Efficient Local Secret Sharing for Distributed Blockchain Systems

Yongjune Kim, Ravi Kiran Raman, Young-Sik Kim, Lav R. Varshney, and Naresh R. Shanbhag

***Abstract*— Blockchain systems store transaction data in the form of a distributed ledger where each peer is to maintain an identical copy. Blockchain systems resemble repetition codes, incurring high storage cost. Recently, distributed storage blockchain (DSB) systems have been proposed to improve storage efficiency by incorporating secret sharing, private key encryption, and information dispersal algorithms. However, the DSB results in significant communication cost when peer failures occur due to denial of service attacks. In this letter, we propose a new DSB approach based on a local secret sharing (LSS) scheme with a hierarchical secret structure of one global secret and several local secrets. The proposed DSB approach with LSS improves the storage and recovery communication costs.**

***Index Terms*— Blockchain, secret sharing, distributed storage.**

I. Introduction

BLOCKCHAIN systems establish a cryptographically secure data structure to store transaction data in the form of a hash chain. Their distributed and shared ledgers of transactions reduce the friction in financial networks from different intermediaries using different technology infrastructures, and even reduce the need for intermediaries to validate financial transactions. Blockchain systems have created a new environment of business transactions and self-regulated cryptocurrencies [1]. However, blockchain works on the premise that every peer stores the entire ledger of transactions in the form of a hash chain, even though they are meaningless to peers that are not party to the transaction. Consequently, individual nodes incur a significant and ever-increasing storage cost [1]–[3].

To reduce this storage cost of blockchain systems, a distributed storage blockchain (DSB) scheme has been proposed [2], [3]. Inspired by [4], the DSB combines Shamir's secret sharing scheme [5], private key encryption, and information dispersal algorithm (IDA) [6]. The DSB reduces the storage to a fraction of the original blockchain's load.

A drawback of the DSB is that it incurs much higher *recovery communication cost* when peer failures occur due to denial of service (DoS) attacks. When a single peer failure occurs, the original blockchain systems can recover this failure by accessing any other peer because every peer has a copy of the ledger. On the other hand, a single peer failure results in effective data loss from a subset of peers as they loose their private encryption key, which incurs much more communication cost compared to traditional blockchain systems.

In this letter, we propose *local secret sharing (LSS)* and incorporate it into the DSB to improve the storage and communication costs. The proposed LSS has a hierarchical secrecy structure of one global secret and several local secrets. As in locally recoverable codes (LRCs) [7], each subset of peers can tolerate single peer failure. Hence, any single peer failure can be recovered locally, which reduces the recovery communication cost compared to the DSB. Further, the proposed LSS can also improve the storage cost of the DSB. In the original DSB, the private keys act as the local secrets for subsets of peers and the hashes are the global secrets. These local and global secrets are stored by using two independent secret sharing schemes. On the other hand, the LSS efficiently incorporates local secrets and global secrets into a hierarchical secret sharing scheme. Hence, the DSB with the proposed LSS can reduce the storage overhead for hash values and private keys *by half*. We show that this storage efficiency leads to lower recovery communication cost.

We characterize trade-offs between storage and communication costs of traditional blockchains, the original DSB, and the proposed DSB with LSS. These trade-offs explicitly show how the proposed approach improves the storage and recovery communication costs.

II. Background

A. Distributed Storage Blockchain (DSB)

Blockchain systems such as bitcoin store transaction data as a distributed ledger wherein each node in the network stores a current copy of the sequence of transactions (ledger) as a hash chain [1]. Let $B^{(t)}$ be the tth data block and $H^{(t)} = h_1(W^{(t)})$ be the hash value stored with the $(t+1)$th transaction, where $W^{(t)} = (H^{(t-1)}, h_2(B^{(t)}))$ is the concatenation of the previous hash value and a hash value of the current data block where $h_1(\cdot)$ and $h_2(\cdot)$ are two hash functions.

Every peer in traditional blockchain systems stores the entire ledger of transactions. Such data replication creates significant storage cost. Assuming that $B^{(t)} \in \mathbb{F}_\eta$ (a finite field of order η) and $W^{(t)} \in \mathbb{F}_q$, the blockchain's storage cost per transaction per peer is

$$\mathsf{S_B} = \log_2 \eta + \log_2 q. \tag{1}$$

In [2] and [3], DSB was proposed to reduce this storage cost. In the DSB, each transaction is encrypted using distinct private keys and distributed among subsets of peers. The private keys and hash values are stored by secret sharing scheme among the peers of the subset.

Suppose n peers are partitioned into L subsets of size $r+1$ (i.e., $L = \frac{n}{r+1}$). For a transaction at time t, the corresponding

Manuscript received September 16, 2018; revised October 22, 2018 and November 24, 2018; accepted November 28, 2018. Date of publication December 10, 2018; date of current version February 11, 2019. This work was supported in part by the National Research Foundation of Korea (NRF) grant funded by the Korea government (Ministry of Science and ICT) (No. NRF-2017R1A2B2010588). The associate editor coordinating the review of this paper and approving it for publication was M. Baldi. *(Corresponding author: Young-Sik Kim.)*

Y. Kim was with the Coordinated Science Laboratory, University of Illinois at Urbana–Champaign, Urbana, IL 61801 USA. He is now with Western Digital Research, Milpitas, CA 95035 USA (e-mail: yongjune.kim@wdc.com).

R. K. Raman, L. R. Varshney, and N. R. Shanbhag are with the Coordinated Science Laboratory, University of Illinois at Urbana–Champaign, Urbana, IL 61801 USA.

Y.-S. Kim is with the Department of Information and Communication Engineering, Chosun University, Gwangju 61452, South Korea.

Digital Object Identifier 10.1109/LCOMM.2018.2886016

data block $B^{(t)}$ and the hash value $W^{(t)} = (H^{(t-1)}, h_2(B^{(t)}))$ are stored according to Alg. 1. Note that Φ is an encryption scheme with a random private key $K_l^{(t)}$. In this scheme, the private key $K_l^{(t)}$ and the hash value $W^{(t)}$ are stored by using Shamir's $(r+1, r+1)$ secret sharing scheme.

Algorithm 1 Distributed Storage Blockchain in [2], [3]

Given partition $\mathcal{A} = \{A_1, \ldots, A_{\frac{n}{r+1}}\}$:

1: **for** $l = 1$ to $\frac{n}{r+1}$ **do**
2: Generate a random private key $K_l^{(t)} \in \mathbb{F}_q$.
3: Encrypt $B^{(t)}$ with $K_l^{(t)}$ as $\mathbf{m}_l^{(t)} = \Phi(B^{(t)}; K_l^{(t)})$.
4: Distribute and store $\mathbf{m}_l^{(t)}$ among $r+1$ peers in A_l.
5: Store $K_l^{(t)}$ and $W^{(t)}$ by $(r+1, r+1)$ secret sharing.
6: **end for**

If $\mathbf{m}_l^{(t)} \in \mathbb{F}_\eta^{r+1}$, $K_l^{(t)} \in \mathbb{F}_q$, and $W^{(t)} \in \mathbb{F}_q$, the DSB storage cost per transaction per peer is

$$\mathsf{S}_{\mathsf{DSB}} = \frac{\log_2 \eta}{r+1} + 2\log_2 q. \tag{2}$$

If the size of the private key space is much smaller than the size of data block (i.e., $q \ll \eta$), then the DSB reduces the storage cost significantly [2], [3].

B. Shamir's Secret Sharing

In Shamir's secret sharing scheme [5], a secret s is represented as n distinct values called *shares* such that the secret can be reconstructed only if one has at least k shares out of n shares. Any fewer shares reveal no information about s. It is called (n, k)-secret sharing. A dealer of secret sharing generates a secret s and a random encoding polynomial $f(x)$ of degree $k-1$ that has the secret s as its constant term: $f(x) = s + \sum_{i=1}^{k-1} a_i x^i$ where $s = a_0$ and a_is are the randomly chosen values among $\mathbb{F}_q$. Once $f(x)$ is generated, the dealer generates n points $(x_i, f(x_i))$ (for $i \in [0, n-1] := \{0, \ldots, n-1\}$) and distributes each share $f(x_i)$ to n peers. The secret s can be reconstructed by polynomial interpolation based on any k shares.

C. Locally Recoverable Codes

An (n, k, r) LRC is a code of length n with information (message) length k, minimum distance d, and recovery locality r. If a symbol in the LRC-coded data is lost due to a single peer failure, its value can be recovered by accessing only r other symbols [7]. The relation between d and r is given by $d \le n - k - \lceil \frac{k}{r} \rceil + 2$.

In our LSS, we exploit the LRC construction of [7]. Define a *good polynomial* $g(x)$ satisfying the following conditions:

C1. $\deg(g(x)) = r + 1$.
C2. There exists a partition $\mathcal{A} = \{A_1, \ldots, A_{\frac{n}{r+1}}\}$ of a set $A \subseteq \mathbb{F}_q$ ($|A| = n$ and $|A_l| = r+1$) such that $g(x) = c_l$ for any $x \in A_l$, $l \in [1, \frac{n}{r+1}]$.

Assume that k and n are divisible by r and $r+1$, respectively. These restrictions can be readily lifted [7].

For a message vector $\mathbf{a} = (a_{i,j}) \in \mathbb{F}_q^k$ where $i \in [0, r-1]$ and $j \in [0, \frac{k}{r} - 1]$. The encoding polynomial is

$$f_{\mathbf{a}}(x) = \sum_{i=0}^{r-1} f_i(x) x^i \tag{3}$$

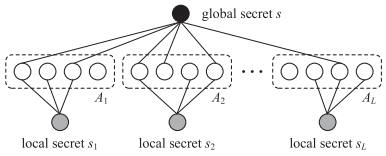

Fig. 1. The LSS scheme where $k' = 6$ and $r = 3$ ($r < k'$). The global secret s can be reconstructed by accessing any k' peers and a local secret s_l can be reconstructed by accessing r peers in the corresponding subset A_l.

where $f_i(x) = \sum_{j=0}^{\frac{k}{r}-1} a_{i,j} g(x)^j$. The codeword for the message $\mathbf{a}$ is obtained by n distinct evaluations of $f_{\mathbf{a}}(\cdot)$, i.e.,

$$\mathcal{C} = \{(f_{\mathbf{a}}(\alpha)) : \alpha \in A\}. \tag{4}$$

It was shown that $\mathcal{C}$ is an optimal (n, k, r) LRC [7].

Suppose $c_\alpha = f_{\mathbf{a}}(\alpha)$ for $\alpha \in A_l$ is lost. Define the *decoding polynomial* as $\delta(x) = \sum_{i=0}^{r-1} f_i(\alpha) x^i$. Since each $f_i(x)$ is a linear combination of powers of $g(x)$, $f_i(x)$ is constant on the set A_l (i.e., $f_i(\beta) = f_i(\alpha)$ for any $\beta \in A_l$). Hence, $\delta(\beta) = \sum_{i=0}^{r-1} f_i(\alpha)\beta^i = \sum_{i=0}^{r-1} f_i(\beta)\beta^i = f_{\mathbf{a}}(\beta)$ which means that the values of the encoding polynomial $f_{\mathbf{a}}(x)$ and the decoding polynomial $\delta(x)$ on the locations of A_l coincide. Since $\deg(\delta(x)) \le r-1$, $\delta(x)$ can be interpolated from r other symbols in A_l.

III. Local Secret Sharing

We propose an LSS scheme, which shares a global secret across all peers and the local secrets among peers in subsets as shown in Fig. 1. The more important secret is set as the global secret, which can be accessed by any k' peers and the less important secrets are reconstructed by r peers of the corresponding subset.

A naive LSS approach is to use two classes of secret sharing schemes. The global secret is shared by an (n, k') secret sharing scheme and the local secret s_l for a subset A_l is distributed by an $(r+1, r)$ secret sharing scheme where $|A_l| = r+1$. In this naive approach, each peer stores two shares: $f(x_i)$ for the global secret s and $f^{(l)}(x_i)$ for local secret s_l where $f^{(l)}(x)$ denotes an encoding polynomial for a subset A_l. Hence, the total number of shares is $2n$.

The proposed LSS scheme is described in Alg. 2. In the proposed LSS scheme, each peer stores only one share $f_{\mathbf{a}}(\alpha)$ for $\alpha \in A$ in (3) thereby having n shares instead of $2n$. As Shamir's secret sharing is based on the maximum distance separable (MDS) codes, the proposed LSS scheme is implemented by using LRCs.

The following lemma shows global secret s can be reconstructed by accessing any $k + \frac{k}{r} - 1$ peers, which is greater than the k peers that Shamir's (n, k) secret sharing code would require. If $r = k$, the LSS reduces to Shamir's scheme.

Lemma 1: The global secret $s = a_{0,0}$ can be reconstructed by accessing any $k' = k + \frac{k}{r} - 1$ peers.

Proof: Since $\deg(f_{\mathbf{a}}(x)) \le k + \frac{k}{r} - 2$, $f_{\mathbf{a}}(x)$ can be interpolated by accessing any $k + \frac{k}{r} - 1$ peers. ■

The local secret s_l for A_l can be reconstructed by accessing any r peers in A_l. Since $r < k$, the local secrets are not as secure as the global secret.

Theorem 2: The local secrets can be reconstructed by accessing any r peers in the corresponding subsets.

Algorithm 2 The Proposed Local Secret Sharing (LSS) Scheme

Given a good polynomial $g(x)$ and $\mathcal{A} = \{A_1, \ldots, A_{\frac{n}{r+1}}\}$:

1: Generate a random vector $\mathbf{a} = (a_{i,j}) \in \mathbb{F}_q^k$ and set $s = a_{0,0}$ where $i \in [0, r-1]$ and $j \in [0, \frac{k}{r} - 1]$.
2: **for** $l = 1$ to $\frac{n}{r+1}$ **do**
3: Set local secret $s_l = f_0(\beta) = \sum_{j=0}^{\frac{k}{r}-1} a_{0,j} g(\beta)^j$ for $\beta \in A_l$.
4: **end for**
5: Compute $(f_{\mathbf{a}}(\alpha) : \alpha \in A)$ by (4) and distribute among n peers.

Proof: Since $g(\alpha) = u_l$ for any $\alpha \in A_l$, $f_i(\alpha) = f_i(\beta) = v_i^{(l)}$ for any $\alpha, \beta \in A_l$. The decoding polynomial for the subset A_l is $\delta_l(x) = \sum_{i=0}^{r-1} f_i(\alpha) x^i = \sum_{i=0}^{r-1} v_i^{(l)} x^i$ where $\alpha \in A_l$ and $|A_l| = r+1$. The decoding polynomial $\delta_l(x)$ can be regarded as an encoding polynomial of an $(r+1, r)$ code. If we define $s_l = v_0^{(l)} = f_0(\alpha)$ as a local secret for A_l, each subset's LSS is equivalent to Shamir's $(r+1, r)$ scheme. ∎

Theorem 3: If $k > r$, then r peers of a subset cannot reconstruct the local secrets of any other subset.

Proof: For two subsets A_l and A_m, $v_0^{(l)}$ and $v_0^{(m)}$ denote local secrets of A_l and A_m, respectively. Then, $v_0^{(l)} = \sum_{j=0}^{\frac{k}{r}-1} a_{0,j} u_l^j$ and $v_0^{(m)} = \sum_{j=0}^{\frac{k}{r}-1} a_{0,j} u_m^j$ where $u_l = g(\alpha)$ for $\alpha \in A_l$ and $u_m = g(\beta)$ for $\beta \in A_m$. Suppose the $(r+1, r)$ code corresponding to A_l has been decoded by accessing r peers in A_l. Then, $(v_0^{(l)}, \ldots, v_{r-1}^{(l)})$ are known to the peers in A_l. We can obtain r equations for k unknowns $(a_{0,0}, \ldots, a_{r-1, \frac{k}{r}-1})$. Because $k > r$, r peers of A_l cannot solve this linear system. ∎

Theorem 4: The global secret can be reconstructed by local secrets from $\frac{k}{r}$ subsets.

Proof: Suppose we know the local secrets of m subsets. Then, we can construct the following linear system:

$$\begin{bmatrix} 1 & u_1 & \ldots & u_1^{\frac{k}{r}-1} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & u_m & \ldots & u_m^{\frac{k}{r}-1} \end{bmatrix} \begin{bmatrix} a_{0,0} \\ \vdots \\ a_{0,\frac{k}{r}-1} \end{bmatrix} = \begin{bmatrix} s_1 \\ \vdots \\ s_m \end{bmatrix}. \tag{5}$$

If $u_i \neq u_j$, then we guarantee the global secret $s = a_{0,0}$ can be reconstructed for $m \geq \frac{k}{r}$ because the matrix in (5) is Vandermonde. Now we show $u_i \neq u_j$ for $i \neq j$ (i.e., $g(\alpha) \neq g(\beta)$ for $\alpha \in A_i$ and $\beta \in A_j$). As explained in Sec. II-C, $\deg(g(x)) = r+1$. Hence, $g(x)$ can have no more than $r+1$ roots. Suppose that $\eta = g(\alpha) = g(\beta)$ for $\alpha \in A_i$ and $\beta \in A_j$. We define $\widetilde{g}(x) = g(x) - \eta$ where $\deg(\widetilde{g}(x)) = r+1$. Because of condition C2 on $g(x)$ (see Section II-C), $\widetilde{g}(x)$ should have $2(r+1)$ roots, which contradicts $\deg(\widetilde{g}(x)) = r+1$. ∎

Theorem 5: The global secret can be reconstructed by accessing k peers if we access peers from $\frac{k}{r}$ subsets and access r peers per subset. The global secret cannot be reconstructed by accessing $k-1$ or fewer peers.

Proof: Accessing r peers in each subset is equivalent to accessing $r+1$ peers due to locality property. Then, the equivalent total number of accessed peers is $(r+1) \cdot \frac{k}{r} = k + \frac{k}{r}$, which is greater than $k + \frac{k}{r} - 1$ in Lemma 1 and the global secret can be reconstructed. Since the dimension of $\mathcal{C}$ is k, the global secret cannot be constructed from less than k peers. ∎

Corollary 6: For a given global secret, the dimension of local secrets of the proposed LSS is $\dim(\mathsf{LS}) = \frac{k}{r} - 1$.

Proof: Since the matrix of (5) is Vandermonde, the dimension of local secrets is the same as the dimension of $(a_{0,0}, \ldots, a_{0,\frac{k}{r}-1})$ for $m = \frac{k}{r}$. If a global secret $s = a_{0,0}$ is given, then $\dim(\mathsf{LS}) = \frac{k}{r} - 1$. ∎

IV. Distributed Storage Blockchains With LSS

Although the original DSB improves the storage cost significantly, a single peer failure leads to an effective failure of $r+1$ peers in the DSB. The reason is that the private key and hash value of the corresponding subset cannot be retrieved. To recover this subset failure, the DSB should access $r+1$ peers in another subset, which incurs considerable communication cost. Note that the traditional blockchain systems can recover a single peer failure by accessing any other peer because every peer stores the entire ledger. Further, a DoS attacker can damage the total ledger by attacking $\frac{n}{r+1}$ peers (i.e., single peer per subset) whereas a traditional blockchain can tolerate up to $n-1$ peer failures.

We incorporate our LSS into the original DSB to reduce storage/communication costs and enhance robustness.

Algorithm 3 Distributed Storage Blockchain With LSS

Given partition $\mathcal{A} = \{A_1, \ldots, A_{\frac{n}{r+1}}\}$:

1: Set hash value $W^{(t)}$ as global secret $s^{(t)} = a_{0,0} \in \mathbb{F}_q$.
2: Generate a random vector $\mathbf{a}_{\setminus a_{0,0}}^{(t)} \in \mathbb{F}_q^{k-1}$.
3: Compute local secrets $s_l^{(t)}$ for $l \in [1, \frac{n}{r+1}]$ and $(f_{\mathbf{a}}^{(t)}(\alpha_i))$ for $i \in [1, n]$ by Alg. 2.
4: Distribute and store $(f_{\mathbf{a}}^{(t)}(\alpha_i))$ into n peers.
5: **for** $l = 1$ to $\frac{n}{r+1}$ **do**
6: Encrypt $B^{(t)}$ with $s_l^{(t)}$ as $\mathbf{m}_l^{(t)} = \Phi(B^{(t)}; s_l^{(t)})$.
7: Encode $\mathbf{m}_l^{(t)}$ into $\mathbf{c}_l^{(t)}$ by $(r+1, r)$ coding.
8: Distribute and store $\mathbf{c}_l^{(t)}$ among peers in A_l.
9: **end for**

We point out that the private keys are *local* to the subsets whereas the hash values are *common* to all peers in Alg. 1. We set private keys as local secrets and hash values as global secrets as in Alg. 3. As shown in Sec. III, the LSS reduces the storage cost for private keys and hash values *by half*.

In the proposed LSS, each subset can tolerate a single peer failure as in LRCs. Hence, we encode the encrypted data block $\mathbf{m}_l^{(t)}$ by $(r+1, r)$ MDS codes. This coding reduces the communication cost to recover single peer failure and improves robustness to DoS attacks. Next, we quantify the storage/communication costs and robustness to DoS.

A. Storage Cost

The storage costs of traditional blockchain and the DSB are given in (1) and (2), respectively. We quantify the storage cost for the DSB with proposed LSS. First, $(r+1, r)$ coding for the encrypted data block increases the storage cost for the

TABLE I
COMPARISON OF STORAGE/COMMUNICATION COSTS AND ROBUSTNESS

	Traditional Blockchain	Original DSB (Algorithm 1)	DSB with LSS (Algorithm 3)
S	$\log_2 \eta + \log_2 q$	$\frac{\log_2 \eta}{r+1} + 2\log_2 q$	$\frac{\log_2 \eta}{r} + \log_2 q$
$\mathsf{C}^{(\mathsf{r})}$	$\log_2 \eta + \log_2 q$	$\log_2 \eta + 2(r+1)\log_2 q + \rho$	$\log_2 \eta + r \log_2 q$
$v^{\dagger}$	$n-1$	$\frac{n}{r+1} - 1$	$2 \cdot \frac{n}{r+1} - 1$

† Recovery of transaction is guaranteed up to any v peers failures.

data block from $\frac{\log_2 \eta}{r+1}$ to $\frac{\log_2 \eta}{r}$. Since hash values and private keys are efficiently stored according to the LSS (Alg. 2), the storage cost of the DSB with LSS is

$$\mathsf{S}_{\mathsf{LSS}} = \frac{\log_2 \eta}{r} + \log_2 q. \tag{6}$$

B. Communication Costs

There are two kinds of communication cost: Cost to store *new* transactions (transaction communication cost) and cost to recover peer failures (recovery communication cost). Since each peer should receive the amount of S for a new transaction, the transaction communication cost per peer $\mathsf{C}^{(\mathsf{n})}$ is proportional to the storage cost per peer per transaction S (i.e., $\mathsf{C}^{(\mathsf{n})} \propto \mathsf{S}$).

The more interesting cost is the recovery communication cost $\mathsf{C}^{(\mathsf{r})}$. A peer under DoS attacks cannot respond to a request and its data is unavailable. This failure can be modeled as an erasure channel [8]. As distributed storage codes such as regenerating codes [9] focus on the communication cost to recover a single node failure, we aim to reduce the communication cost to recover a single peer failure, which is the most common scenario. Since every peer stores the entire ledger of transactions in traditional blockchains, the single peer failure can be recovered by receiving the stored ledger of any other peers. Hence, the recovery communication cost of traditional blockchain is

$$\mathsf{C}_{\mathsf{B}}^{(\mathsf{r})} \propto \log_2 \eta + \log_2 q \tag{7}$$

which is the smallest possible, as repetition codes incur the least communication cost in distributed storage systems.

In the original DSB [2], [3], a single peer failure disables the corresponding subset due to private key loss. In order to recover this subset, we should access $r+1$ peers of another subset. The recovery communication cost is

$$\begin{aligned} \mathsf{C}_{\mathsf{DSB}}^{(\mathsf{r})} &\propto \{(r+1)\mathsf{S}_{\mathsf{DSB}} + \rho\} \\ &= \log_2 \eta + 2(r+1)\log_2 q + \rho \end{aligned} \tag{8}$$

where ρ denotes the additional cost to access another subset.

In the LSS, the data block of each subset can be recovered locally by accessing r peers in the same subset. Hence, the recovery communication cost is given by

$$\mathsf{C}_{\mathsf{LSS}}^{(\mathsf{r})} \propto r\mathsf{S}_{\mathsf{LSS}} = \log_2 \eta + r \log_2 q. \tag{9}$$

Note that the proposed LSS can reduce the recovery communication cost as well as the storage cost.

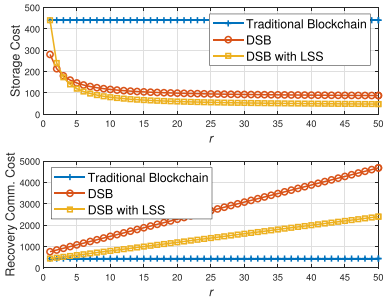

Fig. 2. Tradeoff between storage cost and recovery communication cost ($\eta = 2^{400}, q = 2^{40}, \rho = 200$).

C. Robustness to Peer Failures

Traditional blockchains tolerate $n-1$ peer failures. On the other hand, only $L = \frac{n}{r+1}$ peer failures can cause data loss in the DSB if one peer from every subset undergoes failure [3]. Both the DSBs with the proposed LSS enhance the robustness since each subset can recover a single peer failure. Hence, the DSBs with LSS schemes guarantee transaction data recovery up to $\frac{2n}{r+1} - 1$ failures. Note that the transaction data cannot be recovered if every subset suffers from two peers failures.

Table I and Fig. 2 compare the storage/communication costs and robustness to peer failures. Traditional blockchains show the best recovery communication cost and the worst storage cost like repetition codes in distributed storage systems. The proposed LSS improves the storage cost by incorporating private key values and hash values with a coding-theoretic approach. Furthermore, the LSS reduces the recovery communication cost. Note that the slope of the DSB with LSS is 1 whereas the slope of the DSB is 2 in Fig. 2.

V. CONCLUSION

In this letter, we have proposed a new DSB scheme based on LSS. The proposed scheme improves the storage and recovery communication costs. The extensions of the LSS to more general frameworks would be interesting future topics.

REFERENCES

[1] K. Croman *et al.*, "On scaling decentralized blockchains," in *Proc. Int. Conf. Financial Cryptogr. Data Secur.*, Aug. 2016, pp. 106–125.
[2] R. K. Raman and L. R. Varshney, "Distributed storage meets secret sharing on the blockchain," in *Proc. Inf. Theory Appl. Workshop (ITA)*, Feb. 2018.
[3] R. K. Raman and L. R. Varshney, "Dynamic distributed storage for blockchains," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jun. 2018, pp. 2619–2623. [Online]. Available: https://arxiv.org/pdf/1711.07617.pdf
[4] H. Krawczyk, "Secret sharing made short," in *Proc. Annu. Int. Cryptol. Conf.*, Jan. 1994, pp. 136–146.
[5] A. Shamir, "How to share a secret," *Commun. ACM*, vol. 22, no. 11, pp. 612–613, Nov. 1979.
[6] M. O. Rabin, "Efficient dispersal of information for security, load balancing, and fault tolerance," *J. ACM*, vol. 36, no. 2, pp. 335–348, Apr. 1989.
[7] I. Tamo and A. Barg, "A family of optimal locally recoverable codes," *IEEE Trans. Inf. Theory*, vol. 60, no. 8, pp. 4661–4676, Aug. 2014.
[8] A. Pannetrat and R. Molva, "Efficient multicast packet authentication," in *Proc. Netw. Distrib. Syst. Security Symp. (NDSS)*, Feb. 2003.
[9] A. G. Dimakis, P. B. Godfrey, Y. Wu, M. J. Wainwright, and K. Ramchandran, "Network coding for distributed storage systems," *IEEE Trans. Inf. Theory*, vol. 56, no. 9, pp. 4539–4551, Sep. 2010.