Secret Share Dissemination across a Network

Nihar B. Shah, K. V. Rashmi and Kannan Ramchandran
Dept. of Electrical Engineering and Computer Sciences
University of California, Berkeley
{nihar, rashmikv, kannanr}@eecs.berkeley.edu.

Abstract

Secret sharing is an important component of several cryptographic protocols. These include protocols for secure multiparty function computation, key management, and secure archival storage. Most protocols assume that the dealer has direct communication links with every participant, in which case, the dealer can directly communicate the respective shares to all participants.

In this paper, we consider the problem of disseminating shares of a secret when the dealer and the participants form a general network. We provide an algorithm for secret share dissemination that is communication-efficient, distributed and deterministic. Interestingly, the solution constitutes an instance of a network coding problem admitting a distributed and deterministic solution, and furthermore, handles the case of nodal-eavesdropping, about which very little appears to be known in the literature.

I. Introduction

Shamir's classical (k, n) secret sharing scheme [1] is an essential ingredient of several cryptographic protocols. The scheme considers a set of $(n + 1)$ entities: a *dealer* and n *participants*. The dealer possesses a secret s and wishes to pass functions (called *shares*) of this secret to the n participants, such that the following properties are satisfied:

- k*-secret-recovery*: the shares of any k participants suffice to recover the secret
- $(k-1)$*-collusion-resistance*: the aggregate data gathered by any $(k-1)$ nodes reveals no knowledge (in the information-theoretic sense) about the secret.

Several cryptographic protocols in the literature require execution of one or more instances of secret share dissemination. These include protocols for secure multiparty computation [2]–[7], secure key management [8], [9], general Byzantine agreement between all participants [2], [10]–[12], proactive secret sharing [13], [14], and secure archival storage [15].

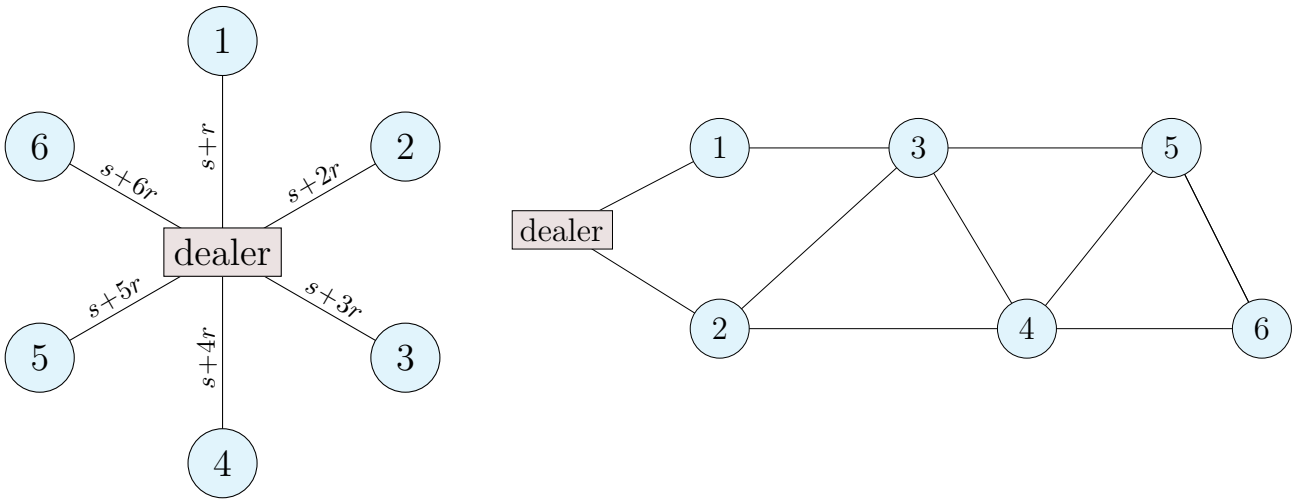
Most protocols including those listed above assume that the dealer has direct communication links to every participant. In this case, the dealer can compute the shares as per Shamir's scheme [1] and directly pass the shares to the respective participants. This setting is depicted in Fig. 1a for the parameters $k = 2$ and $n = 6$.

In several situations, the dealer may not have direct communication links with every participant; instead, the dealer and the participants may form a more general network. Fig. 1b depicts such a scenario. More formally, consider a graph G with $(n + 1)$ vertices. These $(n + 1)$ vertices comprise the dealer and the n participants. An edge in this graph implies a secure communication channel between its two end-points, while the absence of an edge denotes the non-existence of any direct communication channel.

Under a general network G, all communication between the dealer and a participant who is not directly connected to it, must pass through other participants in the network. This poses the challenge of designing protocols where the dealer can disseminate shares to all participants without leaking any additional information to any participant.

A solution that is typically employed in the literature is to execute a pairwise agreement protocol, once separately for each participant. Under such a solution, in order to communicate the designated share to any participant,

Preprint also available at http://arxiv.org/abs/1207.0120 (June 2012).



(a) Dealer connected to all participants

(b) Dealer and participants forming a general network

Fig. 1: Shamir's secret sharing scheme for $k = 2$ and $n = 6$ participants. The share of participant i $(1 \leq i \leq 6)$ is $s + ir$, where s is the secret and r is a value chosen uniformly at random from the finite field of operation. (a) All participants are connected directly to the dealer, allowing the dealer to directly pass the shares. (b) The dealer and the participants form a general network, where the dealer cannot pass shares directly to participants 3, 4, 5 and 6.

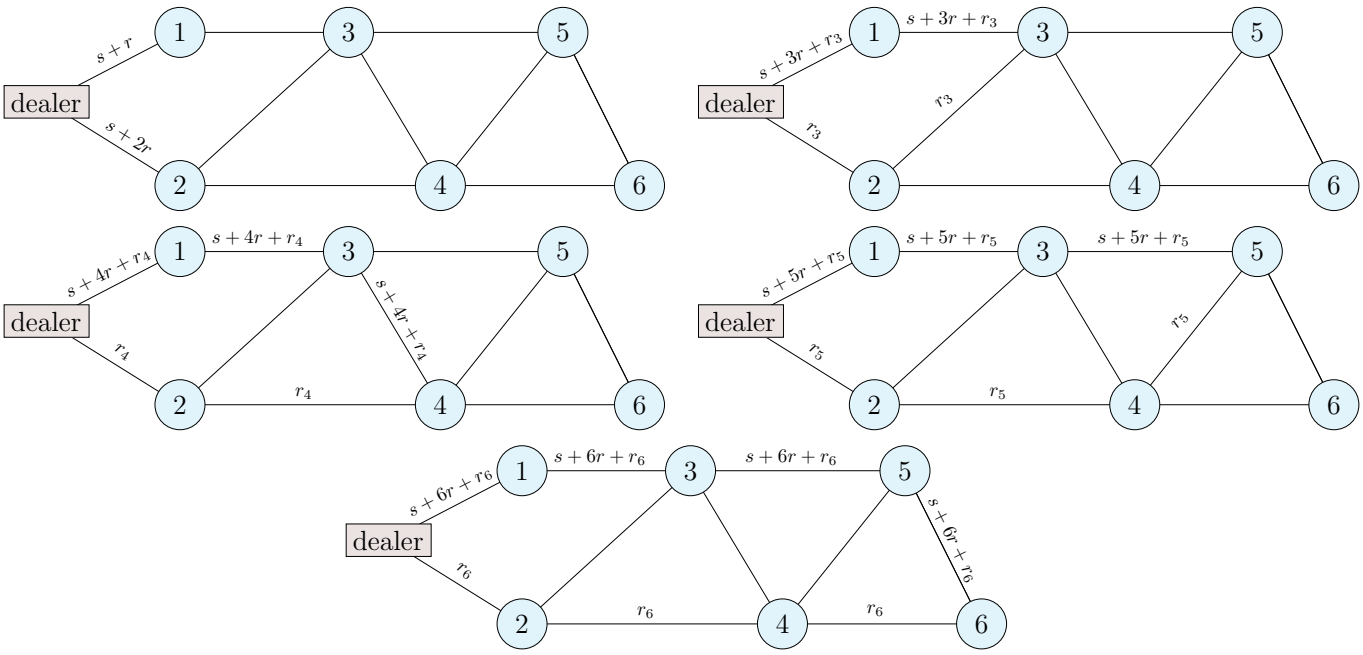
the dealer treats this share as a secret, and employs Shamir's scheme to compute k shares of this secret. The dealer then communicates these k shares to the participant through k vertex disjoint paths (alternatively, the dealer may employ a more general protocol for Byzantine agreement [2], [16], [17] for the communication with each participant). However, such a solution incurs a high communication cost, since the dealer needs to execute the pairwise agreement protocol separately for every participant. Moreover, the requirement of setting up k disjoint paths to every participant requires significant coordination in the network.

We now provide an illustration of a scheme employing pairwise agreement protocols, accompanied by an illustration of our algorithm, on the network of Fig. 1b.

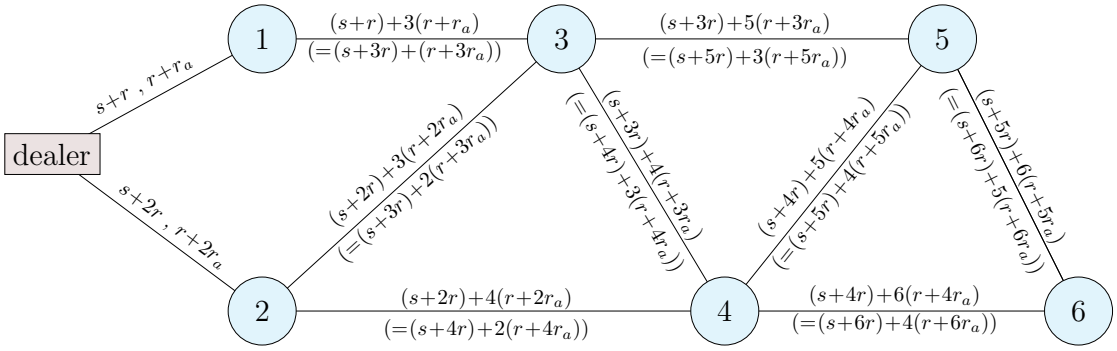
Example 1: Consider the network in Fig. 1b. Let $n = 6$ and $k = 2$, with the alphabet of operation as a finite field $\mathbb{F}_q$ of size $q > 6$. Under Shamir's scheme of encoding the secret s, the share t_i $(1 \leq i \leq 6)$ for participant i is $t_i = s + ir$, where r is a value chosen by the dealer uniformly at random from the alphabet. While the dealer can directly pass the shares t_1 and t_2 to participants 1 and 2 respectively, the difficulty arises in communicating shares to the remaining participants with whom the dealer does not have direct communication links. For instance, if the dealer tries to pass share t_3 to participant 3 by simply communicating t_3 along the path 'dealer $\rightarrow 1 \rightarrow$ 3', then participant 1 gains access to two shares, t_1 and t_3. Using these two shares, participant 1 can unilaterally recover the secret s, thus violating the $(k-1)$-collusion resistance requirement.

A typical method employed in the literature, to overcome this issue, is to use pairwise agreement protocols, as depicted in Fig. 2a. To pass share t_3 to participant 3, the dealer chooses another random value r_3, and passes $(t_3 + r_3)$ along the path 'dealer $\rightarrow 1 \rightarrow$ 3' and r_3 along the path 'dealer $\rightarrow 2 \rightarrow$ 3'. Now, participant 3 can recover its share t_3, and no participant gains any additional information about the secret s in this process. In a similar manner, the dealer can communicate t_i $(4 \leq i \leq 6)$ to participant i by passing $(t_i + r_i)$ and r_i through $k = 2$ vertex disjoint paths. Although this solution guarantees successful share dissemination, it is communication inefficient, and requires considerable coordination in the network to set up the disjoint paths.

Observe that the protocol described above has several random values $\{r_i\}_{i=3}^{6}$ that are transmitted across several hops in the network in a particular step, but which are never used subsequently in the protocol. Thus, in order to design efficient algorithms, one may wish to propagate random values in a manner that allows their subsequent reuse. Fig. 2b depicts our algorithm for secret share dissemination, which requires a communication of only 12 values over the links, as opposed to 24 in the previous algorithm. Furthermore, this algorithm requires the generation



(a) Using a pairwise agreement protocol



(b) Algorithm of this paper

Fig. 2: Two algorithms for secret share dissemination across the network of Fig. 1b, for $n = 6$ and $k = 2$: (a) existing algorithm using a pairwise agreement protocol, and (b) new algorithm proposed in this paper. The text on an edge is the data passed by the node on the left end-point of the edge to the node at the right end-point. The values of $\{r, r_3, r_4, r_5, r_6, r_a\}$ are chosen uniformly and independently at random from $\mathbb{F}_q$. Under both algorithms, each participant i $(1 \leq i \leq 6)$ successfully receives the share $(s + ir)$. The algorithm in (a) requires 24 units of communication, as compared to only 12 under the algorithm in (b).

of only 2 random values, as compared to 5 previously.

■

In this paper, we consider the problem of efficient dissemination of the shares of a secret to participants forming a general communication network. We provide an algorithm that *concurrently* disseminates the shares to all participants, for a wide class of networks. This entails a much lower communication cost. Moreover, the algorithm is completely distributed: the actions of each node are independent of the network topology, and every node needs to know only the identities of its one-hop neighbours. As a result, this algorithm is also robust to any run-time changes in the network topology (e.g., removal or addition of new links). Furthermore, the algorithm has a polynomial time computation complexity. The algorithm can also be extended to perform verification of shares to combat a cheating dealer or actively adversarial participants, two-threshold secret sharing, and addition of new

participants in the absence of the dealer.

The problem of secret share dissemination can also be cast as a specific instance of a *network coding* problem [18], [19], as described later in the paper. The literature on secure network coding largely considers models where a bounded number of links in the network may be compromised to an eavesdropper. However, the problem at hand transforms into a problem where a bounded number of nodes in the network are compromised, about which very little appears to be known in the literature. Thus, the solution presented in this paper turns out to be an instance of a network coding problem that admits a distributed, deterministic and communication-efficient solution.

As an interesting intellectual connection, the algorithm presented in this paper is based on a variant of the *Product-Matrix codes*, which were originally constructed in [20], [21] for distributed storage networks. These codes possess interesting properties, which the algorithm exploits.

The remainder of the paper is organized as follows. Section II provides a formal description of the system model and states the results of this paper. Section III reviews Shamir's secret sharing scheme and the related literature. Section IV describes the algorithm in full generality. Section V presents conclusions. Several properties, extensions and additional applications of the algorithm are discussed in Appendix A.

II. System Model

A. The Secret-share Dissemination Problem

The dealer possesses a secret s that is drawn from some alphabet $\mathcal{A}$, and wishes to pass shares of this secret to n partiipants. The dealer and the participants form a communication network, denoted by graph G. The graph G has $(n+1)$ vertices comprising the dealer and the n participants[1], and an edge in the graph denotes a private communication link between the two end-points. The problem is to design a protocol which will allow the dealer to pass shares (of the secret) to the n participants, meeting the requirements of $(k-1)$-*collusion-resistance* and k-*secret-recovery* (described in Section I). All the participants are assumed to be honest-but-curious, i.e., they follow the protocol correctly, but may store any accessible data to gain information about the secret (the case of active adversaries is considered in Appendix A). The edges in the graph G may be directed or undirected: a directed edge implies existence of only a one way communication link, while an undirected edge implies direct communication links both ways.

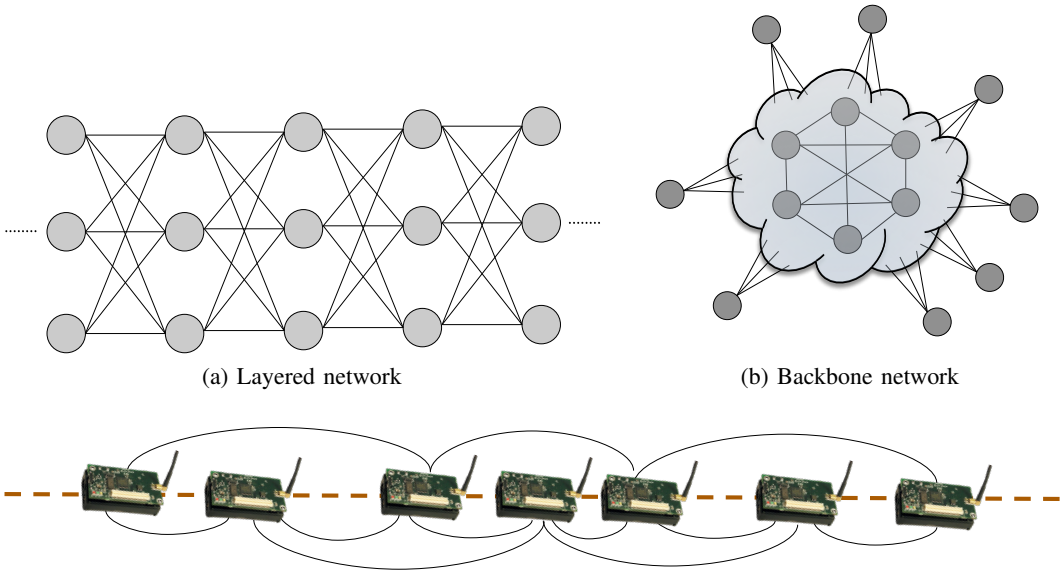
The following is a condition on graph G for *any* secret share dissemination algorithm to successfully perform secret share dissemination.

Condition 1 (k-connected-dealer): Each of the n participants in the graph is either directly connected to the dealer or has at-least k vertex-disjoint paths between the dealer and itself.

Proof: The proof of the necessity of this condition is straightforward. If there is a vertex (say, vertex i) that violates the k-connected dealer condition, then there exists a set $\mathcal{V}_{k-1}$ of other $(k-1)$ vertices such that all paths from the dealer to vertex i necessarily pass through at least one of the vertices in $\mathcal{V}_{k-1}$. Thus the entire share of participant i can be reconstructed by the participants in $\mathcal{V}_{k-1}$. It follows that a collusion of the $(k-1)$ nodes $\mathcal{V}_{k-1}$ can put together their own $(k-1)$ shares along with the share of node i and recover s, thus violating the $(k-1)$ collusion resistance property. ■

Thus no algorithm can operate successfully on all network topologies, and must at least require the graph G to obey the k-connected-dealer condition. Moreover, as typical of many such problems, an algorithm constructed for this problem may require the network topology to satisfy certain additional structural assumptions. However, in practice, the structure of the network graph may not be known beforehand. Moreover, under a dynamic network, the graph structure may also vary with time. This leads to a natural question about the outcome of an algorithm over a network that does not meet the conditions required by the algorithm. Since the security of the data is paramount,

[1]Thus, at times, we will also refer to a participant as a vertex or a node of the graph.



(a) Layered network

(b) Backbone network

(c) One-dimensional geometric network (say, deployment of sensors on the border)

Fig. 3: Examples of networks satisfying the 3-propagating-dealer condition (any node in the network may be the dealer).

it is desirable that the algorithm continues to satisfy the $(k-1)$-collusion-resistance property irrespective of the network topology. We formalize this notion in terms of the following additional requirement.

> *Robustness to network topology*: Consider any algorithm designed to work on a class of graphs $\mathcal{G}$, and let G be the actual realization of the communication graph. If $G \in \mathcal{G}$ then the algorithm must accomplish secret share dissemination, and if $G \notin \mathcal{G}$ then running the algorithm on the network G should leak no information about the secret.

The problem considered here is to construct efficient algorithms for secret share dissemination that satisfy the three conditions of (i) k-secret-recovery, (ii) $(k-1)$-collusion-resistance, and (iii) robustness to the network topology. The algorithm presented in this paper meets these conditions for a wide class of networks. The class of networks on which our algorithms can operate successfully are described below.

B. Class of Networks Considered

The algorithm presented in this paper requires the communication network G to satisfy the following condition.

Condition 2 (k-propagating-dealer): There exists an ordering of the n participants in the graph such that every vertex is either directly connected to the dealer, or to some k nodes preceding it in the ordering.

We note that while the algorithm constructed in this paper requires the *existence* of some such ordering, the execution of the algorithm is completely distributed and *oblivious* to the actual ordering.

As an illustration of this condition, consider the network of Example 1 (Fig. 1b). This network satisfies the 2-propagating-dealer condition, with the ordering $1, 2, 3, 4, 5, 6$; observe that this is also the order in which the participants receive their shares under our algorithm (Fig. 2b). Fig. 3 depicts three examples of graphs that satisfy the 3-propagating dealer condition. These examples can be generalized to the following classes of graphs:

(a) Layered networks, with each layer containing at-least k nodes, and each node connected to all nodes in the neighbouring layers. An ordering that satisfies the k-propagating-dealer condition is the ordering of the nodes with respect to the distance (in terms of number of hops) from the dealer.

(b) Networks with a fully-connected 'backbone' component, where a node outside the backbone is connected directly to at-least k nodes in the backbone. An ordering that satisfies the k-propagating-dealer condition is: neighbours of the dealer, followed by all remaining nodes in the backbone, followed by all remaining nodes not in the backbone.

(c) k-connected one-dimensional geometric networks. A one-dimensional geometric network is formed by arranging the nodes (in an arbitrary fashion) along a line, and connecting a pair of nodes by an edge if the distance between is smaller than a fixed threshold. A one-dimensional geometric network that satisfies the k-connected-dealer condition also satisfies the k-propagating dealer condition. An ordering that satisfies the k-propagating-dealer condition is the arrangement of the nodes in an ascending order of their euclidean distance from the dealer.

In addition, any directed acyclic graph (DAG) that satisfies the k-connected-dealer condition automatically satisfies the k-propagating-dealer condition. Any topological ordering of the DAG will satisfy the k-propagating-dealer condition. Moreover, given any graph G, the condition of k-propagating-dealer can be verified in a computationally efficient manner; this is described later in Section IV-C

Apart from the parameters n and k, an additional parameter d is associated to our algorithm. We saw earlier that the k-connected-dealer condition is necessary for *any* secret share dissemination algorithm, and the algorithm presented in this paper requires the k-propagating-dealer condition to be satisfied. Now, assuming that these necessary conditions have been met, one would intuitively expect the efficiency of the algorithm to increase with the connectivity of the graph. The parameter d is used to capture this intuition: our algorithm takes the parameter d ($\geq k$) as input, and under the assumption that the graph satisfies the *d-propagating-dealer* condition, achieves a greater communication efficiency.

Note that the algorithm is robust to the network topology, and hence will not leak any information in the event of the network not satisfying the specified condition.

C. Precise Statement of the Result

This paper presents a communication-efficient, distributed and deterministic algorithm that takes parameters n, k and d ($\geq k$) as input, and enables a dealer to pass shares of a secret to the n participants, such that the properties of

- k-secret-recovery, when the network satisfies d-propagating-dealer condition
- $(k-1)$-collusion-resistance
- robustness to network topology (i.e., no information leaked if graph does not satisfy required conditions)

are satisfied. The communication-efficiency of the algorithm increases with the value of d.

D. Notational Conventions

Throughout the paper we follow standard convention of denoting vectors in boldface and matrices by upper-case alphabets. A vector will be treated as a column vector by default, and a row vector will be written as the transpose of the corresponding column vector. Transpose of a vector or matrix will be denoted by a superscript T. For any integer $\ell \geq 1$, $[\ell]$ will represent the set $\{1, \ldots, \ell\}$.

III. Related Literature

A. Shamir's Secret Sharing Protocol

We first give a brief review of Shamir's secret sharing protocol [1]. We assume for now that the dealer has a direct (secure) communication link with every participant (as in Fig. 1a).

Assume that the secret s is drawn from some finite field $\mathbb{F}_q$ of size q $(> n)$. The dealer node $(k-1)$ values $\{r_i\}_{i=1}^{k-1}$ uniformly and independently at random from $\mathbb{F}_q$. Define a k-length vector $\boldsymbol{m}$ as[2]

$$\boldsymbol{m}^T = [s \ r_1 \ r_2 \ \cdots \ r_{k-1}] \ . \tag{1}$$

Next, define a set of n vectors $\{\boldsymbol{\psi}_i\}_{i=1}^{n}$, each of length k, as

$$\boldsymbol{\psi}_i^T = [1 \ i \ i^2 \ \cdots \ i^{k-1}] \ . \tag{2}$$

The share t_i of participant i is simply the inner product

$$t_i = \boldsymbol{\psi}_i^T \boldsymbol{m} \ . \tag{3}$$

It can be verified that for any set $\mathcal{I} \subseteq [n]$ of cardinality k, the secret s can be recovered from the set of values $\{\boldsymbol{\psi}_i^T \boldsymbol{m}\}_{i\in\mathcal{I}}$. Furthermore, it can also be verified that for any set $\mathcal{I}' \subseteq [n]$ of cardinality smaller than k, the set $\{\boldsymbol{\psi}_i^T \boldsymbol{m}\}_{i\in\mathcal{I}'}$ provides no knowledge about s.

Under the assumption that the dealer has direct communication links with each of the n participants, the dealer can simply pass t_i to participant i. This completes the description of Shamir's secret sharing protocol.

We now describe some protocols in the literature that address the situation when the dealer may *not* have direct communication links with all participants. Under each of these protocols, we discuss only the conditions of $(k-1)$-collusion-resistance and k-secret-recovery (and not the condition of robustness to network structure).

B. Pairwise Agreement Protocols

This section describes a protocol for secret share dissemination based on pairwise agreement protocols [2], [16], [17]. Fig. 2a in Example 1 is an example of such a protocol. Under this protocol, the dealer first encodes the secret s into n shares $\{t_\ell\}_{\ell=1}^{n}$ using Shamir's secret sharing scheme (3). To every node ℓ directly connected to the dealer, the dealer directly passes its share t_ℓ. For each remaining node, the dealer executes the following protocol of 'pairwise agreement', once separately for each remaining node. Let ℓ now denote a node that is not connected directly to the dealer. The dealer applies Shamir's secret sharing scheme treating t_ℓ as a secret, and computes k shares $\{u_{\ell,j}\}_{j=1}^{k}$, as

$$u_{\ell,j} = [1 \ j \ j^2 \ \cdots \ j^{k-1}] \begin{bmatrix} t_j \\ r_{j,1} \\ r_{j,2} \\ \vdots \\ r_{j,k-1} \end{bmatrix} \ , \tag{4}$$

where the values $\{r_{j,1}, \ldots, r_{j,k-1}\}$ are chosen independently and uniformly at random from $\mathbb{F}_q$. The dealer then finds k vertex-disjoint paths (from itself) to node ℓ, and passes $u_{\ell,j}$ along the j^{th} path $(1 \leq j \leq k)$. At the end of this pairwise agreement protocol, node ℓ receives $\{u_{\ell,j}\}_{j=1}^{k}$ from which it can recover its share t_ℓ. Moreover, since each of the random values are independent, no participant can obtain any information about any other participant's share, or any additional information about the secret s. This process is repeated once for every node that is not connected directly to the dealer.

The protocol described above requires transmission of data across k-vertex disjoint paths once for *every node* that is not connected directly to the dealer. Thus this protocol is not very efficient in terms of communication complexity, and furthermore, is not distributed.

[2]To suit the description of the algorithm developed subsequently in this paper, we deviate from the customary polynomial based description of Shamir's protocol, and employ a matrix-based notation instead.

C. Network Coding

A multicast network coding problem [18] considers transmission of data across a network which is represented by a graph. Each edge in the graph denotes a communication link between its two end points. One of the nodes in this graph is the 'source', where data to be transmitted (called the'message') is generated. One or more nodes in the networks are the 'sinks', and it is required that the entire message be recovered by each of the sinks. The remaining nodes in the network only act as intermediate nodes, and aid in the communication. The data transmitted by a node along an edge incident on it can be an arbitrary function of the data it previously received.

The problem considered in this paper can be modelled as a multicast network coding problem in the following manner. The dealer is the source node, and the secret s is the message. The network graph in the network coding problem is identical to that in the secret sharing problem, but with a set of $\binom{n}{k}$ additional nodes that act as the sinks. Each of the $\binom{n}{k}$ sinks is connected to some set of k participants, with one edge to each of these k participants. This corresponds to the condition of k-secret-recovery. To satisfy the $(k-1)$-collusion-resistance property, no set of $(k-1)$ nodes (excluding the source and sinks) should be able to obtain any information about the message. This is equivalent to a network coding problem requiring secrecy from an eavesdropper that can gain access to a subset of the nodes. However, with respect to this setting, very little appears to be known in the network coding literature.

To the best of our knowledge, the literature on secure network-coding (e.g., [19], [22]–[24]) considers only the setting where the eavesdropper gains access to a subset of the *links*. The problem of node-compromise is treated as a case of link-compromise by allowing the eavesdropper to gain access to all links that are incident upon the compromised nodes. In [19], [24], authors consider the setting wherein a collection of subsets of the links is specified, and an eavesdropper may gain access to precisely one of these subsets. However, the scheme provided is not explicit, requires the size of the finite field to be exponential in n. The algorithm depends on the knowledge of the network topology, and given the network topology, it is computationally difficult to obtain the precise actions to be performed at the nodes under this algorithm. Moreover, the scheme requires the graph to satisfy a particular condition, which is almost always violated in our problem setting. On the other hand, communication-efficient algorithms to secure a network from an eavesdropper having access to a *bounded* number of links are provided in [22], [23]. Given the network topology, the actions to be performed at the nodes can be derived in a computationally efficient manner. However, these algorithms communicate a message of size equal to the difference between the largest message that can be sent in the absence of secrecy requirements, and the bound on number of compromised links. Under our problem setting, this difference will generally be zero or smaller (e.g., the difference is -2 in the network of Fig. 1b), thus rendering these algorithms inapplicable.

The algorithms currently found in the network coding literature, even for the setting where there are no secrecy requirements, are either random (thus not guaranteed) [25], or deterministic but centralized [26]. Thus, the results of this paper present a case where an instance of a network coding problem admits a distributed and deterministic solution.

IV. Algorithm for Secret Share Dissemination

This section presents the main result of the paper. Consider a network G that obeys the d-propagating-dealer condition (Condition 2) for some parameter d $(\geq k)$. The secret s belongs to the alphabet $\mathcal{A}$, and we assume that $\mathcal{A} = \mathbb{F}_q^{d-k+1}$, for some $q > n$. Thus we can equivalently denote the secret as a vector $\mathbf{s}^T = [s_1 \ s_2 \ \cdots \ s_{d-k+1}]$ with each element of this vector belonging to the finite field $\mathbb{F}_q$.

A. Initial Setting up by the Dealer

The dealer first constructs an $(n \times d)$ Vandermonde matrix Ψ, with the i^{th} $(1 \leq i \leq n)$ row of Ψ being

$$\boldsymbol{\psi}_i^T = [1 \ i \ i^2 \ \cdots \ i^{d-1}] \ . \tag{5}$$

The vector $\boldsymbol{\psi}_i^T$ is termed the *encoding vector* of node i.

Next, the dealer constructs a $(d \times d)$ *symmetric* matrix M comprising the secret $\mathbf{s}$ and a collection of randomly generated values as follows:[3]

$$M = \underbrace{\begin{bmatrix} s_A & \boldsymbol{r_a}^T & \boldsymbol{s_B}^T \\ \boldsymbol{r_a} & R_b & R_c^T \\ \boldsymbol{s_B} & R_c & 0 \end{bmatrix}}_{\underbrace{1}_{} \quad \underbrace{k-1}_{} \quad \underbrace{d-k}_{}} \Bigg\}_{d} \tag{6}$$

where the depicted sub-matrices of M are

- $s_A = s_{d-k+1}$ is a scalar,
- $\boldsymbol{s_B} = [s_1 \cdots s_{d-k}]^T$ is a vector of length $(d-k)$,
- $\boldsymbol{r_a}$ is a $((k-1) \times 1)$ is a vector of length $(k-1)$,
- R_b is a $((k-1) \times (k-1))$ *symmetric* matrix with its $\frac{k(k-1)}{2}$ entries populated by random values,
- R_c is a $((d-k) \times (k-1))$ matrix with its $(d-k)(k-1)$ entries populated by random values.

These random values are all picked independently and uniformly from $\mathbb{F}_q$. Note that the total number of random values in M is

$$\begin{aligned} R &= (k-1) + \frac{k(k-1)}{2} + (k-1)(d-k) \\ &= (k-1)d - \binom{k-1}{2} . \end{aligned} \tag{7}$$

The entire secret is contained in the components s_A and $\boldsymbol{s_B}$ as $\mathbf{s} = [s_1 \cdots s_{d-k+1}] = [\boldsymbol{s_B}^T \quad s_A]$.

Observe that the structure of M as described in (6), along with the symmetry of matrix R_b, makes the matrix M *symmetric*.

The share $\mathbf{t}_j$ for participant j $(1 \leq j \leq n)$ is a vector of length $(d-k+1)$:

$$\boldsymbol{t}_j^T = \boldsymbol{\psi}_j^T \begin{bmatrix} s_A & \boldsymbol{s_B}^T \\ \boldsymbol{r_a} & R_c^T \\ \boldsymbol{s_B} & 0 \end{bmatrix} . \tag{8}$$

We shall show subsequently in Theorem 2 that any k of these shares suffice to recover the entire secret.

Remark 1: To see these shares in the conventional polynomial representation of Shamir's secret sharing scheme, recall that the vector $\boldsymbol{\psi}_j^T$ is drawn from a Vandermonde matrix. Thus each column of $\boldsymbol{t}_j$ in (8) can be seen as the evaluation of a polynomial at value j. Thus there is one polynomial for each secret value s_i $(1 \leq i \leq d-k+1)$, having the corresponding secret symbol as its constant term and the remaining coefficients picked randomly.

B. Communication across the Network

For any participant j $(1 \leq j \leq n)$, denote the set of its neighbours by $\mathcal{N}(j)$. Denote the set of neighbours of the dealer as $\mathcal{N}(\text{dealer})$. Algorithm 1 describes the communication protocol to securely transmit the shares $\{\boldsymbol{t}_j\}_{j=1}^n$ to the n participants.

[3]The reader familiar with the literature on regenerating codes for distributed storage may notice that we employ the MBR version (and not the MSR version) of the product-matrix codes [20]. We make this choice to guarantee secrecy from honest-but-curious participants, who may store all the data that they receive, a characteristic of the MBR point on the storage-bandwidth tradeoff [27].

Algorithm 1 Communication Protocol

Dealer: For every $j \in \mathcal{N}(\text{dealer})$, compute and pass the d-length vector $\boldsymbol{\psi}_j^T M$ to participant j.

Participant $\boldsymbol{\ell \in \mathcal{N}(\text{dealer})}$: Wait until receipt of data $\boldsymbol{\psi}_\ell^T M$ from the dealer. Upon receipt, perform the following actions. For every $j \in \mathcal{N}(\ell)$, compute inner product of the data $\boldsymbol{\psi}_\ell^T M$ with the encoding vector $\boldsymbol{\psi}_j$ of participant j. Transmit the resultant value $\boldsymbol{\psi}_\ell^T M \boldsymbol{\psi}_j$ to participant j.

Participant $\boldsymbol{\ell \notin \mathcal{N}(\text{dealer})}$: Wait until receipt of one value each from any d neighbours, and then perform the following actions (if more than d neighbours pass data, retain data from some arbitrary d of these nodes). Denote this set of d neighbours as $\{i_1, \ldots, i_d\} \subseteq \mathcal{N}(\ell)$, and the values received from them as $\{\sigma_1, \ldots, \sigma_d\}$ respectively. Compute the vector

$$\boldsymbol{v} = \begin{bmatrix} & \boldsymbol{\psi}_{i_1}^T & \\ & \vdots & \\ & \boldsymbol{\psi}_{i_d}^T & \end{bmatrix}^{-1} \begin{bmatrix} \sigma_1 \\ \vdots \\ \sigma_d \end{bmatrix}.$$

For every neighbour $i \in \mathcal{N}(\ell)$ from whom you did not receive data, compute and pass the inner product $\boldsymbol{v}^T \boldsymbol{\psi}_i$ to participant i.

Remark 2: Under this algorithm, unnecessary communication may occur when (i) more than d participants simultaneously attempt to transmit data to a participant who is not directly connected to the dealer, or when (ii) neighbouring participants that are also connected directly to the dealer attempt to transmit data to each other. This can be avoided by employing a simple handshaking protocol between neighbours: a participant who is ready to transmit data to its neighbour, requests the neighbour for an approval of this transmission, prior to actually sending the data.

C. Correctness of the Protocol

The following theorems show that each participant indeed receives its intended share (8), and the algorithm satisfies the properties of k-secret-recovery, $(k-1)$-collusion-resistance and robustness to network structure. The communication and computational efficiency of the algorithm are discussed in Appendix A.

Theorem 1 (Successful share dissemination): Under the algorithm presented, every participant $\ell \in [n]$ can recover $\boldsymbol{\psi}_\ell^T M$, and hence obtain its intended share

$$\boldsymbol{t}_\ell^T = \boldsymbol{\psi}_\ell^T \begin{bmatrix} s_A & \boldsymbol{s_B}^T \\ \boldsymbol{r_a} & R_c^T \\ \boldsymbol{s_B} & 0 \end{bmatrix}.$$

.

Proof: Recall that the graph satisfies the d-dealer propagation condition. Let us assume without loss of generality that that the ordering of vertices satisfying this condition is $1, \ldots, n$. It follows that the first d vertices in this ordering must be connected directly to the dealer.

The proof proceeds via induction. The induction hypothesis is as follows: every participant ℓ can recover the data $\boldsymbol{\psi}_\ell^T M$, and if ℓ passes any data to any other node $j \in \mathcal{N}(\ell)$ then this data is precisely the value $\boldsymbol{\psi}_\ell^T M \boldsymbol{\psi}_j$. Consider the base case of node 1. Since this node is connected directly to the dealer, it receives the data $\boldsymbol{\psi}_1^T M$ from the dealer. Moreover, following the communication protocol, it passes $\boldsymbol{\psi}_1^T M \boldsymbol{\psi}_j$ to its neighbours $j \in \mathcal{N}(1)$. Let us now assume that the hypothesis holds true for the first $(\ell-1)$ nodes in the ordering. If node ℓ is connected directly to the dealer, then the hypothesis is satisfied for this node by an argument identical to the case of node 1. Suppose ℓ is not connected to the dealer. It follows that node ℓ must be connected to at least d other nodes preceding it in the ordering, and furthermore, must receive data from at least d of these nodes (say, nodes $\{j_1, \ldots, j_d\} \subseteq [\ell-1]$).

By our hypothesis, these d nodes pass the d values $\{\boldsymbol{\psi}_{j_1}^T M\boldsymbol{\psi}_\ell, \ldots, \boldsymbol{\psi}_{j_d}^T M\boldsymbol{\psi}_\ell\}$. It follows that the algorithm running at node ℓ operates on the input

$$\begin{bmatrix} \sigma_1 \\ \vdots \\ \sigma_d \end{bmatrix} = \begin{bmatrix} \boldsymbol{\psi}_{j_1}^T \\ \vdots \\ \boldsymbol{\psi}_{j_d}^T \end{bmatrix} M\boldsymbol{\psi}_\ell \ . \tag{9}$$

By construction, the matrix in (9) comprising $\{\boldsymbol{\psi}_{j_1}^T, \ldots, \boldsymbol{\psi}_{j_d}^T\}$ as its rows is a $(d \times d)$ Vandermonde matrix, and hence is invertible. Thus, computing $\boldsymbol{v}$ in the algorithm can be performed efficiently using standard Reed-Solomon decoding algorithms [28], [29]. Thus, we have $\boldsymbol{v} = M\boldsymbol{\psi}_\ell$, and since M is a symmetric matrix, we get $\boldsymbol{v}^T = \boldsymbol{\psi}_\ell^T M^T = \boldsymbol{\psi}_\ell^T M$. Finally, the data passed by node ℓ to any other node $i \in \mathcal{N}(\ell)$, according to the protocol, is $v^T\boldsymbol{\psi}_i = \boldsymbol{\psi}_\ell^T M\boldsymbol{\psi}_i$. This proves the hypothesis for node ℓ.

Due to the specific structure (6) of M, the desired share $\boldsymbol{t}_\ell$ is a subset of the elements of the vector $\boldsymbol{\psi}_\ell^T M$. Thus, every participant obtains its intended share. ■

Theorem 2 (k-secret-recovery): Any k shares suffice to recover the secret.

Proof: Let $\mathcal{I} \subseteq [n]$ denote the set of the k participants attempting to recover the secret. Let $\Psi_\mathcal{I}$ be a $(k \times d)$ matrix with its k rows comprising $\{\boldsymbol{\psi}_i^T\}_{i\in\mathcal{I}}$. Further, let $\tilde{\Psi}_\mathcal{I}$ denote the $(k \times k)$ submatrix of $\Psi_\mathcal{I}$ comprising the first k columns of $\Psi_\mathcal{I}$. In terms of this notation, these k participants collectively have access to the data

$$\Psi_\mathcal{I} \begin{bmatrix} s_A & \boldsymbol{s_B}^T \\ \boldsymbol{r_a} & R_c \\ \boldsymbol{s_B} & 0 \end{bmatrix}.$$

Consider the last k columns of this data, i.e.,

$$\Psi_\mathcal{I} \begin{bmatrix} \boldsymbol{s_B}^T \\ R_c^T \\ 0 \end{bmatrix} = \tilde{\Psi}_\mathcal{I} \begin{bmatrix} \boldsymbol{s_B}^T \\ R_c^T \end{bmatrix}.$$

Since $\Psi_\mathcal{I}$ is a $(k \times d)$ Vandermonde matrix, $\tilde{\Psi}_\mathcal{I}$ is $(k \times k)$ Vandermonde matrix. Thus, $\tilde{\Psi}_\mathcal{I}$ is invertible. This allows for the decoding of $\boldsymbol{s_B}$ (via an algorithm [28], [29] identical to decoding under Shamir's classical secret sharing scheme). It remains to recover s_A and to this end consider the first column of the data, i.e.,

$$\Psi_\mathcal{I} \begin{bmatrix} s_A \\ \boldsymbol{r_a} \\ \boldsymbol{s_B} \end{bmatrix}.$$

Since the value of $\boldsymbol{s_B}$ is now known, its effect can be subtracted from this data to obtain

$$\Psi_\mathcal{I} \begin{bmatrix} s_A \\ \boldsymbol{r_a} \\ 0 \end{bmatrix} = \tilde{\Psi}_\mathcal{I} \begin{bmatrix} s_A \\ \boldsymbol{r_a} \end{bmatrix}.$$

Since $\tilde{\Psi}_\mathcal{I}$ is invertible, the value of s_A can now be decoded from this data. ■

Theorem 3 ($(k-1)$-collusion-resistance): Any set of $(k-1)$ or fewer colluding participants can gain no information about the secret.

Proof: The proof of this theorem is provided in Appendix B. ■

Corollary 4 (Robustness to the network topology): It follows from the proof of Theorem 3 that the $(k-1)$-collusion-resistance property holds irrespective of the network topology. This, along with the results of Theorem 1 and Theorem 2, implies the property of *robustness to the network topology*.

This completes the verification of the correctness of our algorithm.

Remark 3: In certain scenarios, the communication network may be known beforehand, and it may be desired to verify whether it satisfies the d-propagating-dealer condition. This task can be performed efficiently by simply simulating communication protocol of Algorithm 1 on this network: the d-propagating dealer condition is satisfied if and only if all nodes successfully obtain their shares.

V. Conclusion

This paper presents an algorithm to disseminate shares of a secret in a setting where the dealer and the participants may form a general network. The algorithm is communication-efficient, distributed, and deterministic (guaranteed). The algorithm successfully disseminates the shares if the network satisfies the d-propagating-dealer condition, and does not leak any information otherwise. The result of this paper is an instance of a network coding problem admitting a deterministic and distributed solution. Moreover, it handles the case of nodal-eavesdropping in network coding, about which very little appears to be known in the literature.

A distinctive feature of the algorithm is that it is both distributed and deterministic. A future goal is to construct distributed and deterministic algorithms for broader classes of networks, for secret share dissemination and other communication problems (perhaps by further exploiting the special structure of the product-matrix framework [20]).

Acknowledgement

The authors would like to thank Prakash Ishwar and Matthieu Finiasz for helpful discussions.

References

[1] A. Shamir, "How to share a secret," *Communications of the ACM*, vol. 22, no. 11, pp. 612–613, 1979.
[2] M. Ben-Or, S. Goldwasser, and A. Wigderson, "Completeness theorems for non-cryptographic fault-tolerant distributed computation," in *Proceedings of the twentieth annual ACM symposium on Theory of computing*, 1988, pp. 1–10.
[3] D. Chaum, C. Crépeau, and I. Damgard, "Multiparty unconditionally secure protocols," in *Proceedings of the twentieth annual ACM symposium on Theory of computing*, 1988, pp. 11–19.
[4] T. Rabin and M. Ben-Or, "Verifiable secret sharing and multiparty protocols with honest majority," in *Proceedings of the twenty-first annual ACM symposium on Theory of computing*, 1989, pp. 73–85.
[5] R. Cramer, M. Franklin, B. Schoenmakers, and M. Yung, "Multi-authority secret-ballot elections with linear work," in *Advances in Cryptology–EUROCRYPT*, 1996, pp. 72–83.
[6] M. Hirt, U. Maurer, and B. Przydatek, "Efficient secure multi-party computation," *Advances in Cryptology–ASIACRYPT*, pp. 143–161, 2000.
[7] I. Damgård, Y. Ishai, and M. Krøigaard, "Perfectly secure multiparty computation and the computational overhead of cryptography," *Advances in Cryptology–EUROCRYPT*, pp. 445–465, 2010.
[8] T. Pedersen, "A threshold cryptosystem without a trusted party," in *Advances in Cryptology–EUROCRYPT*, 1991, pp. 522–526.
[9] M. Marsh and F. Schneider, "Codex: A robust and secure secret distribution system," *Dependable and Secure Computing, IEEE Transactions on*, vol. 1, no. 1, pp. 34–47, 2004.
[10] M. Rabin, "Randomized Byzantine generals," in *Foundations of Computer Science, 1983., 24th Annual Symposium on*, 1983, pp. 403–409.
[11] P. Feldman and S. Micali, "Optimal algorithms for Byzantine agreement," in *Proceedings of the twentieth annual ACM symposium on Theory of computing*. ACM, 1988, pp. 148–161.
[12] I. Ingemarsson and G. Simmons, "A protocol to set up shared secret schemes without the assistance of a mutually trusted party," in *Advances in Cryptology–EUROCRYPT*, 1990, pp. 266–282.
[13] R. Ostrovsky and M. Yung, "How to withstand mobile virus attacks," in *Proceedings of the tenth annual ACM symposium on Principles of distributed computing*, 1991, pp. 51–59.
[14] A. Herzberg, S. Jarecki, H. Krawczyk, and M. Yung, "Proactive secret sharing or: How to cope with perpetual leakage," *Advances in Cryptology–CRYPT0*, pp. 339–352, 1995.
[15] M. Storer, K. Greenan, E. Miller, and K. Voruganti, "Potshardsa secure, recoverable, long-term archival storage system," *ACM Transactions on Storage (TOS)*, vol. 5, no. 2, p. 5, 2009.
[16] D. Dolev, C. Dwork, O. Waarts, and M. Yung, "Perfectly secure message transmission," *Journal of the ACM*, vol. 40, no. 1, pp. 17–47, 1993.
[17] Y. Desmedt and Y. Wang, "Perfectly secure message transmission revisited," in *Advances in Cryptology – EUROCRYPT*, 2002, pp. 502–517.
[18] R. Ahlswede, N. Cai, S. Li, and R. Yeung, "Network information flow," *Information Theory, IEEE Transactions on*, vol. 46, no. 4, pp. 1204–1216, 2000.

[19] N. Cai and R. Yeung, "Secure network coding on a wiretap network," *Information Theory, IEEE Transactions on*, vol. 57, no. 1, pp. 424–435, 2011.
[20] K. V. Rashmi, N. B. Shah, and P. V. Kumar, "Optimal exact-regenerating codes for the MSR and MBR points via a product-matrix construction," *IEEE Transactions on Information Theory*, vol. 57, no. 8, pp. 5227–5239, Aug. 2011.
[21] N. B. Shah, K. V. Rashmi, and P. V. Kumar, "Information-theoretically secure regenerating codes for distributed storage," in *Proc. Globecom*, Houston, Dec. 2011.
[22] H. Yao, D. Silva, S. Jaggi, and M. Langberg, "Network codes resilient to jamming and eavesdropping," in *Network Coding (NetCod), 2010 IEEE International Symposium on*, 2010, pp. 1–6.
[23] C. Ngai and R. Yeung, "Secure error-correcting (sec) network codes," in *Network Coding, Theory, and Applications, 2009. NetCod'09. Workshop on*, 2009, pp. 98–103.
[24] J. Feldman, T. Malkin, C. Stein, and R. Servedio, "On the capacity of secure network coding," in *Proc. 42nd Annual Allerton Conference on Communication, Control, and Computing*, 2004.
[25] T. Ho, M. Médard, R. Koetter, D. Karger, M. Effros, J. Shi, and B. Leong, "A random linear network coding approach to multicast," *Information Theory, IEEE Transactions on*, vol. 52, no. 10, pp. 4413–4430, 2006.
[26] S. Jaggi, P. Sanders, P. Chou, M. Effros, S. Egner, K. Jain, and L. Tolhuizen, "Polynomial time algorithms for multicast network code construction," *Information Theory, IEEE Transactions on*, vol. 51, no. 6, pp. 1973–1982, 2005.
[27] A. G. Dimakis, P. B. Godfrey, Y. Wu, M. Wainwright, and K. Ramchandran, "Network coding for distributed storage systems," *IEEE Transactions on Information Theory*, vol. 56, no. 9, pp. 4539–4551, 2010.
[28] R. Blahut, "Theory and practice of error control codes," 1983.
[29] I. Gohberg and V. Olshevsky, "Fast algorithms with preprocessing for matrix-vector multiplication problems," *Journal of Complexity*, vol. 10, no. 4, pp. 411–427, 1994.
[30] J. Douceur, "The sybil attack," *Peer-to-peer Systems*, pp. 251–260, 2002.
[31] M. Franklin and M. Yung, "Communication complexity of secure computation," in *Proceedings of the twenty-fourth annual ACM symposium on Theory of computing*, 1992, pp. 699–710.
[32] K. V. Rashmi, N. B. Shah, P. V. Kumar, and K. Ramchandran, "Explicit construction of optimal exact regenerating codes for distributed storage," in *Proc. 47th Annual Allerton Conference on Communication, Control, and Computing*, Urbana-Champaign, Sep. 2009, pp. 1243–1249.
[33] N. B. Shah, K. V. Rashmi, P. V. Kumar, and K. Ramchandran, "Interference alignment in regenerating codes for distributed storage: Necessity and code constructions," *IEEE Transactions on Information Theory*, vol. 58, no. 4, pp. 2134–2158, Apr. 2012.
[34] K. V. Rashmi, N. B. Shah, K. Ramchandran, and P. Kumar, "Regenerating codes for errors and erasures in distributed storage," in *Proc. International Symposium on Information Theory (to appear)*, Jul. 2012.
[35] N. B. Shah, K. V. Rashmi, P. V. Kumar, and K. Ramchandran, "Distributed storage codes with repair-by-transfer and non-achievability of interior points on the storage-bandwidth tradeoff," *IEEE Transactions on Information Theory*, vol. 58, no. 3, pp. 1837–1852, Mar. 2012.
[36] S. Pawar, S. El Rouayheb, and K. Ramchandran, "Securing dynamic distributed storage systems against eavesdropping and adversarial attacks," *IEEE Transactions on Information Theory*, vol. 57, no. 10, pp. 6734–6753, 2011.
[37] V. Cadambe and S. Jafar, "Interference alignment and spatial degrees of freedom for the k user interference channel," *IEEE Transactions on Information Theory*, vol. 54, no. 8, pp. 3425–3441, Aug. 2008.
[38] M. Maddah-Ali, A. Motahari, and A. Khandani, "Communication over MIMO X channels: Interference alignment, decomposition, and performance analysis," *IEEE Transactions on Information Theory*, vol. 54, no. 8, pp. 3457–3470, Aug. 2008.
[39] N. Saxena, G. Tsudik, and J. Yi, "Efficient node admission for short-lived mobile ad hoc networks," in *13th IEEE International Conference on Network Protocols*, 2005.
[40] R. Cramer, I. Damgård, and U. Maurer, "General secure multi-party computation from any linear secret-sharing scheme," in *Advances in Cryptology–EUROCRYPT*, 2000, pp. 316–334.

APPENDIX A
PROPERTIES AND EXTENSIONS OF THE ALGORITHM

A. Communication and Computational Efficiency

We measure the communication efficiency of the algorithm in terms of the amount of data downloaded by each node (normalized with respect to the size of the secret). Recall that the size of the secret is $(d-k+1)$ values over $\mathbb{F}_q$. First, consider a participant ℓ $(1 \leq \ell \leq n)$ that is not directly connected to the dealer. Under the algorithm, participant ℓ downloads one value each from d other participants. To satisfy the properties of k-secret-recovery and $(k-1)$-collusion-resistance, participant ℓ must have $(d-k+1)$ values in addition to what it obtains from any set of $(k-1)$ other participants. Thus this is the minimum possible download required to recover the share by connecting to d other participants. On the other hand, participants connected directly to the dealer download d values from it. The minimum download required is $(d-k+1)$ in this case; the additional downloaded values aid in transmitting shares to the participants that are not directly connected to the dealer.

The algorithm is computationally efficient since every participant is required to perform no more than one decoding and one encoding of a Reed-Solomon code, for which several efficient algorithms are known [28], [29].

B. Addition of New Participants in Absence of Trusted Entities

In several applications of interest, one may be faced with a scenario where the dealer exits the system after distributing the shares to the n participants, and a new participant is to be added to the system in the absence of the dealer or any other trusted entity. The new share is to be created and passed to the new participant such that the properties of $(k-1)$-collusion-resistance and k-secret-recovery continue to hold. Furthermore, the information obtained by the new participant must be precisely what it would have obtained if the dealer was present in the system.

We first describe a naive means to accomplish this task. Suppose the system currently has n participants, and it is desired to add a new participant $(n+1)$. Observe that since all the data initially available to the dealer can be recovered from any k shares, and the share $\boldsymbol{t}_{n+1}$ of the new participant is a function of this data. Thus, one can treat the share $\boldsymbol{t}_{n+1}$ as a function of these k shares, and employ a secure multiparty computation protocol (e.g., the BGW protocol of [2]) to provide $\boldsymbol{t}_{n+1}$ to participant $(n+1)$. However, this method requires a considerable amount of communication and coordination among the existing participants.

We now present an efficient method to adding new participants, by employing the algorithm presented in Section IV. Recall that the algorithm is associated to two parameters k and d $(\geq k)$, and the size of the secret is $(d-k+1)$ values over $\mathbb{F}_q$. Assume that $q > (n+1)$. Under our algorithm, the process of adding a new participant requires a consensus of d existing participants. The pair of parameters k and d allow for two levels of threshold, the former to recover the secret and the latter to add new participants. A higher value of the parameter d may be useful in various scenarios. For instance, a higher threshold for adding new participants would help in guarding against Sybil attacks [30], where a malicious participant may attempt to obtain additional shares of the secret, by presenting itself as multiple new participants. In this situation, the parameter d determines the level of scrutiny while adding a new participant.

To provision for the addition of participants in the absence of the dealer, we perform a small modification in the algorithm of Section IV: each participant ℓ $(1 \leq \ell \leq n)$ stores the d values $\boldsymbol{\psi}_\ell^T M$ instead of storing only $\boldsymbol{t}_\ell$ (which is a subset of $\boldsymbol{\psi}_\ell^T M$). Let $\boldsymbol{\psi}_{n+1}^T = [1 \ (n+1) \ (n+1)^2 \ \cdots \ (n+1)^{d-1}]$ be the encoding vector of the new participant. In the presence of a dealer or a trusted entity, its share in the scheme would have been $\boldsymbol{\psi}_{(n+1)}^T M$. Now, assume that some d existing participants (say, participants $1, \ldots, d$) agree to add the a new participant $(n+1)$. Each of these d participants $1 \leq j \leq d$ passes $\boldsymbol{\psi}_j^T M \boldsymbol{\psi}_{n+1}$ to participant $(n+1)$. As shown in Theorem 1, the new participant can recover its desired data $\boldsymbol{\psi}_{n+1}^T M$ from $\{\boldsymbol{\psi}_j^T M \boldsymbol{\psi}_{n+1}\}_{j=1}^d$. Clearly, under this algorithm for addition of a new participant, no participant obtains any additional information.

C. Active Adversaries and Verification of Shares

Throughout the paper we assumed a honest-but-curious model, where the participants honestly follow the protocol, but may gather any available information. Now, we consider the case when some participants may be active adversaries, i.e., may pass corrupt values to its neighbours (in addition to trying to gather information about the secret). We show how to modify the communication algorithm of Section IV to handle the case when there are upto t active adversaries in the system, for some given parameter t.

The modified algorithm requires the network to satisfy a $(d+2t)$-propagating-dealer condition; let us assume this holds. Under the algorithm, the dealer computes the matrix M and the encoding vectors, as described in Section IV. As before, to each participant i directly connected to the dealer, it passes the data $\boldsymbol{\psi}_i^T M$. The modification is in the downloads performed by the remaining participants. Every participant ℓ who is not connected to the dealer waits for receiving data from $(d+2t)$ of its neighbours. Let us assume that participant ℓ receives data form neighbours $\{j_1, \ldots, j_{d+2t}\}$. According to the protocol, this data is the set of $(d+2t)$ values $\{\boldsymbol{\psi}_{j_1}^T M \boldsymbol{\psi}_\ell, \ldots, \boldsymbol{\psi}_{j_{d+2t}}^T M \boldsymbol{\psi}_\ell\}$. By construction, any d vectors from the set $\{\boldsymbol{\psi}_{j_1}^T, \ldots, \boldsymbol{\psi}_{j_{d+2t}}^T\}$ are linearly independent. Thus, the $(d+2t)$ values downloaded by node ℓ form a Maximum-Distance-Separable (MDS) encoding of the d-length vector $M\boldsymbol{\psi}_\ell$. Furthermore, since at-most t of the participants may be actively adversarial, no more than t out of the $(d+2t)$ downloaded values can be in error. Thus, participant ℓ can apply standard Reed-Solomon code decoding algorithms [28] and

recover $M\boldsymbol{\psi}_\ell$ correctly. Finally, since M is symmetric by construction (6), participant ℓ equivalently obtains its desired data $\boldsymbol{\psi}_\ell^T M$.

Let us now consider the case when the dealer may be malicious. A dealer is malicious if the shares it passes to the participants are not consistent with each other, i.e., different sets of k shares may decode to different values of the secret s. Under the algorithm presented, the participants can detect a malicious dealer without leaking any information, by comparing parts of their shares with each other. Observe that under the algorithm, every pair of participants, say participants i and j, store one common value $\boldsymbol{\psi}_i^T M \boldsymbol{\psi}_j = \boldsymbol{\psi}_j^T M \boldsymbol{\psi}_i$. Moreover, as discussed above, the values a participant (say, participant i) stores in common with all other participants, form an MDS encoding $\{\boldsymbol{\psi}_i^T M \boldsymbol{\psi}_j\}_{j\in[n]}$ of $\boldsymbol{\psi}_i^T M$. Thus, inconsistencies in the shares due to a malicious dealer can be detected via pairwise comparisons of the common symbols among the participants.

D. Two-threshold Secret Sharing

In [31], authors introduced a modification of Shamir's secret sharing scheme to include two thresholds k and k' $(< k)$. The modified scheme satisfies the properties of k-secret-recovery and k'-collusion-resistance (Shamir's original scheme is a special case with $k' = k-1$). The relaxation of k' to a value smaller than $(k-1)$ allows for the reduction of the size of each share (normalized by the message size), thus requiring the dealer to transmit a smaller amount of data, and the participants to store lesser data.

We now generalize the algorithm of Section IV, for secret share dissemination across a general network, to accommodate two thresholds. The generalization only modifies the structure of matrix M in (6) in the original algorithm. Given two thresholds k and k', the dimensions of the constituent submatrices of M are changed to

$$M = \underbrace{\underbrace{\begin{bmatrix} S_A & R_a^T & S_B^T \\ R_a & R_b & R_c^T \\ S_B & R_c & 0 \end{bmatrix}}_{k'} \underbrace{}_{k-k'} \underbrace{}_{d-k}}_{d} \tag{10}$$

where

- S_A is a *symmetric* $(k' \times k')$ matrix containing $\frac{k'(k'+1)}{2}$ secret values,
- S_B is $((d-k) \times k')$ matrix containing $k'(d-k)$ secret values,
- R_a is a $((k-k') \times k')$ matrix containing $k'(k-k')$ random values,
- R_b is a $((k-k') \times (k-k'))$ *symmetric* matrix containing $\frac{k(k-1)}{2}$ by random values,
- R_c is a $((d-k) \times (k-k'))$ matrix containing $(d-k)(k-k')$ random values.

Each random or secret value is drawn from the finite field $\mathbb{F}_q$. Note that M continues to be a $(d \times d)$ symmetric matrix. The remaining algorithm remains the same as in Section IV. The properties of k-secret-recovery, k'-collusion-resistance and robustness to network structure can be verified via arguments similar to those in Section IV-C for the original algorithm.

Remark 4 (Content distribution): In the absence of security requirements, this scheme can be used for content distribution across a network, by replacing all random values by message values (i.e., with $k' = 0$). In this scenario, each receiver in the content distribution network can recover the entire data M by connecting to any arbitrary set of k of the n nodes.

E. Degree of the Share Polynomial

In the secret share dissemination algorithm of Section IV, observe from (8) that when $d \geq (k+1)$, the first secret value is encoded as a polynomial of degree $(d-1)$ (all other secret values are encoded as polynomials of

degree $(k-1)$). In certain applications, however, it may be required to have polynomials of degree no more than $(k-1)$. In such a situation, one may simply replace the secret value S_A in (6) with a random value. While this step reduces the communication efficiency of the algorithm, it retains all other essential properties including its distributed nature, k-secret-recovery, $(k-1)$-collusion-resistance, and robustness to the network structure.

F. Other related literature

1) Regenerating Codes for Distributed Storage:
Regenerating codes [27] are a class of codes for distributed storage systems that aim to provide reliability and efficient fault-handling. Product-matrix codes [20], upon which our algorithm is based, are a class of explicit constructions of regenerating codes. These codes have the following special properties that make them suitable for secret share dissemination: (a) scalability, i.e., a node under repair is not constrained to connect to all remaining nodes (the only other scalable regenerating code constructions are the high-rate MDS codes of [32], [33]; however, no secure versions of these codes are known), (b) information-theoretically secure [21], [34] (the only other secure codes are the secure repair-by-transfer codes of [32], [35], [36]; however, these codes are not scalable), (c) a failed node can be repaired from *any* d remaining nodes (thus aiding in the distributed nature), (d) data a node passes to the failed node is independent of the identities of the other nodes helping in repair (again, making it distributed).

2) Interference Alignment in Wireless Communication:
Consider the $(n=6, k=2)$ toy example of our algorithm, depicted in Fig. 2b. Here, the data passed by any participant j to its neighbour ℓ is $((s+\ell r)+j(r+\ell r_a))$. The share desired by participant ℓ is $(s+\ell r)$, and by design, the data passed by participant j to ℓ is a linear combination of participant ℓ's share $(s+\ell r)$, and a random term $(r+\ell r_a)$. The random term $(r+\ell r_a)$ ensures that participant j does not possess any information about the share $(s+\ell r)$ of participant ℓ. Now, to enable participant ℓ to remove this (undesired) random component, the algorithm ensures that the data passed by any participant i to participant ℓ is $(s+\ell r)$ obfuscated with (a multiple of) the same random term $(r+\ell r_a)$. The linear dependence among these (undesired) random components allows participant ℓ to solve for the value of $(s+\ell r)$ using the fewest number of equations (i.e., least amount of download), thereby reducing the communication complexity. The general algorithm presented in Section IV also forces the undesired components, in the data downloaded by any participant, to span a small dimension. Interestingly, such a phenomenon of restricting the dimension of undesired components has recently received considerable attention in the wireless communication literature, and is termed *interference alignment* [37], [38]. This arises in a setting where multiple transmitter-receiver pairs communicate simultaneously over a wireless channel. At any receiver, the signals transmitted by all other transmitters constitute (undesired) interference, which need to be restricted to a small dimension in order to achieve a higher communication efficiency.

3) Certain Authentication Protocols:
Although our path leading to this result was via the area of distributed storage, an authentication protocol for MANETS presented in [39], and a commitment-verification protocol of [40] turn out to be special cases of the encoding part of our algorithm. In these works, the dealer constructs a random symmetric bivariate polynomial with the secret as its constant term, and provides each participant with an evaluation of this polynomial in the first coordinate at the participant's index (leaving the second coordinate as a variable). This is analogous to the encodings in our scheme when $d=k$.

APPENDIX B
PROOF OF $(k-1)$-COLLUSION-RESISTANCE

Proof of Theorem 3: The proof is a modification of [21, Theorem 1]. Let $\mathcal{J} \subset [n]$ denote the set of $(k-1)$ participants colluding in an attempt to recover information about the secret $\mathbf{s}$. Denote the number of secret values (over $\mathbb{F}_q$) by $S = d-k+1$. Further, let $\mathbf{r}$ denote the collection of (from (7)) $R = (k-1)d - \binom{k-1}{2}$ random values introduced initially by the dealer.

The data obtained by any participant $j \in \mathcal{J}$ is a subset of the values $\psi_j^T M$ and $\{\psi_\ell^T M \psi_j\}_{\ell \in [n]}$. Since matrix M is symmetric, $\{\psi_\ell^T M \psi_j\}_{\ell \in [n]} = \{\psi_j^T M \psi_\ell\}_{\ell \in [n]}$. Thus, participant j obtains at-most the d values $\psi_j^T M$ in the

execution of the protocol. On the other hand, Theorem 3 shows that participant j can completely recover $\psi_j^T M$. Now, let $\Psi_{\mathcal{J}}$ be the $((k-1)\times d)$ submatrix of Ψ comprising of the $(k-1)$ vectors $\{\boldsymbol{\psi}_j^T\}_{j\in\mathcal{J}}$ as its rows. Under this notation, the $(k-1)$ colluding participants together have access to at-most the $(k-1)d$ values

$$C_{\mathcal{J}} = \Psi_{\mathcal{J}} M = \Psi_{\mathcal{J}} \begin{bmatrix} S_A & R_a^T & S_B^T \\ R_a & R_b & R_c^T \\ S_B & R_c & 0 \end{bmatrix} .$$

Let $\mathbf{e}$ denote the set of these $(k-1)d$ values.

Throughout the proof, we will use the function $H(.)$ to denote the Shannon entropy. All logarithms in the computation of the entropy functions are assumed to be taken to the base q.

As an intermediate step in the proof, we shall show that given all the secret values $\mathbf{s}$ as side-information, the $(k-1)$ colluding participants can recover all the R random values, i.e., $H(\mathbf{r}|\mathbf{e},\mathbf{s}) = 0$. To this end, observe that if the secret values S_A and S_B are known to the eavesdropper, and since the code is linear, it can subtract the components of S_A and S_B from $C_{\mathcal{J}}$, to obtain

$$C_{\mathcal{J}}^{'} = \Psi_{\mathcal{J}} \begin{bmatrix} 0 & R_a^T & 0^T \\ R_a & R_b & R_c^T \\ 0 & R_c & 0 \end{bmatrix} .$$

Since $\Psi_{\mathcal{J}}$ is Vandermonde with all non-zero entries, when restricted to columns 2 to $(k-1)$, it forms a $((k-1)\times(k-1))$ invertible matrix. This allows recovery of the random values in R_a and R_c. Subtracting the components of these decoded values, one is left with

$$C_{\mathcal{J}}^{''} = \Psi_{\mathcal{J}} \begin{bmatrix} 0 & 0^T & 0^T \\ 0 & R_b & 0^T \\ 0 & 0 & 0 \end{bmatrix} ,$$

and in a manner identical to that of decoding R_a and R_c, the eavesdropper can decode the remaining random values R_b. Thus, given the secret values, the $(k-1)$ participants can decode all the random values, and hence

$$H(\mathbf{r}|\mathbf{e},\mathbf{s}) = 0 . \tag{11}$$

As another intermediate step in the proof, we will now show that all but R of the values obtained by the $(k-1)$ participants are functions of the other values that they possess, i.e., $H(\mathbf{e}) \leq R$. From the value of R in (7), it suffices to show that out of the $(k-1)d$ values that the eavesdropper has access to, $\binom{k-1}{2}$ of them are functions (linear combinations) of the rest. Consider, the $((k-1)\times(k-1))$ matrix

$$C_{\mathcal{J}}\Psi_{\mathcal{J}}^T = \Psi_{\mathcal{J}} M \Psi_{\mathcal{J}}^T . \tag{12}$$

Since M is symmetric, this $((k-1)\times(k-1))$ matrix in (12) is also symmetric. Thus $\binom{k-1}{2}$ dependencies among the elements of $\mathcal{J}$ can be described by the $\binom{k-1}{2}$ upper-triangular elements of the expression

$$C_{\mathcal{J}}\Psi_{\mathcal{J}}^T - \Psi_{\mathcal{J}} C_{\mathcal{J}}^T = 0 . \tag{13}$$

Since the rows of $\Psi_{\mathcal{J}}$ are linearly independent, these $\binom{k-1}{2}$ redundant equations are independent. Thus the eavesdropper has access to at-most $(k-1)d - \binom{k-1}{2}$ independent values, which equals the value of R, and hence

$$H(\mathbf{e}) \leq R. \tag{14}$$

We finally show that the two conditions (11) and (14) above must necessarily imply that the mutual information

between the secret values $\mathbf{s}$ and the values obtained by the eavesdropper $\mathbf{e}$ is zero, i.e., $I(\mathbf{s};\mathbf{e}) = 0$.

$$\begin{aligned} I(\mathbf{s};\mathbf{e}) &= H(\mathbf{e}) - H(\mathbf{e}|\mathbf{s}) && (15)\\ &\leq R - H(\mathbf{e}|\mathbf{s}) && (16)\\ &= R - H(\mathbf{e}|\mathbf{s}) + H(\mathbf{e}|\mathbf{s},\mathbf{r}) && (17)\\ &= R - I(\mathbf{e};\mathbf{r}|\mathbf{s}) && (18)\\ &= R - (H(\mathbf{r}|\mathbf{s}) - H(\mathbf{r}|\mathbf{e},\mathbf{s})) && (19)\\ &= R - H(\mathbf{r}|\mathbf{s}) && (20)\\ &= R - R && (21)\\ &= 0\ , && (22) \end{aligned}$$

where (16) follows from (14); (17) follows since every value in the system is a function of $\mathbf{s}$ and $\mathbf{r}$, giving $H(\mathbf{e}|\mathbf{s},\mathbf{r}) = 0$; (20) follows from (11); and (21) follows since the random values are independent of the secret values. ■