Received: 5 July 2019 | Revised: 25 September 2019 | Accepted: 16 October 2019
DOI: 10.1002/spy2.97

ORIGINAL ARTICLE

A privacy-preserving Blockchain with fine-grained access control

Carlisle Adams

School of Electrical Engineering and Computer Science, University of Ottawa, Ottawa, Ontario, Canada

Correspondence
Carlisle Adams, School of Electrical Engineering and Computer Science, University of Ottawa, Ottawa, Ontario, Canada.
Email: cadams@uottawa.ca

Funding information
Natural Sciences and Engineering Research Council of Canada, Discovery Grant #210361

Abstract

This article proposes a simple, efficient, and easy-to-use mechanism to add privacy and fine-grained access control features to a traditional Blockchain. It uses standard cryptographic algorithms and techniques, along with a novel key derivation algorithm and a fuzzy extractor component (that derives a cryptographic key from a biometric), to make access control functionality very simple for nonexpert users. Such a Blockchain would be suitable for the storage of participant data postings in long-term–isolated environments.

KEYWORDS
access control, Blockchain, cryptography, key derivation, mental health, privacy

1 | INTRODUCTION

In a long-term–isolated environment (such as an extended submarine voyage, arctic research station project, or deep-space mission), there are many concerns, but one that has been receiving increasing attention in recent times is the mental health of the participants. Typically, in such environments, there is a relatively small team of people; this team is isolated from “the rest of the world” for long periods (weeks or months). They may have no, or only intermittent, access to the Internet and generally have no other forms of communication with the outside world (ie, phone calls, letters, radio, television, etc.). Consequently, there is interest in detecting early warning signs of depression, anxiety, suicidal thoughts, and other forms of deteriorating mental wellness.

In these isolated environments, there is often one (or perhaps more) participant with medical training, such as a Chief Medical Officer (CMO). However, technology is increasingly being used to assist the CMO with his/her assessment: the environment may be equipped with multiple video cameras to observe behavior, facial expression, reaction times, level of agitation, and so on, of all participants; microphones may record conversations; and wearable sensors may record physical characteristics (such as heart rate, sleep patterns, and so on).

Another technology that may be applicable to such environments is text analysis. For example, participants may generate written text, either as part of their official duties or in their free time (eg, through blog articles, diary entries, or social media messages for posting on Twitter/Facebook/etc. when an Internet connection is re-established). In such cases, it may be possible for software to mine the text for words or phrases that may indicate mental problems; this mining can use sentiment analysis and other known text mining techniques.

However, clearly such text mining can only be done with the permission of the participant who wrote the text (particularly for personal [ie, nonduty-related] writing). Thus, a mechanism is needed that will allow a participant to enable another participant (such as the CMO), or a group of participants, to read and perform textual analysis on specific entries. Clearly, there are commercial products that provide such fine-grained access control over a database of records.

However, given the particular scenario that is being examined (ie, possible mental health deterioration), it is also required to ensure that a database record cannot be modified once it has been written, either by the data author or by any other participant in the environment. For this, the permanence and immutability properties of a Blockchain (or distributed ledger) are very attractive. Unfortunately, however, typical Blockchain implementations do not allow fine-grained control over access to portions of data in the blocks, and furthermore do not provide any level of data privacy (Blockchain implementers normally explicitly desire transparency of data and transactions; the only privacy provided is anonymity of the entities that perform the transactions).

In this article, we propose a cryptographic scheme to construct a privacy-preserving Blockchain with fine-grained access control over who can read specific entries. This novel scheme takes a basic Blockchain structure and adds both privacy and access control for the data stored on the chain (properties that are not found in typical Blockchain implementations). An important component of this scheme is a novel key derivation algorithm that minimizes the burden on the user (in terms of what the user needs to remember) while securely constructing all the keys needed for fine-grained access control over the various portions of data that the user wishes to share. The proposed mechanism is simple, efficient, and would be very easy to use even for a participant with deteriorating mental health conditions.

2 | BACKGROUND

An example of a long-term–isolated environment is an extended space mission, such as Mars exploration or moon habitation. Over the course of such a mission, the astronauts will be exposed to long periods of weightlessness, solar and cosmic radiation, atmospheric pressure, and periods of no (or severely delayed) communications with Earth.

NASA has stated (see, for example, their "Human Research Roadmap" website[1]) that the extended duration of current and future missions, coupled with the isolated, confined, and extreme environments of space travel, give rise to the possibility of adverse cognitive or behavioral conditions (including developing mental disorders) which can affect crew health and performance. Thus, NASA and similar organizations (such as the Canadian Space Agency) support research that can help to lower the risk of behavioral and psychiatric conditions during space missions.

One avenue that researchers are exploring in this direction is a decision support system to assess mental health conditions by analyzing clinical and behavioral data, including texts, voice signals, video signals, and wearable body sensor measurements. As mentioned above, however, much of this data will be private to the individual astronaut, prompting a need for long-term immutable storage with strong privacy protection, as well as fine-grained access control by the astronaut over his/her specific entries. In this article, we propose such a system based on Blockchain technology. (For the sake of illustration, we focus on text data posted by participants, but other types of data are equally applicable.) Note that a Blockchain is clearly not the only technology that could serve in this environment: a secure database is also a viable option. However, as mentioned, the long-term immutability property of Blockchain makes it especially attractive in this scenario; the question, then, is how best to add data privacy and access control to this technology.

In recent years, there has been much discussion about the pros and cons of privacy in Blockchain applications (see, for example, several recent papers[2-4]). Note, however, that much of that discussion has focused on protecting the identities of *users* of the Blockchain (ie, the property of anonymous or pseudonymous creators of transactions), rather than on the privacy of the transactional data itself (as is required for our target environment). For long-term–isolated environments, the participants in the system are all known and visible to everyone, but the content of the data that they post must have privacy protection. The authors of a recent blog[5] give a survey of privacy options, but many of these have a high cost in terms of implementation complexity or computational requirements.

There has also been research into providing access control in Blockchain applications (see, for example, several recent papers[6-8]). Many of these proposed schemes are quite complex (either to implement, or to use, or both) and are not ideally suited to our environment of interest (because the access control is not sufficiently fine-grained, or because policy and metadata management overly increases the user burden, or because the size of the required data storage grows unreasonably large).

Our goal is to design a simple, efficient, easy-to-use mechanism that provides both privacy and fine-grained access control in a system that will be a very good fit for our target environment.

3 | MATERIALS AND METHODS

To create a privacy-preserving Blockchain with fine-grained access control, we employ the following materials: standard cryptographic algorithms for encryption, hashing, and digital signatures; a biometric authentication primitive for master key construction; and a traditional Blockchain infrastructure. Our overall approach is to use the methods listed below (presented in point form to show a very high-level overview of the functionality of our system):

- Hashing to derive the symmetric keys that guarantee the desired privacy and access control features (see Section 4.1).
- Encryption for confidentiality (see Section 4.1).
- The Blockchain to define syntax and format for transactions and blocks, and to provide long-term immutability of all data (see Section 4.1).
- A fuzzy extractor component to achieve security, authenticity, and ease-of-use for all participants, even those with deteriorating mental health conditions (see Section 4.2).
- Signatures for authenticity and integrity (see Section 4.3).

Each method in the above bulleted list requires further explanation to describe how it achieves the security and/or usability goals of our proposed system. The next section presents the details of the creation and operation of our design, fleshing out the brief bullet points given above with concrete construction procedures.

4 | DETAILED DESIGN

The proposed system uses well-known and trusted cryptographic algorithms: AES-256 for encryption; SHA3-512 for hashing; ECDSA with a 521-bit prime modulus for digital signatures; and a hash chain of hash trees to construct the Blockchain. Note that the algorithms and parameters provide 256 bits of security throughout the system; alternative algorithms can of course be chosen as long as all algorithms provide at least a given minimum level of security.

4.1 | Constructing the Blockchain

Assume that Alice has K, an initial random string of 256 bits (see the following section for how to create this initial string).

At initialization time, for all users, we choose a minimum duration; this determines the smallest amount of Alice's data that she can make available to Bob at a time. For the sake of simple illustration, assume that we choose a duration of a single day. This means, for example, that Alice can reveal to Bob all of her Twitter posts for a given day, but cannot give him access to only the posts from 1:00 PM to 2:00 PM on that day.

Each duration creates a single block on the Blockchain. Within that block, all of Alice's postings will be stored, separated by the "venue" of the posting (eg, all of the Facebook posts will be together, all of the blog entries will be together, all of the Tweets will be together, etc.). These postings make up a hash tree of entries for a given duration, and the consecutive durations form a hash chain of blocks on the Blockchain.

Alice derives symmetric encryption keys from her initial string K as follows.

- The duration key DK = H(K, *duration*). For example, if the duration is a single day, then the duration key for 30 October 2018, would be H(K, "October 30, 2018"). (Any format for specifying a given duration is acceptable as long as all participants use the same format.)
- The venue key DK_v = H(DK, *venue*). For example, if the venue is Twitter, then DK_{Twitter} = H(DK, "Twitter"). Again, any agreed-upon format for identifying venues is acceptable.

Note that the chosen hash function (SHA3-512) has a 512-bit output size. Any agreed-upon process can select 256 bits for the AES key (eg, the first half of the hash, the second half of the hash, an XOR of the two halves, etc.). Furthermore, AES can encrypt using CBC mode, counter mode, or any other agreed-upon mode of operation.

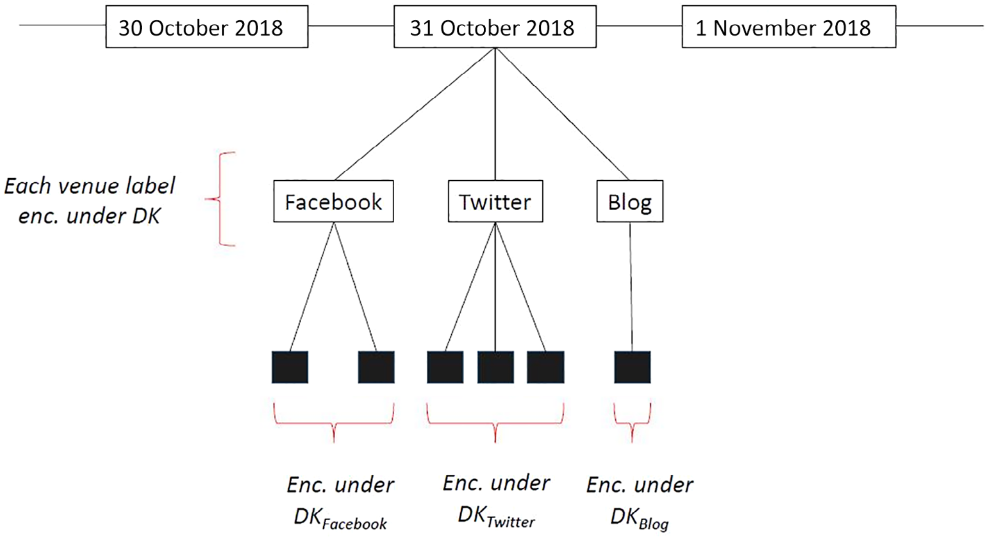

FIGURE 1 The constructed Blockchain

This key derivation scheme is highly efficient: the two hashes that are required to create a venue key can be computed in milliseconds, far faster than the time it would take for Alice to compose any amount of text to post in that venue. AES-256 encryption is also very fast: over 48 MB/sec in C++ software on a modest Pentium processor (see, for example, a 2006 paper[9]). Thus, the overall performance penalty for Blockchain privacy and access control is clearly negligible.

The Blockchain has the following construction (see Figure 1).

As can be seen in Figure 1, the duration (in this example, a specific day) is in plaintext, visible to everyone. This label identifies the root of the hash tree for that particular duration. Under that root, the various venues form the next layer of the tree, and DK (the duration key) encrypts each venue identifier individually. Under each venue, the various postings for that venue form the leaves of the tree, and DK_v (the venue key) encrypts each posting individually. Thus, every individual posting explicitly points to its venue and every venue to its duration (this is the property of a hash tree). Note, however, that the system designer can additionally choose to link the leaves for a particular venue together in a hash chain if he/she desires to guarantee the chronological order of postings within a given venue (ie, each posting points to the previous posting, as well as to its venue).

It is important to note that key derivation algorithm described above simply creates the keys that encrypt the text content that is posted. In all other respects, the overall structure is a traditional Blockchain (a hash chain of blocks). That is, blocks conform to a specified syntax and format, and each block (which the poster digitally signs) contains the hash of the previous block. This is why the immutability property holds. The only difference between this and a traditional Blockchain is that a "transaction" is not (eg,) a transfer of Bitcoin from one entity to another, but rather an array of free-form text within a specified transaction data structure. Thus, a sequence of "transactions" in a block (ie, the black squares in Figure 1) does not convey a provenance of ownership of an item (such as a Bitcoin), but rather conveys a record of (potentially independent) text postings within a given duration.

If Alice wishes to share her postings with Bob, she can give him DK for 31 October 2018, which will allow him to decrypt the different venue identifiers, derive the different venue keys, and decrypt all of her postings for that day. Alternatively, Alice can give to Bob only $DK_{Twitter}$, for example, along with the ciphertext AES(DK, "Twitter"). Bob can easily compare this ciphertext with the different nodes in the second layer of the tree to determine which branch contains the Twitter posts, and then use $DK_{Twitter}$ to decrypt all of Alice's Tweets for that day. Of course, it is also possible for Alice to give Bob her initial string K; this will give him access to everything she has posted (or will post) in the entire system.

How does Alice give to Bob the specific keys she wishes him to have? There are multiple possibilities and the system designer can choose the one that is most appropriate for his/her target environment. For example, every participant can have an encryption key pair (ie, a public key certificate and a corresponding private key); this works, but implies a public key infrastructure (PKI) in which Alice can obtain Bob's certificate and validate it using a CA public key that she trusts (ie, a trust anchor), including certificate path processing, revocation checks, and so on.

Another option is that Alice and Bob establish a secure channel (eg, using SSL/TLS or an authenticated Diffie-Hellman protocol), but this will also require some form of certificate infrastructure in order to function correctly.

Finally, in some environments (such as the long-term–isolated environment that is the focus of this article) it may be possible for this key transfer to happen out-of-band: Alice and Bob meet physically and her application downloads the specified keys to his application. This method can certainly work in our context, but of course does not scale well to larger environments.

In any case, once the system designer has chosen an effective key transfer method, Alice has fine-grained control over access to her postings, at the granularity of *duration* and above. If *duration* is set at 1 hour, or 10 minutes, the granularity is very fine indeed, but the price is increased size of the Blockchain and increased computation/processing in deriving and transferring all the necessary encryption keys. Note that key sharing happens *after* Alice has incorporated a duration block into the Blockchain. Thus, even though Bob now knows a valid venue key or duration key, he is not able to pretend to be Alice by adding new postings to that duration (and, of course, such an attack is also prevented because the full duration block was signed by Alice when she submitted it). Furthermore, the key that Bob knows is not valid for posting (as Alice) to any future duration block.

4.2 | Creating the initial string K

How is Alice's initial string K (which serves as her master key in this system) created? Recall that in our environment of interest, we are concerned with possible deterioration of the mental wellness of participants. Thus, we cannot assume that participants will be able to remember, or even properly enter as input, long strings of random bits or complex (hard-to-guess) passwords. Furthermore, since participants and devices are not necessarily fully trusted, we want to avoid storing Alice's string on her computer (or anywhere else).

We can accomplish this by deriving K from a biometric of Alice (ideally, a biometric that will not change drastically with mood or mental state and will not be easily found elsewhere in the environment; a good candidate might be a retinal scan). Note that the choice of biometric is important. Some biometrics, such as fingerprints, may serve well as public identifiers, but are not secrets: Alice's fingerprints typically exist all over her environment, including her coffee cup, pen, desk, doorknob, and so on. On the other hand, a retinal scan (particularly if the scanner includes liveness detection and does not store the scan after use) is a strong identifier for Alice that retains its secrecy.

The use of a biometric to create a key builds on work that has been done in the area of fuzzy extractors,[10,11] which attempts to derive very high entropy cryptographic keys from "fuzzy" (variable) input like biometrics. Such schemes often make use of an underlying error correcting code C and, while many variations are possible, can work (in very general terms) as follows: given a biometric value w, choose a random value x, compute a code word $c = \mathrm{C}(x)$ from x, and compute $p = w \oplus c$. The resulting value p is called "helper data" and is not sensitive (ie, it can be stored publicly). For our system, p will be stored on Alice's computer. The value x is the random string that we use as Alice's initial string K; it is generated at initialization time using a cryptographically strong pseudorandom number generator with device- and human-randomness as a seed. At a subsequent time, Alice submits a new biometric w' (eg, the system takes a new retinal scan). Given w' and p, the system computes $c' = p \oplus w'$, corrects c' to c, and decodes c to x (ie, Alice's K is recreated). Such fuzzy extractor schemes are predicated on the assumption that Alice's new biometric will be sufficiently close to her original biometric that the error correcting code will be able to correct these "errors" to recover the original code word. On the other hand, the biometric of anyone that is not Alice will be sufficiently far (by some distance measure, such as Hamming distance) from her original biometric that the error correcting code will fail and output an incorrect code word, thus yielding a completely different value for x.

Note that, as desired, Alice's computer does not need to store her value K in order to authenticate her at login. Instead, a small file encrypted with K can be stored on her computer. When Alice logs in, her application derives a key from her biometric; if that key properly decrypts this file then it is the correct K and she can begin composing her blog, Tweet, or Facebook post.

4.3 | Adding a new block to the Blockchain

Depending on the application and the choice of the system designer, there may be a need for a new block from Alice to be authenticated (as truly coming from Alice) before it is officially and permanently added to the Blockchain. For our environment of interest, there may not be a need for complex proof-of-work or other types of consensus algorithms (as seen in other Blockchain applications); however, it may still be desirable for Alice to append a digital signature to her

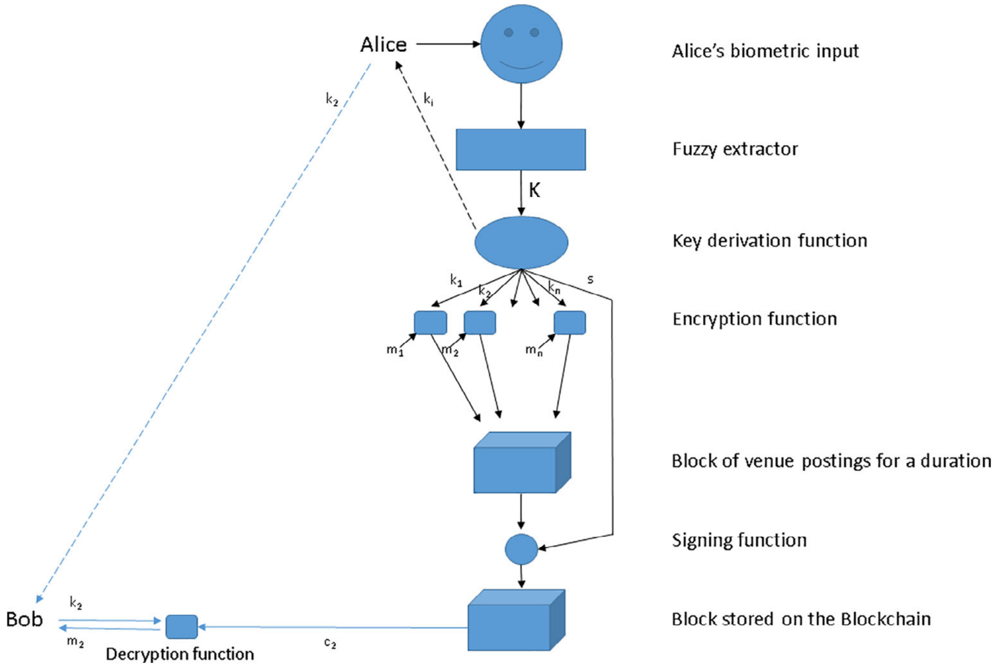

FIGURE 2 Overall process

submitted block so that others can verify that it came from her before adding it to the chain. We can use ECDSA signatures for this (as is done in the Bitcoin Blockchain, for example).

Again, for our environment of interest (potentially deteriorating mental wellness), a primary consideration is user ease and convenience, but this must be achieved without sacrificing security or privacy. Therefore, we do not want Alice to have to know many keys, nor do we want her keys to be stored on her computer (or on any device in the system). We accomplish this as follows. For ECDSA, choose public parameters p, a, b, G, n, and H, where

- p is a prime of size 521 bits.
- a and b are integers in Zp, and $y^2 = x^3 + ax + b$ (mod p) is the equation defining the elliptic curve.
- G is a point on the curve (a generator point).
- n is the order of G (note that n is approximately equal to p).
- H is a hash function with a 512-bit output (eg, SHA3-512).

Given those parameters (which are known and used by all participants), let $s = \mathrm{H}(\mathrm{K}, p, a, b, \mathrm{G}, n)$ be Alice's private signature key; her public verification key is then $\mathrm{V} = s\mathrm{G}$. As with her duration and venue encryption keys, this signature key pair is efficiently derived from her biometric; Alice does not need to remember or input any kind of password, and the key is regenerated at login and securely erased when she logs off and so it is not stored on any device. Note that her computer (or a separate server) can store a copy of her public signature verification key so that it can easily confirm that she has generated the correct private key upon login (using a simple challenge-response protocol, for example).

4.4 | Overall process

The overall process of our proposed construction is shown in Figure 2. Alice encrypts her various postings in a block on the Blockchain, and gives to Bob the keys for the specific postings she is willing to have him access. More specifically, Alice's biometric (eg, her retinal scan) is input to the fuzzy extractor component, which produces a uniformly random key, K. The key derivation function derives n symmetric encryption keys from K, along with a private signing key s. The symmetric keys are used to encrypt Alice's n postings (for a given duration and venue); these encrypted postings are assembled into a block. The key s is used to digitally sign the block, and the signed block is appended to the Blockchain. Alice has the keys that were output from the key derivation algorithm; if she wishes to share posting #2 with Bob, she shares key k_2 with him. Bob can then read ciphertext c_2 from the Blockchain, decrypt it with k_2, and read the resulting plaintext posting m_2.

5 | RESULTS AND DISCUSSION

The proposal given above describes the basic construction of a privacy-preserving Blockchain with fine-grained access control. Extensive technical evaluation of our proposed construction (in terms of a detailed security analysis and a proof of privacy preservation) is beyond the scope of this article but, in brief, relies largely on the security properties of the well-studied cryptographic algorithms it employs: the illegibility of AES-256 encryption; the unforgeability of ECDSA-521 digital signature operations; the one-wayness and collision-resistance of SHA3-512 hashing; and the randomness of fuzzy extractor-generated binary strings. Our proposal combines these (unaltered) algorithms in a novel construction that adds data control features to a simple Blockchain structure. Our threat model is a computationally bounded attacker that has access to the Blockchain and to all communications between participants in the environment, but does not know the input biometric of the user being targeted. This attacker wishes to learn the content of a specific post by Alice stored on the Blockchain. Alice can protect the privacy of her data by fully controlling with whom it is shared in the presence of such an attacker.

The above basic construction achieves the privacy and access control goals we desired for our environment of interest. However, we envision a number of variations and extensions for other applications or environments; we discuss some of these in this section.

- The above description creates a Blockchain for Alice's postings, so there is one Blockchain per participant in the system. Alternatively, it is possible to create a single Blockchain for all participants. For this, all that is required is to add the participant identifier as the second level in the tree (ie, below *duration* and above *venue*). Alice's identifier can be encrypted under DK and, as is done with *venue*, Alice can give AES(DK, *identifier*) to Bob so that he knows where in the tree to decrypt posts.
- This article has used Facebook and Twitter as examples of data that Alice might post, but we described the data as residing on the Blockchain itself. In more traditional environments (ie, not long-term–isolated environments), Alice would of course use the existing Facebook and Twitter systems for her posts. In such cases, Alice would store an encrypted URL (eg, link to the Twitter or Facebook post), rather than actual data, on the Blockchain. In this way, Alice could choose to post anonymously or pseudonymously on the real systems, and claim ownership of the posts on the Blockchain (revealing this claimed ownership to other participants, as desired).
- This article has described a very basic Blockchain (ie, a single instance of a hash chain of hash trees). In more traditional (nonisolated) environments, the Blockchain is likely to take the form of a distributed ledger: copies of the Blockchain stored all over the world (to give permanence and immutability), and a distributed consensus algorithm (proof of work or alternatives) to determine who can add the next block. We note that a full distributed ledger implementation of the Blockchain is completely compatible with our proposed scheme (although it is not appropriate for the isolated environments discussed in this article because of the confined space, limited number of computing systems, and sporadic communication with the outside world). Furthermore, even in the isolated environments that are the focus of this article, a constrained form of distributed ledger implementation is still possible. In particular, whenever Internet communication is available, the spaceship can replicate the current state of the Blockchain to all external nodes, and these nodes can use consensus algorithms that confirm the correctness of block format and syntax (and not just authorship through the digital signature) prior to the acceptance of any new block.
- An important consideration in some environments is the security of the Blockchain in the very long term. For such situations, there is a requirement for postquantum cryptographic algorithms (ie, algorithms that remain secure even when large-scale quantum computers are ubiquitous). For the algorithms discussed in this article, AES-256 will provide 128 bits of security in a postquantum world and SHA3-512 will still provide 256 bits of security, but ECDSA with the above parameters will provide roughly 30 bits of security ($O(|p|^3)$, where $|p|$ is the size of the prime modulus p, using Shor's algorithm[12]). Therefore, participants will need a postquantum digital signature algorithm with at least 128 bits of postquantum security to match AES-256. Schemes such as SPHINCS+,[13] XMSS,[14] or BPQS[15] may be appropriate for this level of security.
- A topic of interest in some environments is revocation: the ability for Alice to remove Bob's access to a post that she previously shared with him. This is not a topic addressed in this work, but we plan to address it in a forthcoming paper.

6 | CONCLUSION

This article proposes a privacy-preserving Blockchain with fine-grained access control for the storage of participant postings in long-term–isolated environments. The proposed system uses standard cryptographic algorithms and techniques, but with a novel key derivation algorithm that enables privacy and access control in a way that is extremely simple and efficient. Coupled with a fuzzy extractor component that reliably generates a master cryptographic key from a user's biometric, this system is very easy to use for all participants, including any with deteriorating mental wellness, which is an area of increasing concern in these types of environments.

The system described in Section 4 above has been designed for long-term–isolated environments with their particular constraints, but it would work equally well in more traditional environments. Thus, we have also discussed a number of possible extensions and enhancements that will increase its applicability in these other contexts. The proposal presented in this article allows a user to grant fine-grained access to his/her postings, but an open question is how to efficiently and securely achieve revocation in this scheme (ie, to remove access to posts that were previously shared). This is a direction for future work in this area that may be important to a number of real-world environments.

CONFLICT OF INTEREST

The author declares that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

ORCID

Carlisle Adams https://orcid.org/0000-0002-7335-9968

REFERENCES

1. NASA. Human Research Roadmap. https://humanresearchroadmap.nasa.gov/. Accessed April 1, 2019.
2. Feng, Q., D. He, S. Zeadally, M. K. Khan, and N. Kumar, “A survey on privacy protection in Blockchain system”, *Journal of Network and Computer Applications*, 126, 2019, pp. 45–59. https://www.sciencedirect.com/science/article/pii/S1084804518303485. Accessed April 1, 2019.
3. Henry, R., A. Herzberg, and A. Kate, “Blockchain access privacy: challenges and directions”, *IEEE Security & Privacy*, vol. 16, 2018, pp. 38–45. https://www.computer.org/csdl/magazine/sp/2018/04/msp2018040011/13rRUxBJhE9. Accessed April 1, 2019.
4. Rahulamathavan Y, Phan RC-W, Rajarajan M, Misra S. Privacy-preserving Blockchain based IoT ecosystem using attribute-based encryption. Paper presented at: IEEE International Conference on Advanced Networks and Telecommunications Systems; December 17-20, 2017. https://www.linkedin.com/pulse/how-achieve-fine-grained-privacy-blockchain-rahulamathavan-phd. Accessed April 1, 2019.
5. Buterin V. Privacy on the Blockchain. blog posted on January 15, 2016. https://blog.ethereum.org/2016/01/15/privacy-on-the-blockchain/. Accessed April 1, 2019.
6. Dukkipati, C., Y. Zhang, and L. C. Cheng. Decentralized, Blockchain based access control framework for the heterogeneous internet of things. Paper presented at: Proceedings of the Third ACM Workshop on Attribute Based Access Control; March 21, 2018: 61-69. https://dl.acm.org/citation.cfm?id=3180458. Accessed April 1, 2019.
7. Wang, S., Y. Zhang, and Y. Zhang, “A Blockchain-based framework for data sharing with fine-grained access control in decentralized storage systems”, *IEEE Access*, 6, 2018, pp. 38437–38450. https://ieeexplore.ieee.org/document/8400511. Accessed April 1, 2019.
8. Zhang Y, He D, Choo K-KR. BaDS: Blockchain-based architecture for data sharing with ABS and CP-ABE in IoT. *Wireless Communications and Mobile Computing*, 2018, 2019, pp. 1–9. https://www.hindawi.com/journals/wcmc/2018/2783658/. Accessed April 1, 2019.
9. Al Tamimi A-K. Performance analysis of data encryption algorithms; 2006. https://www.cse.wustl.edu/~jain/cse567-06/ftp/encryption_perf/. Accessed May 28, 2019.
10. Dodis, Y., L. Reyzin, and A. Smith, “Fuzzy extractors: a brief survey of results from 2004 to 2006”, in P. Tuyls, B. Skoric, and T. Kevenaar, *Security with Noisy Data*, Springer-Verlag, 2007. http://www.cs.bu.edu/~reyzin/papers/fuzzysurvey.pdf. Accessed April 1, 2019.
11. Dodis, Y., R. Ostrovsky, L. Reyzin, and A. Smith, “Fuzzy extractors: how to generate strong keys from biometrics and other Noisy data”, *SIAM Journal on Computing*, 38(1), 2008, pp. 97–139. http://web.cs.ucla.edu/~rafail/PUBLIC/89.pdf. Accessed April 1, 2019.
12. Shor P. Algorithms for quantum computation: discrete logarithms and factoring. Paper presented at: Proceedings of the 35th Annual Symposium on Foundations in Computer Science, November 20-22, 1994: 124-134. https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=365700. Accessed April 1, 2019.
13. Hulsing A, Aumasson J-P, Bernstein D, et al. SPHINCS+: a stateless, hash-based signature scheme. http://sphincs.org/. Accessed April 1, 2019.
14. Buchmann J, Dahmen E, Hulsing A. XMSS: a practical forward secure signature scheme based on minimal security assumptions. Paper presented at: Proceedings of PQCrypto, Springer LNCS 7071; 2011: 117-129. https://eprint.iacr.org/2011/484.pdf. Accessed April 1, 2019.

15. Chalkias K, Brown J, Hearn M, Lillehagen T, Nitto I, Schroeter T. Blockchained post-quantum signatures. Paper presented at: Proceedings of the IEEE International Conference on Blockchain (Cybermatics-2018), 2018: 1196–1203. https://eprint.iacr.org/2018/658.pdf. Accessed April 1, 2019.

How to cite this article: Adams C. A privacy-preserving Blockchain with fine-grained access control. *Security and Privacy*. 2019;e97. https://doi.org/10.1002/spy2.97