論文 / 著書情報
Article / Book Information

題目(和文)	
Title(English)	Cryptographic Credential Schemes and Their Applications in Contact Tracing
著者(和文)	WangPengfei
Author(English)	Pengfei Wang
出典(和文)	学位:博士(理学), 学位授与機関:東京工業大学, 報告番号:甲第12832号, 授与年月日:2024年9月20日, 学位の種別:課程博士, 審査員:田中 圭介,伊東 利哉,尾形 わかは,鹿島 亮,安永 憲司
Citation(English)	Degree:Doctor (Science), Conferring organization: Tokyo Institute of Technology, Report number:甲第12832号, Conferred date:2024/9/20, Degree Type:Course doctor, Examiner:,,,,
学位種別(和文)	博士論文
Type(English)	Doctoral Thesis

Cryptographic Credential Schemes and Their Applications in Contact Tracing

Pengfei Wang
Supervisor: Keisuke Tanaka

Department of Mathematical and Computing Science
Tokyo Institute of Technology

June 4, 2024

Acknowledgements

Foremost, I would like to express my sincere and heartfelt gratitude to Professor Keisuke Tanaka for his invaluable guidance, unwavering support, and profound insights throughout my doctoral journey — it has always been my honor to work with exceptional classmates in his laboratory. Tokyo Institute of Technology was my first stop in Japan, and Professor Tanaka's advice and mentorship have been invaluable in shaping my academic career within school, my career path after graduation, and my daily life outside apart from above, especially during the craziest days of COVID-19 pandemic.

I am deeply thankful to Professor Mario Larangeira for his scholarly guidance and constructive feedback during my academic researches. His expertise and dedication have greatly enriched my understanding in cryptography.

I would also like to express my thanks to my research partner Xiangyu Su, for our cooperation in multiple researches, and our friendship outside school. Despite starting our doctoral courses in the same year, he had already been a part of Professor Tanaka's laboratory before I joined. His guidance and support during those initial days were invaluable, especially as I was still getting acquainted with everyone.

I am also indebted to research partner Maxim Jourenko, for his collaboration and contributions to our research projects. His knowledge, enthusiasm and camaraderie have made my journey rewarding and enjoyable.

I would also like to extend my gratitude to all my friends who helped me during these years in other departments of Tokyo Institute of Technology, and other research societies outside Tokyo Institute of Technology, because my multi-disciplinary researches sometimes involve knowledge outside of my expertise, such as aerodynamics, epidemiology, and microbiology. Their patient explanation and selfless effort helped me a lot in my researches.

Finally, I am immensely thankful to my girlfriend and my family for their understanding and encouragement throughout this journey. Their love, encouragement, and belief in me have been my source of strength and inspiration.

Hasta la vista.

Contents

Chapter 1

Introduction

1.1 Backgrounds

Although it has been many years since COVID-19's outbreak, many concerns raised by COVID-19 pandemic still remain — not only in epidemiology, but also in cryptography.

Due to COVID-19's high infectiousness, it is our first time in history of using digital contact tracing approach. Contact tracing, by its name, means to trace contacts of infected individuals to control the spread of disease. This technique is traditionally done manually, but the prevalence of digital devices makes digital contact tracing possible. Yet, such digital contact tracing is not free from problems. In a world that values privacy, how can we apply this new technology while respecting individual freedom? Furthermore, how can the cryptographic community contribute to ensuring its robustness?

As an example, the privacy preserving contact tracing system developed by Apple and Google does not filter scan results by Bluetooth Low Energy (BLE) outside the transmission range. However, the transmission range of droplets is much shorter compared to the scanning range of BLE (See Section 1.2). Therefore, when all (or even only some) people who got warned try to access hospitals, the yield would be too large for local medical systems to endure.

Also, these systems do not take airborne transmission (different from droplet transmission [1]) into consideration. Droplets may remain in the air for a specific amount of time, while these floating particles may also be infectious during their "lifetime".

Besides, the best case of privacy should be the inability of **anyone** to collect private user information since a secure system would better not rely on trust. However, all current contact tracing systems trust that service

providers will not collect private user information. Hence, these applications intend to prevent unauthorized access, but they do not categorize service providers as "unauthorized entity", while service providers and attackers should be treated similarly from the perspective of privacy protection.

There have been multiple data leakages reports of contact tracing database (*e.g.* [2], [3], [4]). In April 2020, a joint statement [5] signed by 177 scientists from the U.K. indicated that if there is a database, then it must not allow service providers to de-anonymize the data by any means. Blockchain, by its nature, is designed to be pseudonymous, publicly verifiable, non-repudiable, and non-modifiable.

Furthermore, all users of these existing systems allow attackers to fabricate fake contact records or fake user identities.

To solve these problems, an effective contact tracing system will need to be capable of embedding attributes in committed records, having two transmission modes (airborne, droplet), and allowing users conduct tracing independently (because airborne transmission is non-interactive). More importantly, the system needs to maintain confidentiality and integrity — unauthorized access or modification should be prohibited. Furthermore, the system needs to be scalable — to ensure performance even in crowded areas, since handshakes of Bluetooth, backbone of most contact tracing systems, takes considerable time to accomplish.

Hence, we designed our Environmental Adaptive Contact Tracing System (EACT) based on needs described above – capable of dealing with airborne and droplet transmission, providing confidentiality and integrity, and ensures performance even in densely crowded areas. However, contradictions hide beneath these needs, which leave several vulnerabilities of EACT:

1. Embedding positional and environmental data into records might violate confidentiality as these records, containing sensitive personal identifiable information, could potentially be accessed and misused by unauthorized entities while revealing these records.

2. EACT's two transmission modes' threat models are not compatible — users in airborne transmission generate records independently, while droplet transmission is based on user interactions. Therefore, because of decentralized nature of airborne transmission detection, in EACT, issuing one's own records fails to ensure integrity of the system.

These two problems urge us to turn our eyes to a more unified scheme, which is secure for both transmission modes and threat models of which

are compatible to each other. Moreover, we need a scheme to provide better confidentiality to personal identifiable information within positional and environmental data embedded.

The similarity between our needs and a self-issuing decentralized credentials scheme [6] leads us to credentials schemes, typically the *attribute-based* ones (ABC). Note that we consider ABC schemes, *e.g.*, [7], [8], instead of more general anonymous credentials, *e.g.*, [9]–[11] — because the capability of embedding attributes in credentials empowers contact tracing systems to manage environmental factors as attributes. To the best of our knowledge, this approach has seen limited exploration or association with contact tracing in previous works despite its inherent viability[1].

We explain the reason as follows. Recall that an ABC scheme involves issuers, users, and verifiers. In the issuance phase, an issuer grants a credential to a user on the user's attributes. The user can then prove possession (showing) of the credential on their attributes without revealing identities, but they *cannot* prove attributes that are not embedded in their credentials. Hence, by building contact tracing systems atop ABC schemes: (1) users can take environmental factors and local information as attributes; (2) users can issue others credentials on these attributes as contact records; (3) users can anonymously prove their records to potentially malicious verifiers (in contact tracing, medical agencies). It is also convenient to bring the broad spectrum of functionalities in ABC to contact tracing, *e.g.*, selective showing [7], proof of disjoint attributes [8], issuer-hiding [8], [13], delegation [14], *traceability* [15], etc.

Moreover, the security of ABC, *i.e.*, anonymity and unforgeability, can also be adapted to contact tracing (as we will show in Section 5.2.3)[2]. Intuitively, given any honest user's showing of contact records, anonymity prevents other users and medical agencies (even if they collude) from identifying the user or learning anything except what is intentionally revealed by the user. Whereas, unforgeability guarantees that no user can perform a valid showing if she does not possess a corresponding contact record (*i.e.*, a credential issued by another user according to some committed attributes, *e.g.*, environmental factors). The two properties resemble the "(pseudonym and trace) unlinkability" and "integrity" of contact tracing systems proposed in [16] (more discussion in Section 5.2.3).

[1]Silde and Strand [12] proposed a contact tracing system based on anonymous tokens, *i.e.*, an anonymous credential variant that does not support attributes.

[2]Notably, game-based and simulation-based security definitions of contact tracing systems have been proposed in [16], [17], respectively. This work will focus on the game-based ones because we proceed from the perspective of ABC schemes with game-based definitions.

However, existing ABC schemes cannot be utilized directly to build contact tracing systems due to the lack of tracing capability. Note that the traceability in [15] is similar to group signatures, *i.e.*, to revoke the anonymity of *regular users*. In contrast, the traceability of contact systems should enable the issuer of a contact record to be notified whenever the record is being shown. For example, when two users (A and B) have contact, they first exchange contact records by issuing each other a credential based on the contact. Then, when one user (say, user A) is diagnosed and presents her credential issued by her counterparty (user B) to medical agencies, user B should be able to check if the presented credential is issued by herself. If so, user B can confirm that she had close contact with user A. We formalize this functionality as auditability of issuers in the underlying credentials scheme, which we call an auditable ABC.

Despite improvements described above, due to interactive nature of ABC scheme, this approach suffers from scalability problem — as the number of surrounding users increases, performance of the system might deteriorate caused by queued handshakes and security challenges awaiting.

1.2 Related Work

Despite the broad functionalities of ABC schemes, no existing work considers the same traceability (revoking issuer's anonymity) as in contact tracing systems. Regarding auditability of ABC schemes, existing works [18], [19] focused on auditing the credentials back to their holders, *i.e.*, regular users. Instead, our auditability intends to identify issuers. This is because, as shown in Section 4.2.1, modifications are made into ABC syntax so that verifiers cannot identify *the issuer* of shown credentials even if they collude with the original issuer. However, such a property opens the gate of fabricating issuers. Hence, it is crucial to let issuers (or designated third parties chosen by the issuer, called auditors) check if a shown credential originates from the issuer herself.

Theoretically, to compare with existing contact tracing systems, we consider three aspects: (1) security (*i.e.*, anonymity/unlinkability and tracing-soundness/integrity); (2) extensibility (*e.g.*, the capability of handling different transmission modes and environmental factors); (3) efficiency (*e.g.*, requiring BLE handshakes during the recording phase or not). We notice that these requirements may contradict each other (in fact, unlinkability and integrity may also have contradictions [16]). For example, as mentioned above, revealing more data (extensibility) inevitably incurs breaches in anonymity (security). Then, to fix such breaches, we have to rely on heavy mechanisms

that burden the system's efficiency. In the following, we evaluate several cryptographic contact tracing systems to prove our observation.

A simple paradigm of contact tracing utilizes symmetric primitives (*e.g.*, pseudo-random permutations/functions/generators (PRP/PRF/PRG) [20], [21] and hash functions [22]) to generate period-specific keys and pseudonyms (here, the period can be several hours or days). As shown in [16], these systems can achieve *unlinkability* (*i.e.*, period-specific pseudonyms are unlinkable) due to the pseudo-randomness of the underlying building blocks; and *integrity* (*i.e.*, no adversary can forge recorded pseudonyms to trigger users' tracing) due to the pre-image resistance of these primitives; but suffer from *the relay (and replay) attacks* (*i.e.*, the adversary can relay or replay previous records to break integrity) [19] because users cannot tell if a pseudonym has been presented or not (without checking timestamps). The simplicity of this paradigm allows us to construct highly efficient systems. However, the simplicity also prevents us from recording anything but pseudonyms, hence, limiting the extension capability.

One method to enhance the aforementioned systems is to use re-randomizable primitives (*e.g.*, signature schemes) as in [23] and in our work. Concretely, a user obtains a piece of authorized information (in most cases, a signature) from her counterparty (in [24], the counterparty can be regarded as the user herself) during a close contact, and then presents an updated (re-randomized) signature to medical agencies when she is diagnosed. This approach achieves *unlinkability* from the re-randomizability of the building blocks; and *integrity* from unforgeability. Previous works [23], [24] consider semi-honest verifiers (medical agencies) who only approve valid signatures to extend the bulletin board. Hence, they can prevent *the relay and replay attacks* by requiring additionally the freshness of signatures.

Inherently shown in [21], [22], where authors present various constructions of contact tracing systems with varying levels of security and efficiency, the trade-off among these requirements (security, extensibility, and efficiency) prevents us from finding the ultimate solution for contact tracing. We argue that our system is *secure* despite handling more sensitive data; is *extensible* to tackle new epidemiology findings; and is *efficient enough* to be implemented in real life.

On the practical side of contact tracing systems, we evaluated several existing works and concluded that these systems all have similar deficiencies (See Table 1.1). Here, "Precision" means whether to filter out false positives (get warned but not in danger) from the scanned results; "Demo" means whether a software demonstration exists; "Confid." means confidentiality; "Integrity" means whether someone can fabricate non-existent records or identities; "Scalability" means whether the system can be deployed in

large-scale use-cases, *e.g.* densely populated scenarios or heavy calculations; "Flooding" means whether the system can prevent users from flooding the system with either legitimate records or illegitimate records. Additionally, "△" means this property exists, without being fully applied, which will be explained in the following paragraphs or sections, and "N/A" means this property is not considered as a topic by the researcher(s) in their corresponding research(es).

One of the major problems is the lack of precision. Based on our research, there is currently no contact tracing system focusing on reducing the false-positive rate of the scanning. For example, the first contact tracing system for COVID-19 is the privacy preserving Contact Tracing system designed by Apple and Google ("GAEN") [20], using Bluetooth Low Energy (BLE) to search for surrounding devices without excluding results outside effect range of droplet transmission. Although the range of BLE varies by device, most of them are around 50–100 meters. In contrast, based on epidemiological research, the distance of droplet transmission is typically between 0–20 meters approximately [25], and can be affected by environmental factors. This means that individuals scanned outside the effective range would be false-positive cases.

Also, **all** current contact tracing systems ignore the fact that people do not have to contact others closely to become "close contacts" because, according to Chen [26], droplets ejected from the human body will float for a specific amount of time. Therefore, these systems do not consider the duration time for the infection tracking.

More importantly, personal privacy is not thoroughly preserved in current contact tracing systems. In this regard, GAEN is a representative model for typical contact tracing. Users hold a periodically re-generated temporary key (TEK) and generate other keys using a key derivation function. Keys derived from TEK are used to encrypt the payload with AES [27]. After being diagnosed positive, encrypted payload and TEKs will be uploaded for normal users to trace their contacts. However, for GAEN, its cryptographic design based on AES and random key generation cannot guarantee integrity, *i.e.*, fabricating never-happened meetings or fake identities. It is worth mentioning that all current contact tracing systems do not prevent such fabrication, which would harm data integrity of contact tracing systems.

Although all of the contact tracing systems we reviewed emphasize confidentiality, which means to prevent unauthorized access, most of them have not clarified the problem about who shall be categorized as "unauthorized entities". For example, the contact tracing application designed by N.C.S.C. of the U.K. [28] holds a public key pair for each user and stores the private key in the server. Hence, the service providers can access user data

using the private key while preventing unauthorized entities from accessing it. However, from the perspective of privacy protection, service providers and unauthorized entities show no difference — both can act as threats to users' data security, and it is not possible to prevent this from happening with traditional public-key approaches.

Systems proposed by Kim et al. [29] and An et al. [30] use functional encryption and homomorphic encryption to circumvent the aforementioned issues caused by traditional encryption. However, neither of the primitive is practical in the real world. Moreover, these systems even collect more personally identifiable information than it is necessary, *e.g.*, credit card information.

There are also articles mainly focused on engineering issues, such as [31]–[33]. Unlike this work which proposes new generic cryptographic frameworks for contact tracing systems, these studies focus on implementing existing technologies and building concrete applications. Moreover, they also suffer from the issues mentioned at the beginning of this section. For example, [32] uses a different approach that involves IoT, and only traces diagnosed users. Whereas [31] uses geolocation data in contact tracing but only presents possible methods to achieve privacy without analysis.

It should be specially notified that, the term "airborne-based" used in [33] refers to droplet transmission, which is **not** the same as the formal definition of airborne transmission, defined by CDC [34], as we describes in Section 2.2.

Finally, we believe contact-tracing applications should follow the ethical guidelines proposed in [35]. However, as our goal focuses more on proposing a general cryptographic framework for contact-tracing systems rather than coding an application, we argue that our work does not violate the spirit of the guideline.

1.3 Our Contributions

Our contribution can be divided into two parts: Environmental Adaptive Contact Tracing System, and Auditable Attribute-Based Credentials Scheme. The reason why we choose to provide two approaches is because they both have advantages and disadvantages due to the difference in their underlying cryptographic scheme: decentralized anonymous credential and attribute-based credential. Environmental Adaptive Contact Tracing System has better performance in large-scale implement, but lacks a robust and rigorous cryptographic scheme and assumes a honest revealer, as mentioned in Section 1.1. On the other hand, Attribute-Based Credential's cryptographic scheme is robust, rigorous, and secure, but may suffer from performance

Table 1.1: Comparison with Related Works

C.T. Systems			Prec.	Demo	Confid.	Integ.	Scal.	Flooding
IRL	**Gov.**	UK [28]	X	✓	X	X	✓	X
		JP [36]	X	✓	X	X	✓	X
		China	X	✓	X	X	N/A	✓
	Other	GAEN [20]	X	✓	X	X	✓	X
Research	Liu et al. [23]		X	△	✓	X	X	X
	Canetti et al. [22]		X	X	✓	X	✓	X
	Kim et al. [29]		X	X	✓	X	X	X
	An et al. [30]		X	✓	✓	X	X	X
	José et al. [31]		X	X	N/A	N/A	N/A	N/A
	Manjul et al. [33]		X	✓	X	X	✓	X
	Lalit et al. [32]		X	✓	N/A	N/A	N/A	N/A
	Our Work		✓	✓	✓	✓	✓	✓

C.T. = Contact Tracing; Prec. = Precision; Confid. = Confidentiality; Integ. = Integrity; Scal. = Scalability; IRL = In Real Life

issue in large-scale implement because of its interactive nature.

1.3.1 Environmental Adaptive Contact Tracing System

A New Cryptographic Framework To solve the dilemma between anonymity and integrity, we propose a cryptographic framework for contact tracing and provide a construction based on a public key rerandomizable BLS signature scheme. The scheme is based on the original BLS signature [37] and the updatable public key mechanism [38], and it satisfies public key rerandomizability and signature aggregability. The scheme enables users to update their identities to preserve privacy while keeping the integrity with signatures to deny malicious messages. The result is that users cannot fabricate fake meeting records with other users, undermining the integrity of the bulletin board. Thus, we utilize the signature scheme as our main building block for the contact tracing system and analyze the computational complexity of the resulting scheme.

We also clarify the integration between our framework and a bulletin board, which can be implemented with a blockchain. Finally, we consider several practical issues of our blockchain-based bulletin-board, such as privacy, scalability, and storage overhead.

A New Contact Tracing System Based on our cryptographic framework described above, we propose a new contact tracing system, which uses environmental factors affecting particle dynamics, lifetime to filter scanning results, and lower the false positive rate, to prove the practicality of our cryptographic framework. The dynamics determines the coordinates in which users should be informed, and lifetime determines the time interval during which users should be informed. Our cryptographic framework is a variant of credential systems, and thus it can embed these arbitrary environment factors inherently. Moreover, according to new requirements (airborne transmission) in contact tracing, our system includes a new contact-tracing scheme called discrete real-time tracking. See Figure 5.1 for the detailed design.

Our system uses a perfect hiding commitment scheme to prevent multiple identities for the same user. Users need to commit their unique initial public keys to authorized organizations. Then, doctors need to check 1) whether the user knows the corresponding secret key and 2) the commitment is computed correctly from the public key and commitment key, without knowing either public, secret, or commitment keys. Hence, doctors can verify whether the submitted contact records are signed by the committed public key.

We also developed a demonstration on Android OS to show the efficiency of our work on mobile devices.

1.3.2 Auditable Attribute-Based Credentials Scheme.

In order to build an auditable ABC scheme, we first propose a cryptographic tool called the "auditable public keys (APK)" mechanism, which extends the updatable public key given in [38]. The APK embeds extra structure in the secret and public key pair with a new auditing key. The structure will be preserved even after updating the public key and can be verified (we call it audit) by the auditing key. That is, a participant who holds the auditing key corresponding to some key pair can audit if a given public key is updated from the corresponding public key without knowing the secret randomness in the update algorithm. Similar to the updatable public key, our APK can be used as a plug-in for many different cryptographic primitives (We show two examples in Chapter 4).

Next, we adapt APK to the existing ABC scheme [7] and define the formal syntax of our auditable ABC. We show a concrete construction for the APK mechanism based on the matrix Diffie-Hellman problem over matrix distributions [39], [40]. We prove that our APK construction can be inserted into the structure-preserving signatures on equivalence classes (SPS-EQ) scheme [8] without breaking the security of the original SPS-EQ (though incurring a slight reduction loss). By employing our modified SPS-EQ, a set-commitment

scheme from [7], and a zero-knowledge proofs of knowledge protocol [41], we present a construction for the auditable ABC scheme.

Finally, we refine the EACT framework [24] and provide a construction based on our auditable ABC scheme. Hence, we can unify the tracing process of the conventional Bluetooth Low Energy (BLE)-based setting for droplet mode and their discrete-location-tracing setting (DLT) for airborne mode. Then, we argue that the security of the refined EACT can be derived from our auditable ABC scheme but requires sufficient adaptions, *e.g.*, in contact tracing, the verifier of credentials may be malicious and approve falsely shown credentials. We explain these adaptions and finally show an implementation (in Section 5.2.4) of our construction on real-life Android devices to demonstrate practicality.

Our contributions regarding auditable ABC scheme are threefold: (1) we propose an APK mechanism that can be used as a plug-in tool for many cryptographic primitives; (2) we propose an auditable ABC scheme that inherits auditability from APK. Then, we show concrete constructions for APK and the auditable ABC scheme; (3) we refine and construct the EACT framework [24] based on credentials schemes. We also provide formal security definitions and implement the construction. Additionally, we add algorithms to jPBC library [42] to support matrix-based bilinear pairing operations during implementation.

Chapter 2

Preliminaries

2.1 Notations

Throughout this thesis, we use λ for the security parameter and $\mathsf{negl}(\cdot)$ for the negligible function. PPT is short for probabilistic polynomial time. For an integer q, $[q]$ denotes the set $\{1, \ldots, q\}$. Given a set $\mathbb{X}$, $x \xleftarrow{\$} \mathbb{X}$ denotes that x is randomly and uniformly sampled from $\mathbb{X}$; whereas, for an algorithm Alg, $x \leftarrow \mathsf{Alg}$ denotes that x is assigned the output of an algorithm Alg on fresh randomness. Let $\mathsf{Alg}_1, \mathsf{Alg}_2$ be two algorithms, $\langle \mathsf{Alg}_1, \mathsf{Alg}_2 \rangle$ denotes a potentially interactive protocol between the two algorithms. Let H denote a collision-free hash function. For an additive group $\mathbb{G}$, $\mathbb{G}^*$ denotes $\mathbb{G} \setminus \{0_{\mathbb{G}}\}$. For a set $\mathbb{X} \subseteq \mathbb{Z}_p$, we refer to a monic polynomial of order $|\mathbb{X}|$ defined over $\mathbb{Z}_p[X]$, $\mathsf{Ch}_{\mathbb{X}}(X) \triangleq \Pi_{x \in \mathbb{X}}(X - x) = \sum_{i=0}^{|\mathbb{X}|} c_i \cdot X^i$ as $\mathbb{X}$'s characteristic polynomial.

We denote the asymmetric bilinear group generator as $\mathsf{BG} \leftarrow \mathsf{BGGen}(1^\lambda)$ where $\mathsf{BG} \triangleq (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, P_1, P_2, e)$. Here, $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ are additive cyclic groups of prime order p with $\lceil \log_2 p \rceil = \lambda$, P_1, P_2 are generators of $\mathbb{G}_1, \mathbb{G}_2$, and $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ is a type-3, *i.e.*, efficiently computable non-degenerate bilinear map with no efficiently computable isomorphism between $\mathbb{G}_1$ and $\mathbb{G}_2$. For an element $a \in \mathbb{Z}_p$ and $i \in \{1, 2\}$, $[a]_i$ denotes $aP_i \in \mathbb{G}_i$ as the representation of a in group $\mathbb{G}_i$. As mentioned in [8], for vectors or matrices $\mathbf{A}, \mathbf{B}$, the bilinear map e computes $e([\mathbf{A}]_1, [\mathbf{B}]_2) = [\mathbf{AB}]_T \in \mathbb{G}_T$.

Additionally, Table 2.1 shows the notations used when introducing the impact of environmental factors in contact tracing systems .

Table 2.1: Table of Notations

Notations	Definitions	Notations	Definitions
B	Spalding mass transfer number	C	Circle
D	Binary diffusion coefficient	F^G	Gravitational force
MP	Measured power	RH	Relative humidity
T	Temperature	Y	Mass fraction
d	Estimated distance (by RSSI)	d_0	Initial droplet size
dr	Shift of the coordinate at dt	dt	Discrete time point
f	Liquid water	L	Location-stamp
m	Mass of the droplet	R	Radius of the droplet
t	Droplet Lifetime	t_r	Record time
u_0	Peak value of airflow velocity	v_0	Initial ejection velocity
v_t	Downward terminal velocity	w, s	Water at droplet surface
w, ∞	Ambient water	γ	Path loss exponent
η	Dynamic viscosity	λ	Drag coefficient
ξ	Noise term	ρ	Density

2.2 Environmental Factors in Epidemiology

Environmental factors play an important role in contact tracing, thus we evaluate and incorporate several epidemiological studies in our system.

The three ways of transmission for respiratory viruses, as suggested by CDC [34], are: contact transmission, droplet transmission, and airborne transmission.

In this thesis, according to the definition [34], "contact transmission" means direct physical contact with infectees; "droplet transmission" refers to the infections due to viruses ejected with droplets by sneezes or coughs; "airborne transmission" means infections caused by floating liquid drops carrying viruses suspended in the air.

Assumptions

We need to make some assumptions to exclude immeasurable or uncontrollable factors.

Excluding Contact Transmission Without extra wearable devices or sensors, cellphones are not able to detect direct contacts (*i.e.*, touching), and contact tracing systems should solely rely on the most widely-used portable devices (cellphones, in this case).

Enclosed Indoor Environments For indoor environments, it is true that ventilation is crucial for epidemic preparedness [1]. However, it is hardly possible for cellphones to sense the ventilation status of the current room. Hence, we need to decide whether to assume **all** indoor cases are enclosed or ventilated. If the environment is ventilated, the airflow organization will become more complex. In that case, we need to either simulate particle dynamics, which is impractical for mobile devices, or to assume indoor airflow organization models. Moreover, since an enclosed environment is more dangerous than a ventilated environment [1], as airflow can dilute viruses in the air and thus reduce their infectivity, we prefer a worst-case assumption. Consequently, we assume that all indoor environments are enclosed in our system, including balconies and opened windows.

End of "Lifetime" $\neq$ Death of Virus As the researchers indicate in their study about lifetime [26], lifetime means vaporization. However, vaporization of the droplet that carries and transmits the virus does not necessarily mean the virus' death. The lifetime of pathogens differs from one disease to another and depends on surfaces' materials, which are not measurable by cellphones. Therefore, we will not consider the real lifetime of pathogens.

Overview.

Under the assumptions described above, three major measurable factors are temperature T, relative humidity RH, and air velocity. There are two types of transmission, transmission by droplets and by airborne. All researchers mentioned in this section have provided experimental data, respectively, with values of environmental factors staying in reasonable ranges as real-world scenarios may have.

This section is represented by the box "Discrete Tracing" and "BLE Scanning" in the design diagram (Figure 5.1 in Section 5.1).

We must emphasize that we do not intend to pay attention to the accuracy of epidemiological data and equations mentioned in this section. We believe that such researches are subject to future works of physicists and epidemiologists. Existing contact tracing approaches completely lacked consideration of such fields, and our focus here is to implement epidemiological studies in contact tracing.

Droplet Transmission. According to Das et al.'s study, the influence of temperature on space-time evolution of droplets' motion, from 0 °C to 40 °C, is quite limited (about 10%) [25]. Therefore, we will not take temperature into consideration of droplet transmission.

Instead, the sizes of the particles and airflow play significant roles in droplet transmission. Das et al.'s data is based on the Langevin equation [25], *i.e.*, Equation (2.1).

$$\frac{4}{3}\pi R^3 \rho \frac{d^2 r_i}{dt^2} = -\lambda \frac{dr}{dt} + \xi(t) + F^G \tag{2.1}$$

In order to measure the size (*i.e.* radius) of the droplets that determines the mass m (i.e. $4\pi R^3\rho/3$), Han et al.'s research provides statistical data of diameter of droplets, measured at 100 ms after the ejection from the human body [43].

Together with the Langevin Equation, the impact of airflow velocity also needs to be considered, as shown in Equation (2.2). The impact of airflow velocity needs to be evaluated with the downward terminal velocity, v_t [25].

$$v_t = \frac{2R^2(\rho_{droplet} - \rho_{air})g}{9\eta} \tag{2.2}$$

Das et al. have explained thoroughly how to solve Equation 2.1 and also the effects of airflow velocity in their paper, and thus we will not cover these details in this section. Since the authors claim that their experiment data matches calculated assumptions perfectly [25], both can be implemented in our system.

We use 0.5m/s air velocity as the threshold to determine whether the current outdoor environment is ventilated enough, as suggested by NFPA in laboratory ventilation standards [44]. If air velocity is faster than the threshold, transmission radius will be set to 0. This rule also applies to lifetime in airborne transmission, as discussed in the following section.

Airborne Transmission. Chen's work provides an important equation in his research as Equation (2.3) shows for determining lifetime of the particles. Spalding mass transfer number B is computed from mass fraction $Y_{w,s}$ and $Y_{w,\infty}$. In this equation, temperature has impact on diffusion coefficient D and relative humidity determines the ratio between $Y_{w,s}$ and $Y_{w,\infty}$.

$$t = \frac{\rho_f d_0^2}{8\rho_w D_w (ln(1+B))},\ where\ B = \frac{Y_{w,s} - Y_{w,\infty}}{1 - Y_{w,s}} \tag{2.3}$$

Using the equation, data in Chen's research include particles of 1–1000 µm diameter (d_0), with relative humidity RH from 0% to 100% and temperature at 20 °C and 30 °C. Both data and equation can be implemented in our system to predict lifetime.

Take note that when relative humidity approaches approximately 55.7%[1], lifetime approaches infinity, as $Y_{w,\infty}$ approaches $Y_{w,s}$, shown in Equation (2.3).

Measurement.

Table 2.2: Regulations about Temperature and Relative Humidity

Nation	Temperature (in Celsius)	RH	Reference
Japan	17–28	40%–70%	[45]
U.S.A.	20–24	20%–60%	[46]
U.K.	$\geq$16 or $\geq$13	N/A	[47]
France	19	N/A	[48]
China	26–28 or 16–20	40%–65%	[49]

It is crucial to distinguish indoor environments from outdoor environments for measurements, as most countries have laws or regulations on temperature/humidity in public areas. Online mapping services, such as Google Maps, are capable of categorizing place types (*e.g.*, bank, city hall, zoo, and etc.). According to users' location, these services can be used to infer whether the user is staying indoor or outdoor, public buildings or residential buildings. Temperature/humidity indoor can be estimated from either these laws or regulations, assuming that all public buildings keep following these laws or regulations, or users' personal favorite ambient temperature/humidity at home. Again, we assume all indoor areas are enclosed, as stated in Section 2.2.

In Table 2.2, we take five countries as examples, Japan, United States of America, United Kingdom, French Republic, and People's Republic of China. By embedding these data (along with other countries) into the system in advance, our system can operate in different nations or regions.

Besides, outdoor temperature/humidity/airflow is easy to measure — using data from government-held organizations such as the department of meteorology or from commercial weather service providers.

Some mobile phones are also equipped with hygrometers, or humidity sensors, which can measure ambient relative humidity and should be more precise than estimation based on regulations.

[1]Derived and explained in [26].

Chapter 3

Cryptographic Assumptions and Building Blocks

This chapter introduces cryptographic assumptions and building blocks that we will utilize to construct our results.

3.1 Assumptions

Recall the type-3 bilinear map given in Chapter 2.1 with respect to the generator BGGen: $\mathsf{BG} \leftarrow \mathsf{BGGen}(1^\lambda)$ such that $\mathsf{BG} \stackrel{\Delta}{=} (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, P_1, P_2, e)$ where: $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ are additive cyclic groups of prime order p with $\lceil \log_2 p \rceil = \lambda$; P_1, P_2 are generators of $\mathbb{G}_1, \mathbb{G}_2$; and $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ is a type-3, *i.e.*, efficiently computable non-degenerate bilinear map with no efficiently computable isomorphism between $\mathbb{G}_1$ and $\mathbb{G}_2$. We assume the q-co-discrete-logarithm (q-co-DL) assumption, the Computational co-Diffie-Hellman (co-CDH) [50], and the Diffie-Hellman (DDH) assumption hold over this bilinear group. We further assume the following assumptions hold over matrix distribution: the matrix decisional Diffie-Hellman (MDDH) assumption [39] and the kernel matrix Diffie-Hellman (KerMDH) assumption [40]. The formal definitions are presented as follows.

Definition 1 (q-co-DL Assumption) *The q-co-DL assumption holds for BGGen, if for all $\mathsf{BG} \leftarrow \mathsf{BGGen}(1^\lambda), a \stackrel{\$}{\leftarrow} \mathbb{Z}_p$ and all PPT adversary $\mathcal{A}$, the following advantage is negligible of λ:*

$$\Pr[a' \leftarrow \mathcal{A}(\mathsf{BG}, ([a^j]_1, [a^j]_2)_{j \in [q]} : a' = a].$$

Definition 2 (co-CDH) *The co-CDH assumption holds for BGGen, if for all $\mathsf{BG} \leftarrow \mathsf{BGGen}(1^\lambda), x, y \stackrel{\$}{\leftarrow} \mathbb{Z}_p$ and all PPT adversary $\mathcal{A}$, the following*

advantage is negligible of λ*:*

$$\Pr\left[xyP_1 \leftarrow \mathcal{A}(P_1, P_2, xP_1, yP_2)\right].$$

Definition 3 (DDH Assumption) *The DDH assumption holds in* $\mathbb{G}_i \in \mathsf{BG}$ *where* $i \in \{1,2\}$ *for* BGGen*, if for all* $\mathsf{BG} \leftarrow \mathsf{BGGen}(1^\lambda), x, y, z \xleftarrow{\$} \mathbb{Z}_p$ *and all PPT adversary* $\mathcal{A}$*, the following advantage is negligible of* λ*:*

$$|\Pr[\mathcal{A}(\mathsf{BG}, xP_i, yP_i, xyP_i) = 1] - \Pr[\mathcal{A}(\mathsf{BG}, xP_i, yP_i, zP_i) = 1]|.$$

Particularly, we need the DDH assumption to hold for both $i \in \{1,2\}$, *i.e.*, the Symmetric eXternal Diffie-Hellman (SXDH) assumption [51], in our DDH-based auditable public keys mechanism (Construction 2), we also show its definition in the following.

Definition 4 (SXDH assumption) *We say the SXDH assumption holds for* BGGen*, if the DDH assumption (Definition 3) holds in* $\mathbb{G}_1$ and $\mathbb{G}_2$*.*

Next, we define the matrix distribution and show assumptions with respect to a given matrix distribution $\mathcal{D}_{l,k}$.

Definition 5 (Matrix Distribution) *Let* $l, k \in \mathbb{N}$ *with* $l > k$*.* $\mathcal{D}_{l,k}$ *is a matrix distribution that outputs matrices in* $\mathbb{Z}_p^{l\times k}$ *of full rank* k *in polynomial time. We further denote* $\mathcal{D}_k \triangleq \mathcal{D}_{k+1,k}$*.*

Definition 6 ($\mathcal{D}_{l,k}$-MDDH Assumption) *The* $\mathcal{D}_{l,k}$*-MDDH assumption holds in group* $\mathbb{G}_i \in \mathsf{BG}$ *where* $i \in \{1,2,T\}$ *relative to* BGGen*, if for all* $\mathsf{BG} \leftarrow \mathsf{BGGen}(1^\lambda), \mathbf{A} \xleftarrow{\$} \mathcal{D}_{l,k}, \mathbf{w} \xleftarrow{\$} \mathbb{Z}_p^k, \mathbf{u} \xleftarrow{\$} \mathbb{Z}_p^l$ *and all PPT adversary* $\mathcal{A}$*, the following advantage is negligible of* λ*:*

$$\mathsf{Adv}^{MDDH}_{\mathcal{D}_{l,k},\mathbb{G}_i} = |\Pr[\mathcal{A}(\mathsf{BG}, [\mathbf{A}]_i, [\mathbf{Aw}]_i) = 1] - \Pr[\mathcal{A}(\mathsf{BG}, [\mathbf{A}]_i, [\mathbf{u}]_i) = 1]|.$$

Definition 7 ($\mathcal{D}_{l,k}$-KerMDH Assumption) *The* $\mathcal{D}_{l,k}$*-KerMDH assumption holds in group* $\mathbb{G}_i \in \mathsf{BG}$ *where* $i \in \{1,2\}$ *relative to* BGGen*, if for all* $\mathsf{BG} \leftarrow \mathsf{BGGen}(1^\lambda), \mathbf{A} \xleftarrow{\$} \mathcal{D}_{l,k}$ *and all PPT adversary* $\mathcal{A}$*, the following advantage is negligible of* λ*:*

$$\Pr[[\mathbf{x}]_{3-i} \leftarrow \mathcal{A}(\mathsf{BG}, [\mathbf{A}]_i) : e([\mathbf{x}^\top]_{3-i}, [\mathbf{A}]_i) = [\mathbf{0}]_T \wedge \mathbf{x} \neq \mathbf{0}].$$

3.2 Black-Box Building Blocks

Next, we introduce the syntax and security definitions of several cryptographic primitives that will be used as a black-box in this work.

1. A digital signature scheme $\mathsf{SIG} \triangleq (\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$ that satisfies correctness and existentially unforgeability under adaptive chosen-message attacks (EUF-CMA) [52];
2. A set-commitment scheme that is correct, binding, subset-sound, and hiding [7]: $\mathsf{SC} \triangleq (\mathsf{Setup}, \mathsf{Commit}, \mathsf{Reveal}, \mathsf{OpenSubset}, \mathsf{VerifySubset})$;
3. A zero-knowledge proofs of knowledge (ZKPoK) protocol Π that satisfies completeness, perfect zero-knowledge, and knowledge-soundness [41].

3.2.1 Digital Signature Schemes

The tuple of algorithm in a digital signature scheme $\mathsf{SIG} = (\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$ works as follows.

- $\mathsf{KeyGen}(1^\lambda)$ takes as input the security parameter λ and outputs a key pair $(\mathsf{sk}, \mathsf{pk})$;
- $\mathsf{Sign}(\mathsf{sk}, \mathrm{m})$ takes as input the secret key sk and a message m. It outputs a signature σ on m under sk;
- $\mathsf{Verify}(\mathsf{pk}, \mathrm{m}, \sigma)$ takes as input the public key pk, the message m and the signature σ. It outputs 1 if the signature is valid and 0 otherwise.

This work employs a signature scheme that satisfies correctness and the existential unforgeability under adaptive chosen message attacks (EUF-CMA) [52].

Definition 8 (Correctness) *A signature scheme satisfies correctness, if for any $\lambda > 0$ and $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$:*

$$\Pr\left[\mathsf{Verify}(\mathsf{pk}, m, \mathsf{Sign}(\mathsf{sk}, m)) = 1\right] = 1.$$

Definition 9 (EUF-CMA) *A signature scheme satisfies EUF-CMA, if for any adversary that has access to a signing oracle $\mathcal{O}_{\mathsf{Sign}}(\mathsf{sk}, \cdot)$ with queries $m \in \mathsf{Q}$, the following probability is negligible of λ for any $\lambda > 0$ and $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$:*

$$\Pr\left[(m^*, \sigma^*) \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{Sign}}}(\mathsf{pk}) : m^* \notin \mathsf{Q} \wedge \mathsf{Verify}(m^*, \sigma^*, \mathsf{pk}) = 1\right].$$

3.2.2 Set-Commitment Scheme

The set-commitment $\mathsf{SC} \triangleq (\mathsf{Setup}, \mathsf{Commit}, \mathsf{Reveal}, \mathsf{OpenSubset}, \mathsf{VerifySubset})$ is defined as:

- $\mathsf{Setup}(1^\lambda, q)$ takes as input the security parameter λ and the size upper bound q of committed sets. It outputs the public parameter pp;
- $\mathsf{Commit}(\mathsf{pp}, \mathbb{X})$ takes as input pp and a non-empty set $\mathbb{X}$. It outputs a commitment comm and opening information O;
- $\mathsf{Reveal}(\mathsf{pp}, \mathsf{comm}, \mathbb{X}, O)$ takes as input pp, a commitment comm, a set $\mathbb{X}$, and opening information O. It outputs 1 if O is a valid opening of comm to $\mathbb{X}$; and 0 otherwise;
- $\mathsf{OpenSubset}(\mathsf{pp}, \mathsf{comm}, \mathbb{X}, O, D)$ takes as input pp, a commitment comm, a set $\mathbb{X}$, opening information O, and a non-empty set D. If $D \subseteq \mathbb{X}$, it outputs a witness W that shows D being a subset of the set $\mathbb{X}$ committed in comm;
- $\mathsf{VerifySubset}(\mathsf{pp}, \mathsf{comm}, D, W)$ takes as input pp, a commitment comm, a non-empty set D, and a witness W. It outputs 1 if W is a witness that shows D being a subset of the set $\mathbb{X}$ committed in comm; and 0 otherwise.

Definition 10 (Correctness) *A set-commitment scheme satisfies correctness, if for any $\lambda, q \geq 0$, any $\mathbb{X}$ and non-empty $D \subseteq \mathbb{X}$, and any $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda, q)$:*

$$\Pr\left[(\mathsf{comm}, O) \leftarrow \mathsf{Commit}(\mathsf{pp}, \mathbb{X}) : \mathsf{Reveal}(\mathsf{pp}, \mathsf{comm}, \mathbb{X}, O) = 1\right] = 1 \wedge$$
$$\Pr\left[\begin{array}{l} (\mathsf{comm}, O) \leftarrow \mathsf{Commit}(\mathsf{pp}, \mathbb{X}) : \mathsf{VerifySubset}(\mathsf{pp}, \mathsf{comm}, D, W) = 1 \\ W \leftarrow \mathsf{OpenSubset}(\mathsf{pp}, \mathsf{comm}, \mathbb{X}, O, D) \end{array}\right] = 1.$$

Definition 11 (Binding) *A set-commitment scheme is binding, if for any $\lambda, q \geq 0$, any $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda, q)$, and all PPT adversary $\mathcal{A}$, the following probability is negligible of λ:*

$$\Pr\left[\begin{array}{r} \mathsf{Reveal}(\mathsf{pp}, \mathsf{comm}, \mathbb{X}, O) = 1 \wedge \\ (\mathsf{comm}, \mathbb{X}, O, \mathbb{X}', O') \leftarrow \mathcal{A}(\mathsf{pp}) \quad : \mathsf{Reveal}(\mathsf{pp}, \mathsf{comm}, \mathbb{X}', O') = 1 \wedge \\ \mathbb{X} \neq \mathbb{X}' \end{array}\right].$$

Definition 12 (Subset-Soundness) *We say a set-commitment scheme is subset-sound, if for any $\lambda, q \geq 0$, any $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda, q)$, and all PPT adversary $\mathcal{A}$, the following probability is negligible of λ:*

$$\Pr\left[\begin{array}{r} \mathsf{Reveal}(\mathsf{pp}, \mathsf{comm}, \mathbb{X}, O) = 1 \wedge \\ (\mathsf{comm}, \mathbb{X}, O, D, W) \leftarrow \mathcal{A}(\mathsf{pp}) : \mathsf{VerifySubset}(\mathsf{pp}, \mathsf{comm}, D, W) = 1 \wedge \\ D \not\subseteq \mathbb{X} \end{array}\right].$$

Definition 13 (Hiding) *A set-commitment scheme is hiding, if for any* $\lambda, q \geq 0$*, any* $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda, q)$*, and all PPT adversary* $\mathcal{A}$ *with accessibility to an* $\mathcal{O}_{\mathsf{OpenSubset}}$ *oracle that opens the challenge commitment to any subset in the* intersection *of the two committed sets, the following probability is less than* $\mathsf{negl}(\lambda)$*.*

$$\left| \Pr \left[\begin{array}{ll} b \xleftarrow{\$} \{0,1\}; (\mathbb{X}_0, \mathbb{X}_1, \mathsf{st}) \leftarrow \mathcal{A}(\mathsf{pp}) & \\ (\mathsf{comm}, O) \leftarrow \mathsf{Commit}(\mathsf{pp}, \mathbb{X}_b); & : b^* = b \\ b^* \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{OpenSubset}}(\mathsf{pp},\mathsf{comm},\mathbb{X}_b,O,\cdot\subseteq\mathbb{X}_0\cap\mathbb{X}_1)}(\mathsf{st},\mathsf{comm}) & \end{array} \right] - \frac{1}{2} \right|$$

3.2.3 Zero-Knowledge Proofs of Knowledge (ZKPoK) Protocol

We follow the generic definition of ZKPoK given in [7]. Let $\mathcal{L}_\mathcal{R} = \{x : \exists w, \text{s.t.}, (x,w) \in \mathcal{R}\} \subseteq \{0,1\}^*$ be a formal language with respect to a binary, polynomial-time (witness) relation $\mathcal{R} \subseteq \{0,1\}^* \times \{0,1\}^*$, *i.e.*, when given a polynomial-size (of $|x|$) witness w that certifies $(x,w) \in \mathcal{R}$, $x \in \mathcal{L}_\mathcal{R}$ can be decided in the polynomial time of $|x|$. Denote the interactive protocol between a (potentially unbounded) prover $\mathcal{P}$ and a PPT verifier $\mathcal{V}$ with $\langle \cdot, b \rangle \leftarrow \langle \mathcal{P}(\cdot,\cdot), \mathcal{V}(\cdot) \rangle$ where $b \in \{0,1\}$. Here, $b = 1$ indicates that $\mathcal{V}$ accepts the conversation with $\mathcal{P}$; and $b = 0$ indicates $\mathcal{V}$ rejects. If an interactive protocol satisfies completeness, perfect zero-knowledge, and knowledge-soundness [41], we call it a ZKPoK.

Definition 14 (Completeness) *An interactive protocol* $\langle \mathcal{P}, \mathcal{V} \rangle$ *for a relation* $\mathcal{R}$ *satisfies completeness, if for any* $x \in \mathcal{L}_\mathcal{R}$ *and* w *s.t.,* $(x,w) \in \mathcal{R}$*:*

$$\Pr[\langle \cdot, 1 \rangle \leftarrow \langle \mathcal{P}(x,w), \mathcal{V}(x) \rangle] = 1.$$

Definition 15 ((Perfect) Zero-Knowledge) *An interactive protocol* $\langle \mathcal{P}, \mathcal{V} \rangle$ *for a relation* $\mathcal{R}$ *is (perfect) zero-knowledge, if for any PPT adversary* $\mathcal{A}$ *there exists a simulator* $\mathcal{S}$ *such that* $\{\mathcal{S}^\mathcal{A}(x)\}_{x \in \mathcal{L}_\mathcal{R}} \approx \{\langle \mathcal{P}(x,w), \mathcal{A}(x) \rangle\}_{(x,w) \in \mathcal{R}}$ *where* $\mathcal{S}$ *is PPT and* $\langle \mathcal{P}(\cdot,\cdot), \mathcal{A}(\cdot) \rangle$ *denotes the transcript of the interaction between* $\mathcal{P}$ *and* $\mathcal{A}$*, and "*$\approx$*" denotes (perfect) indistinguishability.*

Definition 16 (Knowledge-Soundness) *An interactive protocol* $\langle \mathcal{P}, \mathcal{V} \rangle$ *is proofs of knowledge (PoK) relative to an NP relation* $\mathcal{R}$*, if for any potentially unbounded adversarial prover* $\mathcal{A}$ *accepted by* $\mathcal{V}$ *on* x*, i.e.,* $\langle \cdot, 1 \rangle \leftarrow \langle \mathcal{A}(x), \mathcal{V}(x) \rangle$*, with probability greater than* ϵ*, then there exists a PPT knowledge extractor* $\mathcal{K}^\mathcal{A}(x)$ *(denoting that the extractor has rewinding black-box access to* $\mathcal{A}$*) that can output a value* w *satisfying* $(x,w) \in \mathcal{R}$ *with probability polynomial of* ϵ*.*

3.3 Base Primitives

Additionally, we will present another two primitives, *i.e.*, the B(G)LS signature scheme [37], [50] and the state-of-the-art structure-preserving signatures on equivalence classes (SPS-EQ) scheme [8]. These primitives will be significantly modified in our constructions, and the modified versions are considered our contributions.

3.3.1 The B(G)LS Signature Scheme

Proposed by Boneh et al. [50], the BLS is a signature scheme that enables signature aggregation without requiring signers' secret keys. The aggregation suffers from the "rogue public key" attack [37]. The following definition adopts the distinct message set solution from [37]. The modified scheme is usually called the BGLS signature, however, we use "BLS" to keep consistency. Let $\mathsf{H}_0 : \mathbb{M} \to \mathbb{G}_1$ be a hash function where $\mathbb{M}$ is the message space.

The BLS signature scheme involves the following algorithms:

- $\mathsf{KeyGen}(1^\lambda) \to (\mathsf{sk}, \mathsf{pk})$. With a security parameter λ as input, KeyGen chooses a random $sk \xleftarrow{\$} \mathbb{Z}_q$ and outputs $(\mathsf{sk}, \mathsf{pk})$ where $\mathsf{pk} \leftarrow \mathsf{sk} \cdot P_2 \in \mathbb{G}_2$;
- $\mathsf{Sign}(\mathsf{sk}, m) \to \sigma$. Sign takes in signer's secret key sk and a message $m \in \mathbb{M}$, it outputs $\sigma \leftarrow \mathsf{sk} \cdot \mathsf{H}_0(m) \in \mathbb{G}_1$;
- $\mathsf{Vrfy}(\mathsf{pk}, m, \sigma) \to \{0, 1\}$. If $e(\sigma, P_2) = e(\mathsf{H}_0(m), \mathsf{pk})$, Vrfy outputs 1, otherwise outputs 0;
- $\mathsf{Aggre}(\{\sigma_i\}) \to \sigma$. Each signer provides a signature $\sigma_i \in \mathbb{G}_1$ on distinct messages m_i, and anyone can verify with Vrfy and compute $\sigma \leftarrow \Pi_{i=1}^{n} \sigma_i$ on valid signatures for the aggregate signature;
- $\mathsf{VrfyAggre}(\{\mathsf{pk}_i\}, \{m_i\}, \sigma) \to \{0, 1\}$. Given an aggregate signature $\sigma \in \mathbb{G}_1$ and (m_i, pk_i) of signers who is involved in the aggregate signature, verifiers need to: (1) ensure the messages m_i are all distinct and output 0 otherwise; (2) compute $h_i \leftarrow \mathsf{H}_0(m_i)$ for all users and accept if $e(\sigma, P_2) = \Pi_{i=1}^{n} e(h_i, \mathsf{pk}_i)$ holds.

BLS signature schemes should be correct and EUF-CMA secure. Moreover, we adopt the "aggregate chosen-key" security model from [37] for the aggregatable functionality.

In the n-user setting, the adversary $\mathcal{A}$ is provided with a single public key. Its goal is the existential forgery of an aggregate signature. The adversary can choose all of the public keys except the challenge public key. The adversary can also access a signing oracle on the challenge key. The game is between a challenger and an adversary $\mathcal{A}$.

- **Setup.** The challenger runs KeyGen to generate a key pair $(\mathsf{sk}, \mathsf{pk})$ and sends pk to the adversary.
- **Queries.** $\mathcal{A}$ chooses messages from $\mathbb{M}$ adaptively and receives signatures from the challenger with sk.
- **Forgery.** $\mathcal{A}$ outputs $n-1$ additional keys $\mathsf{pk}_2, \ldots, \mathsf{pk}_n$, n messages $m_1, \ldots, m_n$, and an aggregate signature σ generated from n messages and keys (*i.e.*, n pairs of $(m, \mathsf{sk}, \mathsf{pk})$).

$\mathcal{A}$ wins if the aggregate signature σ is a valid signature on $(m_1, \ldots, m_n)$ under public keys $(\mathsf{pk}, \mathsf{pk}_2, \ldots, \mathsf{pk}_n)$, without querying to the signing oracle with m_1 under pk. We model the hash function as a random oracle.

Definition 17 (Aggregation Unforgeability) *An aggregate signature scheme is (t, q_H, q_S)-secure against existential forgery in the aggregate chosen-key model if any adversary $\mathcal{A}$ runs within t time; makes at most q_H queries to the hash function (random oracle) and at most q_S queries to the signing oracle. The following condition holds:*

$$\Pr[\mathcal{A} \textit{ wins the game}] = \mathsf{negl}(\lambda).$$

The probability takes over the randomness of KeyGen *and* $\mathcal{A}$.

By [37], we have the following lemma.

Lemma 1 *The BLS signature scheme is aggregation unforgeable in the random oracle model if the co-CDH assumption holds for* $\mathsf{BG} \leftarrow \mathsf{BGGen}(1^\lambda)$.

3.3.2 The State-of-the-Art SPS-EQ Scheme

We show the SPS-EQ scheme given by [8] with respect to a fully adaptive non-interactive zero-knowledge (NIZK) argument $\mathsf{NIZK} \triangleq (\mathsf{PGen}, \mathsf{PPro}, \mathsf{PSim}, \mathsf{PRVer}, \mathsf{PVer}, \mathsf{ZKEval})$.

Construction 1 (SPS-EQ Scheme SPSEQ**)** *The algorithms are:*

- $\mathsf{Setup}(1^\lambda)$. *Run* $\mathsf{BG} \leftarrow \mathsf{BGGen}(1^\lambda)$ *and sample matrices* $\mathbf{A}, \mathbf{A}_0, \mathbf{A}_1 \xleftarrow{\$} \mathcal{D}_1$ *from matrix distribution. Generate a common reference string and trapdoor for the malleable NIZK argument with* $(\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{NIZK.PGen}(1^\lambda, \mathsf{BG})$. *Return* $\mathsf{pp} = (\mathsf{BG}, [\mathbf{A}]_2, [\mathbf{A}_0]_1, [\mathbf{A}_1]_1, \mathsf{crs}, \ell)$;
- $\mathsf{KeyGen}(\mathsf{pp})$. *Sample* $\mathbf{K}_0 \xleftarrow{\$} \mathbb{Z}_p^{2\times 2}, \mathbf{K} \xleftarrow{\$} \mathbb{Z}_p^{\ell\times 2}$. *Compute* $[\mathbf{B}]_2 = [\mathbf{K}_0]_2[\mathbf{A}]_2$ *and* $[\mathbf{C}]_2 = [\mathbf{K}]_2[\mathbf{A}]_2$. *Set* $\mathsf{sk} = (\mathbf{K}_0, \mathbf{K})$ *and* $\mathsf{pk} = ([\mathbf{B}]_2, [\mathbf{C}]_2)$. *Return* $(\mathsf{sk}, \mathsf{pk})$;

- $\mathsf{Sign}(\mathsf{pp}, \mathsf{sk}, [\mathbf{m}]_1)$. *Sample* $r_1, r_2 \xleftarrow{\$} \mathbb{Z}_p$. *Compute* $[\mathbf{t}]_1 = [\mathbf{A}_0]_1 r_1$ *and* $[\mathbf{w}]_1 = [\mathbf{A}_0]_1 r_2$. *Compute* $\mathbf{u}_1 = \mathbf{K}_0^\top [\mathbf{t}]_1 + \mathbf{K}^\top [\mathbf{m}]_1$ *and* $\mathbf{u}_2 = \mathbf{K}_0^\top [\mathbf{w}]_1$. *Generate proof with* $(\Omega_1, \Omega_2, [z_0]_2, [z_1]_2, Z_1) \leftarrow \mathsf{NIZK.PPro}(\mathsf{crs}, [\mathbf{t}]_1, r_1, [\mathbf{w}]_1, r_2)$. *Set* $\sigma = ([\mathbf{u}_1]_1, [\mathbf{t}]_1, \Omega_1, [z_0]_2, [z_1]_2, Z_1)$ *and* $\tau = ([\mathbf{u}_2]_1, [\mathbf{w}]_1, \Omega_2)$. *Return* (σ, τ)*;*
- $\mathsf{ChgRep}(\mathsf{pp}, [\mathbf{m}]_1, (\sigma, \tau), \mu, \rho, \mathsf{pk})$. *Parse* $\sigma = ([\mathbf{u}_1]_1, [\mathbf{t}]_1, \Omega_1, [z_0]_2, [z_1]_2, Z_1)$ *and* $\tau \in \{([\mathbf{u}_2]_1, [\mathbf{w}]_1, \Omega_2), \perp\}$. *Let* $\Omega = (\Omega_1, \Omega_2, [z_0]_2, [z_1]_2, Z_1)$. *Check proof with* $\mathsf{NIZK.PVer}(\mathsf{crs}, [\mathbf{t}]_1, [\mathbf{w}]_1, \Omega)$. *Check if* $e([\mathbf{u}_2]_1^\top, \mathbf{A}]_2) = e([\mathbf{w}]_1^\top, \mathbf{B}]_2)$ *and* $e([\mathbf{u}_1]_1^\top, \mathbf{A}]_2) = e([\mathbf{t}]_1^\top, \mathbf{B}]_2) + e([\mathbf{m}]_1^\top, \mathbf{C}]_2)$. *Sample* $\alpha, \beta \xleftarrow{\$} \mathbb{Z}_p^*$. *Compute* $[\mathbf{u'}_1]_1 = \rho(\mu[\mathbf{u}_1]_1 + \beta[\mathbf{u}_2]_1)$ *and* $[\mathbf{t'}]_1 = \mu[\mathbf{t}]_1 + \beta[\mathbf{w}]_1 = [\mathbf{A}_0]_1(\mu r_1 + \beta r_2)$. *And for* $i \in \{0, 1\}$*, compute* $[z'_i]_2 = \alpha[z_i]_2, [\mathbf{a'}_i]_1 = \alpha\mu[\mathbf{a}_i^1]_1 + \alpha\beta[\mathbf{a}_i^2]_1$, $[d'_i]_2 = \alpha\mu[d_i^1]_2 + \alpha\beta[d_i^2]_2$. *Set* $\Omega' = (([\mathbf{a'}_i]_1, [d'_i]_2, [z'_i]_2)_{i\in\{0,1\}}, \alpha Z_1)$. *Set* $\sigma' = ([\mathbf{u'}_1]_1, [\mathbf{t'}]_1, \Omega')$. *Return* $(\mu[\mathbf{m}]_1, \sigma')$*;*
- $\mathsf{Verify}(\mathsf{pp}, (\rho, \mathsf{pk}), [\mathbf{m}]_1, (\sigma, \tau))$. *Parse* $\sigma = ([\mathbf{u}_1]_1, [\mathbf{t}]_1, \Omega_1, [z_0]_2, [z_1]_2, Z_1)$ *and* $\tau \in \{([\mathbf{u}_2]_1, [\mathbf{w}]_1, \Omega_2), \perp\}$. *Check proof* Ω_1 *with* $\mathsf{NIZK.PRVer}(\mathsf{crs}, [\mathbf{t}]_1, \Omega_1, [z_0]_2, [z_1]_2, Z_1)$ *and check if* $e([\mathbf{u}_1]_1^\top, \mathbf{A}]_2) = e([\mathbf{t}]_1^\top, \mathbf{B}]_2) + e([\mathbf{m}]_1^\top, \mathbf{C}]_2)$. *If* $\tau \neq \perp$*, then check proof* Ω_2 *with* $\mathsf{NIZK.PRVer}(\mathsf{crs}, [\mathbf{w}]_1, \Omega_2, [z_0]_2, [z_1]_2, Z_1)$ *and check if* $e([\mathbf{u}_2]_1^\top, \mathbf{A}]_2) = e([\mathbf{w}]_1^\top, \mathbf{B}]_2)$.

The given SPS-EQ construction (Construction 1) satisfies correctness, the EUF-CMA, and the property of Perfect Adaption of Signatures with respect to Message Space. The formal definitions are as follows.

Definition 18 (Correctness) *An SPS-EQ scheme satisfies correctness, if for any* $\lambda > 0, \ell > 1$, $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$, *and* $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KeyGen}(\mathsf{pp})$*:*

$$\Pr\left[\mathsf{Verify}(\mathsf{pk}, \mathsf{Sign}(\mathsf{sk}, [\mathbf{m}]_1)\right] = 1 \wedge$$
$$\Pr\left[\mathsf{Verify}(\rho \cdot \mathsf{pk}, \mathsf{ChgRep}([\mathbf{m}]_1, \mathsf{Sign}(\mathsf{sk}, [\mathbf{m}]_1), \mu, \rho, \mathsf{pk}))\right] = 1.$$

Definition 19 (EUF-CMA) *An SPS-EQ scheme satisfies EUF-CMA, if for any adversary that has access to a signing oracle* $\mathcal{O}_{\mathsf{Sign}}(\mathsf{sk}, \cdot)$ *with queries* $[\mathbf{m}]_i \in \mathsf{Q}$*, the following probability is negligible of* λ *for any* $\lambda > 0, \ell > 1$ *and* $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$*:*

$$\Pr\left[\begin{array}{ll} (\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KeyGen}(\mathsf{pp}); : & \forall [\mathbf{m}]_i \in \mathsf{Q}, [\mathbf{m}^*]_{\mathcal{R}} \neq [\mathbf{m}]_{\mathcal{R}} \wedge \\ ([\mathbf{m}]_i^*, \sigma^*) \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{Sign}}}(\mathsf{pk}) & \mathsf{Verify}([\mathbf{m}]_i^*, \sigma^*, \mathsf{pk}) = 1 \end{array}\right].$$

Finally, we define the perfect adaption of signatures with respect to message space (under malicious keys in the honest parameters model).

Definition 20 *An SPS-EQ scheme over a message space* $\mathbb{M} \subseteq (\mathbb{G}_i^*)^\ell$ *perfectly adapts signatures with respect to the message space, if for all tuples* $(\mathsf{pp}, [\mathsf{pk}]_j, [\mathbf{m}]_i, (\sigma, \tau), \mu, \rho)$ *such that* $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda), [\mathbf{m}]_i \in \mathbb{M}$, $\mu, \rho \in \mathbb{Z}_p^*$, *and* $\mathsf{Verify}(\mathsf{pk}, [\mathbf{m}]_i, (\sigma, \tau)) = 1$*, we have the output* $([\mu \cdot \mathbf{m}]_i, \sigma^*) \leftarrow \mathsf{ChgRep}([\mathbf{m}]_i, (\sigma, \tau), \mu, \rho, [\mathsf{pk}]_j)$ *where* σ^* *is a* random *element in the signature space such that* $\mathsf{Verify}([\rho \cdot \mathsf{pk}, \mu \cdot \mathbf{m}]_i, \sigma^*) = 1$.

Chapter 4

Our New Functionalities to Anonymous Credentials

In this chapter, we present two constructions that introduce new functionalities to existing anonymous credentials schemes. Concretely, we bring credential aggregation to the decentralized anonymous credentials (DAC) scheme [6]; and bring auditability (of issuers) to the attribute-based credentials (ABC) scheme [7], [8]. Both of our derived schemes can be further applied to enhance contact tracing systems.

We start from showing a generic cryptographic plug-in tool that we build to achieve these functionalities, *i.e.*, the auditable public keys (APK) mechanism (APK).

4.1 Our APK Mechanism

Proposed in [38], the updatable public key mechanism is a generic tool that can be integrated into many cryptographic primitives, *e.g.*, digital signature and public key encryption schemes. The mechanism enables public keys to be updated in a public fashion, and updated public keys are indistinguishable from freshly generated ones. The verification of public keys either requires the corresponding secret key (verifying the key pair) or the randomness used in the updating algorithm. However, these approaches are insufficient in multi-user cases, *e.g.*, in credentials schemes and contact tracing systems. The reasons are: (1) secret keys should only be known to their holders; (2) asking the user who runs the updating algorithm to store its random value or keep the value secret may require impractical assumptions (*e.g.*, assuming *every* user to be honest).

Therefore, we propose an APK mechanism to extend the updatable public

key by embedding a structure represented by an auditing key into public keys. The structure enables designated third parties who hold the auditing key, the auditors, to decide whether a public key is updated from the corresponding public key of the auditing key. Moreover, we require that no auditor can learn the corresponding *secret* key from its auditing key. Hence, we separate the role of users, *i.e.*, a user can delegate her capability of auditing to an auditor without revealing the secret key, and a user who performs the updating algorithm can discard her randomness without the concern of being asked to provide it.

The formal syntax and security definitions of APK are given in the following. We recall and extend the definitions from [38].

Definition 21 (Auditable Public Key Mechanism) *An auditable public key (APK) mechanism involves algorithms* $\mathsf{APK} \triangleq (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Update}, \mathsf{VerifyKP}, \mathsf{VerifyAK}, \mathsf{Audit})$ *that are performed as follows.*

- $\mathsf{Setup}(1^\lambda)$ *takes as input the security parameter* λ *and outputs the public parameter* pp *that includes secret, public and auditing key space* $\mathcal{SK}, \mathcal{PK}, \mathcal{AK}$*. These are given implicitly as input to all other algorithms;*
- $\mathsf{KeyGen}(\mathsf{pp})$ *takes as input the public parameter* pp *and outputs a secret and public key pair* $(\mathsf{sk}, \mathsf{pk}) \in \mathcal{SK} \times \mathcal{PK}$*, and an auditing key* $\mathsf{ak} \in \mathcal{AK}$*. Later, we omit* pp *in algorithm inputs;*
- $\mathsf{Update}(\mathsf{pk}; r)$ *takes as input a public key* pk *and a randomness* r*. It outputs a new public key* $\mathsf{pk}' \in \mathcal{PK}$*;*
- $\mathsf{VerifyKP}(\mathsf{sk}, \mathsf{pk}', r)$ *is deterministic, and takes as input a secret key* $\mathsf{sk} \in \mathcal{SK}$*, a value* r *and a public key* $\mathsf{pk}' \in \mathcal{PK}$*. It outputs* 1 *if* $\mathsf{pk}' \leftarrow \mathsf{Update}(\mathsf{pk}; r)$ *given* $(\mathsf{sk}, \mathsf{pk}, \cdot) \leftarrow \mathsf{KeyGen}(\mathsf{pp})$*, or* 0 *otherwise;*
- $\mathsf{VerifyAK}(\mathsf{sk}, \mathsf{ak})$ *is deterministic, and takes as input a secret key* $\mathsf{sk} \in \mathcal{SK}$ *and an auditing key* $\mathsf{ak} \in \mathcal{AK}$*. It outputs* 1 *if* $(\mathsf{sk}, \cdot, \mathsf{ak}) \leftarrow \mathsf{KeyGen}(\mathsf{pp})$*, or* 0 *otherwise;*
- $\mathsf{Audit}(\mathsf{ak}, \mathsf{pk}', \mathsf{pk})$ *is deterministic and is performed by a designated auditor who holds the auditing key* $\mathsf{ak} \in \mathcal{AK}$ *of a secret and public key pair* $(\mathsf{sk}, \mathsf{pk}) \in \mathcal{SK} \times \mathcal{PK}$*.* Audit *takes as input a public key* $\mathsf{pk}' \in \mathcal{PK}$*, the auditing key* ak *and the public key* pk*. It outputs* 1 *if* pk' *is updated from* pk*, i.e., there exists* r *such that* $\mathsf{pk}' \leftarrow \mathsf{Update}(\mathsf{pk}; r)$*, or* 0 *otherwise.*

APK mechanism satisfies correctness, indistinguishability, and unforgeability.

Definition 22 (Correctness) *An APK mechanism is perfectly correct if the following properties hold for any* $\lambda > 0, \mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$*, and* $(\mathsf{sk}, \mathsf{pk}, \mathsf{ak}) \leftarrow \mathsf{KeyGen}(\mathsf{pp})$*: (1) the update process verifies with* $\mathsf{VerifyKP}(\mathsf{sk}, \mathsf{Update}(\mathsf{pk}; r),$

$r) = 1$*; (2) the auditing key verifies with* $\mathsf{VerifyAK}(\mathsf{sk}, \mathsf{ak}) = 1$*; (3) the auditing process verifies with* $\mathsf{Audit}(\mathsf{ak}, \mathsf{pk}', \mathsf{pk}) = 1$ *for any* $\mathsf{pk}' \leftarrow \mathsf{Update}(\mathsf{pk})$.

The indistinguishability of APK follows [38], *i.e.*, no adversary can distinguish between an updated known public key and a freshly generated one. Note that (also applies in unforgeability) the adversary can query to KeyGen and Update since these algorithms are publicly available.

Definition 23 (Indistinguishability) *An APK mechanism satisfies indistinguishability if for any PPT adversary* $\mathcal{A}$*, the following probability holds for any* $\lambda > 0, \mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$*, and* $(\mathsf{sk}^*, \mathsf{pk}^*, \mathsf{ak}^*) \leftarrow \mathsf{KeyGen}(\mathsf{pp})$

$$\left| \Pr \left[\begin{array}{l} b \xleftarrow{\$} \{0,1\}; \mathsf{pk}_0 \leftarrow \mathsf{Update}(\mathsf{pk}^*); \\ (\mathsf{sk}_1, \mathsf{pk}_1, \mathsf{ak}_1) \leftarrow \mathsf{KeyGen}(\mathsf{pp}); \quad : b^* = b \\ b^* \leftarrow \mathcal{A}(\mathsf{pk}^*, \mathsf{pk}_b) \end{array} \right] - \frac{1}{2} \right| \leq \mathsf{negl}(\lambda).$$

We formalize two types of unforgeability, *i.e.*, for secret key and auditing key. Concretely, the former requires that given an auditing key with its corresponding public key, the adversary cannot produce a secret and public key pair, and a randomness, such that: (1) the output public key is updated from the secret key's corresponding public key with respect to the randomness; (2) the output secret key and the given auditing key pass the verification given by $\mathsf{VerifyAK}$; (3) the auditing key, the output public key and the given public key pass the auditing given by Audit. This property captures adversarial auditors who hold an auditing key and intend to recover the corresponding secret key. Hence, it covers the one given in [38], in which the adversary is only given a public key.

Next, the auditing key unforgeability requires that given a public key, the adversary cannot produce an auditing key such that the corresponding secret key of the public key verifies the auditing key. This property captures adversarial participants who intend to trigger the auditing algorithm to output 1 for arbitrary public keys. The formal definitions are as follows.

Definition 24 (Secret Key Unforgeability) *An APK mechanism satisfies secret key unforgeability if for any PPT adversary* $\mathcal{A}$*, the following probability holds for any* $\lambda > 0, \mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$*, and* $(\mathsf{sk}, \mathsf{pk}, \mathsf{ak}) \leftarrow \mathsf{KeyGen}(\mathsf{pp})$

$$\Pr \left[\begin{array}{r} \mathsf{VerifyKP}(\mathsf{sk}', \mathsf{pk}', r) = 1 \wedge \\ (\mathsf{sk}', \mathsf{pk}', r) \leftarrow \mathcal{A}(\mathsf{ak}, \mathsf{pk}) \quad : \mathsf{VerifyAK}(\mathsf{sk}', \mathsf{ak}) = 1 \wedge \\ \mathsf{Audit}(\mathsf{ak}, \mathsf{pk}', \mathsf{pk}) = 1 \end{array} \right] \leq \mathsf{negl}(\lambda).$$

Definition 25 (Auditing Key Unforgeability) *An APK mechanism is auditing key unforgeable if for any PPT adversary $\mathcal{A}$, the following probability holds for any $\lambda > 0, \mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$, and $(\mathsf{sk}, \mathsf{pk}, \mathsf{ak}) \leftarrow \mathsf{KeyGen}(\mathsf{pp})$*

$$\Pr\left[\mathsf{ak}' \leftarrow \mathcal{A}(\mathsf{pk}) : \mathsf{VerifyAK}(\mathsf{sk}, \mathsf{ak}') = 1\right] \leq \mathsf{negl}(\lambda).$$

For constructions, similar to the updatable public key [38], our APK can be constructed from the DDH problem and its variants. Here, we show two constructions, one is based on the DDH problem, and the other one is based on the MDDH problem. Later, we use the DDH-based APK to extend the BLS signature (given in Section 3.3.1) and the MDDH-based APK to extend the SPS-EQ scheme (given in Section 3.3.2).

4.1.1 Our DDH-Based APK and APK-BLS

Construction 2 (DDH-Based APK $\mathsf{APK}_{\mathrm{DDH}}$) *Let $\mathsf{BG} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, P_1, P_2, e)$ be the output of the bilinear group generator $\mathsf{BGGen}(1^\lambda)$. The algorithms of APK are as follows.*

- $\mathsf{KeyGen}(\mathsf{BG})$*: Sample* $\mathsf{ak}, \mathsf{sk}_0 \xleftarrow{\$} \mathbb{Z}_p$*. Set* $\mathsf{sk}_1 = \mathsf{ak} \cdot \mathsf{sk}_0$*. Then, compute* $\mathsf{pk}_0 = \mathsf{sk}_0 \cdot P_2$ *and* $\mathsf{pk}_1 = \mathsf{sk}_1 \cdot P_2$*. Finally, set* $\mathsf{sk} = (\mathsf{sk}_0, \mathsf{sk}_1), \mathsf{pk} = (\mathsf{pk}_0, \mathsf{pk}_1)$ *and output* $(\mathsf{sk}, \mathsf{ak}, \mathsf{pk})$*;*
- $\mathsf{Update}(\mathsf{pk}; r)$*: Parse* $\mathsf{pk} = (\mathsf{pk}_0, \mathsf{pk}_1)$*. Sample* $r \xleftarrow{\$} \mathbb{Z}_p$ *and compute* $\mathsf{pk}'_0 = r \cdot \mathsf{pk}_0, \mathsf{pk}'_1 = r \cdot \mathsf{pk}_1$*. Output* $\mathsf{pk}' = (\mathsf{pk}'_0, \mathsf{pk}'_1)$*;*
- $\mathsf{VerifyKP}(\mathsf{sk}, \mathsf{pk}', r)$*: Parse* $\mathsf{sk} = (\mathsf{sk}_0, \mathsf{sk}_1)$ *and* $\mathsf{pk}' = (\mathsf{pk}'_0, \mathsf{pk}'_1)$*. Output* 1 *if* $\mathsf{pk}'_0 = r \cdot \mathsf{sk}_0 \cdot P_2 \wedge \mathsf{pk}'_1 = r \cdot \mathsf{sk}_1 \cdot P_2$*, or* 0 *otherwise;*
- $\mathsf{VerifyAK}(\mathsf{sk}, \mathsf{ak})$*: Parse* $\mathsf{sk} = (\mathsf{sk}_0, \mathsf{sk}_1)$*. Output* 1 *if* $\mathsf{sk}_1 = \mathsf{ak} \cdot \mathsf{sk}_0$*, or* 0 *otherwise;*
- $\mathsf{Audit}(\mathsf{ak}, \mathsf{pk}', \mathsf{pk})$*: Parse* $\mathsf{pk}' = (\mathsf{pk}'_0, \mathsf{pk}'_1), \mathsf{pk} = (\mathsf{pk}_0, \mathsf{pk}_1)$*. Output* 1 *if* $\mathsf{pk}_1 = \mathsf{ak} \cdot \mathsf{pk}_0 \wedge \mathsf{pk}'_1 = \mathsf{ak} \cdot \mathsf{pk}'_0$*, or* 0 *otherwise.*

Therefore, the integration works as follows.

Construction 3 (APK-BLS) *Let $\mathsf{BG} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, P_1, P_2, e)$ be the output of the bilinear group generator $\mathsf{BGGen}(1^\lambda)$. Let $\mathsf{H} : \mathbb{M} \to \mathbb{G}_1$ be a cryptographic hash function where $\mathbb{M}$ denotes the message space. Let $\mathsf{APK}_{\mathrm{DDH}} = (\mathsf{KeyGen}, \mathsf{Update}, \mathsf{VerifyKP}, \mathsf{VerifyAK}, \mathsf{Audit})$ be a DDH-based APK mechanism. The algorithms of BLS with APK are as follows.*

- $\mathsf{KeyGen}, \mathsf{VerifyKP}, \mathsf{VerifyAK}, \mathsf{Audit}$ *are the same as in* $\mathsf{APK}_{\mathrm{DDH}}$*;*
- $\mathsf{Sign}(\mathsf{sk}, m)$*: Parse* $\mathsf{sk} = (\mathsf{sk}_0, \mathsf{sk}_1)$ *and output* $\sigma = \mathsf{sk}_1 \cdot \mathsf{H}(m) \in \mathbb{G}_1$*;*

- $\mathsf{Verify}(\mathsf{pk}, m, \sigma)$: *Parse* $\mathsf{pk} = (\mathsf{pk}_0, \mathsf{pk}_1)$ *and output* 1 *if* $e(\sigma, P_2) = e(\mathsf{H}(m), \mathsf{pk}_1)$, *or* 0 *otherwise;*
- $\mathsf{Update}(\mathsf{pk}, \sigma; r)$: *Run* $\mathsf{APK}_{\mathrm{DDH}}.\mathsf{Update}(\mathsf{pk}; r) \to \mathsf{pk}'$ *and compute* $\sigma' = r \cdot \sigma$. *Output* (pk', σ').
- Aggre *and* $\mathsf{VerifyAggre}$ *remain the same as in the BLS scheme.*

Since the EUF-CMA security of the (type-3) BLS signature is proven under the co-CDH assumption [37], and the APK given in Construction 2 considers the DDH problem in $\mathbb{G}_2$, it is convenient to assume the SXDH assumption to hold for BGGen to prove the EUF-CMA security of our extended BLS construction (Construction 3). Hence, we have the following lemma.

Lemma 2 *Our APK-BLS scheme satisfies correctness, EUF-CMA and aggregation unforgeability if the co-CDH assumption and the SXDH assumption hold for* $\mathsf{BG} \leftarrow \mathsf{BGGen}(1^\lambda)$.

Proof *Correctness is straightforward to verify. To prove* update-indistinguishability, *the SXDH challenger* $\mathcal{CH}$ *with* BG *first samples* $x, y \xleftarrow{\$} \mathbb{Z}_p$ *and a random bit* $b \xleftarrow{\$} \{0,1\}$. $\mathcal{CH}$ *sets* $z = xy$ *if* $b = 0$ *or* $z \xleftarrow{\$} \mathbb{Z}_p$ *if* $b = 1$, *and sends* $(P_2, x \cdot P_2, y \cdot P_2, z \cdot P_2)$ *to the reduction algorithm* $\mathcal{R}_{SXDH}$ *as a challenge.* $\mathcal{R}_{SXDH}$ *samples a random* $r \xleftarrow{\$} \mathbb{Z}_p$ *and sets* $\mathsf{pk} = (P_2, {P_2}^r, xr \cdot P_2), \mathsf{pk}_b = (P_2, yr \cdot P_2, zr \cdot P_2)$. *It sends* $(\mathsf{pk}, \mathsf{pk}_b)$ *to the update-indistinguishability adversary* $\mathcal{A}_{UI}$. *To implement the signing oracle, with message* $m \in \mathbb{M}$ *as input, where* $\mathbb{M}$ *is the message space, the reduction algorithm first forwards* (m, r) *and* $\mathcal{A}_{UI}$*'s oracle choice to the challenger. If* $\mathcal{A}_{UI}$ *queries for* pk*'s corresponding secret key,* $\mathcal{R}_{SXDH}$ *forwards the challenger's reply* $xr \cdot \mathsf{H}_0(m)$ *to the adversary. Otherwise, for* pk_b*'s corresponding secret key, it forwards* $zr \cdot \mathsf{H}_0(m)$. *The adversary outputs a bit* b^* *for the update-indistinguishability game. The reduction forwards* b^* *as an answer for the DDH challenge on group* $\mathbb{G}_2$. *The game is illustrated in Figure 4.1.*

For $b = 0$, *i.e.,* $z = xy$, *the challenge is a DDH tuple on* $\mathbb{G}_2$, *then* pk_0 *distributes identically to* pk. *Otherwise, for* $b = 1$, pk_1 *distributes identically to a fresh generated public key. For an adversary running* Verify *algorithm, it receives:*

$$\mathsf{Verify}(\mathsf{pk}, m, xr \cdot \mathsf{H}_0(m)) = 1, \quad \mathsf{Verify}(\mathsf{pk}, m, zr \cdot \mathsf{H}_0(m)) = 0$$
$$\mathsf{Verify}(\mathsf{pk}_b, m, xr \cdot \mathsf{H}_0(m)) = 0, \quad \mathsf{Verify}(\mathsf{pk}_b, m, zr \cdot \mathsf{H}_0(m)) = 1$$

for any queried results. Thus, the reduction algorithm has the same advantage in the DDH game on $\mathbb{G}_2$ *(SXDH game) as the adversary in the update-indistinguishability game has.*

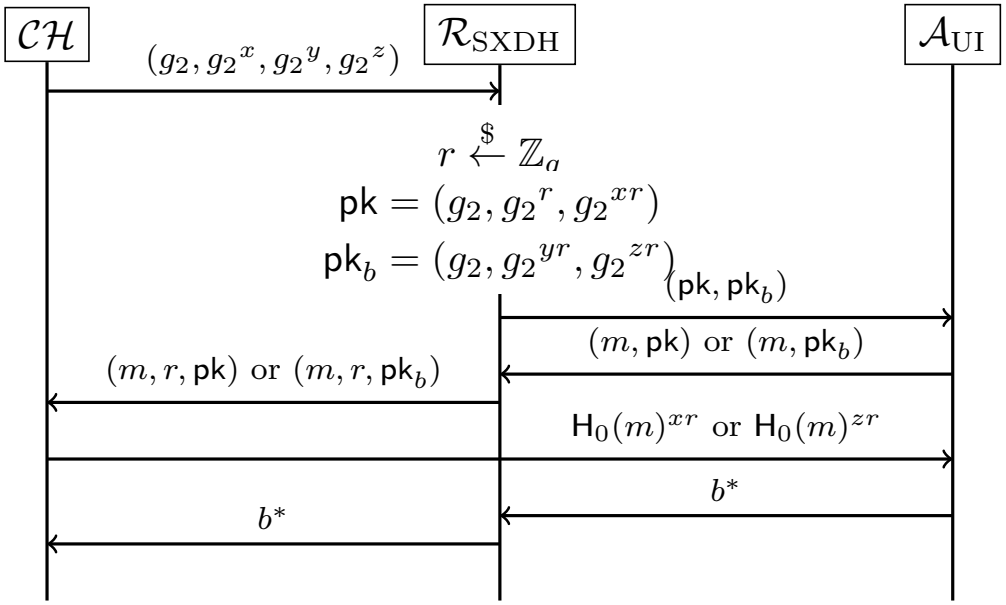


Figure 4.1: Update-Indistinguishability Game

To prove EUF-CMA security *and* aggregation unforgeability*, we first prove* secret key unforgeability*. On receiving the SXDH challenge* $\mathsf{ch} = (P_2, x \cdot P_2, y \cdot P_2, z \cdot P_2)$*, the reduction* $\mathcal{R}_{SXDH}$ *first invokes three parallel instances of a reduction algorithm* $\mathcal{R}_{DL}$ *for discrete logarithm (DL) challenges on* $\mathbb{G}_2$*, i.e.,* $(P_2, h = s \cdot P_2)$*, where* $s \in \{x, y, z\}$*. The implementation of the signing oracle is similar as it is in the updating indistinguishability game. We illustrate the secret key unforgeability game in Figure 4.2.*

The input of $\mathcal{A}_{SKU}$ *is identical as in Definition 24. The winning condition of* $\mathcal{A}_{SKU}$ *requires its output* (pk', s', r') *to satisfy the verification algorithms, phrasing* $\mathsf{pk}' = (P_2, P_2', h')$*, i.e., by* $\mathsf{VerifyKP}$*, it holds* $P_2' = rr' \cdot P_2$ *and* $h' = s' \cdot rr' \cdot P_2$*. The above situation indicates that* $h = s' \cdot P_2$*, which is* $s' = s$*. As for* $s \in \{x, y, z\}$*,* $\mathcal{R}_{SXDH}$ *can easily check if* $z = xy$ *and output* b^**.*

With secret key unforgeability, i.e., no adversary can gain advantage from the public key rerandomizing mechanism, we can consider $\mathsf{sk}_1 \cdot \mathsf{sk}_2$ *as the secret key of the BLS scheme and reduce our scheme to the original one.*

4.1.2 An MDDH-Based APK and APK-SPS-EQ

In order to work with the SPS-EQ given in [8], the setup algorithm Setup runs $\mathsf{BG} \leftarrow \mathsf{BGGen}(1^\lambda)$ and samples a matrix $\mathbf{A} \xleftarrow{\$} \mathcal{D}_1$. It outputs $\mathsf{pp} \triangleq (\mathsf{BG}, [\mathbf{A}]_2, \ell)$ where $\mathsf{BG} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, P_1, P_2, e)$, and ℓ is a parameter for

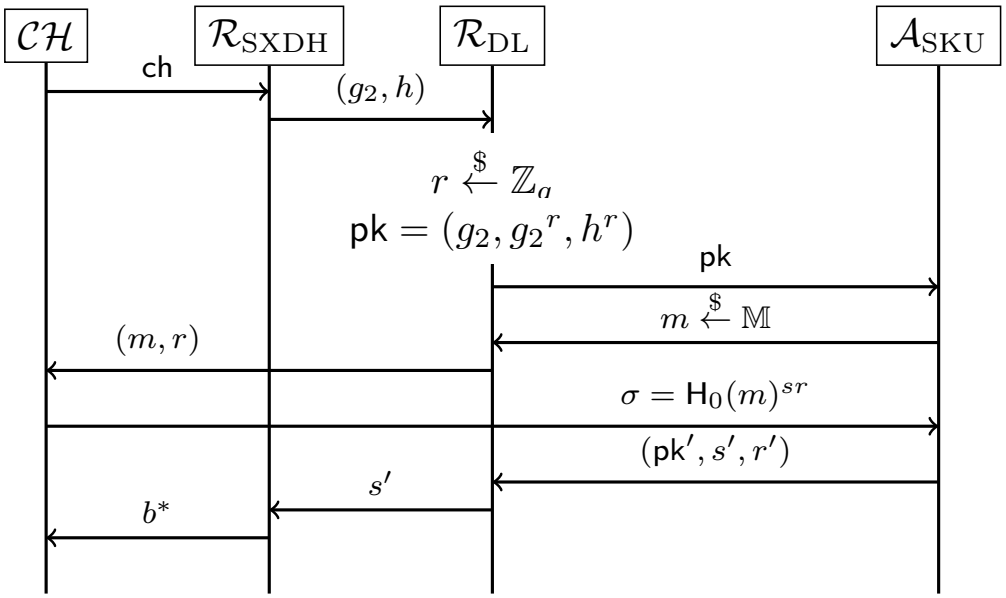


Figure 4.2: Secret Key Unforgeability Game

message size in the SPS-EQ. We present a construction of APK based on group $(\mathbb{G}_2, P_2, p)$ where the MDDH and KerMDH assumptions are believed to hold.

Construction 4 (MDDH-Based APK APK) *The rest of the algorithms are:*

- $\mathsf{KeyGen}(\mathsf{pp})$*: Sample matrices* $\mathbf{K}_0 \xleftarrow{\$} \mathcal{D}_{\ell,2}$ *and* $\mathbf{K}_1 \xleftarrow{\$} \mathbb{Z}_p^{2\times 2}$ *of full rank* 2*. Set* $\mathbf{K} = \mathbf{K}_0\mathbf{K}_1$*. Then, compute* $[\mathbf{B}]_2 = [\mathbf{K}_1\mathbf{A}]_2$ *and* $[\mathbf{C}]_2 = [\mathbf{K}\mathbf{A}]_2$*. Finally, set* $\mathsf{sk} = (\mathbf{K}_1, \mathbf{K}), \mathsf{pk} = ([\mathbf{B}]_2, [\mathbf{C}]_2), \mathsf{ak} = \mathbf{K}_0$ *and output* $(\mathsf{sk}, \mathsf{pk}, \mathsf{ak})$*;*
- $\mathsf{Update}(\mathsf{pk}; r)$*: Sample* $r \xleftarrow{\$} \mathbb{Z}_p$ *and compute* $[\mathbf{B}']_2 = r \cdot [\mathbf{B}]_2, [\mathbf{C}']_2 = r \cdot [\mathbf{C}]_2$*. Output* $\mathsf{pk}' = ([\mathbf{B}']_2, [\mathbf{C}']_2)$*;*
- $\mathsf{VerifyKP}(\mathsf{sk}, \mathsf{pk}', r)$*: Parse* $\mathsf{sk} = (\mathsf{sk}_0, \mathsf{sk}_1)$ *and* $\mathsf{pk}' = (\mathsf{pk}'_0, \mathsf{pk}'_1)$*. Output* 1 *if* $\mathsf{pk}'_0 = r \cdot \mathsf{sk}_0 \cdot [\mathbf{A}]_2 \wedge \mathsf{pk}'_1 = r \cdot \mathsf{sk}_1 \cdot [\mathbf{A}]_2$*, or* 0 *otherwise;*
- $\mathsf{VerifyAK}(\mathsf{sk}, \mathsf{ak})$*: Parse* $\mathsf{sk} = (\mathsf{sk}_0, \mathsf{sk}_1)$*. Output* 1 *if* $\mathsf{sk}_1 = \mathsf{ak} \cdot \mathsf{sk}_0$*, or* 0 *otherwise;*
- $\mathsf{Audit}(\mathsf{ak}, \mathsf{pk}', \mathsf{pk})$*: Parse* $\mathsf{pk}' = (\mathsf{pk}'_0, \mathsf{pk}'_1), \mathsf{pk} = (\mathsf{pk}_0, \mathsf{pk}_1)$*. Output* 1 *if* $\mathsf{pk}_1 = \mathsf{ak} \cdot \mathsf{pk}_0 \wedge \mathsf{pk}'_1 = \mathsf{ak} \cdot \mathsf{pk}'_0$*, or* 0 *otherwise.*

Hence, we have the following theorem.

Theorem 1 *The APK mechanism* APK *given by Construction 4 satisfies the following properties.*

- *Correctness (Definition 22);*
- *Indistinguishability (Definition 23) if the $\mathcal{D}_{l,1}$-MDDH assumption where $l \in \{2, \ell\}$ holds on $\mathbb{G}_2$;*
- *Secret key and auditing key unforgeability (Definition 24 and 25) if the $\mathcal{D}_1$-KerMDH holds on $\mathbb{G}_2$.*

Proof *On the additive cyclic group $\mathbb{G}_2$, APK* correctness *can be yielded directly from our construction. To prove* indistinguishability, *let* $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$ *where* $\mathsf{pp} = (\mathsf{BG}, [\mathbf{A}]_2, \ell)$ *are given as above. The reduction receives an MDDH challenge over $\mathbb{G}_2$,* $\mathsf{chl} = (P_2, [\mathbf{X}]_2, [\mathbf{z}]_2)$ *where* $\mathbf{X} \xleftarrow{\$} \mathcal{D}_{l,1}$. *According the challenge bit* $b \in \{0, 1\}$, $\mathbf{z}$ *is set to* $\mathbf{X}y$ *with* $y \xleftarrow{\$} \mathbb{Z}_p$ *(when* $b = 0$*) or* $\mathbf{z} \xleftarrow{\$} \mathbb{Z}_p^l$ *(when* $b = 1$*).* l *takes its value from* $\{2, \ell\}$ *because the two components in a public key,* $[\mathbf{B}]_2$ *and* $[\mathbf{C}]_2$*, are matrices of size* 2×1 *and* $\ell\times1$*, respectively. Note that the reduction needs to prepare both components of the public key. That is, it samples a full-ranked* $\mathbf{X}' \xleftarrow{\$} \mathbb{Z}_p^{l'\times l}$ *such that* $l' \in \{2, \ell\} \wedge l' \neq l$*, and embeds the MDDH challenge* chl *by setting* $\mathsf{pk}^* \triangleq ([\mathbf{X}]_2, \mathbf{X}'[\mathbf{X}]_2)$ *and* $\mathsf{pk}' \triangleq ([\mathbf{z}]_2, \mathbf{X}'[\mathbf{z}]_2)$*. The indistinguishability adversary $\mathcal{A}$ takes as input* $(\mathsf{pk}^*, \mathsf{pk}')$*. If the challenge tuple satisfies* $[\mathbf{z}]_2 = [\mathbf{X}y]_2$ *(when* $b = 0$*), then* pk' *is distributed identically to* pk_0 *(*$\mathsf{pk}_0 \leftarrow \mathsf{Update}(\mathsf{pk}^*)$*, the adversary will output* $b^* = 0$*). Otherwise (when* $b = 1$*),* pk' *is distributed identically to* pk_1 *(a freshly generated public key, the adversary outputs* $b^* = 1$*). Therefore, the reduction has the same advantage in the $\mathcal{D}_{l,1}$-MDDH (*$l \in \{2, \ell\}$*) game as the adversary in the indistinguishability game of Definition 23.*

The proofs of two types of unforgeability are similar. For secret key unforgeability, *the reduction receives a KerMDH challenge over $\mathbb{G}_2$,* $\mathsf{chl} = (P_2, [\mathbf{A}]_2)$ *where* $\mathbf{A} \xleftarrow{\$} \mathcal{D}_1$. *The reduction prepares the inputs for the unforgeability adversary $\mathcal{A}$. That is, it samples* $\mathbf{K}_0 \xleftarrow{\$} \mathcal{D}_{\ell,2}$ *and* $\mathbf{K}_1 \xleftarrow{\$} \mathbb{Z}_p^{2\times2}$ *of full rank* 2*. Then, let* $[\mathbf{X}]_2 = [\mathbf{K}_1\mathbf{A}]_2$*, the reduction embeds the challenge* chl *by setting* $\mathsf{ak} \triangleq \mathbf{K}_0$ *and* $\mathsf{pk} \triangleq ([\mathbf{X}]_2, [\mathbf{K}_0\mathbf{X}]_2)$*. Hence, the input to the adversary in the reduction is distributed identically as in the definition of unforgeability. Suppose the adversary $\mathcal{A}$ breaks secret key unforgeability, which means that* $\mathsf{VerifyKP}, \mathsf{VerifyAK}, \mathsf{Audit}$ *verify the output tuple* $(\mathsf{sk}', \mathsf{pk}', r)$*. More precisely, parse* $\mathsf{sk}' = (\mathsf{sk}'_0, \mathsf{sk}'_1)$*, it holds that* $\mathsf{sk}'_0[\mathbf{A}]_2 = [\mathbf{X}]_2 = [\mathbf{K}_1\mathbf{A}]_2$ *and* $\mathsf{sk}'_1[\mathbf{A}]_2 = [\mathbf{K}_0\mathbf{X}]_2$*. If the adversary can find a non-zero vector* $\mathsf{sk}'_0 - \mathbf{K}_1$ *in the kernel of* $\mathbf{A}$*, it can break the secret key unforgeability. However, finding* sk'_0 *is equivalent to solving a $\mathcal{D}_1$-KerMDH problem (with* sk'_0*, the reduction outputs* $[\mathsf{sk}'_0 - \mathbf{K}_1]_1$ *to the KerMDH challenge and* $e([\mathsf{sk}'_0 - \mathbf{K}_1]_1, [\mathbf{A}]_2) = [0]_T$*). Thus, the reduction advantage in the $\mathcal{D}_1$-KerMDH game is the same as the adversary in the secret key unforgeability game.*

Similarly, the auditing key unforgeability *reduction receives a KerMDH challenge over* $\mathbb{G}_2$, $\mathsf{chl} = (P_2, [\mathbf{X}]_2)$ *where* $\mathbf{X} \in \mathcal{D}_1$. *The reduction samples* $\mathbf{K}_0 \in \mathbb{Z}_p^{\ell \times 2}$ *of full rank* 2 *and relays* $(P_2, [\mathbf{X}]_2, [\mathbf{K}_0\mathbf{X}]_2)$ *to the adversary. Hence, the input of the adversary, i.e.,* $\mathsf{pk} = ([\mathbf{X}]_2, [\mathbf{K}_0\mathbf{X}]_2)$, *distributes identically to the definition. Suppose the adversary* $\mathcal{A}$ *breaks auditing key unforgeability, which means it finds* ak' *such that* $\mathsf{VerifyAK}(\mathsf{sk}, \mathsf{ak}') = 1$. *Note that although the reduction cannot prepare the corresponding secret key, the structure preserves in the public key, i.e., the adversary must output a non-zero* $\mathsf{ak}' - \mathbf{K}_0$ *in the kernel of* $\mathbf{X}$. *As explained before, the reduction cannot gain advantages in the* $\mathcal{D}_1$*-KerMDH game by invoking the auditing key unforgeability adversary.*

Extending the SPS-EQ [8]. We show the original key generation of the SPS-EQ in the following. Recall that the Setup algorithm outputs $\mathsf{pp} = (\mathsf{BG}, [\mathbf{A}]_2, \ell)$.

- $\mathsf{SPSEQ.KeyGen}(\mathsf{pp})$: Sample matrices $\mathbf{K}_1 \xleftarrow{\$} \mathbb{Z}_p^{2\times 2}$ and $\mathbf{K} \xleftarrow{\$} \mathcal{D}_{\ell,2}$ of full rank 2. Then, compute $[\mathbf{B}]_2 = [\mathbf{K}_1\mathbf{A}]_2$ and $[\mathbf{C}]_2 = [\mathbf{K}\mathbf{A}]_2$. Finally, set $\mathsf{sk} = (\mathbf{K}_1, \mathbf{K}), \mathsf{pk} = ([\mathbf{B}]_2, [\mathbf{C}]_2)$ and output $(\mathsf{sk}, \mathsf{pk})$.

The only difference here is that we further sample $\mathbf{K}_0 \xleftarrow{\$} \mathcal{D}_{\ell,2}$ of full rank 2 and compute $\mathbf{K}$ by the multiplication of $\mathbf{K}_0$ and $\mathbf{K}_1$. In the following lemma, we show that this change only increases the SPS-EQ adversary's advantage by at most the advantage of solving a $\mathcal{D}_{\ell,2}$-MDDH problem over $\mathbb{G}_2$.

Lemma 3 *Replacing* $\mathsf{SPSEQ.KeyGen}$ *with* $\mathsf{APK.KeyGen}$ *in* SPSEQ *given in Construction 1 preserves the correctness, EUF-CMA and perfect adaption of signatures with respect to message space of the original scheme.*

Proof *Correctness is straightforward as proven in Theorem 1. We unify the proofs of EUF-CMA and perfect adaption of signatures with respect to message space by considering a sequence of two games:* Game_1 *is the EUF-CMA and perfect adaption of signatures with respect to message space games for the original SPS-EQ scheme with* $\mathsf{SPSEQ.KeyGen}$ *given in Definition 19 and 20; and* Game_0 *substitutes* $\mathsf{SPSEQ.KeyGen}$ *with our APK mechanism's* $\mathsf{APK.KeyGen}$. *Hence,* Game_0 *is the game for our modified scheme. We further denote the adversary* $\mathcal{A}$*'s advantage with* Adv_i *for each game* Game_i *where* $\in \{0, 1\}$. *In the transition of* $\mathsf{Game}_0 \to \mathsf{Game}_1$, $\mathsf{pk}_{\mathsf{Game}_1} = ([\mathbf{K}_1\mathbf{A}]_2, [\mathbf{K}_{\mathsf{Game}_1}\mathbf{A}]_2)$ *replaces* $\mathsf{pk}_{\mathsf{Game}_0} = ([\mathbf{K}_1\mathbf{A}]_2, [\mathbf{K}_0\mathbf{K}_1\mathbf{A}]_2)$. *Note that all matrices are of full rank* 2, *hence, distinguishing* $\mathsf{pk}_{\mathsf{Game}_1}$ *and* $\mathsf{pk}_{\mathsf{Game}_0}$ *is equivalent to solve a challenge of* $\mathcal{D}_{\ell,2}$*-MDDH problem (because* $\mathbf{K}_{\mathsf{Game}_1}$ *is an* $\ell \times 2$ *matrix of full*

rank 2). That is, $|\mathsf{Adv}_0 - \mathsf{Adv}_1| \leq \mathsf{Adv}^{MDDH}_{\mathcal{D}_{\ell,2},\mathbb{G}_2}$. *Moreover, as shown in [8], the original SPS-EQ scheme satisfies EUF-CMA and perfect adaption of signatures with respect to message space. We conclude Lemma 3.*

4.2 Auditable ABC from APK-SPE-EQ

Conventionally, an attribute-based credentials (ABC) scheme involves three types of participants: Issuer (also called organization), user, and verifier. An issuer grants credentials to a user on the user's attributes. The user can then prove possession of credentials with respect to her attributes to verifiers. The basic requirements of a secure ABC include correctness, anonymity, and unforgeability [7]. On a high level, correctness guarantees that verifiers always accept the showing of a credential if the credential is issued honestly; Anonymity prevents verifiers and (malicious) issuers (even by colluding) from identifying the user or exposing information during a showing against the user's will; Unforgeability requires that users (even by colluding) cannot perform a valid showing of attributes if the users do not possess credentials for the attributes.

The recent specifications of decentralized identifiers and verifiable credentials [53], [54] refueled the interest of the community in researching ABC schemes. New functionalities have been proposed to broaden the application of ABC schemes. Abstracted from the demands of contact tracing systems, we propose yet another functionality, *i.e.*, the auditability, that enables designated users to verify the particular *issuer* of a shown credential.

4.2.1 Formal Syntax of Our Auditable ABC

The starting point of our auditable ABC is [7] which supports selective showing on subsets of attributes. Then, we integrate APK by modifying the key generation algorithm of issuers and adding the auditing algorithm. Given a credential showing, the auditing algorithm with an auditing key outputs 1 or 0 to indicate whether the shown credential is issued by a secret key corresponding to the auditing key. We show the formal syntax of auditable ABC in the following.

Definition 26 (Auditable ABC Scheme) *An auditable ABC* AABC *consists of PPT algorithms* $(\mathsf{Setup}, \mathsf{OrgKGen}, \mathsf{UsrKGen})$*, two potentially interactive protocols* $\langle \mathsf{Obtain}, \mathsf{Issue} \rangle$ *and* $\langle \mathsf{Show}, \mathsf{Verify} \rangle$*, and a deterministic algorithm* Audit*. The participants in* AABC *perform as follows.*

- $\mathsf{Setup}(1^\lambda, q)$ *takes as input the security parameter* λ *and the size upper bound* q *of attribute sets. It outputs the public parameter* pp*;*

- $\mathsf{OrgKGen}(\mathsf{pp})$ *is executed by issuers.* $\mathsf{OrgKGen}$ *takes as input the public parameter* pp*. It outputs an issuer-secret and issuer-public key pair* $(\mathsf{osk}, \mathsf{opk})$ *with an auditing key* ak*. The issuer delegates* ak *to users (auditors) selected by herself (if there is none, the issuer is the auditor);*
- $\mathsf{UsrKGen}(\mathsf{pp})$ *is executed by users.* $\mathsf{UsrKGen}$ *takes as input the public parameter* pp*. It outputs a user-secret and user-public key pair* $(\mathsf{usk}, \mathsf{upk})$*. Later, we omit* pp *in algorithm inputs;*
- $\langle \mathsf{Obtain}(\mathsf{usk}, \mathsf{opk}, \mathbb{X}), \mathsf{Issue}(\mathsf{upk}, \mathsf{osk}, \mathbb{X}) \rangle$ *are PPT algorithms executed between a user and an issuer, respectively.* Obtain *takes as input the user-secret key* usk*, the issuer-public key* opk *and an attribute set* $\mathbb{X}$ *of size* $|\mathbb{X}| \leq q$*;* Issue *takes as input the user-public key* upk*, the issuer-secret key* osk *and the attribute set* $\mathbb{X}$*.* Obtain *returns* cred *on* $\mathbb{X}$ *to the user, and* $\mathsf{cred} = \perp$ *if protocol execution fails. The protocol outputs* (cred, I) *where* I *denotes the issuer's transcript;*
- $\langle \mathsf{Show}(\mathsf{opk}, \mathbb{X}, D, \mathsf{cred}), \mathsf{Verify}(D) \rangle$ *are executed between a user and a verifier, respectively, where* Show *is a PPT algorithm, and* Verify *is deterministic.* Show *takes as input an issuer-public key* opk*, an attribute set* $\mathbb{X}$ *of size* $|\mathbb{X}| \leq q$*, a non-empty set* $D \subseteq \mathbb{X}$ *representing the attributes to be shown, and a credential* cred*;* Verify *takes as input the set of shown attributes* D*.* Verify *returns* 1 *if the credential showing is accepted, or* 0 *otherwise. The protocol outputs* (S, b) *where* S *denotes the showing (user's transcript), and* $b \in \{0, 1\}$*. For convenience, we also write* $b \leftarrow \langle \mathsf{Show}, \mathsf{Verify} \rangle(S)$*;*
- $\mathsf{Audit}(\mathsf{ak}, S, \mathsf{opk})$ *is executed by a designated auditor with an auditing key* ak *such that corresponding issuer-key pair is* $(\mathsf{osk}, \mathsf{opk})$*.* Audit *also takes as input a showing of credential* $(S, \cdot) \leftarrow \langle \mathsf{Show}, \mathsf{Verify} \rangle$ *and the issuer-public key* opk*. It outputs* 1 *if the shown credential is issued with* osk*, or* 0 *otherwise.*

In addition to the auditing process, we make two modifications to the scheme from [7]. First, we write protocol transcriptions of $\langle \mathsf{Obtain}, \mathsf{Issue} \rangle$ and $\langle \mathsf{Show}, \mathsf{Verify} \rangle$ explicitly in our syntax concerning that the application in contact tracing may involve non-interactive proofs and require some transcripts to be publicly accessible (Section 5.2). In contrast, the previous works [7], [8] only mentioned them in security definitions.

Second, our Verify algorithm of $\langle \mathsf{Show}, \mathsf{Verify} \rangle$ takes as input only the attribute sets to be shown. In contrast, the original scheme also takes the issuer-public key opk of the Show algorithm. Their purpose is to prevent credentials from being issued by unidentified issuers. However, as shown in [8], the exposure of the issuer identity affects the users' anonymity. Although some previous works [8], [13] proposed the issuer-hiding property so that users can hide their credential issuers' identities within a list of identified

issuers, achieving such security incurs heavy mechanisms. Here, we rely on the Audit algorithm to provide an extra verification layer. That is, given an updated issuer-public key in a credential showing, the auditor who holds an auditing key *corresponding to an identified public key* must prove whether the shown credential is issued by the corresponding secret key.

4.2.2 Security Properties

We formally define correctness, anonymity, and unforgeability (two types) for our auditable ABC scheme. Concretely, correctness requires auditors to output 1 on any valid showing of credentials if the credential was issued by the corresponding secret key of the auditing key. The unforgeability game grants its adversary access to auditing keys. In the following, we omit pp if the algorithm takes as input other variables.

Definition 27 (Correctness) *An* AABC *scheme satisfies perfect correctness, if the following properties hold for any* $\lambda > 0, q > 0$*, any non-empty sets* $\mathbb{X}, D$ *such that* $|\mathbb{X}| \leq q$ *and* $D \subseteq \mathbb{X}$*, and* $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda, q), (\mathsf{osk}, \mathsf{opk}, \mathsf{ak}) \leftarrow \mathsf{OrgKGen}(\mathsf{pp}), (\mathsf{usk}, \mathsf{upk}) \leftarrow \mathsf{UsrKGen}(\mathsf{pp}), (\mathsf{cred}, \cdot) \leftarrow \langle \mathsf{Obtain}(\mathsf{usk}, \mathsf{opk}, \mathbb{X}), \mathsf{Issue}(\mathsf{upk}, \mathsf{osk}, \mathbb{X})\rangle$*: (1) the credential showing verifies, i.e.,* $(\cdot, 1) \leftarrow \langle \mathsf{Show}(\mathsf{opk}, \mathbb{X}, D, \mathsf{cred}), \mathsf{Verify}(D)\rangle$*; (2) if the credential showing is accepted, the auditing verifies, i.e.,* $\mathsf{Audit}(\mathsf{ak}, S, \mathsf{opk}) = 1$ *for any* $(S, 1) \leftarrow \langle \mathsf{Show}, \mathsf{Verify}\rangle$*.*

For anonymity and unforgeability, we follow the approach given by [7], in which adversaries can corrupt some participants. We first introduce the following lists and oracles to model the adversary.

Lists and oracles. At the beginning of each experiment, either the experiment generates the key tuple $(\mathsf{osk}, \mathsf{opk}, \mathsf{ak})$, or the adversary outputs opk. The sets HU, CU track all honest and corrupt users. We use the lists $\mathsf{USK}, \mathsf{UPK}, \mathsf{CRED}, \mathsf{attrs}, \mathsf{OWNER}$ to track user-secret keys, user-public keys, issued credentials with the corresponding attribute sets, and the users who obtain the credentials. In the anonymity games, we use $J_{\mathsf{LoR}}, I_{\mathsf{LoR}}$ to store the issuance indices and the corresponding users that have been set during the first query to the left-or-right oracle. The adversary is required to guess a bit b.

Considering a PPT adversary $\mathcal{A}$, the oracles are listed in the following. Note that we add the $\mathcal{O}_{\mathsf{Audit}}$ oracle for the unforgeability experiment.

- $\mathcal{O}_{\mathsf{HU}}(i)$ takes as input a user index i. If $i \in \mathsf{HU} \cup \mathsf{CU}$, the oracle returns $\perp$; Otherwise, it creates a new honest user i with $(\mathsf{USK}[i], \mathsf{UPK}[i]) \leftarrow$

$\mathsf{UsrKGen}(\mathsf{pp})$ and adds the user to the honest user list HU. It returns $\mathsf{UPK}[i]$ to the adversary.

- $\mathcal{O}_{\mathsf{CU}}(i, \mathsf{upk})$ takes as input i and (optionally) a user public key upk. If $i \in \mathsf{CU}$ or $i \in I_{\mathsf{LoR}}$, the oracle returns $\perp$; If $i \in \mathsf{HU}$, it moves i from HU to CU and returns $\mathsf{USK}[i]$ and $\mathsf{CRED}[j]$ for all j such that $\mathsf{OWNER}[j] = i$; If $i \notin \mathsf{HU} \cup \mathsf{CU}$, it adds i to CU and sets $\mathsf{UPK}[i] = \mathsf{upk}$.
- $\mathcal{O}_{\mathsf{ObtIss}}(i, \mathbb{X})$ takes as input i and a set of attributes $\mathbb{X}$. If $i \notin \mathsf{HU}$, the oracle returns $\perp$; Otherwise, it generates a credential with $(\mathsf{cred}, \top) \leftarrow \langle \mathsf{Obtain}(\mathsf{USK}[i], \mathsf{opk}, \mathbb{X}), \mathsf{Issue}(\mathsf{UPK}[i], \mathsf{osk}, \mathbb{X}) \rangle$. If $\mathsf{cred} = \perp$, the oracle returns $\perp$; Otherwise, it adds $(i, \mathsf{cred}, \mathbb{X})$ to $(\mathsf{OWNER}, \mathsf{CRED}, \mathsf{attrs})$ and returns $\top$.
- $\mathcal{O}_{\mathsf{Obtain}}(i, \mathbb{X})$ takes as input i and $\mathbb{X}$. If $i \notin \mathsf{HU}$, the oracle returns $\perp$; Otherwise, it runs $(\mathsf{cred}, \cdot) \leftarrow \langle \mathsf{Obtain}(\mathsf{USK}[i], \mathsf{opk}, \mathbb{X}), \cdot \rangle$ by interacting with the adversary $\mathcal{A}$ running Issue. If $\mathsf{cred} = \perp$, the oracle returns $\perp$; Otherwise, it adds $(i, \mathsf{cred}, \mathbb{X})$ to $(\mathsf{OWNER}, \mathsf{CRED}, \mathsf{attrs})$ and returns $\top$.
- $\mathcal{O}_{\mathsf{Issue}}(i, \mathbb{X})$ takes as input i and $\mathbb{X}$. If $i \notin \mathsf{CU}$, the oracle returns $\perp$; Otherwise, it runs $(\cdot, I) \leftarrow \langle \mathsf{Obtain}(\mathsf{USK}[i], \mathsf{opk}, \mathbb{X}), \cdot \rangle$ by interacting with the adversary $\mathcal{A}$ running Obtain. If $I = \perp$, the oracle returns $\perp$; Otherwise, it adds $(i, \perp, \mathbb{X})$ to $(\mathsf{OWNER}, \mathsf{CRED}, \mathsf{attrs})$ and returns $\top$.
- $\mathcal{O}_{\mathsf{Show}}(j, D)$ takes in the index j and a set of attributes D. Let $i = \mathsf{OWNER}[j]$, if $i \notin \mathsf{HU}$, the oracle returns $\perp$; Otherwise, it runs $(S, \cdot) \leftarrow \langle \mathsf{Show}(\mathsf{opk}, \mathsf{attrs}[j], D, \mathsf{CRED}[j]), \cdot \rangle$ by interacting with the adversary $\mathcal{A}$ running Verify.
- $\mathcal{O}_{\mathsf{Audit}}(S)$ is an oracle that holds public and auditing keys for all identified *issuers*. Given a showing transcript of a credential S, it runs $b \leftarrow \langle \mathsf{Show}, \mathsf{Verify} \rangle(S)$. If there exists opk and ak pair such that $\mathsf{Audit}(\mathsf{ak}, S, \mathsf{opk}) = 1$, the oracle returns $(\mathsf{opk}, b, 1)$ to the adversary; Otherwise, it returns $\perp$.
- $\mathcal{O}_{\mathsf{LoR}}(j_0, j_1, D; b)$ takes as input two issuance indices j_0, j_1, a set of attributes D and a challenge bit $b \xleftarrow{\$} \{0, 1\}$. If $J_{\mathsf{LoR}} \neq \emptyset$ and $J_{\mathsf{LoR}} \neq \{j_0, j_1\}$, the oracle returns $\perp$. Let $i_0 = \mathsf{OWNER}[j_0], i_1 = \mathsf{OWNER}[j_1]$. If $J_{\mathsf{LoR}} = \emptyset$, it sets $J_{\mathsf{LoR}} = \{j_0, j_1\}, I_{\mathsf{LoR}} = \{i_0, i_1\}$. If $i_0, i_1 \notin \mathsf{HU}$ or $D \nsubseteq (\mathsf{attrs}[j_0] \cap \mathsf{attrs}[j_1])$, the oracle returns $\perp$; Otherwise, it runs $(S_b, \cdot) \leftarrow \langle \mathsf{Show}(\mathsf{opk}_b, \mathsf{attrs}[j_b], D, \mathsf{CRED}[j_b]), \cdot \rangle$ by interacting with the adversary $\mathcal{A}$ running Verify.

Then, the formal definitions are as follows.

Definition 28 (Anonymity) *An* AABC *scheme satisfies anonymity if for any PPT adversary $\mathcal{A}$ that has access to oracles $\mathcal{O} = \{\mathcal{O}_{\mathsf{HU}}, \mathcal{O}_{\mathsf{CU}}, \mathcal{O}_{\mathsf{ObtIss}}, \mathcal{O}_{\mathsf{Issue}}, \mathcal{O}_{\mathsf{Show}}, \mathcal{O}_{\mathsf{LoR}}\}$, the following probability holds for any $\lambda, q > 0, \mathsf{pp} \leftarrow$*

$\mathsf{Setup}(1^\lambda, q)$*:*

$$\left| \Pr \left[\begin{array}{ll} (\mathsf{opk}_0, \mathsf{opk}_1, \mathsf{st}) \leftarrow \mathcal{A}(\mathsf{pp}); & \\ b \xleftarrow{\$} \{0,1\}; & : b^* = b \\ b^* \leftarrow \mathcal{A}^{\mathcal{O}}(\mathsf{st}) & \end{array} \right] - \frac{1}{2} \right| \leq \mathsf{negl}(\lambda).$$

Note that we modify the Verify in $\langle \mathsf{Show}, \mathsf{Verify} \rangle$ so that it does not take as input issuer-public keys. Hence, our anonymity also captures the indistinguishability of these keys. The definition above is arguably more close to the unlinkability from [13] because the $\mathcal{O}_{\mathsf{LoR}}$ oracle runs the Show algorithm with opk_b according to the challenge bit $b \xleftarrow{\$} \{0,1\}$.

Definition 29 (Unforgeability) *An* AABC *scheme satisfies unforgeability, if for any PPT adversary* $\mathcal{A}$ *that has access to oracles* $\mathcal{O} = \{\mathcal{O}_{\mathsf{HU}}, \mathcal{O}_{\mathsf{CU}}, \mathcal{O}_{\mathsf{ObtIss}}, \mathcal{O}_{\mathsf{Issue}}, \mathcal{O}_{\mathsf{Show}}, \mathcal{O}_{\mathsf{Audit}}\}$*, the following probability holds for any* $\lambda > 0, q > 0, \mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda, q)$*, and* $(\mathsf{osk}, \mathsf{opk}, \mathsf{ak}) \leftarrow \mathsf{OrgKGen}(\mathsf{pp})$

$$\Pr \left[\begin{array}{ll} (D, \mathsf{st}) \leftarrow \mathcal{A}^{\mathcal{O}}(\mathsf{opk}, \mathsf{ak}); & : b = 1 \wedge \textit{If } \mathsf{OWNER}[j] \in \mathsf{CU}, \\ (S, b) \leftarrow \langle \mathcal{A}(\mathsf{st}), \mathsf{Verify}(D) \rangle & \quad D \notin \mathsf{attrs}[j] \end{array} \right] \leq \mathsf{negl}(\lambda).$$

Like APK, unforgeability regarding to auditing keys is needed. A user should not recover the auditing key of a given public key even after querying the auditing oracles on other key tuples for polynomial times. Since the adversary can run key generation on its own in APK, the auditing unforgeability of auditable ABC is equivalent to the auditing key unforgeability in Definition 25.

4.2.3 Our Constructions and Analysis

Our auditable ABC construction has the same approach of [8], which is based on an SPS-EQ and a set-commit schemes. We extend their ABC construction with our APK-SPS-EQ.

Let BGGen be the bilinear group generation, $\mathsf{SC} = (\mathsf{Setup}, \mathsf{Commit}, \mathsf{Reveal}, \mathsf{OpenSubset}, \mathsf{VerifySubset})$ be the set-commitment [7] that satisfies correctness, binding, subset-soundness and hiding (definitions in Section 3.2.2), and ZKPoK be a general ZKPoK protocol that satisfies completeness, perfect zero-knowledge and knowledge-soundness (definitions in Section 3.2.3). With the necessary algorithms from our APK mechanism and the SPS-EQ [8], *i.e.*, $(\mathsf{KeyGen}, \mathsf{Update}, \mathsf{Audit}) \in \mathsf{APK}$ and $(\mathsf{Setup}, \mathsf{Sign}, \mathsf{ChgRep}, \mathsf{Verify}) \in \mathsf{SPSEQ}$, we show an auditable ABC AABC in the following. Note that the SPS-EQ scheme [8] utilizes a non-interactive zero-knowledge argument (which we take as a black-box) under the common reference string (CRS) model.

Construction 5 (Auditable ABC AABC) *The algorithms are as follows.*

- $\mathsf{Setup}(1^\lambda, q)$*: Run* $\mathsf{BG}\leftarrow\mathsf{BGGen}(1^\lambda)$ *where* $\mathsf{BG}=(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, P_1, P_2, e)$*. Sample* $a \xleftarrow{\$} \mathbb{Z}_p^*$ *and compute* $([a^i]_1, [a^i]_2)_{i\in[q]}$*. Sample matrices* $[\mathbf{A}]_2, [\mathbf{A}_0]_1$, $[\mathbf{A}_1]_1 \xleftarrow{\$} \mathcal{D}_1$*, and a common reference string* crs *for* SPSEQ*. Output* $\mathsf{pp}=(\mathsf{BG}, ([a^i]_1, [a^i]_2)_{i\in[q]}, ([\mathbf{A}]_2, [\mathbf{A}_0]_1, [\mathbf{A}_1]_1), \mathsf{crs}, \ell=3)$*;*
- $\mathsf{OrgKGen}(\mathsf{pp})$*: Output* $(\mathsf{osk}, \mathsf{opk}, \mathsf{ak}) \leftarrow \mathsf{APK.KeyGen}(\mathsf{BG}, [\mathbf{A}]_2, \ell)$ *and delegate* ak *to auditors selected by the issuer;*
- $\mathsf{UsrKGen}(\mathsf{pp})$*: Sample* $\mathsf{usk} \xleftarrow{\$} \mathbb{Z}_p^*$ *and output* $(\mathsf{usk}, \mathsf{upk} = \mathsf{usk}P_1)$*;*
- $\langle\mathsf{Obtain}, \mathsf{Issue}\rangle$ *and* $\langle\mathsf{Show}, \mathsf{Verify}\rangle$*: See Figure 4.3. In* $\langle\mathsf{Obtain}, \mathsf{Issue}\rangle$*, following the arguments in [8], we consider malicious issuer-keys and user-keys. Hence, both the issuer and the user should run a ZKPoK protocol to prove their public keys to each other, i.e.,* $\mathsf{ZKPoK}^{\mathsf{osk}}(\mathsf{opk})$ *and* $\mathsf{ZKPoK}^{\mathsf{usk}}(\mathsf{upk})$ *in Figure 4.3; Whereas, in* $\langle\mathsf{Show}, \mathsf{Verify}\rangle$*, the ZKPoK protocol, i.e.,* $(\pi_1, \pi_2, \pi_3)\leftarrow\mathsf{ZKPoK}^{(\mathsf{comm}, r\mathsf{comm}, P_1)}(\mathsf{comm}_1, \mathsf{comm}_2, \mathsf{comm}_3)$*, proves freshness to prevent transcripts of valid showings from being replayed by someone not in possession of the credential [7];*
- $\mathsf{Audit}(\mathsf{ak}, S, \mathsf{opk})$*: Parse* $S=(\mathsf{opk}', \mathsf{cred}', W; D)$*. Return* $\mathsf{APK.Audit}(\mathsf{ak}, \mathsf{opk}', \mathsf{opk})$*.*

Therefore, we have the following result.

Theorem 2 *The auditable ABC scheme* AABC *given by Construction 5 satisfies the following properties.*

- *Correctness (Definition 27);*
- *Anonymity (Definition 28) if the DDH assumption holds, the ZKPoK protocol has perfect zero-knowledge, the underlying APK satisfies indistinguishability and auditing key unforgeability, and the SPS-EQ scheme perfectly adapts signatures with respect to message space;*
- *Unforgeability (Definition 29) if the q-co-DL holds, the ZKPoK protocol has perfect zero-knowledge, the set-commitment* SC *satisfies subset-soundness, the APK satisfies secret key unforgeability, and SPS-EQ satisfies EUF-CMA;*
- *Auditing unforgeability (Definition 25) if the* $\mathcal{D}_1$*-KerMDH holds on* $\mathbb{G}_2$*.*

Proof *We show a brief proof here.* Correctness *follows directly from the correctness of building blocks. From now, we describe the proof rationale for anonymity, unforgeability, and auditing unforgeability properties, respectively. In general, we rely on [8] (Theorem 6 and 7) for properties of the ABC scheme and Theorem 1 for the APK part of our construction.*

The proof for anonymity (Definition 28) is an adaptation of the one given in Theorem 7 of [8]. Note there are two slight modifications to the anonymity definition in our work: (1) the adversary in the anonymity game generates two issuer-public keys $(\mathsf{opk}_0, \mathsf{opk}_1)$*; (2) the challenge oracle* $\mathcal{O}_{\mathsf{LoR}}$ *requires the adversary also to distinguish under which issuer-public key is the credential issued. To adjust the previous proof for these modifications, we first consider the two issuer-public keys* $(\mathsf{opk}_0, \mathsf{opk}_1)$ *and the updated issuer-public key in the credential showing, i.e.,* $\mathsf{opk}'_b \leftarrow \mathsf{APK.Update}(\mathsf{opk}_b)$ *where* $b \in \{0, 1\}$ *is the challenge bit in* $\mathcal{O}_{\mathsf{LoR}}$*. The adversary can win our introduced anonymity game if: (1) it can distinguish* opk'_0 *from* opk'_1*, then, the adversary can also win the* indistinguishability *game of the APK mechanism given in Definition 23; (2) it can recover the corresponding auditing keys* $(\mathsf{ak}_0, \mathsf{ak}_1)$ *from the issuer-public keys (hence, winning the anonymity game trivially by running* $\mathsf{APK.Audit}(\mathsf{ak}^*_b, \mathsf{opk}'_b, \mathsf{opk}_b)$*), which means the adversary wins the* auditing key unforgeability *game of the APK mechanism given in Definition 25. However, since the aforementioned APK properties have been proven in Theorem 1, our defined anonymity adversary gains no advantage over the anonymity adversary of [8].*

Next, by the proof in [8], the anonymity of ABC requires that the DDH assumption holds, the ZKPoK protocol has perfect zero-knowledge (which is taken as a black-box in this work), and the SPS-EQ perfectly adapts signatures with respect to message space (which has been proven in Lemma 3), thus we conclude the anonymity part for our auditable ABC scheme.

Similarly, our proof of unforgeability (Definition 29) adapts the one given in Theorem 6 of [8]. The only modification we make is that the unforgeability game provides its adversary with auditing keys. With additional auditing keys, the adversary can win the unforgeability game if it can recover the secret key of the underlying APK mechanism, which violates the secret key unforgeability *of APK as proven in Theorem 1. Then, by the proof in [8], the unforgeability of ABC requires that the q-co-DL assumption holds, the ZKPoK protocol has perfect zero-knowledge, the set-commitment scheme* SC *satisfies subset-soundness (which is also taken as a black-box in this work), and SPS-EQ satisfies EUF-CMA (which has been proven in Lemma 3), thus we conclude unforgeability for our auditable ABC scheme.*

Finally, for the auditing unforgeability, the adversary in the auditing unforgeability game aims to recover the issuer-public key's corresponding auditing key. Hence, the definition is equivalent to the auditing key unforgeability of APK (the same definition), which has been proven in Theorem 1.

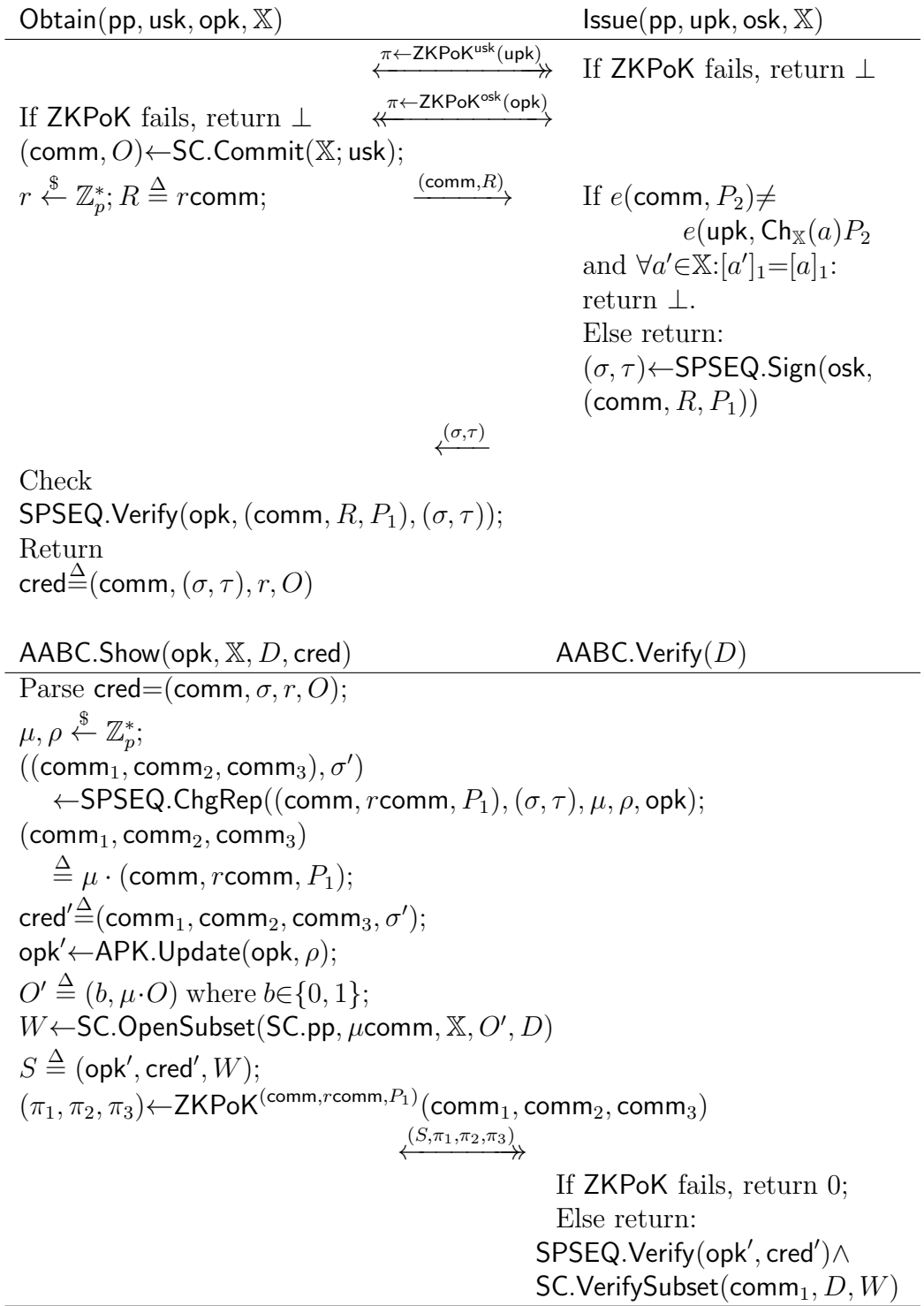


Figure 4.3: ⟨Obtain, Issue⟩, ⟨Show, Verify⟩ in AABC.

Chapter 5

The Enhanced Contact Tracing Frameworks

As we mentioned in the very beginning, this work aims to enhance contact tracing systems by taking epidemiology research results into consideration, *i.e.*, transmission models and the impact of environmental factors. By constructing ourselves an APK-BLS scheme that enables auditing and aggregation and an auditable ABC that has strong privacy guarantees, this chapter proposes our final contributions.

Concretely, we present two variants of environmental-adaptive contact tracing (EACT) framework based on the two primitives we constructed in previous chapters. The first one is under a decentralized model so that we eliminate interactions between users when recording contacts. However, due to the weakness in privacy, we failed to formalize security properties for the framework, but used a threat model instead. In contrast, the second framework is built atop our auditable ABC, which inherently requires communications between users. However, the underlying auditable ABC empowered us to give rigorous security definitions. We consider this trade-off an interesting topic and leave it as our future work.

5.1 A Decentralized EACT

There are three protocols in our EACT system: key management, recording, and tracing. For protocols, we present the formal definitions of algorithms.

A key management protocol is executed by users and authorized organizations. A user generates its initial public key according to parameters chosen by authorized organizations. This user updates the public key to obtain pseudonyms. This process can be done continuously alongside time slots.

The key management involves the following algorithms and a key registration sub-protocol.

- $\mathsf{Setup}(1^\lambda) \to \mathsf{pp}$. Run by authorized organizations, with security parameter λ, Setup outputs a public parameter pp. pp includes information to specify input/output spaces for other system algorithms;
- $\mathsf{KeyGen}(\mathsf{pp}) \to (\mathsf{pk}, \mathsf{sk}, \mathsf{ck})$. Run by a user, with public parameter pp, KeyGen outputs a public and secret key pair $(\mathsf{pk}, \mathsf{sk})$ for a signature scheme and a commitment key ck for a commitment scheme;
- $\mathsf{FormNym}(\mathsf{pp}, \mathsf{pk}, \mathsf{sk}, \mathsf{attrs}) \to \mathsf{nym}$. Run by a user, FormNym creates a pseudonym nym according to the input key pair $(\mathsf{pk}, \mathsf{sk})$ and time slot index $t \in \mathsf{attrs}$. The attributes may also contain location and other environmental factors. Users should continuously update their pseudonym.

In the key registration sub-protocol, the user commits its initial public key to authorized organizations with commitment key. An authorized organization registers the commitment if and only if the user holds the corresponding secret key and the user has never been registered before. We denote users' real-world identities as misc, *e.g.*, driver's licenses.

- $\mathsf{KeyCommit}(\mathsf{pp}, \mathsf{pk}, \mathsf{ck}) \to \mathsf{comm}$. Run by a user, KeyCommit outputs a commitment comm from a public key pk and a commitment key ck;
- $\mathsf{KeyReg}(\mathsf{pp}, \mathsf{comm}, \mathsf{misc}) \to \{0, 1\}$. Run by authorized organizations, with key commitment comm and the user's social identity misc, KeyReg returns $b = 1$ if and only if the user:

  1. computes $\mathsf{KeyCommit}(\mathsf{pp}, \mathsf{pk}, \mathsf{ck})$ correctly;
  2. knows the corresponding sk;
  3. has never used its misc in KeyReg before.

We use a (perfect hiding) commitment scheme on initial public keys instead of plain-text registrations because authorized organizations may leak users' registered public keys. Hence, a malicious user cannot take some initial public keys and generate pseudonyms on behalf of the victims without being noticed. In such a case, our system can prevent malicious users from flooding the bulletin board with fabricated contact records.

Moreover, we require users' social identities misc in registration to prevent users from registering multiple public keys to the system. It should be noticed that misc is sensitive and should be kept private. We present the way to prove the validity of users' public keys in Section 5.1.3.

Nest, the recording sub-protocol is executed by users. Users create and store records of contact or discrete location. It involves the following algorithms:

- $\mathsf{Assert}(\mathsf{pp}, \mathsf{nym}, \mathsf{sk}, \mathsf{attrs}) \rightarrow (\mathsf{assertion}, \pi_{\mathsf{A}})$. Assert outputs an assertion assertion and a proof π_{A}. The assertion embeds the user's nym and attrs. It may serve as a Bluetooth identifier when using the Bluetooth approach [20] for contact tracing;
- $\mathsf{VrfyAssert}(\mathsf{pp}, \mathsf{assertion}, \pi_{\mathsf{A}}) \rightarrow \{0, 1\}$. VrfyAssert validates assertion concerning π_{A}, returns 1 if valid, and returns 0 otherwise;
- $\mathsf{Save}(\mathsf{pp}, \{(\mathsf{assertion}, \pi_{\mathsf{A}})\}) \rightarrow \mathsf{compressedAssertion}$. With a list of assertions and the corresponding proofs as input, Save runs VrfyAssert on each pair. It compresses and saves the valid assertions to a compressedAssertion.

Notice that the Save algorithm takes in assertions regardless of their ownership. A user can run Save on its own sets of assertions. Henceforth, our framework is capable of both contact-tracing and discrete-location-tracing.

Finally, the tracing sub-protocol is executed by users and doctors with a bulletin board $\mathbb{B}$. Algorithms are as follows:

- $\mathsf{Show}(\mathsf{pp}, \mathsf{sk}, \mathsf{compressedAssertion}) \rightarrow (\mathsf{record}, \pi_{\mathsf{S}})$. Run by a user, with a locally compressed assertion compressedAssertion as input, Show rerandomizes it, then outputs a record record and a proof π_{S}. The proof should also prove the user's possession of its sk. Notice that users can run Show iteratively while showing multiple records;
- $\mathsf{VrfyShow}(\mathsf{pp}, \mathsf{record}, \pi_{\mathsf{S}}) \rightarrow \{0, 1\}$. Run by a doctor, VrfyShow validates record and checks if record embeds the user's sk concerning π_{S}. VrfyShow returns 1 if valid, and returns 0 otherwise;
- $\mathsf{Merge}(\mathsf{pp}, \{(\mathsf{record}, \pi_{\mathsf{S}})\}, \mathbb{B}) \rightarrow \mathbb{B}'$. Run by a doctor with two sub-routines:
  - $\mathsf{VrfyUser}(\mathsf{pp}, \mathsf{record}, \pi_{\mathsf{S}}) \rightarrow \{0, 1\}$. With a list of records and the corresponding proof, VrfyUser checks if record embeds the user's sk concerning π_{S}. It returns 1 if true, and returns 0 otherwise;
  - $\mathsf{VrfyShow}(\mathsf{pp}, \mathsf{record}, \pi_{\mathsf{S}}) \rightarrow \{0, 1\}$. With a list of records and the corresponding proof, VrfyShow checks if record is obtained correctly from the previous algorithms. It returns 1 if true, and returns 0 otherwise.

  Hence, Merge creates a collection of records from all the valid records and updates the bulletin board with $\mathbb{B}' = \mathbb{B}||\mathsf{records}$. The Merge algorithm can also take in the doctor's secret key $\mathsf{sk}_{\mathcal{D}}$ and update the bulletin board with $(\mathsf{records}, \sigma_{\mathcal{D}})$, where $\sigma_{\mathcal{D}} \leftarrow \mathsf{SIG.Sign}(\mathsf{sk}_{\mathcal{D}}, \mathsf{records})$ from a secure signature scheme SIG;
- $\mathsf{Trace}(\mathsf{pp}, \mathsf{sk}, \mathbb{B}) \rightarrow \{0, 1\}$. Run by a user with secret key sk, we phrase the current bulletin board with $\mathbb{B} = \mathsf{records}_0||\mathsf{records}_1||\ldots$. For each i and $\mathsf{record} \in \mathsf{records}_i$, if sk is involved in record, Trace returns 1, and

returns 0 otherwise. Additionally, Trace algorithm can output record and corresponding attributes.

Remark (Discrete Real-time Tracking) *The* Trace *algorithm behaves differently between contact tracing and discrete real-time tracking. Concretely, in contact tracing, a user checks if any of its secret keys are involved in* records *on the bulletin board. While in the discrete real-time tracking setting, records are based on the location history of diagnosed users, and other users need to check for collision concerning their history and the records. Such process only requires comparison of attributes* attrs *but no operations on cryptographic keys, and even no counter-party is required. Henceforth, no integrity can be achieved in the discrete real-time tracking setting.*

5.1.1 Threat Model

We rely only on authorized organizations in the key management protocol to register users' public keys. Henceforth, only initial public keys are known to authorized organizations, and we therefore regard the trust of authorized organizations as "out-of-band". We assume doctors are semi-honest, *i.e.*, following the protocol instructions but intending to collect as much information as possible. We assume adversarial users are malicious, *e.g.*, behaving arbitrarily for collecting sensitive data or generating fake contact records. However, as this work focuses on cryptographic attacks, we restrict the adversary from software engineering attacks, *e.g.*, turning off BLE or deleting records from the application. According to previous works [16], [23], we define the following threat model to our system.

- **Traceability Correctness.** Users can trace all their contacts with diagnosed users by records on the bulletin board;
- **Pseudonym Unlinkability.** A user's pseudonyms from different attributes cannot be linked with each other when being used in different close contacts;
- **Trace Unlinkability.** Pseudonyms in records on the bulletin board cannot be linked with any other pseudonyms of the same user by any other user, including medical doctors;
- **Integrity.** If the two users have no close contact, regardless of being diagnosed or not, one cannot trigger the other's tracing algorithm. Integrity only works for Bluetooth contact tracing but not for discrete real-time tracking;
- **Identity Authorization & Anonymity.** Users should not be able to submit records signed with unauthorized identity (*i.e.* public key). In the

meantime, such authorized identities should neither be personally identifiable nor be able to disclose anonymous users' full track.

The restrictions on adversaries are crucial. If one can go offline in the recording protocol or delete records in its storage, traceability correctness is unachievable due to the loss of records in such case.

The unlinkability preserves the anonymity of users' identities, and the integrity guarantees that no honest user's alert can be falsely triggered. Moreover, we argue that our system has the following property

- *Flooding Proof.* The Merge algorithm enables doctors to run a signature scheme before merging new records to the bulletin board, thus preventing adversaries from flooding the bulletin board with unauthenticated records;
- *Privacy Disclosure.* Hiding only users' identities may fail to preserve privacy when only very few diagnosed users exist in a small area. In such a situation, the user can be identified by the scope of activity, *i.e.*, according to the time and location data in its attributes.

5.1.2 System Overview

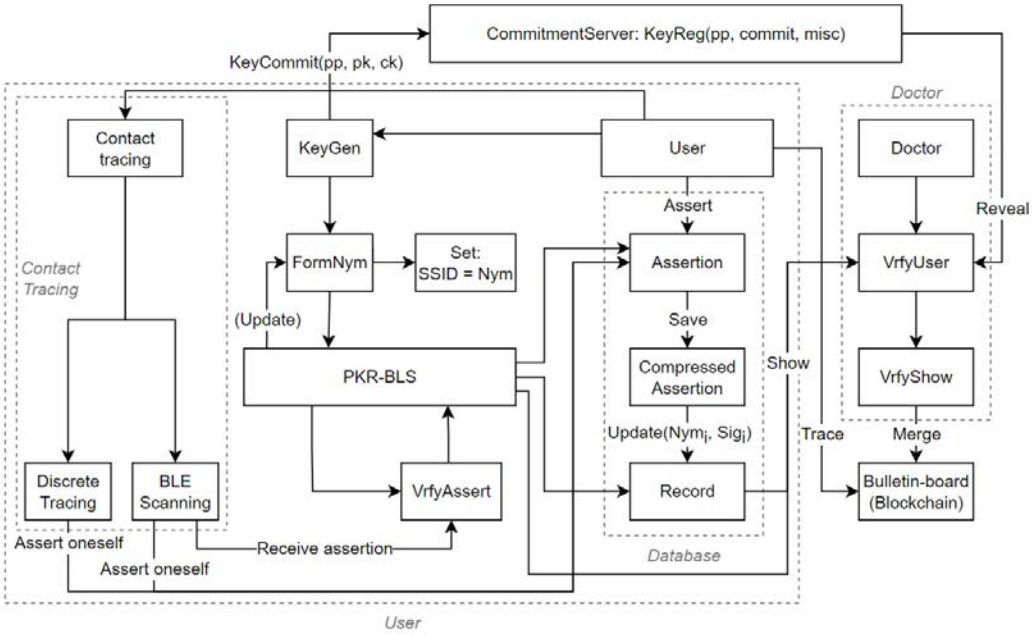
To summarize all components we have mentioned in previous chapters, in Figure 5.1, we clarify their relationships and behaviors.

User's components can be roughly divided into three categories — contact tracing, APK-BLS (See Section 4.1.1), and database. Contact tracing system is responsible for receiving assertions or generating discrete records, which contains all the attributes needed for judging whether the user is in danger or not. Assertions received will be fed into APK-BLS, and APK-BLS is responsible for all the cryptographic operations, such as key generation, pseudonym generation, pseudonym updating, assertion verification, etc. The database is used for storing assertions, compressed assertions, and records. Take note that "Record" is not included previously in Section 5.1.2 — we will explain why we need a new component from the output of aggregate key update later in Section 5.1.4.

The doctor's responsibilities include user verification, aggregate assertion verification, and merging (bulletin board).

Details of these components are explained in the following sections, and our implementation is also based on the design described in this section, which will be discussed in Section 5.1.4.

Figure 5.1: Overall Design of The Contact Tracing System



5.1.3 Our Construction

Our construction is based on a general commitment scheme Comm, a signature scheme SIG, and the APK-BLS scheme PKRBLS is as follows:

Construction 6 (Contact Tracing System) *Recall the system framework in Section 5.1.*

Key management protocol:

- Setup*: Generate parameters for the commitment scheme and the APK-BLS scheme, denoting them with* $\mathsf{Comm.pp}$ *and* $\mathsf{PKRBLS.pp}$*. Choose a hash function* $\mathsf{H}_1 : \{0,1\}^* \to \mathbb{Z}_p$*. Output* $\mathsf{pp} = (\mathsf{Comm.pp}, \mathsf{PKRBLS.pp}, \mathsf{H}_1)$ *where* $\mathsf{PKRBLS.pp} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, P_1, P_2, \mathsf{H}_0)$*;*
- KeyGen*: Run* $\mathsf{PKRBLS.KeyGen}(\mathsf{PKRBLS.pp}) \to (\mathsf{pk}, \mathsf{sk}_1, \mathsf{sk}_2)$ *and* $\mathsf{Comm.KeyGen}(\mathsf{Comm.pp}) \to \mathsf{ck}$*. Output* $(\mathsf{pk}, \mathsf{sk}_1, \mathsf{sk}_2, \mathsf{ck})$*, where* $\mathsf{pk} = (P_2, \mathsf{sk}_1 P_2, \mathsf{sk}_1 \cdot \mathsf{sk}_2 P_2)$*;*
- $\mathsf{FormNym}$*: Get time slot index* t*. Run* $\mathsf{H}_1(\mathsf{sk}_1, \mathsf{sk}_2, t) \to r$ *and* $\mathsf{PKRBLS.Update}(\mathsf{pk}; r) \to \mathsf{pk}'$*. Output* $\mathsf{nym} = \mathsf{pk}'$*;*
- ***Key registration sub-protocol:***

* KeyCommit*: Run* $\mathsf{Comm.Commit}(\mathsf{Comm.pp}, \mathsf{pk}, \mathsf{ck}) \rightarrow \mathsf{comm}$ *and output the commitment* comm*;*
* KeyReg*: Send* comm*, a proof of knowledge of relation* $\mathsf{comm} = \mathsf{Commit}(\mathsf{pk}, \mathsf{ck}) \wedge \mathsf{sk}_2\mathsf{pk}[1] = \mathsf{pk}[2]$*, and the user's social identities* misc *to the authorized organizations. If* misc *exists, return 0.*

Recording protocol:

- Assert*: Run* $\mathsf{PKRBLS.Sign}(\mathsf{sk}_1, \mathsf{sk}_2, \mathsf{attrs}; r) \rightarrow \sigma$*, where* r *is the randomness in* FormNym*. Set* π_{A} *as a proof of relation* $\mathcal{R}_1 : \{\mathsf{sk}_2\mathsf{nym}[1] = \mathsf{nym}[2]\}$ *where* $\mathsf{nym} = (P_2, r\mathsf{sk}_1 P_2, r\mathsf{sk}_1 \cdot \mathsf{sk}_2 P_2)$*. Output* $\mathsf{assertion} = (\mathsf{nym}, \mathsf{attrs}, \sigma)$ *and* π_{A}*;*
- VrfyAssert*: Phrase* $\mathsf{assertion} = (\mathsf{nym}, \mathsf{attrs}, \sigma)$ *and run* $\mathsf{PKRBLS.Vrfy}(\mathsf{nym}, \mathsf{attrs}, \sigma)$*. Verify relation* $\mathcal{R}_1$ *according to* π_{A}*. Output* $b = 1$ *if both verifications return 1;*
- Save*: For an assertion list* $\{\mathsf{assertion}\}$*, phrase the list with* $(\{\mathsf{nym}_i\}, \{\mathsf{attrs}_i\}, \{\sigma_i\})$*, where each* attrs_i *is unique. For each* i*, run* $\mathsf{PKRBLS.Aggre}(\sigma_i) \rightarrow \sigma$*. Output* $(\{\mathsf{nym}_i\}, \{\mathsf{attrs}_i\}, \sigma)$ *as* compressedAssertion.

Tracing protocol:

- Show*: Phrase* $\mathsf{record} = (\{\mathsf{nym}_i\}, \{\mathsf{attrs}_i\}, \sigma)$*, sample* $r' \xleftarrow{\$} \mathbb{Z}_p$ *and run* $\mathsf{PKRBLS.Update}((\{\mathsf{nym}_i\}, \sigma); r') \rightarrow (\{\mathsf{nym}'_i\}, \sigma')$*. Set* π_{S} *as a proof of relation* $\mathcal{R}_2 : \{\mathsf{nym}_{\mathcal{U}} \in \{\mathsf{nym}_i\} \wedge \mathsf{sk}_2\mathsf{nym}_{\mathcal{U}}[1] = \mathsf{nym}_{\mathcal{U}}[2]\}$ *where* $\mathsf{nym}_{\mathcal{U}} = (P_2, r\mathsf{sk}_1 P_2, r\mathsf{sk}_1 \cdot \mathsf{sk}_2 P_2)$*. Output* $\mathsf{record}' = (\{\mathsf{nym}'_i\}, \{\mathsf{attrs}_i\}, \sigma')$ *and* π_{S}*;*
- Merge*: Phrase* $\mathsf{record} = (\{\mathsf{nym}_i\}, \{\mathsf{attrs}_i\}, \sigma)$*, and run the following subroutines:*
  * VrfyUser*: Verify user's proof of possession on the record, i.e.,* $\mathcal{R}_2$*, according to* π_{S} *and the commitment scheme* Comm*;*
  * VrfyShow*: Verify with* $\mathsf{PKRBLS.VrfyAggre}$*.*

  If both verifications return 1, drop the signatures and denote the collection of valid records as $\mathsf{records} = \{(\{\mathsf{nym}_j\}, \{\mathsf{attrs}_j\})\}$*, which contains all* $\{\mathsf{nym}_j\}$ *and* $\{\mathsf{attrs}_j\}$ *in the valid records. Run* $\mathsf{SIG.KeyGen}(\mathsf{pp}) \rightarrow (\mathsf{pk}_{\mathcal{D}}, \mathsf{sk}_{\mathcal{D}})$ *and sign with* $\mathsf{SIG.Sign}(\mathsf{sk}_{\mathcal{D}}, \mathsf{records}) \rightarrow \sigma_{\mathcal{D}}$*. Output* BC' *as* $\mathsf{BC}||(\mathsf{records}, \sigma_{\mathcal{D}})$*;*
- Trace*: Phrase* $\mathsf{BC} = \mathsf{records}_0||\cdots$*. Run* $\mathsf{PKRBLS.VrfyKP}(\mathsf{nym}_j, \mathsf{sk}_2) \rightarrow b$*, for each* $\mathsf{records}_i = \{(\{\mathsf{nym}_j\}_i, \{\mathsf{attrs}_j\}_i)\} \in \mathsf{BC}$*. If* $b = 1$*, output* attrs *for each corresponding* nym_j*; else if* $b = 0$ *for all* nym*, output* 0*.*

Remark (Zero-Knowledge Proof-of-Knowledge (ZKPoK))
Notice that we use proof-of-knowledge protocols in three algorithms of Construction 6, i.e., KeyReg*,* VrfyAssert*, and* VrfyUser*. Intuitively, a ZKPoK enables provers to prove NP statements without revealing any knowledge to verifiers. In our case, a user intends to register itself to authorized organizations without revealing its public key. The user needs to prove its knowledge of the corresponding secret key when registering, contacting other users, and seeing a doctor. Thus, we prevent malicious users from spoofing honest ones. Specially, we require a set membership proof [55] in* VrfyShow *to prove that the user's pseudonym is included in its own records. Such property guarantees the integrity and preserves users' pseudonyms from being exposed to the doctor.*

5.1.4 Security Analysis

Construction 6 is traceability-correct if for any $\mathsf{nym} \in \mathsf{BC}$ and users' sk_2, we have $\mathsf{PKRBLS.VrfyKP}(\mathsf{nym}, \mathsf{sk}_2) = 1$, which is guaranteed by the update-correctness of the APK-BLS scheme.

For pseudonym unlinkability of the recording phase, users $\mathcal{U}_1$ and $\mathcal{U}_2$ have contact in time slots t_1 and t_2. $\mathcal{U}_1$ asserts with different pseduonyms and attributes to generate two signatures: $\sigma^{t_1} \leftarrow \mathsf{Sign}(\mathsf{nym}^{t_1}, \mathsf{attrs}^{t_1})$ and $\sigma^{t_2} \leftarrow \mathsf{Sign}(\mathsf{nym}^{t_2}, \mathsf{attrs}^{t_2})$. APK-BLS scheme's update-indistinguishability guarantees that $\mathcal{U}_2$ cannot distinguish nym^{t_1} from nym^{t_2} even after learning σ^{t_1} and σ^{t_2}.

For trace unlinkability of the tracing phase, we require two aspects:

1. For a doctor, we take a black-box use of a zero-knowledge protocol to prove the user's $(\mathsf{sk}_1, \mathsf{sk}_2)$ is corresponding to **one of** the pseudonyms in its records. Also, each user updates its records before showing them to doctors, thus, preserving the trace unlinkability from doctors;

2. For other users who run the Trace algorithm, by APK-BLS scheme's update-indistinguishability, pseudonyms on the blockchain are indistinguishable from those stored in their local storage. Thus, our protocol is able to preserve the trace unlinkability from normal users.

For integrity, although adversary can create assertions on arbitrary attributes, to win the integrity game, *i.e.*, to trigger an honest user $\mathcal{U}$'s Trace algorithm to output 1, a diagnosed adversary $\mathcal{A}$ has to create a record involving one of the honest user's assertion. However, we need to notice that $\mathcal{A}$ and $\mathcal{U}$ have no contact. Thus $\mathcal{A}$ does not know $\mathcal{U}$'s signature, nor can $\mathcal{A}$ forge one (by EUF-CMA). Moreover, after $\mathcal{A}$ learns $(\{\mathsf{nym}\}, \{\mathsf{attrs}\})$ from BC and

intends to forge a valid record $\mathsf{record}_{\mathcal{A}} = (\{\mathsf{nym}\} \cup \mathsf{nym}_{\mathcal{A}}, \{\mathsf{attrs}\} \cup \mathsf{attrs}_{\mathcal{A}}, \sigma_{\mathcal{A}})$, $\mathcal{A}$ will fail with significant probability due to the aggregation unforgeability (as the adversary needs to forge on multiple signatures). Thus, integrity is preserved due to the EUF-CMA secure and aggregation unforgeability of our APK-BLS scheme.

5.1.5 Implementation

Our work is involved with cryptographic algorithms and is expected to operate on mobile devices, whose performance is comparatively limited compared to traditional cryptographic scenarios. Therefore, we decide to implement our algorithm in the Android application to demonstrate the practicality and performance of our system[1].

In this section, we will explain how we implement our system, as illustrated in Figure 5.1, with focus on some engineering issues and how they affect the framework.

A user holds a Database, which includes assertion, compressedAssertion, and record. We will explain why we need 3 types of asserts later in this section. Public keys needs to be periodically updated to prevent the whole movement track from being exposed (*i.e.* FormNym). Database is used for storing assertions either 1) generated by a signature scheme (*e.g.*, APK-BLS scheme in Section 4.1.1) using attributes or 2) received from BLE scanning service and verified by the signature. The assertion(id, pk, attrs,σ, g, g^r) contains auto-generated device-wide unique assertion ID, pseudonym (*i.e.* nym), message (*i.e.* attrs), signature (*i.e.* σ), group generator (*i.e.* g) and contact tracing mode flag (*i.e.* isBLE). Although g is contained in nym, we still need it to convert pk from primitive data type (which can be stored in database) to abstract data type used for cryptographic calculations (*i.e.*, computing g).

Therefore, we cannot compute g directly from nym in primitive data type stored in the database, and that is why we need to make assertions contain g. Moreover, g^r is used for checking close contacts after collecting records from the blockchain.

Database also stores compressedAssertions aggregated from assertions by APK-BLS scheme. Theoretically, as described above, compressedassertion should contain an aggregated signature (*i.e.* PKRBLS.Aggre$(\{\sigma_i\}) \rightarrow \sigma$), a list of pseudonyms (*i.e.* $\{\mathsf{nym}_i\}$), a list of attributes (*i.e.* $\{\mathsf{attr}_i\}$), and a list of group generators (*i.e.* $\{g_i\}$). Since we need to prevent duplicate data stored in the device, we prefer storing pointers in compressedAssertions to assertions using primary keys (*i.e.* ids of assertions). Hence:

[1] `https://github.com/TaihouKai/PBK_Test`

$$\mathsf{compressedAssertion} = (\sigma, \{\mathsf{id}_i\}) \tag{5.1}$$

An infected user will update compressedAssertion using PKRBLS.Update, and therefore generate records, containing all associated assertions using ids, as the system needs to submit "real" data, instead of pointers.

Hence, user send (*i.e.* Show) a list of record and corresponding proof π_{S} to doctor. The doctor will use π_{S}, together with commitments associated with this user, which is registered to authorized organizations, to verify the user's identity (*i.e.* VrfyUser). The doctor also needs to check all assertions contained in each record to verify the relationship between committed public key and key used to sign submitted assertions.

As stated in Section 5.1.1, because the user's own assertions and received assertions cannot be distinguished due to security considerations, the system behaves differently for discrete and Bluetooth records. Since all discrete records are asserted by the user him/herself, the system only needs to verify all the nyms one by one. For Bluetooth assertions (*i.e.* $\mathsf{isBLE} = True$), the user needs to periodically compress assertions received with the user's own assertion, based on how many assertions are there left to be compressed. Therefore, if we assume the user operates compression for every N assertions received, the number of assertions belonging to the user him/herself should equal to $\frac{Number of Assertions}{N}$.

After identity verification, signature verification and merging have been explained explicitly. Therefore, we will not repeat them in this section.

It needs to be emphasized that, as our recording sub-protocol indicates, besides pseudonym, key-pair, and database, a user also holds secret explicit randomness r to generate signature after key updating. After the user updates the public key, if this user uses the previous randomness to generate a new signature, key and updated key's relationship can be easily referred if an attacker runs the verification function twice: to check if $\mathsf{PKRBLS.Vrfy}(\mathsf{nym}, \mathsf{attrs}, \sigma) = \mathsf{PKRBLS.Vrfy}(\mathsf{nym}', \mathsf{attrs}, \sigma)$.

As mentioned in the Show algorithm in Section 5.1, we require users to update their stored records, *i.e.*, users should update the pseudonym list and the corresponding aggregate signature with a newly sampled randomness. We guarantee that even if doctors misbehave and store previous users' records (thus know some pseudonyms), they cannot link those pseudonyms with any new-coming users. By the correctness of our signature scheme, doctors run VrfyAggre in their verification phase and the algorithm will output 1 if the aggregate signature is correct and is updated by the same randomness with its corresponding pseudonyms.

During tracing, our system can distinguish BLE records from discrete

Table 5.1: Experiment Results (Time in milliseconds)

Func. Frequent	Time Consump.	Func. Infrequent	Time Consump.
Assertion Gen.	99	Key Gen.	44
Assertion Verif.	163	Key Update	41
		Save	See Figure 5.2

* Average value of 100 attempts in Google Pixel 3A, Android 11.0 (R).
* Show and Show verification only happen once during medical treatment and thus we do not think its efficiency is worth worrying.

records using isBLE flag since BLE records and discrete records have different ways to trace. (See Section 5.1 and Section 2.2 for tracing mechanisms.)

We implemented our system on both Android Virtual Device (AVD) and physical devices (Samsung Galaxy Note 10+ & Google Pixel 3A[2])to prove its functionality and performance. Furthermore, our application utilizes the Java Pairing Based Cryptography Library (JPBC) [42], extending and modifying it based on our design.

There are roughly two types of functions in our contact tracing system — what needs to be performed frequently and what needs to be performed less frequently.

To be specific, KeyGen only needs to be performed once per user generation; Update needs to be performed periodically (*e.g.*, once per hour); Save needs to be performed less frequently than Update (*e.g.*, once per day); Show and its verification only happen during diagnosis. Since time consumption of these function is significantly shorter than time intervals between two launches, as shown in Table 5.1, we consider our system to be practical regarding these functions mentioned above.

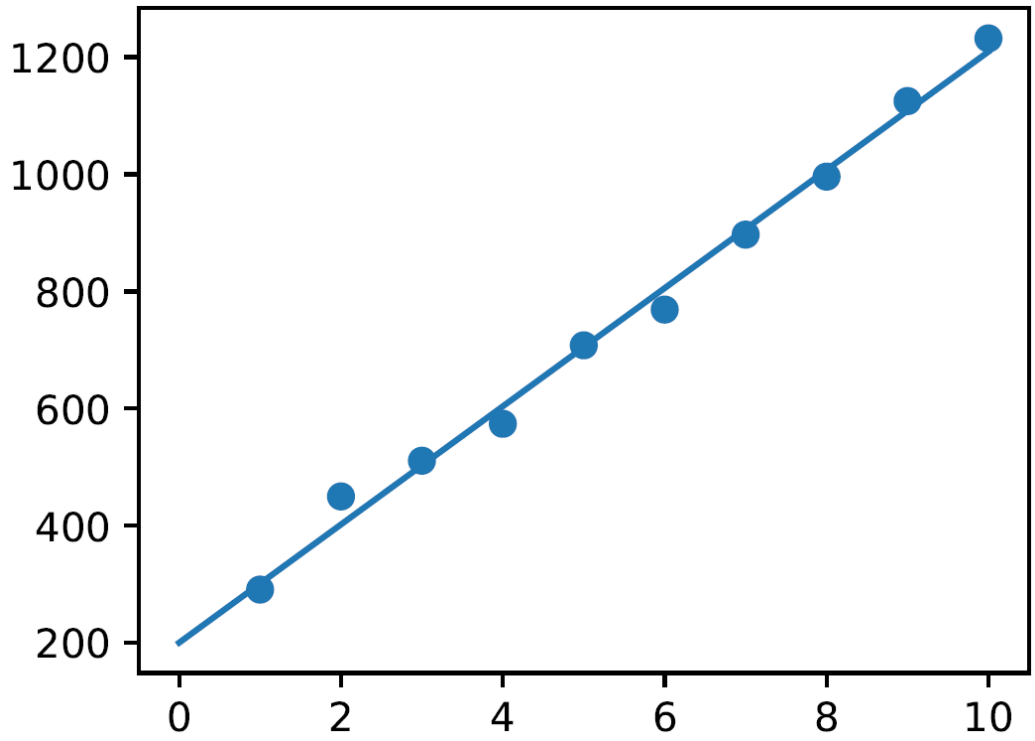
It needs to be mentioned that time consumption of Save depends on the amount of assertions the system trying to aggregate. We provide its time consumption with different amount of assertions in Figure 5.2.

However, the most time-consuming functions are expected to be performed frequently, *i.e.* assertion generation and assertion verification. Bluetooth scanning scenario requires both generation and verification, while discrete real-time tracking requires only assertion generation (since users do not care about other users' assertions under this scenario).

Consider an extreme case in which a user is tightly surrounded by 100

[2]We did not discover any fundamental difference in time consumption of our algorithms running in both of our physical devices, and therefore we only record time consumption of one of them, the one with worse performance – Google Pixel 3A.

Figure 5.2: Time Consumption of Save



X-axis — number of assertions are aggregated; Y-axis — time-consumption (in milliseconds). Each point denotes an average value of three 100-time attempts.

users [3]. In such a case, the discrete scenario will take (approx.) 13.5s to generate assertions, which is acceptable in our opinion. However, the Bluetooth scenario will take (approx.) 36.3s to generate and verify assertions, which is significantly slower. However, since Bluetooth scanning only happens in public transports as explained in Section 2.2, this user would have plenty of time to move together with the same group of surrounding users. At the same time, assertion generation and verification can be performed. Also, 36.3s is fast enough for the functions to be completed before the next stop.

Although Bluetooth scenario seems to take longer time to operate, we still expect our system to have notably better performance than current contact tracing systems in whatever scenario due to our system's interaction-less nature, since simultaneously handshaking with 100 users would cost a significant amount of time.

5.2 The Auditable ABC-Based EACT

From the perspective of credentials, we review the environmental-adaptive contact tracing (EACT) framework proposed in [24]. We provide a construction based on our auditable ABC scheme and argue that the game-based security definitions of auditable ABC suffice the requirements in contact tracing systems. Finally, we implement our construction to showcase its practicality.

5.2.1 Overview

We start by recalling the decentralized EACT framework above. It utilizes a bulletin board to store contact records, which can be instantiated by a blockchain protocol satisfying the robust ledger properties [57], *i.e.*, the capability of achieving immutable consensus atomically. Concerning different virus transmission modes (droplet and airborne), EACT considered tracing approaches via Bluetooth Low Energy (BLE) and self-reported discrete location (DLT). However, the framework cannot unify the tracing approach

[3]We decide not to conduct a real-world physical experiment involving such amount of users to evaluate the performance of the contact tracing systems because such interaction-based scanning heavily depends on transmission environment and devices, and letting hundreds of participants sticking tightly together during the pandemic violates social distancing rule and may cause group infection which would risk participants' personal health or even life, as COVID-19 may cause severe sequelae to infectees [56]. We cannot simply put devices together since the human body, movement and surrounding structures also interfere with Bluetooth communication. Therefore, such experiment is only held between two physical and two virtual devices and accumulated statistics is calculated using multiplication, by repeating the experiment between single pairs of devices.

in both settings because the recorded data are of different structures. As we will show later, ABC schemes enable us to circumvent this problem by regarding environmental and location data as *attributes*. Here, for completeness, we define a comparison algorithm to decide close contacts for BLE and DLT, *i.e.*, $\mathsf{Compare}_{\{\mathrm{BLE},\mathrm{DLT}\}}(\mathsf{envpp}, D, \mathbb{X})$ takes as input the environmental parameters envpp, an opened attributed set D (from other users, potentially downloaded from the bulletin board) and an attribute set $\mathbb{X}$ (of the user who runs the algorithm). We say the algorithm is "*well-defined*" if it outputs 1 when attributes in D and $\mathbb{X}$ are regarded as close contact concerning the tracing setting in $\{\mathrm{BLE}, \mathrm{DLT}\}$, and 0 otherwise.

The EACT framework involves three phases: key management, recording, and tracing, with two types of participants: user $\mathcal{U}$ and medical agency $\mathcal{D}$. We refine the algorithms with respect to our auditable ABC scheme. Intuitively, in the key management phase, users generate key pairs for participating as both issuers and regular users. The medical agency generates its key pair from a signature scheme. Users need to register their *issuer-public keys*, and medical agencies need to register their public keys. Then, in the recording phase, when users contact (two users in BLE or one user in DLT), we consider a pairwise executed $\langle \mathsf{Obtain}, \mathsf{Issue} \rangle$, *i.e.*, each user performs as an ABC issuer to grant its counterparty (itself in DLT) a credential on the attributes of current environmental data (or location data). This approach in the DLT setting can be easily adapted to the case in which the user can communicate to BLE beacons, providing additional evidence for the user's location data. Finally, in the tracing phase, whenever a user searches for medical treatment, she shows her records while the agency performs verification. The medical agency uploads records to the bulletin board (as we will show in tracing-soundness (Definition 31), we assume a malicious agency who can upload invalid records). Hence, users can refer to the bulletin board and audit if any shown credential is issued by themselves. Then, by comparing the environmental factors, they can detect close contact. The following section presents the full construction, including our modifications to the original framework.

5.2.2 Our Construction

Let $\mathsf{SIG} = (\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$ be a signature scheme that satisfies correctness and EUF-CMA (definitions given in Section 3.2.1), and let AABC be our auditable ABC construction given in Construction 5.

Construction 7 (Refined EACT REACT) *Our refined EACT framework involves three phases, i.e., Key management:* $(\mathsf{Setup}, \mathsf{OrgKGen}, \mathsf{UsrKGen},$

$\mathsf{MedKGen}, \mathsf{KReg}$); *Recording:* Exchange*; Tracing:* ($\langle \mathsf{Show}, \mathsf{Verify} \rangle, \mathsf{Merge}$, Trace). *The algorithms are performed as follows.*

- $\mathsf{Setup}(1^\lambda, q, \mathsf{envpp})$ *is run by the system where* envpp *denotes the environmental parameters. It runs* $\mathsf{AABC.pp} \leftarrow \mathsf{AABC.Setup}(1^\lambda, q)$ *and outputs* $\mathsf{pp} = (\mathsf{AABC.pp}, \mathsf{envpp})$.
- $\mathsf{OrgKGen}(\mathsf{pp})$ *is run by a user and outputs* $(\mathsf{osk}, \mathsf{opk}, \mathsf{ak}) \leftarrow \mathsf{AABC.OrgKGen}(\mathsf{pp})$. *Note that in contact tracing, we consider the user auditing for herself;*
- $\mathsf{UsrKGen}(\mathsf{pp})$ *outputs* $(\mathsf{usk}, \mathsf{upk}) \leftarrow \mathsf{AABC.UsrKGen}(\mathsf{pp})$*;*
- $\mathsf{MedKGen}(\mathsf{pp})$ *is run by a medical agency. It outputs a medical agent key pair with* $(\mathsf{msk}, \mathsf{mpk}) \leftarrow \mathsf{SIG.KeyGen}(1^\lambda)$. *Later, we omit* pp *in algorithm inputs;*
- $\mathsf{KReg}(\mathsf{pk}, \mathsf{misc}, \mathbb{B})$ *is a DID [53] black-box, which takes as input a public key* $\mathsf{pk} \in \{\mathsf{opk}, \mathsf{mpk}\}$*, auxiliary information* misc*, and a bulletin board* $\mathbb{B}$. KReg *registers* pk *with the corresponding* misc *on* $\mathbb{B}$.
- $\mathsf{Exchange}(\{(\mathsf{osk}_i, \mathsf{opk}_i), (\mathsf{usk}_i, \mathsf{upk}_i), \mathbb{X}_i\}_{i\in\{0,1\}})$ *is an interactive protocol executed between two users* $\mathcal{U}_0, \mathcal{U}_1$*, who may be identical, e.g., in the DLT setting. For* $i \in \{0, 1\}$*, both users perform* $(\mathsf{cred}_i, \cdot) \leftarrow \langle \mathsf{Obtain}(\mathsf{usk}_i, \mathsf{opk}_{1-i}, \mathbb{X}_i), \mathsf{Issue}(\mathsf{upk}_i, \mathsf{osk}_{1-i}, \mathbb{X}_i) \rangle$ *to grant each other a credential. The protocol outputs* cred_0 *and* cred_1 *for each user, respectively.*
- $\langle \mathsf{Show}, \mathsf{Verify} \rangle$ *is the showing and verification protocol in our auditable ABC, which here, is executed between a user* $\mathcal{U}$ *and a medical agency* $\mathcal{D}$*. The protocol outputs* $(S, b) \leftarrow \mathsf{AABC}.\langle \mathsf{Show}, \mathsf{Verify} \rangle$ *where* S *is a showing of the credential and* $b \in \{0, 1\}$*. Note that we explicitly add revealed attributes to* S*, i.e.,* $S \triangleq (\mathsf{opk}', \mathsf{cred}', W, D)$*. Moreover, we enable this protocol to process in batches, i.e., it can take a list of* n *credentials and verifies for each entry;*
- $\mathsf{Merge}(\mathsf{msk}, (S, b), \mathbb{B})$ *is run by a medical agency* $\mathcal{D}$*. If* $b = 1$*,* Merge *runs* $\sigma \leftarrow \mathsf{SIG.Sign}(\mathsf{msk}, S)$ *and outputs* $\mathbb{B}||(\mathsf{mpk}, S, \sigma)$*, or aborts otherwise;*
- $\mathsf{Trace}(\mathsf{ak}, \mathbb{X}, \mathbb{B})$ *is run by a user* $\mathcal{U}$ *with issuer-public and auditing keys* $\mathsf{opk}, \mathsf{ak}$*. It parses* $\mathbb{B} = \{(\mathsf{mpk}_j, S_j, \sigma_j)\}_{j\in[|\mathbb{B}|]}$*, and for each entry, parses* $S_j = (\mathsf{opk}'_j, \mathsf{cred}'_j, W_j, D_j)$*. Then, for each entry, it runs* $b \leftarrow \mathsf{SIG.Verify}(\mathsf{mpk}_j, S_j, \sigma_j)$*, and* $b' \leftarrow \mathsf{AABC.Audit}(\mathsf{ak}, S_j, \mathsf{opk})$ *(which is* $\mathsf{APK.Audit}(\mathsf{ak}, \mathsf{opk}'_j, \mathsf{opk})$*). For all* $j \in [|\mathbb{B}|]$ *such that* $b=1 \wedge b'=1$*, it compares according to environmental parameters and tracing settings, i.e.,* $b_j \leftarrow \mathsf{Compare}_{\{BLE,DLT\}}(\mathsf{envpp}, D_j, \mathbb{X})$*. If there exists any* j *that satisfies* $b_j = 1$*,* Trace *outputs* 1*; Otherwise, it outputs* 0.

5.2.3 Security and Analysis

We directly employ the cryptographic game-based security definitions from our auditable ABC scheme given in Section 4.2.1, including correctness, anonymity, and unforgeability. Moreover, we consider two separate properties for tracing, *i.e.*, traceability and tracing-soundness. At the end of this section, we will compare our ABC-based security definitions to the ones in [16].

The refined EACT requires signatures from medical agencies in Merge and on the bulletin board $\mathbb{B}$ (satisfying robust ledger properties [57]). Hence, we first formalize the tracing process correctness to capture these new requirements.

Definition 30 (Traceability) *Given the bulletin board* $\mathbb{B}$*, a* REACT *system satisfies traceability, if for any* $\lambda > 0, q > 0$*, any non-empty sets* $\mathbb{X}$ *with* $|\mathbb{X}| \leq q$*, and for any honest user* $\mathcal{U}$ *with a key tuple* $(\mathsf{osk}, \mathsf{opk}, \mathsf{ak}) \xleftarrow{\$} \mathsf{OrgKGen}(\mathsf{pp})$ *where* $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda, 1^q)$*, if there exists* $(\mathsf{mpk}, S, \sigma) \in \mathbb{B}$ *such that* $\langle \mathsf{Show}, \mathsf{Verify} \rangle(S) = 1$*,* $\mathsf{SIG.Verify}(\mathsf{mpk}, S, \sigma) = 1$*,* $D \in S$ *such that* $\mathsf{Compare}_{\{BLE,DLT\}}(\mathsf{envpp}, D, \mathbb{X}) = 1$*, then* $\Pr[\mathsf{Trace}(\mathsf{ak}, \mathbb{X}, \mathbb{B}) = 1] = 1$ *where* $\mathbb{X}$ *is the attribute set of* $\mathcal{U}$ *when she issues the credential being shown in* S*.*

Then, we have the following lemma.

Lemma 4 *Let the bulletin board satisfy the robust ledger properties [57]. The refined EACT* REACT *given by construction 7 satisfies traceability if* AABC *and* SIG *satisfy correctness, and the* $\mathsf{Compare}$ *algorithm is well-defined.*

The proof follows directly from the correctness of AABC and SIG, and the well-defined comparing algorithm $\mathsf{Compare}_{\{\mathrm{BLE,DLT}\}}(\mathsf{envpp}, \cdot, \cdot)$. Moreover, we require the bulletin board to satisfy the robust ledger properties [57] so that any entry on it cannot be erased or modified after a period of time.

Next, we consider the soundness of tracing, *i.e.*, the situation in which an honest user's Trace outputs 1 falsely. The PPT adversary $\mathcal{A}$ either: (1) forges a valid credential on behalf of honest users; or (2) colludes with a malicious medical agency so that arbitrary showings can be uploaded to the bulletin board. The first case has been captured by our unforgeability game in the auditable ABC scheme (Definition 29) with additional assumptions for the bulletin board, signature scheme, and comparing algorithm (like in Lemma 4).

However, the second one is dedicated to contact tracing. The reason lies in the different use cases, *i.e.*, in auditable ABC, auditors audit credential showings on behalf of the original issuer, hence, triggering the auditing algorithm of another auditor gains the adversary no benefits; whereas, in contact

tracing, it will cause false positive errors to the original issuer. In order to prevent such an attack, we require the proof in $\mathsf{AABC}.\langle\mathsf{Show},\mathsf{Verify}\rangle$ to be non-interactive. Concretely, since our bulletin board is instantiated by a blockchain, users can apply the Fiat-Shamir transformation on the head block of the blockchain, *i.e.*, embedding the hash of the blockchain's head block into their proof. Note that this approach also guarantees the freshness of the proof that prevents the relay and replay attacks [19]. This is because the adversary cannot guess the head of the blockchain priorly due to the security of blockchain protocols.

As shown in Theorem 2, the anonymity and unforgeability of AABC (also for REACT in Theorem 3) requires perfect zero-knowledge of ZKPoK. Hence, we must rely on heavy mechanisms, *e.g.*, [58], to make such a protocol non-interactive. An alternative way is to prove these theorems with computational zero-knowledge with a looser security reduction. The transformation to a non-interactive protocol with computational zero-knowledge can be achieved with the Fiat-Shamir heuristic [59] to trade security tightness for efficiency. Then, the showing of a credential becomes publicly verifiable so that even if a malicious medical agency falsely uploads credential showings to the bulletin board, every user (including the one who runs Trace) can verify the showing.

Compared to the unforgeability of auditable ABC in Definition 29, due to the malicious $\mathcal{A}_{\mathcal{D}}$ setting, tracing-soundness removes the requirement of $b = 1$ (the credential showing can be invalid) but embeds the proof of freshness (the showing must be presented at most once). We formally define tracing-soundness (with respect to malicious $\mathcal{A}_{\mathcal{D}}$).

Definition 31 (Tracing-Soundness) *Given the bulletin board $\mathbb{B}$, REACT system satisfies tracing-soundness (with respect to malicious $\mathcal{A}_{\mathcal{D}}$), if for any PPT adversary $\mathcal{A}$ that has access to oracles $\mathcal{O} = \{\mathcal{O}_{\mathsf{HU}}, \mathcal{O}_{\mathsf{CU}}, \mathcal{O}_{\mathsf{ObtIss}}, \mathcal{O}_{\mathsf{Issue}}, \mathcal{O}_{\mathsf{Show}}, \mathcal{O}_{\mathsf{Audit}}\}$, the following probability is less than $\mathsf{negl}(\lambda)$ for any $\lambda > 0, q > 0, \mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda, q, \mathsf{envpp})$, and $(\mathsf{osk}, \mathsf{opk}, \mathsf{ak}) \leftarrow \mathsf{OrgKGen}(\mathsf{pp})$:*

$$\Pr\left[\begin{array}{ll} (D, \mathsf{st}) \leftarrow \mathcal{A}^{\mathcal{O}}(\mathsf{opk}, \mathsf{ak}); & \mathsf{Trace}(\mathsf{ak}, \cdot, (S, \pi)) = 1 \wedge \\ ((S, \pi), b) \leftarrow \langle \mathcal{A}(\mathsf{st}), \mathsf{Verify}(D)\rangle : \textit{If } \mathsf{OWNER}[j] \in \mathsf{CU}, D \notin \mathsf{attrs}[j] & \end{array}\right],$$

where $\pi = (\pi_1, \pi_2, \pi_3) \leftarrow \mathsf{ZKPoK}^{(\mathsf{comm}, r\mathsf{comm}, P_1)}(\mathsf{comm}_1, \mathsf{comm}_2, \mathsf{comm}_3)$, and the variables are given in Figure 4.3 of auditable ABC construction.

Finally, we have the following theorem.

Theorem 3 *Our refined EACT REACT satisfies correctness, anonymity, unforgeability, traceability, and tracing-soundness (with respect to malicious $\mathcal{A}_{\mathcal{D}}$).*

Proof *The proofs of correctness, anonymity, and unforgeability follow directly from the underlying auditable ABC scheme, which has been given in Theorem 2. Traceability has been captured by Lemma 4. Hence, we only show the proof of tracing-soundness with respect to malicious* $\mathcal{A}_{\mathcal{D}}$.

Let $(\mathsf{mpk}, (S, \pi)\sigma) \in \mathbb{B}$ *be an entry stored on the bulletin board that triggers an honest user's tracing algorithm, i.e.,* $\mathsf{Trace}(\mathsf{ak}, \mathbb{X}, (\mathsf{mpk}, (S, \pi), \sigma)) = 1$. *Here,* ak *and* $\mathbb{X}$ *are the user's auditing key and attribute set when issuing the shown credential; whereas,* mpk *and* σ *are the public key and signature of the malicious* $\mathcal{A}_{\mathcal{D}}$. *Now, we consider the adversary's outputs,* $((S, \pi), b) \leftarrow \langle \mathcal{A}(\mathsf{st}), \mathsf{Verify}(D) \rangle$. *By traceability (Lemma 4), if* $b = 1$, *then the* Trace *algorithm will output* 1. *However, by the unforgeability of the underlying auditable ABC, the probability of the adversary outputting* $b = 1$ *is negligible of* λ. *In contrast, since we assume the medical agency* $\mathcal{A}_{\mathcal{D}}$ *to be malicious, it may approve invalid showing transcripts, i.e.,* $b = 0$, *to be uploaded to the bulletin board. However, as discussed above, the non-interactive proof* π *enhanced the showing transcript* S *to be publicly verifiable (freshness and validity). Hence, the user who runs* Trace *can reproduce* $b' \leftarrow \langle \mathsf{Show}, \mathsf{Verify} \rangle (S, \pi)$ *on herself. Considering the situation where* $b = 0$, *we should have* $b' = 0$. *Therefore,* Trace *will also output* 0, *i.e., the adversary fails to break tracing-soundness even if it colludes with a malicious medical agency.*

Comparison with existing game-based security definitions The authors [16] consider unlinkability and integrity. The former requires the adversary to distinguish two given pseudonyms, and the latter focuses on the false positive attack, *i.e.*, the adversary tries to trigger an honest user's tracing algorithm to output 1. Given the differences in syntax (their model [16] only considers pseudonyms; whereas, our system considers showing of credentials), we compare our definitions with theirs as follows.

Concretely, the unlinkability in [16] is further separated into pseudonym unlinkability (during the recording phase in which other users' pseudonyms are stored locally) and trace unlinkability (during the tracing phase in which recorded pseudonyms become publicly available). In comparison, the anonymity of our auditable ABC schemes (Definition 28) guarantees that no adversary can distinguish any two credential showings (similar to the revealed pseudonyms in trace unlinkability), even when the adversary has full control over issuing the challenge credentials. Note that the adversary can control issuer-public keys but not the corresponding auditing keys (this is guaranteed by the auditing key unforgeability of the underlying APK mechanism). Otherwise, it can trivially win the anonymity game by auditing the

updated public keys in credential showings (as we explained in the proof of Theorem 2). Moreover, user-public keys (similar to the pseudonyms in [16]) are generated with $\mathsf{AABC.UsrKGen}$, which are distributed uniformly on $\mathbb{G}_1$ (since usk is sampled uniformly at random from $\mathbb{Z}_p^*$). Hence, no adversary, as in the pseudonym unlinkability game, can distinguish any two distinct user-public keys in our system. Therefore, our anonymity captures both of the unlinkability definitions.

Next, the integrity in [16] provides its adversary with oracles to: (1) create new users; (2) generate, record, and send pseudonyms (during the recording phase, similar to credentials in our system); (3) generate and upload pseudonyms for tracing (during tracing phase, similar to our credential showings); (4) set time periods. In comparison, our tracing-soundness (Definition 31) provides similar oracle accessibility to the adversary, *i.e.*, (1) create new users; (2) issue and obtain credentials (contact records); (3) show credentials and upload the showing. Hence, it resembles the integrity in [16] except for the oracle that sets time for the adversary (this is because our system model is not period-specific like the one in [16]).

5.2.4 Implementation

We provide a proof-of-concept implementation for the refined EACT construction to prove its practicality on mobile devices with comparatively limited performance. The implementation uses Java/Kotlin for the raw Android environment. However, we also implement necessary functions since the Java Pairing-Based Cryptography (jPBC) library [42] cannot fully support matrix-based bilinear pairing operations. The library-level implementation, together with extended parts for jPBC [42] library, can also be found in our anonymous repository (`https://anonymous.4open.science/r/EAHT_MODULE_TEST`).

We implement $(\mathsf{Setup}, \mathsf{OrgKGen}, \mathsf{UsrKGen}, \mathsf{Exchange}) \in \mathsf{REACT}$. Moreover, in $\mathsf{Exchange}$, we need to measure the performance of algorithms and transmission (which is written in the form of $\mathsf{Transmit}(\cdot)$ for simplicity), separately. Hence, we further divide $\mathsf{Exchange}$ into $(\mathsf{Obtain\text{-}1}, \mathsf{Transmit}_1, \mathsf{AABC.Issue}, \mathsf{Transmit}_2, \mathsf{Obtain\text{-}2})$. The results are shown in Table 5.2.

Setup and $\mathsf{OrgKGen}$ are performance-insensitive because they only need to be executed once. We implement them merely to support other algorithms. Although we do not require user key pairs to be renewed once per contact, $\mathsf{UsrKGen}$ should be run periodically (*e.g.*, once per hour) to prevent a user's complete track under its public key from being exposed. We leave the setting of the renewal interval for real-life users to decide. Finally, for $\mathsf{Exchange}$, we consider the performance of $\mathsf{AABC.Obtain} = (\mathsf{Obtain\text{-}1}, \mathsf{Obtain\text{-}2})$, $\mathsf{AABC.Issue}$ and the time cost of data transmission, *i.e.*, $\mathsf{Transmit}(\pi, \mathsf{comm}, R)$ and

Table 5.2: Experiment Results (Time in milliseconds)

Algorithms	Time	Algorithms	Time	Algorithms	Time
Setup	168.99	Obtain-1	40.08	Transmit(σ, τ)	75.16
OrgKGen	54.18	Transmit(π, comm, R)	38.32	Obtain-2	164.81
UsrKGen	9.05	AABC.Issue	257.50	GenProof	0.26

Experiment device: Samsung SM-S9080 Android 12, Bluetooth 5.0 (Bluetooth Low Energy); Time consumption is calculated with the average of 100 attempts.

Transmit(σ, τ). A one-sided round trip, *e.g.*, $\mathcal{U}_0$ issuing a credential to $\mathcal{U}_1$ is performed with ($\mathcal{U}_0$.Obtain-1 $\longrightarrow$ $\mathcal{U}_0$.Transmit(π, comm, R) $\longrightarrow$ $\mathcal{U}_1$.Issue $\longrightarrow$ $\mathcal{U}_1$.Transmit(σ, τ) $\longrightarrow$ $\mathcal{U}_0$.Obtain-2) takes approximately 575.87 milliseconds in total. Consider the worst case, *e.g.*, when a crowded train is filled with 101 users. Each of them needs to Exchange with the other 100, hence, taking approximately 57.6 seconds to finish the execution and transmission. We consider this result to be reasonable and plausible, but obviously worse than EACT's performance data we observed in Table 5.1.

Chapter 6

Conclusion

In conclusion, within this work, we first identified some problems of existing contact tracing systems:

- Do not filter contacts, which would hence yield a extraordinary large number of contacts, and it is practically impossible to track and might flood local medical agencies;
- Do not take airborne transmission into consideration, which behaves differently than droplet transmissions;
- Do not prevent potential unauthorized access from service providers as these systems rely on trust rather than cryptographic algorithms;
- Do not ensure integrity of contact data, and hence users may modify their records to generate falseful contact records.

Therefore, we concluded that contact tracing systems needs to address problems above, and we hence designed the following two approaches: Environmental Adaptive Contact Tracing System, and Auditable Attribute-Based Credentials Scheme.

Environmental Adaptive Contact Tracing System Environmental Adaptive Contact Tracing System is based on decentralized anonymous credential. The construction is built upon the BLS signature [37] and the updatable public key mechanism [38]. This system uses environmental factors influencing particle dynamics, lifetime, to filter scanned results, which lowers the false positive rate. It is a variant of credential systems (decentralized anonymous credential), and thus it is able to carry arbitrary environmental factors. In order to include airborne transmission, we invented discrete

real-time tracking contact-tracing scheme. This system also implements a perfect hiding commitment scheme in order to disallow multiple identities of the same user.

Auditable Attribute-Based Credentials Scheme Auditable Attribute-Based Credentials Scheme is built upon auditable public keys (APK) mechanism, which is an extended version of the updatable public key mechanism [38] mentioned previously in EACT. APK adds a new component, auditing key, to the legacy public-private keypair, which allows auditing even if the public key has been updated. Then, we inserted APK into SPS-EQ scheme, maintaining SPS-EQ's security definition, to construct our modified SPS-EQ. We thus constructed our Auditable ABC scheme based on 1) such modified SPS-EQ described above, 2) a set-commitment scheme from [7], and 3) a zero-knowledge proofs of knowledge protocol [41]. Using such construction, we re-designed EACT framework and unified droplet transmission and airborne transmission's threat models, even holding the fact that these two transmissions' natures (interactive and non-interactive) are different.

By analyzing security of these two approaches with threat models (see Section 5.1.1 & 5.2.3), implementing them on Android devices, and comparing their performances (see Section 5.1.5 & 5.2.4), we concluded that, although Environmental Adaptive Contact Tracing System suffers from 1) incompatibility of 2 separate security definitions because of the difference in droplet and airborne transmission and 2) risky security assumptions, it has better performance in large-scale implementation. And although Auditable Attribute-Based Credentials Scheme solved security problems left in Environmental Adaptive Contact Tracing System, it suffers from potential performance issues because of its interactive nature.

We will hence leave the following 2 problems for researchers in the future to consider:

- A detailed definition of when these 2 approaches should be used, respectively;
- A new contact tracing system to merge these 2 approaches (Environmental Adaptive Contact Tracing System & Auditable Attribute-Based Credentials Scheme), which can eliminate their problems, while maintaining their advantages.

We sincerely hope that our cryptographic scheme can contribute to improved privacy and efficiency in the response to future epidemics, while fervently wishing that such circumstances never arise again.

Bibliography

[1] National Center for Immunization and Respiratory Diseases, "Scientific brief: Sars-cov-2 and potential airborne transmission," Centers for Disease Control and Prevention, Atlanta, Georgia, United States, Tech. Rep., Oct. 2020.

[2] X. Shen, *Personal information collected to fight covid-19 is being spread online in china*, [Online; accessed February 2021], May 2020. [Online]. Available: https://www.scmp.com/abacus/tech/article/3084064/personal-information-collected-fight-covid-19-being-spread-online-china.

[3] T. Starks, *Early covid-19 tracking apps easy prey for hackers, and it might get worse before it gets better*, [Online; accessed February 2021], Jul. 2020. [Online]. Available: https://www.politico.com/news/2020/07/06/coronavirus-tracking-app-hacking-348601.

[4] Z. Whittaker, *Fearing coronavirus, a michigan college is tracking its students with a flawed app*, [Online; accessed February 2021], Aug. 2020. [Online]. Available: https://techcrunch.com/2020/08/19/coronavirus-albion-security-flaws-app/.

[5] Signees of the Joint Statement, *Joint statement*, [Online; accessed February 2021], Apr. 2020. [Online]. Available: https://drive.google.com/file/d/1uB4LcQHMVP-oLzIIHA9SjKj1uMd3erGu/view.

[6] C. Garman, M. Green, and I. Miers, "Decentralized anonymous credentials," in *21st Annual Network and Distributed System Security Symposium, NDSS 2014, San Diego, California, USA, February 23-26, 2014*, The Internet Society, 2014. [Online]. Available: https://www.ndss-symposium.org/ndss2014/decentralized-anonymous-credentials.

[7] G. Fuchsbauer, C. Hanser, and D. Slamanig, "Structure-preserving signatures on equivalence classes and constant-size anonymous credentials," *J. Cryptol.*, vol. 32, no. 2, pp. 498–546, 2019. [Online]. Available: https://doi.org/10.1007/s00145-018-9281-4.

[8] A. Connolly, P. Lafourcade, and O. Perez-Kempner, "Improved constructions of anonymous credentials from structure-preserving signatures on equivalence classes," in *Public-Key Cryptography - PKC 2022 - 25th IACR International Conference on Practice and Theory of Public-Key Cryptography, Virtual Event, March 8-11, 2022, Proceedings, Part I*, G. Hanaoka, J. Shikata, and Y. Watanabe, Eds., ser. Lecture Notes in Computer Science, vol. 13177, Springer, 2022, pp. 409–438. [Online]. Available: https://doi.org/10.1007/978-3-030-97121-2%5C_15.

[9] J. Camenisch and A. Lysyanskaya, "An efficient system for non-transferable anonymous credentials with optional anonymity revocation," in *Advances in Cryptology - EUROCRYPT 2001, International Conference on the Theory and Application of Cryptographic Techniques, Innsbruck, Austria, May 6-10, 2001, Proceeding*, B. Pfitzmann, Ed., ser. Lecture Notes in Computer Science, vol. 2045, Springer, 2001, pp. 93–118. [Online]. Available: https://doi.org/10.1007/3-540-44987-6%5C_7.

[10] J. Camenisch and A. Lysyanskaya, "A signature scheme with efficient protocols," in *Security in Communication Networks, Third International Conference, SCN 2002, Amalfi, Italy, September 11-13, 2002. Revised Papers*, S. Cimato, C. Galdi, and G. Persiano, Eds., ser. Lecture Notes in Computer Science, vol. 2576, Springer, 2002, pp. 268–289. [Online]. Available: https://doi.org/10.1007/3-540-36413-7%5C_20.

[11] J. Camenisch and A. Lysyanskaya, "Signature schemes and anonymous credentials from bilinear maps," in *Advances in Cryptology - CRYPTO 2004, 24th Annual International CryptologyConference, Santa Barbara, California, USA, August 15-19, 2004, Proceedings*, M. K. Franklin, Ed., ser. Lecture Notes in Computer Science, vol. 3152, Springer, 2004, pp. 56–72. [Online]. Available: https://doi.org/10.1007/978-3-540-28628-8%5C_4.

[12] T. Silde and M. Strand, "Anonymous tokens with public metadata and applications to private contact tracing," in *Financial Cryptography and Data Security - 26th International Conference, FC 2022, Grenada, May 2-6, 2022, Revised Selected Papers*, I. Eyal and J. A. Garay, Eds., ser. Lecture Notes in Computer Science, vol. 13411, Springer, 2022, pp. 179–199. [Online]. Available: https://doi.org/10.1007/978-3-031-18283-9%5C_9.

[13] J. Bobolz, F. Eidens, S. Krenn, S. Ramacher, and K. Samelin, "Issuer-hiding attribute based credentials," in *Cryptology and Network Security*

- 20th International Conference, CANS 2021, Vienna, Austria, December 13-15, 2021, Proceedings, M. Conti, M. Stevens, and S. Krenn, Eds., ser. Lecture Notes in Computer Science, vol. 13099, Springer, 2021, pp. 158–178. [Online]. Available: https://doi.org/10.1007/978-3-030-92548-2%5C_9.

[14] M. Belenkiy, J. Camenisch, M. Chase, M. Kohlweiss, A. Lysyanskaya, and H. Shacham, "Randomizable proofs and delegatable anonymous credentials," in *Advances in Cryptology - CRYPTO 2009, 29th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2009. Proceedings*, S. Halevi, Ed., ser. Lecture Notes in Computer Science, vol. 5677, Springer, 2009, pp. 108–125. [Online]. Available: https://doi.org/10.1007/978-3-642-03356-8%5C_7.

[15] C. Hébant and D. Pointcheval, "Traceable constant-size multi-authority credentials," in *Security and Cryptography for Networks - 13th International Conference, SCN 2022, Amalfi, Italy, September 12-14, 2022, Proceedings*, C. Galdi and S. Jarecki, Eds., ser. Lecture Notes in Computer Science, vol. 13409, Springer, 2022, pp. 411–434. [Online]. Available: https://doi.org/10.1007/978-3-031-14791-3%5C_18.

[16] N. Danz, O. Derwisch, A. Lehmann, W. Pünter, M. Stolle, and J. Ziemann, "Security and privacy of decentralized cryptographic contact tracing," *IACR Cryptol. ePrint Arch.*, p. 1309, 2020. [Online]. Available: https://eprint.iacr.org/2020/1309.

[17] W. Beskorovajnov, F. Dörre, G. Hartung, A. Koch, J. Müller-Quade, and T. Strufe, "Contra corona: Contact tracing against the coronavirus by bridging the centralized-decentralized divide for stronger privacy," in *Advances in Cryptology - ASIACRYPT 2021 - 27th International Conference on the Theory and Application of Cryptology and Information Security, Singapore, December 6-10, 2021, Proceedings, Part II*, M. Tibouchi and H. Wang, Eds., ser. Lecture Notes in Computer Science, vol. 13091, Springer, 2021, pp. 665–695. DOI: 10.1007/978-3-030-92075-3_23. [Online]. Available: https://doi.org/10.1007/978-3-030-92075-3%5C_23.

[18] D. Bogatov, A. D. Caro, K. Elkhiyaoui, and B. Tackmann, "Anonymous transactions with revocation and auditing in hyperledger fabric," in *Cryptology and Network Security - 20th International Conference, CANS 2021, Vienna, Austria, December 13-15, 2021, Proceedings*, M. Conti, M. Stevens, and S. Krenn, Eds., ser. Lecture Notes in Computer Science, vol. 13099, Springer, 2021, pp. 435–459. [Online]. Available: https://doi.org/10.1007/978-3-030-92548-2%5C_23.

[19] A. Connolly, J. Deschamps, P. Lafourcade, and O. Perez-Kempner, "Protego: Efficient, revocable and auditable anonymous credentials with applications to hyperledger fabric," in *Progress in Cryptology - INDOCRYPT 2022 - 23rd International Conference on Cryptology in India, Kolkata, India, December 11-14, 2022, Proceedings*, T. Isobe and S. Sarkar, Eds., ser. Lecture Notes in Computer Science, vol. 13774, Springer, 2022, pp. 249–271. [Online]. Available: https://doi.org/10.1007/978-3-031-22912-1%5C_11.

[20] Apple Inc. and Google LLC, *Privacy-preserving contact tracing*, 2020. [Online]. Available: https://covid19.apple.com/contacttracing.

[21] F. AISEC, "Pandemic contact tracing apps: Dp-3t, PEPP-PT ntk, and ROBERT from a privacy perspective," *IACR Cryptol. ePrint Arch.*, p. 489, 2020. [Online]. Available: https://eprint.iacr.org/2020/489.

[22] R. Canetti, Y. T. Kalai, A. Lysyanskaya, *et al.*, "Privacy-preserving automated exposure notification," *IACR Cryptol. ePrint Arch.*, p. 863, 2020. [Online]. Available: https://eprint.iacr.org/2020/863.

[23] J. K. Liu, M. H. Au, T. H. Yuen, *et al.*, "Privacy-preserving COVID-19 contact tracing app: A zero-knowledge proof approach," *IACR Cryptol. ePrint Arch.*, p. 528, 2020. [Online]. Available: https://eprint.iacr.org/2020/528.

[24] P. Wang, X. Su, M. Jourenko, Z. Jiang, M. Larangeira, and K. Tanaka, "Environmental adaptive privacy preserving contact tracing system for respiratory infectious diseases," in *Cyberspace Safety and Security - 13th International Symposium, CSS 2021, Virtual Event, November 9-11, 2021, Proceedings*, W. Meng and M. Conti, Eds., ser. Lecture Notes in Computer Science, vol. 13172, Springer, 2021, pp. 131–144. [Online]. Available: https://doi.org/10.1007/978-3-030-94029-4%5C_10.

[25] S. K. Das, J.-E. Alam, S. Plumari, and V. Greco, "Transmission of airborne virus through sneezed and coughed droplets," *Physics of Fluids*, vol. 32, no. 9, p. 097 102, 2020.

[26] L.-D. Chen, "Effects of ambient temperature and humidity on droplet lifetime – a perspective of exhalation sneeze droplets with COVID-19 virus transmission," *International Journal of Hygiene and Environmental Health*, vol. 229, p. 113 568, Aug. 2020. DOI: 10.1016/j.ijheh.2020.113568.

[27] J. Daemen and V. Rijmen, "The block cipher rijndael," in *Smart Card Research and Applications, This International Conference, CARDIS '98, Louvain-la-Neuve, Belgium, September 14-16, 1998, Proceedings*, J. Quisquater and B. Schneier, Eds., ser. Lecture Notes in Computer Science, vol. 1820, Springer, 1998, pp. 277–284. DOI: 10.1007/10721064_26. [Online]. Available: https://doi.org/10.1007/10721064%5C_26.

[28] I. Levy, "High level privacy and security design for nhs covid-19 contact tracing app," National Cyber Security Centre, Victoria, London, England, United Kingdom, Tech. Rep., May 2020.

[29] W. Kim, H. Lee, and Y. D. Chung, "Safe contact tracing for COVID-19: A method without privacy breach using functional encryption techniques based-on spatio-temporal trajectory data," *PLOS ONE*, vol. 15, no. 12, H. Debiao, Ed., e0242758, Dec. 2020. DOI: 10.1371/journal.pone.0242758. [Online]. Available: https://doi.org/10.1371/journal.pone.0242758.

[30] Y. An, S. Lee, S. Jung, H. Park, Y. Song, and T. Ko, "Privacy-oriented technique for COVID-19 contact tracing (PROTECT) using homomorphic encryption: Design and development study," *Journal of Medical Internet Research*, vol. 23, no. 7, e26371, Jul. 2021. DOI: 10.2196/26371. [Online]. Available: https://doi.org/10.2196/26371.

[31] J. González-Cabañas, A. Cuevas, R. Cuevas, and M. Maier, "Digital contact tracing: Large-scale geolocation data as an alternative to bluetooth-based apps failure," *Electronics*, vol. 10, no. 9, 2021, ISSN: 2079-9292. DOI: 10.3390/electronics10091093. [Online]. Available: https://www.mdpi.com/2079-9292/10/9/1093.

[32] L. Garg, E. Chukwu, N. Nasser, C. Chakraborty, and G. Garg, "Anonymity preserving iot-based covid-19 and other infectious disease contact tracing model," *IEEE Access*, vol. 8, pp. 159 402–159 414, 2020. DOI: 10.1109/ACCESS.2020.3020513.

[33] M. Shrestha, X. Liu, and M. Sreekanthan, "A bluetooth-based contact-tracing mobile app for airborne-based epidemic control," in *2021 International Symposium on Networks, Computers and Communications (ISNCC)*, 2021, pp. 1–5. DOI: 10.1109/ISNCC52172.2021.9615701.

[34] C. for Disease Control and Prevention, *Transmission-based precautions*, Jan. 2016. [Online]. Available: https://www.cdc.gov/infectioncontrol/basics/transmission-based-precautions.html#anchor_1564058235.

[35] J. Morley, J. Cowls, M. Taddeo, and L. Floridi, *Ethical guidelines for covid-19 tracing apps*, May 2020. DOI: 10.1038/d41586-020-01578-0.. [Online]. Available: https://www.nature.com/articles/d41586-020-01578-0.

[36] Ministry of Health, "Covid-19 contact-confirming application," Labour and Welfare of Japan, Tokyo, Japan, Tech. Rep., Dec. 2020.

[37] D. Boneh, C. Gentry, B. Lynn, and H. Shacham, "Aggregate and verifiably encrypted signatures from bilinear maps," in *Advances in Cryptology - EUROCRYPT 2003, International Conference on the Theory and Applications of Cryptographic Techniques, Warsaw, Poland, May 4-8, 2003, Proceedings*, E. Biham, Ed., ser. Lecture Notes in Computer Science, vol. 2656, Springer, 2003, pp. 416–432. DOI: 10.1007/3-540-39200-9_26. [Online]. Available: https://doi.org/10.1007/3-540-39200-9%5C_26.

[38] P. Fauzi, S. Meiklejohn, R. Mercer, and C. Orlandi, "Quisquis: A new design for anonymous cryptocurrencies," in *Advances in Cryptology - ASIACRYPT 2019 - 25th International Conference on the Theory and Application of Cryptology and Information Security, Kobe, Japan, December 8-12, 2019, Proceedings, Part I*, S. D. Galbraith and S. Moriai, Eds., ser. Lecture Notes in Computer Science, vol. 11921, Springer, 2019, pp. 649–678. DOI: 10.1007/978-3-030-34578-5_23. [Online]. Available: https://doi.org/10.1007/978-3-030-34578-5%5C_23.

[39] A. Escala, G. Herold, E. Kiltz, C. Ràfols, and J. L. Villar, "An algebraic framework for diffie-hellman assumptions," in *Advances in Cryptology - CRYPTO 2013 - 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part II*, R. Canetti and J. A. Garay, Eds., ser. Lecture Notes in Computer Science, vol. 8043, Springer, 2013, pp. 129–147. [Online]. Available: https://doi.org/10.1007/978-3-642-40084-1%5C_8.

[40] P. Morillo, C. Ràfols, and J. L. Villar, "The kernel matrix diffie-hellman assumption," in *Advances in Cryptology - ASIACRYPT 2016 - 22nd International Conference on the Theory and Application of Cryptology and Information Security, Hanoi, Vietnam, December 4-8, 2016, Proceedings, Part I*, J. H. Cheon and T. Takagi, Eds., ser. Lecture Notes in Computer Science, vol. 10031, 2016, pp. 729–758. [Online]. Available: https://doi.org/10.1007/978-3-662-53887-6%5C_27.

[41] U. Feige and A. Shamir, "Zero knowledge proofs of knowledge in two rounds," in *Advances in Cryptology - CRYPTO '89, 9th Annual International Cryptology Conference, Santa Barbara, California, USA,*

August 20-24, 1989, Proceedings, G. Brassard, Ed., ser. Lecture Notes in Computer Science, vol. 435, Springer, 1989, pp. 526–544. [Online]. Available: https://doi.org/10.1007/0-387-34805-0%5C_46.

[42] A. D. Caro and V. Iovino, "Jpbc: Java pairing based cryptography," in *Proceedings of the 16th IEEE Symposium on Computers and Communications, ISCC 2011, Kerkyra, Corfu, Greece, June 28 - July 1, 2011*, IEEE Computer Society, 2011, pp. 850–855. [Online]. Available: https://doi.org/10.1109/ISCC.2011.5983948.

[43] Z. Y. Han, W. G. Weng, and Q. Y. Huang, "Characterizations of particle size distribution of the droplets exhaled by sneeze," *Journal of The Royal Society Interface*, vol. 10, no. 88, p. 20 130 560, Nov. 2013. DOI: 10.1098/rsif.2013.0560.

[44] National Fire and Protection Agency, *Nfpa-45 standard on fire protection for laboratories using chemicals*, Section 6-4.5, 2019.

[45] Cabinet of Japan, *Act on maintenance of sanitation in buildings*, Act No. 20 of 1970, 1970.

[46] United States Department of Labor, *Osha policy on indoor air quality: Office temperature/humidity and environmental tobacco smoke*, Standard Number: 1910.1000, Feb. 2003.

[47] Health and Safety Executive, *Workplace (health, safety and welfare) regulations 1992*, No.61 ACOP 7, 1992.

[48] Parlement français, *Code de l'énergie*, Article R241-26, 2016.

[49] National Health Commission of the People's Republic of China, *Norms for public health administration of central air-conditioning systems in public places*, WS394-2012, 2012.

[50] D. Boneh, B. Lynn, and H. Shacham, "Short signatures from the weil pairing," in *Advances in Cryptology - ASIACRYPT 2001*, ser. Lecture Notes in Computer Science, vol. 2248, Springer, 2001, pp. 514–532. DOI: 10.1007/3-540-45682-1_30. [Online]. Available: https://doi.org/10.1007/3-540-45682-1%5C_30.

[51] L. Ballard, M. Green, B. de Medeiros, and F. Monrose, "Correlation-resistant storage via keyword-searchable encryption," *IACR Cryptol. ePrint Arch.*, p. 417, 2005. [Online]. Available: http://eprint.iacr.org/2005/417.

[52] S. Goldwasser, S. Micali, and R. L. Rivest, "A digital signature scheme secure against adaptive chosen-message attacks," *SIAM J. Comput.*, vol. 17, no. 2, pp. 281–308, 1988. [Online]. Available: https://doi.org/10.1137/0217017.

[53] D. Reed, M. Sporny, M. Sabadello, and A. Guy, "Decentralized identifiers (DIDs) v1.0," W3C, W3C Recommendation, Jul. 2022.

[54] M. Jones, M. Sporny, O. Terbu, G. Cohen, and O. Steele, "Verifiable credentials data model v2.0," W3C, W3C Working Draft, Jul. 2023, https://www.w3.org/TR/2023/WD-vc-data-model-2.0-20230718/.

[55] J. Camenisch, R. Chaabouni, and A. Shelat, "Efficient protocols for set membership and range proofs," in *Advances in Cryptology - ASIACRYPT 2008, 14th International Conference on the Theory and Application of Cryptology and Information Security, Melbourne, Australia, December 7-11, 2008. Proceedings*, J. Pieprzyk, Ed., ser. Lecture Notes in Computer Science, vol. 5350, Springer, 2008, pp. 234–252. DOI: 10.1007/978-3-540-89255-7_15. [Online]. Available: https://doi.org/10.1007/978-3-540-89255-7%5C_15.

[56] National Institutes of Health, *When covid-19 symptoms linger*, Aug. 2021. [Online]. Available: https://covid19.nih.gov/news-and-stories/when-COVID-19-symptoms-linger.

[57] J. A. Garay, A. Kiayias, and N. Leonardos, "The bitcoin backbone protocol: Analysis and applications," in *Advances in Cryptology - EUROCRYPT 2015 - 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, April 26-30, 2015, Proceedings, Part II*, E. Oswald and M. Fischlin, Eds., ser. Lecture Notes in Computer Science, vol. 9057, Springer, 2015, pp. 281–310. [Online]. Available: https://doi.org/10.1007/978-3-662-46803-6%5C_10.

[58] J. Groth, R. Ostrovsky, and A. Sahai, "Perfect non-interactive zero knowledge for NP," in *Advances in Cryptology - EUROCRYPT 2006, 25th Annual International Conference on the Theory and Applications of Cryptographic Techniques, St. Petersburg, Russia, May 28 - June 1, 2006, Proceedings*, S. Vaudenay, Ed., ser. Lecture Notes in Computer Science, vol. 4004, Springer, 2006, pp. 339–358. [Online]. Available: https://doi.org/10.1007/11761679%5C_21.

[59] A. Fiat and A. Shamir, "How to prove yourself: Practical solutions to identification and signature problems," in *Advances in Cryptology - CRYPTO '86, Santa Barbara, California, USA, 1986, Proceedings*,

A. M. Odlyzko, Ed., ser. Lecture Notes in Computer Science, vol. 263, Springer, 1986, pp. 186–194. [Online]. Available: https://doi.org/10.1007/3-540-47721-7%5C_12.