PB-PKI: a Privacy-Aware Blockchain-Based PKI

Louise Axon and Michael Goldsmith
Department of Computer Science, University of Oxford, Parks Road, Oxford, UK
{*louise.axon, michael.goldsmith*}*@cs.ox.ac.uk*

Keywords: Public-key infrastructure; Blockchain; Privacy-awareness; Security

Abstract: Conventional public-key infrastructure (PKI) designs using certificate authorities and web-of-trust are not optimal and have security flaws. The properties afforded by the Bitcoin blockchain are a natural solution to some of the problems with PKI - in particular, certificate transparency and elimination of single points-of-failure. Proposed blockchain-based PKI designs are built as public ledgers linking identity with public key, providing no privacy. We consider cases requiring privacy-aware PKIs, which do not link identity with public key. We show that blockchain technology can be used to construct a privacy-aware PKI while eliminating some of the problems of conventional PKI, and present PB-PKI, a privacy-aware blockchain-based PKI.

1 INTRODUCTION

Internet communications rely on the security of their underlying public-key infrastructure (PKI), by which the keys that entities use to establish communications channels are managed. The conventional approach to PKI uses certificate authorities (CAs). Web-of-trust (WoT) models [Barenghi et al., 2015], and simple public-key infrastructure (SPKI) are also used to construct PKIs [Buchmann et al., 2013]. These approaches have security flaws: CAs are single points-of-failure; WoT PKIs have a high barrier to entry [Roosa and Schultze, 2013]. High-profile events such as the 2011 hacking of CA DigiNotar have encouraged work to improve PKI security [Leavitt, 2011].

An emerging solution to constructing secure PKIs is blockchain, a design for distributed public ledgers introduced as the transaction record underlying the Bitcoin cryptocurrency [Nakamoto, 2008]. In theory, blockchain meets many PKI requirements, and addresses some security problems of conventional approaches: in a decentralised blockchain-based PKI, the single points-of-failure that CAs represent are eliminated, and a ledger of PKI events is published that is reliable as long as the majority of blockchain contributors are honest [Fromknecht et al., 2014]. The construction of blockchain-based PKIs has been proposed in prior work [Hari and Lakshman, 2016].

Proposed blockchain-based PKI solutions provide desirable security properties; however, since they link entities publicly with public keys, they are unsuited to applications in which a level of privacy is required. In PKI applications such as the Internet of Things (IoT), ad-hoc networks and smart cards, preventing tracing of entities and their actions is important. We therefore address privacy-awareness in blockchain-based PKI.

In this paper, we adapt Certcoin, a blockchain-based PKI [Fromknecht et al., 2014], to be privacy-aware. Our contribution, PB-PKI, does not publicly link identity with public key. PB-PKI provides *unlinkable short-term key updates* and *user-controlled disclosure*, in which a user's identity and previously used public keys can be disclosed either by the user himself, or through consensus of a network majority.

We begin by presenting relevant background on conventional approaches to PKI and the use of blockchain in Section 2. In Section 3, we consider the type of privacy-awareness required for a set of PKI use-cases. We present PB-PKI in Section 4, and discuss its uses and limitations in Section 5. We conclude and indicate future work in Section 6.

2 BACKGROUND

2.1 Conventional Approaches to PKI

Public key cryptography requires entities to have a public and secret key pair. A PKI manages these keys, usually based on certificates which provide verification of ownership of a public key by some entity. PKIs must support the registration and update of public keys, and provide mechanisms, e.g., key revocation, for coping with key compromise or loss.

The most common approach to PKI is CA-based – specifically, the X.509 standard. CAs are trusted parties, who will issue a signed certificate verifying an entity's ownership of a public key on request. In order to "trust" a CA, a device accepts a root certificate for that CA into its store. A hierarchical certificate chain stems from this root, in which any certificates signed using a trusted certificate are also trusted.

WoT-based PKI are also widely used. Members of the network establish trust by verifying that others have a certificate signed by an entity in whom the verifier has previously established trust. Unlike in CA-based PKI, trust is decentralised in WoT – certificate issuance can be performed by any party.

2.2 Blockchain-Based PKI

Blockchain was first introduced as the transaction record for the Bitcoin cryptocurrency [Nakamoto, 2008]. Alternative blockchains have since been developed, including the Namecoin blockchain, on which Certcoin and PB-PKI are built. Namecoin works as a decentralised domain name server (DNS) which, unlike the Bitcoin blockchain, is able to store data, making it suitable for wider applications [Kalodner et al., 2015]. A blockchain is a public ledger to which events are posted and verified by network members, before being "mined" in an incentivised system in which members compete to complete some proof-of-work – usually a cryptographic challenge.

Blockchain has a unique combination of properties that make it suitable for a number of applications: it is decentralised (it is controlled through majority concensus of members), and the transaction record is reliable (events recorded in the past cannot be altered without consensus of a majority of the network's mining power). Proposed and existing applications include smart contracts, reputation systems, and IoT device interactions. In theory, blockchain provides desirable security properties for PKI: certificate transparency and revocation, elimination of central points-of-failure, and a reliable transaction record.

Building decentralised PKIs using blockchain removes the potential points-of-failure created by the use of CAs which, if subverted, can compromise entire certificate chains [Ellison and Schneier, 2000]. Furthermore, blockchain-based PKI, as a public append-only log, naturally provides the certificate transparency (CT) property implemented by Google to improve CA-based PKI security through public logging and monitoring of certificates [Laurie, 2014].

Blockchain-based PKI also has potential advantages over WoT-based PKI, where the need to establish trust results in a high barrier to entry. The amount of work required to build a web that proves "trustworthiness" to a usefully large proportion of the network is significant. In blockchain-based PKI, entities do not require this web of attesting members, so the work needed to perform as a network member is removed.

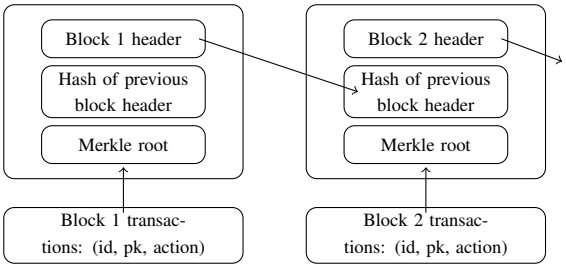

Figure 1. Blockchain PKI structure.

The structure of blockchain-based PKI is illustrated in Figure 1. Key registration, update and revocation are performed by posting the identity and the public key to the blockchain as a transaction. The contents of previously mined blocks are hashed and contained with the following block, creating a reliable transaction record that can only be altered by a network majority mining power. The merkle root is a hash of transactions per block, and can be used to securely verify transactions, eliminating the need to download the entire blockchain for verification.

Blockchain-based PKIs proposed in prior work do not provide privacy-awareness. We focus on Certcoin blockchain-based PKI [Fromknecht et al., 2014], the output of a Massachusetts Institute of Technology class project, on which PB-PKI is based. Certcoin is not suitable where privacy is required, since it is built as a ledger to which identity is posted publicly with public key, along with action (registration, update, verification). Hence all actions carried out using a public key can be traced to the owning identity by any entity who viewing ledger. Furthermore, the key update verification stage links all updated public keys to the updating entity's previous public keys.

3 PRIVACY-AWARENESS IN PKI

Privacy-aware PKI describes PKI built to protect user privacy, where we consider privacy to be *the ability of the user to control their disclosure of information*. Prior work has detailed the need for privacy in PKI [Brands, 2000], and proposed approaches to its provision, e.g. using group signature schemes [Ren et al., 2008]. To our knowledge, no privacy-aware blockchain-based PKI has been constructed, and existing proposals are unsuited to uses requiring privacy.

3.1 Privacy-Aware PKI Use-Cases

We provide a set of use-cases in which PKI are used, but in which the linking of public keys with identities is undesirable. Such situations arise particularly where it is required that entities' actions cannot be tracked by their use of public keys.

- **Ubiquitous computing and the IoT.** A user's interactions with a computing system may occur through multiple devices such as laptops and smartphones, and IoT devices such as wearables. A user's actions and location may be traced if linked across multiple devices. Privacy is required such that the user's identity and public key cannot be linked across devices [Zeng, 2006].
- **Vehicular networks.** PKI is required for secure inter-vehicular communications, but its use must not enable remote tracking of a vehicle's actions. The identity corresponding to a public key must not be publicly disclosed, or keys linked at update.
- **Anonymous forums and networks.** Such networks requiring user anonymity need a PKI in which users can verify network membership, but need disclose no further information pertaining to their identity or linking their separate actions. In this case, an entity's public keys should be frequently updated, and key updates not linkable. Identity should not be linked with public key.
- **Smart cards.** Smart cards have multiple uses – authenticating payments, and proving credentials or identity. A single smart card may be used in multiple locations and for multiple purposes, so its use should not be traceable by repeated use of the same public key, or by linkable updates.

3.2 Notions of Privacy

The privacy levels needed in the identified use-cases vary. For example, anonymous forums require complete anonymity, whereas for vehicular networks the PKI should prevent tracking by all entities except those remaining within line-of-sight of the vehicle (it being futile to prevent tracking in this case). We can concede a lower level of privacy for the latter case, in which each entity's actions are linkable by a small subset of "neighbours" in the network. We term such a subset a *neighbour group*. We address the varying privacy needs using *total anonymity*, *neighbour group anonymity*, and *user-controlled disclosure*.

- *Total anonymity:* for each entity **E**, no other entity can link a public key owned by **E** to any other of **E**'s public keys, or to **E**'s identity. Network members pool public keys within view of the rest of the network, so that messages can be sent to non-specific network members, or broadcast to the network as a whole, without knowledge of the identities corresponding to the public keys.
- *Neighbour group anonymity:* the actions of an entity **E** are identifiable within a *neighbour group* containing **E**, but **E** remains *totally anonymous* to the rest of the network. Members of a *neighbour group* disclose some identifying, or key-linking, information at updates to their *neighbour group*. The other members can then attest the correctness of the update to the rest of the network.
- *User-controlled disclosure:* each entity chooses to disclose their identity or past public keys.

There is a trade-off between PKI security and the level of privacy the PKI provides: with the concession of some anonymity in order to move from *total anonymity* to *neighbour group anonymity* comes an increase in the functionalities of the PKI, with respect to tracing misbehaving entities in particular. Security is weaker in the case of *total anonymity*, and *neighbour group anonymity* provides better security properties. The basic version of PB-PKI, which we present in Section 4, provides *total anonymity*; we then show how *neighbour group anonymity* can be achieved.

To derive the required functionalities for PB-PKI, presented below, we adapted the requirements for Certcoin (the necessary functionalities listed for that PKI [Fromknecht et al., 2014]) to provide *total anonymity*. The requirements for registration and update are identical. Processes for look-up, verification and revocation of a public key are required with respect to a given identity in Certcoin, but with respect to the network in PB-PKI. It should therefore be possible to verify that a public key corresponds to *some* network member, and to revoke it from the network.

1. Registering an identity with a public key
2. Updating the public key corresponding to a previously registered identity
3. Looking up a public key valid on the network
4. Verifying that a public key is valid on the network
5. Revoking a public key from the network

4 PB-PKI: A PRIVACY-AWARE BLOCKCHAIN-BASED PKI

PB-PKI is a privacy-aware adaptation of the Certcoin blockchain-based PKI [Fromknecht et al., 2014] described in Section 2. Our proposal for PB-PKI takes root in the observation that in order to achieve *total anonymity*, as described in Section 3, identity and public key should not be publicly linked. For trac-

ing and revocation purposes, in the case of misbehaving entities on the network or of key compromise, the link between an identity and its public keys should be available when required (by law, for example).

PB-PKI avoids the public linking of public key with identity, or with previous public keys, by publicly separating the identity value *id* from the short-term public keys *pk* posted to the blockchain. In Certcoin, both identity and public key are posted at registration and update, while in PB-PKI once an identity *id* is established, its key updates are anonymous.

PB-PKI also provides decentralised control over access to linking information. The identity owner may choose (*user-controlled disclosure*) to reveal the hidden links between his past key updates (e.g., to prove ownership or action in case of key compromise), using his offline secret keys (see Section 4.1). The link can also be revealed upon concensus of a network majority (for legal investigations, for example).

In Figure 2 we show the structure of PB-PKI. An entity posts identity and public key to the blockchain at registration, and then posts a new public key, without identity, at updates, storing the offline and secret keys. Table 1 presents the setup and initial key registration processes for PB-PKI, and Table 2 shows its key update process, compared with Certcoin.

4.1 Key Updates in PB-PKI

The main difference between PB-PKI and Certcoin is the key update process. In PB-PKI, this process gives no public link between the updated public key and either identity or the previous public key. Instead, a hidden link (an offline key pair) is created between key updates that traces back to an initially posted identity. The user can thus update his public key anonymously.

We illustrate the offline key linking process of PB-PKI in Figure 3. For an entity **E**, *pkf* and *skf* are the offline public and secret keys respectively, while *pkn* and *skn* are the online keys. The new online public key at each update is computed as a function of the previous online public key and the offline secret key. Hence, while each short-term online public key *pkn* posted is publicly unlinkable to the last, **E** retains a linking record of his offline keys (used in the online key update function). **E** can use this record to prove his ownership of past online keys and prove the link between his public key and his identity. In this way, a chain of online public keys is created that can be verified right back to the initial identity registration, and which the identity owner can choose to reveal.

As shown in Table 2, the offline key pair at each update is randomly generated, as in the initial registration stage. A nonce R_n is also randomly generated for the n^{th} update. In order to update a key, entities must prove that they are network members – that they already have a current online public key registered, as detailed in the *update verification* stage.

For the key updates, as shown in the *offline/online key generation* stage in Table 2, the n^{th} online public and secret keys are generated as functions of the previous online key pair (pkn_{n-1}, skn_{n-1}) and the newly generated offline key pair (pkf_n, skf_n). This creates a verifiable chain of anonymous online public key updates for an entity owner **E**, which **E** can choose to disclose by proving that his chain of online public keys were generated using his offline keys (which should only be known by **E**). In the case of compromise of these offline keys, an adversary could impersonate **E**, and we therefore require the master offline key pair $(mpkf, mskf)$. This is an additional security feature: the master public key $mpkf$ is posted at the initial identity registration (see Table 1) and the owner can use his master secret key $mskf$ to prove identity

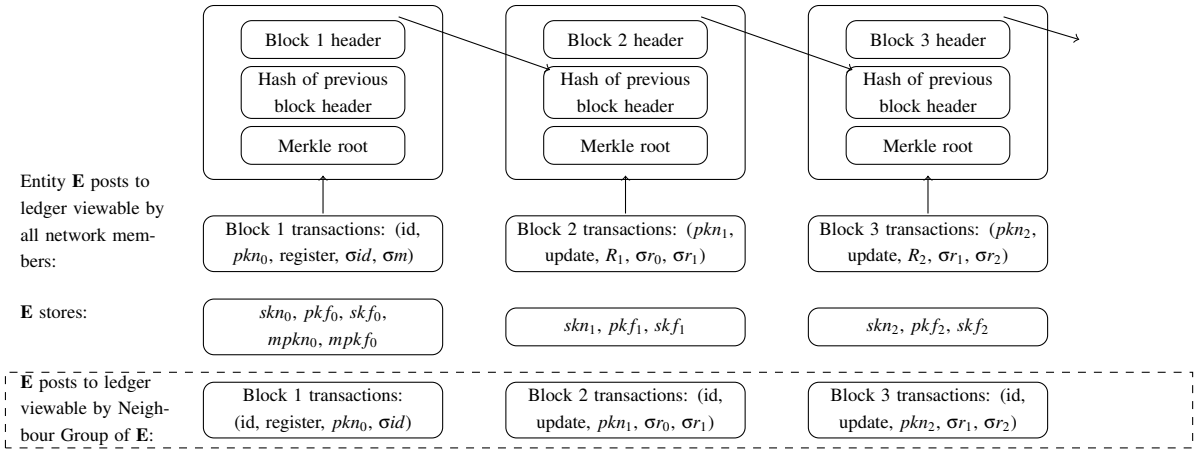

Figure 2. PB-PKI structure.

TABLE 1. Comparison of Certcoin and PB-PKI structure: setup and initial registration.

Certcoin	**PB-PKI**
sig is a digital signature algorithm; ver is a signature verification algorithm evaluating to 0 or 1 **Offline/online key generation**: Identity owner **E** generates (locally): • an online public and secret key pair (pkn_0, skn_0) • an offline public and secret key pair (pkf_0, skf_0) **Key registration**: **E** posts: • (id, register, online, values=(pkn_0, σ_{n0})), where σ_{n0}=sig(skn_0, id) demonstrates ownership of the online secret key skn_0 corresponding to public key pkn_0 • (id, register, offline, values=(pkf_0, σ_{f0})), where σ_{f0}=sig(skf_0, id) demonstrates ownership of the offline secret key skf_0 corresponding to public key pkf_0 **Verification**: It must be verified that: • id has not been registered previously • ver(pkn_0, σ_n, id)=1 • ver(pkf_0, σ_f, id)=1	sig is a digital signature algorithm; ver is a signature verification algorithm **Offline/online key generation**: Identity owner **E** generates (locally): • an online public and secret key pair (pkn_0, skn_0) such that $pkn_0 \times skn_0 \equiv 1(modNn)$, where Nn is the online key pair modulus • an offline public and secret key pair (pkf_0, skf_0) such that $pkf_0 \times skf_0 \equiv 1(modNf_0)$, where Nf_0 is the offline key pair modulus • a master offline key pair $(mpkf, mskf)$ **Key registration**: **E** posts (id, register, online, T_0, values=(pkn_0, σ_i, σ_m)), where T_0 is a timestamp, σ_i is the initial value signature sig(skn_0, id) - the identity id signed with online private key skn_0 (this proves **E**'s ownership of the online secret key skn_0 corresponding to online public key pkn_0) - and σ_m is the master key pair signature sig($mskf$, id) - the identity id signed with the master offline private key $mskf$. The remaining information generated (skn_0, pkf_0, skf_0, $mpkn$, $mpkf$) is retained by **E** **Verification**: It must be verified that: • id has not been registered previously • pkn_0 has not been registered previously • ver(id, pkn_0, σ_1)=1 **Encryption and Digital Signatures** A message m can be encrypted using **E**'s online public key pkn_0 to send to **E**, and decrypted by **E** using the online secret key skn_0 as $(m^{pkn_0})^{skn_0} \equiv m^1 \equiv m(modNn)$. Similarly, **E**'s digital signature on a message m^{skn_0} can be verified using the online public key pkn_0, as $(m^{skn_0})^{pkn_0} \equiv m(modNn)$.

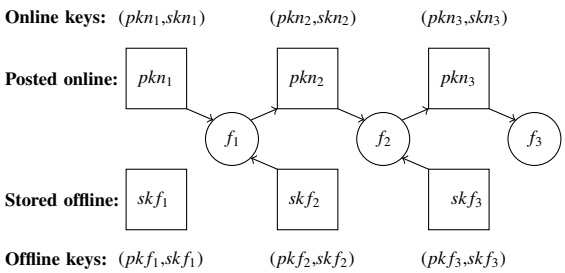

Figure 3. Using offline keys for hidden linking process in PB-PKI.

ownership in case of lost offline keys.

In order to enable tracing in case of misbehaviour, at the point of updating his online public key each entity should share his offline secret key between a majority of the network using a secret-sharing scheme. This is further detailed in Table 2 and Section 4.2 and means that a misbehaving entity can be traced if required through collusion of a majority of the network.

If at any time only one network member performs a key update, the previous key can be linked with the new one based on time, so the transaction becomes linkable. There are two ways of addressing this.

- **Random time delay.** Where *total anonymity* is required, a random time delay can be instigated from the time of posting a new online public key to the time to old one is discarded. During this time delay, both are valid public keys for the entity. This prevents key updates from being linked based on time, while preserving *total anonymity*.
- **Simultaneous key updates.** Where *neighbour group anonymity* is appropriate, *neighbour group* members can update their keys simultaneously whenever any single member needs to update.

4.2 Recovery, Revocation and Tracing

We begin by summarising the key recovery and revocation methods for Certcoin, and then present those for PB-PKI. In Certcoin, recovery of lost keys is enabled through social backup: the secret key for an entity must be secret shared (by Shamir secret sharing, for example [Shamir, 1979]) between trusted "friends", and reconstructed with a threshold. Both online and offline secret keys can be reconstructed through the key shares in the case of key loss.

The revocation process in Certcoin differs depending on which key is accessed or stolen. If only the online secret key is lost or stolen, then the ownership of the offline secret key means the true owner can prove his ownership of the secret key. If the adversary gains access to, but does not steal, both online and offline secret keys, then the adversary cannot be distinguished from the true owner, so the owner can use both keys to invalidate their use. If both keys are

TABLE 2. Comparison of Certcoin and PB-PKI architecture: key updates.

Certcoin	**PB-PKI**
Offline/online key generation: For the n^{th} key update, identity owner **E** generates: • an online public and secret key pair (pkn_n, skn_n), or • an offline public and secret key pair (pkf_n, skf_n) **Update registration**: **E** posts, for online/offline key update: • (id, update, values=(pkn_{n-1}, pkn_n, σ_{nn1}), σ_{nn2}, aux), where σ_{nn1}=sig(skn_{n-1}, (id, pkn_n) is the identity and new public key, signed by the old secret key, and is given to demonstrate ownership of the old secret key skn_{n-1} corresponding to public key pkn_{n-1}. σ_{nn2}= sig(skn_n, id) is the identity signed by the new secret key, and is given to demonstrate ownership of the new secret key skn_n corresponding to new public key pkn_n. aux is an auxiliary message that may be required in case of key compromise, or • (id, update, offline, values=(pkf_{n-1}, pkf_n, σ_{fn1}), σ_{fn2}, aux), where σ_{fn1}=sig(skf_{n-1}, (id, pkf_n) is the identity and new public key, signed by the old secret key, and is given to demonstrate ownership of the old secret key skf_{n-1} corresponding to public key pkf_{n-1}. σ_{fn2}= sig(skn_f, id) is the identity signed by the new secret key, and is given to demonstrate ownership of the new secret key skf_n corresponding to new public key pkf_n **Update verification**: It must be verified that: • pk_{n-1} corresponds to id • ver(pk_{n-1}, σ_{n1}, (id, pk_n))=1 • ver(pk_n, σ_{n2}, id)=1	**Offline key generation**: Identity owner **E** generates a new offline public/secret key pair, for the nth update (pkf_n, skf_n), such that $pkf_n \times skf_n \equiv 1(modNf_n)$. **Online key generation**: for the n^{th} online key pair, **E** calculates: • $pkn_n = \mathbf{f}_1(pkn_{n-1}, skf_n, Nn) = pkn_{n-1} \times skf_n(modNn)$, and • $skn_n = \mathbf{f}_2(skn_{n-1}, skf_n, Nn) = skn_{n-1}/skf_n(modNn)$. Then for the updated key pair, $pkn_n \times skn_n \equiv (pkn_{n-1} \times skf_n) \times (skn_{n-1}/skf_n) \equiv pkn_{n-1} \times skn_{n-1} \equiv 1(modNn)$. Note: this is cryptosystem-specific and the above assumes a Discrete Logarithm Problem-based cryptosystem. For an elliptic curve-based cryptosystem, for example, the above values would need to be constructed differently **Update registration**: **E** posts (update, T_n, values=(pkn_n, R_n, R_{n-1}, $enc_{PK}(\sigma_{rn-1})$, σ_{rn})) • T_n is a timestamp, • R_n, R_{n-1} are nonces, • σ_{rn-1} is the signature sig(skn_{n-1}, R_{n-1}) (this is used to verify that **E** already owns a public key on the network). It is posted, encrypted with a function enc_{PK} that encrypts σ_{rn-1} with each of a set of public keys PK of a subset of network members to be involved in the verification; the subset is chosen randomly at each update, and • σ_{rn} is the signature sig(skn_n, R_n) (this proves **E**'s ownership of the online secret key skn_n corresponding to new online public key pkn_n) **E** then secret shares (using e.g. Shamir secret sharing scheme [Shamir, 1979]) the updated offline key skf_n between a majority of network members **Update verification**: It must be verified that: • number of ids=number of public keys • pkn_n has not been registered previously • ver(R_{n-1}, pk, σ_{rn-1})=1. This step verifies that the submitter already had a registered public key previously. Here, pk is some public key currently registered as in use on the network. Verifiers (those in the subset PK of network public keys, involved in this verification) check that one current online public key in the PKI satisfies this verification. • ver(R_n, pkn_n, σ_{rn})=1 • the offline secret key skf_n has been secret shared between a majority of the network

stolen, then there is no revocation process in Certcoin; the keys are controlled by an adversary. Certcoin public keys expire after a given lifetime.

In PB-PKI, a lost online secret key can be reconstructed using the offline secret keys, since online secret keys are a function of offline secret keys and previous online public keys (see Table 2). For this reason, online secret keys should be stored offline after updating. If an offline secret key is lost, then the owner cannot prove his identity, and the key must be retrieved or the identity re-established. The owner can invalidate his online public key using his online secret key, and then return to the registration stage, re-establishing identity against a new online public key.

Another option is social backup, as in Certcoin: an owner may choose to secret share his offline secret keys with a few trusted "friends". In the case of offline secret key loss, the key can then be reconstructed from the secret shares. If both secret keys are lost, then the offline secret key can be retrieved through social backup, as described, and used to prove the link between current online public key and identity. The online public key can then be invalidated, and identity registered again against a new online public key.

An entity can revoke his online public key by invalidating it using his online secret key. If a key is compromised, an owner has several possible actions, depending which keys are affected. We consider an attacker with all possible types of key access or theft.

- **Case 1: online secret key only, accessed.** The owner may use his online secret key to invalidate the online public key, before re-establishing his identity against a new online public key.
- **Case 2: both secret keys, accessed.** The owner may use his online secret key to invalidate the online public key, then re-register identity against a new online public key. The adversary, in possession of the online secret key, could perform the same action, invalidating the online public key. An adversary's capabilities against the targeted

entity end here, since the identity becomes void.

- **Case 3: online secret key only, stolen.** The owner may use his offline secret keys to prove his ownership of the current online public key (since $pkn_{n-1} \equiv \frac{pkn_n}{skf_n} (modNn)$), and thus authenticate an invalidation of that public key, before re-registering identity with a new online public key.
- **Case 4: both secret keys, stolen.** The owner may reconstruct his offline secret keys using social backup, and use this to prove ownership of the current online public key. The case is then reduced to Case 2 – access. If social backup has not been enabled, then the owner may prove ownership of the master key pair $(mpkf, mskf)$ used in the initial *key registration* stage (see Table 1), and use this to invalidate the current online public key.
- **Case 5: all keys, including past offline secret keys, and master keys, stolen.** There is no mechanism for dealing with this case, in which the owner cannot prove identity using either past offline, or master, keys. Past offline keys and master keys should be stored separately for this reason.

Where identity must be traced (e.g., for liability cases in vehicular networks) authorities can ask a public key to disclose his offline secret keys and nonces, present and historic. The original identity posting can thus be traced back, and identity reliably retrieved. This procedure is secure against identity spoofing, since without knowledge of all their offline secret keys, an entity cannot pose as any network member other than himself. If the public key owner does not cooperate, a network majority can use the key shares from each update to reconstruct the keys of the target.

4.3 Neighbour Group Anonymity

Entities in the PKI may form *neighbour groups* of "trusted" members. This trust may be based on social knowledge, in online forums, or on physical proximity, in vehicular networks, for example. Since entities are not anonymous within their *neighbour groups*, other group members can attest the correctness of their actions (key updates, revocations) to the rest of the network, improving the security of certain PKI functionalities. The members of these neighbour groups can also perform simultaneous key updates, preventing linking of key updates by timestamp.

Vehicular ad-hoc networks are an example of an application in which *neighbour group anonymity* is appropriate. In this case, it is unnecessary to seek unlinkability of actions towards the group of entities that are physically nearby, the prevention of tracking through keys by whom is futile. Vehicles in physical proximity can therefore form temporary *neighbour groups*, attesting to the rest of the network that the correct entity is performing key updates. Systems for the management of these trusted groups across different use-cases are outside the scope of this paper.

5 DISCUSSION

An adversary might have some attack capabilities against PB-PKI aside from those gained from stealing or accessing a party's secret keys. Network members have greater adversarial capabilities than non-network members. Since the verification process at key updates involves only checking that the party involved is a network member, an adversarial network member may update the public key of a targeted party to a new public key under his control. Similarly, an adversarial network member may revoke the online public key of another member, preventing communication until ownership is proved and the key retrieved.

In the *total anonymity* case, it is possible for network members to cause disruption by updating or revoking other members' keys in this way. Our mechanism allows retrieval of prior keys and therefore identity, but does not prevent other members from changing the keys for the period of time until retrieval. Since network members are anonymous, such attacks cannot be targeted at identities but at public keys. *Neighbour groups* prevent adversarial network members from attacking in this way, since members of a targeted party's *neighbour group* can attest to the network that the change was not initiated by the correct identity, and the change would not be processed.

To achieve the *total anonymity* privacy level using PB-PKI, there is a trade-off with security. As described, an entity **E** can be temporarily disrupted, leaving a time gap in which an adversarial network member may pose as **E**. In the *total anonymity* case, this is intuitively the best we can do. To prove ownership of the current public key, or prove the identity at the time of update, would mean revealing either the current and updated keys together, or identity – enabling either linking of public keys, or knowledge of the identity to which the new public key belongs. This security weakness can be addressed using *neighbour groups* who can attest the correctness of an entity's actions in cases where this privacy level is sufficient.

The security of the Namecoin blockchain itself relies on the majority of miners being honest parties. A collusive majority of dishonest network members could undermine the security of PB-PKI, since its security relies on the underlying blockchain being unsubverted. PB-PKI enables identification of adversarial network members by the two methods be-

low, which correspond to the *total anonymity* case. For *neighbour group anonymity*, these methods apply, and the members of the misbehaving user's *neighbour group* can also aid identification.

- **Majority consensus.** The offline secret key sharing process at each update (see Table 2) means that a majority concensus of network members can provide the secret key shares required to reconstruct the offline secret keys of the target. Using this, authorities can trace back through updates to recover the identity.
- **Network member consent.** Each network member can choose to prove ownership of previous public keys, or the link between public keys and identity. This means that authorities can order a misbehaving public key to reveal this information.

We consider the efficiency of network member verification, which involves traversing all public keys in the network until one is met which satisfies the verification algorithm; this confirms the network membership of the entity posting the updated public key. The time involved in this traversing process can become part of the blockchain proof-of-work described in Section 2.2, as long as it is smaller than the required proof-of-work for that particular network. In Certcoin, the efficiency of network membership verification is improved using cryptographic accumulators. This method can be applied to improve the efficiency of the PB-PKI key update verification stage.

6 CONCLUSIONS AND FUTURE WORK

Constructing PKI using blockchain is a viable alternative to the conventional CA-based and WoT-based approaches, and offers desirable security properties, but existing proposals not provide the privacy-awareness that is required of PKI in certain present and emerging applications. In PB-PKI, we showed how a blockchain-based PKI can be constructed to provide varying levels of privacy-awareness. The proposal achieves *total anonymity* at some security cost: network members may tamper in the short term with the public keys of others. A slightly lower level of privacy can be achieved through attestation by *neighbour groups*, who verify key changes at updates, and the security of the PKI is improved by this adjustment.

As future work, we will explore the management of group trust for particular use-cases of PB-PKI, and formally assess its security. We intend to investigate methods for improving the efficiency, experimentation with which is outside the scope of this paper. We intend also to implement a proof-of-concept to assess the feasability and efficiency of PB-PKI.

References

[Barenghi et al., 2015] Barenghi, A., Di Federico, A., Pelosi, G., and Sanfilippo, S. (2015). Challenging the trustworthiness of pgp: Is the web-of-trust tear-proof? In *European Symposium on Research in Computer Security*, pages 429–446. Springer.

[Brands, 2000] Brands, S. A. (2000). *Rethinking public key infrastructures and digital certificates: building in privacy*. MIT Press.

[Buchmann et al., 2013] Buchmann, J. A., Karatsiolis, E., and Wiesmaier, A. (2013). *Introduction to public key infrastructures*. Springer Science & Business Media.

[Ellison and Schneier, 2000] Ellison, C. and Schneier, B. (2000). Ten risks of PKI: What you're not being told about public key infrastructure. *Computer Security Journal*, 16(1):1–7.

[Fromknecht et al., 2014] Fromknecht, C., Velicanu, D., and Yakoubov, S. (2014). Certcoin: A namecoin based decentralized authentication system 6.857 class project. Unpublished class project.

[Hari and Lakshman, 2016] Hari, A. and Lakshman, T. (2016). The internet blockchain: A distributed, tamper-resistant transaction framework for the internet. In *Proceedings of the 15th ACM Workshop on Hot Topics in Networks*, pages 204–210. ACM.

[Kalodner et al., 2015] Kalodner, H., Carlsten, M., Ellenbogen, P., Bonneau, J., and Narayanan, A. (2015). An empirical study of namecoin and lessons for decentralized namespace design. In *Workshop on the Economics of Information Security (WEIS)*. Citeseer.

[Laurie, 2014] Laurie, B. (2014). Certificate transparency. *Queue*, 12(8):10.

[Leavitt, 2011] Leavitt, N. (2011). Internet security under attack: The undermining of digital certificates. *Computer*, 44(12):17–20.

[Nakamoto, 2008] Nakamoto, S. (2008). Bitcoin: A peer-to-peer electronic cash system. *Consulted*, 1(2012):28.

[Ren et al., 2008] Ren, W., Ren, K., Lou, W., and Zhang, Y. (2008). Efficient user revocation for privacy-aware pki. In *Proceedings of the 5th International ICST Conference on Heterogeneous Networking for Quality, Reliability, Security and Robustness*, page 11. ICST.

[Roosa and Schultze, 2013] Roosa, S. B. and Schultze, S. (2013). Trust darknet: Control and compromise in the internet's certificate authority model. *IEEE Internet Computing*, 17(3):18–25.

[Shamir, 1979] Shamir, A. (1979). How to share a secret. *Communications of the ACM*, 22(11):612–613.

[Zeng, 2006] Zeng, K. (2006). Pseudonymous PKI for ubiquitous computing. In *Public Key Infrastructure*, pages 207–222. Springer.