IDIS (2009) 2:241–267
DOI 10.1007/s12394-009-0023-5

Accountable privacy supporting services

Jan Camenisch · Thomas Groß · Thomas Scott Heydt-Benjamin

Received: 1 April 2009 / Accepted: 24 July 2009 / Published online: 1 October 2009

Abstract As privacy concerns among consumers rise, service providers increasingly want to provide services that support privacy enhancing technologies. At the same time, online service providers must be able to protect themselves against misbehaving users. For instance, users that do not pay their bill must be held accountable for their behavior. This tension between privacy and accountability is fundamental, however a tradeoff is not always required. In this article we propose the concept of a time capsule, that is, a verifiable encryption with timed and revocable decryptability. The time capsule together with its related protocols offer support of privacy while retaining strong accountability. In our scheme an honest user may enjoy full anonymity, but dishonest users who do not pay their bill have their identity revealed. In contrast to existing revocable anonymity systems, our proposed scheme requires less trust in an external authority, while simultaneously making accountability easier (and less costly) to achieve.

Keywords Accountability · Privacy · Service provision · Time capsule · Zero-knowledge proof of knowledge

J. Camenisch · T. Groß · T. S. Heydt-Benjamin (✉)
IBM Research, Zurich Research Laboratory, Säumerstrasse 4, 8803 Rüschlikon, Switzerland
e-mail: hey@zurich.ibm.com

J. Camenisch
e-mail: jca@zurich.ibm.com

T. Groß
e-mail: tgr@zurich.ibm.com

Introduction

As consumers become increasingly aware of privacy issues, and as identity theft gains visibility as a leading threat, consumers will demand privacy support from their service providers. As this demand and legislative pressure grows (e.g., privacy act), a wide range of providers are potentially effected. For example provision of television and video content, online gaming, news and other articles may need to be executed while taking great care with users' personal data. Privacy experts recommend for such cases a principle of least information, in which services only gather data for which there is a critical business need. (This is a privacy focused reformulation of the principle of least privilege Saltzer and Schroeder 1975). By not holding data which they don't need, service providers are further protected against the dangerous possibility of a data leak which would expose them to bad PR and potential legal action. Furthermore, there are already laws in some jurisdictions that prevent providers from gathering certain personal data entirely.

At the same time, however, commercial service providers require accountability. On the most basic level, a commercial service needs to be paid, and must have the ability to identify and take action against users who do not pay. For example, by permitting anonymous use of a service provided that identities may be revealed when necessary. A common approach to this scenario is that used by identity escrow systems (Kilian and Petrank 1997).

In identity escrow a trusted third party such as a court of law is entrusted with de-anonymization keys. Users enroll with their service provider anonymously, but give the provider a form of their identity which is encrypted to the court. In a dispute, the service provider can ask the court to decrypt this sealed identity to hold the user accountable for improper actions such as non-payment.

There are several problems with traditional identity escrow which we examine further in the body of this article. Most of these problems have to do with the large amount of trust placed in a single trusted party, and the commensurate cost of interacting with that trusted part. Succinctly: There is a significant cost in time, money, and liability to dispute a user action in court. In our scheme we use a novel cryptographic protocol based on a new construction called a time capsule to address these concerns. Our protocol can enforce conditions such as a time by which a payment must be made, and can enforce that the service provider learn a user's identity if payment is not made by this time. In contrast to identity escrow, however, trust is split between multiple parties with well defined business rolls. The reduced trust in the individual trusted parties means that the trusted parties do not need to be governmental authorities. In fact, we will see that some of the trusted parties may be entities like banks, and our protocol requires only the sort of trust that we would already place in a bank. A further contribution of our protocol is that most parties in the system remain oblivious to user identities and hence could not discriminate different users. Thus this promoting privacy further than is possible with identity escrow.

Our contributions in this article are:

1. Examine and formalize the problem of accountable privacy enabled service provision.
2. Propose the time capsule and its related transactions as an improved method of accountability for such services.
3. Show that the time capsule permits identification of a user if and only if they fail to fulfill a contract by the end of the grace period.
4. Contribute an analysis of possible trust models for such a system and a realization of the time capsule with a relaxed trust model compared to the traditional identity escrow.

In the remainder of this article we examine the background and problem of accountable privacy enabled service provision. We consider at a high level our proposed construction, the time capsule, and its related protocols. We make a concrete problem statement, and formalize the requirements of accountable privacy enabled services. We provide cryptographic background and a technical overview of our construction and its protocols. Finally we consider related work and conclude.

Privacy supporting services and identity escrow

Privacy supporting services

To examine the problem of privacy supporting service provision we will use online television broadcasting as an illustrative example. Let us start with a broadcaster that does not need accountability: a public television station. Figure 1 shows a citizen of Elbonia enrolling with the Elbonian Public Broadcasting (EPB) online service. As is common in several countries, licensing restrictions in Elbonia are such that video content that has been shown over traditional television distribution on EPB may also be streamed online to any Elbonian citizen, but should not be available online to non-citizens. Elbonian rules also state than only citizens over 16 years of age should have full access, since some content may be rated for mature viewers. The technical challenge here is for the service provider to enroll a user while ensuring these two constraints and learning no additional personal information about the viewer. Once this challenge has been solved, the user is awarded a service token

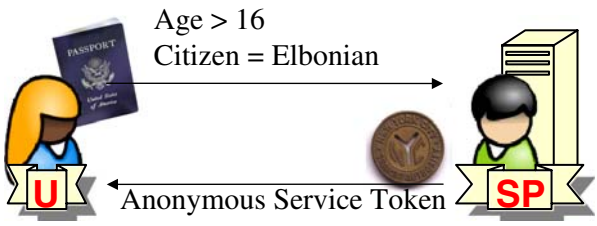

Fig. 1 A user enrolls with a service provider, proving some things about herself and receiving a token for subsequent access to online content

that permits subsequent access to the service in an anonymous manner. This completes enrollment.

It is important to note that the challenges of secure content *delivery* are orthogonal to the challenge of secure and privacy preserving account management we consider in this article. In other words, how the user uses the token to obtain content, and how that content is protected against the user's sharing it with others is a well studied problem separate from the problem of initial user enrollment.

Anonymous credential systems exist that provide methods to achieve a wide range of accountability and privacy goals (Camenisch and Lysyanskaya 2001, 2004; Camenisch and Herreweghen 2002; Camenisch et al. 2006). In particular anonymous credentials systems already have the capability to selectively disclose statements about a user, e.g., proving the user's age without revealing the user's actual date of birth. A critical feature of such a system is user-centricity; that is the user chooses exactly what personal data will be revealed before participating in a credential transaction. This empowers the user with choices about which of their personal data to disclose to whom.

In our example, the Elbonian passport is issued together with an anonymous credential in which the government certifies the user's date of birth and citizenship. Due to the nature of anonymous credentials, the user can now use her passport to complete the enrollment shown in Fig. 1. The resulting token issued to the user may be another anonymous credential, in which case the user's subsequent content access is fully anonymous due to the unlinkability property of anonymous credentials. Thus Elbonian Public Broadcasting is a privacy preserving service provider that nevertheless can enforce restrictions based on user personal data.

Accountability with identity escrow

Whereas a public broadcaster may only need to enforce restrictions at enrollment, typical private broadcasters may need to enforce contract provisions such as payment. Figure 2 shows a user enrolling with a private broadcaster in

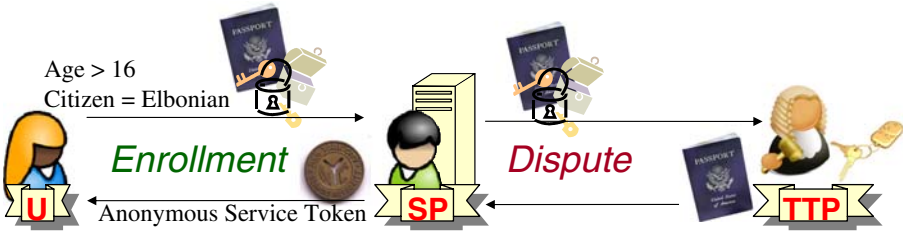

Fig. 2 A user enrolls in a service in a privacy preserving manner, but provides a sealed verifiable encryption of her true identity. This sealed identity can be opened by the authorities in case of a dispute

Elbonia that has the same enrollment requirements as the previous example, yet will have additional accountability requirements.

Anonymous credentials, such as the one on the Elbonian passport, can operate together with the complementary primitive of verifiable encryption (Camenisch and Shoup 2003). Together these technologies can provide accountability without infringing on privacy requirements through a protocol known as identity escrow. In identity escrow a user U can encrypt her true identity to the authorities, provide this encrypted data to a service provider SP, and convince SP with a zero-knowledge proof of knowledge that this encrypted data contains a valid user identity that can be opened by the authorities.

In our example, as a condition of enrollment the user provides an encrypted copy of her complete passport data. This passport data is encrypted such that it may only be opened by order of the court, which holds the decryption key. For instance if the user fails to pay her bill by a specified time, the service provider can open up a case with the court, provide the court with the sealed passport, and ask for the passport to be unlocked. Even though accountable and privacy-preserving identity management methods for services are already possible with a monolithic and fully trusted third party, we propose to rethink the corresponding primitives to better meet the actual needs of services.

Of particular concern is the role of the "authorities". In general, in cryptographic protocols we call the party taking on the role of the authorities the trusted third party (TTP). It is reasonable in many protocols to trust some particular party to perform some action (such as providing its public key honestly), or to refrain from some action (such as publicizing its secret key). A problem with traditional identity escrow is that the TTP requires too much trust, which is why it is usually cast as a law enforcement entity. This presents three challenges:

1. With a fully trusted TTP there is no graceful degradation of privacy and security should the TTP become compromised. In particular, identity escrow permits the TTP to learn the identity of all users, and in some implementations to maintain a database of all transactions users engage in within the system (The TTP learns too much).
2. A malicious service provider holding a verifiable encryption may attempt to betray the user by opening a decryption case at the TTP without good cause. (The TTP alone resolves disputes).
3. Honest service providers find the traditional system encumbering because of the need to involve such highly trusted authorities for even minor dispute cases. For example, to bring a case to law enforcement in the real world is likely to have a non-trivial cost, both in the time required, and in support from legal council. Since the high trust in a monolithic TTP is external to the protocol, disputes must likewise be brought outside the system for resolution.

To address these challenges we aim at breaking up the functionality of the TTP into several sub-functionalities and then distribute these to several designated

trusted parties. Furthermore, to address the last challenge we aim to support resolution of disputes within the system; without or at least with less reliance on outside processes.

Time capsule

To improve on accountability beyond what is possible with traditional identity escrow we propose the time capsule and its related protocols. The time capsule construction when used together with a privacy preserving identity system permits semantically rich service access policies while keeping a chosen portion of a user's identity secret. This secret data (such as full name or passport number), can be revealed only under circumstances (such as an overdue unpaid bill) and will only be revealed to the relevant service provider.

Our scheme requires less trust in a single third party, which simultaneously provides stronger privacy for the user, and a lower cost to the service provider for obtaining accountability when the user misbehaves. Additionally our scheme provides end-to-end unlinkability of users such that transactions cannot be linked with one another except for when a service provider learns the identity of a misbehaving user. Even in this case only the service provider learns the identity in contrast with traditional identity escrow. Further, because of this unlinkability property it is impossible for any entity in the system to provide preferential treatment to a chosen user, or to target a chosen user for an attack.

We will show that these properties provide a system that addresses all three of the above mentioned challenges in existing privacy supporting service provision. Greater detail follows in the body of this paper, but at a high level:

1. Our revocation authority RA is a weaker TTP that cannot link user's transactions within the system. If the RA becomes compromised it is still restricted in what information it can reveal.
   - RA cannot learn any information about the user before the user's bill is past due.
   - RA processes only blinded information.
     When an anonymity revocation is requested by a service provider, RA only knows that it is checking the key for a legitimate transaction, without knowing which transaction or which user. Therefore, RA cannot block requests selectively or collude against any specific user.
   - Even when the bill is past due, and the user has not paid it, RA cannot link this fact to any particular contract or user.
2. Our system contains a mechanism for verifiable, yet privacy supporting, proof of fulfillment of the contracted terms of the service. The RA can easily detect an unfounded request for opening an encrypted identity, and will not service such requests.

3. Our system permits automatic identity revocation in the event that a contract is not fulfilled by the user. Because the contract satisfaction condition can be machine verified, external authorities such as law enforcement do not need to be involved.

We achieve these goals by introducing a special kind of verifiable encryption called a time capsule, represented by Θ. The time capsule has the following properties:

- Θ is issued by the user to the service provider at time α and set not to open till time event ω.
- The service provider is convinced in zero-knowledge that the time capsule contains the expected piece of data (say, the user's certified identity)
- Beginning with time ω, the service provider can decrypt the time capsule (say, if after a payment grace period of a month).
- If the user fulfills a revocation condition before time event ω, the service provider cannot decrypt the time capsule (say, if the user has fulfilled her payment obligation).

For ease of discourse we treat "time" as real clock time, but it should be noted that without loss of generality any other ordered sequence of events suffice for "time", which may be determined by the system implementation.

The time capsule improves on the general verifiable encryption primitive by becoming sensitive to external events. For time events, the verifiable encryption becomes capable of implementing statements of linear temporal logic.

Problem statement

Concrete scenario

We pursue our discussion of accountable and privacy-preserving identity management for services with a concrete example to illustrate the needs of different entities, and how those needs may be met. First, let us name the principles that interact in our example system (illustrated in Fig. 3):

Identity Provider (IDP, *not shown*) issues identity credentials to a user. The IDP could be a passport authority or similar. This entity is well described by existing literature such as Camenisch and Lysyanskaya (2001, 2004) and will not be discussed further here.

Service Provider (SP) provides privacy supporting services to users.

User (U) holds identity credentials issued by the IDP and attempts to use a contracted service of SP.

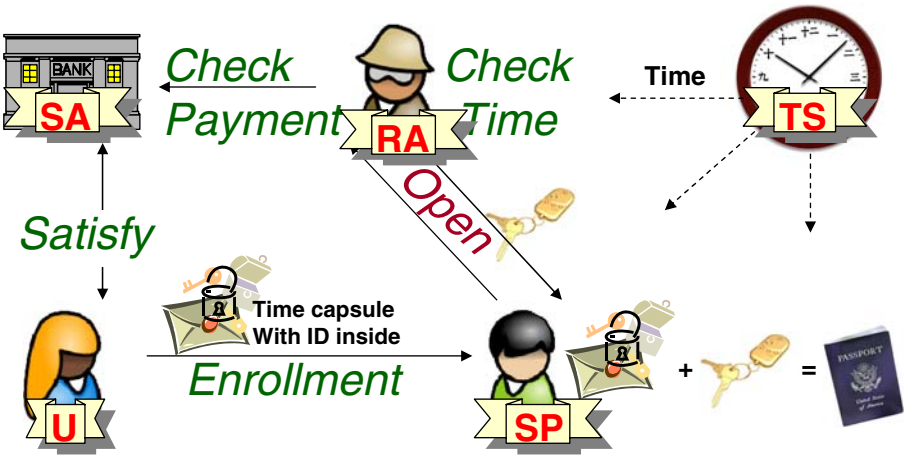

Fig. 3 Relationship diagram of the principles in our accountable privacy enhancing service provision scenario. (Not shown: The Identity Provider that certifies an anonymous credential (e.g., passport), the Bulletin Board.)

Time server (TS) issues a stream of time symbols t_i at regular intervals.

Satisfaction Authority (SA) the authority that determines when the satisfaction condition of a contract is fulfilled and issues a satisfaction token. For example, in the case when the satisfaction condition is payment of a bill, SA could be a bank, and the satisfaction token would be a receipt with which to prove payment.

Revocation Authority (RA) the authority that manages the keypairs used to encrypt and decrypt time capsules. U may communicates with this authority in order to encrypt a time capsule, and SP must communicate with RA in order to attempt opening of the time capsule.

Bulletin Board (BB, *not shown*) A public forum in which items posted are available for anyone to download. The BB in our system is used to post satisfaction tokens by which the user proves payment.

In our example scenario a user U obtains a service from a service provider SP. The user proves that they have an identity attested to by identity provider IDP in zero-knowledge, which proof alone does not identify the user. This proof is sealed in the form of a time capsule, allowing the identity to be revealed at a later time under exacting circumstances. The crux of our scenario is that if U behaves properly (pays her bill on time) then U enjoys anonymous service. Furthermore all of U's transactions in which U has behaved honestly are unlinkable: they cannot be shown to be connected to one another, nor

can they be shown to be associated with the same identity. If, however, U misbehaves, SP has recourse to de-anonymize the service contract through communication with a revocation authority RA. This revocation authority does not know the identities of users in the system, but has the ability to take the record of an anonymous service contract and combine it with information from a predetermined time server TS and satisfaction authority SA to determine a key which will then allow SP to unlock the identity sealed into the service agreement.

In our scenario, a user U has a relationship with an identity provider IDP from which it has obtained identity credentials. U wants to anonymously access a service provided by SP. The SP permits anonymous or pseudonymous contracts for service, yet, wants to ensure that he can hold the user accountable in cases of dispute. SP requires a dead pledge in the form of a *time capsule* Θ through which he can obtain U's true identity if U fails to pay for the service. Thus, the dead pledge Θ can be opened making U's identity accessible to SP after a well-defined grace-period. U on the other hand has the requirement that all her transactions shall be unlinkable provided that contracted obligations are fulfilled before the end of this grace period. Once U has fulfilled her obligations the service provider shall no longer be able to learn her true identity.

Requirements

Verifiability At the conclusion of an enrollment transaction, the service provider receives a time capsule Θ. The service provider must be able to securely determine the following before service is granted:

1. Θ contains an identity certified by IDP.
2. Θ can be opened by SP if payment has not been made by time t_ω.

End to End Unlinkability If the revocation condition occurs before t_ω then no principal will be able to link the user's transactions together nor link any transaction to a specific user. This is a stronger notion of privacy than currently offered in existing systems.

Accountability If the user does not fulfill the revocation condition before time ω, the SP must be able to open Θ and retrieve U's identity

Privacy

- Before the predetermined release time ω no principal can decrypt Θ.
- If the user fulfills the revocation condition before time ω, neither the SP nor any other principal will be able to open the sealed identity.
- If the user does not fulfill the revocation condition before time ω, only SP can open Θ.

Trust model

TS is trusted to:

- Release time symbols t_i sequentially and on time
- Not reveal any t_i for an i that has not yet occurred.

RA is trusted to:

- Not reveal it's own secret key sk_{RA}
- Not reveal any secret time capsule key sk_{Θ} if it detects that the time capsule's contracted revocation condition has occurred.
- To reveal sk_{Θ} if the revocation condition has occurred.
- Note that we do *not* need to trust that the RA will not reveal sk_{Θ} before the contract has expired, this is cryptographically enforced.

IDP is trusted not to reveal its secret key, and to correctly sign identities which it endorses.
SA is trusted to honestly report when payment has occurred.
BB is trusted to report all items posted on the bulletin board (although this can also be checked by the user)

System definition

We define a time capsule system to be a system that supports the following transactions while obeying the requirements of outlined above. Some of the transactions are illustrated in Fig. 3.

We use the following notation to define the transaction's interface. An *non-interactive* interface specification consists of a procedure *Transaction*, inputs and outputs.

$$(outputs) \leftarrow Transaction(inputs)$$

This means that a certain *Transaction* is called with an input tuple *inputs*. A successful execution of the *Transaction* will result in an output tuple of the form *outputs*. The textual specification may specify further outputs for failure cases.

An *interactive* interface specification defines protocols with multiple principals involved. By convention, we use verbatim font for principal names, such as user U or service provider SP. An interactive interface specification names the inputs and outputs of all principals involved together with the principal's name:

$$(\mathsf{U}(outputs), \mathsf{SP}()) \leftarrow Transaction(\mathsf{U}(), \mathsf{SP}(inputs))$$

This *Transaction* is between a user U and a service provider SP, where U does not provide any inputs and receives the output tuple *outputs*. The service provider SP provides *inputs* and does not receive any output.

Setup: $(\mathsf{U}(id_{U,RA})\mathsf{RA}()) \leftarrow Setup(\mathsf{U}, \mathsf{RA})$
RA issues a unique pseudonym $id_{U,RA}$ to user U, which can be used for multiple service contracts within the system. This value is used only in blinded form, thus multiple transactions with this same pseudonym are unlinkable.

Enrollment: $(\mathsf{U}(id_{sat}), \mathsf{SP}(\Theta)) \leftarrow Enroll(\mathsf{U}(id_{U,RA}, id_{U,IDP}), \mathsf{SP})$
During enrollment the user makes a contract with the service provider, receives a token that permits use of the service, and leaves the user with a mechanism to prove fulfillment of the contract (payment). It leaves the service provider with a time capsule Θ that can be opened later to identify the user if the contract is not fulfilled.

Satisfaction of Revocation Condition: $(\mathsf{U}(\mathsf{tok}(id_{sat}'), id_{sat}'), \mathsf{SA}(id_{sat}')) \leftarrow Satisfy(\mathsf{U}(id_{sat}'), \mathsf{SA}(sk_{SA}))$
The user invokes this protocol to fulfill her contract (pay her service bill). The satisfaction authority (the bank) provides a secure receipt that the user later publishes to prove payment of the bill. The receipt is blinded such that it can prove payment for the transaction without revealing the identity of the user.

Publish: $() \leftarrow Publish(\mathsf{U}(\mathsf{tok}(id_{sat}'), id_{sat}'), \mathsf{BB})$
By publishing the receipt of payment the user prevents the service provider from being able to reveal her identity.

Check: $(\mathsf{RA}(\mathsf{tok}(id_{sat}')), \mathsf{BB}()) \leftarrow Check(\mathsf{RA}(id_{sat}'), \mathsf{BB}(id_{sat}'))$
The revocation authority RA checks for revocation of Θ by searching the BB for the secure receipt, and verifying that it is a correct payment for the correct bill. If a receipt is found and verified then the check succeeds, indicating that revocation has occurred and the contract is complete. In other words, if check succeeds then Θ should never be opened.

Open: $(\mathsf{SP}(sk_{\Theta}), \mathsf{RA}(), \mathsf{TS}()) \leftarrow Open(\mathsf{SP}(\Theta), \mathsf{RA}(sk_{RA}), \mathsf{TS}(T_i))$
In the open transaction the service provider attempts to gain knowledge of the user's real identity. The SP requests this knowledge by providing the revocation authority with a designated portion of the time capsule. If the user has not payed their bill, and the bill is overdue, the revocation authority will be able

to extract the secret key that unlocks the rest of the time capsule. If the user has honestly paid her bill, however, this transaction will not succeed, and the service provider learns nothing.

On network disruption

In this article, due to space constraints, we do not fully consider the interesting effects of network disruption on our system. It is important to note that in implementing any system such as ours the system designer has many choices to make with respect to handling of network disruption, and that such choices have profound impact on the behavior of the system. For example, the implementers may set up the system such that if an adversary performs a denial of service attack upon SA, that RA should not reveal any time capsule keys for time capsules bound to SA. This protects against some attacks, for example an attack aiming to identify a user by claiming non-payment and obscuring the fact of payment through making the SA inaccessible. At the same time, it opens the system to other attacks: e.g. preventing legitimate identification of non-paying users. In these simple cases, the design choice of "fail-open" or "fail-closed" have obvious implications. In more complicated cases, sophisticated targeted service disruption could aid in traffic analysis in ways similar to analysis of anonymous communications networks (Back et al. 2001; Serjantov et al. 2002).

Impossibility result

We believe that given the trust model outline above, an external trusted entity is required for accountability of anonymous access. We consider three arguments for necessary conditions for a timed and revocable verifiable encryption. We analyze the requirement of a trusted clock, a revocation authority, and the nature of their information flow.

First, let us analyze the need for a trusted clock. As we do not trust the service provider, it may manipulate its local clock. Thus, the system requires at least a trusted clock publishing user-independent immutable time events.[1] This result accounts for the involvement of the time server TS and the value $\mathsf{TS}(t_i)$ in the *Open*() transaction. It is a *necessary* condition for any system with the given privacy requirements outlined in the Requirements section. For a time capsule system that only permits decryption after time event ω has passed, this is *sufficient*.

Second, we turn to the requirement of a partially trusted revocation authority. A trusted clock solely publishing user-independent time events is not

[1]This may be either realized by an explicit time server TS as defined in the system interface above or by a distributed system of trusted clock components at each principal.

sufficient for a system that accepts a revocation condition. Let us assume there exist two systems with identical state apart from the fact that a user's verifiable encryption is revoked in one system and valid in the other. Thus, the information obtained by the service provider in the *Open*() transaction must convey at least one bit of information indicating whether a user's verifiable encryption is revoked or not. If the service provider SP interacts solely with the user-independent time server TS, the system cannot fulfill this *necessary* condition. By this information flow argument, we realize the need to assume an additional principal RA that

- holds a revocation state, and
- creates a decryption event that is impacted by user-specific information flow.

Third, we can also determine the nature of the information flow to the service provider SP by a generic argument. Because we do not trust the service provider SP, we assume it may ignore arbitrary pieces of information or rewind its internal state. The piece of information provided by the principal RA in the *Open*() is therefore *necessary* for the decryption of the verifiable encryption. That is, it must contain a secret piece of information without which the verifiable encryption cannot be decrypted up to the security of the underlying encryption method. The service provider SP could otherwise neglect this piece of information and decrypt the verifiable encryption anyway.

It therefore is a *necessary* condition that a trusted entity governs the revocation process and holds imperative decryption information confidential.

Cryptographic building blocks

We build our system on top of several cryptographic primitives. We present them in an overview and derive how their properties can be composed to fulfill our problem statement.

In general, we follow the approach to define all our building blocks bottom-up. We start with the mathematical assumptions that guarantee the security of the higher building blocks. The first such foundation is the Strong RSA assumption.

Secondly, we introduce two basic primitives that elevate these assumptions to simple security goals:

Integer Commitments allow a user to bind herself to a chosen integer value without disclosing the value, comparable to a sealed envelope.

Zero-Knowledge Proofs of Knowledge allow a verifier to convince a prover without doubt that the verifier knows a secret x without leaking any information about x.

The third layer contains three higher cryptographic primitives, which build on top of the basic primitives and realize complex service interfaces:

The Camenisch-Lysyanskaya (CL) Signature Scheme is an advanced form of electronic signatures that enables a signer (a) to certify attribute values and messages as part of the signature itself and (b) to render the signature unlinkable over multiple transaction. CL-signatures are particularly adequate to certify a user's identity information (including the identity attributes in the signature itself).

Identity-Based Encryption allows a sender to encrypt messages to the identity of the recipient, be it the recipients e-mail address or a unique user identifier. For instance, a reader of this paper could encrypt a message to the author's of this paper by using the respective e-mail address as encryption key.

Verifiable Encryption allows a sender to encrypt messages to a third party and convince a verifier of attributes of the encrypted message. For instance, a user may encrypt the data of her credit card towards the clearing house or bank. The verifiable encryption allows a vendor to validate that the encryption indeed contains the true credit card data of the user even if the vendor cannot access the decryption key.

We use the notation from Section System Definition as language to specify the interface of the cryptographic building as well. Recall that we distinguish between non-interactive and interactive interfaces. Non-interactive interfaces are specified without principal names. Such transactions can be called by a single principal:

$$(outputs) \leftarrow Transaction(inputs) \ .$$

Interactive interfaces specify transactions between multiple principals and name the corresponding principals with their inputs and outputs. Let us recall the example of a user U and a service provider SP executing a simple interactive transaction:

$$(\mathsf{U}(outputs), \mathsf{SP}()) \leftarrow Transaction(\mathsf{U}(), \mathsf{SP}(inputs)) \ .$$

Assumptions

Cryptography founds its security claims upon mathematical problems that are widely assumed to be particularly hard to compute. "Hard to compute" means that an adversary playing to win against the problem only has a negligible success probability and will fail in the vast majority of cases. Cryptographic primitives build upon those assumptions by proving reductions: if an adversary can break the cryptographic primitive with a non-negligible success probability, then the adversary can also defeat the underlying mathematical assumption.

Strong RSA Assumption (Rivest et al. 1978; Fujisaki and Okamoto 1997) Given an RSA modulus n and a random element $g \in \mathbb{Z}_n^*$, it is hard to compute $h \in \mathbb{Z}_n^*$ and integer $e > 1$ such that $h^e \equiv g \bmod n$. The modulus n is of a special form pq, where $p = 2p' + 1$ and $q = 2q' + 1$ are safe primes.

Integer commitments

Integer commitments bind the user to a value and make it impossible to alter the value later-on. One can imagine the user to put the committed integer in a locked box and give it to the verifier. It prevents the value's disclosure and alteration. The user retains the possibility of opening the commitment, that is, revealing the value later.

Recall the Pedersen commitment scheme (Pedersen 1992), in which the public parameters are a group G of prime order q, and generators $(g_0, \ldots, g_m)$. In order to commit to the values $(v_1, \ldots, v_m) \in \mathbb{Z}_q^m$, pick a random $r \in \mathbb{Z}_q$ and set

$$C = \mathsf{Com}(v_1, \ldots, v_m; r) = g_0^r \prod_{i=1}^{m} g_i^{v_i}.$$

Damgård and Fujisaki (2001) show that if the group G is an RSA group and the committer is not privy of the factorization of the modulus, then in fact the Pedersen commitment scheme can be used to commit to *integers* of arbitrary size.

We specify an abstract interface for integer commitments:

$$(C) \leftarrow \mathit{Com}(v_1, \ldots, v_m; r)$$

for values v_i and a random number r produces an integer commitment C on the values v_i

Known discrete-logarithm-based, zero-knowledge proofs

Zero-knowledge proofs of knowledge allow a prover to convince a verifier that the prover knows a secret x without leaking information about x. For instance, a user can prove that she knows an—otherwise secret—credit card number. Zero-knowledge proofs of knowledge are a versatile primitive of recent cryptographic research that can easily be extended to prove equality and intervals of values or to prove elaborate logical formulas involving AND and OR statements.

In the common parameters model, we use several previously known results for proving statements about discrete logarithms, such as (1) proof of knowledge of a discrete logarithm modulo a prime (Schnorr 1991) or a composite (Fujisaki and Okamoto 1997; Damgård and Fujisaki 2001), (2) proof of knowledge of equality of representation modulo two (possibly different) prime (Chaum and Pedersen 1993) or composite (Camenisch and Michels 1999b) moduli, (3) proof that a commitment opens to the product of two other

committed values (Camenisch and Michels 1999a; Camenisch 1998; Brands 1997), (4) proof that a committed value lies in a given integer interval (Chan et al. 1998; Camenisch and Michels 1999a, b; Boudot 2000), and also (5) proof of the disjunction or conjunction of any two of the previous (Cramer et al. 1994). These protocols modulo a composite are secure under the strong RSA assumption and modulo a prime under the discrete logarithm assumption.

Normally, we use the notation introduced by Camenisch and Stadler (1997) to express Zero-Knowledge Proofs of Knowledge. For this publication, we present a simplified notation tailored for a seamless composition with the service interfaces of our cryptographic building blocks. The interface expresses a proof of knowledge as follows:

$$PK(\{x_0, x_1, \ldots, x_{n-1}\} : \mathsf{fn}(x_0, x_1, \ldots, x_{n-1})) \quad .$$

This notation means a "proof of knowledge that the executing principals knows a set of secrets $\{x_0, x_1, \ldots, x_{n-1}\}$ such that the relation $\mathsf{fn}(\cdot)$ over these secrets (and further public inputs) is fulfilled. We propose this notation well aware that not all relations can efficiently realized by Zero-Knowledge Proofs of Knowledge. However, there exist efficient realizations for all relevant relations $\mathsf{fn}(\cdot)$ of our cryptographic building blocks. The Zero-Knowledge Proofs of Knowledge can therefore act as versatile connective between different primitives. Particularly, efficient realizations exist for equality of attributes and messages, inequality relations, interval range membership, as well as content of integer commitments, CL-signatures, Identity-based Encryption identities, and properties of verifiably encrypted messages. We introduce the relation "$\stackrel{!}{=}$" as simplified test whether values are equal or whether a transaction resulted in a specified output.

Camenisch-Lysyanskaya (CL) signatures

Camenisch-Lysyanskaya is an electronic signature system that allows to (a) include attribute values and messages in the signature itself and (b) to render the signature untraceable over multiple transactions. Camenisch-Lysyanskaya signatures compose exceptionally with Zero-knowledge proofs of knowledge and integer commitments.

Let us for a moment consider the differences to an traditional electronic signature. In a traditional electronic signature, the signer signs a message (say an X.509 certificate) treating the message as an unstructured string. A standard signature only wraps and binds the message. A CL-signature includes important attribute values or message parts in the signature itself and renders them usable for further cryptographic operations. For instance, the signature could contain the user's name, date of birth, credit card number. Where a traditional signature or a corresponding certificate must be revealed in full to be verified, the CL-signature allows a selective disclosure of attribute values. Also, the traditional signature on a certificate is a unique bitstring and can be traced over multiple transactions: whenever the user applies her certificate, she

leaves a trace. In contrast to this, the CL-signatures guarantee that the user remains unlinkable even if all other participants collude to attack the user's privacy. Thus, the CL signatures provide strong authentication with attributes and strong privacy guarantees.

Let us recall the Camenisch-Lysyanskaya signature scheme. We introduce their high-level principles in this section and refer to Camenisch and Lysyanskaya (2003) for a rigorous definition. For this article's contribution we do not depend on a deep understanding of the internals of CL-Signatures, but leverage them as high-level building block.

It consists of two secure two protocols:

(1) An efficient protocol between a user and a signer with keys $(pk_{\mathsf{IDP}}, sk_{\mathsf{IDP}})$. The common input consists of pk_{IDP} and C, a Damgård and Fujisaki commitment as introduced above. The user's secret input is the set of values $(v_1, \ldots, v_\ell, r)$ such that $C = Com(v_1, \ldots, v_\ell; r) \pmod n$. As a result of the protocol, the user obtains a signature $\sigma_{pk_{\mathsf{IDP}}}(v_1, \ldots, v_\ell)$ on his committed values, while the signer does not learn anything about them.
(2) An efficient proof of knowledge of a signature protocol between a user and a verifier. The common inputs are pk_{IDP} and a commitment C. The user's private inputs are the values $(v_1, \ldots, v_\ell, r)$, and $\sigma_{pk_{\mathsf{IDP}}}(v_1, \ldots, v_\ell)$ such that $C = Com(v_1, \ldots, v_\ell; r)$. These signatures are secure under the strong RSA assumption.

Note how we specify signatures of the Camenisch-Lysyanskaya system:

$$\sigma_{pk_{\mathsf{IDP}}}(v_1, \ldots, v_\ell)$$

means a CL-signature σ verifiable under public key pk_{IDP} on the values $(v_1, \ldots, v_\ell)$.

Camenisch-Lysyanskaya use the integer commitment primitive $Com()$ introduced above and be accessed by zero-knowledge proofs of knowledge in the Camenisch-Stadler notation. We provide an additional interface to the scheme:

- $(parCL) \leftarrow CLSetup(1^k)$

  outputs parameters $parCL$.

- $(pk_{\mathsf{IDP}}, sk_{\mathsf{IDP}}) \leftarrow CLKeyGen(parCL)$

  outputs a public key/private key keypair $(pk_{\mathsf{IDP}}, sk_{\mathsf{IDP}})$.

- $(\mathsf{U}(\sigma_{pk_{\mathsf{IDP}}}(v_1, \ldots, v_\ell)), \mathsf{IDP}()) \leftarrow CLSign(\mathsf{U}(pk_{\mathsf{IDP}}, C, (v_1, \ldots, v_\ell, r)), \mathsf{IDP}(pk_{\mathsf{IDP}}, sk_{\mathsf{IDP}}, C))$

  outputs a CL signature $\sigma_{pk_{\mathsf{IDP}}}(v_1, \ldots, v_\ell)$ on the user's values $(v_1, \ldots, v_\ell)$ to the user U and nothing to the IDP, iff $C = Com(v_1, \ldots, v_\ell, r)$.

- $(\mathsf{U}(), \mathsf{V}(\mathsf{valid})) \leftarrow CLVerify(\mathsf{U}(pk_{\mathsf{IDP}}, C, (v_1, \ldots, v_\ell, r), \sigma_{pk_{\mathsf{IDP}}}(v_1, \ldots, v_\ell)), \mathsf{V}(pk_{\mathsf{IDP}}, C))$

  outputs valid to the verifier V and nothing to the user U iff $C = Com(v_1, \ldots, v_\ell, r)$. Otherwise, it outputs $\perp$ to both principals.

Zero-Knowledge Proofs of Knowledge easily interact with Camenisch-Lysyanskaya signatures, particularly, refer to all certified attributes as secrets for further proofs. A typical proof is structured as follows:

$$PK(\{\sigma_{pk_{\mathsf{IDP}}}(v_1, \ldots, v_\ell), v_1, \ldots, v_\ell\} : \mathsf{fn}(v_1, \ldots, v_\ell))$$

This means the user U can provide a proof of knowledge involving all certified values v_i of the CL-signature $\sigma_{pk_{\mathsf{IDP}}}(v_1, \ldots, v_\ell)$.

Committed blind anonymous IBE

An *Identity-based Encryption* (IBE) scheme is an encryption scheme that allows the public key of a recipient to be an identity string *id*. Examples could be the recipient's e-mail address, yet, also the identifier of a service transaction. We employ a special variant of identity-based encryption that (a) keeps the identity string *id* private through-out transactions and (b) provides the identity as committed value compatible with integer commitments. In general, IBE schemes rely on a trusted third party, the *Key Generation Center* (KGC), to derive the user's secret key for an identity sk_{id}. The function to generate this secret key from the identity string is called *IBEExtract*().

We use an IBE scheme with two additional properties: anonymity and blind extraction. Anonymous IBE as introduced by Abdalla et al. (2005) ensures that it is infeasible to derive the identity string *id* to which the message was encrypted from the cyphertext. Green and Hohenberger (2007) introduced blind extract IBE, which provides the capability to generate the secret key to an identity sk_{id} in a blinded fashion. Where normal IBE schemes transfer the identity string in plain text to the Key Generation Center, Blind Extract IBE keeps the identity *id* confidential from it.

Camenisch et al. (2008) propose an IBE primitive that combines both properties. Their *Committed Blind Anonymous IBE* allows for a blind extraction of the secret key sk_{id} by giving the Key Generation Center a commitment on the identity string. The user of the system may either disclose partial information about the identity string or prove statements with efficient Zero-Knowledge Proofs of Knowledge about the commitment. This may, for instance, convey statements about linear relations (Brands 1997) or range proofs (Boudot 2000). Clearly, such proofs can be easily combined with proofs over attributes certified by a Camenisch-Lysyanskaya signature as introduced above. For instance, the a user can generate a proof of knowledge showing that the identity *id* committed in *C* for the *IBEBlindExtract*() is actually equally to an identity attribute in a CL-Signature:

$$PK(\{\sigma_{pk_{\mathsf{IDP}}}(id), id, C\} : C \stackrel{!}{=} Com(id, open) \wedge CLVerify(\mathsf{U}(pk_{\mathsf{IDP}}, id, \sigma_{pk_{\mathsf{IDP}}}(id)), \mathsf{V}(pk_{\mathsf{IDP}})) \stackrel{!}{=} \mathsf{V}(\mathsf{valid}))$$

We refer to the system interface of Camenisch et al. (2008):

- $(parIBE, msk) \leftarrow IBESetup(1^k)$

  outputs parameters $parIBE$ and master secret msk.

- $(\mathsf{A}(sk_{id}), \mathsf{KGC}()) \leftarrow IBEBlindExtract(\mathsf{A}(parIBE, id, open), \mathsf{KGC}(msk, C))$

  generates the secret decryption key sk_{id} for a user A's identity id in an interactive key issuing protocol between A and the KGC. If $C = Com(id, open)$, then the user's output is a decryption key sk_{id} and the output of the KGC is empty. Otherwise both parties output $\perp$.

- $(ct) \leftarrow IBEEnc(parIBE, id, m)$

  outputs cyphertext ct encrypting m under id.

- $(m') \leftarrow IBEDec(parIBE, sk_{id}, ct)$

  outputs message m' encrypted in ct iff the decryption key sk_{id} matches the id of ct.

We assume existence of two additional functions for the IBE scheme that are not originally defined in Camenisch et al. (2008):

- $(\mathsf{valid}) \leftarrow IBEVerify(parIBE, id)$

  outputs valid iff the identity id is suitable to blindly extract the corresponding decryption key sk_{id}. This function is used to verify in zero-knowledge that the identity id is indeed correctly generated.

- $(id) \leftarrow IBEKeyGen(parIBE, (v_1, \ldots, v_\ell)$

  constructs a valid identity/public key string from multiple values v_i. The values may be provided as integer commitment or raw value, say an partial identity string. This function is dependent on the actual implementation of the IBE scheme and the internal format of the identity strings.

Verifiable encryption

Verifiable encryption schemes allow for proving properties about encrypted data. It allows a sender to convince a verifier of the content of an encryption, even if the verifier is not privy of the corresponding decryption key. For instance, a sender could encrypt her true identity towards a trusted custodian and convince a vendor that the encryption indeed contains the true identity of the sender.

In their current form, they were introduced by Camenisch and Shoup (2003), who provided the first efficient construction without resorting to cut-and-choose proofs. Their contribution focuses on discrete-log problems. It can be combined with Pedersen's as well as Damgård and Fujisaki's commitment

schemes (Pedersen 1992; Damgård and Fujisaki 2001) and the Camenisch-Lysyanskaya signature scheme introduced.

Suppose a principal T owns a public key/secret key pair $(pk_{\mathsf{T}}, sk_{\mathsf{T}})$. Suppose further that A encrypts a message m under the public key pk_{T}, derives a cyphertext ct and sends it to B. A can prove a statement about the encrypted m in an efficient Zero-Knowledge Proof of Knowledge to B, while B cannot decrypt the cyphertext ct on its own. B may however gain confidence by A's proof that the cyphertext ct contains a valuable piece of information that principal T is able to retrieve. For instance, A may encrypt its own true identity id_{A} under the public key pk_{T}. It may prove in zero-knowledge that the cyphertext contains the very same identity string as certified in a credential A possesses. Then a receiving service B can be confident that it can have T decrypt the true identity of A if a dispute occurs.

We use the following interface for Camenisch and Shoup's construction:

- $(parVE) \leftarrow VESetup(1^k)$

  outputs parameters $parVE$ and a public key/secret key pair $(pk_{\mathsf{T}}, sk_{\mathsf{T}})$.

- $(ct) \leftarrow VEEnc(pk_{\mathsf{T}}, m)$

  Encrypts the message m to the public key pk_{T} and outputs a cyphertext ct.

- $(m') \leftarrow VEDec(sk_{\mathsf{T}}, ct)$

  Decrypts the cyphertext ct with secret key sk_{T}.

- $(\mathsf{valid}) \leftarrow VEVerify(pk_{\mathsf{T}}, ct, m)$

  Allows a zero-knowledge validation of the message m with respect cyphertext ct encrypted towards public key pk_{T}.

The proof statements about properties of encrypted messages are orthogonal to the verifiable encryption primitive. We can use the same efficient Zero-Knowledge Proof of Knowledge mechanisms as for proofs over commitments, Camenisch-Lysyanskaya signatures, or Committed Blind Anonymous IBE identity strings. Let us consider the example above, in which a user A encrypts the identity id certified in a Camenisch-Lysyanskaya signature $\sigma_{pk_{\mathsf{IDP}}}(id)$ to a trusted party T and proves this fact to a verifier B.

$$\begin{aligned} ct \leftarrow VEEnc(pk_{\mathsf{T}}, id); PK(\{\sigma_{pk_{\mathsf{IDP}}}(id), id, ct\} : \\ CLVerify(\mathsf{U}(pk_{\mathsf{IDP}}, id, \sigma_{pk_{\mathsf{IDP}}}(id)), \mathsf{V}(pk_{\mathsf{IDP}})) \\ \stackrel{!}{=} \mathsf{V}(\mathsf{valid}) \wedge VEVerify(pk_{\mathsf{T}}, ct, id) \stackrel{!}{=} (\mathsf{valid})) \end{aligned}$$

Solution overview

We present an example cryptographic implementation of our system using building blocks from anonymous credentials systems such as (Camenisch and

Lysyanskaya 2001, 2004) and the relatively new *committed blind anonymous identity based encryption* (Green and Hohenberger 2007). We describe here at a high level how these building blocks can implement a system with the requirements and definitions listed in prior sections. More details about the properties of the building blocks are explained in the Cryptographic Primitives section.

System Setup:

RA acting as an IBE KGC and a participant in a verifiable cryptosystem performs $IBESetup(1^k)$ and $VESetup(1^k)$. RA publishes the resultant public information $(parIBE, parVE, pk_{SA})$. Hereafter we will implicitly assume that these public parameters are available to all principals.

Setup: $(\mathsf{U}(id_{U,RA}), \mathsf{RA}()) \leftarrow Setup(\mathsf{U}, \mathsf{RA})$

U is issued a unique pseudonym $id_{U,RA}$ by RA. This pseudonym will form a part of the time capsule's key. By using committed blind anonymous IBE the pseudonym will be later used only in blinded form, preventing linkability between transactions.

Enrollment: $(\mathsf{U}(stok, pk_{\Theta}), \mathsf{SP}(pk_{\Theta}, \Theta)) \leftarrow Enroll(\mathsf{U}(id_{U,RA}, id_{U,IDP}), \mathsf{SP})$

1. U and SP jointly compute a random transaction identifier id_{sat} which is then known to both parties, and a blinded form of the same identifier known to neither U or SP, but rather encrypted to SA and RA respectively: $VEEnc(pk_{SA}, id_{sat}')$, $VEEnc(pk_{RA}, id_{sat}')$
   - The Joint computation of id_{sat} which is then known to both parties can be realized using standard techniques. Joint computation of the blinded id_{sat}' that is known to neither computing party, but is verifiably encrypted to other parties can be realized using standard homomorphic encryption techniques with the addition of blind anonymous verifiable encryption of the jointly computed value.
2. SP mints a satisfaction token $stok \leftarrow \mathsf{tok}(cond, id_{sat}')$ which will later be used by U to prove to SA that the satisfaction conditions have been met. More details on the construction of this token appear below in the overview of the *Satisfaction* transaction.
3. U computes the public portion of a keypair $(pk_{\Theta}, sk_{\Theta})$ which is generated s.t. the keypair depends on $(id_{U,RA}, t_{\omega}, id_{sat})$.
   - The creation of a keypair that depends on values $(id_{U,RA}, t_{\omega}, id_{sat})$ is accomplished using a pub lic key generation function $IBEKeyGen(id_{U,RA}, t_{\omega}, id_{sat}) = (pk_{\Theta}, C_{\Theta})$ such that those values serve as the identity in the generation of a committed blind anonymous IBE, and C_{Θ} serves as extraction information dependant on these values. C_{Θ} contains commitments on $id_{U,RA}$ and id_{sat} that are sufficient for the IBE key extraction, as well as a function of t_{ω} that will permit extraction only after t_{ω} is released by TS. Do to the nature of

committed blind anonymous IBE the resultant pk_Θ can be released, and an encryption against this key can be verified, without revealing the identity or corresponding sk_Θ. Additionally, the later extraction of sk_Θ is a blind anonymous extraction that will not reveal these values. Note that the exact nature of function F will depend on the algebraic structure of the underlying IBE scheme.

4. U computes the time capsule as follows:

$$\Theta = (ct_1 = VEEnc(pk_\Theta, id_{U,IDP}), ct_2 = VEEnc(pk_{RA}, id_{sat}{}' \cdot C_\Theta))$$

U sends Θ to SP

5. SP verifies in zero-knowledge that Θ can be decrypted by sk_Θ and that RA can derive sk_Θ.

- $PK(\{id_{U,RA}, t_\omega, id_{sat}, id_{sat}{}'\} : VEVerify(pk_\Theta, ct_1, id_{U,IDP}) \stackrel{!}{=} (\mathsf{valid}) \wedge VEVerify(pk_{RA}, ct_2, id_{sat}{}') \stackrel{!}{=} (\mathsf{valid}) \wedge IBEVerify(parIBE, pk_\Theta) \stackrel{!}{=} (\mathsf{valid}) \wedge id_{sat}{}' \stackrel{!}{=} \mathsf{blind}(id_{sat}) \wedge CLVerify(\mathsf{U}(pk_{\mathsf{IDP}}, C, (id_{U,IDP}, r), \sigma_{pk_{\mathsf{IDP}}}(id_{U,IDP})), \mathsf{SP}(pk_{\mathsf{IDP}}, C))) \stackrel{!}{=} \mathsf{SP}(\mathsf{valid}))$
- The verifiable encryption primitive of Green and Hohenberger (2007) permits the properties that we require of the time capsule Θ. In particular it is suitable for the encryption key pk_Θ such that proof of the properties of the contents of the encryption and the decryptability under the secret values $(id_{U,RA}, t_\omega, id_{sat})$ is possible with further zero-knowledge proofs.
- The integer commitment C on the user's true identity $id_{U,IDP}$ may server SP as basis to generate an authentication token for U.

6. RA verifies in zero-knowledge that Θ contains the user's identity with respect to the system: $id_{U,IDP}$.
7. SP sends to U a token *stok* permitting access to the service.

Satisfaction of Revocation Condition: $(\mathsf{U}(rtok), \mathsf{SA}()) \leftarrow Satisfy(\mathsf{U}(id_{sat}{}'), \mathsf{SA}(sk_{SA}))$

In this protocol U sends SA the satisfaction token given to U during enrollment, as well as the copy of $id_{sat}{}'$ which was encrypted towards SA's secret key during enrollment. SA decrypts $id_{sat}{}'$, verifies the satisfaction condition in the satisfaction token, and verifies that the satisfaction token is bound to $id_{sat}{}'$ by a commitment. If all of this verifies, U receives from SA a revocation token *rtok* which can be used to prove satisfaction has occurred. Such a token could be implemented for example as a signed statement from SA that satisfaction has occurred with respect to $id_{sat}{}'$. The proof of the properties of the satisfaction token can be accomplished with standard anonymous credential techniques:

- The satisfaction token minted by SP during enrollment with respect to $id_{sat}{}'$ can be accomplished by encoding the satisfaction conditions *cond*

(such as an amount of money to be paid, and the destination of that money) as attributes in an anonymous credential that has $id_{sat}{}'$ encoded as a committed attribute.

- SA verifies that U has performed the correct actions sufficient for satisfaction (eg. payed the correct amount of money to the correct entity) by verifying these conditions against the *cond* attributes of the satisfaction token $\mathsf{tok}(cond, id_{sat}{}')$. When SA decrypts its copy of $id_{sat}{}'$ it additionally verifies that the now satisfied conditions belong to the correct transaction by checking that the $id_{sat}{}'$ attribute of the satisfaction token matches the decrypted value.
- SA can then sign a statement that the transaction $id_{sat}{}'$ has been fulfilled using any secure signature scheme.
- $ct_{SA} = VEEnc(pk_{SA}, id_{sat}{}')$;
  $PK(\{id_{sat}, id_{sat}{}', cond\}$:
  $CLVerify(\mathsf{U}(pk_{\mathsf{SP}}, C, (id_{sat}{}', cond, r), \sigma_{pk_{\mathsf{SP}}}(id_{sat}{}', cond)), \mathsf{SP}(pk_{\mathsf{SP}}, C)) \stackrel{!}{=}$
  $\mathsf{SP}(\mathsf{valid}) \wedge id_{sat}{}' \stackrel{!}{=} \mathsf{blind}(id_{sat}))$

Revocation: $(\mathsf{U}(), \mathsf{BB}(rtok)) \leftarrow Revoke(\mathsf{U}, \mathsf{BB})$

The user publishes to the bulletin board BB $rtok = \mathsf{tok}(id_{sat}{}')$ and $id_{sat}{}'$.

- The user can revoke the ability of SP to open the time capsule by broadcasting proof of payment on BB. Since only $id_{sat}{}'$ and a signature on $id_{sat}{}'$ are broadcast, and since $id_{sat}{}'$ cannot be linked with the user, this satisfies the privacy requirements of the system.

Check: $(\mathsf{RA}(id_{sat}{}'), \mathsf{BB}()) \leftarrow Check(\mathsf{RA}(id_{sat}{}'), \mathsf{BB}(id_{sat}{}'))$

The RA checks for revocation by searching the BB for $id_{sat}{}'$, and verifying $\mathsf{tok}(id_{sat}{}')$. If they are found and verified then the check succeeds, indicating that revocation has occurred and the contract is complete.

- During the *Check* operation $id_{sat}{}'$ is revealed to RA by decryption under sk_{RA}. This suffices for RA to determine whether the contract has been satisfied, but RA cannot link the *Enroll* event for this transaction with this check, which provides the end to end unlinkability required for honest users.

Open: $(\mathsf{SP}(sk_{\Theta}), \mathsf{RA}(), \mathsf{TS}()) \leftarrow Open(\mathsf{SP}(\Theta), \mathsf{RA}(sk_{RA}), \mathsf{TS}(T_i))$

1. SP sends $VEEnc(pk_{RA}, id_{sat}{}')$ to RA and requests sk_{Θ}.
2. RA decrypts $VEEnc(pk_{RA}, id_{sat}{}' \cdot C_{\Theta})$ to retrieve $id_{sat}{}'$ (which is not linkable by RA with any other data).
3. RA performs *Check* $(id_{sat}{}')$. If *Check* succeeds, RA detects revocation and halts without returning sk_{Θ}.
4. If $i >= \omega$, RA executes $IBEBlindExtract(\mathsf{SP}(parIBE, pk_{\Theta}, open), \mathsf{RA}(msk, C_{\Theta}))$ and returns sk_{Θ}.
   - Since t_{ω} is now revealed, this information combined with C_{Θ} suffices for the blind committed extraction.

5. SP decrypts and reveals $id_{U,IDP}$
   - Only if the revocation is not detected by RA and key extraction is possible (it is after time ω) will sk_{Θ} be revealed to SP. If the key is revealed to SP then the capsule can be opened and the user's true identity revealed. This satisfies the privacy properties and the accountability properties of the system.

Other related works

Chaum pioneered privacy-preserving protocols that minimize the amount of personal data disclosed. His work put forth the principles of anonymous credentials (Chaum 1981, 1985; Chaum and Evertse 1987), group signatures (Chaum and van Heyst 1991), and electronic cash Chaum (1983). Subsequently, a number of authors provided more efficient implementations of these primitives, e.g., group signatures (Ateniese et al. 2000; Boneh et al. 2004; Kiayias and Yung 2006), e-cash (Brands 1993; Camenisch et al. 2005; Frankel et al. 1998), anonymous credentials (Brands 1995a, b, 2000; Camenisch and Lysyanskaya 2001, 2004), traceable signatures (Kiayias et al. 2004), anonymous auctions (Naor et al. 1999), and electronic voting based on blind-signatures (Fujioka et al. 1992).

All these primitives have in common that some party issues a user some form of certificate that often contains information about the user encoded as attributes. Typically, these attributes are encoded as a discrete logarithm or, more generally, as an element (exponent) of a representation of a group element.

There are also some works (Cramer et al. 1994; Brands 1997; Camenisch and Michels 1998; Boudot 2000; Fujisaki and Okamoto 1997) that these authors employ to prove AND, OR and NOT statement about attributes, e.g., "a user has attribute a OR b," basically by showing that some committed value equals a given value OR some other given value.

Conclusion

In this article we presented a novel scheme to better fill the needs of service providers. We did this by introducing the concept of the time capsule: a cryptographic device that contains the user's real identity, and can only be opened if the user does not fulfill the terms of a predetermined service contract such as a contract to pay a bill on time.

By leveraging the recently developed anonymous blind committed identity based encryption scheme we are able to form an encryption key for the time capsule that depends on the user's pseudonym within the system, as well as conditions such as time and payment status. We have shown that due to the

properties of this and other blinding mechanisms, the secret key for decrypting the time capsule can only be derived if the user misbehaves.

By contrast with existing anonymity revocation schemes, we do not require a fully trusted third party, such as a law enforcement agency, or judicial process. Due to the weaker trust requirements for the TTP it is easier for service providers to perform the anonymity revocation when a user misbehaves, relieving what may be a prohibitive expense for routine transactions.

While we have shown that a system with our desired properties can be constructed from known cryptographic primitives, it remains for future work to describe an exact implementation. Furthermore we believe that there may be more than one way to implement a system matching our requirements. Future work should investigate which implementations might offer the best efficiency, or the weakest assumptions.

Acknowledgement The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/ 2007-2013) under grant agreement n 216483.

Abbreviations

BB	Bulletin Board
EPB	Elbonian Public Broadcasting
IBE	Identity Based Encryption
IDP	Identity Provider
KGC	Key Generation Center
RA	Revocation Authority
SA	Satisfaction Authority
SP	Service Provider
TS	Time Service
TTP	Trusted Third Party
U	User

References

Abdalla M, Bellare M, Catalano D, Kiltz E, Kohno T, Lange T, et al. Searchable encryption revisited: consistency properties, relation to anonymous ibe, and extensions. In: Shoup V, editor. Advances in cryptology—CRYPTO 2005. Lecture notes in computer science. New York: Springer; 2005. vol. 3621, p. 205–222.

Ateniese G, Camenisch J, Joye M, Tsudik G. A practical and provably secure coalition-resistant group signature scheme. In: Bellare M, editor. Advances in cryptology—CRYPTO 2000. Lecture notes in computer science. New York: Springer; 2000. vol. 1880, p. 255–270.

Back A, Möller U, Stiglic A. Traffic analysis attacks and trade-offs in anonymity providing systems. In: Moskowitz IS, editor. Proceedings of information hiding workshop (IH 2001). LNCS. New York: Springer; 2001. vol. 2137, p. 245–257.

Boneh D, Boyen X, Shacham H. Short group signatures. In: Franklin MK, editor. Advances in cryptology—CRYPTO 2004. Lecture Notes in Computer Science. New York: Springer; 2004. vol. 3152, p. 41–55.

Boudot F. Efficient proofs that a committed number lies in an interval. In: Preneel B, editor. Advances in cryptology—EUROCRYPT 2000. Lecture Notes in Computer Science. New York: Springer; 2000. vol. 1807, p. 431–444.
Brands S. An efficient off-line electronic cash system based on the representation problem. Technical Report CS-R9323, CWI; Apr. 1993.
Brands S. Restrictive blinding of secret-key certificates. Technical Report CS-R9509, CWI; Sept. 1995a.
Brands S. Secret-key certificates. Technical Report CS-R9510, CWI, Sept. 1995b.
Brands S. Rapid demonstration of linear relations connected by boolean operators. In: Fumy W, editor. Advances in cryptology—EUROCRYPT '97. Lecture Notes in Computer Science. New York: Springer; 1997. vol. 1233, p. 318–333.
Brands S. Rethinking public key infrastructures and digital certificates: building in privacy. Cambridge: MIT; 2000.
Camenisch JL. Group signature schemes and payment systems based on the discrete logarithm problem. PhD thesis, ETH Zürich; 1998. Diss. ETH No. 12520, Hartung Gorre Verlag, Konstanz.
Camenisch J, Herreweghen EV. Design and implementation of the idemix anonymous credential system. In: Proc. 9th ACM conference on computer and communications security. New York: ACM; 2002.
Camenisch J, Lysyanskaya A. Efficient non-transferable anonymous multi-show credential system with optional anonymity revocation. In: Pfitzmann B, editor. Advances in cryptology—EUROCRYPT 2001. Lecture notes in computer science. New York: Springer; 2001. vol. 2045, p. 93–118.
Camenisch J, Lysyanskaya A. A signature scheme with efficient protocols. In: Cimato S, Galdi C, Persiano G, editors. Security in communication networks, third international conference, SCN 2002. Lecture notes in computer science. New York: Springer; 2003. vol. 2576, p. 268–289.
Camenisch J, Lysyanskaya A. Signature schemes and anonymous credentials from bilinear maps. In: Franklin MK, editor. Advances in cryptology—CRYPTO 2004. Lecture notes in computer science. New York: Springer; 2004. vol. 3152, p. 56–72.
Camenisch J, Michels M. Proving in zero-knowledge that a number *n* is the product of two safe primes. Technical Report RS-98-29, BRICS, Departement of Computer Science, University of Aarhus; Nov. 1998.
Camenisch J, Michels M. Proving in zero-knowledge that a number *n* is the product of two safe primes. In: Stern J, editor. Advances in cryptology—EUROCRYPT '99. Lecture notes in computer science. New York: Springer; 1999a. vol. 1592, p. 107–122.
Camenisch J, Michels M. Separability and efficiency for generic group signature schemes. In: Wiener M, editor. Advances in cryptology—CRYPTO '99. Lecture Notes in Computer Science. New York: Springer; 1999b. vol. 1666, p. 413–430.
Camenisch J, Shoup V. Practical verifiable encryption and decryption of discrete logarithms. In: Boneh D, editor. Advances in cryptology—CRYPTO 2003. Lecture notes in computer science. 2003. vol. 2729, p. 126–144.
Camenisch J, Stadler M. Efficient group signature schemes for large groups. In: Kaliski B, editor. Advances in cryptology—CRYPTO '97. Lecture notes in computer science. New York: Springer; 1997. vol. 1296, p. 410–424.
Camenisch J, Hohenberger S, Lysyanskaya A. Compact E-cash. In: Cramer R, editor. Advances in cryptology—Eurocrypt 2005. Lecture notes in computer science. New York: Springer; 2005. vol. 3494, p. 302–321.
Camenisch J, Hohenberger S, Kohlweiss M, Lysyanskaya A, Meyerovich, M. How to win the clonewars: efficient periodic n-times anonymous authentication. In: Juels A, Wright RN, di Vimercati SDC, editors. ACM conference on computer and communications security. New York: ACM; 2006. p. 201–210.
Camenisch J, Kohlweiss M, Durán AR, Sheedy C. Public key encryption with oblivious keyword search and its application to data retention. In: IBM Research Report RZ 3708, IBM Research. 2008. http://domino.research.ibm.com/library/cyberdig.nsf/index.html. May 2008.
Chan A, Frankel Y, Tsiounis Y. Easy come–easy go divisible cash. In: Nyberg K, editor. Advances in cryptology—EUROCRYPT '98. Lecture Notes in Computer Science. New York: Springer; 1998. vol. 1403, p. 561–575.

Chaum D. Untraceable electronic mail, return addresses, and digital pseudonyms. Commun ACM. 1981;24(2):84–88.
Chaum D. Blind signatures for untraceable payments. In: Chaum D, Rivest RL, Sherman AT, editors. Advances in cryptology—proceedings of CRYPTO '82. New York: Plenum; 1983. p. 199–203.
Chaum D. Security without identification: transaction systems to make big brother obsolete. Commun ACM 1985;28(10):1030–1044.
Chaum D, Evertse J-H. A secure and privacy-protecting protocol for transmitting personal information between organizations. In: Odlyzko M, editor. Advances in cryptology—CRYPTO '86. Lecture notes in computer science. New York: Springer; 1987. vol. 263, p. 118–167.
Chaum D, Pedersen TP. Wallet databases with observers. In: Brickell EF, editor. Advances in cryptology—CRYPTO '92. Lecture Notes in Computer Science. Springer; 1993. vol. 740, p. 89–105.
Chaum D, van Heyst, E. Group signatures. In: Davies DW, editor. Advances in cryptology—EUROCRYPT '91. Lecture notes in computer science. Springer; 1991. vol. 547, p. 257–265.
Cramer R, Damgård I, Schoenmakers B. Proofs of partial knowledge and simplified design of witness hiding protocols. In: Desmedt YG, editor. Advances in cryptology—CRYPTO '94. Lecture notes in computer science. Springer; 1994. vol. 839, p. 174–187.
Damgård I, Fujisaki E. An integer commitment scheme based on groups with hidden order. http://eprint.iacr.org/2001 (2001).
Frankel Y, Tsiounis Y, Yung M. Fair off-line e-cash made easy. In: Kim K, Matsumoto T, editors. Advances in cryptology—ASIACRYPT '98. Lecture notes in computer science. New York: Springer; 1998. vol. 1514, p. 257–270.
Fujioka A, Okamoto T, Ohta K. A practical secret voting scheme for large scale elections. In: Seberry J, Zheng Y, editors. ASIACRYPT. Lecture notes in computer science. New York: Springer; 1992. vol. 718, p. 244–251.
Fujisaki E, Okamoto T. Statistical zero knowledge protocols to prove modular polynomial relations. In: Kaliski B, editor. Advances in cryptology—CRYPTO '97. Lecture notes in computer science. New York: Springer; 1997. vol. 1294, p. 16–30.
Green M, Hohenberger S. Blind identity-based encryption and simulatable oblivious transfer. In: ASIACRYPT; 2007. p. 265–282.
Kiayias A, Yung M. Secure scalable group signature with dynamic joins and separable authorities. IJSN 2006;1(1/2):24–45.
Kiayias A, Yung M, Tsiounis Y. Traceable signatures. In: Cachin C, Camenisch J, editors. Advances in cryptology—EUROCRYPT 2004. Lecture notes in computer science. New York: Springer; 2004. vol. 3027, p. 571–589.
Kilian J, Petrank E. Identity escrow. In: Advances in cryptology Ů CRYPTO '98. New York: Springer; 1997. p. 169–185.
Naor M, Pinkas B, Sumner R. Privacy preserving auctions and mechanism design. In: Proc. 1st ACM conference on electronic commerce. 1999.
Pedersen TP. Non-interactive and information-theoretic secure verifiable secret sharing. In: Feigenbaum J, editor. Advances in cryptology—CRYPTO '91. Lecture notes in computer science. New York: Springer; 1992. vol. 576, p. 129–140.
Rivest RL, Shamir A, Adleman L. A method for obtaining digital signatures and public-key cryptosystems. Commun ACM. 1978;21(2):120–126.
Saltzer JH, Schroeder MD. The protection of information in computer systems. 1975.
Schnorr CP. Efficient signature generation for smart cards. J Cryptol. 1991;4(3):239–252.
Serjantov A, Dingledine R, Syverson P. From a trickle to a flood: active attacks on several mix types. In: Petitcolas F, editor. Proceedings of information hiding workshop (IH 2002). LNCS. New York: Springer; 2002. vol. 2578.