all 4 APDU messages during the `authentication` phase are shown. The total runtimes of the `authentication` phase are presented for 6 combinations: the MultOS and Basic smart cards with the SAM module-equipped verifier and the MultOS and Basic with the PC verifier (without SAM). The total time of the `authentication` phase between the Basic Card and PC takes less than 0.5 s for all 3 versions. Versions 1 and 2 in all 6 combinations take less than 1 s.

Performance Comparison

The performance of the proposed `authentication` phase with several related works is compared. All compared schemes consist of several cryptographic operations. Fast operations with runtimes < 2 ms on a smart card are omitted, e.g., XOR, truncate of arrays, bit rotations and random number generator operations. Table 4.7 depicts the cryptographic operations, used notation and measured runtimes on the Basic Card ZC 7.6 without communication overhead. The measured times of the operations depend on the security level (key sizes) and chosen cryptographic primitives.

The performance evaluations and comparisons in [152] and [207] use the security level with 160 bits for $\mathbb{Z}_q$ and 1024 bits for $\mathbb{F}_p$.

The comparison employs additive notation and the 160/1024-bit security level despite the fact that the proposed system is designed for higher security levels with

Tab. 4.6: Runtimes [ms] during `Authentication` Phase for Different Versions.

Communication and processing times of messages	Version 1	Version 2	Version 3
AutMsg1 (MultOS Card)	147	160	244
AutMsg2 (MultOS Card)	180	193	261
Total time on user side (MultOS Card)	327	353	505
AutMsg1 (Basic Card)	225	255	380
AutMsg2 (Basic Card)	71	72	99
Total time on user side (Basic Card)	296	327	479
AutSAM1 (Multos Card)	149	162	210
AutSAM2 (Multos Card)	285	305	429
Total time on verifier side with SAM (Multos Card)	434	467	640
AutSAM1 (Basic Card)	77	83	97
AutSAM2 (Basic Card)	470	530	680
Total time on verifier side with SAM (Basic Card)	547	613	777
AutSAM1 (PC)	2	2	2
AutSAM2 (PC)	10	12	15
Total time on verifier side with PC	12	14	17
Runtime of authentication with various devices:	Version 1	Version 2	Version 3
Total time (MultOS Card - SAM MultOS Card)	761	820	1145
Total time (Basic Card - SAM MultOS Card)	730	794	1119
Total time (MultOS Card - SAM Basic Card)	874	966	1282
Total time (Basic Card - SAM Basic Card)	843	940	1256
Total time (MultOS Card - PC)	339	367	522
Total time (Basic Card - PC)	308	341	496

224 b, 256 b, 384 b curves. In order to fairly compare the system with related works, it is assumed 160/1024-bit parameters, i.e., 160-bit curves.

Moreover, several authentication solutions assume that smart card is employed only on the prover side. Therefore, the number of operations needed on the prover side only is used. Table 4.8 presents the number of atomic operations during the `authentication` phase on the user smart card (the prover side) for the proposed system and related works. In addition, it is compared communication overheads, i.e., the total bitlength [b] of all messages during the `authentication` phase.

Tab. 4.7: Runtimes of Cryptography Operations on Basic Card ZC 7.6.

Abbreviation	Operation	Time [ms]
T_{pm}	Time of one scalar EC point multiplication over $E/\mathbb{F}_p$	156
T_{mm}	Time of one modular multiplication on $\mathbb{F}_p$	11
T_{pa}	Time of one point addition over $E/\mathbb{F}_p$	23
T_{ma}	Time of one modular addition on $\mathbb{F}_p$	2
T_s	Time of one symmetric encryption/decryption (e.g., AES-256)	2
T_h	Time of one hash function (e.g., SHA-256)	13

Fig. 4.6 presents the estimated runtimes of `authentication` phases on a smart card side (a prover side) for the proposed system and related schemes (Choi *et al.*'s scheme [65], Wu *et al.*'s scheme [207], Mishra *et al.*'s scheme [152], Teh *et al.*'s scheme [189] and the PACE scheme [28]). The estimated runtimes are computed by the number of operations (Table 4.8) multiplied by atomic operation runtimes from Table 4.7. The results do not contain communication overhead and initial times of a smart card OS. Nevertheless, the results indicate that the proposed solution is more efficient than the related schemes on the prover side.

Tab. 4.8: Comparison with Related Work.

Scheme	Cryptographic operations on the prover side	Communication cost [b]
Choi *et al.*'s scheme [65]	$3T_{pm} + 7T_h$	4220
Wu *et al.*'s scheme [207]	$2T_{pm} + T_s + 11T_h$	3712
Mishra *et al.*'s scheme [152]	$3T_{pm} + 11T_h$	896
Teh *et al.*'s scheme [189]	$2T_{pm} + 1T_{pa}$	2368
PACE [28]	$5T_{pm} + T_{pa} + 4T_h + 1T_{ma} + T_s$	896
Our solution	$T_{pm} + T_{mm} + T_h + T_{ma}$	648

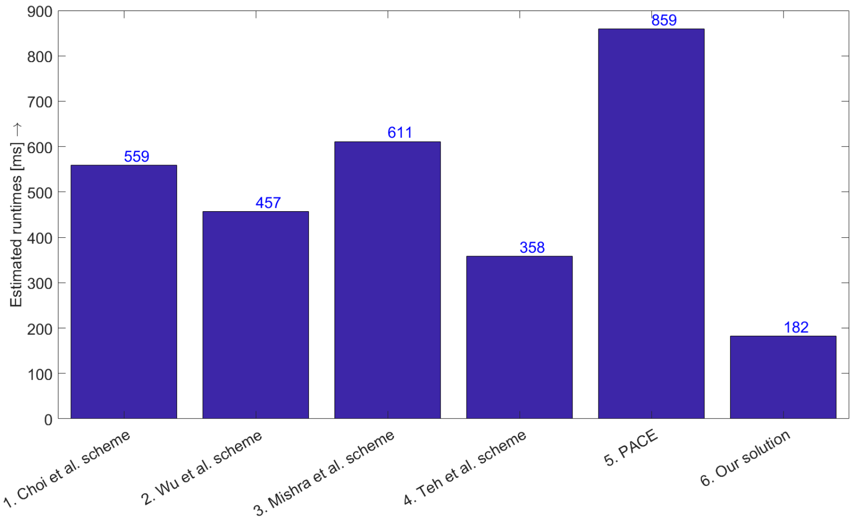


Fig. 4.6: Estimated runtimes [ms] of compared schemes during `authentication` phases on smart card side.

4.1.6 Summary

This section presents the secure and efficient authentication system based on zero-knowledge authentication. The provided implementations and experimental measurement prove that the proposed system can be deployed on current off-the-shelf programmable smart cards, i.e., Basic Cards and MultOS Cards. The experimental measurements of times for data transmission on current smart cards indicate that schemes with many APDU messages and sizable parameters may have significant communication delay, on the contact interface especially. The communication and cryptographic costs of the proposed solution is well balanced due to using elliptic curves. The measured runtimes of the authentication phase are practical for all 3 solution versions using various lengths of elliptic curves. Moreover, the proposed system is more efficient than most related authentication schemes. Further, the solution allows us to trace users' IDs. This feature can be used for the immediate revocation of malicious users and leaving users. The proposed solution uses the secure cryptographic primitives and key lengths that meet the NIST requirements for 2017 - 2030. This proposed system can be used in access control and identification systems deployed in secure environments.

4.2 Secure and Privacy-preserving Data Transfer System Based on Light-Weight Group Signatures with Time-Bound Membership

This section presents a novel proposal of a secure and privacy-preserving data transfer system for many-to-one communications, data collection services, data gathering services, vehicular networks, smart grids, etc. The proposed system provides message authenticity, integrity and non-repudiation while message senders are anonymous and untraceable. The system is based on cryptographic group signatures with a time-bound membership. The system is designed to achieve efficiency on the client side where constrained devices are usually employed. On the other hand, the verification of many messages is efficient as well.

The amended version of the text below is part of the author's paper in the journal with an impact factor, namely, Security and Communication Networks, [17]. The proposed system improves and significantly enhances the basic idea of author's scheme published as the MHM13 scheme in [146].

4.2.1 Introduction, State of the Art and Contribution

Nowadays, digital information and communication technologies provide various services and applications that gather, collect and act on various information. Data gathering, which is usually based on many-to-one communication, collects data from users/clients to a central server or several servers. Data security is an important issue for many service providers who manage data collection services. The collected data must be original and come from the clients who are authenticated in the communication system. Besides common security requirements such as data authenticity and integrity, user privacy becomes an essential requirement as well.

There are several ways how to achieve security and privacy in many-to-one communication systems which collect data from a large group of clients. Security designers usually use the special types of zero-knowledge protocols or group signature schemes together with common cryptographic schemes (the ElGamal encryption scheme [81], AES [73], ECDSA [115], etc.). Nevertheless, employing the group signature schemes, which contain many expensive pairing operations, is not optimal for systems where constrained devices are used. Moreover, the revocation of users makes current group signature schemes more expensive. Also the revocation using key/accumulator update mechanisms is not suitable for large and heterogeneous networks with many client devices. On the other hand, a solution based on a group signature scheme that is efficient on the client side and mitigates the expensiveness of

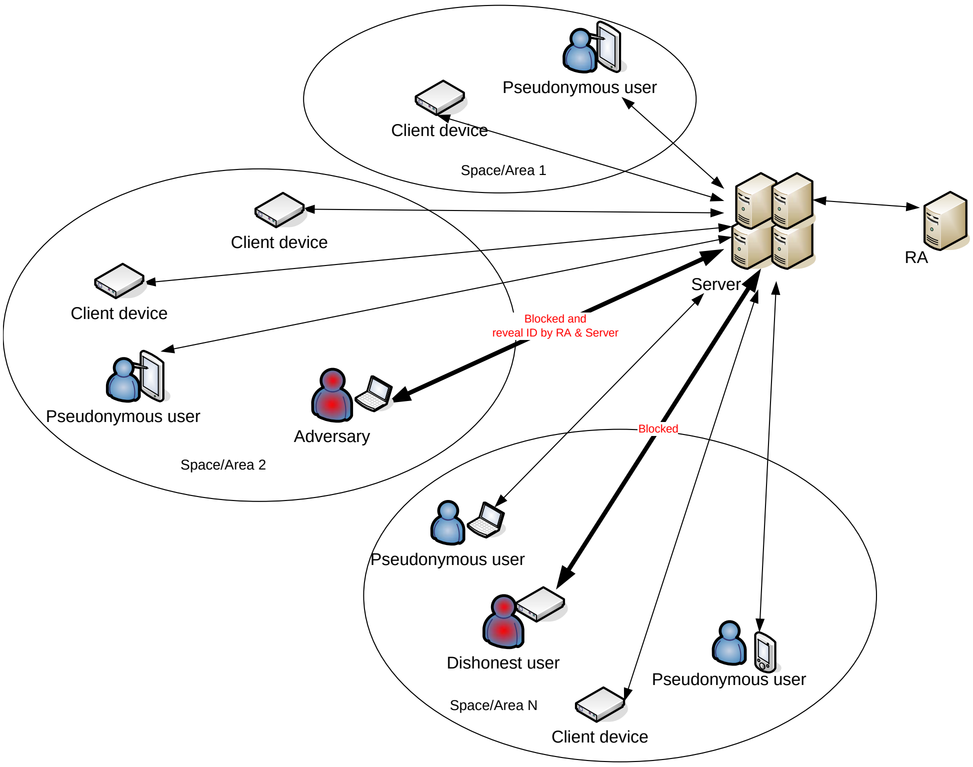


Fig. 4.7: Basic scenario of many-to-one communication.

the revocation may be implementable for communication systems with many users.

The goal of the proposal is to design and develop an efficient cryptographic protocol providing the security and user privacy in many-to-one communication systems. The privacy-preserving system is based on a group signature scheme that can be employed in data collection applications where clients anonymously send data to a server. The basic scenario is depicted in Figure 4.7. The honest clients stay anonymous for observers, other clients and service providers. On the other hand, if a client breaks a policy then his/her identity can be revealed. The system provides user privacy, message authenticity and message integrity in communication between clients and a server.

State of the Art

Recently, several papers have aimed at privacy-aware systems that collect data from clients and proposed privacy-preserving and security frameworks.

Yang *et al.* [214] deal with an anonymity-preserving data collection and propose a solution for anonymous and online data collection services. Their solution is based on the ElGamal encryption scheme [81] and zero-knowledge proofs. Nevertheless, the paper [50] presents some flaws in their solution. The paper [50] presents another protocol for an anonymity-preserving data collection. The protocol does not rely on zero-knowledge proofs and provides an online-verifiable shuffle in order to be practical for data mining applications. In comparison to this protocol, our solution is designed to work at the application layer and focuses on the efficiency while providing the revocation of users.

Shin *et al.* [180] propose a framework for anonymous opportunistic sensing. The framework ensures the privacy of users and protects the integrity and confidentiality of the reported messages. The framework is based on the existed cryptographic schemes. User privacy is achieved by using the BBS04 scheme [43]. The framework provides sending the report messages that are signed by using the BBS scheme. Nevertheless, the paper does not deal with the revocation of malicious users and omits optimization techniques applied on the group signature scheme that may improve the efficiency of the framework.

Rottondi *et al.* [168] propose a pseudonymization framework for data gathering by smart meters. To perform data anonymization, the framework is based on the secret sharing scheme proposed by Shamir [178]. Other techniques such as Chaum mixing [61] and Identity-Based Proxy Re-Encryption scheme [94] are supported as well. Nevertheless, the framework focuses on the smart-grid applications where certain nodes that perform data pseudonymization are employed. The proposed framework has not been designed for many-to-one communication scenarios.

Other research works in smart-grid and smart-metering, e.g., [67] and [128], often require different security properties, e.g., message linkability, tracebility, etc. The paper [51] describes Scalable Secure Transport Protocol SSTP (presented in [118]) as an appropriate solution for securing the smart grid measurement data. Nevertheless, SSTP is designed for smart grid networks and does not deal with privacy and data pseudonymity. Yukun *et al.* [216] present a secure and privacy-preserving scheme for data collection from smart meters by using homomorphic encryption. Nevertheless, the proposed scheme needs a trusted party for data checking. Also, the authentication of users is not addressed in this paper.

People-centric sensing systems also collect data from clients. The paper [91] redefines several privacy and security requirements and evaluates several state-of-the-art solutions with respect to these requirements. In the paper [91], the scheme called PEPSI (Privacy-Enhanced Participatory Sensing Infrastructure) [72] has been evaluated as a promising solution. This solution is based on the blind and anonymous identity based encryption and provides node privacy and data unlinkability. Nevertheless, the user of this solution needs to employ a trusted hardware and his/her mobile node must perform 2 bilinear pairing operations during sending the data report to a service provider. On the other hand, the second disadvantage can be overcame by Oblivious Pseudo-Random Functions (OPRF) but mobile nodes and their quires, which subscribe the information collected, cannot be disjointed.

Further, several works deal with privacy in online electronic services, e.g., [117], and in anonymous geolocation and geosocial systems such as [204] and [145]. However, these schemes are usually not convenient for anonymous/pseudonymous many-to-one communication systems that collect data from client devices.

We aim mainly at solutions that employ group signature schemes. There are only few efficient group signature schemes such as the BBS04 scheme [43], [76] and [111] that need 0 pairing operations in the signing phase and are suitable for computationally restricted clients. These schemes usually revoke users by a key update mechanism that is inconvenient for large groups. On the other hand, the group signature schemes with the verifier local revocation, e.g., [44] and [66], are more suitable for large groups but these schemes take several computationally expensive pairing operations during the signing phase.

The efficient verification is important on the verifier side which receives many signed messages and must be as fast as possible. There are few group signature scheme proposals, e.g., [83], [116] and [142], which employ the batch verification techniques to increase the efficiency of the verification phase.

Another problem is the unrestricted increase of a revocation list if the verifier local revocation technique is used, e.g., in the group signature scheme which is proposed by Boneh and Shacham (BS04) [44]. Chu *et al.* [66] propose to use time-bound secret group member keys that are revoked by a time expiration. The expired keys can be removed from the revocation list because the keys become invalid after a certain time. Their scheme is based on the group signature scheme BBS04 [43] and tries to address all security issues of group signatures, e.g., forward security, backward unlinkability, etc. The disadvantage of their scheme is the fact that many pairing and modular operations are used in signing and verification phases. Hence, such a scheme is not appropriate for restricted devices used in the client side. The group signature scheme [146], which is based on the BS04 scheme [44], implements the time-bound secret keys from the work [66]. The scheme [146] preserves privacy and security and is more efficient than the scheme [66]. Nevertheless, the signing phase still needs to compute 2 expensive pairing operations that make the scheme inconvenient for restricted clients.

In this section, a privacy-enhancing cryptographic protocol based on the group signatures is designed. The system is suitable for data collection services and is applicable at the application layer. The system focuses on services that are based on many-to-one communication and may employ constrained devices on the client side.

Contribution

The contribution of the proposed system is outlined in the following text:

- **User privacy in many-to-one communication.** We use strong cryptographic primitives to achieve the pseudonymous and secure many-to-one communication which is suitable for systems such as data collection and gathering services. Every honest client sends data to a server without leakage his/her identity (ID). Messages sent by a single client are not linkable. Moreover, a service provider with his/her server cannot reveal the clients' IDs and/or track their activities and communication. A trusted third party manages the clients' IDs.
- **Practical security.** The solution is designed to be practical even on computationally constrained devices. The proposed solution offers the signing phase with no pairing operations and the efficient verification phase with the batch verification algorithm that reduces the number of pairing operations to a constant number. This is suitable for systems with a large number of clients. The experimental implementation proves the efficiency of the solution.
- **Practical revocation.** Servers issuing data gathering/collection services are able to block revoked clients by using a revocation list. The natural time expiration of the clients' group secret keys helps to reduce the size of the revocation list. The revocation process rids the clients who behave maliciously. A trusted third party can reveal their identities and can revoke them. Nevertheless, if the client's device is stolen or compromised by an attacker, there is an option to revoke the client temporarily for a certain time.

4.2.2 Background

This subsection presents cryptography used in the proposed solution and a system model.

Cryptography Used

The proposed system is based on a **group signature scheme** proposed by Chu *et al.* (Chu12) [66], which is based on the group signature scheme BBS04 [43] that ensures anonymity, authenticity, message integrity, non-repudiation, unlinkability and tracebility. The group signature scheme uses bilinear pairings and is based on the q-SDH problem, the DL problem and the DDH problem. The problems are described in [66]. This scheme is modified to ensure more efficient signing and verification algorithms that are more suitable for many-to-one communication systems with a large number of clients. The signing algorithm is optimized by the precomputation trick published in [43] and [55]. The verification algorithm is improved by a batch verification. The verifier-local revocation with time-bound group member secret keys is provided. The system employs the methods called 0-encoding/1-encoding presented in [66] to make time-bound group secret member keys. The scheme uses a minimum of bilinear pairing operations [43].

0/1-Encoding The 0-Encoding and 1-Encoding reduce the *greater than* predicate to the *set intersection* predicate by converting a date format into a binary string to a value in $\mathbb{Z}_p$. To convert the elements of binary strings to the value in $\mathbb{Z}_p$, it is used the procedure presented in [66] which is defined as follows:

1. It is used the 0/1-Encoding of a l-bit binary string $t = t_{[l]}t_{[l-1]}...t_{[1]}$, where t is the date encoded in a binary string and $t_{[i]}$ denotes the i-th bit of t by
   $T_t^0 = \{t_{[l]}t_{[l-1]}...t_{[i+1]}1 | t_{[i]} = 0, 1 \leq i \leq l\}$,
   $T_t^1 = \{t_{[l]}t_{[l-1]}...t_{[i]} | t_{[i]} = 1, 1 \leq i \leq l\}$.

   Based on the theorem in [132], $x > y$ iff T_x^1 and T_y^0 have a common element.

2. It is ensured that the sets start with '1' by adding '1' such as
   $\overline{T_t^0} = \{1 \cdot 10^{l-i+1} + t_{[l]} \cdot 10^{l-i} + t_{[l-1]} \cdot 10^{l-i-1} ... t_{[i+1]} \cdot 10^1 + 1 | t_{[i]} = 0, 1 \leq i \leq l\}$,
   $\overline{T_t^1} = \{1 \cdot 10^{l-i+1} + t_{[l]} \cdot 10^{l-i} + t_{[l-1]} \cdot 10^{l-i-1} ... t_{[i+1]} \cdot 10^1 + t_{[i]} | t_{[i]} = 1, 1 \leq i \leq l\}$.

3. The sets are filled up with incomparable dummy elements to achieve an equal number of elements:
   $\{t_l, t_{l-1}, ..., t_1\} \leftarrow$ 0-ENC(t), where $t_i \leftarrow \{z \text{ if } z \in \overline{T_t^0} \wedge 2 \cdot 10^i \text{ otherwise}\}$,
   $\{t_l, t_{l-1}, ..., t_1\} \leftarrow$ 1-ENC(t), where $t_i \leftarrow \{z \text{ if } z \in \overline{T_t^1} \wedge 3 \cdot 10^i \text{ otherwise}\}$.

The following text presents the example published in [139] in order to clarify the 0/1-Encoding method. The example uses two dates y='1301' and x='1303' (2013-January and 2013-March) in the date format 'YYMM'. The 0/1-Encoding indicates which of dates is the newer one. If common element appears then $x > y$. Dates '1301' and '1303' are encoded into binary strings as y=101 0001 0101 and x=101 0001 0111. The offset based on a number of months from the present can map much longer time period by using the same lengths of bits.

We employ the 0/1-Encoding on $x = 10100010111, y = 1010001010$ and get:

$$T_y^0 = \{11, 1011, 10101, 101001, 10100011,\ 1010001011\},$$
$$T_x^1 = \{1, 101, 1010001, 101000101, 1010001011,\ 10100010111\},$$

$$\overline{T_y^0} = \{111, 11011, 110101, 1101001,\ 110100011, 11010001011\},$$
$$\overline{T_x^1} = \{11, 1101, 11010001, 1101000101,\ 11010001011, 110100010111\},$$

$\{20, 111, 2000, 11011, 110101, 1101001, 20000000,$ $110100011, 2000000000,$ **11010001011**,200000000000$\} \leftarrow$ 0-ENC(y),
$\{11, 300, 1101, 30000, 300000, 3000000, 11010001,$ $300000000, 1101000101,$ **11010001011**,110100010111$\} \leftarrow$ 1-ENC(x).

If the element **11010001011** is in both sets then $x > y$. The sketch of the proof can be found in [66].

System Model

The system model consists of three parties:

- Registration Authority (RA) - We assume that RA is a trusted party. RA initializes group signature parameters, one group public key, one group manager secret key and group member secret keys. RA also manages a registration list, a revocation list that includes revoked users, and the global reference clock for the synchronization of time stamps used in the system. RA is able to trace a signer from a message and a valid signature.
- Server (S) - We assume that S is managed by a service provider. S checks only signed messages by a group public key and if a user is on the revocation list or not. S also has the database of clients with their IDs and their states but IDs are not linkable with clients' group member secret keys.
- Client (C) - C is a user who correctly joins a group. C can sign any message by his/her group member secret key and send the signature with the message to server S.

4.2.3 Proposed System

In this subsection, the proposed system is outlined. The system consists of six main phases: setup, join, sign, verify, open and revocation. The system is based on the group signature schemes [43] and [66] and is efficient by using time-bound secret keys with the batch verification and non-pairing signing.

Setup

In the `setup phase` **Setup**(λ) $\rightarrow$ ($parameters$, $keys, gpk, gmsk$), RA and S set ECDSA keys, and RA sets ElGamal keys and group signature parameters, a group public key and a group manager secret key as follows:

- Based on the length of a security parameter λ, group signature parameters $\mathbb{G}_1, \mathbb{G}_2, g_1, g_2,$ ψ, e are established since $g_1 = \psi(g_2)$ if $e(\psi(g_2), g_1) \neq 1$. H is a

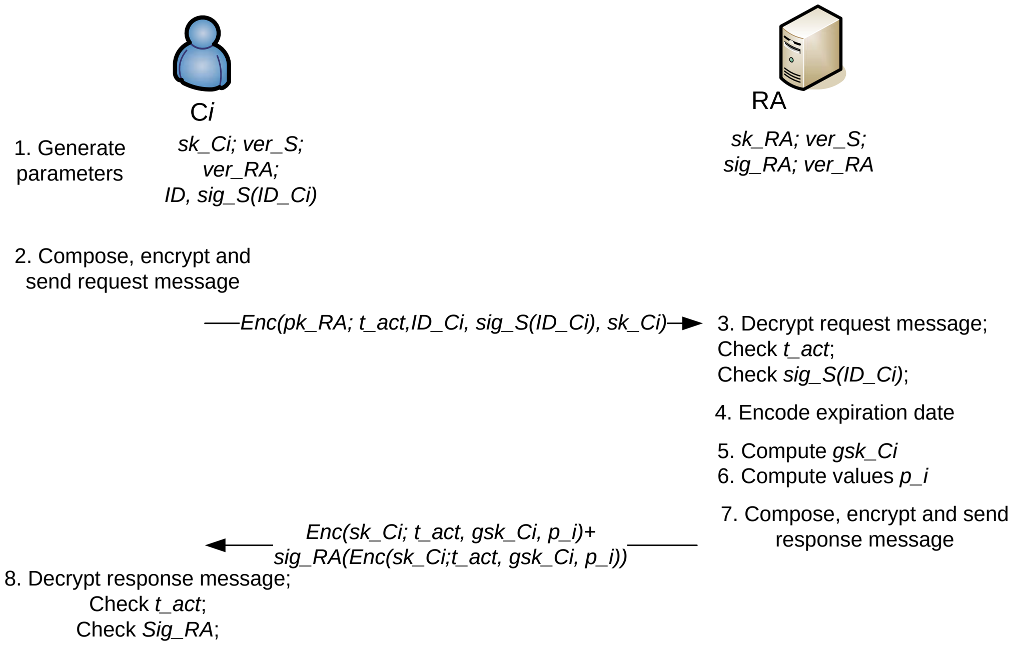

Ci
RA
1. Generate parameters
sk_Ci; ver_S; ver_RA; ID, sig_S(ID_Ci)
sk_RA; ver_S; sig_RA; ver_RA
2. Compose, encrypt and send request message
Enc(pk_RA; t_act,ID_Ci, sig_S(ID_Ci), sk_Ci)
3. Decrypt request message; Check t_act; Check sig_S(ID_Ci);
4. Encode expiration date
5. Compute gsk_Ci
6. Compute values p_i
7. Compose, encrypt and send response message
Enc(sk_Ci; t_act, gsk_Ci, p_i)+ sig_RA(Enc(sk_Ci;t_act, gsk_Ci, p_i))
8. Decrypt response message; Check t_act; Check Sig_RA;

Fig. 4.8: Join phase.

hash function in range $\mathbb{Z}_p$. RA issues the group signature scheme parameters and keys. RA randomly selects $\gamma \in \mathbb{Z}_p^*, h \in \mathbb{G}_1^*$. Then, RA computes $w = g_2^\gamma$. The group public key is $gpk = (g_1, g_2, h, w)$. The group manager secret key is $gmsk = (\gamma)$. RA sends gpk to servers.

- RA generates ElGamal private (sk_{RA}) and public (pk_{RA}) keys. The ElGamal encryption [81] is used for secure sending group member secret keys and revocation tokens. To ensure data authentication, the ECDSA signature scheme [115] is used. RA generates an ECDSA key pair sig_{RA}/ver_{RA} and publishes the public ECDSA key (ver_{RA}). RA manages and distributes the global reference clock which is used in time stamps by other entities (clients and S) in the system.
- S generates an ECDSA key pair (sig_S/ver_S) and publishes the public ECDSA key (ver_S).

Join

In the `join phase` (see Figure 4.8), the i-th client C$_i$ joins a group which is managed by RA or only refreshes his/her gsk_{U_i} which is expired. The `join phase` **Join**$(ID, gpk, gmsk) \rightarrow (gsk_{C_i}, p_{1ik}, p_2, p_3, p_4)$ is performed between RA and C$_i$ as follows:

1. The client generates his/her secret key sk_{C_i} used for the secure download of a group member secret key and other parameters by symmetric encryption,

e.g., AES [73]. Then, it is assumed that the client owns the signed ID (e.g., the serial number of client's device with the manufacture date, etc.) from a service provider by sig_S, e.g., by a physical registration, a web registration.

2. The client C_i encrypts an actual time stamp, the client's ID, the signed client's ID by sig_S and the generated secret key sk_{C_i} by RA's ElGamal public key pk_{RA} and sends the cipher text by a request message to RA.
3. RA receives the encrypted request message and decrypts the content by his/her ElGamal private key sk_{RA}. Further, RA checks the freshness of the message by the time stamp, the validity of signed ID by ver_S and the client's status (e.g., permanently revoked, temporary revoked, unrevoked) from the database obtained from a service provider.
4. Based on the variable values such as the length of the revocation list, the reputation of C_i by ID, the client's status, RA decides about the duration of an expiration date τ_i for the group member secret key gsk_{C_i}. RA encodes the expiration date τ_i by the 1-Encoding: $\{\tau_{ij}\}_{j\in[1,l]} \leftarrow$1-ENC($\tau_i$) where l is the length of a date format. As a proper data format, we use the 8-bit ofset from the current month.
5. For $(j = 0; j \leq l; j + +)$, RA selects $x_{ij} \overset{R}{\leftarrow} \mathbb{Z}_p^*$ and sets $A_{ij} \leftarrow g_1^{\frac{1}{\tau_{ij}\gamma+x_{ij}}} \in \mathbb{G}_1$, where $\tau_{ij}\gamma + x_{ij} \neq 0$.
6. RA precomputes the public pairing values p_{1ij} for each A_{ij} and p_2, p_3, p_4:

$$\begin{aligned} &p_{1ij} = e(A_{ij}, g_2), p_2 = e(h, g_2),\\ &p_3 = e(h, w), p_4 = e(g_1, g_2), \end{aligned} \tag{4.5}$$

   where elements $p_{1ij}, p_2, p_3, p_4 \in \mathbb{G}_T$.
7. RA stores a revocation token $(\tau_i, \{x_{ij}\})$, the group member secret key $(\tau_i, \{A_{ij}, x_{ij}\})$ and the client's ID. RA responses to the client by sending the actual time stamp, the group member secret key, the group public key and public precomputed pairing values p_{1ij}, p_2, p_3, p_4 via a secured connection which is encrypted by the client's secret key (sk_{C_i}). Moreover, this response message from RA is signed by the RA's ECDSA private key (sig_{RA}) to ensure data authenticity.
8. C_i checks the signature on the response message by the RA's ECDSA public key (ver_{RA}), and if it is valid, then C_i decrypts the response message from RA by the client's secret key. Further, the client checks if the time stamp is actual. If the message is actual, C_i saves the values (gsk_{C_i} and precomputed pairing values).

Signing

Every client C$_i$ who sends a message to a server signs this message by a group signature scheme. Every C$_i$ has a group member secret key $gsk_{C_i} = \tau_i, \{A_{ij}, x_{ij}\}$ and a group public key $gpk = (g_1, g_2, h, w)$. C$_i$ signs a message $M \in$ (0,1)* and outputs the group signature $\sigma = (t_{cur}, k, B, K, T, c,\ s_\alpha, s_x, s_\delta, R_3)$.

The `Signing phase` **Sign**$(M, gsk_{C_i}, gpk, t_{cur}) \rightarrow \sigma$ is performed by C$_i$ as follows:

- C$_i$ checks if gsk_{C_i} is not expired by $t_{cur} < \tau_i$, where t_{cur} is a current date (e.g., a current month or a current date in the format 'YYMMDDHHMMSS' as in [66]) or the date of the signature expiration. If $t_{cur} \geq \tau_i$, the algorithm halts. The t_{cur} is parsed on the 8-bit ofset month value $t_{cur'}$.
- The dates are converted into *intersection check* by the 0/1-Encoding: $\{\tau_{ij}\}_{j\in[1,l]} \leftarrow$1-ENC$(\tau_i)$ and $\{t_j\}_{j\in[1,l]} \leftarrow$0-ENC$(t_{cur'})$ where l is the length of the date format used.
- The index $k \in \{1, 2, ..., l\}$ is found such that $\tau_{ik} = t_k$ and the pair of A_{ik}, x_{ik}, p_{1ik} from gsk_{C_i} is selected.
- C$_i$ chooses random elements $\alpha, r_\alpha, r_x, r_\delta \in \mathbb{Z}_p^*$ and $B \in \mathbb{G}_1$.
- C$_i$ loads precomputed parameters p_2 and p_3.
- C$_i$ computes the group signature by the following steps:

$$K = B^{x_{ik}}, T = A_{ik}h^\alpha, \tag{4.6}$$

$$\delta = \alpha x_{ik}, \tag{4.7}$$

$$\begin{aligned} &R_1 = B^{r_x}, R_2 = K^{r_\alpha} B^{-r_\delta}, \\ &R_3 = \ p_1^{-r_x} p_2^{-r_x\alpha + r_\delta}\ p_3^{r_\alpha \tau_{ik}}, \end{aligned} \tag{4.8}$$

where elements $K, T, R_1, R_2 \in G_1,\ R_3 \in G_T$ and $\delta \in \mathbb{Z}_p^*$,

$$c = H(gpk, t_{cur}, M, B, K, T, R_1, R_2, R_3), \tag{4.9}$$

where $c \in \mathbb{Z}_p$,

$$\begin{aligned} &s_\alpha = r_\alpha + c\alpha, \\ &s_x = r_x + cx_{ik}, \\ &s_\delta = r_\delta + c\delta. \end{aligned} \tag{4.10}$$

where elements $s_\alpha, s_x, s_\delta \in \mathbb{Z}_p^*$.

- C$_i$ sends the message M with the signature $\sigma = (t_{cur}, k, B, K, T, c, s_\alpha, s_x, s_\delta, R_3)$.

Verification

A server (S) verifies each message received from a pseudonymous client. This solution employs a batch verification which makes the verification phase more efficient. The batch verification algorithm, which is analyzed in [83], allows to verify n messages in a single batch. If the server receives more messages during a certain time period then these messages are verified by the batch verification algorithm. If S receives only one message during this period then this message can be verified by the individual verification algorithm.

The `Individual verification algorithm` **InVerify** $(M, gpk, \sigma,\ t_{act}, RL) \rightarrow$ valid/invalid is performed by S as follows:

- The time validity of the signature is checked by $t_{act} > t_{cur}$, if *yes* then the algorithm halts. To continue the algorithm, the value t_{cur} must be equal or newer than the actual date t_{act} measured by the verifier.
- The date t_{cur} is firstly parsed on the ofset value $t_{cur'}$ and is converted into the *intersection check* by the 0-Encoding: $\{t_j\}_{j\in[1,l]} \leftarrow$0-ENC$(t_{cur})$ and by k from the signature is found t_k.
- S restores $\overline{R}_1$ and $\overline{R}_2$:
$$\overline{R}_1 = B^{s_x} K^{-c}, \overline{R}_2 = B^{-s_\delta} K^{s_\alpha}. \tag{4.11}$$
- S computes a new control hash value c' from the received and restored values: $c' = H(gpk, t_k, M, B, K, T, \overline{R}_1, \overline{R}_2, R_3)$,
and checks if $c' = c$. If yes, then S continues. If no, the individual verification halts and the signature of the message is marked as invalid and this message is refused.
- S checks if
$$R_3 = e(T^{-s_x} h^{s_\delta} g_1^c, g_2) e(h^{s_\alpha} T^{-c}, w^{t_k}). \tag{4.12}$$
- The signed message is valid if Equation 4.12 holds.

- The verification phase continues by the revocation check in the following subsection.

Then, the server performs the revocation check process. The server opens the actual revocation list RL =$(\tau_i, \{x_{ij}\})$ with r revoked users where $j \in [1, l]$ and $i \in [1, r]$ to check if the signed message is received from a revoked or unrevoked user. The `Revocation check algorithm` **RevCheck**$(RL, \sigma) \rightarrow$ revoked/unrevoked is performed as follows:

- For each i-pair of $\tau_i, \{x_{ij}\}$, S recomputes by the 1-Encoding: $\{\tau_{ij}\} \leftarrow$1-ENC(τ_i) and finds the index m $(1 \leq m \leq l)$ such that $\tau_{im} = t_k$ and checks if
$$K = B^{x_{im}}. \tag{4.13}$$

- If Equation 4.13 holds then the client's signed message will be discarded because the i-th client with x_{im} is revoked.

If a client must be revoked, then RA sends the refreshed revocation list with a new revocation token grt to the server. Further, the server discards old records with obsolete pairs $\tau_i, \{x_{ij}\}$ to reduce the length of RL. If S receives more messages in one short period, then S can verify these signed messages in one batch. S firstly checks that the z-th received message M_z contains the real and consistent data of a service. If yes, then S inputs this signed message to the batch verification algorithm, otherwise, the message is refused. The batch verification algorithm is valid only if all group signatures of the messages are valid. If the batch verification is not valid, then the invalid signatures must be identified by the individual verification. We employ the divide and conquer approach to enhance this process. The `Batch Verification algorithm` **BatchVerify**$(M_1, M_2, .., M_n, \sigma_1, \sigma_2, .., \sigma_n, gpk, t_{act}, RL) \rightarrow$ valid/invalid. S uses $gpk = (g_1, g_2, h, w)$ to verify n messages with $\sigma_z = (t_{zcur}, k_z, B_z, K_z, T_z, R_{z3}, c_z, s_{z\alpha}, s_{zx}, s_{z\delta})$ for $(z = 1; z \leq n; z++)$, and does:

- S checks the time validity (of the signature) by $t_{act} > t_{zcur}$. If *yes*, then the algorithm aborts. To continue the algorithm, the value t_{zcur} must be equal or newer than the actual date t_{act} measured by the server.
- The date t_{zcur} is firstly parsed on the ofset value $t_{zcur'}$ and is converted into *intersection check* by the 0/1-Encoding: $\{t_{zj}\}_{j \in [1,l]} \leftarrow$0-ENC$(t_{zcur})$ and by k_z from the signature is found t_{zk}.
- S restores $\overline{R}_{z1}$ and $\overline{R}_{z2}$:

$$\overline{R}_{z1} = B_z^{s_{zx}} K_z^{-c_z}, \overline{R}_{z2} = B_z^{-s_{z\delta}} K_z^{s_{z\alpha}}, \tag{4.14}$$

- S computes a new control hash value c'_z from the received and restored values: $c'_z = H(gpk, t_{zcur}, M_z, B_z, K_z, T_z, \overline{R}_{z1}, \overline{R}_{z2}, R_{z3})$, and checks if $c'_z = c_z$. If yes then S continues. If no, the z signed message is refused but S continues with the batch verification algorithm only with the consistent messages.
- S randomly selects $\theta_1, \theta_2, ..., \theta_n \in Z_p$ with l_b bit, checks the batch if

$$\prod_{z=1}^{z=n} R_{z3}^{\theta_z} = e(\prod_{z=1}^{z=n} (T_z^{-s_{zx}} h_z^{s_{z\delta}} g_1^{c_z})^{\theta_z}, g_2) e(\prod_{z=1}^{z=n} (T_z^{-c_z} h_z^{s_{j\alpha}})^{\theta_j}, \prod_{z=1}^{z=n} (w^{t_{zk}})) \tag{4.15}$$

- The batch with signed messages is valid if Equation 4.15 holds.
- S performs `Revocation check algorithm` to ensure that there are no messages from already revoked users.

Equations 4.12 and 4.15 show that the individual verification algorithm costs 2 pairing operations per 1 message but the batch verification algorithm costs only 2 pairing operations per n messages.

Open

The open phase `Open` $(gpk, \sigma, M, grt_i) \rightarrow ID_i$. RA stores group member secret keys $(\tau_i, \{x_{ij}\})$ and IDs of all users. Every correctly signed message M with the group signature σ and the group public key can be opened by RA and a user index i, which is connected with a user ID stored in a database, can be revealed by checking Equation 4.13 for each revocation token. To perform this phase, S and RA must collaborate. S has to send the message M and the valid signature σ to RA. RA uses grt_i to reveal the client ID from his/her database.

Revocation

If the client has to be revoked and has the unexpired group member secret key, then RA can put this client (his/her $\tau_i, \{x_{ij}\}$) onto the revocation list (RL) and sends it to the server. After that, if the revoked client tries to send the new signed message, then S checks refreshed RL and blocks the signed message. Moreover, RA can decide if the client is revoked permanently (the complete revocation token) or temporary (a part of the revocation token).

4.2.4 Security Analysis

This subsection presents the security analysis of the proposed system and possible threats. It is assumed that an attacker can eavesdrop messages, can tamper and resend messages. Furthermore, it is assumed that the attackers do not have computational power that allows them to break current cryptography schemes that are considered to be secure.

Adversaries can be external or internal. Internal adversaries can be some clients or a server (S). It is assumed that the Registration Authority (RA) as the trusted third party is fully trusted. On the other hand, the server can be controlled by a private company (a service provider). It is expected that S performs the phases in an honest way and does not tamper with messages, refuses valid messages, etc. However, S may try to gather personal data, retrieve clients' identities or make clients' profiles, which is a privacy threat. Thus, it is assumed that S can break the privacy of clients.

This security analysis assumes attacks that are passive and active. An adversary must have access to the communication to make a passive attack. The attacker can try to compromise clients' anonymity and/or message unlinkability by tracking and linking messages sent by a certain client. Then, he/she can try to retrieve the identity of the client and can eavesdrop messages transmitted between C_i and RA in the Join phase, and in the communication between C_i and S. Active attackers can try to modify valid messages, generate fake messages, etc. Then, the message integrity and/or message authenticity can be compromised. The active adversaries try to modify messages transmitted between C_i and RA in the Join phase, and in the communication between C_i and S. The attacker can try to create fake but valid messages during the join phase and during the communication between clients and the server. Further, active attackers and unauthorized clients may try to generate fake messages and perform a replay attack by sending the captured messages.

The proposed group signature scheme is correct.

If a non-revoked client creates the group signature with a non-expired key, this signature is verified as valid. Otherwise, the signature is invalid. The verifier (Server) checks that c equals c'. To obtain the valid c', he/she must correctly restore $\overline{R_1}, \overline{R_2}$ by computing Equation 4.11, and then, he/she checks if the received R_3 is correct by computing Equation 4.12. The proposed scheme is correct if $\overline{R_1} = R_1, \overline{R_2} = R_2$ and the received R_3 holds in Equation 4.12. The proof of correctness is shown in the following equations 4.16, 4.17 and 4.18:

$$\overline{R_1} = B^{s_x} K^{-c} = B^{r_x + c x_{ik}} K^{-c} = B^{r_x} B^{c x_{ik}} B^{-x_{ik} c} = B^{r_x} = R_1, \tag{4.16}$$

$$\overline{R_2} = B^{-s_\delta} K^{s_\alpha} = B^{-r_\delta - c\delta} B^{x_{ik}(r_\alpha + c\alpha)} = B^{-r_\delta - c\alpha x_{ik}} B^{x_{ik}(r_\alpha + c\alpha)} = B^{r_\delta} K^{r_\alpha} = R_2, \tag{4.17}$$

$$\begin{aligned}
R_3 &= e(T^{-s_x} h^{s_\delta} g_1^c, g_2) e(h^{s_\alpha} T^{-c}, w^{t_k}) \\
&= e(T, g_2)^{-s_x} e(h, g_2)^{s_\delta} e(h, w)^{t_k s_\alpha} e(g_1, g_2)^c e(T, w^{t_k})^{-c} \\
&= e(T, g_2)^{-r_x} e(h, g_2)^{r_\delta} e(h, w)^{t_k r_\alpha} (e(T, g_2)^{-x} e(h, g_2)^{x\alpha} e(h, w)^{t_k \alpha} e(g_1, g_2)/e(T, w^{t_k}))^c \\
&= R_3 (e(Ah^\alpha, g_2)^{-x} e(h^\alpha, w^{t_k} g_2^x) e(g_1, g_2)/e(T, w^{t_k}))^c \\
&= R_3 (e(Ah^\alpha, g_2)^{-x} e(h^\alpha, g_2^{\lambda t_k + x}) e(g_1, g_2)/e(T, w^{t_k}))^c \\
&= R_3 (e(Ah^\alpha, g_2)^{-x} e(g_1^{1/(\lambda t_k + x)} h^\alpha, g_2^{\lambda t_k + x})/e(T, w^{t_k}))^c \\
&= R_3 (e(Ah^\alpha, g_2^{-x}) e(Ah^\alpha, g_2^{\lambda t_k + x})/e(T, w^{t_k}))^c = R_3 (e(T, w^{t_k})/e(T, w^{t_k}))^c \\
&= R_3 1^c = R_3.
\end{aligned} \tag{4.18}$$

The system protects against tracing the messages sent by a certain client

All clients sign messages for the server by the short group signature scheme proposed in Section 4.2.3. The variable group member's pseudonym values (K, T) are the part of every valid group signature. Nevertheless, the pseudonym is changed for every message by random values B, α. Due to this fact, the unlinkability of the message signatures from one client is provided. To get the first part of the group member secret key (x_i) and random value α, an attacker has to solve the Discrete Logarithm (DL) problem in $\mathbb{G}_1$. To get the second part of the group member secret key (A_i), the attacker has to solve the Decision Diffie-Hellman (DDH) problem in $\mathbb{G}_1$. Nevertheless, the pairing type-2 providing no efficient computable homomorphism from $\mathbb{G}_1$ to $\mathbb{G}_2$ is used. Therefore, the DDH problem remains hard. The proofs can be found in [43]. Further, it is assumed that other ways how to trace client's messages are mitigated by common privacy preserving technologies such as onion routing, aggregation, etc.

The system protects against extracting the identity of a certain client

In the join phase, a client sends his/her ID to the trusted registration authority. Moreover, the request message with the client's ID is encrypted. The client uses the ElGamal public key pk_{RA}. Only RA with his/her ElGamal private key sk_{RA} is able to decrypt this message. If an adversary tries to get the client identity, he/she has to decrypt the message. The adversary successes only in case that he/she has the valid ElGamal private key of RA or breaks the security of ElGamal which is regarded as secure [191].

The system protects against eavesdropping

C_i sends the message request ($enc_{pk_{RA}}(pk_{C_i}||ID_{C_i})$) in order to get the client's group secret member key and other group parameters from RA. The response message from RA to C_i is encrypted by the client's secret key sk_{C_i}. The request messages are encrypted by the ElGamal encryption scheme. Nowadays, the ElGamal scheme is regarded as secure [191]. The response messages is encrypted by a key known only to the client and RA. Therefore, the attacker is not able to decrypt these messages and gets data without the knowledge of the secret keys sk_{RA} and sk_{C_i}.

Further, it is assumed that the communication between the registration authority and the server is secured by TLS, which provides security properties like message authenticity, integrity and confidentiality. Due to this fact, opening a signed message and the revocation phase are secured. On the other hand, we assume that the messages sent from C_i to the server during the collection service do not contain private and confidential information that can be used to trace the certain clients. Due to this assumption, the data sent from C_i to the server are not encrypted.

The system protects against tampering with messages

The ECDSA signature scheme provides data integrity and authenticity in the join phase. The response messages from RA to clients are signed by the RA's ECDSA private key (sig_{RA}). ECDSA provides data authenticity, data integrity and non-repudiation. Assuming that ECDSA with SHA-2 is regarded as secure and is used in a strong way [196], then the ECDSA verification algorithm refuses the messages that are modified.

Clients, who send messages to a server, sign the messages by the short group signature proposed in Section 4.2.3. Then, the server checks the data authenticity and data integrity of these messages by the verification phase. If an attacker without a valid $gsk_{C_i} = (A_i, x_i)$ tampers signed message, then he/she has to recompute the hash c_j with an actual time stamp and a tampered message M, and compute new signature values $(s_{j\alpha}, s_{jx}, s_{j\delta})$. The computation of valid values for $s_{jx}, s_{j\delta}$ without knowing x_i is unfeasible if the Discrete Logarithm problem holds. The formal proof can be found in [43].

Furthermore, we assume that the connection between RA and S is secured against tampering with messages by using TLS.

The system protects against creating a forged but valid request, response or client messages

An attacker without the valid ECDSA private key sig_S is not able to create a valid request message that contains signed ID by a service provider. Only the service provider knows this key.

To create a valid response message that contains data signed by RA, the attacker has to know the valid ECDSA private key sig_{RA}. Only RA knows this key. Assuming that ECDSA is secured nowadays [196], the attacker is not able to create fake but valid response and request messages.

The proposed group signature scheme provides data authenticity, data integrity and message unlinkability in the signing and verification phases. Only the registration authority and the rightful group member C_i are able to create a valid group signature. If an attacker tries to forge a certain message then he/she must be able to compute some signature parts and a new hash c. Nevertheless, he/she does not know the valid $gsk_{C_i} = (A_i, x_i)$. If the hash function used is considered to be secure and the DL problem holds, recomputing the signature parts ($s_{j\alpha}$, s_{jx}, $s_{j\delta}$) without knowing x_i and A_i is considered unfeasible [43].

The system protects against replay attacks

In the communication between C$_i$ and S and in the registration of clients, all signed or encrypted messages include actual time stamps. In the registration phase, the actual time stamps are included in the both messages (request and response messages) to protect against replay attacks. An attacker who captures a request message is not able to use this message in the future because the message contains the time stamp and is encrypted by the pk_{RA}. The attacker is not able to change this time stamp because he/she is not able to decrypt the message without knowing sk_{RA}. The response message is protected similarly because it contains the time stamp which is encrypted and signed together with the rest of the data.

Further, the freshness of the time stamp included in a signed message is verified before the group signature verification. If the attacker without a valid $gsk_{C_i} = (A_i, x_i)$ wants use a captured signed message, then he/she has to recompute the hash c_j with an actual time stamp and recompute the signature values $(s_{j\alpha}, s_{jx}, s_{j\delta})$. The computation of valid values for $s_{jx}, s_{j\delta}$ without knowing x_i is unfeasible if the Discrete Logarithm problem holds [43].

The system protects against clients who misbehave

In the solution, honest clients stay anonymous and unlinkable. Other clients and a server (a service provider) are not able to link and trace honest clients. Nevertheless, the client's group member secret key can be easily revoked in case that the client loses his/her device or the device is controlled by an attacker. RA checks the conditions, and sends the revocation token to the server. Due to this step, the signed messages which are generated by this device are blocked on the server side.

If the client breaks the rules of the service provider, then the service provider sends his/her signed message to RA. Only RA is able to open the client's identity and can find the signed client's ID in the RA's database by the open phase. Then, RA can send to S the signed client ID and the revocation token of the malicious client. S can block the signed messages from the client by checking the revocation list where the revocation token of the client has been added. Based on the identity of the client, RA is able to deny the request message from the revoked client in the next join phase.

4.2.5 Performance Evaluation

This subsection presents the performance evaluation of the proposed system and compare it with related work. Further, the experimental implementation and results of the proposed solution are outlined.

Performance Evaluation of Group Signature Schemes

The group signature scheme is the main part in the proposed system and is used to secure messages sent from clients to a server. In Table 4.9, we evaluate and compare our modified group signature scheme with other group signature schemes regarding the signing and verification phases.

Tab. 4.9: Evaluation of Group Signature Schemes with VLR - Signing and Verification Phases.

GS scheme:	Proposed scheme in our system	MHM13 scheme [146]	BS04 scheme[44]	Chu *et al.* (Chu12) scheme[66]	NF07 scheme [153]
Batch	yes	yes	no	no	no
Revocation	Revocation List with Time Expiration	Revocation List with Time Expiration	Revocation List	Revocation List with Time Expiration	Revocation List for Intervals
Length of signature	$3\mathbb{G}_1, \mathbb{G}_T, 4\mathbb{Z}_p$ (2254 bits)	$2\mathbb{G}_1, \mathbb{G}_T, 4\mathbb{Z}_p$ (2059 bits)	$2\mathbb{G}_1, 5\mathbb{Z}_p$ (1192 bits)	$4\mathbb{G}_1, 5\mathbb{Z}_p$ (1549 bits)	$3\mathbb{G}_1, 6\mathbb{Z}_p$ (1533 bits)
Verification of n messages with r revoked users in RL:					
Pairings	2	2	$3n + 2nr$	$7n$	$2n + 2nr$
Exponentiation	$10n$+ $1nr * 8$	$10n$+ $1nr * 8$	$6n$	$13n + 1nr * 14$	$6n$
Multiplication	$9n$+1	$9n$+1	$6n$+$1nr$	$9n$	$6n + 1nr$
Signing:					
Pairings	0	2	2	5	1
Exponentiation	8	8	8	12	7
Multiplication	10	9	9	10	8

Table 4.9 depicts our comparison with related solutions with a verifier local revocation (the MHM13 scheme [146], the BS04 scheme[44], the Chu *et al.* scheme [66] and the NF07 scheme [153]). To be noted that the verification of n messages also includes the revocation check of r revoked users. Assuming that p is a 170-bit prime, elements in $\mathbb{G}_1$ have length 171 bits. It is recommended to use the date format for 255 months (21 years) formed in the time offset since the setup of the system. Then, the date format and the index k (defined in Section 4.3) take only 11 bits (8 bits for date, 3 bits for index k).

The bilinear pairing operation is generally considered as the most expensive operation in pairing based schemes, and exponentiation operations are more expensive than multiplication operations. The fast operations are omit such as addition, subtraction and hash functions in this performance evaluation.

The verification phase of the scheme in the system has only 2 pairing operations and is as efficient as the MHM13 scheme [146] and more efficient than the BS04 scheme[44], the Chu *et al.* scheme [66] and the NF07 scheme [153]. Nevertheless, the signing phase of the scheme in our solution has 0 pairing operations and is more efficient than the signing phases in the related schemes that have 1 or more pairing operations.

The join phase employs the ElGamal scheme and the ECDSA signature scheme. These schemes are more efficient than group signature schemes due to few modular operations (modular multiplicative inverses, multiplications, additions). However, the ECDSA scheme does not ensure user privacy. Due to this fact, the ECDSA scheme is proper for non-privacy communication, i.e., the join phase and on messages signed by RA and a service provider.

Implementation and Results on Restricted Devices

The proposed cryptographic protocol is implemented by using JAVA (JDK 7). The ECDSA scheme employs a 256-bit key size and uses the 256-bit SHA-2 hash function. The proposed group signature scheme is based on the Java Pairing Based Cryptography (jPBC) Library [1]. The implementation uses MNT curves with the embedding degree $k = 6$ and the 171-bit order because the MNT curves enable asymmetric pairing operations and are convenient for our group signature scheme based on the BBS scheme [43]. The join phase uses the 1024-bit ElGamal encryption.

The implemented cryptographic protocol is tested on restricted devices such as a microcomputer device and mobile devices, and on a personal computer (PC). The PC machine (CPU: Intel(R) Xeon(R) X3440 2.53GHz, RAM: 4 GB) simulates RA and servers. Nevertheless, it is assumed that these parties have more powerful devices in practice. The measurement is performed in 100 iterations and outlines the average values. On the personal computer, the ECDSA verification of 1 signed messages takes about 2 ms and signing the message by ECDSA takes about 3 ms (for 500 B messages). The ElGamal encryption takes about 4 ms for 0.5 kB messages and decryption takes about 2 ms for same-sized messages.

Nevertheless, the most expensive phases of the proposed protocol are the group signature signing phase and the verification phase. The verification phase of the proposed protocol runs on the servers and can be measured on the PC machine used. The verification of 1 message takes about 100 ms. Figure 4.9 shows the time of the verification phase with increasing the messages on the PC machine. This time is measured up to 100 messages which can be a realistic number of messages in the middle-sized communication systems. In case of large-sized systems, it is assumed to employ more servers to perform the verification phase.

[1] (available on http://gas.dia.unisa.it/projects/jpbc/index.html)

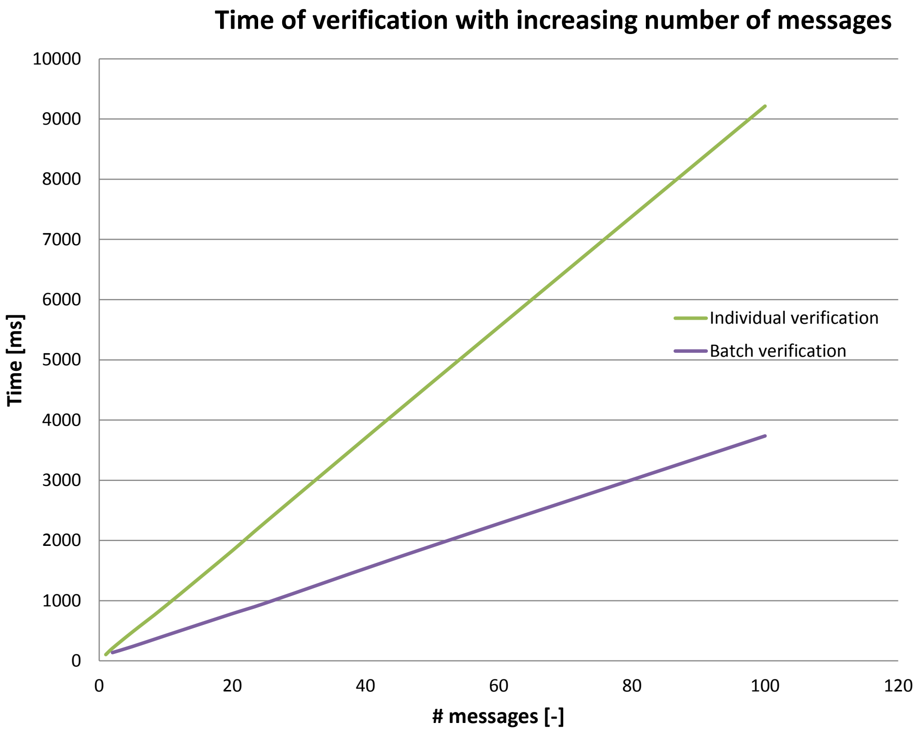


Fig. 4.9: The time of verification with increasing number of messages on PC.

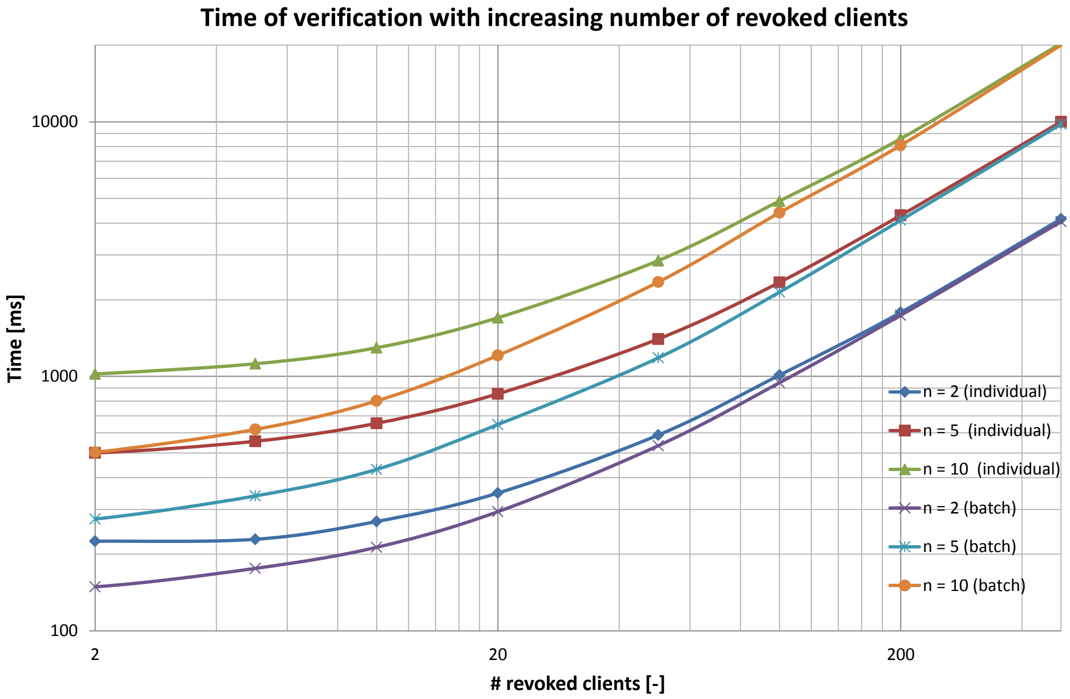


Fig. 4.10: The time of verification with increasing number of revoked clients on PC.

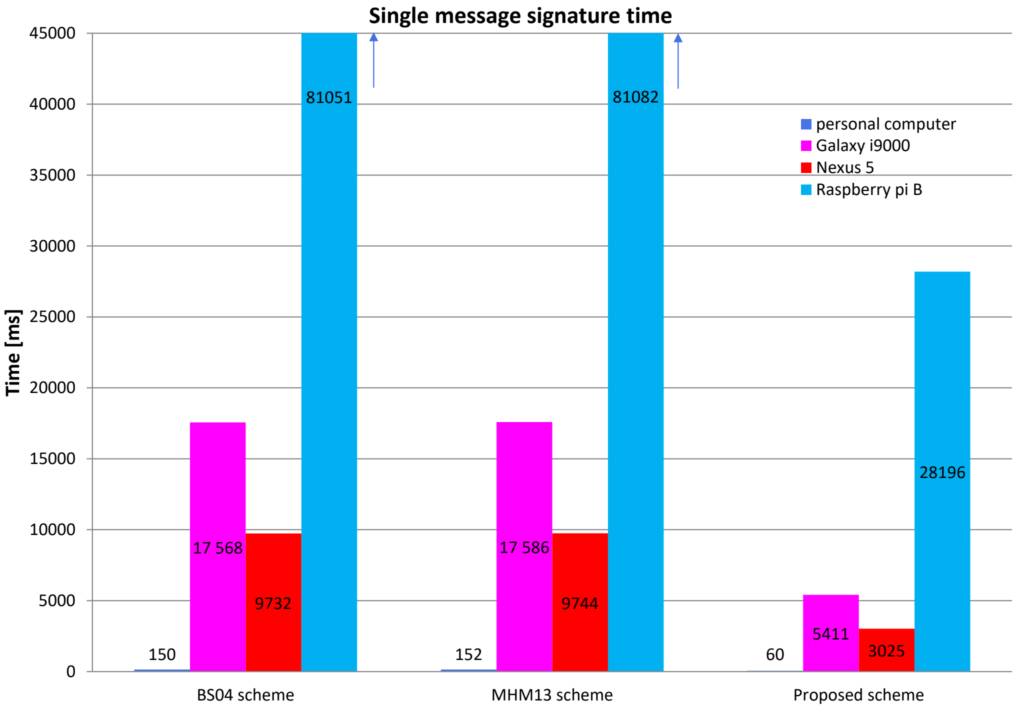


Fig. 4.11: The time of signing on various devices.

In Figure 4.9, the batch verification and the individual verification are compared. The batch verification is more efficient than the individual verification for a large number of messages. Figure 4.10 shows the time of verification of n messages with increasing the number of revoked clients on the PC machine. If RL contains the 100 revoked clients, then the verification phase takes about 1 s for 2 messages. Nevertheless, it is assumed that servers are more powerful than the PC machine used in this measurement.

The signing phase on the PC machine takes only about 60 ms. Furthermore, tthe signing phase of the proposed protocol is tested on various low-performance devices, i.e., a microcomputer device (raspberry Pi Model B, CPU: 700 MHz, RAM: 512 MB) and two mobile devices (LG Nexus 5, CPU: ARMv7 Quad-core 2.3 GHz, RAM: 2 GB; Samsung i9000, CPU: 1 GHz, RAM: 512 MB).

Figure 4.11 shows the time of the signing phase measured on these devices. In this figure, the proposed scheme, the MHM13 scheme [146] and the BS04 scheme [44] are compared. The signing phases of the group signature schemes are quite time-consuming on some restricted devices but these schemes can be employed in some delay-tolerant applications such as data collection services. According to the results, the signing phase of the proposed scheme is approximately three times more efficient than the MHM13 scheme [146] and the BS04 scheme [44].

4.2.6 Summary

The proposed system is designed to ensure the anonymity of honest clients of the data gathering/collection services. The group signature scheme used in the system ensures data authenticity, integrity, unlinkability and anonymity. The proposed scheme provides a signing phase without pairing operations, thus, the solution can run on computationally restricted client nodes that sign messages in delay-tolerant services. The proposed verification phase employs the batch verification technique and the efficient revocation check phase to achieve sufficient performance on the servers that maintain services with a large number of users. Moreover, our system employs the revocation list that is reduced by the expiration of group revocation tokens. Hence, the revocation list is not increasing infinitely but remains short. To prove practical usability of our system, the benchmarks on real hardware are presented.

4.3 Decentralized Privacy-Preserving Transactions Based on Lightweight Ring Signatures

This section presents the novel system based on a lightweight privacy-preserving ring signature scheme that is suitable for anonymous transactions and e-voting services which run in an environment with constrained devices such as handheld devices and IoT nodes. The proposed system provides the fast verification of signatures without using heavy operations such as pairings and exponentiation. Further, signature linkability and uniqueness properties are added in order to provide double-spending protection.

The amended version of the text below is the part of the the Secrypt 2018 conference proceedings [13].

4.3.1 Introduction, State of the Art and Contribution

Modern digital services such as e-voting or electronic payment transactions including various cryptocurrencies, smart contracts and e-coins try to employ privacy protection and security properties for their users. These properties can be achieved by using many technologies such as privacy-enhancing cryptographic constructions as zero knowledge protocols, ring signatures, blind signatures or by various mixing network protocols. In current e-voting solutions, voter's anonymity is the main requirement as in classic votes. E-voting systems should not be used without this property. Similarly the users of e-transactions also require privacy protection. The privacy-preserving transactions can attract more users who concern about their privacy. Nevertheless, these privacy-preserving services must handle security risks that could be caused by anonymity. Hence, the solutions should be resistant to potential misusing, e.g., double-spending, double-voting, tracing of voters/transaction senders and more.

There are several advanced cryptographic constructions that can be deployed in order to provide anonymous and secure transactions or votes. Group signatures (GS) allow any group member to anonymously sign a message on behalf of the group. Only group managers/issuers are able to add users and trace or revoke users. Nevertheless, GS schemes are often centralized and the group manager has to be a trusted party. The environment of transactions is mostly decentralized. Therefore, ring signatures (RS) that are similar to group signatures could be interesting constructions for these decentralized digital services.

RS provide a perfect privacy (untracebility) and signer is not able to prove his/her signature (non-repudation). In order to employ these constructions also in transactions, double-spending protection must be solved and provided.

State of the Art

Since the paper [167] published in 2001, ring signatures and their implementation in e-voting, anonymous data sharing, e-cash services and other privacy-preserving services have been studied in many works, e.g., [135], [190], [208], [57], [177], [86], [133], [134], [213], [157], [187]. Ring signatures provide various properties (e.g., linkability, deniability, exculpability, disavowal) and security assumptions. For example, Wu *et al.* [208] present ad hoc group signatures that combine some properties of group signatures and ring signatures. These schemes provide the privacy protection for self-organized groups. The ad hoc group signature scheme removes the trusted third party such as a group manager from a system and adds the self-traceability property to ring signatures. In a decentralized system, signers can produce constant-sized anonymous signatures on behalf of the group (a variable set of members). Furthermore, the non-interactive deniable ring signature scheme [217] provides the confirmation of signing (e.g., a lottery game winner) and signature disavowal for non-signers in the ring in order to a signer detection. Nevertheless, both advanced ring signature schemes do not offer double-spending protection.

Liu *et al.* [135] propose a linkable, spontaneous and anonymous group (LSAG) signature scheme. The scheme provides the culpability property that allows an investigator to conduct that the authorship of the signature belongs to the user. This scheme also provides the linkability of two signatures. Tsang and Wei [190] extend the short ring signature scheme construction of Dodis *et al.* [80] and discuss the application of their scheme to E-voting, offline anonymous electronic cash and direct anonymous attestation. Dodis *et al.* offer a constant-sized ring signature scheme secured in Random Oracle model. Both constructions are based on a three-move zero-knowledge proof-of-knowledge system using the Fiat-Shamir transformation. Fujisaku and Suzuki [86] propose traceable ring signatures that use tags. The tag consists of a list of ring members and the issue of the event. The signer can sign only once per the event in order to stay anonymous in the system. Van Saberhagen [195] proposes CryptoNote transactions that are based on a ring signature. Each user of CryptoNote uses a set of public keys and private keys. CryptoNote combines a Diffie-Hellman exchange, one-time signatures and the modification of ring signatures [86]. These ring signatures have size $n+1$, where n is the size of the sender anonymity. A verifier also checks if transactions have been already spent or not by a Link procedure. Noether *et al.* [157] propose Ring Confidential Transactions (Ring CT) that enhance the original CryptoNote protocol. They propose a Multilayered Linkable Spontaneous Anonymous Group signature (MLSAG) scheme that provides a signature on a set of n key vectors. Nevertheless, many ring signature schemes have several heavy computations (e..g. pairings, exponentiation, point multiplication) and sizable signatures that depends on the number of ring members.

The most related paper Yang *et al.* [213] present the ring signatures based on the Rabin cryptosystem. In their paper, the comparison with existing ring signatures shows that the ring signature scheme is very efficient in sign and verify phases and does not need any pairings. Nevertheless, the Rabin signature in the signing phase usually take similar time like exponentiation in the RSA decryption. Moreover, the scheme defines only two properties: unconditionally signer ambiguity and existentially unforgeability and does not solve double-spending by linkability and signature uniqueness. The following proposal aims to provide efficient and privacy-preserving ring signature solution that supports signature uniqueness and protect against double-spending which is important in e-voting or anonymous transactions.

Contribution

The proposed lightweight privacy-preserving signature solution can be suitable for anonymous transactions or e-voting services in a constrained environment such as IoT. The efficient Yang's ring signature scheme [213] is modified by the employing key image tags. Thus, the proposed solution adds a signature uniqueness property that provides double-spending protection of each transaction or vote. Furthermore, a public key shuffling property is added in order to increase user anonymity during signing messages (e.g., transactions or votes). In the origin description of the Yang's ring signature scheme [213], the signer's public key is the last key in the list of public keys. Therefore, the actual signer can be tracked by his/her public key. In our proposed solution, the verifiers or observers are not able to recognize the actual signer public key that could be any from the list. Moreover, several steps describing how to employ the solution in anonymous transactions and e-voting scenarios are introduced.

4.3.2 Background

In this subsection, the cryptography background and security properties are outlined.

Cryptography Used

The ring signature scheme [213] based on the Rabin cryptosystem [164] is modified and is used as the basic part in this proposal. The description of the Rabin cryptosystem is provided in Subsection 2.2.5.

Security Properties

The proposed solution provides these security properties: Correctness, Signer Anonymity, Signature Uniqueness, Signature Unforgeability and Signature Linkability.

- **Correctness** - a valid signature is always accepted (completeness) and an invalid signature is always rejected (soundness).
- **Signer Anonymity** - a signature is produced by one member from the set of public key holders. Therefore, the identity of a signer is hidden in the group and no one can determine the actual signer from the signature.
- **Signature Uniqueness** - a valid signature on the message could be created only once by a honest signer. The second signature from the same signer during one event (transaction, e-voting) is linked by a key image and is rejected.
- **Signature Unforgeability** - a produced signature is unforgeable. An attacker with negligible probability can produce a valid signature without the corresponding private key.
- **Signature Linkability** - two valid signatures on the same message m with one private/public keypair can be linked by the key image. This property implies the double-spending/voting protection.

4.3.3 Proposed Solution

This subsection describes the proposal for secure and privacy-preserving transactions or voting based on ring signatures. The proposal consists of 3 parties: a signer (a sender, a voter), a verifier (a receiver of the transaction or polling manager/bulletin board application) and an investigator (a trusted third party which detects dishonest signers). Our solution consists of these phases: Key Generation, Signature Generation, Signature Validation and Link Procedure.

Key Generation

In this phase, key pairs are generated. For $i = 1, ..., n$, where n is the number of ring users, each i-th user selects two safe primes p_i, q_i such that $p_i = 2p_i' + 1, q_i = 2q_i' + 1$ where p_i', q_i' are primes. The i-th user securely stores a private key that is p_i, q_i and computes a public key as $N_i = p_i q_i$. The public key is then sent to an ad hoc group of n users. The public parameters are a set of public keys $L = (N_1, ..., N_n)$, a defined hash functions $H_i : \{0,1\}^* \rightarrow \mathbb{Z}_{N_i}$ for $i = 1, ..., n$ and a hash function $H : \{0,1\}^* \rightarrow QR(N_i)$ used for key images, where $QR(N_i) = \{x \in \mathbb{Z}_{N_i}\text{s.t. } x = y^2$ for some $y \in \mathbb{Z}_{N_i}\}$.

Signature Generation

It is assumed that a signer (e.g., a transaction sender/ a voter) **S** signs the message m (e.g., transaction amount with a metadata, ballot in e-voting) by the ring signature scheme.

The proposal modifies Yang *et al.* ring signature scheme [213] that is based on the Rabin scheme. Yang *et al.*'s ring signature scheme [213] defines only two properties: unconditionally signer ambiguity and existentially unforgeability. That scheme is modified and enhanced by the unique tag in order to achieve a double-spending protection. Moreover, the proposal shuffles actual user public key in the list, then, a verifier (an observer) is not able to determine which the public key has been used.

Let $L = (N_1, ..., N_n)$ is a list of n ring users' public keys, the signer j uses his/her private key (p_j, q_j) to produce a signature of the message m as (L, m, σ). The j-th signer $(\mathbf{S}_j)$ also computes a key image

$$I = H(p_j||N_j||ID_{event})^{1/2} \mod N_j \tag{4.19}$$

by the knowledge of the factorization of N_j and by applying the Chinese remainder theorem. In order to enable the signer reuses the keypair in more events (e.g., more transactions or e-votes), the signer maps also an event identifier ID_{event} (i.e., a transaction number or an e-voting event). The key image commits signer's public and private keys and prevents the reuse the same keys during one event.

The signer knows his/her private key (p_j, q_j) and public key N_j and executes following steps:

1. $\mathbf{S}_j$ chooses a random element $r_j \in \mathbb{Z}_{N_j}$ and computes:

$$h = H_1(L||m||ID_{event}), c_{j+1} = H_{j+1}(h||r_j). \tag{4.20}$$

2. For $i = 1$,...,n and $i \neq j$, $\mathbf{S}_j$ randomly generates element $x_i \in \mathbb{Z}_{N_i}$, i.e., for all other ring members.
3. $\mathbf{S}_j$ successively computes in j modulo n, i.e., for each i started from $j+1$, $j+2 \ldots 0 \ldots j-1$:

$$c_{i+1} = H_{i+1}(h||c_i I + x_i^2 \mod N_i). \tag{4.21}$$

4. If $r_j - c_j I \mod N_j \in QR(N_j)$ then $\mathbf{S}_j$ assigns

$$t_j = r_j - c_j I \mod N_j, \tag{4.22}$$

otherwise $\mathbf{S}_j$ chooses another element $x_{j-1} \in \mathbb{Z}_{N_{j-1}}$ and computes new c_j from step 3 by Equation 4.21 until $r_j - c_j I$ is a quadratic residue.

5. $\mathbf{S}_j$ solves

$$x_j = t_j^{1/2} \mod N_j, \tag{4.23}$$

by the knowledge of the factorization of N_j with using the Chinese remainder theorem. Square roots could be computed by the Tonelli - Shanks algorithm or by its modifications.

Finally, the signer produces the signature σ =$(I, c_1, x_1, ..., x_n)$ on the message m in the event ID_{event}.

The computational and memory complexity could be reduced if the signer chooses smaller subset of k users' public keys from n ring members. Nevertheless, the level of signer privacy is reduced as well.

Signature Validation

A verifier (a transaction receiver or a polling manager/bulletin board service) $\mathbf{V}$ checks the signature on the message by checking the ring signature σ on the message m and by checking its uniqueness in the event ID_{event}. The verifier uses public parameters (L, H) and checks the received ring signature σ =$(I, c_1, x_1, ..., x_n)$ on the message m during the event ID_{event}.

1. $\mathbf{V}$ computes

$$h = H_1(L||m||ID_{event}). \tag{4.24}$$

2. For each i = 1,...,n, $\mathbf{V}$ restores

$$r_i = c_i I + x_i^2 \mod N_i. \tag{4.25}$$

3. For each i = 1,...,$n-1$, $\mathbf{V}$ computes

$$c_{i+1} = H_{i+1}(h||r_i). \tag{4.26}$$

4. If $c_1 = H_1(h||r_n)$, the output is true and the signature is accepted and $\mathbf{V}$ continues by checking the signature uniqueness. Otherwise, $\mathbf{V}$ rejects the signature and the algorithm halts.

Further, the verifier checks the uniqueness of the signature. $\mathbf{V}$ checks if the image key I of the signature has not been used in past signatures in the event ID_{event}. In case that the key image I is not presented in a dataset of key images, the verifier accepts the signature. Then, the key image of the signature is added to the dataset of key images in order to prevent double spending in the future. Otherwise, the signature of the message (e.g., a transaction/vote) is marked as the duplicated and it is rejected.

Link Procedure

In case that the n+1 or more ring signatures occur at the end of an event (e.g., transaction bulk, closing e-voting) with n participants, an investigator (i.e., a third trusted party) runs this procedure in order to detect among the members of the ring such a malicious signer who produces more valid signatures. The investigator precomputes all I^2. Further, each honest signer, which knows such private key p_j, securely sends to the investigator a set of $(H(r_1||N_1||ID_{event})$ mod $N_1, \dots H(p_j||N_j||ID_{event})$ mod $N_j, \dots H(r_n||N_n||ID_{event})$ mod $N_n)$ in randomized order. The investigator then checks that at least one received $H(p_i||N_i)$ mod $N_i = I^2$ for $i = 0 \dots n$. The mixed set of hash hides the index of the signer so the signer is still anonymous against external parties.

4.3.4 Security Analysis

This subsection provides the security analysis of the proposed solution. These security properties are discussed: correctness, signer anonymity, signature uniqueness, signature unforgeability, signature linkability.

Theorem 1. *Correctness* - Completeness and soundness are provided. A honest verifier is always able to accept a valid ring signature and reject a false signature.
Proof. Suppose that a verifier has correct public parameters such as set of public keys $L = (N_1, ..., N_n)$ and a set of defined hash functions. He/she can check a signature σ =$(I, c_1, x_1, ..., x_n)$ on a message m by restoring parameters r_i and c_i for each i from 1 to n and finally by checking $c_1 = H_1(h||r_n)$. Assume that $r_n = c_nI + x_n^2 \text{mod} N_n$ and somewhere in the ring $c_{j+1} = H_{j+1}(h||r_j) = H_{j+1}(h||c_jI + x_j^2 \text{mod } N_j\ )$ where $x_j^2 = t_j$ mod $N_j = r_j - c_jI$ mod N_j so that $c_{j+1} = H_{j+1}(h||c_jI + r_j - c_jI \text{mod } N_j = H_{j+1}(h||r_j)$.

Theorem 2. *Signer Anonymity* - It is infeasible to identify which private key creates the ring signature.
Proof. A verifier uses a set L of n public keys and is not able to identify which public key belongs to a signer. The chance of guessing correctly which public key is used to generate a given signature is negligibly greater than $1/n$. It is assumed that the private key is chosen at random and an adversary only knows the public keys and not the other private keys. If the adversary knows k private keys then the guessing of signer key is negligibly greater than $1/(n-k)$.

Further, the key image I does not leak the signer identity if the private keys are chosen at random. The user anonymity holds also in the link procedure for external observers due the signers who prove their honesty by sending only basic hash of values in randomized order.

Theorem 3. *Signature Uniqueness* - a signer is able to produce only one valid signature on the message by the one public/private keypair.

Proof. The key image of j-th user (computed by Equation 4.19) maps public and private keys and is integrated in the produced signature. A verifier restores $r_i = c_i I + x_i^2 \text{mod} N_i$ for each i from 1 to n where I is a part in all r_i. In fact, if a malicious user tries to re-use more times the same signature, the verifier can detect the re-use by checking the dataset of key images.

Theorem 4. *Signature Unforgeability* - it is hard to produce a valid signature without a private key.

Proof. A signer without a private key p_j, q_j is not able to solve $x_j = t_j^{1/2} \text{mod} N_j$ by using the knowledge of the factorization of N_j with factors p_j, q_j. If an adversary is successful in forgery, he/she must output x_j that satisfies such $c_{j+1} = H_{j+1}(h||c_j I + x_j^2 \text{mod } N_j)$ which causes that $c_1 = H_1(h||r_n) = H_1(h||c_n I + x_n^2 \text{mod} N_n)$ and encloses the ring. More formal analysis for the property can be found in [213].

Theorem 5. *Signature Linkability* - it is hard to produce n+1 valid signatures on the message by the n public/private keypair.

Proof. The key image computed by Equation 4.19 maps public and private keys of a honest signer, ID_{event} and is integrated in the produced signature. ID_{event} is also used in the ring signature. Any observer (a verifier) can link two signatures on the same message during one event by I from one honest signer. Hence, a honest signer cannot re-use more times the valid signatures of one private/public keypair and a correct H function. In case that a malicious signer will try to produce a new signature with a different key image but with same keypair, then the Link procedure detects this signature and rejects it. All signatures with incorrect key images can be detected.

4.3.5 Performance Evaluation

This subsection discusses the computational complexity of the proposed solution and compares signature sizes and the complexity of most significant phases such as signing and verification with other related works that are based on ring signatures and provide linkability. Table 4.10 provides the comparison of performance and memory costs of the proposed ring signature scheme and related schemes. The notification of operations is: a pairing operation as P, exponentiation as E, multiplication as M, squaring as S. The relatively fast operations such as addition and a hash function are omitted. N denotes the number of users in a ring/ad hoc group. In order to evaluate the length of signatures, the following notation is used, e.g., $O(1)$ - constant size, $O(\sqrt{N})$ - semi-linear size, $O(N)$ - linear size.

In the proposed solution and Yang *et al.* scheme [213], the signing procedure

Tab. 4.10: Performance Comparison of Related Schemes.

Scheme	Sign	Verify	Signature size
Liu *et al.* [135]	(3+4(N − 1))E + (1+2N)M	(4N)E + 2NM	$O(N)$, $N+2$
Fujisaki and Suzuki *et al.* [86]	(3+2N)E + (2+3N)M	(4N)E + (3N)M	$O(N)$, $2N+1$
Chandran *et al.* [57]	(5+6$\sqrt{N}$+(N+1)/3)E+ (6$\sqrt{N}$+8)M	(6+6$\sqrt{N}$)P + (3$\sqrt{N}$+1)E + (4$\sqrt{N}$ +1)M	$O(\sqrt{N})$, $6\sqrt{N}+6$
Liu *et al.* [133]	NE	N E	$O(N)$, $2N+1$
Liu *et al.* [134]	(5+N)E+(4+N)M	(4+N)E+(3+N)M	$O(N)$, $N+3$
Yang *et al.* [213]	E+NS	NS	$O(N)$, $N+1$
This solution	2E+2M+NS	M+NS	$O(N)$, $N+2$

employs the Rabin signing that computes the square root of the parameter in modular arithmetic. It is considered that this operation as expensive as 1 exponentiation, therefore it is noted as E also in our comparison. Yang *et al.* [213] is the most efficient scheme from the compared schemes but does not support linkability. Then, the proposed solution, which provides signature uniqueness and linkability by adding a key image, is very efficient during signing and verification in comparing with other related schemes.

4.3.6 Summary

The proposed lightweight privacy-preserving solution based on ring signatures provides anonymity, uniqueness, linkability and unforgeability, and can be applied in applications that require double-spending and double-voting protection. The solution does not use heavy operations. The ring signature verification takes only 1 multiplication and N squaring which depends on the size of ring (N). Therefore, the solution could be implemented in services running in heterogeneous networks with small and medium groups of constrained devices.

5 Conclusion

This habilitation thesis provided the overview about modern cryptographic constructions used in asymmetric cryptographic protocols and digital signature schemes. The thesis focused on conventional and advanced digital signature schemes such as ring and group signatures. Further, the thesis investigated the deployment of these schemes on constrained and small devices. Finally, the thesis presented three author's proposals that are designed for different applications, i.e., an authentication method for a secure access control system, a group signature scheme for secure and privacy-preserving data transfer system, and a ring signature scheme for anonymous transactions. All systems and schemes are designed in order to be suitable for constrained or small devices. The expected contribution is threefold pedagogical, i.e., to produce a unified overview about conventional and advanced digital signature schemes that can serve as study resource, practical, i.e., to present the comprehensive assessment of cryptographic protocols deployed on constrained and small devices, and scientific, i.e., to produce novel protocols with advanced features that are suitable for constrained devices. The pedagogical contribution is addressed in Chapter 2 that presents the theoretical background with the description of underlying cryptographic methods, digital signature schemes and advanced digital signature schemes such as group and ring signatures. The chapter presents the examples of signature schemes and their theoretical evaluation. Moreover, the chapter also introduces other privacy-preserving cryptographic protocols and perspective post-quantum cryptographic protocols. The practical contribution is addressed in Chapter 3 that evaluates the performance of chosen cryptographic primitives and schemes on various constrained devices. The chapter discusses the feasibility of common, privacy-preserving and post-quantum cryptography on constrained and small devices. These practical results and lessons learned can help cryptographers and security experts with designing and deploying security solutions in an environment with constrained devices. The scientific contribution is addressed in Chapter 4 that contains three novel proposals based on advanced cryptographic constructions that can be deployed on constrained and small devices. The proposed protocols are presented in three sections. Section 4.1 presents the secure and efficient two-factor zero-knowledge authentication system based on smart cards. Section 4.2 presents the secure privacy-preserving data transfer system based on light-weight group signatures with time-bound membership. This solution is suitable for small devices because signing phase does not need pairing operations. Section 4.3 presents the solution for decentralized privacy-preserving transactions based on lightweight ring signatures. The mentioned results and proposals have been published in journals with impact factors and international conferences dedicated to security.

Author's selected publications since 2015

[1] Dzurenda, Petr, Hajny, Jan, Malina, Lukas, and Ricci, Sara. *Anonymous Credentials with Practical Revocation using Elliptic Curves.* In In Proceedings of the 14th International Joint Conference on e- Business and Telecommunications (ICETE 2017) - Volume 4: SECRYPT, pp. 534–539. 2017.

[2] Dzurenda, Petr, Ricci, Sara, Hajny, Jan, and Malina, Lukas. *Performance Analysis and Comparison of Different Elliptic Curves on Smart Cards.* In International Conference on Privacy, Security and Trust (PST). 2017.

[3] Fujdiak, Radek, Blazek, Petr, Mikhaylov, Konstantin, Malina, Lukas, Mlynek, Petr, Misurec, Jiri, and Blazek, Vojtech. *On Track of Sigfox Confidentiality with End-to-End Encryption.* In Proceedings of the 13th International Conference on Availability, Reliability and Security, ARES 2018, pp. 19:1–19:6. New York, NY, USA: ACM, 2018. ISBN 978-1-4503-6448-5. doi:10.1145/3230833.3232805.

[4] Hajny, Jan, Dzurenda, Petr, and Malina, Lukas. *Attribute-based credentials with cryptographic collusion prevention.* Security and Communication Networks, 8(18):3836–3846, 2015.

[5] Hajny, Jan, Dzurenda, Petr, and Malina, Lukas. *Multi-Device Authentication using Wearables and IoT.* In Proceedings of the 13th International Joint Conference on e-Business and Telecommunications (ICETE 2016) - Volume 4: SECRYPT, pp. 483–488. 2016.

[6] Hajny, Jan, Dzurenda, Petr, and Malina, Lukas. *Multidevice Authentication with Strong Privacy Protection.* Wireless Communications and Mobile Computing, 2018, 2018.

[7] Hajny, Jan, Dzurenda, Petr, Ricci, Sara, Malina, Lukas, and Vrba, Kamil. *Performance Analysis of Pairing-Based Elliptic Curve Cryptography on Constrained Devices.* In Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT), 2018 10th International Congress on, pp. 300–305. IEEE, 2018.

[8] Malina, L., Dzurenda, P., Hajny, J., and Martinasek, Z. *Assessment of Cryptography Support and Security on Programmable Smart Cards.* In 2018 41st International Conference on Telecommunications and Signal Processing (TSP), pp. 1–5. 2018. doi:10.1109/TSP.2018.8441334.

[9] Malina, L, Hajny, J, Dzurenda, P, and Zeman, V. *Privacy-preserving security solution for cloud services.* Journal of Applied Research and Technology, 13(1):20–31, 2015.

[10] Malina, Lukas, Benes, Vlastimil, Hajny, Jan, and Dzurenda, Petr. *Efficient and secure access control system based on programmable smart cards.* In Telecommunications and Signal Processing (TSP), 2017 40th International Conference on, pp. 32–36. IEEE, 2017.

[11] Malina, Lukas, Dzurenda, Petr, and Hajny, Jan. *Evaluation of anonymous digital signatures for privacy-enhancing mobile applications.* International Journal of Security and Networks, 13(1):27–41, 2018.

[12] Malina, Lukas, Dzurenda, Petr, Hajny, Jan, and Martinasek, Zdenek. *Secure and efficient two-factor zero-knowledge authentication solution for access control systems.* Computers & Security, 77:500–513, 2018.

[13] Malina, Lukas, Hajny, Jan, Dzurenda, Petr, and Ricci, Sara. *Lightweight Ring Signatures for Decentralized Privacy-preserving Transactions.* In Proceedings of the 15th International Joint Conference on e-Business and Telecommunications (ICETE 2018) - Volume 2: SECRYPT, pp. 526–531. 2018.

[14] Malina, Lukas, Hajny, Jan, Fujdiak, Radek, and Hosek, Jiri. *On perspective of security and privacy-preserving solutions in the internet of things.* Computer Networks, 102:83–95, 2016.

[15] Malina, Lukas, Hajny, Jan, and Martinasek, Zdenek. *Privacy-preserving authentication systems using smart devices.* In Telecommunications and Signal Processing (TSP), 2016 39th International Conference on, pp. 11–14. IEEE, 2016.

[16] Malina, Lukas, Hajny, Jan, Mlynek, Petr, Machacek, Jiri, and Svoboda, Radomir. *Towards Efficient Application of Cryptographic Schemes on Constrained Microcontrollers.* Journal of Circuits, Systems and Computers, 25(10):1650129, 2016.

[17] Malina, Lukas, Hajny, Jan, and Zeman, Vaclav. *Light-weight group signatures with time-bound membership.* Security and Communication Networks, 9(7):599–612, 2016.

[18] Malina, Lukas, Hajny, Jan, Zeman, Vaclav, and Vrba, Kamil. *Security and privacy in the smart grid services.* In Telecommunications and Signal Processing (TSP), 2015 38th International Conference on, pp. 71–75. IEEE, 2015.

[19] Malina, Lukas, Horvath, Tomas, Munster, Petr, and Hajny, Jan. *Security solution with signal propagation measurement for Gigabit Passive Optical Networks.* Optik-International Journal for Light and Electron Optics, 127(16):6715–6725, 2016.

[20] Malina, Lukas, Konecny, Jakub, Dzurenda, Petr, and Hajny, Jan. *On practical deployment of smart card based authenticated key agreement schemes.* In Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT), 2017 9th International Congress on, pp. 277–282. IEEE, 2017.

[21] Malina, Lukas, Munster, Petr, Hajny, Jan, and Horvath, Tomas. *Towards secure gigabit passive optical networks: signal propagation based key establishment.* In 12th International Joint Conference on e-Business and Telecommunications (ICETE) - SECRYPT, vol. 4, pp. 349–354. IEEE, 2015.

[22] Malina, Lukas, Popelova, Lucie, Dzurenda, Petr, Hajny, Jan, and Martinasek, Zdenek. *On Feasibility of Post-Quantum Cryptography on Small Devices.* IFAC-PapersOnLine, 51(6):462–467, 2018.

[23] Malina, Lukas, Vives-Guasch, Arnau, Castellà-Roca, Jordi, Viejo, Alexandre, and Hajny, Jan. *Efficient group signatures for privacy-preserving vehicular networks.* Telecommunication Systems, 58(4):293–311, 2015.

[24] Martinasek, Zdenek, Hajny, Jan, Smekal, David, Malina, Lukas, Matousek, Denis, Kekely, Michal, and Mentens, Nele. *200 Gbps Hardware Accelerated Encryption System for FPGA Network Cards.* In Proceedings of the 2018 Workshop on Attacks and Solutions in Hardware Security, pp. 11–17. ACM, 2018.

[25] Martinasek, Zdenek, Iglesias, Felix, Malina, Lukas, and Martinasek, Josef. *Crucial pitfall of DPA Contest V4. 2 implementation.* Security and Communication Networks, 9(18):6094–6110, 2016.

[26] Martinasek, Zdenek, Zeman, Vaclav, Malina, Lukas, and Martinasek, Josef. *k-Nearest neighbors algorithm in profiling power analysis attack.* Radioengineering, 25(2):365–382, 2016.

[27] Ometov, Aleksandr, Masek, Pavel, Malina, Lukas, Florea, Roman, Hosek, Jiri, Andreev, Sergey, Hajny, Jan, Niutanen, Jussi, and Koucheryavy, Yevgeni. *Feasibility characterization of cryptographic primitives for constrained (wearable) IoT devices.* In PerCom Workshops, pp. 1–6. 2016.

Other publications

[28] *BSI, Technical Guideline TR-03110: Advanced Security Mechanisms for Machine Readable Travel Documents Extended Access Control (EAC) and Password Authenticated Connection Establishment (PACE), Version 2.0.* 2008.

[29] *ISO/IEC 20008-2: Information technology - Security techniques - Anonymous digital signatures - Part 2: Mechanisms using a group public key. Stage 60.60*, 2013.

[30] Alcaide, Almudena, Palomar, Esther, Montero-Castillo, José, and Ribagorda, Arturo. *Anonymous authentication for privacy-preserving IoT target-driven applications.* Computers & Security, 37:111–123, 2013.

[31] Alcaraz, Cristina and Zeadally, Sherali. *Critical infrastructure protection: requirements and challenges for the 21st century.* International journal of critical infrastructure protection, 8:53–66, 2015.

[32] Alkim, Erdem, Ducas, Léo, Pöppelmann, Thomas, and Schwabe, Peter. *Post-quantum Key Exchange-A New Hope.* In USENIX Security Symposium, vol. 2016. 2016.

[33] Ateniese, Giuseppe, Camenisch, Jan, Joye, Marc, and Tsudik, Gene. *A practical and provably secure coalition-resistant group signature scheme.* In Advances in CryptologyCRYPTO 2000, pp. 255–270. Springer, 2000.

[34] Ateniese, Giuseppe, Song, Dawn, and Tsudik, Gene. *Quasi-efficient revocation of group signatures.* In Financial Cryptography, pp. 183–197. Springer, 2003.

[35] Atzori, Luigi, Iera, Antonio, and Morabito, Giacomo. *The internet of things: A survey.* Computer networks, 54(15):2787–2805, 2010.

[36] Barnaghi, Payam, Wang, Wei, Henson, Cory, and Taylor, Kerry. *Semantics for the Internet of Things: early progress and back to the future.* International Journal on Semantic Web and Information Systems (IJSWIS), 8(1):1–21, 2012.

[37] Bellare, Mihir, Micciancio, Daniele, and Warinschi, Bogdan. *Foundations of group signatures: Formal definitions, simplified requirements, and a construction based on general assumptions.* In Advances in CryptologyEurocrypt 2003, pp. 614–629. Springer, 2003.

[38] Benaloh, Josh. *Dense probabilistic encryption.* In Proceedings of the workshop on selected areas of cryptography, pp. 120–128. 1994.

[39] Bernstein, Daniel J. *Introduction to post-quantum cryptography.* In Post-quantum cryptography, pp. 1–14. Springer, 2009.

[40] Bernstein, Daniel J, Chou, Tung, and Schwabe, Peter. *McBits: fast constant-time code-based cryptography.* In International Workshop on Cryptographic Hardware and Embedded Systems, pp. 250–272. Springer, 2013.

[41] Bernstein, Daniel J, Duif, Niels, Lange, Tanja, Schwabe, Peter, and Yang, Bo-Yin. *High-speed high-security signatures.* Journal of Cryptographic Engineering, 2(2):77–89, 2012.

[42] Bernstein, Daniel J, Josefsson, Simon, Lange, Tanja, Schwabe, Peter, and Yang, Bo-Yin. *EdDSA for more curves.* IACR Cryptology ePrint Archive, 2015:677, 2015.

[43] Boneh, Dan, Boyen, Xavier, and Shacham, Hovav. *Short group signatures.* In Proc. Adv. Cryptology-Crypto 04, ser. LNCS 3152, pp. 41–55. Springer-Verlag, 2004.

[44] Boneh, Dan and Shacham, Hovav. *Group signatures with verifier-local revocation.* In Proceedings of the 11th ACM conference on Computer and communications security, pp. 168–177. ACM, 2004.

[45] Borst, Johan, Preneel, Bart, and Rijmen, Vincent. *Cryptography on smart cards.* Computer Networks, 36(4):423–435, 2001. ISSN 1389-1286. doi:https://doi.org/10.1016/S1389-1286(01)00164-5.

[46] Bos, Joppe, Costello, Craig, Ducas, Léo, Mironov, Ilya, Naehrig, Michael, Nikolaenko, Valeria, Raghunathan, Ananth, and Stebila, Douglas. *Frodo: Take off the ring! practical, quantum-secure key exchange from LWE.* In Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, pp. 1006–1018. ACM, 2016.

[47] Bos, Joppe, Ducas, Léo, Kiltz, Eike, Lepoint, Tancrède, Lyubashevsky, Vadim, Schanck, John M, Schwabe, Peter, Seiler, Gregor, and Stehlé, Damien. *CRYSTALS-Kyber: a CCA-secure module-lattice-based KEM.* In 2018 IEEE European Symposium on Security and Privacy (EuroS&P). IEEE, 2018.

[48] Bos, Joppe W, Costello, Craig, Naehrig, Michael, and Stebila, Douglas. *Post-quantum key exchange for the TLS protocol from the ring learning with errors problem.* In Security and Privacy (SP), 2015 IEEE Symposium on, pp. 553–570. IEEE, 2015.

[49] Brakerski, Zvika and Vaikuntanathan, Vinod. *Efficient fully homomorphic encryption from (standard) LWE.* SIAM Journal on Computing, 43(2):831–871, 2014.

[50] Brickell, Justin and Shmatikov, Vitaly. *Efficient anonymity-preserving data collection.* In Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 76–85. ACM, 2006.

[51] Budka, Kenneth C, Deshpande, Jayant G, and Thottan, Marina. *Smart Grid data management.* In Communication networks for Smart Grids, pp. 265–284. Springer, 2014.

[52] Butin, Denis. *Hash-based signatures: State of play.* IEEE Security & Privacy, 15(4):37–43, 2017.

[53] Camenisch, Jan and Groth, Jens. *Group signatures: Better efficiency and new theoretical aspects.* In Security in Communication Networks, pp. 120–133. Springer, 2005.

[54] Camenisch, Jan and Van Herreweghen, Els. *Design and implementation of the idemix anonymous credential system.* In Proceedings of the 9th ACM conference on Computer and communications security, pp. 21–30. ACM, 2002. ISBN 1-58113-612-9. doi:10.1145/586110.586114.

[55] Canard, Sébastien, Desmoulins, Nicolas, Devigne, Julien, and Traoré, Jacques. *On the implementation of a pairing-based cryptographic protocol in a constrained device.* In Pairing-Based Cryptography–Pairing 2012, pp. 210–217. Springer, 2013.

[56] Cervantes, Christian, Poplade, Diego, Nogueira, Michele, and Santos, Aldri. *Detection of sinkhole attacks for supporting secure routing on 6LoWPAN for Internet of Things.* In Integrated Network Management (IM), 2015 IFIP/IEEE International Symposium on, pp. 606–611. IEEE, 2015.

[57] Chandran, Nishanth, Groth, Jens, and Sahai, Amit. *Ring signatures of sub-linear size without random oracles.* In International Colloquium on Automata, Languages, and Programming, pp. 423–434. Springer, 2007.

[58] Chang, C-C and Wu, T-C. *Remote password authentication with smart cards.* IEE Proceedings E-Computers and Digital Techniques, 138(3):165–168, 1991.

[59] Chatterjee, Sanjit and Menezes, Alfred. *On cryptographic protocols employing asymmetric pairings the role of ψ revisited.* Discrete Applied Mathematics, 159(13):1311–1322, 2011.

[60] Chaum, David and Van Heyst, Eugene. *Group signatures.* In Proceedings of the 10th annual international conference on Theory and application of cryptographic techniques, EUROCRYPT'91, pp. 257–265. Berlin, Heidelberg: Springer-Verlag, 1991. ISBN 3-540-54620-0.

[61] Chaum, David L. *Untraceable electronic mail, return addresses, and digital pseudonyms.* Communications of the ACM, 24(2):84–90, 1981.

[62] Chen, Bae-Ling, Kuo, Wen-Chung, and Wuu, Lih-Chyau. *Robust smart-card-based remote user password authenticationscheme.* International Journal of Communication Systems, 27(2):377–389, 2014. ISSN 1099-1131. doi:10.1002/dac.2368.

[63] Chen, Jiun-Ming and Yang, Bo-Yin. *A more secure and efficacious TTS signature scheme.* In International Conference on Information Security and Cryptology, pp. 320–338. Springer, 2003.

[64] Chen, Tien-Ho, Hsiang, Han-Cheng, and Shih, Wei-Kuan. *Security enhancement on an improvement on two remote user authentication schemes using smart cards.* Future Generation Computer Systems, 27(4):377–380, 2011. ISSN 0167-739X. doi:https://doi.org/10.1016/j.future.2010.08.007.

[65] Choi, Younsung, Lee, Donghoon, Kim, Jiye, Jung, Jaewook, Nam, Junghyun, and Won, Dongho. *Security enhanced user authentication protocol for wireless sensor networks using elliptic curves cryptography.* Sensors, 14(6):10081–10106, 2014. doi:10.3390/s140610081.

[66] Chu, Cheng-Kang, Liu, Joseph K, Huang, Xinyi, and Zhou, Jianying. *Verifier-local revocation group signatures with time-bound keys.* In Proceedings of the 7th ACM Symposium on Information, Computer and Communications Security, pp. 26–27. ACM, 2012.

[67] Chu, Cheng-Kang, Liu, Joseph K, Wong, Jun Wen, Zhao, Yunlei, and Zhou, Jianying. *Privacy-preserving smart metering with regional statistics and personal enquiry services.* In Proceedings of the 8th ACM SIGSAC symposium on Information, computer and communications security, pp. 369–380. ACM, 2013.

[68] Cohen, Henri, Frey, Gerhard, Avanzi, Roberto, Doche, Christophe, Lange, Tanja, Nguyen, Kim, and Vercauteren, Frederik. Handbook of elliptic and hyperelliptic curve cryptography. CRC press, 2005.

[69] Coron, Jean-Sébastien, Naccache, David, and Tibouchi, Mehdi. *Public key compression and modulus switching for fully homomorphic encryption over the integers.* In Advances in Cryptology–EUROCRYPT 2012, pp. 446–464. Springer, 2012.

[70] Costello, Craig, Jao, David, Longa, Patrick, Naehrig, Michael, Renes, Joost, and Urbanik, David. *Efficient compression of SIDH public keys.* In Annual International Conference on the Theory and Applications of Cryptographic Techniques, pp. 679–706. Springer, 2017.

[71] Costello, Craig, Longa, Patrick, and Naehrig, Michael. *Efficient algorithms for supersingular isogeny Diffie-Hellman.* In Annual Cryptology Conference, pp. 572–601. Springer, 2016.

[72] Cristofaro, Emiliano De and Soriente, Claudio. *Extended Capabilities for a Privacy-Enhanced Participatory Sensing Infrastructure (PEPSI).* CoRR, abs/1308.2921, 2013.

[73] Daemen, J. and Rijmen, V. The design of Rijndael: AES-the advanced encryption standard. Springer, 2002.

[74] Dagdelen, Özgür, Fischlin, Marc, Gagliardoni, Tommaso, Marson, Giorgia Azzurra, Mittelbach, Arno, and Onete, Cristina. *A cryptographic analysis of OPACITY.* In Computer Security–ESORICS 2013, pp. 345–362. Springer, 2013. ISBN 978-3-642-40203-6. doi:10.1007/978-3-642-40203-6$_$20.

[75] Damgård, Ivan. *Commitment schemes and zero-knowledge protocols.* In School organized by the European Educational Forum, pp. 63–86. Springer, 1998.

[76] Delerablée, Cécile and Pointcheval, David. *Dynamic fully anonymous short group signatures.* In Progress in Cryptology-VIETCRYPT 2006, pp. 193–210. Springer, 2006.

[77] Ding, Jintai and Petzoldt, Albrecht. *Current state of multivariate cryptography.* IEEE Security & Privacy, 15(4):28–36, 2017.

[78] Ding, Jintai and Schmidt, Dieter. *Rainbow, a new multivariable polynomial signature scheme.* In International Conference on Applied Cryptography and Network Security, pp. 164–175. Springer, 2005.

[79] Dingledine, Roger. *Tor: anonymity online*, 2012.

[80] Dodis, Yevgeniy, Kiayias, Aggelos, Nicolosi, Antonio, and Shoup, Victor. *Anonymous identification in ad hoc groups.* In International Conference on the Theory and Applications of Cryptographic Techniques, pp. 609–626. Springer, 2004.

[81] ElGamal, Taher. *A public key cryptosystem and a signature scheme based on discrete logarithms.* In Advances in Cryptology, pp. 10–18. Springer, 1985.

[82] Emura, Keita and Hayashi, Takuya. *A light-weight group signature scheme with time-token dependent linking.* In International Workshop on Lightweight Cryptography for Security and Privacy, pp. 37–57. Springer, 2015.

[83] Ferrara, Anna Lisa, Green, Matthew, Hohenberger, Susan, and Pedersen, Michael . *Practical Short Signature Batch Verification.* In Proceedings of the The Cryptographers' Track at the RSA Conference 2009 on Topics in Cryptology, CT-RSA '09, pp. 309–324. Springer-Verlag, 2009. ISBN 978-3-642-00861-0.

[84] Fiat, Amos and Shamir, Adi. *How to prove yourself: Practical solutions to identification and signature problems.* In Advances in Cryptology—CRYPTO'86, pp. 186–194. Springer, 1986.

[85] Freeman, David, Scott, Michael, and Teske, Edlyn. *A Taxonomy of Pairing-Friendly Elliptic Curves.* Journal of Cryptology, 23(2):224–280, 2010. ISSN 0933-2790. doi:10.1007/s00145-009-9048-z.

[86] Fujisaki, Eiichiro and Suzuki, Koutarou. *Traceable ring signature.* In International Workshop on Public Key Cryptography, pp. 181–200. Springer, 2007.

[87] Galbraith, Steven D, Petit, Christophe, and Silva, Javier. *Identification protocols and signature schemes based on supersingular isogeny problems.* In International Conference on the Theory and Application of Cryptology and Information Security, pp. 3–33. Springer, 2017.

[88] Gandolfi, Karine, Mourtel, Christophe, and Olivier, Francis. *Electromagnetic analysis: Concrete results.* In Cryptographic Hardware and Embedded Systems - CHES 2001, pp. 251–261. Springer, 2001. ISBN 978-3-540-44709-2. doi:10.1007/3-540-44709-1$_$21.

[89] Garcia, Flavio D, Van Rossum, Peter, Verdult, Roel, and Schreur, Ronny Wichers. *Wirelessly pickpocketing a Mifare Classic card.* In Security and Privacy, 2009 30th IEEE Symposium on, pp. 3–15. IEEE, 2009. ISBN 978-0-7695-3633-0. doi:10.1109/SP.2009.6.

[90] Gentry, Craig and Halevi, Shai. *Implementing Gentrys fully-homomorphic encryption scheme.* In Advances in Cryptology–EUROCRYPT 2011, pp. 129–148. Springer, 2011.

[91] Giannetsos, T., Gisdakis, S., and Papadimitratos, P. *Trustworthy People-Centric Sensing: Privacy, security and user incentives road-map.* In Ad Hoc Networking Workshop (MED-HOC-NET), 2014 13th Annual Mediterranean, pp. 39–46. 2014. doi:10.1109/MedHocNet.2014.6849103.

[92] Goldreich, Oded. *A short tutorial of zero-knowledge*, 2010.

[93] Goyal, Vipul, Pandey, Omkant, Sahai, Amit, and Waters, Brent. *Attribute-based encryption for fine-grained access control of encrypted data.* In Proceedings of the 13th ACM conference on Computer and communications security, pp. 89–98. Acm, 2006.

[94] Green, Matthew and Ateniese, Giuseppe. *Identity-based proxy re-encryption.* In Applied Cryptography and Network Security, pp. 288–306. Springer, 2007.

[95] Green, Matthew, Hohenberger, Susan, and Waters, Brent. *Outsourcing the Decryption of ABE Ciphertexts.* In USENIX Security Symposium, vol. 2011. 2011.

[96] Gubbi, Jayavardhana, Buyya, Rajkumar, Marusic, Slaven, and Palaniswami, Marimuthu. *Internet of Things (IoT): A vision, architectural elements, and future directions.* Future Generation Computer Systems, 29(7):1645–1660, 2013.

[97] Hajny, J. and Malina, L. *Practical Revocable Anonymous Credentials.* In Communications and Multimedia Security, pp. 211–213. Springer, 2012.

[98] Hajny, Jan, Dzurenda, Petr, and Malina, Lukas. *Privacy-enhanced data collection scheme for smart-metering.* In International Conference on Information Security and Cryptology, pp. 413–429. Springer, 2015.

[99] Hajny, Jan, Dzurenda, Petr, and Malina, Lukas. *Secure physical access control with strong cryptographic protection.* In e-Business and Telecommunications (ICETE), 2015 12th International Joint Conference on, vol. 4, pp. 220–227. IEEE, 2015.

[100] Hajny, Jan and Malina, Lukas. *Anonymous credentials with practical revocation.* In Satellite Telecommunications (ESTEL), 2012 IEEE First AESS European Conference on, pp. 1–6. IEEE, 2012.

[101] Hajny, Jan and Malina, Lukas. *Unlinkable attribute-based credentials with practical revocation on smart-cards.* In Proceedings of the 11th international conference on Smart Card Research and Advanced Applications, CARDIS'12, pp. 62–76. Springer-Verlag, 2013. ISBN 978-3-642-37287-2.

[102] Hajny, Jan, Malina, Lukas, Martinasek, Zdenek, and Zeman, Vaclav. *Privacy-preserving SVANETs-Privacy-preserving Simple Vehicular Ad-hoc Networks.* In SECRYPT, pp. 267–274. 2013.

[103] Hajny, Jan, Malina, Lukas, and Tethal, Ondrej. *Privacy-friendly access control based on personal attributes.* In International Workshop on Security, pp. 1–16. Springer, 2014. ISBN 978-3-319-09843-2. doi:10.1007/978-3-319-09843-2$_$1.

[104] Hajny, Jan, Malina, Lukas, and Zeman, Vaclav. *PRACTICAL ANONYMOUS AUTHENTICATION: Designing Anonymous Authentication for Everyday Use.* In Proceedings of the Secrypt 2011, pp. 405–408. 2011. ISBN 978-989-8425-18- 8.

[105] Hanzlik, Lucjan, Krzywiecki, Łukasz, and Kutyłowski, Mirosław. *Simplified PACE|AA Protocol.* In Information Security Practice and Experience, pp. 218–232. Springer, 2013. ISBN 978-3-642-38033-4. doi:10.1007/978-3-642-38033-4$_$16.

[106] Henzl, Martin and Hanacek, Petr. *A Security Formal Verification Method for Protocols Using Cryptographic Contactless Smart Cards.* Radioengineering, 25(1), 2016. doi:10.13164/re.2016.0132.

[107] Hoffstein, Jeffrey, Pipher, Jill, and Silverman, Joseph H. *NTRU: A ring-based public key cryptosystem.* In International Algorithmic Number Theory Symposium, pp. 267–288. Springer, 1998.

[108] Horvath, Tomas, Malina, Lukas, and Munster, Petr. *On security in gigabit passive optical networks.* In Fiber Optics in Access Network (FOAN), 2015 International Workshop on, pp. 51–55. IEEE, 2015.

[109] Huang, Xin, Fu, Rong, Chen, Bangdao, Zhang, Tingting, and Roscoe, AW. *User interactive internet of things privacy preserved access control.* In Internet Technology And Secured Transactions, 2012 International Conference for, pp. 597–602. IEEE, 2012.

[110] Hwang, Jung Yeon, Chen, Liqun, Cho, Hyun Sook, and Nyang, DaeHun. *Short dynamic group signature scheme supporting controllable linkability.* IEEE Transactions on Information Forensics and Security, 10(6):1109–1124, 2015.

[111] Hwang, Jung Yeon, Lee, Sokjoon, Chung, Byung-Ho, Cho, Hyun Sook, and Nyang, DaeHun. *Short group signatures with controllable linkability.* In Lightweight Security & Privacy: Devices, Protocols and Applications (LightSec), 2011 Workshop on, pp. 44–52. IEEE, 2011.

[112] Isshiki, Toshiyuki, Mori, Kengo, Sako, Kazue, Teranishi, Isamu, and Yonezawa, Shoko. *Using group signatures for identity management and its implementation.* In Proceedings of the second ACM workshop on Digital identity management, pp. 73–78. ACM, 2006.

[113] Jalali, Amir, Azarderakhsh, Reza, and Mozaffari-Kermani, Mehran. *Efficient post-quantum undeniable signature on 64-bit ARM.* In International Conference on Selected Areas in Cryptography, pp. 281–298. Springer, 2017.

[114] Jao, David and De Feo, Luca. *Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies.* In International Workshop on Post-Quantum Cryptography, pp. 19–34. Springer, 2011.

[115] Johnson, Don, Menezes, Alfred, and Vanstone, Scott. *The elliptic curve digital signature algorithm (ECDSA).* International journal of information security, 1(1):36–63, 2001.

[116] Kim, Kitae, Yie, Ikkwon, Lim, Seongan, and Nyang, Daehun. *Batch Verification and Finding Invalid Signatures in a Group Signature Scheme.* IJ Network Security, 13(2):61–70, 2011.

[117] Kim, Mucheol, Seo, Jiwan, Noh, Sanghyun, and Han, Sangyong. *Identity management-based social trust model for mediating information sharing and privacy enhancement.* Security and Communication Networks, 5(8):887–897, 2012. ISSN 1939-0122. doi:10.1002/sec.379.

[118] Kim, Young-Jin, Kolesnikov, Vladimir, Kim, Hongseok, and Thottan, Marina. *SSTP: a scalable and secure transport protocol for smart grid data collection.* In Smart Grid Communications (SmartGridComm), 2011 IEEE International Conference on, pp. 161–166. IEEE, 2011.

[119] Kipnis, Aviad, Patarin, Jacques, and Goubin, Louis. *Unbalanced oil and vinegar signature schemes.* In International Conference on the Theory and Applications of Cryptographic Techniques, pp. 206–222. Springer, 1999.

[120] Kocher, Paul, Jaffe, Joshua, and Jun, Benjamin. *Differential power analysis.* In Advances in cryptology - CRYPTO 99, pp. 789–789. Springer, 1999. ISBN 978-3-540-48405-9. doi: 10.1007/3-540-48405-1$_$25.

[121] Kocher, Paul C. *Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems.* In Annual International Cryptology Conference, pp. 104–113. Springer, 1996. ISBN 978-3-540-68697-2. doi:10.1007/3-540-68697-5$_$9.

[122] Krzywiecki, Łukasz. *Schnorr-Like Identification Scheme Resistant to Malicious Subliminal Setting of Ephemeral Secret.* In International Conference for Information Technology and Communications, pp. 137–148. Springer, 2016. ISBN 978-3-319-47238-6. doi: 10.1007/978-3-319-47238-6$_$10.

[123] Kurosawa, Kaoru and Heng, Swee-Huay. *From digital signature to ID-based identification/signature.* In International Workshop on Public Key Cryptography, pp. 248–261. Springer, 2004. ISBN 978-3-540-24632-9. doi:10.1007/978-3-540-24632-9$_$18.

[124] Kwon, Taekyoung. *Practical authenticated key agreement using passwords.* In International Conference on Information Security, pp. 1–12. Springer, 2004.

[125] Lamport, Leslie. *Constructing digital signatures from a one-way function.* Tech. rep., Technical Report CSL-98, SRI International Palo Alto, 1979.

[126] Li, Chun-Ta. *Secure smart card based password authentication scheme with user anonymity.* Information Technology and Control, 40(2):157–162, 2011.

[127] Li, Fagen, Zheng, Zhaohui, and Jin, Chunhua. *Secure and efficient data transmission in the Internet of Things.* Telecommunication Systems, pp. 1–12, 2015.

[128] Li, Hongwei, Lin, Xiaodong, Yang, Haomiao, Liang, X, Lu, R, and Shen, X. *EPPDR: An Efficient Privacy-Preserving Demand Response Scheme with Adaptive Key Evolution in Smart Grid.* IEEE Transactions on Parallel and Distributed Systems, 1, 2013.

[129] Li, Jin, Au, Man Ho, Susilo, Willy, Xie, Dongqing, and Ren, Kui. *Attribute-based signature and its applications.* In Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security, pp. 60–69. ACM, 2010.

[130] Li, Xiong, Niu, Jianwei, Khan, Muhammad Khurram, and Liao, Junguo. *An enhanced smart card based remote user password authentication scheme.* Journal of Network and Computer Applications, 36(5):1365–1371, 2013.

[131] Li, Xiong, Xiong, Yongping, Ma, Jian, and Wang, Wendong. *An efficient and security dynamic identity based authentication protocol for multi-server architecture using smart cards.* Journal of Network and Computer Applications, 35(2):763–769, 2012.

[132] Lin, Hsiao-Ying and Tzeng, Wen-Guey. *An efficient solution to the millionaire problem based on homomorphic encryption.* In Applied Cryptography and Network Security, pp. 97–134. Springer, 2005.

[133] Liu, Joseph K, Au, Man Ho, Susilo, Willy, and Zhou, Jianying. *Online/offline ring signature scheme.* In International Conference on Information and Communications Security, pp. 80–90. Springer, 2009.

[134] Liu, Joseph K, Au, Man Ho, Susilo, Willy, and Zhou, Jianying. *Linkable ring signature with unconditional anonymity.* IEEE Transactions on Knowledge and Data Engineering, 26(1):157–165, 2014.

[135] Liu, Joseph K, Wei, Victor K, and Wong, Duncan S. *Linkable spontaneous anonymous group signature for ad hoc groups.* In Australasian Conference on Information Security and Privacy, pp. 325–335. Springer, 2004.

[136] Longa, Patrick and Naehrig, Michael. *Speeding up the number theoretic transform for faster ideal lattice-based cryptography.* In International Conference on Cryptology and Network Security, pp. 124–139. Springer, 2016.

[137] Lu, Rongxing, Liang, Xiaohui, Li, Xu, Lin, Xiaodong, and Shen, Xuemin Sherman. *Eppa: An efficient and privacy-preserving aggregation scheme for secure smart grid communications.* Parallel and Distributed Systems, IEEE Transactions on, 23(9):1621–1631, 2012.

[138] Madhusudhan, R and Mittal, RC. *Dynamic ID-based remote user password authentication schemes using smart cards: A review.* Journal of Network and Computer Applications, 35(4):1235–1248, 2012. ISSN 1084-8045. doi:https://doi.org/10.1016/j.jnca.2012.01.007.

[139] Malina, L. *Privacy Preserving Cryptographic Protocols for Secure Heterogeneous Networks*, 2014.

[140] Malina, L., Dzurenda, P., Hajny, J., and Martinasek, Z. *Assessment of Cryptography Support and Security on Programmable Smart Cards.* In 2018 41st International Conference on Telecommunications and Signal Processing (TSP), pp. 1–5. 2018. doi:10.1109/TSP.2018.8441334.

[141] Malina, L. and Hajny, J. *Accelerated modular arithmetic for low-performance devices.* In the 34th International Conference on Telecommunications and Signal Processing (TSP), pp. 131 –135. 2011.

[142] Malina, Lukas, Castellà-Roca, Jordi, Vives-Guasch, Arnau, and Hajny, Jan. *Short-term linkable group signatures with categorized batch verification.* In Foundations and Practice of Security, pp. 244–260. Springer, 2013.

[143] Malina, Lukas, Clupek, Vlastimil, Martinasek, Zdenek, Hajny, Jan, Oguchi, Kimio, and Zeman, Vaclav. *Evaluation of Software-Oriented Block Ciphers on Smartphones.* In Foundations and Practice of Security, Lecture Notes in Computer Science, pp. 353–368. Springer International Publishing, 2014. ISBN 978-3-319-05301-1. doi:10.1007/978-3-319-05302-8_22.

[144] Malina, Lukas and Hajny, Jan. *Efficient Modular Multiplication for Programmable Smart-Cards.* Telecommunication Systems, 55:1–9, 2014. ISSN 1018-4864.

[145] Malina, Lukas and Hajny, Jan. *Privacy-preserving framework for geosocial applications.* Security and Communication Networks, 7(11):1764–1779, 2014.

[146] Malina, Lukas, Hajny, Jan, and Martinasek, Zdenek. *Efficient Group Signatures with Verifier-local Revocation Employing a Natural Expiration.* In SECRYPT, pp. 555–560. 2013.

[147] Malina, Lukas, Hajny, Jan, and Zeman, Vaclav. *Group signatures for secure and privacy preserving vehicular ad hoc networks.* In Proceedings of the 8h ACM symposium on QoS and security for wireless and mobile networks, pp. 71–74. ACM, 2012.

[148] Malina, Lukas, Hajny, Jan, and Zeman, Vaclav. *Trade-off between signature aggregation and batch verification.* In Telecommunications and Signal Processing (TSP), 2013 36th International Conference on, pp. 57–61. IEEE, 2013.

[149] Mceliece, Robert J. *A public-key cryptosystem based on algebraic.* Coding Thv, 4244:114–116, 1978.

[150] Merkle, Ralph C. *A certified digital signature.* In Conference on the Theory and Application of Cryptology, pp. 218–238. Springer, 1989.

[151] Miller, Victor S. *The Weil Pairing, and Its Efficient Calculation.* J. Cryptol., 17(4):235–261, 2004. ISSN 0933-2790.

[152] Mishra, Dheerendra, Das, Ashok Kumar, and Mukhopadhyay, Sourav. *A secure and efficient ECC-based user anonymity-preserving session initiation authentication protocol using smart card.* Peer-to-peer networking and applications, 9(1):171–192, 2016. ISSN 1936-6450. doi:10.1007/s12083-014-0321-z.

[153] Nakanishi, Toru and Funabiki, Nobuo. *A short verifier-local revocation group signature scheme with backward unlinkability.* IEICE transactions on fundamentals of electronics, communications and computer sciences, 90(9):1793–1802, 2007.

[154] Nejatollahi, Hamid, Dutt, Nikil, Ray, Sandip, Regazzoni, Francesco, Banerjee, Indranil, and Cammarota, Rosario. *Software and Hardware Implementation of Lattice-Cased Cryptography Schemes.* 2017.

[155] Nemec, Matus, Sys, Marek, Svenda, Petr, Klinec, Dusan, and Matyas, Vashek. *The Return of Coppersmith's Attack: Practical Factorization of Widely Used RSA Moduli.* In 24th ACM Conference on Computer and Communications Security (CCS'2017), pp. 1631–1648. ACM, 2017. ISBN 978-1-4503-4946-8/17/10.

[156] Niederreiter, Harald. *Knapsack-type cryptosystems and algebraic coding theory.* Prob. Control and Inf. Theory, 15(2):159–166, 1986.

[157] Noether, Shen, Mackenzie, Adam, et al. *Ring confidential transactions.* Ledger, 1:1–18, 2016.

[158] Okamoto, Tatsuaki and Uchiyama, Shigenori. *A new public-key cryptosystem as secure as factoring.* In Advances in CryptologyEUROCRYPT'98, pp. 308–318. Springer, 1998.

[159] Paillier, Pascal. *Public-key cryptosystems based on composite degree residuosity classes.* In Advances in cryptologyEUROCRYPT99, pp. 223–238. Springer, 1999.

[160] Paquin, Christian and Zaverucha, Greg. *U-prove cryptographic specification v1. 1.* Tech. rep., Microsoft Technical Report, http://connect. microsoft. com/site1188, 2011.

[161] Patarin, Jacques. *Hidden fields equations (HFE) and isomorphisms of polynomials (IP): Two new families of asymmetric algorithms.* In International Conference on the Theory and Applications of Cryptographic Techniques, pp. 33–48. Springer, 1996.

[162] Pawlowski, M.P., Jara, A.J., and Ogorzalek, M.J. *EAP for IoT: More Efficient Transport of Authentication Data – TEPANOM Case Study.* In Advanced Information Networking and Applications Workshops (WAINA), 2015 IEEE 29th International Conference on, pp. 694–699. 2015. doi:10.1109/WAINA.2015.53.

[163] Pedersen, Torben Pryds. *Non-interactive and information-theoretic secure verifiable secret sharing.* In Advances in CryptologyCRYPTO91, pp. 129–140. Springer, 1992.

[164] Rabin, Michael O. *Digitalized signatures and public-key functions as intractable as factorization.* Tech. rep., Massachusetts ints. of tech. Cambridge lab for computer science, 1979.

[165] Raza, Shahid. Lightweight Security Solutions for the Internet of Things. Ph.D. thesis, Mälardalen University, Västerås, Sweden, 2013.

[166] Rescorla, Eric and Modadugu, Nagendra. *Datagram transport layer security version 1.2.* 2012.

[167] Rivest, Ronald L, Shamir, Adi, and Tauman, Yael. *How to leak a secret.* In International Conference on the Theory and Application of Cryptology and Information Security, pp. 552–565. Springer, 2001.

[168] Rottondi, Cristina, Mauri, Giulia, and Verticale, Giacomo. *A protocol for metering data pseudonymization in smart grids.* Transactions on Emerging Telecommunications Technologies, 26(5):876–892, 2015.

[169] Sahai, Amit and Vadhan, Salil. *A complete problem for statistical zero knowledge.* Journal of the ACM (JACM), 50(2):196–249, 2003.

[170] Schindler, Werner, Lemke, Kerstin, and Paar, Christof. *A stochastic model for differential side channel cryptanalysis.* In International Workshop on Cryptographic Hardware and Embedded Systems, pp. 30–46. Springer, 2005. ISBN 3-540-28474-5, 978-3-540-28474-1. doi:10.1007/11545262$_$3.

[171] Schnorr, Claus-Peter. *Efficient identification and signatures for smart cards.* In Conference on the Theory and Application of Cryptology, pp. 239–252. Springer, 1989.

[172] Schnorr, Claus-Peter. *Efficient signature generation by smart cards.* Journal of cryptology, 4(3):161–174, 1991.

[173] Scott, Mike. *Efficient Implementation of Cryptographic pairings.* 2009.

[174] Sen, Jaydip. *Privacy preservation technologies in Internet of Things.* In Proceedings of the International Conference on Emerging Trends in Mathematics, Technology and Management, pp. 496–504. 2010.

[175] Sendrier, Nicolas. *Code-Based Cryptography: State of the Art and Perspectives.* IEEE Security & Privacy, 15(4):44–50, 2017.

[176] Seuschek, Hermann, Khurana, Piyush, and Sigl, Georg. *HiPeC—High Performance Cryptographic Service for Heterogeneous Network-on-Chip Systems.* IFAC-PapersOnLine, 48(4):31–36, 2015.

[177] Shacham, Hovav and Waters, Brent. *Efficient ring signatures without random oracles.* In International Workshop on Public Key Cryptography, pp. 166–180. Springer, 2007.

[178] Shamir, Adi. *How to share a secret.* Communications of the ACM, 22(11):612–613, 1979.

[179] Shelby, Z, Hartke, K, Bormann, C, and Frank, B. *Constrained application protocol (CoAP), draft-ietf-core-coap-18 (work in progress), sl: IETF 2013.*

[180] Shin, Minho, Cornelius, Cory, Peebles, Dan, Kapadia, Apu, Kotz, David, and Triandopoulos, Nikos. *AnonySense: A system for anonymous opportunistic sensing.* Pervasive and Mobile Computing, 7(1):16–30, 2011.

[181] Sicari, Sabrina, Grieco, Luigi Alfredo, Boggia, Gennaro, and Coen-Porisini, Alberto. *DyDAP: A dynamic data aggregation scheme for privacy aware wireless sensor networks.* Journal of Systems and Software, 85(1):152–166, 2012.

[182] Sinha, Anshuman. *A survey of system security in contactless electronic passports.* International Journal of Critical Infrastructure Protection, 4(3):154–164, 2011.

[183] Song, Ronggong. *Advanced smart card based password authentication protocol.* Computer Standards & Interfaces, 32(5):321–325, 2010. ISSN 0920-5489. doi:10.1016/j.csi.2010.03.008.

[184] Stebila, Douglas and Mosca, Michele. *Post-quantum key exchange for the internet and the open quantum safe project.* In International Conference on Selected Areas in Cryptography, pp. 14–37. Springer, 2016.

[185] Su, Jinshu, Cao, Dan, Zhao, Baokang, Wang, Xiaofeng, and You, Ilsun. *ePASS: An expressive attribute-based signature scheme with privacy and an unforgeability guarantee for the Internet of Things.* Future Generation Computer Systems, 33:11–18, 2014.

[186] Sun, Guozi, Huang, Siqi, Bao, Wan, Yang, Yitao, and Wang, Zhiwei. *A privacy protection policy combined with privacy homomorphism in the Internet of Things.* In Computer Communication and Networks (ICCCN), 2014 23rd International Conference on, pp. 1–6. IEEE, 2014.

[187] Sun, Shi-Feng, Au, Man Ho, Liu, Joseph K, and Yuen, Tsz Hon. *RingCT 2.0: A Compact Accumulator-Based (Linkable Ring Signature) Protocol for Blockchain Cryptocurrency Monero.* In European Symposium on Research in Computer Security, pp. 456–474. Springer, 2017.

[188] Talpur, Mir Sajjad Hussain, Bhuiyan, Md Zakirul Alam, and Wang, Guojun. *Shared–node IoT network architecture with ubiquitous homomorphic encryption for healthcare monitoring.* International Journal of Embedded Systems, 7(1):43–54, 2014.

[189] Teh, Thong-Yun, Lee, Yik-Shu, Cheah, Zi-Yik, and Chin, Ji-Jian. *IBI-Mobile Authentication: A Prototype to Facilitate Access Control Using Identity-Based Identification on Mobile Smart Devices.* Wireless Personal Communications, 94(1):127–144, 2017. ISSN 1572-834X. doi:10.1007/s11277-016-3320-y.

[190] Tsang, Patrick P and Wei, Victor K. *Short linkable ring signatures for e-voting, e-cash and attestation.* In International Conference on Information Security Practice and Experience, pp. 48–60. Springer, 2005.

[191] Tsiounis, Y. and Yung, M. *On the security of ElGamal based encryption.* In Public Key Cryptography, pp. 117–134. Springer, 1998.

[192] Ukil, Arijit, Bandyopadhyay, Soma, Joseph, Joel, Banahatti, Vijayanand, and Lodha, Sachin. *Negotiation-based privacy preservation scheme in internet of things platform.* In Proceedings of the First International Conference on Security of Internet of Things, pp. 75–84. ACM, 2012.

[193] Ullmann, Markus, Kügler, Dennis, Neumann, Heike, Stappert, Sebastian, and Vögeler, Matthias. *Password authenticated key agreement for contactless smart cards.* Communications of the ACM, 177, 2008.

[194] Unterluggauer, Thomas and Wenger, Erich. *Efficient pairings and ECC for embedded systems.* In International Workshop on Cryptographic Hardware and Embedded Systems, pp. 298–315. Springer, 2014.

[195] Van Saberhagen, Nicolas. *Cryptonote v 2. 0*, 2013.

[196] Vaudenay, Serge. *The Security of DSA and ECDSA.* In Public Key CryptographyPKC 2003, pp. 309–323. Springer, 2002.

[197] Vučinić, Mališa, Tourancheau, Bernard, Rousseau, Franck, Duda, Andrzej, Damon, Laurent, and Guizzetti, Roberto. *OSCAR: Object security architecture for the Internet of Things.* Ad Hoc Networks, 2014.

[198] Wang, Chenyu and Xu, Guoai. *Cryptanalysis of Three Password-Based Remote User Authentication Schemes with Non-Tamper-Resistant Smart Card.* Security and Communication Networks, 2017, 2017. doi:10.1155/2017/1619741.

[199] Wang, Ding and Ma, Chunguang. *On the (in) security of some smart-card-based password authentication schemes for WSN.* IACR Cryptology ePrint Archive, 2012:581, 2012.

[200] Wang, Ding and Wang, Ping. *On the anonymity of two-factor authentication schemes for wireless sensor networks: Attacks, principle and solutions.* Computer Networks, 73:41–57, 2014. ISSN 1389-1286. doi:10.1016/j.comnet.2014.07.010.

[201] Wang, Ding and Wang, Ping. *Offline dictionary attack on password authentication schemes using smart cards.* In Information Security, pp. 221–237. Springer, 2015. ISBN 978-3-319-27659-5. doi:10.1007/978-3-319-27659-5$_$16.

[202] Washington, Lawrence C. Elliptic curves: number theory and cryptography. Chapman and Hall/CRC, 2003.

[203] Waters, Brent. *Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization.* In Public Key Cryptography–PKC 2011, pp. 53–70. Springer, 2011.

[204] Werner, Martin. *Privacy-protected communication for location-based services.* Security and Communication Networks, pp. n/a–n/a, 2011. ISSN 1939-0122. doi:10.1002/sec.330.

[205] Whitnall, Carolyn and Oswald, Elisabeth. *Robust profiling for DPA-style attacks.* In International Workshop on Cryptographic Hardware and Embedded Systems, pp. 3–21. Springer, 2015. ISBN 978-3-662-48324-4. doi:10.1007/978-3-662-48324-4$_$1.

[206] Wong, Kok-Seng and Kim, Myung Ho. *Towards Self-Awareness Privacy Protection for Internet of Things Data Collection.* Journal of Applied Mathematics, 2014, 2014.

[207] Wu, Fan, Xu, Lili, Kumari, Saru, and Li, Xiong. *A new and secure authentication scheme for wireless sensor networks with formal proof.* Peer-to-Peer Networking and Applications, 10(1):16–30, 2017. ISSN 1936-6450. doi:10.1007/s12083-015-0404-5.

[208] Wu, Qianhong, Susilo, Willy, Mu, Yi, and Zhang, Fangguo. *Ad hoc group signatures.* In International Workshop on Security, pp. 120–135. Springer, 2006.

[209] Xie, Qi, Liu, Wen-hao, Wang, Sheng-bao, Hu, Bin, Dong, Na, and Yu, Xiu-yuan. *Robust password and smart card based authentication scheme with smart card revocation.* Journal of Shanghai Jiaotong University (Science), 19:418–424, 2014.

[210] Xie, Qi, Tang, Zhixiong, and Chen, Kefei. *Cryptanalysis and improvement on anonymous three-factor authentication scheme for mobile networks.* Computers & Electrical Engineering, 59(Supplement C):218 – 230, 2017. ISSN 0045-7906. doi:https://doi.org/10.1016/j.compeleceng.2016.11.038.

[211] Xue, Kaiping, Hong, Peilin, and Ma, Changsha. *A lightweight dynamic pseudonym identity based authentication and key agreement protocol without verification tables for multi-server architecture.* Journal of Computer and System Sciences, 80(1):195–206, 2014. ISSN 0022-0000. doi:https://doi.org/10.1016/j.jcss.2013.07.004.

[212] Yang, Haomin, Zhang, Yaoxue, Zhou, Yuezhi, Fu, Xiaoming, Liu, Hao, and Vasilakos, Athanasios V. *Provably secure three-party authenticated key agreement protocol using smart cards.* Computer Networks, 58:29–38, 2014. ISSN 1389-1286. doi:https://doi.org/10.1016/j.comnet.2013.08.020.

[213] Yang, Xu, Wu, Wei, Liu, Joseph K, and Chen, Xiaofeng. *Lightweight anonymous authentication for ad hoc group: A ring signature approach.* In International Conference on Provable Security, pp. 215–226. Springer, 2015.

[214] Yang, Zhiqiang, Zhong, Sheng, and Wright, Rebecca N. *Anonymity-preserving data collection.* In Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining, pp. 334–343. ACM, 2005.

[215] Yeh, Hsiu-Lien, Chen, Tien-Ho, Liu, Pin-Chuan, Kim, Tai-Hoo, and Wei, Hsin-Wen. *A secured authentication protocol for wireless sensor networks using elliptic curves cryptography.* Sensors, 11(5):4767–4779, 2011. doi:10.3390/s110504767.

[216] Yukun, Niu, Xiaobin, Tan, Shi, Chen, Haifeng, Wang, Kai, Yu, and Zhiyong, Bu. *A security privacy protection scheme for data collection of smart meters based on homomorphic encryption.* In EUROCON, 2013 IEEE, pp. 1401–1405. 2013. doi:10.1109/EUROCON.2013.6625161.

[217] Zeng, Shengke, Li, Qinyi, Qin, Zhiguang, and Lu, Qing. *Non-interactive deniable ring signature without random oracles*. Security and Communication Networks, 9(12):1810–1819, 2016.

[218] Zhang, Kuan, Liang, Xiaohui, Lu, Rongxing, and Shen, Xuemin. *Sybil Attacks and Their Defenses in the Internet of Things*. Internet of Things Journal, IEEE, 1(5):372–383, 2014.

[219] Zhou, Yuanyuan, Yu, Yu, Standaert, François-Xavier, and Quisquater, Jean-Jacques. *On the need of physical security for small embedded devices: a case study with COMP128-1 implementations in SIM cards*. In International Conference on Financial Cryptography and Data Security, pp. 230–238. Springer, 2013. ISBN 978-3-642-39884-1. doi:10.1007/978-3-642-39884-1$_$20.

List of Abbreviations

3PAKA	Three-Party Authenticated Key Agreement
ABE	Attribute-Based Encryption
ABS	Attribute-Based Signatures
ADS	Anonymous Digital Signatures
AES	Advanced Encryption Standard
AKA	Authenticated Key Agreement
ANSSI	Agence nationale de la srits systemes d'information
APDU	Application Protocol Data Unit
API	Application Programming Interface
BBS04	the Boneh, Boyen and Shacham group signature scheme
BCNS	the lattice-based scheme proposed by Bos, Costello, Naehrig, Stebila
BDHP	Bilinear Diffie-Hellman Problem
BIDHP	Bilinear Inverse Diffie-Hellman Problem
BN	Barreto-Naehrig Curves
BS04	the Boneh, Shacham group signature scheme
BSDHP	Bilinear Square Diffie-Hellman Problem
CDHP	Computational Diffie-Hellman Problem
CG	the Camenisch and Groth group signature scheme
COAP	Constrained Application Protocol
CP-ABE	Ciphertext-Policy Attribute-Based Encryption
CPU	Central Processing Unit
CRT	Chinese Remainder Theorem
CZK	Computational Zero-Knowledge protocol
DBS	Database Server
DDHP	Decision Diffie-Hellman Problem
DES	Data Encryption Standard
DH	Diffie-Hellman Protocol
DL	Discrete Logarithm
DLIN	Decisional Linear
DLP	Discrete Logarithm Problem
DLDHP	Decision Linear Diffie-Hellman Problem
DP	the Delerablee and Pointcheval group signature scheme
DSA	Digital Signature Algorithm
DTLS	Datagram Transport Layer Security
EC	Elliptic Curves
ECC	Elliptic Curves Cryptography
ECDH	Elliptic Curve Diffie Hellman protocol
ECDLP	Elliptic Curve Discrete Logarithm Problem
ECDSA	Elliptic Curve Digital Signature Algorithm
EdDSA	Edwards-Curve Digital Signature Algorithm
EEPROM	Electrically Erasable Programmable Read-Only Memory
EH	the Emura Hayashi group signature scheme
FHE	Full Homomorphic Encryption
FPGA	Field-Programmable Gate Array
GM	Group Manager

GMP	GNU Multiple Precision Arithmetic Library
GMSK	Group Manager Secret Key
GPK	Group Public Key
H	Hash function
HFE	Hidden Field Equations
HIP	Host Identity Protocol
HLCCN	the Hwang *et al.* group signature scheme
HM	the Hajny Malina attribute authentication scheme
HTTP	Hypertext Transfer Protocol
HVZK	Honest Verifier Zero Knowledge
ICT	Information and Communications Technology
ID	Identity
IoT	Internet of Things
IP	Internet Protocol
IPsec	Internet Protocol Security protocol
ISO/IEC	International Organization for Standardization/International Electrotechnical Commission
IrDA	Infrared Data Association
IZK	Interactive Zero-Knowledge protocol
JDK	Java Development Kit
JNI	Java Native Interface
JPBC	Java Pairing-Based Cryptography Library
KP-ABE	Key-Policy - Attribute Based Encryption
LIBPQP	Python post-quantum library
LORAWAN	Long Range Wide Area Network)
LSAG	Linkable Spontaneous Anonymous Group signature
MAC	Medium Access Control
MCL	the pairing-based cryptography library
MD5	Message-Digest algorithm
MHM13	the Malina Hajny Martinasek group signature scheme
MIRACL	Multiprecision Integer and Rational Arithmetic Cryptographic Library
MLSAG	Multilayered Linkable Spontaneous Anonymous Group signature
MNT	Miyaji-Nakabayashi-Takano curves
MPKC	Multivariate Public-Key Cryptosystems
MSS	Merkle Signature Scheme
MULTOS	Multi-application smart card Operating System
NDK	Android Native Development Kit
NF07	the Nakanishi Funabiki group signature scheme
NFC	Near Field Communication
NIST	National Institute of Standards and Technology
NIWI	Non-Interactive Witness-Indistinguishable
NIZK	Non-Interactive Zero-Knowledge protocol
NTRU	a lattice-based public key cryptosystem
OPACITY	The Open Protocol for Access Control Identification and Ticketing with PrivacY
OPRF	Oblivious Pseudo-Random Functions
P	Prover

PACE	Password Authenticated Connection Establishment
PAN	Personal Area Network
PBC	Pairing-Based Cryptography
PC	Personal Computer
PCD	Proximity Coupling Device
PETs	Privacy Enhancing Technologies
PHE	Partially Homomorphic Encryption
PICC	Proximity Integrated Circuit Cards
PIN	Personal Identification Number
PK	Proof of Knowledge
PKDL	Proof of Knowledge of Discrete Logarithm
PKI	Public Key Infrastructure
PQC	Post-Quantum Cryptography
PRNG	Pseudo-Random Number Generator
PZK	Perfect Zero-Knowledge protocol
qSDHP	q-Strong Diffie-Hellman problem
RA	Registration Authority
RAM	Random-Access Memory
RELIC	an Efficient LIbrary for Cryptography
RFC	Request For Comments
RFID	Radio-Frequency IDentification
RLWE	Ring Learning With Errors
RM	Revocation Manager
RND	a Random Number function
RNG	Random Number Generator
RO	Random Oracle
ROM	Read-Only Memory
RS	Ring Signatures
RSA	the Rivest, Shamir, Adleman scheme
RSAP	the RSA problem
SAM	Secure Access Module
SCADA	Supervisory Control And Data Acquisition
SDHP	Strong Diffie-Hellman Problem
SEAPOL	Slim Extensible Authentication Protocol Over LAN
SHA	Secure Hash Algorithm
SIDH	the isogeny-based Supersingular Isogeny Diffie-Hellman (SIDH) scheme
SIM	Subscriber Identification Module
SRSAP	the Strong RSA Problem
SP	Service Provider
SSH	Secure Shell)
SSL	Secure Sockets Layer
SSTP	Secure Socket Tunneling Protocol
SVP	the Shortest Vector Problem
SZK	Statistical Zero-Knowledge protocol
TA	Trusted Authority
TCP	Transmission Control Protocol
TDES	Triple Data Encryption Standard

TEPANOM	Trust Extension Protocol for Authentication of New de- ployed Objects and sensors through the Manufacturer
TEPLA	University of Tsukuba Elliptic Curve and Pairing Library
TLS	Transport Layer Security Protocol
TP-AMP	the TP-AMP protocol
TPM	Trusted Platform Module
TRNG	True Random Number Generator
TTP	Third Trusted Party
U	User
UOV	the Unbalanced Oil and Vinegar Cryptosystem
VANET	Vehicular Ad-Hoc Network
VLR	Verifier Local Revocation
WSN	Wireless Sensor Network
XDHP	eXternal Diffie-Hellman Problem
XTEA	the eXternal Tiny Encryption Algorithm
ZK	Zero-Knowledge