Privacy-Preserving Digital Vaccine Passport

Thai Duong, Jiahui Gao, Duong Hieu Phan, Ni Trieu

Privacy-Preserving Digital Vaccine Passport

Thai Duong[1], Jiahui Gao[2], Duong Hieu Phan[3], and Ni Trieu[4]

[1] Google LLC, USA
thai@calif.io
[2] Arizona State University, USA
jgao76@asu.edu
[3] LTCI, Telecom Paris, Institut Polytechnique de Paris, France
hieu.phan@telecom-paris.fr
[4] Arizona State University, USA
nitrieu@asu.edu

Abstract. The global lockdown imposed during the Covid-19 pandemic has resulted in significant social and economic challenges. In an effort to reopen economies and simultaneously control the spread of the disease, the implementation of contact tracing and digital vaccine passport technologies has been introduced. While contact tracing methods have been extensively studied and scrutinized for security concerns through numerous publications, vaccine passports have not received the same level of attention in terms of defining the problems they address, establishing security requirements, or developing efficient systems. Many of the existing methods employed currently suffer from privacy issues.

This work introduces PPass, an advanced digital vaccine passport system that prioritizes user privacy. We begin by outlining the essential security requirements for an ideal vaccine passport system. To address these requirements, we present two efficient constructions that enable PPass to function effectively across various environments while upholding user privacy. By estimating its performance, we demonstrate the practical feasibility of PPass. Our findings suggest that PPass can efficiently verify a passenger's vaccine passport in just 7 milliseconds, with a modest bandwidth requirement of 480KB.

1 Introduction

As we navigate into the third year of the unprecedented global disruptions caused by the COVID-19 pandemic, there is a noticeable improvement in our circumstances. The accelerated development and widespread distribution of vaccines have played a vital role in expediting the pandemic's resolution and enhancing our preparedness for future outbreaks. However, it is crucial to recognize that privacy is equally significant in the context of vaccine passports as it is in contact tracing. Surprisingly, despite the multitude of proposals introduced by the scientific community last year for contact tracing, vaccine passports have not received the same level of attention when it comes to defining the problems they aim

to address, establishing security requirements, or developing efficient systems. Consequently, many of the current methods employed in vaccine passport implementation suffer from privacy issues. In this study, we specifically focus on the privacy implications of a vaccine passport rollout (the term "vaccine passport" is used here to emphasize the privacy concern, but the discussion applies to all types of digital certificates).

It is important to acknowledge that the incentives for the general public to adopt a vaccine passport are considerably higher compared to those for using a contact tracing app. The use of a vaccine passport directly and immediately benefits the passport holder by allowing them to resume a normal life. On the other hand, the purpose of a contact tracing application primarily revolves around reducing the circulation of the virus in a more abstract manner. This disparity may explain why people tend to be more skeptical about the privacy and effectiveness of contact tracing while being more supportive of vaccine passports.

A survey conducted by Ipsos [17] further sheds light on public sentiment. The survey, which encompassed over 21,000 individuals across 28 countries between March 26 and April 9, 2021, revealed that 78% of respondents supported the requirement of COVID-19 vaccine passports for travelers. Interestingly, the same survey found that, on average across the 28 countries, only 50% of individuals felt comfortable with their government accessing their personal health information, with the number dropping to 40% in the case of private companies. Meanwhile, amidst the evolving landscape, several governments have made the decision to implement vaccine passports. Notably, we examine instances within countries known for their commitment to privacy:

- European leaders have reached an agreement to expedite the implementation of an EU-wide Digital Green Certificate as a matter of utmost importance.
- Some European countries, including Denmark, Sweden, and Iceland, have taken the initiative to launch their own vaccine passports. Denmark, for instance, has introduced the "Coronapas" vaccine passport domestically, with potential plans to utilize it for international travel purposes as well.
- Estonia has introduced the VaccineGuard, a distributed data exchange platform, to issue vaccination certificates in adherence to the EU's green certificate proposal. Additionally, in collaboration with the World Health Organization (WHO), Estonia has been involved in the creation of a "smart yellow card," which serves as a global vaccine certificate.
- In the United Kingdom, the primary National Health Service (NHS) has undergone updates to facilitate the presentation of COVID-19 vaccination or test results by the public when traveling or attending public events.

Essentially, the aforementioned solutions involve issuing a certificate to individuals after they have been vaccinated. This certificate is signed by the relevant authority and can be utilized for various purposes. In the case of the EU-wide Digital Green Certificate [1], it is specified that the certificate incorporates a QR code that is protected by a digital signature to prevent counterfeiting. During the verification process, the QR code is scanned and the signature is validated. To safeguard user privacy, the certificate contains only a limited set of information

that cannot be retained by the visited countries, and all health data remains under the jurisdiction of the member state that issued the certificate. However, it is important to note that this approach necessitates placing trust in the issuing authority. If the QR code is compromised or marked, it becomes possible to trace the movements of an individual. Moreover, even if the QR code is solely scanned by a machine, a security breach could result in the linkage of an individual's entire movement history. The reliance on trust in the authority and the potential linkability of data raise significant privacy concerns. Therefore, it is imperative to urgently address privacy concerns through a privacy-by-design approach rather than relying solely on trust. This entails proposing methods that offer the highest level of privacy protection.

1.1 Our Contribution

In this work, we present PPass, an innovative solution for safeguarding privacy in digital vaccine passports. Our system combines robust security measures with low resource requirements, ensuring an efficient and cost-effective approach. To enable PPass, we introduce two cryptographic constructions that function seamlessly both online and offline during the passport verification process. These constructions are specifically designed to optimize performance on resource-limited devices, such as mobile phones, while accommodating a substantial user base. Moreover, our proposed constructions fulfill all security and privacy objectives, which we will elaborate on in subsequent discussions. In summary, our contributions can be summarized as follows.

1. Problem Definition and Desirable Properties: We provide a formal definition of the digital vaccine passport problem and outline the essential security and performance requirements for an ideal scheme. To the best of our knowledge, this work represents the first formal study of this problem.
2. Efficient Constructions for PPass: Leveraging an untrusted cloud server, we propose two efficient constructions for PPass, namely a digital signature-based construction for offline verification and a PIR-based construction for online verification. For each construction, we conduct a thorough analysis of their security properties and assess their computational and communication costs.
3. Performance Evaluation: To demonstrate the feasibility of PPass, we estimate its performance in practical scenarios. Remarkably, the computational requirements for the involved entities—the health authority, the client/user (phone's holder), and the service (verifier)—are lightweight during the passport verification process. Specifically:
   - The authority needs a mere 0.054ms per client to generate/sign a valid vaccination certificate, with its runtime scaling linearly with the number of clients.
   - The client's computation cost is constant per redeemed token, requiring up to 13 milliseconds to redeem a vaccine passport certificate.
   - In the PIR-based construction, the service's computation cost grows logarithmically with the number of valid tokens held by the cloud server.

However, in the signature-based construction, it remains constant per redeemed token, taking only 155 milliseconds.

Note that the cloud server bears the highest computation cost, but this can be mitigated by employing a more powerful machine or distributing the workload across multiple servers/cores. Importantly, PPass ensures no information leakage to the untrusted cloud server, enabling computation outsourcing without any privacy risks.

2 Problem Statement and Desirable Properties

In this section, we will elucidate the issue concerning digital vaccine passports that we aim to address. We will outline its security definition and expound upon the desirable properties of the system we propose.

Problem Definition. The problem at hand revolves around digital vaccine passports, which are mobile applications designed to verify an individual's vaccination status for a specific disease (e.g., COVID-19). The digital vaccine passport system comprises three primary participants: the client ($\mathcal{C}$), the health authority ($\mathcal{A}$), and the service or verifier ($\mathcal{S}$). When a client ($\mathcal{C}$) receives a vaccination, they obtain a vaccination certificate (σ) from a health authority ($\mathcal{A}$). The client ($\mathcal{C}$) can utilize this certificate (σ) to authenticate their vaccination status to a service without disclosing the actual certificate (σ) itself. The proof process involves leveraging information from the health authority ($\mathcal{A}$) that issued the certificate. In our proposed system, known as PPass, we employ an untrusted cloud server ($\mathcal{H}$) that performs the computational workload of the health authority ($\mathcal{A}$) to enhance system efficiency while ensuring the cloud server ($\mathcal{H}$) remains unaware of any sensitive information.

2.1 Security Definition

The vaccine passport system involves four types of participants: a client (or phone holder) $\mathcal{C}$, an authority $\mathcal{A}$, a cloud server $\mathcal{H}$, and a service (or verifier) $\mathcal{S}$. All participants have agreed upon a specific functionality, which is vaccine passport verification, and have consented to share the final result with a designated party. The computational process ensures that nothing is revealed except the final output.

For simplicity, we assume the presence of an authenticated secure channel (e.g., with TLS) between each pair of participants. In this work, we specifically focus on the semi-honest setting and the colluding model. In the ideal execution, the participants interact with a trusted party that evaluates the function while a simulator corrupts the same subset of participants. In the real-world execution, the protocol is performed in the presence of an adversary who can corrupt a subset of the participants. Privacy of the users is guaranteed as long as the adversary can only corrupt parties and does not compromise the authority server $\mathcal{A}$ and the service $\mathcal{S}$. Further details regarding the formal security definition and the security of our system can be found in Appendix D and Section 4, respectively.

2.2 Desirable Security

We outline the security and privacy requirements for the privacy-preserving digital vaccine passport system. One of the primary objectives is to ensure that the actions of honest clients, as well as other participants such as the authority server $\mathcal{A}$, cloud server $\mathcal{H}$, and service $\mathcal{S}$, are indistinguishable from each other. In other words, an ideal digital vaccine passport system would guarantee that executing the system in the real model is equivalent to executing it in an ideal model with a trusted party. This requirement aligns with the standard security definitions presented in [27]. Based on this definition, we consider the following security and privacy properties for the vaccine passport system:

- **Anonymous Identity:** The real identity of a client $\mathcal{C}$ should not be revealed to the untrusted cloud server $\mathcal{H}$. Furthermore, unless necessary, the service $\mathcal{S}$ should remain unaware of the client's identity. It is important to note that our PPass system does not maintain anonymity for clients if they willingly publish identifiable information. The authority $\mathcal{A}$ is only allowed to know the identity of a vaccinated client $\mathcal{C}$.
- **Token Unlinkability:** Valid tokens, generated from the same vaccination certificate σ, can be redeemed at multiple services $\mathcal{S}$. However, it should not be possible for any participant to link tokens belonging to the same client. Our PPass system does not guarantee token unlinkability if a group of services collude with the authority server $\mathcal{A}$.
- **Token Unforgeability:** All vaccination tokens must be unforgeable. A client should not be able to compute a valid token unless it corresponds to their valid vaccination certificate σ. Similarly, a client should not be able to compute a valid token generated from another client's vaccination certificate σ. Clients should be unable to redeem forged tokens, and any attempt to do so should be detected.
- **Token Unreusability:** Each valid token should be usable only once. Once a token is redeemed, it should be immediately deleted from the client's device. Clients and all participants, including services, should not be able to reuse redeemed tokens.

2.3 Desirable Performance

In addition to ensuring security and privacy, an ideal privacy-preserving digital vaccine passport system should possess certain performance requirements. We consider the following desirable performance properties:

- **Efficiency:** The digital vaccine passport system should be capable of processing a verification computation within a few seconds and should be scalable to accommodate a large number of users. Furthermore, participants, especially the authority $\mathcal{A}$ and the client $\mathcal{C}$, should perform lightweight tasks to ensure efficient operation.
- **Flexibility:** In certain scenarios where the client's ID is required to be collected by a service (e.g., at the airport), the vaccine passport system cannot maintain anonymous identity. Therefore, the system should be flexible

enough to provide a trade-off between performance and privacy in such cases. Similarly, the system can be optimized for efficiency in other scenarios where presenting an ID is not necessary.
- **Offline/Online Redeem:** In practice, a service may experience a slow network connection or be unable to connect to the internet during the verification process. The system should be designed to function correctly under different network conditions, supporting both offline and online redemption processes.

3 Cryptographic Preliminaries

This section introduces the notations and cryptographic primitives used in the later sections. For $n \in \mathbb{N}$, we write $[n]$ to denote the set of integers $\{1, \ldots, n\}$. We use '||' to denote string concatenation. In this work, the computational and statistical security parameters are denoted by κ, λ, respectively. Our PPass system is essentially based on the CDH or DDH assumption in a cyclic group [11].

3.1 Randomizable Signature Scheme

The use of digital signatures [25, 33] in various applications has been crucial, serving as a fundamental building block. With the integration of advanced features like randomizability, digital signatures have become even more valuable. This added functionality allows for the derivation of a new valid signature $\sigma^\star$ on the same message, given an original valid signature σ. Importantly, randomizability ensures that these two signatures remain unlinkable, even for the signer themselves. The initial construction achieving this property was proposed by Camenisch and Lysyanskaya [7], which has since been further enhanced by Pointcheval and Sanders [31, 32]. The use of randomizable signature schemes proves highly advantageous in our scenario as it enables the preservation of user privacy, even if the authority responsible for providing signed certificates is compromised.

Bilinear Group Setting. In the context of bilinear groups, a *bilinear group generator* $\mathcal{G}$ refers to an algorithm that takes a security parameter λ as input and produces a tuple $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, p, g_1, g_2, e)$. Here, $\mathbb{G}_1 = \langle g_1 \rangle$ and $\mathbb{G}_2 = \langle g_2 \rangle$ are cyclic groups of prime order p (a λ-bit prime integer), generated by g_1 and g_2 respectively. The function $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ is an *admissible pairing* satisfying the following properties:
- Bilinearity: For all $a, b \in \mathbb{Z}_p$, $e(g_1^a, g_2^b) = e(g_1, g_2)^{ab}$.
- Efficiency: e can be computed efficiently in polynomial time with respect to the security parameter λ.
- Non-degeneracy: $e(g_1, g_2) \neq 1$.

Additionally, the bilinear setting $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, p, g_1, g_2, e)$ is considered *asymmetric* when $\mathbb{G}_1 \neq \mathbb{G}_2$. There exist three types of pairings:

1. Type 1: $\mathbb{G}_1 = \mathbb{G}_2$.

2. Type 2: $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ is asymmetric, but an efficient homomorphism exists from $\mathbb{G}_2$ to $\mathbb{G}_1$, while no efficient homomorphism exists in the reverse direction.
3. Type 3: e is asymmetric, and no efficiently computable homomorphism exists between $\mathbb{G}_1$ and $\mathbb{G}_2$.

The Camenisch and Lysyanskaya signature scheme utilizes pairings of type 1, while the Pointcheval and Sanders signature scheme uses type 3 with a constant-size signature. The Pointcheval-Sanders scheme's unlinkability is based on the Decisional Diffie-Hellman (DDH) assumption in G_1, and its unforgeability relies on a complex assumption defined in [31]. In our PPass system, we rely on the Pointcheval-Sanders signature scheme because pairing type 3 offers the best performance among the three types.

3.2 Private Information Retrieval

Private Information Retrieval (PIR) allows a client to request information from one or multiple servers in such a way that the servers do not know which information the client queried. The basic concept of PIR is that the server(s) hold a database DB of N strings, and the client wishes to read data record $DB[i]$ without revealing the value of i. The construction of PIR [4] typically involves three procedures:

- $\mathsf{PIR.Query}(pk, i) \to k$: a randomized algorithm that takes the index $i \in [N]$ and public key pk as input and outputs an evaluated key k.
- $\mathsf{PIR.Answer}(pk, k, DB) \to c$: a deterministic algorithm that takes an evaluated key k, public key pk, and the DB as input and returns the response c.
- $\mathsf{PIR.Extract}(sk, c) \to d$: a deterministic algorithm that takes the private key sk and the response c as input and returns the desired data d.

A PIR construction is correct if and only if $d = DB[i]$. We say that PIR is (symmetric) secure if an evaluated key k reveals nothing about the index i and the answer c reveals nothing about other database record $DB[j], j \in [N], j \neq i$.

Keyword-PIR. A variant of PIR called keyword-PIR was introduced by Chor, et al. [8]. In keyword-PIR, the client has an item x, the server has a database DB, and the client learns whether $x \in DB$. The most efficient keyword PIR [4] is implemented using bucketing with Cuckoo hashing [12]. In this paper, we are interested in Keyword PIR based on 1-server PIR [4, 26, 29], but our protocol can use multiple-server PIR [6, 10, 28] to speed up the system's performance.

Similar to traditional PIR, a keyword-PIR construction [4] comprises four procedures. However, in keyword-PIR, the $\mathsf{PIR.Query}(pk, x)$ procedure takes a keyword x as input, and $\mathsf{PIR.Extract}$ returns a bit d indicating whether x exists in the server's database DB. Utilizing hashing techniques [4, 5], keyword-PIR exhibits similar computational and communication costs as traditional PIR. Angel et al.'s work [5, Figure 5] demonstrates that a PIR query on a database of size 2^{20} incurs approximately 7.62 milliseconds of client-side processing time and 80 milliseconds of server-side processing time (online time). Furthermore, the query requires 480KB of bandwidth for communication.

PIR-with-Default. Another variant of PIR, known as PIR-with-Default, was introduced by Lepoint et al. [20]. In PIR-with-Default, the server maintains a set of key-value pairs $P = (x_1, v_1), \ldots, (y_n, v_n)$, where y_i are distinct values and v_i are pseudo-random values. Additionally, there is a default (pseudo-random) value w. When the client submits an item x, it receives v_i if $x = v_i$, and w otherwise. The default value w needs to be refreshed for each query. This variant of PIR has found applications in private join and compute scenarios.

Similar to keyword-PIR, a PIR-with-Default construction also consists of the same procedures. However, in PIR-with-Default, the $\mathsf{PIR.Answer}(pk, k, P, w)$ procedure takes P and w as input, and $\mathsf{PIR.Extract}$ returns a value v. The PIR-with-Default protocol proposed by Lepoint et al. [20] is highly efficient, enabling 2^8 PIR with default lookups on a database of size 2^{20} with a communication cost of 7MB and an online computation time of 2.43 milliseconds.

3.3 Private Matching

A Private Matching (PM) is a two-party communication protocol where a sender possessing an input string m_0 interacts with a receiver who holds an input string m_1. The goal of this protocol is for the receiver to determine whether m_0 is equal to m_1, while ensuring that the sender learns nothing about m_1". The receiver obtains a single bit as output, indicating the equality result, but no additional information is revealed.

To the best of our knowledge, the concept of Private Equality Testing (PM) was first introduced in the works of Meadows [23] and FNW [14]. PM plays a crucial role in private set intersection (PSI) protocols [15]. Performing a batch of PM instances efficiently can be achieved using Oblivious Transfer (OT) extension techniques. For instance, a study by KKRT [19] demonstrates that the amortized cost of each PM instance, with an unbounded input domain $0, 1^*$, amounts to only a few symmetric-key operations and involves a communication of 488 bits. However, our protocol necessitates executing one PM instance at a time, rendering the construction of [19] unsuitable for our requirements. In our PPass system, we adopt a DH-based PM scheme, please refer Appendix E for the details.

4 Digital Vaccine Passport Constructions

We begin with describing the overview of our PPass system. We then present two cryptographic constructions: one for online verification and the other for offline verification.

4.1 System Overview

The purpose of a digital vaccine passport is to provide a means of verifying whether the individual holding the phone (referred to as the client) has been vaccinated for a specific disease. In this section, we present an overview of our proposed PPass system, which encompasses three primary procedures.

– `RegistrationRequest`$(\kappa, \text{inf}) \rightarrow \sigma$: The client $\mathcal{C}$ initiates this protocol by submitting a certificate request to the health authority $\mathcal{A}$. The $\mathcal{A}$ verifies whether the client has been vaccinated. If so, $\mathcal{A}$ generates a valid vaccination certificate σ and returns it to the client $\mathcal{C}$. Additionally, $\mathcal{A}$ sends certain anonymous information to the cloud server $\mathcal{H}$.
– `TokenGeneration`$(\sigma, n) \rightarrow \{tok_1, ..., tok_n\}$: The client $\mathcal{C}$ engages in this protocol with the cloud server $\mathcal{H}$ to generate a list of n vaccination tokens. The client $\mathcal{C}$ provides her vaccination certificate σ and specifies the number n, resulting in the generation of n tokens as output.
– `TokenRedeem`$(\text{tok}_t, \text{inf}) \rightarrow \{0, 1\}$: At time t, the client $\mathcal{C}$ redeems a token tok_t. The protocol takes as input the token tok_t and the client's information inf if required. Optionally, a service provider (verifier) $\mathcal{S}$ interacts with the cloud server $\mathcal{H}$ to verify the validity of tok_t and its association with the token holder. The output may be returned to the client $\mathcal{C}$. Once redeemed, the token tok_i becomes invalid.

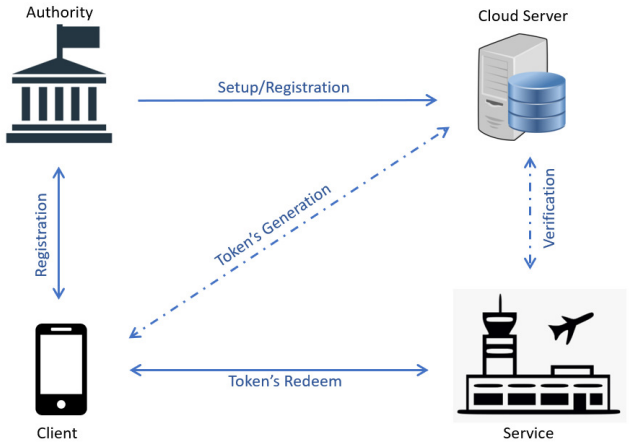

Fig. 1. The Overview of our PPass System. It consists of three main phases: Registration, Token's Generation, Token's Redeem. The solid and dashed lines show the required and optimal communication/connection between the participants, respectively.

4.2 PIR-based Construction (Online Verification)

In this section, we present the vaccine passport construction where a service requires to be online for verifying whether a token is valid. The construction heavily relies on different PIR variants.

4.2.1 Technical Overview. At a starting point, we consider a blueprint solution in which a client $\mathcal{C}$ obtains a vaccination certificate σ from the authority $\mathcal{A}$, after being vaccinated. When visiting a location and needing to demonstrate vaccination status, the client $\mathcal{C}$ presents the certificate σ to a service, which

securely communicates with the authority to validate σ in a privacy-preserving manner. A similar variation of this blueprint solution is currently implemented by the Smart Health Cards Framework, discussed in Section B.

Although the above solution provides a basic functionality for a digital vaccine passport, it falls short in terms of the desired security measures described in Section 2.2. For instance, it enables multiple compromised services to link tokens belonging to the same client. Moreover, the blueprint solution requires the authority to perform computationally intensive secure computations for token verification, contradicting the desired performance outlined in Section 2.3.

To meet the desirable security criteria of a vaccine passport system, we modify the blueprint construction to enable the client $\mathcal{C}$ to prove to the service $\mathcal{S}$ that she has been vaccinated, while keeping the vaccination certificate σ confidential. Specifically, we generate redeem tokens by computing a Pseudorandom Function (PRF), denoted as $tok \leftarrow F(\sigma, t)$, where t represents the token's redemption time. This process ensures that all generated tokens are unlinkable due to the underlying PRF, and each token can be redeemed individually. For certain locations, each token can be associated with an encryption of the client's ID to prevent unauthorized usage of a valid token by other clients.

Regarding the system's performance, we observe that the authority $\mathcal{A}$ can delegate its computations to an untrusted cloud server, denoted as $\mathcal{H}$. If the vaccination certificate σ is computed from a random key r and the client's information, revealing σ to $\mathcal{H}$ does not compromise privacy as long as r remains secret and known only to the authority $\mathcal{A}$. In our construction, $\sigma = F(r, I_2||...||I_n)$, where r is randomly chosen by the authority, I_1 represents the client's ID, and $I_i \in \{0,1\}^\star$ denotes additional information about the client's vaccine, such as the "type of vaccine" and "effective date." With the vaccination certificate σ, the cloud server $\mathcal{H}$ can generate a list T consisting of valid tokens $tok \leftarrow F(\sigma, t)$. Additionally, the authority $\mathcal{A}$ sends the cloud server $\mathcal{H}$ the group element $m = g^{H(I_1)}$ for anonymizing the user's identification where H is a one-way hash function. Based on the Diffie–Hellman assumption, m reveals no information about the client's actual ID.

To verify the token's validity, the service $\mathcal{S}$, possessing a token tok obtained from the client, aims to determine whether tok exists in the list of valid tokens held by the cloud server $\mathcal{H}$. This verification can be achieved using Keyword-PIR, as described in Section 3.2. Specifically, the service $\mathcal{S}$ sends a PIR request as $\mathsf{PIR.Query}(pk, tok) \rightarrow k$ and receives $\mathsf{PIR.Answer}(pk, k, T) \rightarrow c$ from the cloud server $\mathcal{H}$. By utilizing $\mathsf{PIR.Extract}(sk, c) \rightarrow 0, 1$, the service $\mathcal{S}$ can determine the validity of the token tok. Here, the public-private key pair pk, sk is generated by the service $\mathcal{S}$ using $\mathsf{PIR.Gen}(\kappa) \rightarrow (pk, sk)$.

Depending on whether the client's ID is required by the service, we consider two cases. In the first case, where the service $\mathcal{S}$ (e.g., an airline company) possesses the client's identity I_1 in clear, $\mathcal{S}$ can compute $m = g^{H(I_1)}$ and append it to the token as $tok||m$ before sending a PIR request. Similarly, the cloud server modifies T to include a set of $tok||m$ before returning a PIR answer to the service.

In the second case, where the service is not permitted to collect the client's ID, our construction relies on PIR-with-Default and Private Matching. Specifically, the cloud server $\mathcal{H}$ creates pairs (tok, m) and allows the service $\mathcal{S}$ to retrieve either m or a random default value using PIR-with-Default. Subsequently, the service $\mathcal{S}$ and the client $\mathcal{C}$, possessing m, engage in a private matching instance, leveraging the obtained PIR output, to determine whether the client redeemed a valid token generated from her vaccination certificate σ.

4.2.2 Construction. Figure 2 illustrates the construction of our PIR-based vaccine passport. The construction closely adheres to the technical overview described earlier. We organize the construction into three phases, aligning with the system overview detailed in Section 4.1. The first phase involves the computation and distribution of the vaccination certificate by the authority $\mathcal{A}$. In the second phase, each client $\mathcal{C}$ and the cloud server $\mathcal{H}$ independently generate valid tokens based on the obtained vaccination certificate. The final phase entails the redemption process of the tokens, wherein all participants except the authority $\mathcal{A}$ are involved (as the authority's role is limited to the first phase).

It is easy to see that correctness is obvious from the definitions of PIR variants, private matching, and Diffie–Hellman's assumption.

4.2.3 Security. We analyze the security of the proposed PIR-based construction according to our desirable security and privacy of a digital vaccine passport.

Anonymous Identity. To ensure anonymity, we demonstrate that the client's identity is not revealed to the cloud server. We assume that the corrupt cloud server $\mathcal{H}$ does not collude with the authority $\mathcal{A}$. The view of $\mathcal{H}$ includes the vaccination certificate σ, the exponentiation $m = g^{H(I_1)}$, and PIR transcripts. As $\mathcal{H}$ does not know the authority's secret value r, σ appears random to $\mathcal{H}$. Our construction relies on the difficulty of the discrete log problem. Therefore, given $m = g^{H(I_1)}$, $\mathcal{H}$ cannot recover the client's identity I_1.

We consider two cases: one where the service does not require collecting the client's identity I_1 but mandates presenting the ID (Step IV,3 in Figure 2), and another where presenting the client's identity is not required. In the former case, the view of the corrupt service $\mathcal{S}$ consists of the redeemed token tok'_t, PIR's and private matching transcripts. The token tok'_t is generated from the PRF key σ which is unknown to $\mathcal{S}$. Thus, tok'_t looks random to him. Because of PIR and private matching pseudorandomness property, the real identity of the client is protected. For the latter case, the analysis of anonymous identity security remains similar to the first one.

Token Unlinkability. Token unlinkability is crucial to prevent the disclosure of a user's travel history, safeguarding their privacy. In this section, we discuss how PPass ensures token unlinkability. We focus on the steps of the protocol and show the difficulty an attacker faces when attempting to link multiple individual tokens. We assume that clients use secure channels for communication with service providers and disregard attacks involving IP address matching.

PARAMETERS:

- A client $\mathcal{C}$, an authority $\mathcal{A}$, a service $\mathcal{S}$, and a cloud server $\mathcal{H}$.
- A cyclic group $G = \langle g \rangle$ of prime order p
- A PRF function $F : (\{0,1\}^{\kappa}, \{0,1\}^{\star}) \to \{0,1\}^{\kappa}$
- A hash function $H : \{0,1\}^{\star} \to \{0,1\}^{\kappa}$
- A Keyword-PIR, PIR-with-Default, and Private Matching primitives.

INPUTS: When a client $\mathcal{C}$ is vaccinated, it is associated with an information vector $\boldsymbol{I} = \{I_1, \ldots, I_n\} \in (\{0,1\}^{\star})^n$, where I_1 represent the client's ID, other $I_i \in \{0,1\}^{\star}$ represents information about the vaccine taken by the client such as "type of vaccine", "effective date", etc.

PROTOCOL:

I. **Registration Phase**
- The client $\mathcal{C}$ sends $\boldsymbol{I} = \{I_1, \ldots, I_n\}$ to the authority $\mathcal{A}$
- $\mathcal{A}$ chooses a random value r, computes $\sigma = F(r, I_2||...||I_n)$ and $m = g^{H(I_1)}$
- $\mathcal{A}$ distributes σ to both $\mathcal{C}$ and $\mathcal{H}$. Besides, $\mathcal{A}$ sends m to $\mathcal{H}$.

II. **Tokens Generation Phase**
- For each pair (σ, m) received from $\mathcal{A}$, the cloud server $\mathcal{H}$ computes a set T_m of valid tokens $F(\sigma, t)$, where t indicates the time of redeem (e.g. every 15 minutes).
- $\mathcal{C}$ generates a list of tokens $tok'_t = F(\sigma, t)$, where t is the redeem time.

III. **Token's Redeem Phase**: At the time t,

1. If presenting the client's identity I_1 is not required (light verification):
   - $\mathcal{C}$ sends tok'_t to $\mathcal{S}$
   - $\mathcal{S}$ and $\mathcal{H}$ involve a keyword-PIR instance:
     * $\mathcal{S}$ sends a PIR query $\mathsf{PIR.Query}(pk, tok'_t) \to k$ to $\mathcal{H}$
     * $\mathcal{H}$ replies $\mathsf{PIR.Answer}(pk, k, D) \to c$ to $\mathcal{S}$ where D is a set of tok, for all $tok \in T_m$
     * $\mathcal{S}$ outputs $\mathsf{PIR.Extract}(sk, c)$
2. If the client's identity I_1 is collected by the service $\mathcal{S}$
   - $\mathcal{C}$ sends (tok'_t, I_1) to $\mathcal{S}$ who computes $m' = g^{H(I_1)}$
   - $\mathcal{S}$ and $\mathcal{H}$ involve a keyword-PIR instance:
     * $\mathcal{S}$ sends a PIR query $\mathsf{PIR.Query}(pk, tok'_t||m') \to k$ to $\mathcal{H}$
     * $\mathcal{H}$ replies $\mathsf{PIR.Answer}(pk, k, D) \to c$ to $\mathcal{S}$ where D is a set of $tok||m$, for all $tok \in T_m$
     * $\mathcal{S}$ outputs $\mathsf{PIR.Extract}(sk, c)$
3. If presenting the client's identity I_1 is required, but the service does not allow to collect the client's identity I_1
   - $\mathcal{C}$ sends tok'_t to $\mathcal{S}$
   - $\mathcal{S}$ and $\mathcal{H}$ involve a PIR-with-Default instance:
     * $\mathcal{S}$ sends a PIR query $\mathsf{PIR.Query}(pk, tok'_t) \to k$ to $\mathcal{H}$
     * $\mathcal{H}$ replies $\mathsf{PIR.Answer}(pk, k, D, w) \to c$ to $\mathcal{S}$ where D is a set of pairs (tok, m), for all $tok \in T_m$, and w is a zero string.
     * $\mathcal{S}$ computes $\mathsf{PIR.Extract}(sk, c) \to v$
   - If v is a zero string, the $\mathcal{S}$ outputs 0 (i.e. tok'_t is invalid). Otherwise, the service $\mathcal{S}$ and the client $\mathcal{C}$ involve a private matching instance:
     * $\mathcal{C}$ acts as a sender with input $m' = g^{H(I_1)}$.
     * $\mathcal{S}$ acts as a receiver with input v and output whether $v = m'$.

Fig. 2. Our PIR-based Vaccine Passport Construction.

– Phase 1. Registration Phase: During registration, the client and the cloud server communicate with the authority $\mathcal{A}$ to obtain a vaccination certificate σ and a value m. It is impractical for an attacker to recover the client's information $I_2, \ldots, I_n$ from the PRF value without the authority's secret value r, unless $\mathcal{A}$ is compromised. Additionally, due to the Diffie-Hellman assumption, the value m appears random to the attacker.
– Phase 2. This phase involves local computation by individual clients and the cloud server, without any communication or computation between the participants. Hence, no information is leaked. However, if an attacker compromises the cloud server, they can identify which tokens are generated from the same vaccination certificate σ, but they cannot determine where the tokens are redeemed (Phase 3).
– Phase 3. Token's Redeem Phase: During this phase, if an attacker controls a subset of service providers $\mathcal{S}$ who collect the client's IDs, PPass cannot provide unlinkability. However, if the attacker collects a list of redeemed tokens from different service providers $\mathcal{S}$, all the redeemed tokens appear random, even if they were generated from the same key σ. Furthermore, each token is designed for one-time use only. Therefore, no linkability can be established between tokens. If the attacker compromises the cloud server, they also gain no information due to the PIR ideal functionality. The cloud server does not know which tokens were redeemed at which places.

Token Unforgeability. According to our construction, if a token *tok* is not generated from a valid vaccination certificate, the service is able to detect this event via PIR. Recall that the cloud server $\mathcal{H}$ has a set of valid tokens, PIR functionality allows the service to check whether *tok* is in the $\mathcal{H}$'s database. Moreover, PIR-with-default allows $\mathcal{S}$ to retrieve anonymous information of the client's identity m. Private matching between $\mathcal{S}$ and $\mathcal{C}$ prevents an attacker to redeem tokens of another client.

Token Unreusability. Each token is associated with a specific redeem time, which prevents an attacker from reusing the token later. However, PPass cannot prevent an attacker from attempting to redeem the same token simultaneously at two different service providers, unless ID presentation is required. Therefore, we rely on end-user devices to delete the token after redemption. To eliminate the need for trust in end-user devices, secure deletion can be employed to obliviously remove redeemed tokens from the cloud server's database T. However, the cloud server can observe which token was deleted. To address this issue, multiple cloud servers can be used, with each holding secret shares of T. After executing an oblivious deletion event, all the shares must be re-randomized.

Finally, we state the security of our PIR-based construction using the following theorem. The proof of security straightforwardly follows from the security of its constituent building blocks and the security analysis presented above. Therefore, we omit the proof.

Theorem 1. *Given the Keyword-PIR, PIR-with-Defalt, Private Matching functionalities described in Section 3, the PIR-based construction of Figure 2 securely*

implements the digitial vaccine passport described in Section 2 in semi-honest setting.

4.2.4 Complexity. We begin with the analysis of the computational complexity. As desired, the authority only needs to perform one PRF (e.g. AES), and one exponentiation per client who was vaccinated.

The cloud server $\mathcal{H}$ requires to perform N AES calls to generate the set T_m. N can set to be 80 if assuming that a token is generated every 15 minutes for approximately 20 hours a day. The $\mathcal{H}$ also involves PIR with the service in Phase 3. Denote the computational cost of PIR as $\texttt{|PIR|}$ which is $O(Nn)$, where n is the number of vaccinated clients. The computational complexity of $\mathcal{H}$ is $N + \texttt{|PIR|}$.

The client needs to compute N AES instances and one exponentiation in Phase 2. In the token redemption phase, she may need to perform private matching with the service, involving two additional exponentiations as described in Section 3.3. The computation on the service's side includes PIR, private matching (if the client presents their ID but doesn't allow the service to collect it), and one exponentiation (if the service can collect the ID).

In terms of communication complexity, the $\mathcal{A}$ sends a κ-bit σ to the client $\mathcal{C}$ and a 3κ-bit $\sigma||m$ to the cloud server. The client sends the service a κ-bit token along with 2κ-bits m, if required. Additionally, all participants except $\mathcal{A}$ send and receive transcripts/randomness from PIR or private matching executions.

4.3 Digital Signature-based Construction (Offline Verification)

The vaccine passport construction described here eliminates the need for an online service provider, $\mathcal{S}$, to verify the authenticity of a token. It relies on randomized signatures and signatures on committed values.

4.3.1 Technical Overview. At the initial stage, the client $\mathcal{C}$ obtains a vaccination certificate σ from the authority after receiving the vaccination. In the PIR-based construction, the validation of a valid token requires an online interaction with the cloud server $\mathcal{H}$ to ensure client privacy. However, our construction eliminates the need for such online verification.

The central concept of our construction, based on digital signatures, involves randomizing the certificate σ into $\sigma^\star$ to ensure their unlinkability. The client can then utilize $\sigma^\star$ during the redemption process in a way that prevents the authority from linking it to the original certificate. The Pointcheval-Sander signature scheme perfectly aligns with our objective as it allows for randomization and offers a scalable solution with constant-sized signatures. Therefore, our signature-based construction relies on the Pointcheval-Sander signature scheme. The authority $\mathcal{A}$ generates a signature σ on the client's information $\boldsymbol{I}$, which includes details such as the client's identity, vaccine type, and effective date. To optimize system performance, the health authority $\mathcal{A}$ only issues a long-term certificate to the client and delegates the generation of short-lived temporary tokens to an untrusted cloud server $\mathcal{H}$.

The design of the system raises the question of how clients can request tokens from the cloud server $\mathcal{H}$. The simplest approach would be for the client to present the cloud server $\mathcal{H}$ with the randomized signature $\sigma^\star$ on the information $\boldsymbol{I}$. However, this would expose all personal information to the cloud server $\mathcal{H}$. Fortunately, the Pointcheval-Sander signature scheme enables us to transform the signature σ on the information $\boldsymbol{I}$ into a randomized signature $\sigma^\star$ on a committed value derived from $\boldsymbol{I}$. Consequently, the cloud server $\mathcal{H}$ can verify the validity of the client's certificate from the authority $\mathcal{A}$ without gaining access to any personal information. Subsequently, the $\mathcal{H}$ can issue tokens to the clients.

Each token includes a signature from the cloud server $\mathcal{H}$ on the committed value derived from $\boldsymbol{I}$, along with additional information t. This additional information t, appended by the cloud server $\mathcal{H}$, primarily comprises the redemption time for the token to prevent any potential reuse. To validate the token, the service $\mathcal{S}$ simply needs to check the validity of the signature, thereby enabling offline verification. For enhanced privacy, the client can also randomize the received token tok into $\mathsf{tok}^\star$ and store only $\mathsf{tok}^\star$ in memory. Consequently, even if the authority and the cloud server collude, they cannot link the utilized token $\mathsf{tok}^\star$ with personal information, ensuring strong privacy guarantees.

Additionally, we propose an optional "light verification" approach where services such as cinemas or restaurants can verify the validity of a token by checking if it is a valid signature from the cloud server $\mathcal{H}$. In this case, the client only needs to present their token tok to the service $\mathcal{S}$, along with aggregated information V related to their personal information, to demonstrate that the token tok is a valid signature issued by the cloud server $\mathcal{H}$. This allows the service $\mathcal{S}$ to quickly verify the token's validity without requiring any personal information from the client. While this approach benefits privacy, its drawback is that the token can be transferred between clients as personal information is not disclosed. For important checks, such as at airports or borders, where identity verification is necessary, the client must present their identity card and provide the information $\boldsymbol{I}$. This enables the service $\mathcal{S}$ to perform a thorough verification of the token against the personal information $\boldsymbol{I}$. In practice, a combination of light and full verification can be employed, wherein daily activities (e.g., restaurants, cinemas, public transport) mainly undergo light verification, with occasional random checks of full verification to mitigate the risk of token transfer between individuals.

4.3.2 Construction. The construction of our signature-based PPass system is outlined in Figure 3, closely adhering to the technical overview provided earlier. Since our construction is based on the Pointcheval-Sander signature scheme [31], we will briefly explain the multi-message version of this signature below:

Setup: A type 3 bilinear map $e : G_1 \times G_2 \to G_T$ with $G_1 = \langle g_1 \rangle, G_2 = \langle g_2 \rangle$, and $G_T = \langle g_T \rangle$ are cyclic group of prime order p.

KeyGen: Choose a secret key $sk = (x, y_1, \ldots, y_n)$ and computes the public key $pk = (g_2, X, Y_1, \ldots, Y_n)$, where $X = g_2^x$ and $Y_i = g_2^{y_i}, i = 1, \ldots, n$.

Sign$(sk, \{m_1, \ldots, m_n\} \in (\mathbb{Z}_p^\star)^n)$: Choose a random $h \in G_1$, define $\sigma_1 = h$ and $\sigma_2 = h^{x+\sum_{j=1}^{n} y_j m_j}$, and output $\sigma = (\sigma_1, \sigma_2)$

Verify$(sk, \{m_1, \ldots, m_n\}, \sigma = (\sigma_1, \sigma_2))$**:** Check whether $\sigma_1 \neq 1_{G_1}$ and $e(\sigma_2, g_2) = e(\sigma_1, X \Pi_{j=1}^{n} Y_j^{m_j})$ are both satisfied, here 1_{G_1} denotes the identity in G_1. If yes, it accepts, otherwise, it rejects.

We use this signature for both authority server $\mathcal{S}$ and cloud server $\mathcal{H}$. In particular, the cloud server $\mathcal{H}$ utilizes the signature to sign the committed value of $m_1, \ldots, m_n$, ensuring that clients' personal information remains concealed from the cloud server $\mathcal{H}$. To accommodate space constraints, we defer the detailed security analysis of our PIR-based construction to Appendix C, where we elaborate on its security.

4.3.3 Complexity. The utilization of the Pointcheval-Sanders signature scheme is particularly advantageous for devices with limited storage capacity. This is because the signature size remains constant, allowing each token to contain only two elements in G_1. In terms of computational requirements, the following observations can be made:

- Client $\mathcal{C}$ performs 2 pairings and n exponentiations in G_1 to verify the validity of each credential or token. However, in practice, $\mathcal{C}$ may directly utilize credentials or tokens without the need for verification. $\mathcal{C}$ needs to randomize each credential or token for privacy, which requires only 2 exponentiations in G_1 per credential or token.
- The cloud server $\mathcal{H}$ performs $n + 2$ pairings to verify each credential because $\mathcal{C}$ only provides $\mathcal{H}$ with the committed values $com = (M_1, \ldots, M_n)$.
- Service $\mathcal{S}$ requires 2 pairings and n exponentiations in G_1 for verification (no exponentiation is necessary for light verification) of each token.
- The generation of credentials or tokens is efficient, requiring just one exponentiation in the group G_1.

5 Performance

In this section, we present an estimation of the performance of our PPass system to demonstrate its feasibility in practical scenarios. We assume that a redeem token is generated every 15 minutes, resulting in approximately 80 distinct tokens per day for each user (denoted as $N = 80$). We consider user information to consist of its identity I_1 and the concatenated vaccine information I_2, which gives us a value of $n = 2$.

For the PIR-based construction, we implement PRF and PRG instances using AES. Each AES operation costs 10 cycles, and on a 2.3 GHz machine, we can expect to compute an AES operation in approximately 0.005 microseconds. In our constructions, participants need to compute exponentiations. For example, the DH-based private matching consists of 3 exponentiation. [30, Table 2] reports the computation cost of DH-based PSI which computes 2^{21} exponentiations in

PARAMETERS:

- A client $\mathcal{C}$, an authority $\mathcal{A}$, a service $\mathcal{S}$, and a cloud server $\mathcal{H}$.
- A bilinear map of type 3 $e : G_1 \times G_2 \to G_T$, where $G_1 = \langle g_1 \rangle, G_2 = \langle g_2 \rangle$, and $G_T = \langle g_T \rangle$ are cyclic groups of prime order p.
- A hash function $H : \{0,1\}^\star \to \mathbb{Z}_p^\star$
- $\mathcal{S}$ chooses a secret key $sk_{\mathcal{S}} = (x, y_1, \ldots, y_n)$ and computes the public key $pk_{\mathcal{S}} = (g_2, X, Y_1, \ldots, Y_n)$, where $X = g_2^x$ and $Y_i = g_2^{y_i}$, for $i \in [n]$.
- $\mathcal{H}$ chooses a secret key $sk_{\mathcal{H}} = (a, b_1, \ldots, b_n, , b_{n+1})$ and computes the public key $pk_{\mathcal{H}} = (g_2, A, B_1, \ldots, B_n, B_{n+1})$, where $A = g_2^a$ and $B_i = g_2^{b_i}$, for $i \in [n+1]$.

INPUTS:

- When a client $\mathcal{C}$ is vaccinated, it is associated with an information vector $\boldsymbol{I} = \{I_1, \ldots, I_n\} \in (\{0,1\}^\star)^n$, where I_1 represent the client's ID, other $I_i \in \{0,1\}^\star$ represents an information about the vaccine taken by the client such as "type of vaccine", "effective date" etc.

PROTOCOL:

I. **Registration Phase**
- Client $\mathcal{C}$ sends $\boldsymbol{I} = \{I_1, \ldots, I_n\}$ to the authority $\mathcal{A}$ who computes $m_i = H(I_i)$ and gets the vector $\boldsymbol{m} = \{m_1, \ldots, m_n\} \in (\mathbb{Z}_p^\star)^n$.
- Authority $\mathcal{A}$ signs on $\boldsymbol{m}$ with the Pointcheval-Sanders signature and outputs $\sigma = (\sigma_1, \sigma_2)$, where:
  - $\sigma_1 = h$, for a random $h \in G_1$
  - $\sigma_2 = h^{x + \Sigma_{j=1}^n y_j m_j}$
- Client receives the credential $\sigma = (\sigma_1, \sigma_2)$ and checks whether $\sigma_1 \neq 1_{G_1}$ and $e(\sigma_1, X \Pi_{j=1}^n Y_j^{m_j}) = e(\sigma_2, g_2)$. If it is positive (otherwise it aborts) then $\mathcal{C}$ randomizes it by randomly chooses $r \in \mathbb{Z}_p$ and outputs $\sigma^\star = (\sigma_1^\star = \sigma_1^r, \sigma_2^\star = \sigma_2^r)$.
- The client computes a committed value of its data: $com = (M_1, \ldots, M_n)$, where $M_j = (\sigma_1^\star)^{m_j}$, for $j = 1, \ldots, n$.

II. **Tokens Generation Phase**
- Client $\mathcal{C}$ sends to $\mathcal{H}$ the committed value com and $\sigma^\star$.
- Upon receipt com and $\sigma^\star$, the server checks the validity whether $\sigma_1^\star \neq 1_{G_1}$ and $e(\sigma_1^\star, X) \Pi_{j=1}^n e(M_j, Y_j) = e(\sigma_2^\star, g_2)$.
- If it is positive (otherwise it aborts), then $\mathcal{H}$ generates tokens for the client.
  - Define $t \in (\mathbb{Z}_p^\star)$ to encode the redeem time for the token.
  - $\mathcal{H}$ randomly chooses $\lambda \in \mathbb{Z}_p^\star$ and computes $f = (\sigma_1^\star)^\lambda$ (equivalently, f is randomly generated from G_1).
  - $\mathcal{H}$ sets $\mathsf{tok} = (\mathsf{tok}_1, \mathsf{tok}_2)$ where $\mathsf{tok}_1 = f$, $\mathsf{tok}_2 = f^a (\Pi_{j=1}^n M_j^{b_j})^\lambda f^{t b_{n+1}} = f^{a + \Sigma_{j=1}^n b_j m_j + b_{n+1} t}$. Each token is thus a Pointcheval-Sanders signature of $\mathcal{H}$ on $(m_1, \ldots, m_n, t)$
- $\mathcal{H}$ sends back (tok, t) to the client $\mathcal{C}$.

III. **Token's Redeem Phase**:
- Upon receipt a token $\mathsf{tok} = (\mathsf{tok}_1, \mathsf{tok}_2)$ and t, the client randomizes it as $\mathsf{tok}^\star = (\mathsf{tok}_1^\star = \mathsf{tok}_1^\gamma, \mathsf{tok}_2^\star = \mathsf{tok}_2^\gamma)$, for a random $\gamma \in \mathbb{Z}_p$.
- Verification: the client shows $\mathsf{tok}^\star$ and its information $\boldsymbol{I} = \{I_1, \ldots, I_n\}$ and t to the service who checks the validity whether: $e(\mathsf{tok}_1^\star, A(\Pi_{j=1}^n B_j^{m_j}) B_{n+1}^t) = e(\mathsf{tok}_2^\star, g_2)$, where $m_i = H(I_i)$ is computed by the service.
- Light verification: The client pre-compute $V = A(\Pi_{j=1}^n B_j^{m_j})$ and then shows $\mathsf{tok}^\star$ and V, t to the service who simply checks the validity whether: $e(\mathsf{tok}_1^\star, V B_{n+1}^t) = e(\mathsf{tok}_2^\star, g_2)$.

Fig. 3. Our signature-based construction.

1148.1 seconds using the miracl library [5]. Using libsodium library [6] which is approximately 10× faster than miracl, we estimate that the time per exponentiation is $1148.1s/2^{21}/10 = 54$ microseconds. Our signature-based construction requires participants to compute pairings. We estimate that each pairing consists of about 30× exponentiations [9,16] which cost about 1620 microseconds.

As mentioned in Section 3.2, a Keyword-PIR query on a database of size 2^{20} requires 7.62 milliseconds on the client's side and 80 milliseconds on the server's side (online time). These queries necessitate a communication bandwidth of 480KB. The PIR-with-Default queries [20] with 2^8 queries on a database of size 2^{20} require a communication of at most 7MB and an online computation time of 2.43 milliseconds.

Table 1 (Appendix A) provides a summary of the estimated running time and communication/size for AES, exponentiation (Exp), two PIR variants (Keyword-PIR and PIR-with-Default) with different running times on the sender/client and receiver/server sides, private matching (PM), and pairing. It is important to note that Keyword-PIR includes a fixed cost for an offline phase on the cloud server's side, which is not included in Table 1.

Based on the information provided in Table 1, we have calculated the running time and communication costs for various implementation options in our PPass system. The estimated values are presented in Table 2 (Appendix A). Upon analysis, we observe that the PIR-based construction generally outperforms the signature-based construction in terms of speed. However, it does come with higher bandwidth costs and relies on an (online) connection between the cloud server $\mathcal{H}$ and the service $\mathcal{S}$. We observe that, a service such as an airport service counter can conduct an online verification to validate the authenticity of a token within just 7 milliseconds, leveraging its authorization to collect the passenger's ID. For offline verification, our protocol takes a maximum of 0.15 seconds. Based on these results, we conclude that the proposed PPass system is practical and feasible for real-world applications.

Acknowledgments.

The second and the fourth authors were partially supported by NSF awards #2101052, #2200161, #2115075, and ARPA-H SP4701-23-C-0074. The third author was partially supported by the BPI VisioConfiance Project.

References

1. European digital green certificates. https://ec.europa.eu.
2. Apple and google privacy-preserving contact tracing. https://www.apple.com/covid19/contacttracing, 2020.

[5] Experiments were done on a machine with an Intel(R) Xeon(R) E5-2699 v3 2.30GHz CPU and 256 GB RAM

[6] https://doc.libsodium.org/

3. A. Abid, S. Cheikhrouhou, S. Kallel, and M. Jmaiel. Novidchain: Blockchain-based privacy-preserving platform for covid-19 test/vaccine certificates. *Software: Practice and Experience*, 52(4):841–867, 2022.
4. A. Ali, T. Lepoint, S. Patel, M. Raykova, P. Schoppmann, K. Seth, and K. Yeo. Communication–computation trade-offs in PIR. Cryptology ePrint Archive, Report 2019/1483, 2019. https://eprint.iacr.org/2019/1483.
5. S. Angel, H. Chen, K. Laine, and S. T. V. Setty. PIR with compressed queries and amortized query processing. In *2018 IEEE Symposium on Security and Privacy*, pages 962–979. IEEE Computer Society Press, May 2018.
6. E. Boyle, N. Gilboa, and Y. Ishai. Function secret sharing: Improvements and extensions. In E. R. Weippl, S. Katzenbeisser, C. Kruegel, A. C. Myers, and S. Halevi, editors, *ACM CCS 2016*, pages 1292–1303. ACM Press, Oct. 2016.
7. J. Camenisch and A. Lysyanskaya. Signature schemes and anonymous credentials from bilinear maps. In M. Franklin, editor, *CRYPTO 2004*, volume 3152 of *LNCS*, pages 56–72. Springer, Heidelberg, Aug. 2004.
8. B. Chor, N. Gilboa, and M. Naor. Private information retrieval by keywords. Cryptology ePrint Archive, Report 1998/003, 1998. https://eprint.iacr.org/1998/003.
9. R. Clarisse, S. Duquesne, and O. Sanders. Curves with fast computations in the first pairing group. In *19th CANS*, 2020.
10. H. Corrigan-Gibbs and D. Kogan. Private information retrieval with sublinear online time. In A. Canteaut and Y. Ishai, editors, *EUROCRYPT 2020, Part I*, volume 12105 of *LNCS*, pages 44–75. Springer, Heidelberg, May 2020.
11. W. Diffie and M. Hellman. New directions in cryptography. 2006.
12. C. Dong and L. Chen. A fast single server private information retrieval protocol with low communication cost. In M. Kutyłowski and J. Vaidya, editors, *Computer Security - ESORICS 2014*, 2014.
13. T. Duong, D. H. Phan, and N. Trieu. Catalic: Delegated PSI cardinality with applications to contact tracing. In S. Moriai and H. Wang, editors, *ASIACRYPT 2020, Part III*, volume 12493 of *LNCS*, pages 870–899. Springer, Heidelberg, Dec. 2020.
14. R. Fagin, M. Naor, and P. Winkler. Comparing information without leaking it. *Commun. ACM*, 39(5):77–85, May 1996.
15. M. J. Freedman, K. Nissim, and B. Pinkas. Efficient private matching and set intersection. In C. Cachin and J. Camenisch, editors, *EUROCRYPT 2004*, volume 3027 of *LNCS*, pages 1–19. Springer, Heidelberg, May 2004.
16. A. Guillevic. *Arithmetic of pairings on algebraic curves for cryptography*. PhD thesis, 2013.
17. IPSOS. Global public backs covid-19 vaccine passports for international travel. https://www.ipsos.com/.
18. S. Kamara, P. Mohassel, and B. Riva. Salus: A system for server-aided secure function evaluation. Cryptology ePrint Archive, Report 2012/542, 2012. https://eprint.iacr.org/2012/542.
19. V. Kolesnikov, R. Kumaresan, M. Rosulek, and N. Trieu. Efficient batched oblivious PRF with applications to private set intersection. In E. R. Weippl, S. Katzenbeisser, C. Kruegel, A. C. Myers, and S. Halevi, editors, *ACM CCS 2016*, pages 818–829. ACM Press, Oct. 2016.
20. T. Lepoint, S. Patel, M. Raykova, K. Seth, and N. Trieu. Private join and compute from pir with default. Cryptology ePrint Archive, Report 2020/1011, 2020. https://eprint.iacr.org/2020/1011.

21. X. Liu, N. Trieu, E. M. Kornaropoulos, and D. Song. Beetrace: A unified platform for secure contact tracing that breaks data silos. *IEEE Data Eng. Bull.*, 43(2):108–120, 2020.
22. P. Madhusudan, P. Miao, L. Ren, and V. Venkatakrishnan. Contrail: Privacy-preserving secure contact tracing. https://github.com/ConTraILProtocols/documents/blob/master/ContrailWhitePaper.pdf, 2020.
23. C. A. Meadows. A more efficient cryptographic matchmaking protocol for use in the absence of a continuously available third party. In *IEEE Symposium on Security and Privacy*, pages 134–137, 1986.
24. W. Meng, Y. Cao, and Y. Cao. Blockchain-based privacy-preserving vaccine passport system. *Security and Communication Networks*, 2022.
25. R. C. Merkle. A certified digital signature. In G. Brassard, editor, *CRYPTO'89*, volume 435 of *LNCS*, pages 218–238. Springer, Heidelberg, Aug. 1990.
26. M. H. Mughees, H. Chen, and L. Ren. Onionpir: Response efficient single-server pir. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, CCS '21, page 2292–2306, New York, NY, USA, 2021. Association for Computing Machinery.
27. G. Oded. *Foundations of Cryptography: Volume 2, Basic Applications*. Cambridge University Press, USA, 1st edition, 2009.
28. S. Patel, G. Persiano, and K. Yeo. Private stateful information retrieval. In D. Lie, M. Mannan, M. Backes, and X. Wang, editors, *ACM CCS 2018*, pages 1002–1019. ACM Press, Oct. 2018.
29. S. Patel, J. Y. Seo, and K. Yeo. Don't be dense: Efficient keyword PIR for sparse databases. In *32nd USENIX Security Symposium (USENIX Security 23)*, pages 3853–3870, Anaheim, CA, Aug. 2023. USENIX Association.
30. B. Pinkas, M. Rosulek, N. Trieu, and A. Yanai. SpOT-light: Lightweight private set intersection from sparse OT extension. In A. Boldyreva and D. Micciancio, editors, *CRYPTO 2019, Part III*, volume 11694 of *LNCS*, pages 401–431. Springer, Heidelberg, Aug. 2019.
31. D. Pointcheval and O. Sanders. Short randomizable signatures. In K. Sako, editor, *CT-RSA 2016*, volume 9610 of *LNCS*, pages 111–126. Springer, Heidelberg, Feb. / Mar. 2016.
32. D. Pointcheval and O. Sanders. Reassessing security of randomizable signatures. In N. P. Smart, editor, *CT-RSA 2018*, volume 10808 of *LNCS*, pages 319–338. Springer, Heidelberg, Apr. 2018.
33. C.-P. Schnorr. Efficient identification and signatures for smart cards. In G. Brassard, editor, *CRYPTO'89*, volume 435 of *LNCS*, pages 239–252. Springer, Heidelberg, Aug. 1990.
34. M. Shakila and A. Rama. Design and analysis of digital certificate verification and validation using blockchain-based technology. In *2023 Eighth International Conference on Science Technology Engineering and Mathematics (ICONSTEM)*, pages 1–9, 2023.
35. D.-H. Shih, P.-L. Shih, T.-W. Wu, S.-H. Liang, and M.-H. Shih. An international federal hyperledger fabric verification framework for digital covid-19 vaccine passport. *Healthcare*, 10(10), 2022.

A Performance

We show the running time of communication cost for building blocks in Table 1 and the performance of our PPass in Table 2.

		AES	Exp.	Keyword-PIR	PIR-with-Default	PM	Pairing
Computation (microsecond)	**Sender**	0.005	54	80000	2430	108	1620
	Receiver			7620	2430	108	
Communication/Size (KB)		0.016	0.032	480	7000	0.096	0.064

Table 1. Estimated running time and communication cost (size) for building blocks and core operations used in our PPass constructions.

	PIR-based Construction		Signature-based Construction	
	Runtime (ms)	Comm. (KB)	Runtime (ms)	Comm. (KB)
Authority $\mathcal{A}$	0.054	0.064	0.054	0.064
Cloud Server $\mathcal{H}$	85.24	480	226492	67108
	5.24*	7000*		
Client $\mathcal{C}$	0.004	0.016	1.836	0.352
	12.99*	0.112*	1.944*	0.384
Service $\mathcal{S}$	7.62	480	142.56	7.68
	2.54*	7000.1*	13.82*	5.12*

Table 2. Estimated running time and communication cost for our PPass system across different implementation options. The client generates $N = 80$ tokens per day. The cloud server has 2^{20} tokens. The numbers with "star" indicate the cost for Step (III,3) where private matching and PIR-with-default are required in the PIR-based construction. The "star" also indicates the cost of light verification in the signature-based construction.

B Related Work

In the realm of controlling the spread of COVID-19, privacy-preserving contact tracing [2, 13, 21, 22] has garnered significant attention. However, there has been limited research on digital vaccine passports, and most existing solutions have privacy vulnerabilities. Notably, none of these solutions have been formally described with their construction and security guarantees.

There have been a few attempts to build digital vaccine passports such as in [3, 24, 34, 35]. However, these works are in the blockchain setting and/or lack performance evaluations. In this section, we review the popular framework – Smart Health Cards Framework[7], which serves as a prominent system for digital vaccine passport cards. This open-source standard has been adopted by numerous companies and organizations, including Microsoft, IBM, and Mayo Clinic. The framework proposes a model involving three parties:

- Issuer (e.g., a lab, pharmacy, healthcare provider, public health department, or immunization information system) generates verifiable tokens (credentials).
- Client (e.g., a phone holder) stores the tokens and presents them when required.
- Verifier/Service (e.g., a restaurant, an airline) receives tokens from the client and verifies their authenticity through proper signatures.

In the Smart Health Cards system, the client is required to disclose personally identifiable information (PII) (e.g., full name and date of birth) and immunization

[7] https://smarthealth.cards

status (e.g., vaccination location, date and time, vaccine type, lot number) to the issuer. Based on this information, the issuer generates multiple tokens, each containing a subset of the client's information along with a digital signature. By choosing which token to present to a verifier, the client can control the granularity of information disclosed to that specific verifier.

Although the Smart Health Cards framework aims to uphold end-user privacy, it fails to meet the desirable properties outlined in Section 2.2: token linkability and reusability. If a client presents tokens to two different verifiers, these verifiers can link the tokens together. Furthermore, if the verifiers collude with the issuer, they can potentially uncover the identity of the token holder.

C Security of our Digital Signature-based Construction.

We proceed with the analysis of the security of our signature-based construction, considering the desired security and privacy properties of a digital vaccine passport system as described in Section 2.2.

Token Unlinkability and Unforgeability. These properties are directly inherited from the unlinkability and unforgeability of the Pointcheval-Sanders signature. Each token corresponds exactly to a signature generated by the cloud server $\mathcal{H}$.

Token Unreusability. To ensure token unreusability, the cloud server $\mathcal{H}$ appends additional information t to each token, indicating its redeem time. Therefore, the token's unreusability outside this redeem time is derived from the unforgeability of the Pointcheval-Sanders signature.

Anonymous Identity. In the digital signature-based construction, clients' anonymity is guaranteed against collusion between the authority $\mathcal{A}$ and the cloud server $\mathcal{H}$. We can outline the proof as follows:

- The authority $\mathcal{A}$ stores all the information $\boldsymbol{I} = \{I_1, \ldots, I_n\}$ of each client as well as the corresponding signature $\sigma = (\sigma_1, \sigma_2)$.
- The cloud server $\mathcal{H}$ stores all the clients' randomized signatures $\sigma^\star = (\sigma_1^\star, \sigma_2^\star)$
- The clients only use their randomized tokens given by the cloud server $\mathcal{H}$. After randomization, the clients do not need to store the original signature σ and the token tok.
- The tokens used by the client $\mathsf{tok}^\star$ are unlinkable to tok and thus unlikable to $\sigma^\star$ and σ.

The unlinkability directly stems from the unlinkability property of the Pointcheval-Sanders signature, which is guaranteed under the Decisional Diffie-Hellman (DDH) assumption. Consequently, client privacy is inherently preserved.

Clearly, when the client is required to present personal information to the $\mathcal{S}$, privacy cannot be guaranteed if the $\mathcal{S}$ is compromised. In this case, we offer the option of light verification, where the service $\mathcal{S}$ only receives $V = A(\Pi_{j=1}^{n} B_j^{m_j})$ and $\mathsf{tok}^\star, t$, without gaining access to any personal information. Even if the service $\mathcal{S}$ colludes with the cloud $\mathcal{H}$, they cannot obtain any personal

information because $\mathsf{tok}^\star$ is unlinkable to $(M_1, \ldots, M_n)$ (due to the unlinkability property of Pointcheval-Sanders). However, if all three parties, $\mathcal{A}, \mathcal{H}, \mathcal{S}$, collude, the identity of the client matching the pre-calculated value V can be revealed. In this scenario, the authority $\mathcal{A}$ can perform an exhaustive search on the entire set of registered clients using their personal information $(m_1, \ldots, m_n)$ and check if V matches $A(\Pi_{j=1}^{n} B_j^{m_j})$. Finally, we state the security of our signature-based construction through the following theorem. The security proof of the construction straightforwardly follows from the security of its building blocks and the security discussion provided above. Therefore, we omit the proof.

Theorem 2. *Given the randomizable (Pointcheval-Sanders) signature scheme described in Section 3.1, the signature-based construction of Figure 3 securely implements the digital vaccine passport described in Section 2 in semi-honest setting.*

D Formal Security Definition

There are two adversarial models and two models of collusion considered.

- **Adversarial Model:** The *semi-honest* adversary follows the protocol but tries to gain additional information from the execution transcript. The *malicious* adversary can employ any arbitrary polynomial-time strategy, deviating from the protocol to extract as much extra information as possible.
- **Collusion Security:** In the *non-colluding* model, independent adversaries observe the views of individual dishonest parties. The model is secure if the distribution of each view can be simulated independently. In the *colluding* model, a single monolithic adversary captures the possibility of collusion between dishonest parties. The model is secure if the joint distribution of these views can be simulated.

Following security definitions of [18, 27], we formally present the security definition considered in this work.

Real-world execution. The real-world execution of protocol Π takes place between a set of users $(\mathcal{C}_1, \ldots, \mathcal{C}_n)$, an authority server $\mathcal{A}$, a cloud server $\mathcal{H}$, a set of services $(\mathcal{S}_1, \ldots, \mathcal{S}_N)$, and a set of adversaries $(\mathsf{Adv}_1, \ldots, \mathsf{Adv}_m)$. Let H denote the honest participants, I denote the set of corrupted and non-colluding participants, and C denote the set of corrupted and colluding participants.

At the beginning of the execution, each participant receives its input x_i, an auxiliary input a_i, and random tape r_i. These values x_i, a_i can be empty. Each adversary $\mathsf{Adv}_{i\in[m-1]}$ receives an index $i \in I$ that indicates the party it corrupts. The adversary Adv_m receives C indicating the set of parties it corrupts.

For all $i \in H$, let out_i denote the output of honest party, let out'_i denote the view of corrupted party for $i \in I \cup C$ during the execution of Π. The i^{th} partial output of a real-world execution of Π between participants in the presence of adversaries $\mathsf{Adv} = (\mathsf{Adv}_1, \ldots, \mathsf{Adv}_m)$ is defined as

$$\mathtt{real}^{i}_{\Pi,\mathsf{Adv},I,C,y_i,r_i}(x_i) \stackrel{\mathsf{def}}{=} \{\mathsf{out}_j \mid j \in H\} \cup \mathsf{out}'_i$$

Ideal-world execution. All the parties interact with a trusted party that evaluates a function f in the ideal-world execution. Similar to the real-world execution, each participant receives its input x_i, an auxiliary input y_i, and random tape r_i at the beginning of the ideal execution. The values x_i, y_i can be empty. Each participant sends their input x'_i to the trusted party, where x'_i is equal to x_i if this user is semi-honest, and is an arbitrary value if he is malicious. If any honest participant sends an abort message ($\perp$), the trusted party returns $\perp$. Otherwise, the trusted party then returns $f(x'_1, \ldots, x'_n)$ to some particular parties as agreed before.

For all $i \in H$, let out_i denote the output returned to the honest participant by the trusted party, and let out'_i denote some value output by corrupted participant $i \in I \cup C$. The i^{th} partial output of a ideal-world execution of Π between participants in the presence of independent simulators $\mathsf{Sim} = (\mathsf{Sim}_1, \ldots, \mathsf{Sim}_m)$ is defined as

$$\mathtt{ideal}^{i}_{\Pi,\mathsf{Sim},I,C,z_i,r_i}(x_i) \stackrel{\text{def}}{=} \{\mathsf{out}_j \mid j \in H\} \cup \mathsf{out}'_i$$

Definition 1. *[18, 27] (Security) Suppose f is a deterministic-time n-party functionality, and Π is the protocol. Let x_i be the parties' respective private inputs to the protocol. Let $I \in [N]$ denote the set of corrupted and non-colluding parties and $C \in [N]$ denote the set of corrupted and colluding parties. We say that protocol $\Pi(I, C)$ securely computes deterministic functionality f with abort in the presence of adversaries $\mathsf{Adv} = (\mathsf{Adv}_1, \ldots, \mathsf{Adv}_m)$ if there exist probabilistic polynomial-time simulators $\mathsf{Sim}_{i \in m}$ for $m < n$ such that for all $\bar{x}, \bar{y}, \bar{r} \leftarrow \{0,1\}^\star$, and for all $i \in [m]$,*

$$\{\mathtt{real}^{i}_{\Pi,\mathsf{Adv},I,C,\bar{y},\bar{r}}(\bar{x}) \widetilde{=} \{\mathtt{ideal}^{i}_{\Pi,\mathsf{Sim},I,C,\bar{y},\bar{r}}(\bar{x})\}$$

Where $\mathsf{Sim} = (\mathsf{Sim}_1, \ldots, \mathsf{Sim}_m)$ and $\mathsf{Sim} = \mathsf{Sim}_i(\mathsf{Adv}_i)$

E Diffie–Hellman - based Private Matching

The DH-based PM operates as follows: The receiver computes $u \leftarrow H(m_1)^r$ using a random, secret exponent r and a one-way hash function H. The computed value u is then sent to the sender. The sender raises u to the power of the random secret k, obtaining u^k. This result is then sent back to the receiver. Upon receiving u^k, the receiver can compute $(u^k)^{1/r}$, which yields $H(m_1)^k$. Next, the sender sends $H(m_0)^k$ to the receiver. The receiver can check whether $H(m_0)^k$ is equal to $H(m_1)^k$ in order to determine the equality of m_0 and m_1. Importantly, in cases where $m_0 \neq m_1$, the receiver learns no information about m_1 from $H(m_1)^k$. This scheme relies on the Diffie-Hellman assumption [11] for its security guarantees. We introduce the Diffie-Hellman assumption in Definition 2. We describe the ideal functionality and the DH-based construction of PM in Figure 4. The computation and communication cost of PM is 3 exponentiations and 3 group elements, respectively.

PARAMETERS: Two parties: sender and receiver
FUNCTIONALITY:
- Wait for input $m_0 \in \{0,1\}^*$ from the sender.
- Wait for input $m_1 \in \{0,1\}^*$ from the receiver.
- Give the receiver output 1 if $m_0 = m_1$ and 0 otherwise.

PROTOCOL:
- The receiver chooses a random exponent r, computes $u \leftarrow H(m_1)^r$ and sends it to the sender
- The sender chooses a random exponent k, computes $v \leftarrow u^k$ and sends it to the receiver
- The sender computes $w \leftarrow H(m_0)^k$ sends it to the receiver
- The receiver output 1 if $v^{1/r} = w$ and 0 otherwise.

Fig. 4. The Private Matching Functionality and DH-based Construction

Definition 2. *[11] Let $\mathbb{G}(\kappa)$ be a group family parameterized by security parameter κ. For every probabilistic adversary Adv that runs in polynomial time in κ, we define the advantage of Adv to be:*

$$|\Pr[\mathsf{Adv}(g, g^a, g^b, g^{ab}) = 1] - \Pr[\mathsf{Adv}(g, g^a, g^b, g^c) = 1]|$$

Where the probability is over a random choice $\mathbb{G}$ from $\mathbb{G}(\kappa)$, random generator g of $\mathbb{G}$, random $a, b, c \in [|G|]$ and the randomness of Adv. We say that the Decisional Diffie–Hellman assumption holds for $\mathbb{G}$ if for every such Adv, there exists a negligible function ϵ such that the advantage of Adv is bounded by $\epsilon(\kappa)$.

Definition 3. *Let $\mathbb{G}$ be a cyclic group of order N, and let g be its generator. The Computational Diffie-Hellman (CDH) problem is hard in $\mathbb{G}$ if no efficient algorithm given (g, g^a, g^b) can compute g^{ab}.*