A User-Centric System for Verified Identities on the Bitcoin Blockchain

Daniel Augot[1,2,3], Hervé Chabanne[4,5], Thomas Chenevier[4], William George[1,2,3(✉)], and Laurent Lambert[4]

[1] INRIA, Palaiseau, France
daniel.augot@inria.fr
[2] Laboratoire LIX, École Polytechnique and CNRS UMR 7161, Palaiseau, France
wgeorge@lix.polytechnique.fr
[3] Université Paris-Saclay, Paris, France
[4] OT-Morpho, Issy-les-Moulineaux, France
[5] Télécom ParisTech, Paris, France
{herve.chabanne,thomas.chenevier}@morpho.com

Abstract. We present an identity management scheme built into the Bitcoin blockchain, allowing for identities that are as indelible as the blockchain itself. Moreover, we take advantage of Bitcoin's decentralized nature to facilitate a shared control between users and identity providers, allowing users to directly manage their own identities, fluidly coordinating identities from different providers, even as identity providers can revoke identities and impose controls.

Keywords: Bitcoin blockchain · Identity proofs · Discrete Logarithm REPresentation (DLREP) · Personal Identity Management Systems (PIMS)

1 Introduction

We live in a world where the ways in which a person's identity is being used are increasingly complex. Appropriately handling sensitive personal data, such as medical, financial, and employment data, is subtle and requires care [21]. In this context, it is important to employ technical solutions that promote good security practices and that ensure that users have appropriate controls over how their data is being used. There are many [5] who advocate for a decentralized approach in which users directly manage their own identities via personal servers, Personal Identity Management Systems (PIMS). Meanwhile, blockchains, most notably Bitcoin [18], have provided new models of decentralization. In this work, we propose a sort of "light-PIMS," to be implemented on the Bitcoin blockchain. The decentralized nature of the blockchain allows us to create a neutral space where identity issuers and users share responsibility for users' identities, providing protections and the capacity for oversight for both parties.

Related Work. In 2015 MIT Media Labs introduced a system for academic certificates on the Bitcoin blockchain [20]. Taking advantage of the blockchain's

J. Garcia-Alfaro et al. (Eds.): DPM/CBT 2017, LNCS 10436, pp. 390–407, 2017.
DOI: 10.1007/978-3-319-67816-0_22

persistence over time, this system gives students a convenient way of proving that they graduated, see Sect. 2. The Blockstack project [6] has implemented decentralized versions of PKI and DNS on the Bitcoin network. In [11], a decentralized scheme to issue credentials in the absence of a trusted third party is proposed using Bitcoin. This scheme incorporates zero-knowledge protections such as those we will deal in Sect. 2.3. The startup CryptID [1] has proposed a system where encrypted records of fingerprints (along with a password) are stored in the Factom blockchain, which is itself periodically committed to the Bitcoin blockchain, replacing the traditional centralized server in fingerprint scanning identification systems with a more lightweight system. We generalize these ideas to permit more flexible user identities that can contain different fields of information useful in interacting with diverse service providers. We further explore the possibilities enabled by performing these interactions on a blockchain. Some architectures propose new, application designed blockchains. For example, the proposal of IDCoins [2] relies on a custom blockchain in which the proof of work is related to the generation of GPG/PGP keys necessary to create a web of trust. The Guardtime KSI blockchain, which forms the base of an electronic records system used in Estonia [4,23], is a permissioned blockchain. In [28] a system is proposed to store user information such as the GPS data from their phone in a distributed hash table and then store pointers to this data and permissions on how it may be used or retrieved on a blockchain. The proposition of ChainAnchor [14] even allows to create a semi-permissioned structure that can be placed on top of an existing blockchain such as that of Bitcoin by changing the incentive structure of miners to promote permissioned transactions. For a survey on other proposals that touch on the relationship between blockchains and identity management, see [16,27].

Our Contribution. We propose an identity management system that will take advantage of the decentralized nature of the Bitcoin blockchain to allow for a balance between the ability for users to manage their own identities and for issuers to establish controls. The different entities of our proposal communicate via Bitcoin transactions, allowing identity issuers to outsource much of the infrastructure required for this system to the Bitcoin network, which as the most robust, most established blockchain, has strong security properties, most notably, that miner's work maintains strong integrity of its data. Privacy during identity verification is ensured thanks to the attribute-based credentials of Brands [8]. While [11] already proposes using Brands credentials in Bitcoin, their protocol could, in fact, be implemented in any blockchain without major modifications. In contrast, our proposal takes advantage of the specifics of the Bitcoin scripting language to encode identity meaning in Bitcoin syntax. Specifically, we build upon the idea of MIT Media Labs [20] that revocation can be encoded in terms of the status of a Bitcoin transaction to enable additional mechanisms for issuer oversight which are then enforced by the Bitcoin network. Particularly, in our system an issuer can limit the number of times an identity can be used, see Sect. 3.5. At the same time, we will see that our system gives a great deal of control to the user over her identity.

Note that in traditional systems the reconciliation between user control and issuer oversight is problematic; in most systems the identity is generally controlled entirely by an on-line issuer with little input from the user [5], or alternatively the issuer will sign an identity to be managed by the user, then the issuer will go offline ceding his capacity for oversight (See [9] for a further discussion on the advantages and disadvantages of these two models.)

Compared to a traditional decentralized system, we offer more integrated issuer controls. For example, compare the revocation mechanism discussed in Sect. 3.4 to the challenges encountered using revocation lists in public key infrastructures (PKIs) [17]. Additionally, we will see that our system has the following advantages compared to centralized systems:

- Our system does not require identity providers to be as "lively" as they must be in traditional, centralized systems. If an identity provider has placed controls on an identity, such as a limit on the number of times it can be used, then even if an identity issuer has a service interruption, a user can continue to use her identity and these limits will continue to be enforced by the robust, worldwide Bitcoin network. A user can even revoke her own identity without intervention by the issuer.
- By providing a common space, control over which is shared between the different actors through the mechanisms of the blockchain, we allow users to coordinate several micro-identities, only needing to trust a small portion of their identities to any given identity provider, see Sect. 3.6. While a similar coordinate scheme is possible without a blockchain, in practice it is highly impractical for a user to coordinate identities from different identity providers each of whom uses his own distinct formatting and infrastructure.

On the other hand, our system has two (potential) drawbacks. First, as authentications are encoded in Bitcoin transactions, this requires paying transaction fees to miners, see Sect. 3.7 for an estimate of these fees. Second, Bitcoin transactions are by their nature public, posing risks to user anonymity. The typical suggestion to ensure (pseudo-)anonymity in Bitcoin is to use each Bitcoin address exactly one time. An analogous idea works here, at the expense of having higher user fees, see Remark 5. Note that users have differing standards regarding the privacy that they expect in their interactions. Some users may be willing to sacrifice some anonymity in exchange for lower fees. In fact, some users, such as those that gladly link their Facebook account to their Instagram account or their favorite blogs, may even prefer that metadata on their transactions be tied to them, allowing them to create a digital presence on which they can build a reputation. See Sect. 5 for a proposal on how a reputation system can be built on top of our architecture. A user should be empowered to make choices regarding how private they want to be.

2 Background

In this section we briefly recall some of the existing ideas, in Bitcoin and in the work of Brands [8], upon which our system is built.

2.1 Bitcoin Relevant Notions

It is a particularity of Bitcoin that all bitcoins exist in the form of Unspent Transaction Outputs (UTXOs) [7, Chap. 5], [18]. Each transaction may have several inputs, each of which was an output UTXO for some previous bitcoin transaction, and it may have several outputs. Most transaction outputs correspond to a bitcoin address, the hash of the public key that can spend it or a hash of a script detailing how the coin can be reclaimed. (These are called Pay to Public Key Hash P2PKH and Pay to Script Hash P2SH outputs respectively.) Particularly, one can create P2SH outputs that can then be spend by an m of n multisig. Also relevant to our work will be `OP_RETURN` outputs; each such output contains up to 80 bytes of space in which the sender of a transaction can store arbitrary information. Note, `OP_RETURN` outputs must have zero bitcoins associated to them; as such, they are provably not usable as inputs to later transactions.

The raw transaction that is broadcast to the nodes contains the amount of bitcoin to associate to each output, the script permitting validation of each output (P2SH, P2PKH, etc.), and the scripts for each input that satisfy the requirements set up when the corresponding input UTXO was created, generally including a signature from a corresponding private key. The hash of this raw transaction becomes the transaction identifier (txid), which is included in the Merkle tree that produces a block header and is ultimately recorded in the block chain in an immutable way. Thus, the given inputs and outputs of a given transaction are provably linked together.

Financial Friction in Bitcoin Transactions. Miners are compensated by "fees." The amount paid in fees for a given transaction is the difference between the combined values of the inputs and the combined values of the outputs. Miners, who are limited in how many bytes they can fit a given block, generally choose to include the transactions with the most profitable fees with respect to the number of bytes in its raw transaction [7, Chap. 5]. When discussing our schema, we will denote the fees for a given transaction by $\mathtt{F}_{\mathtt{NAME\text{-}OF\text{-}TRANSACTION}}$. See Sect. 3.7 for estimations of these amounts.

In order for a Bitcoin transaction to be considered valid it must satisfy certain basic properties such as not double spending a previously spent output, having valid signatures, etc. Any block that contains an invalid transaction will be rejected by the network. In addition, the Bitcoin Core software distributions to miners suggests requirements that transactions need to satisfy in order to be considered "standard." These requirements are implemented at the discretion of each miner and thus vary slightly across the network; a miner may refuse to include a given transaction in the blocks he mines as "non-standard," but if another miner broadcasts a block with this transaction in it, he will still accept that block if the transaction is valid. In particular, for a transaction to be considered standard, each of its non-`OP_RETURN` output must have a minimal value so as to prevent the network from being spammed by extremely low value transactions. Any amount of bitcoin below this minimum is called "dust." As of version 0.14 (March 2017), Bitcoin Core [12] recommends that miners refuse

transactions that have a P2PKH output of less than .00000546 bitcoin, currently (June 2017, 1BTC = 2720 USD) around .01 USD. We denote by $\mathcal{D}$ this minimal amount. Fees and the requirement to leave dust can greatly erode the value of a user's bitcoins if she engages in many transactions of small amounts.

2.2 MIT Media Labs Certificate Issuing Schema

We are inspired by the transaction structure used in [20]. In this system a certificate or diploma is issued to a user who completes a given program of study, encoded in a Bitcoin transaction. The transaction has a single input, from the credential issuer, so the transaction must be signed by the private key corresponding to the issuer's address. Hence, verifiers can be confident that credential was issued by an approved party. There are three outputs. The first is the Bitcoin address of the user. Then the user can authenticate herself as the holder of the credential by signing messages using the corresponding private key. The second output is to an address again belonging to the issuer. If this output is spent, the certificate is seen as being revoked. We view this revocation mechanism as a key innovation of [20], and we integrate and develop it into our system. Finally, the third output is an `OP_RETURN` that contains the certificate information. Note that as each of these UXTOs is thought of as having symbolic meaning, their bitcoin values are secondary; indeed, they are assigned values slightly larger than $\mathcal{D}$.

Input Addresses	Amounts	Output Addresses	Amounts
Issuer	.000155 BTC	Recipient	.000275 BTC
		Issuer (for revocation)	.000275 BTC
		`OP_RETURN`(Certificate info)	0 BTC
		Fees:	.0001 BTC

Fig. 1. Schema of an MIT certificate issuing transaction as in [20]. See, for example, txid: 41740ae0812e5a7804778f43c9fd1f8df50fe1bcd0545e9d627a83ab9d0d3d07

2.3 The DLREP Function

In [8], Brands proposed very efficient ways of revealing parts of an identity to verifiers, relying on discrete logarithms and hash functions. All the following is from [8]. Assume that n identity fields $X_1, \ldots, X_n$ are to be cryptographically blinded for further proofs. Let q be a prime number and G a group of order q, in which the discrete logarithm is hard. Typically, we take G to be the Koblitz elliptic curve secp256k1 where points are represented with 64 bytes (we use multiplicative notation for compatibility with [8]), namely we use the same G that is already being used for the Bitcoin signature protocol. Let $g_0, g_1, \ldots, g_n \in G$. Furthermore, there is the need (see Sect. 3) for an auxiliary random X_0 to protect unknown fields from a dictionary attack when the other fields are known.

Definition 1. *The tuple* $(X_0, X_1, \ldots, X_n) \in \mathbb{Z}_q^{n+1}$ *is called a Discrete Logarithm REPresentation (DLREP) of* $h = \prod_{j=0}^{n} g_j^{X_j} \in G$ *with respect to* $(g_0, g_1, \ldots, g_n)$.

To (non-interactively) prove knowledge of a DLREP of h to a verifier $\mathcal{V}$, a prover $\mathcal{P}$ performs the following protocol steps [8, Sect. 2.4.3]

1. $\mathcal{P}$ generates $n+1$ secret, random numbers $a_0, a_1, \ldots, a_n$ in G. Let $A = \prod_{j=0}^{n} g_j^{a_j}$, and compute c as $c = \mathcal{H}(A)$, where H is a one-way hash function.
2. $\mathcal{P}$ computes $b_j = a_j + cX_j$, $j = 0, 1, \ldots, n$ and sends them, as well as c to $\mathcal{V}$.
3. The verifier $\mathcal{V}$ checks that $\mathcal{H}(\prod_{j=0}^{n} g_j^{b_j} h^{-c}) = c$ holds.

Then, [8, Chap. 3] shows how the DLREP can be used to selectively prove properties about the X_j's, while any other information remains hidden. These techniques can be used to prove arbitrary satisfiable Boolean statements about the X_j's. For example, a prover can demonstrate that she is a French citizen AND that she is either under 18 OR over 65. $\mathcal{P}$ can prove (true) statements about her identity that contain an arbitrary number of ANDs, ORs, and NOTs in such a way that $\mathcal{V}$ only learns information that can be computed using the status of the formulas requested and information available a priori. See [8, Proposition 3.6.1] for a formal statement of this result. Brands [8] also shows that if the discrete logarithm problem is difficult, DLREP is one-way and collision-intractable, preventing an adversary from forging an identity with a given DLREP.

3 Our Proposal

3.1 Actors, Protocol Structure, and Security Assumptions

Our system will have three types of actors: **Identity Providers** ($\mathcal{IP}$), **Service Providers** ($\mathcal{SP}$), and **Users** ($\mathcal{USR}$). We borrow the following from [20]:

Definition 2. *An identity is a tuple* $(X_1, \ldots, X_n)$ *where each* $X_j \in \mathbb{Z}_q$ *stands for a different attribute, as exemplified below.*

An attribute X_j may represent a name, a date of birth, a social insurance number, medical or financial data, or some other personal information about a user. Typically, based on an identity provided by $\mathcal{IP}$, a user ($\mathcal{USR}$) wants to convince $\mathcal{SP}$ to give her access to its services.

Assumptions on Actors. We consider that both the Bitcoin addresses of $\mathcal{IP}$ and $\mathcal{SP}$ are well-established and public, $\mathtt{a}_{\mathcal{IP}}$ and $\mathtt{a}_{\mathcal{SP}}$ respectively. We will consider scenarios in which we have multiple identity providers and service providers, whose addresses are denoted $\mathtt{a}_{\mathcal{IP}_1}, \mathtt{a}_{\mathcal{IP}_2}, \ldots$ and $\mathtt{a}_{\mathcal{SP}_1}, \mathtt{a}_{\mathcal{SP}_2}, \ldots$ respectively. In contrast, $\mathcal{USR}$ may have different Bitcoin addresses $\mathtt{a}_{\mathcal{USR}}^{(1)}, \mathtt{a}_{\mathcal{USR}}^{(2)}, \ldots$ in order to

obfuscate the link between her identity transactions. When discussing a given user's address generally, we write $\mathtt{a}_{\mathcal{USR}}^{(i)}$ to indicate one her addresses. Note that a user should not re-use Bitcoin addresses that she has used for non-identification transactions, in order to not link this identity with her other Bitcoin activity.

We assume that all of $\mathcal{USR}$, $\mathcal{IP}$, and $\mathcal{SP}$ are capable of sending and receiving bitcoin and that they can perform operations in secp256k1. We will explore in Sect. 3.9 further technical requirements on the ability of $\mathcal{SP}$ to track Bitcoin transactions which will depend on $\mathcal{SP}$'s security requirements. We assume that $\mathcal{IP}$ validates a user's real world identity (via a more or less rigorous verification process) and then publishes documents that are correct. Furthermore, $\mathcal{IP}$ should handle user personal data in a way that respects user-privacy. Note that $\mathcal{IP}$ does not need to stay online for the identities it issues to be used, and only participates for issuing and revocation of identities, and certain exceptional maintenance, see Sect. 3.5. Service providers accept identities issued by identity providers they wish to trust. Note that service providers may fail or refuse to provide a service, a fact which can not be managed by our protocol. They may deviate from the protocol (at the risk of impairing their reputation, see below).

Assumptions on Bitcoin Network. We will use the *public ledger* functionality of Bitcoin: it is a "bulletin board" where anyone can post messages and read messages posted. More precisely, [13,22] provide the definitions of *liveness*, i.e. every honest participant will have its posted messages seen by every honest participant after some delay, and *persistence*, which means that every posted message will indefinitely be seen at the same position by all participants. We will also rely on the security semantics of the Bitcoin transaction verification procedure which ensure no double-spending, that each non-generation transaction has inputs linked to previous transaction outputs, etc. Under some quantitative bounds on the relative power of the adversary, be it computing power in [13], and or computing and network power [22], the Bitcoin core protocol is proven to securely provide these functionalities.

The above results are theoretical and quantitative. There could be real world situations in the Bitcoin blockchain where the adversary has enough power to violate the above quantitative bounds, and also accidental cases where problems occurs like small forks, peer-to-peer failures, etc. We will discuss the impact of these possible attacks and failures in Sect. 3.9 below.

Remark 1. Note that there are other relatively well-established blockchains such as Ethereum that can also serve as a "bulletin board." However, by working in Bitcoin, we can use the linking mechanism of Bitcoin transactions, which is not natively present in the account based model of Ethereum [26]. Also, the total hash power of the Bitcoin network is substantially greater than that of Ethereum [15], which can be seen as a sign that Bitcoin has a great resilience to 51% attacks.

There are three steps for our protocol: a **Setup phase**, an **Enrollment phase**, and an **Operational phase**.

3.2 Setup Phase

Each $\mathcal{IP}$ will choose some set of $g_0, g_1 \ldots, g_n \in G$ that will serve as the base for a DLREP function. These g_j should be public and readily available. For example, $\mathcal{IP}$ could create a series of Bitcoin transactions with inputs from his address in which the g_j and the fields they represent are stored in `OP_RETURN` outputs.

3.3 Enrollment Phase

During the Enrollment phase, $\mathcal{USR}$ brings to $\mathcal{IP}$ the (physical, biometric, etc.) elements required to assert that her identity indeed matches all the X_j's. This can be as strong as a physical meeting, in which the user shows a passport, or as light as an authentication on a web server, depending on the policy of $\mathcal{IP}$. During this phase $\mathcal{USR}$ should provide $\mathcal{IP}$ with a Bitcoin address $\mathtt{a}^{(i)}_{\mathcal{USR}}$ that she controls and an element $g_0^{X_0}$ to protect against dictionary attacks, where X_0 is chosen at random by $\mathcal{USR}$ so that $\mathcal{IP}$ does not learn it. Then, $\mathcal{IP}$ can form $h_{\mathtt{a}^{(i)}_{\mathcal{USR}}} = g_0^{X_0} g_1^{X_1} \ldots g_n^{X_n}$, as in Sect. 2.3.

The Enrollment phase corresponds to a single Bitcoin transaction, $\mathtt{TX}_{\mathtt{PUBLISH}}$. The primary purpose of this transaction is to record $h_{\mathtt{a}^{(i)}_{\mathcal{USR}}}$ in the blockchain; however, we see that this transaction will include other structure.

$\mathtt{TX}_{\mathtt{PUBLISH}}$ (Identity Establishment): $\mathcal{IP}$ sends amounts of bitcoin to two outputs. First a minimal amount of bitcoin $\mathcal{D}$ is sent to the user's address $\mathtt{a}^{(i)}_{\mathcal{USR}}$; this ties the user's address to the identity. Also, $\mathcal{IP}$ sends bitcoin to a 1 of 2 P2SH multisig of $\mathtt{a}^{(i)}_{\mathcal{USR}}$ and $\mathtt{a}_{\mathcal{IP}}$, denoted $\mathtt{MSIG1\_2}(\mathtt{a}^{(i)}_{\mathcal{USR}}, \mathtt{a}_{\mathcal{IP}})$, which we view as an **authentication token** that the user will spend upon using her identity. Moreover, either $\mathcal{USR}$ or $\mathcal{IP}$ can prevent further use of the token by $\mathcal{USR}$ by sending it to $\mathcal{IP}$ or even spending it to a random address. This should be seen as **revocation**. More precisely, when using her identity as described below in Sect. 3.4, $\mathcal{USR}$ will send transactions of a specific form that return bitcoin to the same multisg address of $\mathtt{a}_{\mathcal{IP}}$ and $\mathtt{a}_{\mathcal{USR}}$ leaving a transaction output for future authentications; if at any point $\mathcal{USR}$ or $\mathcal{IP}$ spend this output in a transaction that is not of the form of another authentication, then this transaction is a $\mathtt{TX}_{\mathtt{REVOKE}}$ and the identity is seen as revoked. Finally, an `OP_RETURN` contains $h_{\mathtt{a}^{(i)}_{\mathcal{USR}}}$.

The authentication token will be used in subsequent transactions; its amount V will be calibrated to cover the costs of these transactions, see Sect. 3.7.

Note that the structure of $\mathtt{TX}_{\mathtt{PUBLISH}}$ is similar to that of the transactions in the architecture of [20] as shown in Fig. 1. Now, revocation can be performed by both $\mathcal{IP}$ and by $\mathcal{USR}$ as both parties can destroy the authentication token via a $\mathtt{TX}_{\mathtt{REVOKE}}$.

Remark 2. There are alternative zero-knowledge selective credential systems in addition to that of Brands [8]. As discussed above, one advantage of using Brands' scheme is that its cryptographic primitives: discrete logarithms (in our

$\texttt{TX}_{\texttt{PUBLISH}}$ Input Addresses	Amounts	Output Addresses	Amounts
$\texttt{a}_{\mathcal{IP}}$	$V + \mathcal{D} + \texttt{F}_{\texttt{PUBLISH}}$	$\texttt{a}^{(i)}_{\mathcal{USR}}$	$\mathcal{D}$
		$\texttt{MSIG1\_2}(\texttt{a}^{(i)}_{\mathcal{USR}}, \texttt{a}_{\mathcal{IP}})$	V
		$\texttt{OP\_RETURN}\left(h_{\texttt{a}^{(i)}_{\mathcal{USR}}}\right)$	
		Fees:	$\texttt{F}_{\texttt{PUBLISH}}$

Fig. 2. Structure of $\texttt{TX}_{\texttt{PUBLISH}}$.

case on secp256k1) and hash functions are also primitives of Bitcoin, so we minimize the number of cryptographic assumptions necessary. Also, the commitments of Brands are small enough (a compressed elliptic curve point of 33 bytes) to fit in an `OP_RETURN`. In contrast this is not the case for example for the commitments of the Camenisch-Lysyanskaya scheme which produces commitments of 670 bytes [24, Table 2].

Remark 3. One can imagine cases where a hostile or hacked $\mathcal{IP}$ uses the authentication token to obtain services acting as if it were the user, possibly with the aim of harming the user's reputation. However, when spending a multisig output, it is visible which of the public keys one is signing by [7], thus such an attack would be visible and, in fact, damage $\mathcal{IP}$'s reputation.

3.4 Operational Phase

The Operational phase is made up of two further Bitcoin transactions. We think of certain outputs as being distinguished (or colored with a transferable semantic meaning in the sense of Colored Coins [10], [19, Sect. 9.2]), corresponding to the authentication token. The flow of this token will chain the transactions together and ultimately to the creation of the identity in $\texttt{TX}_{\texttt{PUBLISH}}$ (see Fig. 4). We suppose $\mathcal{SP}$ informs $\mathcal{USR}$ of what statement about her identity she needs to prove to authenticate. Then we have the Bitcoin transactions:

$\texttt{TX}_{\texttt{REQUEST}}$ **(Request for Service):** $\mathcal{USR}$ creates a transaction where the input is the $\texttt{MSIG1\_2}(\texttt{a}^{(i)}_{\mathcal{USR}}, \texttt{a}_{\mathcal{IP}})$ from $\texttt{TX}_{\texttt{PUBLISH}}$. One output is sent to $\texttt{a}_{\mathcal{SP}}$. One output is sent back to $\texttt{MSIG1\_2}(\texttt{a}^{(i)}_{\mathcal{USR}}, \texttt{a}_{\mathcal{IP}})$ and will serve as the authentication token for future transactions. $\mathcal{USR}$ proves to $\mathcal{SP}$ the required Boolean statement about the X_j' s without revealing them as in Sect. 2.3 (see below for a discussion of how this proof is transmitted and stored).

$\texttt{TX}_{\texttt{ACCEPT}}$ **(Acknowledgment of the Identity by** $\mathcal{SP}$**):** Upon validating the proof of $\mathcal{USR}$, checking that the authentication token is the result of a series of $\texttt{TX}_{\texttt{REQUEST}}$'s each of whose input is the output of the previous chained back to a $\texttt{TX}_{\texttt{PUBLISH}}$, checking that $\texttt{TX}_{\texttt{PUBLISH}}$ was issued by a trusted $\mathcal{IP}$, and verifying that the multisig output of the most recent $\texttt{TX}_{\texttt{REQUEST}}$ has not been spent (namely that there has not been a $\texttt{TX}_{\texttt{REVOKE}}$), $\mathcal{SP}$ accepts $\mathcal{USR}$'s authentication and uses its output from $\texttt{TX}_{\texttt{REQUEST}}$ to send bitcoins to $\texttt{a}_{\mathcal{IP}}$.

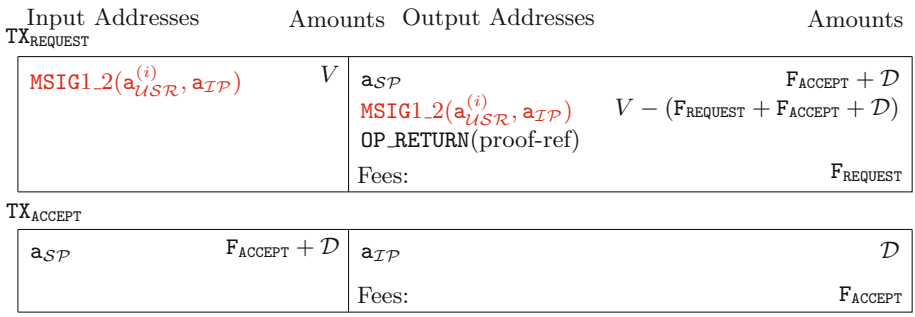

Fig. 3. The transactions that compose a typical authentication The inputs and outputs highlighted in red are thought of as an authentication token that chain the user's transactions together and to $\texttt{TX}_{\texttt{PUBLISH}}$.

Storage of Proofs. A careful reading of [8, Chap. 3] shows that the size of the Brands proofs required to demonstrate a given Boolean statement about an identity $(X_0, \ldots, X_n)$ scales linearly in n, but also depends on the statement being proven. We note that these proofs will generally be too large to be contained directly in an `OP_RETURN`. Depending on the needs of $\mathcal{USR}$ and $\mathcal{SP}$, we propose three different mechanisms by which these proofs might be transmitted and stored. 1. A user can store in the `OP_RETURN` of $\texttt{TX}_{\texttt{REQUEST}}$ a link to a site where the proofs are stored externally as well as a hash of the relevant contents of this site. We denote this information by proof-ref. The hash will be included in mined blocks, so the information on the site has the same protections against mutability as other information on the blockchain. This is similar to how metadata is stored in [10]. 2. A user that is very concerned about privacy, or who is proving a statement that is already sensitive, can transmit the Brands proofs entirely off-chain. 3. If one wants to avoid an off-chain storage mechanism, there are a number of non-`OP_RETURN` ways to store data in the Bitcoin blockchain (see [10]) such as in a vanity address or using a fake 1 of N multisig. Alternatively, one can issue a P2SH output in $\texttt{TX}_{\texttt{REQUEST}}$ with Pubkey Script `OP_HASH160` H(data) `OP_EQUAL` for which the corresponding input Sig Script is simply the data itself (see txid db195e4bfcfb3cc6d47f8d6231cb59e543c31e01d196d557457bca0fa5c1aba0). While there are still limits on how much data can be placed in a single input, through using multiple inputs, one can store larger amounts of data in this fashion in exchange for paying (much) higher transaction fees.

For the remainder of this article (and in Fig. 3) we assume that proofs are being referenced via a link and a hash in an `OP_RETURN`.

Remark 4. $\texttt{TX}_{\texttt{ACCEPT}}$ publicly shows that $\mathcal{SP}$ has accepted $\mathcal{USR}$'s identity proofs as valid, contributing to the reputation of $\mathtt{a}^{(i)}_{USR}$ (see Sect. 5). This is particularly useful if the proofs were conveyed off-chain or are otherwise unavailable. $\texttt{TX}_{\texttt{ACCEPT}}$ can also serve to alert $\mathcal{IP}$ that $\mathcal{SP}$ has used an identity that it provided, and can even be a basis for a payment by $\mathcal{SP}$ for the issuing of this identity.

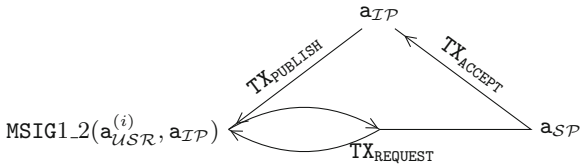

Fig. 4. Scheme for users to prove their identity to service providers. An identity is issued via a $\mathtt{TX}_{\mathtt{PUBLISH}}$. Subsequently, each transaction takes as input the output of a previous transaction ($\mathtt{TX}_{\mathtt{PUBLISH}}$ for the first authentication, $\mathtt{TX}_{\mathtt{REQUEST}}$ thereafter). For simplicity, the `OP_RETURN` output is not shown.

Remark 5. The transaction output of $\mathtt{TX}_{\mathtt{REQUEST}}$ to $\mathtt{MSIG1\_2}(\mathtt{a}^{(i)}_{\mathcal{USR}}, \mathtt{a}_{\mathcal{IP}})$ is necessary if the user wishes to reuse an identity, so that $\mathcal{USR}$ will have an UTXO tied to her identity to spend in a subsequent $\mathtt{TX}_{\mathtt{REQUEST}}$. This cyclic structure chains the various authentications together permitting a verifier to trace any of them back to the original identity issued in $\mathtt{TX}_{\mathtt{PUBLISH}}$. Alternatively, $\mathcal{USR}$ can obtain a new $\mathtt{TX}_{\mathtt{PUBLISH}}$ attached to a different address $\mathtt{a}^{(j)}_{USR}$ for each authentication if she wishes to maintain a more complete anonymity in her authentications. However, doing so is slightly more expensive as additional fees need to be paid for each $\mathtt{TX}_{\mathtt{PUBLISH}}$. Additionally, this limits the ability of a user to take advantage of the reputation system we propose in Sect. 5. (See the discussion in the introduction on user empowerment over her level of privacy and compare to [8, Chap. 5.2.1], where Brands discusses the balance between reputation and anonymity and proposes reuse solutions for his certificates.) In cases where the identity is only designed to be used one time, this transaction output is unnecessary.

3.5 Limited Use Identities and Setting Bitcoin Values

Due to the chaining of authentications, when $\mathcal{SP}$ verifies the continued validity of $\mathcal{USR}$'s identity, the number of times this identity has been used can also be calculated. Thus, if $\mathcal{IP}$ includes a use limit of N authentications with $h_{\mathtt{a}^{(i)}_{USR}}$ in the `OP_RETURN` of $\mathtt{TX}_{\mathtt{PUBLISH}}$, $\mathcal{SP}$ can check if this identity can still be used. Then, $\mathcal{IP}$ should calibrate the amount of bitcoin, V, that is placed in the $\mathtt{a}^{(i)}_{\mathcal{USR}}$ output to cover these N authentications. As we saw in Fig. 3 that each authentication consumes $\mathtt{F}_{\mathtt{REQUEST}}+\mathtt{F}_{\mathtt{ACCEPT}}+\mathcal{D}$ bitcoin before the authentication token is returned to the user, and this returned token needs to have a value of at least $\mathcal{D}$ after the last usage for the transaction to be accepted as standard, V must be at least $N(\mathtt{F}_{\mathtt{REQUEST}} + \mathtt{F}_{\mathtt{ACCEPT}} + \mathcal{D}) + \mathcal{D}$. Note, the fees required for a transaction to be processed in a timely fashion slowly vary based on market forces, so $\mathcal{IP}$ should, in practice, set V to be slightly larger than current market demands in case miners increase their fees. Then, situations requiring $\mathcal{IP}$ to come online and top up its users' balances can be limited to cases of extreme changes in Bitcoin fees.

3.6 Coordinating Multiple Identities

Suppose a given user has obtained identities $h_{\mathtt{a}^{(1)}_{\mathcal{USR}}}$ and $h_{\mathtt{a}^{(2)}_{\mathcal{USR}}}$ from more than one identity provider. We see that these identities can be coordinated.

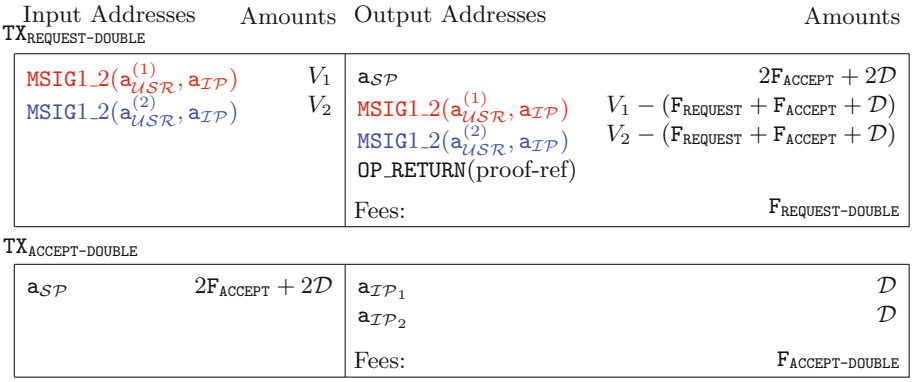

Fig. 5. Use of the authentication tokens of two identities together. The paths of these tokens are colored in red and blue. The Brands proofs referenced in proof-ref are with respect to the DLREP of Eq. 1

Concretely, suppose a user has been issued an identity by $\mathcal{IP}_1$ consisting of $h_{\mathtt{a}^{(1)}_{\mathcal{USR}}} = \prod_{j=0}^{n} g_j^{X_j}$ and another identity by $\mathcal{IP}_2$ consisting of $h_{\mathtt{a}^{(2)}_{\mathcal{USR}}} = \prod_{u=0}^{m} (g'_u)^{Y_u}$. A service provider will be able to verify that each of these values correspond to the respective $\mathtt{TX}_{\mathtt{PUBLISH}}$ transactions issued by the identity providers. Then

$$h_{\mathtt{a}^{(1)}_{\mathcal{USR}}} \cdot h_{\mathtt{a}^{(2)}_{\mathcal{USR}}} = \prod_{j=0}^{n} g_j^{X_j} \prod_{u=0}^{m} (g'_u)^{Y_u} \tag{1}$$

is a DLREP commitment of the union of $X_0, \ldots, X_n$ and $Y_0, \ldots, Y_m$. The user can do proofs with selective disclosure using this commitment.

We preserve the chaining properties by having two transactions $\mathtt{TX}_{\mathtt{REQUEST\text{-}DOUBLE}}$, which takes in the authentication tokens from both identities, and $\mathtt{TX}_{\mathtt{ACCEPT\text{-}DOUBLE}}$, which notifies both identity providers. The amounts used in these transactions are chosen as in Fig. 5 to ensure that a user's balances decrease by no more than what would have been the case for separate authentications with the two identities, in keeping with the calibration of V in Sect. 3.5. (We will see in Sect. 3.7, $\mathtt{F}_{\mathtt{REQUEST\text{-}DOUBLE}} \leq 2\mathtt{F}_{\mathtt{REQUEST}}$ and $\mathtt{F}_{\mathtt{ACCEPT\text{-}DOUBLE}} \leq 2\mathtt{F}_{\mathtt{ACCEPT}}$, so adequate fees are paid here; the change can be split between $\mathcal{USR}$'s authentication tokens for use in case of future Bitcoin fee increases, paid to $\mathcal{SP}$, or left to the miners to increase the speed of the transaction's approvals). This schema can obviously generalize to more than two identities.

Thus, a user can obtain many "micro-identities" - from the government, from her bank, from her employer, from her health care provider - which she can manage together without having to unnecessarily share information between her identity providers. This is very much in the spirit of a PIMS [5].

Remark 6. The ability of $\mathcal{USR}$ to issue a transaction as in Fig. 5, which requires signing with the private key corresponding to the address of each identity, is already a weak way of establishing that these identities belong to the same person. However, it is possible for malicious users to pool the private keys from identities corresponding to distinct people. $\mathcal{USR}$ can provide stronger proof of the connection of her identities if she shows as part of her proof in $\texttt{TX}_{\texttt{REQUEST}}$ that h and h' share common fields, such as name or social insurance number.

3.7 Estimates of Cost

We now estimate the costs of the transactions we have introduced in the preceeding sections. As mentioned in Sect. 2, Bitcoin miners have flexibility in what fees they demand. However, the current standard fee to have one's transaction processed in a timely manner is 360 satoshis, namely .0000036 bitcoins, per byte [3]. Based on our schema, $\texttt{TX}_{\texttt{PUBLISH}}$ will contain one input, one P2PKH output, one P2SH output, and an `OP_RETURN` that contains one (compressed) point on secp256k1. Hence the `OP_RETURN` contains 33 bytes resulting in a total transaction size of roughly 267 bytes (see [7, Chap. 2] for more information on the size of the various components of a Bitcoin transaction) costing .0009612 bitcoin. At current market rates (June 2017, 1BTC = 2720 USD), this corresponds to a minimum transaction fee of approximately 2.61 USD. We compute the sizes and costs of the other transactions similarly (based on proof-ref consisting of a 32 byte SHA-256 hash and a 30 byte url when necessary):

Transaction	$\texttt{TX}_{\texttt{PUBLISH}}$	$\texttt{TX}_{\texttt{REVOKE}}$	$\texttt{TX}_{\texttt{REQUEST}}$	$\texttt{TX}_{\texttt{ACCEPT}}$	$\texttt{TX}_{\texttt{REQUEST-DOUBLE}}$	$\texttt{TX}_{\texttt{ACCEPT-DOUBLE}}$
# Bytes	267	229	334	191	479	225
Cost (USD)	2.61	2.24	3.28	1.87	5.41	2.20

Then, building off Sect. 3.5, the total cost to issue an N use identity is the value of the input issued by $\mathcal{IP}$ in $\texttt{TX}_{\texttt{PUBLISH}}$. As in Fig. 2, this is

$$\begin{aligned}\text{Cost of } N\text{-use Id} &= V + \mathcal{D} + \texttt{F}_{\texttt{PUBLISH}} \\ &= N(\texttt{F}_{\texttt{REQUEST}} + \texttt{F}_{\texttt{ACCEPT}} + \mathcal{D}) + 2\mathcal{D} + \texttt{F}_{\texttt{PUBLISH}} \\ &\approx 5.2N + 2.6 \text{ USD}.\end{aligned}$$

Note that Bitcoin fees have increased substantially recently as the Bitcoin community seeks consensus on how to scale block capacity. It is hoped that a solution to this issue, such as an implementation of SegWit, will reduce fees [25].

3.8 Obtaining Information About the Bitcoin Network

Note that in the processing of an authentication, it is the service provider that must verify the status of past Bitcoin transactions. Service providers with rigorous verification requirements, such as banks and insurance companies, should run a full node or possibly a Simplified Payment Verification (SPV) client, see [7]. Note the SPV protocol, which is already commonly used by vendors who payment in Bitcoin, allows someone who downloads merely the 80 byte header of each block to verify that a given transaction has been included in a block, upon being provided with information related to that transaction by a full node. Hence, a service provider running this protocol can verify that each of the $\texttt{TX}_{\texttt{REQUEST}}$'s a user has issued, chained back to $\texttt{TX}_{\texttt{PUBLISH}}$, counting the number of times the identity has been used. This process also checks that the identity has not been revoked as the SPV client sees that the network has accepted the most recent transaction, so the transaction output controlled by the multisig between $\mathcal{USR}$ and $\mathcal{IP}$ could not have already been spent in a $\texttt{TX}_{\texttt{REVOKE}}$. Service providers with less rigorous standards may retrieve their information from an online block explorer if they accept the additional risks of attacks on these sites.

3.9 Security Considerations in Case of Blockchain Failures

In Sect. 3.1, we place ourselves in a security model in which Bitcoin possesses certain properties of an ideal blockchain. Here we explore the consequences on our system when these properties are not satisfied.

Inconsistencies in the Bitcoin ledger: The integrity of the Bitcoin ledger serves in our system to allow issuer oversight, concretely to allow the issuer to revoke identities and to impose limits on the number of uses. On the other hand, if there is a fork, a dishonest user can to continue to use an identity which an issuer has revoked until the revocation transaction finally appears in the dominant chain. If an attacker can issue a double spend (due to an accidental fork, because the attacker has a large percentage of the mining power, etc.), then she can reuse her authentication token allowing her to exceed her usage limit.

Bitcoin network failure: We also rely on Bitcoin P2P infrastructure to propagate the transactions that make up our protocol, and we rely on being able to download information on previous Bitcoin transactions from nodes to check the state of an identity. An attack on the P2P Bitcoin network can translate into a denial of service attack on our system as one cannot issue $\texttt{TX}_{\texttt{PUBLISH}}$, $\texttt{TX}_{\texttt{REQUEST}}$, etc. if the network does not relay them or if one cannot verify relevant previous transactions. How vulnerable a service provider is to network attacks will depend on how it receives information about the network as in Sect. 3.8. Note that, regardless of this choice, user privacy is protected and impersonation is prevented by the security of Brands' protocols, see [8]. Even a service provider that obtains its information from a block explorer can assure itself of the correctness of Brands proofs and the validity of signatures.

4 Example Use Cases

In this section we propose a few use cases of our system that highlight its advantages versus existing systems.

4.1 University ID

We consider a university where the administration delivers identity credentials to students, teachers, and staff. These credentials provide certificates of various fields related to the user including their name, their status at the university (student, teacher, etc.), and their academic records. Individuals may use such identities, revealing some (or none) of these fields, to authenticate themselves to various university services such as the university pool or medical clinic.

Now imagine that a user wants to claim a discount on car insurance reserved for students with high GPAs. This student may need to coordinate her university identity with a driver's license issued by her local government. Then she can selectively reveal information to the service provider, the insurer, using the multi-$\mathcal{IP}$ protocols described in Sect. 3.6. If her status at the university changes, her university identity can be revoked preventing her from performing such authentications, even as her driver's license identity remains valid.

4.2 Network of Small Museums

We imagine a group of small museums that form a partnership in which any member of one museum is allowed a limited number of visits to the other museums. In this case, the user is a member of one of the museums, the identity provider is the museum that issued the membership, and the other museums are service providers. Then the user may selectively disclose fields such as her membership status or category of membership. More sensitive information may be included in the identity allowing the user to authenticate to tax authorities which give a tax credit for museum memberships. The limit on the number of visits is controlled through the methods of Sect. 3.5.

In contrast to the tax authorities, the security requirements of the museums may allow them to obtain the transaction information from an online block explorer, completely outsourcing the costs of transmitting and storing information to the Bitcoin network similar to how [1] uses the blockchain as a virtual server. This may be substantially cheaper and more streamlined than traditional systems (namely, either for each of the museums to invest in infrastructure that then has to be coordinated or for a single museum to set up infrastructure to manage the entire system which may create conflicts of interest and be unacceptable to the other museums). Thus, our system allows the museums to create a shared, neutral management space, maintaining transparency into exactly how the data is stored and used, that minimizes infrastructure costs.

5 Building a Reputation on the Blockchain

We see in Sect. 4.2, in the case of our museums, that little infrastructure is required of $\mathcal{SP}$. Nonetheless, $\mathcal{SP}$ must be able to compute in secp256k1, perform Bitcoin transactions, and be able to access the blockchain history, as discussed in Sects. 3.1 and 3.9. Imagine that some very lightweight service provider wants to participate in this network, but does not have the security requirements, nor the resources to justify performing these operations. For example, this may be the case of a university pool in the university ID example of Sect. 4.

As all transactions are visible in the blockchain, a user can then simply direct a lightweight service provider to her past transactions, which requires merely an Internet connection, and prove that she controls the private key corresponding to those transactions by issuing a signature. Then, if the lightweight service provider is willing to trust the larger service providers that have already accepted the user's identity (e.g. if the university pool is willing to trust the campus medical clinic in accepting that the user is a member of the university community), it is not necessary to re-validate the relevant Brands proofs. As seen before (see Sect. 3.6), a user may have had her identity established under different Bitcoin addresses and proven to different service providers in such a way that is unknown that these addresses belong to the same user. If the user has used the two identities together in a $\texttt{TX}_{\texttt{REQUEST-DOUBLE}}$, the light service provider may be again willing to trust that the other service provider has verified these two identities as corresponding to the same person. Alternatively, in situations with lower security standards (as per Remark 6), the user can issue signatures for both of the private keys corresponding to the identities used.

Moreover, the collection of transactions of a user, seen as having been accepted via $\texttt{TX}_{\texttt{ACCEPT}}$ transactions, gradually forms a digital footprint of the user. While some users will want to avoid reusing the same $\texttt{TX}_{\texttt{PUBLISH}}$ for multiple authentications for greater anonymity, for other users this digital presence, over which the user has a great deal of direct control, can be a useful addition to the online reputation they develop, for example through social media.

6 Conclusion

The Bitcoin blockchain is a global network, and by building on top of this network, we can take advantage of its existing infrastructure to reach a global scope while minimizing overhead. Moreover, by placing an identity management system in this decentralized space, we have seen that we can strike a more equitable balance between the rights and responsibilities of users and identity issuers.

Acknowledgment. The work leading to this paper has received funding from the European Community's Framework Programme (FP7/2007-2013) under Grant Agreement n° 607049.

References

1. CryptID. Source code https://github.com/CryptidID/Cryptid. Consulted April 2017. http://cryptid.xyz/
2. IDCoins. Consulted April 2017. https://github.com/IDCoin/IDCoin
3. Predicting Bitcoin fees for transactions, Consulted April 2017. https://bitcoinfees.21.co/
4. Estonian e-residency, Consulted March 2017. https://e-estonia.com/e-residents/about/
5. Abiteboul, S., André, B., Kaplan, D.: Managing your digital life. Commun. ACM **58**(5), 32–35 (2015)
6. Ali, M., Nelson, J., Shea, R., Freedman, M.J.: Blockstack: a global naming and storage system secured by blockchains. In: 2016 USENIX Annual Technical Conference (USENIX ATC 2016), Denver, CO, USA, June 22–24, 2016. Proceedings, pp. 181–194 (2016)
7. Antonopoulos, A.M.: Mastering Bitcoin. O'Reilly Media, Sebastopol (2015). ISBN: 978-1-449-37404-4
8. Brands, S.: Rethinking Public Key Infrastructures and Digital Certificates (Building in Privacy). MIT Press, Cambridge (2000)
9. Camenisch, J., Lehmann, A., Neven, G.: Electronic identities need private credentials. IEEE Secur. Priv. **10**(1), 80–83 (2012)
10. Charlon, F.:. Open assets protocol (OAP/1.0) (2011). https://github.com/OpenAssets/open-assets-protocol/blob/master/specification.mediawiki
11. Miers, I., Garman, C., Green, M.: Accountable privacy for decentralized anonymous payments. In: Financial Cryptography and Data Security (2016)
12. The Bitcoin Core developers: Bitcoin transactions primitives code, Consulted March 2017. https://github.com/bitcoin/bitcoin/blob/0.14/src/primitives/transaction.h
13. Garay, J., Kiayias, A., Leonardos, N.: The bitcoin backbone protocol: analysis and applications. In: Oswald, E., Fischlin, M. (eds.) EUROCRYPT 2015. LNCS, vol. 9057, pp. 281–310. Springer, Heidelberg (2015). doi:10.1007/978-3-662-46803-6_10
14. Hardjono, T., Smith, N., (Sandy) Pentland, A.: Anonymous identities for permissioned blockchains, January 2016. http://www.the-blockchain.com/docs/MIT-ChainAnchor-DRAFT.pdf
15. Hay, S.: Bitcoin vs ethereum: Cryptocurrency comparison, March 2017. https://99bitcoins.com/bitcoin-vs-ethereum-cryptocurrency-comparison/
16. Jacobovitz, O.: Blockchain for identity management, December 2016. https://www.cs.bgu.ac.il/%7Efrankel/TechnicalReports/2016/16-02.pdf
17. Liu, Y., Tome, W., Zhang, L., Choffnes, D.R., Levin, D., Maggs, B.M., Mislove, A., Schulman, A., Wilson, C.: An end-to-end measurement of certificate revocation in the web's PKI. In: Proceedings of the 2015 ACM Internet Measurement Conference, IMC 2015, Tokyo, Japan, October 28–30, 2015, pp. 183–196 (2015)
18. Nakamoto, S.: Bitcoin: a peer-to-peer electronic cash system (2008). http://bitcoin.org/bitcoin.pdf
19. Narayanan, A., Bonneau, J., Felten, E., Miller, A., Goldfeder, S.: Bitcoin and Cryptocurrency Technologies: A Comprehensive Introduction. Princeton University Press, Princeton (2016)
20. Nazaré, J., Hamilton, K., Schmidt, P.: Digital certificates project. Source code https://github.com/digital-certificates. Consulted December 2016. http://certificates.media.mit.edu

21. Office of the Privacy Commissionner of Canada. Privacy and your reputation - who shapes your identity online? (2012)
22. Pass, R., Seeman, L., Shelat, A.: Analysis of the blockchain protocol in asynchronous networks. In: Coron, J.-S., Nielsen, J.B. (eds.) EUROCRYPT 2017. LNCS, vol. 10211, pp. 643–673. Springer, Cham (2017). doi:10.1007/978-3-319-56614-6_22
23. Prisco, G.: Estonian government partnerts with bitnation to offer blockchain notarization services to e-residents, November 2015. https://bitcoinmagazine.com/articles/estonian-government-partners-with-bitnation-to-offer-blockchain-notarization-services-to-e-residents-1448915243/
24. Security Team: Specification of the identity mixer cryptographic library version 2.3.0. Technical report RZ 3730, IBM Research, Computer Science Dept, IBM Research - Zurich, Switzerland, 48 pages (2010)
25. Torpey, K.: Are bitcoin miners making more money off small blocks? March 2017. https://bitcoinmagazine.com/articles/are-bitcoin-miners-making-more-money-small-blocks/
26. Wood, G.: Ethereum: a secure decentralised generalised transaction ledger, EIP-150 REVISION (030c1b5 - 10 July 2017). https://ethereum.github.io/yellowpaper/paper.pdf
27. Yang, D., Gavigan, J., Wilcox-O'Hearn, Z.: Survey of confidentiality and privacy preserving technologies for blockchains, November 2016. https://z.cash/static/R3_Confidentiality_and_Privacy_Report.pdf
28. Zyskind, G., Nathan, O., Pentland, A.: Decentralizing privacy: using blockchain to protect personal data. In: 2015 IEEE Symposium on Security and Privacy Workshops, SPW 2015, San Jose, CA, USA, May 21–22, 2015, pp. 180–184. IEEE Computer Society, Los Alamitos (2015)