Phreeli
Privacy By Design Wireless

Double Blind Armadillo: Cryptographic Protocols and Secure Architecture for Privacy-Preserving Phone Service

Version 1.0

Introducing Double Blind Armadillo

Phreeli is a wireless carrier aiming to achieve a privacy-preserving phone service that separates user payment, authentication and login, and phone service operations. The company enlisted Least Authority to design an architecture that prioritizes security and privacy. After evaluating various approaches, Least Authority developed Double Blind Armadillo for Phreeli’s system. Double Blind Armadillo is a new system architecture and cryptographic protocol that integrates unlinkability, blind relay, and blind tokens to enhance user privacy. Traditional, account-based authorization models present privacy risks because all user actions and data can be correlated to create a profile, which is often linked to the user’s real-world identity. Double Blind Armadillo addresses this problem.

Table of Contents

1. Phreeli's Private Phone Service

Who is Phreeli?

Phreeli is a wireless carrier committed to customer privacy. Its approach begins with a simple belief: privacy is not about hiding but about protecting what defines individuals, including their families, daily activities, reputations, personalities, and relationships. Phreeli is designing the service to collect and retain as little information as possible while still maintaining reliable and seamless connectivity. This focus on minimizing data exposure establishes privacy as a default feature rather than an optional one.

What problem are they trying to solve?

At the system level, Phreeli applies privacy-by-design principles that separate identity, payment, and network activity data into distinct, minimal-trust components. These components interact only through strictly defined channels, preventing any single entity from reconstructing a complete view of customer activity. The resulting architecture maintains unlinkability between personal identifiers, payments, and communications while preserving usability and compliance. Least Authority supported the design and implementation of this privacy-enhancing system, contributing expertise in secure architecture and cryptographic protocols to keep customer data compartmentalized and protected throughout. The result of these efforts is Double Privacy Pass, built with Commitments or "Double-Blind Armadillo."

2. Exploring Possible Solutions

Privacy Pass and ZKAPs

Privacy Pass is an anonymous, token-based authentication protocol that enables the creation of unlinkable transactions. This solution works well in situations where a single service is tied to a single proof, such as providing an anonymous alternative to CAPTCHAs. The Phreeli system, however, involves multiple proofs, either user identification or payment, and supports several distinct services and flows, such as account creation and SIM activation. Because of these varied functionalities that require multiple proofs and services, the naive adoption of the Privacy Pass protocol is not feasible for the Phreeli use case.

Although Privacy Pass offers a robust baseline for anonymous authorization, it fails to generalize to more complex, real-world trust relationships. Its lack of contextual awareness, accountability mechanisms, and extensibility limits its applicability. To support broader privacy-preserving applications, including the Phreeli use case and others such as compliance verification, decentralized identity, or auditable anonymity, future protocols must preserve Privacy Pass's unlinkability while incorporating mechanisms for verifiable policy binding and accountable operation.

ZKAPs, or Zero Knowledge Access Passes, are anonymous, token-based authorization protocols based on Privacy Pass. They were created by Least Authority to facilitate online value exchange while separating payment and service data that could otherwise be linked to customers. ZKAPs are based on Privacy Pass, with specific cryptographic and functional enhancements to facilitate anonymous payment flows. Rather than solving a proof-of-humanity puzzle such as a CAPTCHA, ZKAPs require proof of payment to validate a token that a user can then redeem to purchase a service. Most service providers do not need personal data to provide their services; they require it only for processing payments ("accounting").

ZKAPs provide a mechanism to keep payment data separate from personal data. Instead of linking the payment database to the service database, users receive ZKAPs in exchange for proving authorization to use the service. This proof can represent payment, age verification, or membership in an organization. Users can then redeem these tokens to anonymously prove they have permission to access a particular service. With this model, the service provider can verify that a user is authorized to use the service without learning the user's identity. In other words, ZKAPs function as anonymous proofs of payment. Although ZKAPs could have been used for the payment component of the Phreeli system, they are primarily designed to support anonymous payment flows rather than the broader services and operational actions of a phone system.

A Solution to the Problem

Least Authority evaluated the cryptographic options detailed above, as well as additional approaches, but concluded they were unsuitable for further exploration in the Phreeli use case. Privacy Pass solves the problem of private ticketing, allowing the creation of unlinkable transactions. We therefore investigated whether a small adaptation could make it a good solution for this use case. This led to the concept of Double Privacy Pass. However the limitations of this option did not make it a viable option for Phreeli's case. Addressing the limitations in Double Privacy Pass alone, led to adding Commitments or creating Double Privacy Pass with Commitments otherwise named Double Blind Armadillo.

3. Innovation with Double Blind Armadillo

How Double Blind Armadillo (DBA)/Double Privacy Pass With Commitments (DPP+C) Works

Double-Blind Armadillo (aka Double Privacy Pass with Commitments) is a privacy-focused system architecture and cryptographic protocol designed around the principle that no single party should be able to link an individual's real identity, payments, and phone records. Customers should be able to access services, manage payments, and make calls without having their activity tracked across systems. The system achieves this by partitioning critical information related to customer identities, payments, and phone usage into separate service components that communicate only through carefully controlled channels. Each component knows only the information necessary to perform its function and nothing more. For example, the payment service never learns which phone number belongs to a person, and the phone service never learns their name.

This approach enables the privacy property known as unlinkability, which ensures that even when different parts of a system must cooperate to provide a service, no single party can reconstruct a complete picture of who the individual is or what they do. Unlinkability prevents the creation of comprehensive behavioral profiles and significantly reduces some of the risks commonly associated with data exposure. For instance, a database leak containing only phone numbers without contextual information is far less valuable to an attacker than one that includes phone numbers linked to their owners' names.

At the system level, Double-Blind Armadillo implements two techniques that address the limitations of previous systems using Privacy Pass and ZKAPs: blind relays (a mixing service) and blind tokens (cryptographic commitment tokens).

Double Privacy Pass and Blind Relay (Mixing Service)

The system employs a mechanism known as Double Privacy Pass, an adaptation of the original Privacy Pass that applies the Privacy Pass protocol twice.

The following explanation outlines two pieces of privacy-relevant information handled within the Phreeli system that are sensitive in nature:

- A user identifier called "User ID", denoted id_{user}.
- A phone number identifier called "Phone ID," denoted id_{phone}.

These identifiers are involved in a number of actions within the Phreeli system, including:

- Account creation, which creates the id_{user} GUID for internal use.
- eSIM purchase, which generates id_{phone} in the system.

The system also includes a User Service (US), Phone Service (PS), and Mixing Service (MS). The trusted Mixing Service maintains a database that links id_{user} to the id_{phone}.

Double-Blind Armadillo uses a blind relay, or "mixing," service to collect and forward messages between the system's backend components. This relay aggregates common service operations, such as activating new SIM cards or processing payments, across all individual customers and finalizes these operations in large batches at specific times. This design prevents observers at different points in the system, for instance, within the payment processor and the phone network, from inferring the originator of a given operation based on timing patterns.

For example, if Bob pays for service at 9:45 a.m. and a single new number appears on the network at 9:46 a.m., it is reasonable to assume that the number belongs to Bob. However, if all new activations for the day are processed together at a fixed time, such as midnight, correlating a number to a specific user becomes significantly more difficult. This approach ensures that even when information must move between services, there is no direct link connecting a customer's real identity to their phone number. It also helps mitigate timing and correlation attacks that are common in privacy protocols.

A key invariant in the design is that the adapted Privacy Pass protocol is used exclusively to bind a token to the user identifier, id_{user}. The system deliberately avoids using Privacy Pass with other identifiers, particularly id_{phone},to preserve a critical privacy property: the unlinkability between id_{user} and id_{phone}.

This unlinkability is essential. If both identifiers were linked through the same cryptographic token, or through correlations between different tokens, one of the service providers could gain access to information it is not authorized to obtain. For instance:

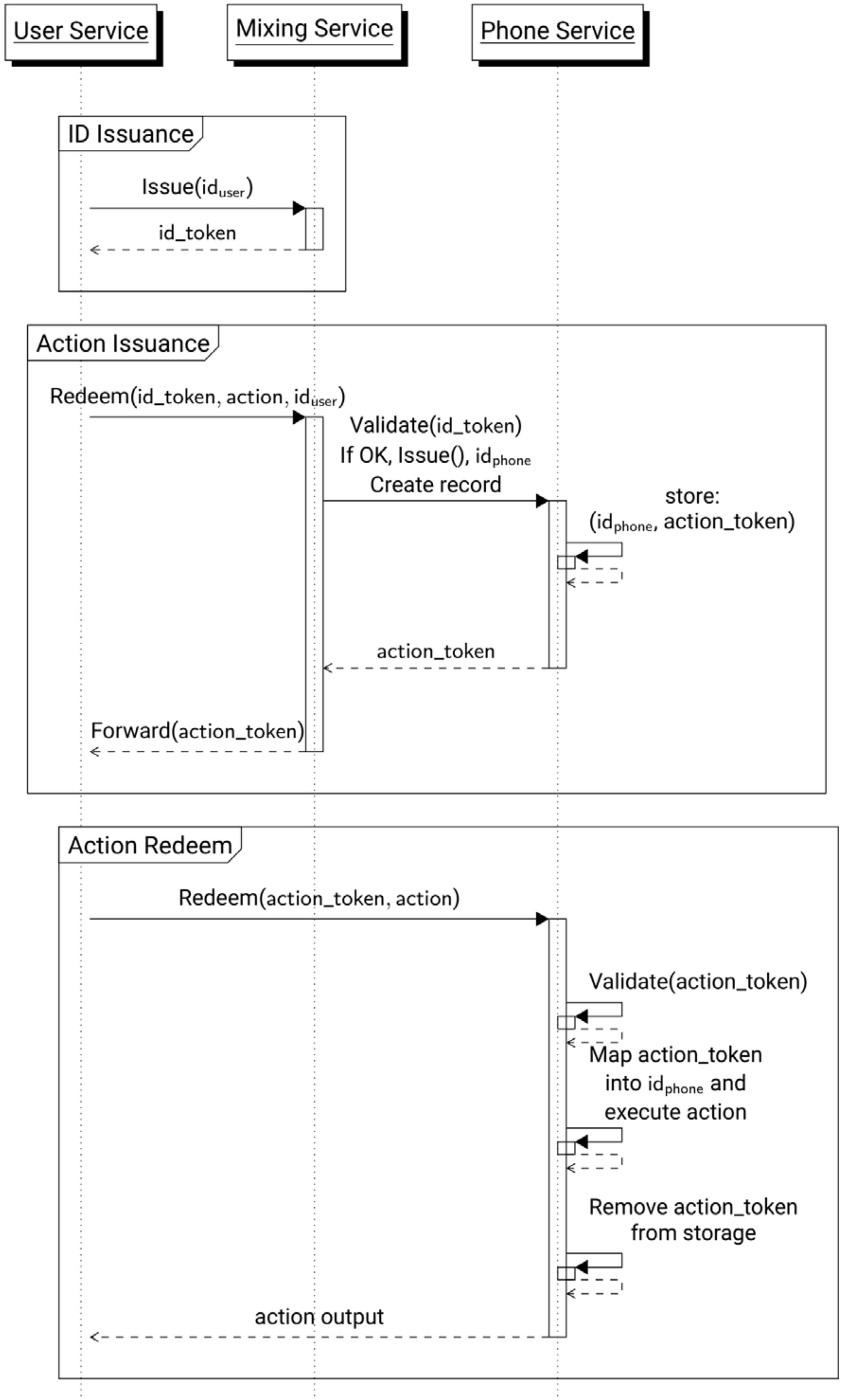
- The User Service (US) knows id_{user}, and if it received a token also linked to id_{phone}, US would need knowledge of id_{phone} to validate the token, which would violate privacy.

- Conversely, the Phone Service (PS) knows id_{phone}, and if the token revealed id_{user}, it could infer the user's real identity. To avoid this, the system uses Privacy Pass twice so that validation of id_{user} occurs on the Mixing Service (MS), which is trusted not to collude with US or PS. Such linkages would directly violate the system's privacy goal of ensuring that no single party can correlate both the user's identity and the phone's identifier.

In summary, using the adapted Privacy Pass only for id_{user} provides the following:

- Accountability: Tokens are cryptographically bound to a legitimate user.
- Privacy: No party can link id_{user} and id_{phone}.
- Unlinkability: The unlinkability between issuance and redemption phases is preserved, as required by the protocol specification.

For the figure below, Issue() represents the simple Privacy Pass, and Issue(id_{user}) represents the adapted Privacy Pass. For consistency, Redeem(. . .) denotes the other parameters of the redemption function and the simple Privacy Pass redeem function call, and Redeem(. . . , id_{user}) represents the double PP variant.


User Service
Mixing Service
Phone Service
ID Issuance
Issue(id_{user})
id_token
Action Issuance
Redeem(id_token, action, id_{user})
Validate(id_token)
If OK, Issue(), id_{phone}
Create record
store:
(id_{phone}, action_token)
action_token
Forward(action_token)
Action Redeem
Redeem(action_token, action)
Validate(action_token)
Map action_token
into id_{phone} and
execute action
Remove action_token
from storage
action output

Blind Tokens with Cryptographic Commitments

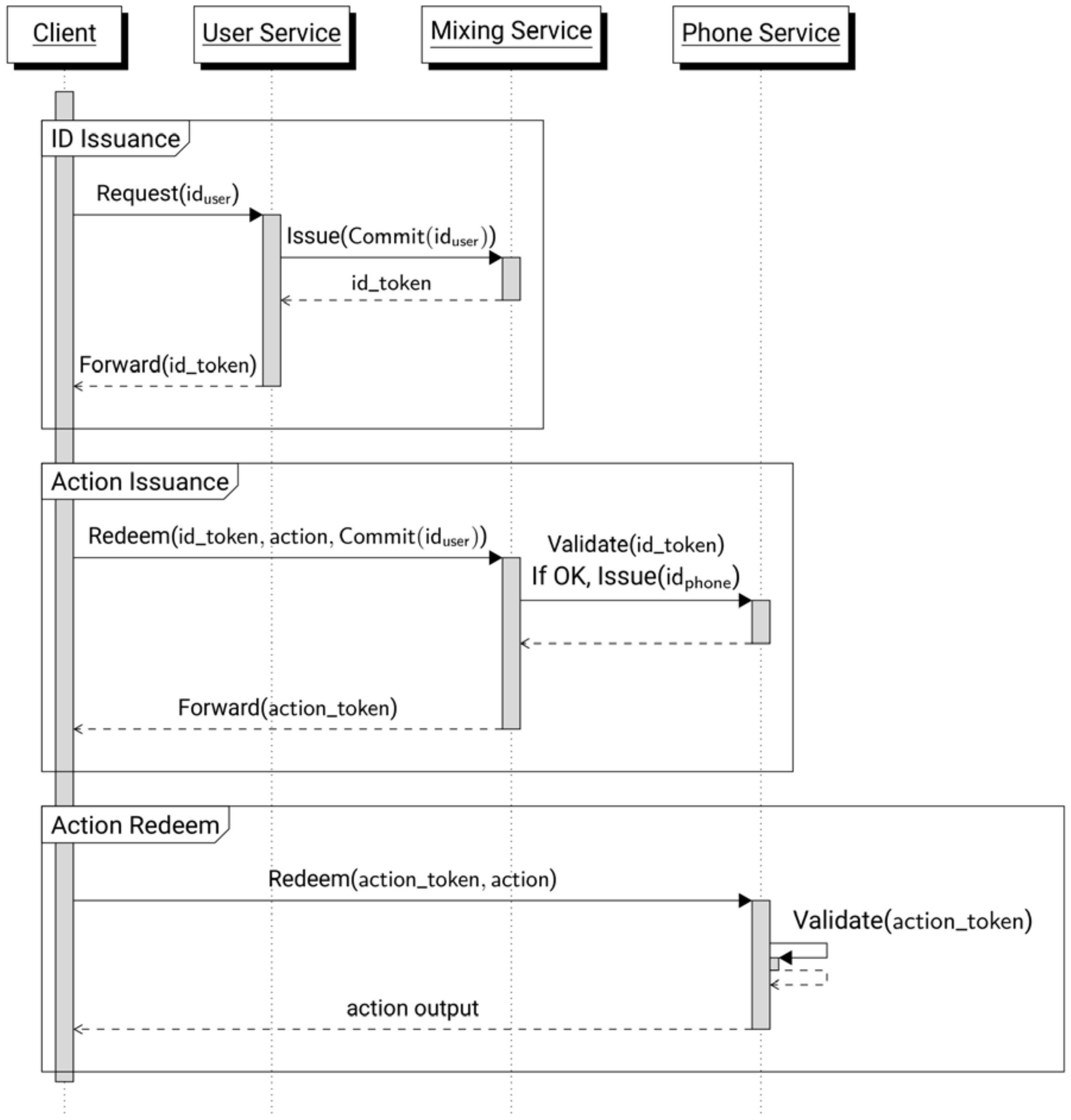
Rather than relying on permanent individual identifiers, the Phreeli system uses short-lived cryptographic "blind tokens" to authorize actions. These tokens, derived from the established Privacy Pass protocol, serve as small digital "proofs" that demonstrate a customer's authorization to perform a given action without revealing their identity, or, conversely, that a phone number is legitimate without revealing its owner. This design allows the system to verify the legitimacy of an account or phone number or to perform operations without exposing or linking the two identifiers.

In the Double Privacy Pass implementation described earlier in this paper, id_{user} and id_{phone} are stored in plaintext. Because the Mixing Service must interpret the session, id_{user}, and the corresponding action to be performed, it must store id_{user} and id_{phone} locally. This can be done in a database but must remain in plaintext for efficiency when searching for the relevant mapping. (Since the communication channels are encrypted and authenticated, id_{user} and id_{phone} are not transmitted in plaintext). To address this limitation, the system implements a cryptographic commitment scheme.

Commitments are cryptographic schemes that allow data to be locked, preventing modification while keeping its value hidden until it is later revealed. This commitment scheme is integrated with the Double Privacy Pass protocol to conceal the values of id_{user} and id_{phone}. Instead of sending id_{user} and id_{phone} in plaintext to the Mixing Service, the User Service and Phone Service send cryptographic commitments of these values to the Mixing Service. As a result, the values of id_{user} and id_{phone} remain hidden but still binding. The Mixing Service stores only the commitments of id_{user} and id_{phone}, not the plaintext values. The User Service and Phone Service may also store this information.

When a client requests an action, it sends an Issue() request to the User Service, which returns its commitment to the User ID id_{user}. Using this information, the client sends a Redeem() request to the Mixing Service along with the desired action, which the Mixing Service then forwards to the Phone Service. The Phone Service can subsequently perform the requested action.

In this configuration, a shared key is required only between the User Service and the Mixing Service. To reverse the flow, the commitments can be opened by the corresponding party, allowing the mapping to be reversed. The message flow in the opposite direction follows the same symmetric structure.



Armadillo Theme Park Example

To further understand how these two properties, unlinkability and time-based batching, work together to preserve privacy, imagine an armadillo-themed amusement park where all employees are required to wear armadillo costumes that completely conceal their faces and any identifying features. Employees clock in at an “Employees Only” building, arriving at various times, but every shift begins at a specific hour. From the perspective of an observer already inside the park, various people may be seen entering the “Employees Only” building at different times. Yet every eight hours, dozens of costumed armadillos emerge simultaneously to begin their shifts.

An observer might watch or follow any of these armadillos as they move around the park, running games and entertaining guests, but determining which person is inside which costume would be especially difficult. In this analogy, the employees' access cards are equivalent to the "blind tokens" used to authorize access to different parts of the system, while the "Employees Only" building is comparable to the "relay" or "mix" service that batches operations together. This illustrates how Double Blind Armadillo maintains a clear separation between individual customers' identities and their actions, thereby enabling unlinkability without sacrificing functionality.

4. An Implementation Version for the Launch

For the initial launch of Phreeli, the determination was made to implement a simpler version of the full solution, namely privacy pass with user commitments instead of double privacy pass with user commitments. In this simpler version, an adapted privacy pass protocol with user commitments is run between the user service and the mixing service. The second run of privacy pass protocol between the mixing service and phone service is eliminated. An acceptable level of security and privacy is still achieved through this simpler version, because of the following reasons: (1) the first run of adapted privacy pass protocol between user service and mixing service (2) trust assumptions on the mixing service that maintains a mapping of Commit(id_{user}) and id_{phone} (3) communication of the client or the user service with the phone service through the mixing service.

5. Summary & Conclusion

The evolution that has led to Double Blind Armadillo reflects a broader maturation of privacy technology, moving from simple anonymity to *verifiable privacy with accountability*. Double Blind Armadillo represents the next logical step, preserving the anonymity guarantees of its predecessors while introducing a cryptographically enforceable notion of trust, context, and policy binding. This move mirrors a broader shift in privacy infrastructure, from systems that *hide who you are* to systems that *prove what you are allowed to do* without revealing your identity. Double Blind Armadillo addresses the problem of private accountability, enabling privacy-preserving verification, governance, and compliance at scale. It extends the zero-knowledge paradigm beyond tokens and payments toward a future where privacy itself becomes a verifiable property of the digital ecosystem.

This transition matters because it transforms privacy from a defensive shield into a *constructive infrastructure* that enables users, organizations, and regulators to coexist within systems that are both private and provably compliant.