Verifiable Secret Redistribution

Theodore M. Wong Jeannette M. Wing

October 2001

CMU-CS-01-155

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Abstract

We present a new protocol to perform non-interactive **verifiable secret redistribution** (VSR) for secrets distributed with Shamir's secret sharing scheme. We base our VSR protocol on Desmedt and Jajodia's redistribution protocol for linear secret sharing schemes, which we specialize for Shamir's scheme. We extend their redistribution protocol with Feldman's non-interactive verifiable secret sharing scheme to ensure that a SUBSHARES-VALID condition is true after redistribution. We show that the SUBSHARES-VALID condition is necessary but not sufficient to guarantee that the new shareholders have valid shares, and present an additional SHARES-VALID condition.

This research is sponsored by the Defense Advanced Research Projects Agency (DARPA), Advanced Technology Office, under the title "Organically Assured and Survivable Information Systems (OASIS)" (Air Force Cooperative Agreement no. F30602-00-2-0523).

Keywords: non-interactive verifiable secret redistribution, linear sharing schemes

1 Introduction

Suppose we have a system that distributes shares of a secret to a set of n servers such that the system can reconstruct the secret, or performs distributed computations, with m of the n shares. An example of such systems is a multiparty signature system [10, 11, 12, 15, 20] in which a dealer distributes shares of a key to a set of signature servers. The servers can then collaborate to create digital signatures, but none will have knowledge of the key. Other examples of such systems include survivable storage systems [24, 23] in which a client stores data objects on remote storage servers. The client can retrieve its objects even if up to $(n-m)$ servers fail, and adversaries that subvert less than m servers gain no knowledge about the objects.

If a server fails or is subverted by an adversary, we may wish to redistribute the remaining shares to a new set of n' servers. The dealer may be unavailable for redistribution of the shares, since it may have gone off-line since distribution. The servers may be available, but they are not trusted with secret. Thus, we require a protocol for redistribution without reconstruction of the secret. We also require verification that the new shareholders have **valid** shares (ones that can be used to reconstruct the secret).

We present a new protocol to perform non-interactive **verifiable secret redistribution** (VSR) for secrets distributed with Shamir's secret sharing scheme [22]. Suppose we have distributed shares of a secret to shareholders in Shamir's (m, n) **threshold scheme** (one in which we require m of n shares to reconstruct the secret), and wish to redistribute the secret to shareholders in a new (m', n') scheme. Furthermore, suppose we wish to avoid reconstruction of the secret. Our VSR protocol enables the redistribution of the secret from the old to new shareholders without reconstruction of the secret by any of the shareholders, and guarantees that the new shareholders have valid shares. Our protocol guards against faulty behavior by up to $n-m$ of the old shareholders provided that $m > \frac{n}{2}$. Figure 1 shows the application of our VSR protocol.

We base our VSR protocol on Desmedt and Jajodia's redistribution protocol for linear secret sharing schemes [8], which we specialize for Shamir's scheme. In their protocol, m of n old shareholders each distribute n' **subshares** of their shares of a secret, and n' new shareholders combine m subshares (one from each old shareholder) to generate new shares. m' new shares are required to reconstruct the secret. Unlike our protocol, their protocol assumes non-faulty old shareholders. Thus, faulty old shareholders, without risk of detection, may cause new shareholders to generate invalid shares by distributing invalid subshares.

We extend Desmedt and Jajodia's redistribution protocol with Feldman's non-interactive **verifiable secret sharing** (VSS) scheme [9] to ensure that a SUBSHARES-VALID condition is true after redistribution. With Feldman's scheme, each old shareholder broadcasts a zero-knowledge proof of the validity of the sub-

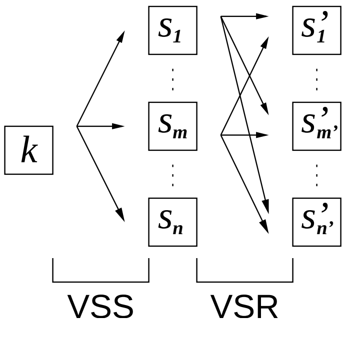

Figure 1: Initial distribution of a secret k with Shamir's (m, n) threshold secret sharing scheme [22], followed by redistribution to an (m', n') scheme. Verifiable secret sharing (VSS) schemes can be used to guarantee that the shares $s_1 \dots s_n$ are valid. Our new verifiable secret redistribution (VSR) protocol can be used to guarantee that the shares $s'_1 \dots s'_{n'}$ are valid.

shares to the new shareholders. The new shareholders verify the proof without further interaction with the old shareholders. Feldman assumes there exist homomorphic encryption functions that are hard to invert, allowing the old shareholder to broadcast encryptions of their share and the subshare generation function without revealing them. Feldman also assumes there exist reliable broadcast communication channels among all participants and private channels between every pair of participants.

We show that the SUBSHARES-VALID condition is necessary but not sufficient to guarantee that new shareholders have valid shares, and present an additional SHARES-VALID condition. The old shareholders broadcast a zero-knowledge proof of the validity of their shares of the secret to the new shareholders. As before, the new shareholders verify the proof without further interaction with the old shareholders. The check of the SHARES-VALID condition also assumes there exist homomorphic encryption functions that are hard to invert, allowing old shareholders to prove the validity of their shares to new shareholders without revealing them. We prove that the SUBSHARES-VALID and SHARES-VALID conditions are necessary and sufficient to guarantee that the new shareholders generate valid shares of the original secret.

2 Related work

Blakley and Shamir invented secret sharing schemes independently. In Blakley's scheme [2], the intersection of m of n vector spaces yields a one-dimensional vector that corresponds to the secret. Desmedt presents a survey of other sharing schemes [7].

Feldman's VSS scheme [9] is one of several to catch a dealer that attempts to distribute invalid shares. Chor *et al* present a scheme in which the dealer and shareholders perform an interactive secure distributed computation [6]. Benaloh [1], Gennaro and Micali [13], Goldreich *et al* [14], and Rabin and Ben-Or [21, 19] subsequently propose schemes in which the dealer and shareholders participate in an interactive zero-knowledge proof of validity; the schemes of Gennaro and Micali, and Rabin and Ben-Or, are information-theoretically secure. Pederson [18] presents a scheme, like Feldman's, in which the dealer broadcasts a non-interactive zero-knowledge proof of validity to the shareholders. Our VSR procotol differs from previous VSS schemes in that the multiple "dealers" of the new shares (the old shareholders) do not have the original secret, and must use other information to generate a proof for the new shareholders. Also, unlike in VSS schemes, each new shareholder must perform two checks: one to verify the validity of the subshares distributed by the old shareholders, and another to verify the validity of the shares generated by the new shareholders.

Desmedt and Jajodia present the first protocol to alter the access structure of a secret sharing scheme by physical redistribution of shares between the old and new shareholders [8]. Cachin proposes a secret sharing scheme that **enrolls** (adds) shareholders in the access structure after the initial sharing [5]. Blakley *et al* consider threshold schemes that **disenroll** (remove) shareholders from the access structure with broadcast messages [3]. For these schemes, the set of new shareholders is not disjoint from the old; rather, it is either a superset (for Cachin) or a subset (for Blakley *et al*). Blundo *et al* presents a scheme in which the dealer uses broadcast messages to activate different, possibly disjoint, authorized subsets [4]. Blundo's scheme requires shareholders to have a share regardless of whether or not they are in the active authorized subset, in contrast to Desmedt and Jajodia's scheme. Our VSR protocol, like Desmedt and Jajodia's protocol, alters the access structure of a scheme by physical redistribution of shares, and additionally provides a proof to the new shareholders that they have valid shares.

Ostrovsky and Yung define **mobile adversaries** that subvert storage servers at a constant rate, and propose a general **proactive secret sharing** (PSS) protocol for the periodic redistribution of shares to counteract them [17]. Their protocol redistributes shares to the same access strucutre. Herzberg *et al* specialize the proactive approach to Shamir's scheme [16], and other researchers use this work to develop robust and secure multiparty signature schemes [10, 11, 12, 15, 20]. Zhou, Schneider, and van Renesse propose a PSS protocol for asynchronous, wide-area networks, and employ it in an on-line certification authority [25]. Our VSR procotol, unlike PSS protocols, can redistribute shares to arbitrary access structures. However, we assume there exist reliable broadcast communication channels among all participants and private channels between every pair of participants in our protocol, which Zhou *et al* avoid in their asynchronous protocol.

3 The building blocks for the VSR protocol

In this section, we outline the cryptographic protocols that form the building blocks for our VSR protocol. We begin with a summary of Desmedt and Jajodia's secret redistribution protocol [8] for linear secret sharing schemes, and we show how to specialize its operation to Shamir's secret sharing scheme [22]. We follow with a recap of Feldman's VSS scheme [9], and present an application by Herzberg *et al* [16] of Feldman's scheme to Shamir's scheme.

3.1 Mathematical notation

A linear secret sharing scheme [8] is an algorithm for the distribution of shares of a secret to a group of shareholders such that the secret is a linear combination of a subset of the shares. We define a secret k to be in set $\mathcal{K}$ of secrets, and each shareholder i to be in the set $\mathcal{P}$ of shareholders. To distribute k, we generate a share s_i for each i in $\mathcal{P}$, where s_i is in the set $\mathcal{S}_i$ of shares, and $\mathcal{S}_i$ is in the set $\mathcal{S}$ of share sets. To reconstruct the secret, we combine s_i from all i in an **authorized subset** $\mathcal{B}$ of $\mathcal{P}$:

$$k = \sum_{i \in \mathcal{B}} \psi_i(s_i) \tag{1}$$

ψ_i is a homomorphism from $\mathcal{S}_i$ to $\mathcal{K}$; we aggregate ψ_i into the set ψ of homomorphisms. The authorized subsets are in the **access structure** $\Gamma_{\mathcal{P}}$. We represent a linear sharing scheme as a tuple $\{\Gamma_{\mathcal{P}}, \mathcal{K}, \mathcal{S}, \psi\}$.

3.2 Shamir's secret sharing scheme

Shamir presents an (m, n) threshold secret sharing scheme based on polynomial interpolation [22]. The secret k is in $\mathbb{Z}_p$ (p prime; $p > n$), and each shareholder i is in the set $\mathcal{P}$ ($|\mathcal{P}| = n$). All mathematical operations are in the finite field $\mathbb{Z}_p$. To distribute k, we select a polynomial $a(x)$ with degree $m-1$ and constant term k, and generate a share s_i for each i in $\mathcal{P}$ with $a(x)$:

$$s_i = k + a_1 i + \ldots a_{m-1} i^{m-1} \tag{2}$$

where s_i is also in $\mathbb{Z}_p$. To reconstruct k, we retrieve m coordinate pairs (i, s_i) of all i in $\mathcal{B}$ ($|\mathcal{B}| = m$; $\mathcal{B} \in \Gamma_{\mathcal{P}}$), and use the pairs in the Lagrange interpolation formula:

$$k = \sum_{i \in \mathcal{B}} b_i s_i \quad \text{where} \quad b_i = \prod_{j \in \mathcal{B}, j \neq i} \frac{j}{(j-i)} \tag{3}$$

We represent Shamir's scheme with the tuple $\{\Gamma_{\mathcal{P}}, \mathbb{Z}_p, \{\mathbb{Z}_p\}, \psi_{\mathcal{P}}\}$, where $\psi_i(s_i) = b_i s_i$ and $\psi_i \in \psi_{\mathcal{P}}$.

3.3 Desmedt and Jajodia's share redistribution protocol

Desmedt and Jajodia present a protocol for the redistribution of secrets distributed by linear sharing schemes without reconstruction of the original secret [8]. Suppose we have distributed shares s_i of a secret k to shareholders i using the scheme $(\Gamma_{\mathcal{P}}, \mathcal{K}, \mathcal{S}, \psi)$, and wish to redistribute it using a different scheme

1. There exists a linear sharing scheme $(\Gamma_{\mathcal{P}}, \mathcal{K}, \mathcal{S}, \psi)$, and each $i \in \mathcal{P}$ has received a share $s_i \in \mathcal{S}_i \in \mathcal{S}$ of $k \in \mathcal{K}$.
2. For each $i \in \mathcal{P}$ there exists an intermediate linear scheme $(\Gamma'_{\mathcal{P}'}, \mathcal{S}_i, \hat{\mathcal{S}}_i, \hat{\psi}_i)$ for distributing shares s_i into subshares $\hat{s}_{ij}$ to each $j \in \mathcal{P}'$.
3. Addition of elements in $\mathcal{K}$ is commutative.
4. For each $i \in \mathcal{B} \in \Gamma_{\mathcal{P}}$ and $j \in \mathcal{B}' \in \Gamma'_{\mathcal{P}'}$, there exist homomorphisms ψ_i, $\hat{\psi}_{ij}$, ψ'_j, and $\hat{\psi}'_{ji}$ such that:

$$\psi_i\left(\hat{\psi}_{ij}(\hat{s}_{ij})\right) = \psi'_j\left(\hat{\psi}'_{ji}(\hat{s}_{ij})\right)$$

Figure 2: Conditions required for the redistribution of shares from linear sharing schemes [8].

Non-verifiable Secret Redistribution protocol:
To redistribute shares s_i of a secret k distributed using the linear sharing scheme $(\Gamma_{\mathcal{P}}, \mathcal{K}, \mathcal{S}, \psi)$ into shares s'_j distributed using the linear sharing scheme $(\Gamma'_{\mathcal{P}'}, \mathcal{K}, \mathcal{S}', \psi')$:

1. Select an authorized subset $\mathcal{B}$ in $\Gamma_{\mathcal{P}}$. Use the intermediate linear scheme $(\Gamma'_{\mathcal{P}'}, \mathcal{S}_i, \hat{\mathcal{S}}_i, \hat{\psi}_i)$ to distribute subshares $\hat{s}_{ij}$ of each share s_i of i in $\mathcal{B}$ to each j in $\mathcal{P}'$.
2. For each $j \in \mathcal{P}'$, compute a new share s'_j by treating the subshares $\hat{s}_{ij}$ as those distributed by another intermediate scheme $(\Gamma_{\mathcal{P}}, \mathcal{S}'_j, \hat{\mathcal{S}}'_j, \hat{\psi}'_j)$, and using a variant of Equation (1):

$$s'_j = \sum_{i \in \mathcal{B}} \hat{\psi}'_{ji}(\hat{s}_{ij})$$

Figure 3: Redistribution protocol for linear sharing schemes [8].

$(\Gamma'_{\mathcal{P}'}, \mathcal{K}, \mathcal{S}', \psi')$. We achieve this by selecting an authorized subset $\mathcal{B}$ in $\Gamma_{\mathcal{P}}$ and using an intermediate scheme $(\Gamma'_{\mathcal{P}'}, \mathcal{S}_i, \hat{\mathcal{S}}_i, \hat{\psi}_i)$ to distribute subshares $\hat{s}_{ij}$ of each s_i of i in $\mathcal{B}$ to each shareholder j in $\mathcal{P}'$, where the set $\hat{\mathcal{S}}_i$ of sets of subshares is:

$$\hat{\mathcal{S}}_i = \left\{ \hat{\mathcal{S}}_{ij} : j \in \mathcal{B}', \mathcal{B}' \in \Gamma'_{\mathcal{P}'} \right\} \tag{4}$$

and the set $\hat{\psi}_i$ of homomorphisms from $\hat{\mathcal{S}}_i$ to $\mathcal{S}_i$ is:

$$\hat{\psi}_i = \left\{ \hat{\psi}_{ij} : j \in \mathcal{B}', \mathcal{B}' \in \Gamma'_{\mathcal{P}'} \right\} \tag{5}$$

If we treat $\hat{s}_{ij}$ as being distributed by another intermediate scheme $(\Gamma_{\mathcal{P}}, \mathcal{S}'_j, \hat{\mathcal{S}}'_j, \hat{\psi}'_j)$ (with $\hat{\mathcal{S}}_j$ and $\hat{\psi}_j$ defined as $\hat{\mathcal{S}}_i$ and $\hat{\psi}_i$ in Equations (4) and (5)), we can generate a share s'_j for each j. For schemes that satisfy the conditions in Figure 2, we can can use the protocol in Figure 3 to redistribute shares.

To redistribute secrets from Shamir's (m, n) threshold secret sharing scheme [22] to an (m', n') scheme using Desmedt and Jajodia's protocol, we first need to show that the conditions in Figure 2 hold. Desmedt and Jajodia present a sketch of the specialization of their protocol to Shamir's scheme, but no details. We represent the (m, n) and (m', n') schemes as $(\Gamma_{\mathcal{P}}, \mathbb{Z}_p, \{\mathbb{Z}_p\}, \psi_{\mathcal{P}})$ and $(\Gamma_{\mathcal{P}'}, \mathbb{Z}_p, \{\mathbb{Z}_p\}, \psi_{\mathcal{P}'})$ respectively.

1. Reconstruction of the original secret from the shares s_i in Equation (3) is a linear recombination in the form of Equation (1), and so the scheme $(\Gamma_{\mathcal{P}}, \mathbb{Z}_p, \{\mathbb{Z}_p\}, \psi_{\mathcal{P}})$ is linear. Thus, Condition 1 holds.

2. Generation of the subshares $\hat{s}_{ij}$ of s_i for each shareholder j in $\mathcal{P}'$ can be performed with the new scheme: $(\Gamma'_{\mathcal{P}'}, \mathcal{S}_i, \hat{\mathcal{S}}_i, \hat{\psi}_i) = (\Gamma_{\mathcal{P}'}, \mathbb{Z}_p, \{\mathbb{Z}_p\}, \psi_{\mathcal{P}'})$. Thus, Condition 2 holds.

3. Addition in $\mathbb{Z}_p$ is commutative. Thus, Condition 3 holds.

4. Given the old scheme $(\Gamma_{\mathcal{P}}, \mathbb{Z}_p, \{\mathbb{Z}_p\}, \psi_{\mathcal{P}})$, the new scheme $(\Gamma_{\mathcal{P}'}, \mathbb{Z}_p, \{\mathbb{Z}_p\}, \psi_{\mathcal{P}'})$, and the intermediate scheme $(\Gamma_{\mathcal{P}'}, \mathbb{Z}_p, \{\mathbb{Z}_p\}, \psi_{\mathcal{P}'})$ (from Condition 2), the homomorphisms ψ_i, $\hat{\psi}_{ij}$, and ψ'_j are:

$$\begin{aligned}
\psi_i(s_i) &= b_i s_i \\
\hat{\psi}_{ij}(\hat{s}_{ij}) &= b'_j \hat{s}_{ij} \quad \text{where} \quad b'_j = \prod_{l \in \mathcal{B}, l \neq j} \frac{l}{(l-j)} \\
\psi'_j(s'_j) &= b'_j s'_j
\end{aligned}$$

We need to find ψ'_{ji}. We have:

$$\begin{aligned}
\psi_i\left(\hat{\psi}_{ij}\left(\hat{s}_{ij}\right)\right) &= b_i\left(b'_j \hat{s}_{ij}\right) && \text{(definitions of } \psi_i \text{ and } \hat{\psi}_{ij}\text{)} \\
&= b'_j\left(b_i \hat{s}_{ij}\right) && (xy = yx; x(yz) = (xy)z = xyz) \\
&= \psi'_j\left(b_i \hat{s}_{ij}\right) && \text{(definition of } \psi'_j\text{)} \\
&= \psi'_j\left(\hat{\psi}'_{ji}\left(\hat{s}_{ij}\right)\right) && \text{(define } \hat{\psi}'_{ji}\left(\hat{s}_{ij}\right) = b_i \hat{s}_{ij}\text{)}
\end{aligned}$$

Thus, Condition 4 holds by defining:

$$\hat{\psi}'_{ji}\left(\hat{s}_{ij}\right) = b_i \hat{s}_{ij}$$

□

Non-verifiable Secret Redistribution protocol (for Shamir's scheme):
To redistribute shares from $(\Gamma_{\mathcal{P}}, \mathbb{Z}_p, \{\mathbb{Z}_p\}, \psi_{\mathcal{P}})$ to $(\Gamma_{\mathcal{P}'}, \mathbb{Z}_p, \{\mathbb{Z}_p\}, \psi_{\mathcal{P}'})$, using an authorized subset $\mathcal{B} \in \Gamma_{\mathcal{P}}$:

1. For each $i \in \mathcal{B}$, for each $j \in \mathcal{P}'$, compute subshares $\hat{s}_{ij}$ from the polynomial $a_i(x)$.
2. For each $j \in \mathcal{P}'$, transfer $\hat{s}_{ij}$.
3. For each $j \in \mathcal{P}'$, compute the new share s'_j using the Lagrange interpolation formula:

$$s'_j = \sum_{i \in \mathcal{B}} b_i \hat{s}_{ij} \quad \text{where} \quad b_i = \prod_{x \in \mathcal{B}, x \neq i} \frac{x}{(x-i)}$$

b_i are constant for each $i \in \mathcal{B}$, are independent of the choice of $a_i(x)$, and may be precomputed.

Figure 4: Protocol to redistribute shares from Shamir's (m, n) threshold secret sharing scheme [22] to an (m', n') scheme [8].

Feldman's Verifiable Secret Sharing scheme (for Shamir's scheme):
To distribute a secret $k \in \mathbb{Z}_p$ to shareholders $\mathcal{P} = \{1, \ldots, n\}$:

1. Compute the shares s_i for secret k using a polynomial $a(x) = k + a_1 i + \ldots + a_{m-1} i^{m-1}$, and distribute the shares to the corresponding $i \in \mathcal{P}$ over private channels.
2. Send g^k and $g^{a_1} \ldots g^{a_{m-1}}$ to all $i \in \mathcal{P}$ over the broadcast channel.
3. For each $i \in \mathcal{P}$, verify that:

$$g^{s_i} \equiv g^k (g^{a_1})^i \ldots (g^{a_{m-1}})^{i^{m-1}}$$

If the check passes, i broadcasts a "commit" message. Otherwise, i broadcasts an "abort" message.

Figure 5: Feldman's verifiable secret sharing scheme [9], as applied to Shamir's (m, n) threshold secret sharing scheme [22] by Herzberg *et al* [16].

To perform redistribution, we treat each of the shares generated by Shamir's (m, n) threshold scheme as a secret to distribute using the (m', n') scheme. We use the scheme $(\Gamma_{\mathcal{P}'}, \mathbb{Z}_p, \{\mathbb{Z}_p\}, \psi_{\mathcal{P}'})$ to compute a $\hat{s}_{ij}$ of s_i for each j in $\mathcal{P}'$, for s_i of each i in $\mathcal{B}$; we note that each i can select its own polynomial $a(x)$ (Equation (2)). Then, each j computes a new share s'_j from $\hat{s}_{ij}$ as described in Figure 3 with $\hat{\psi}'_{ji}$:

$$s'_j = \sum_{i \in \mathcal{B}} b_i \hat{s}_{ij} \tag{6}$$

A summary of the redistribution protocol for Shamir's scheme is shown in Figure 4.

3.4 Feldman's VSS scheme

Feldman presents a VSS scheme that can be used by shareholders of a secret to verify the validity of their shares [9]. Here, we recap an application by Herzberg *et al* [16] of Feldman's scheme to Shamir's secret sharing scheme [22]. Feldman's scheme is shown in Figure 5.

The application of Feldman's VSS scheme to Shamir's scheme takes advantage of the homomorphic properties of exponentiation and the assumption that the computation of discrete logs in a finite field is intractable. As before, we represent Shamir's (m, n) threshold scheme with the tuple $(\Gamma_{\mathcal{P}}, \mathbb{Z}_p, \{\mathbb{Z}_p\}, \psi_{\mathcal{P}})$. Suppose g is a generator for $\mathbb{Z}_p$:

$$\forall b \in \{1, \ldots, p-1\}\, \exists a \in \{1, \ldots, p-1\} : g^a \equiv b \bmod p$$

Then, the dealer of the secret k in set $\mathbb{Z}_p$, in addition to sending shares s_i in $\mathbb{Z}_p$ to each i in the set $\mathcal{P}$ of shareholders, broadcasts exponentiations of k and coefficients $a_1 \ldots a_{m-1}$ of the polynomial used by the dealer to generate the shares (g^k and $g_1^a \ldots g^{a_{m-1}}$). Each i may then verify that their s_i is a valid share of k from the following:

$$g^{s_i} \equiv g^k (g^{a_1})^i \ldots (g^{a_{m-1}})^{i^{m-1}} \tag{7}$$

which is the exponentiation of the polynomial $a(x)$ from Shamir's scheme in Equation (2). Since we have assumed that the computation of discrete logs is intractable, we assume that none of the shareholders can learn k (or $a_1 \ldots a_{m-1}$) from the broadcast of g^k.

4 The non-interactive VSR protocol

We present our non-interactive verifiable secret redistribution protocol for secrets distributed with Shamir's secret sharing scheme [22]. We represent the (m, n) and (m', n') threshold schemes with $\{\Gamma_{\mathcal{P}}, \mathbb{Z}_p, \{\mathbb{Z}_p\}, \psi_{\mathcal{P}}\}$ and $\{\Gamma_{\mathcal{P}'}, \mathbb{Z}_p, \{\mathbb{Z}_p\}, \psi_{\mathcal{P}'}\}$ respectively. We assume the computation of discrete logs in a finite field is intractable, and there exist reliable broadcast communication channels among all participants and private channels between every pair of participants. We also assume that there are at most $n - m$ faulty old shareholders, that $m > \frac{n}{2}$, and that there are n' non-faulty new shareholders.

The initial distribution of a secret (INITIALIZE in Figure 6) proceeds as in Feldman's VSS scheme [9]. The dealer of secret k in $\mathbb{Z}_p$ distributes shares s_i in $\mathbb{Z}_p$ to each shareholder i in the set $\mathcal{P}$ of shareholders, using the polynomial $a(x)$ (step 1 of INITIALIZE). The dealer also broadcasts g^k and $g^{a_1} \ldots g^{a_{m-1}}$, which each i uses to verify the validity of s_i (steps 2 and 3 of INITIALIZE) as in Equation (7). If the check passes, i stores s_i and g^k (step 4 of INITIALIZE). For trusted dealers, we can use Shamir's scheme directly for the initial distribution.

Redistribution of the secret from old to new shareholders (REDISTRIBUTE in Figure 6) proceeds as in Desmedt and Jajodia's protocol [8]. Each i in an authorized subset $\mathcal{B}$ distributes subshares $\hat{s}_{ij}$ in $\mathbb{Z}_p$ of s_i to each shareholder j in the set $\mathcal{P}'$ of shareholders, using the polynomial $a_i'(x)$ (step 1 of REDISTRIBUTE); $a_i'(x)$ for each i may be distinct. Each j generates the new share s_j' (step 4 of REDISTRIBUTE). We may redistribute the secret an arbitrary number of times before we reconstruct it.

For the new shareholders to verify that their shares of the secret are valid after redistribution (step 1 of REDISTRIBUTE in Figure 6), we require that two conditions, SHARES-VALID and SUBSHARES-VALID, are true. When all i in $\mathcal{B}$ ($\mathcal{B}$ in $\Gamma_{\mathcal{P}}$) redistribute s_i to each j in $\mathcal{P}'$, all s_j are valid shares of k if:

SHARES-VALID:
$k = \sum_{i \in \mathcal{B}} b_i s_i$

SUBSHARES-VALID:
$\forall i \in \mathcal{B}, \mathcal{B}' \in \Gamma_{\mathcal{P}'} : s_i = \sum_{j \in \mathcal{B}'} b_j' \hat{s}_{ij}$

We use Feldman's VSS scheme to verify that SUBSHARES-VALID is true in our protocol. The distribution of $\hat{s}_{ij}$ from s_i (step 1 of REDISTRIBUTE in Figure 6) is a simple application of the scheme $\{\Gamma_{\mathcal{P}'}, \mathbb{Z}_p, \{\mathbb{Z}_p\}, \psi_{\mathcal{P}'}\}$. Thus, each i in $\mathcal{B}$ broadcasts g^{s_i} and $g^{a_{i1}} \ldots g^{a_{i(m-1)}}$, which each j uses to verify the validity of $\hat{s}_{ij}$ (step 2 of REDISTRIBUTE). Each j still needs to check whether all s_i of i in $\mathcal{B}$ were valid shares of k.

Verifiable Secret Redistribution protocol:
INITIALIZE: To distribute a secret $k \in \mathbb{Z}_p$ to shareholders $\mathcal{P} = \{1, \ldots, n\}$:

1. Compute the shares s_i for secret k using a polynomial $a(x) = k + a_1 i + \ldots + a_{m-1} i^{m-1}$, and distribute the shares to the corresponding $i \in \mathcal{P}$ over private channels.
2. Send g^k and $g^{a_1} \ldots g^{a_{m-1}}$ to all $i \in \mathcal{P}$ over the broadcast channel.
3. For each $i \in \mathcal{P}$, verify that:
$$g^{s_i} \equiv g^k (g^{a_1})^i \ldots (g^{a_{m-1}})^{i^{m-1}}$$
If the check passes, i broadcasts a "commit" message. Otherwise, i broadcasts an "abort" message.
4. If all n $i \in \mathcal{P}$ agree to commit, each i stores s_i and g^k. Otherwise, they abort the protocol.

REDISTRIBUTE: To redistribute k from shares held by shareholders i in an authorized subset $\mathcal{B} \in \Gamma_{(m,n)}$ to shareholders $\mathcal{P}' = \{1, \ldots, n'\}$:

1. For each $i \in \mathcal{B}$, compute the subshares $\hat{s}_{ij}$ for share s_i using a polynomial $a'_i(x) = s_i + a'_{i1} i + \ldots + a'_{i(m'-1)} i^{m'-1}$, and distribute the subshares to the corresponding $j \in \mathcal{P}'$ over private channels.
2. For each $i \in \mathcal{P}$, send g^k, g^{s_i}, and $g^{a'_{i1}} \ldots g^{a'_{i(m'-1)}}$ to all $j \in \mathcal{P}'$ over the broadcast channel.
3. For each $j \in \mathcal{P}'$, verify that:
$$\forall i \in \mathcal{B} : g^{\hat{s}_{ij}} \equiv g^{s_i} (g^{a'_{i1}})^j \ldots (g^{a'_{i(m'-1)}})^{j^{m'-1}}$$
and:
$$g^k \equiv \prod_{i \in \mathcal{B}} (g^{s_i})^{b_i} \quad \text{where} \quad b_i = \prod_{x \in \mathcal{B}, x \neq i} \frac{x}{(x-i)}$$
If both checks pass, j broadcasts a "commit" message. Otherwise, j broadcasts an "abort" message.
4. If all n' $j \in \mathcal{P}'$ agree to commit, each j computes s'_j:
$$s'_j = \sum_{i \in \mathcal{B}} b_i \hat{s}_{ij}$$
and stores s'_j and g^k. Otherwise, they abort the protocol.

Figure 6: Verifiable secret redistribution protocol for the redistribution of shares from Shamir's (m, n) threshold secret sharing scheme to an (m', n') scheme.

Unfortunately, we cannot use Feldman's VSS scheme to check if SHARES-VALID is true. For example, suppose each i in $\mathcal{P}$ used the scheme to verify the validity of s_i of k. Each i in $\mathcal{P}$ could store g^k, g^{s_i}, and $g^{a_1} \ldots g^{a_{m-1}}$, and broadcast them to each j in $\mathcal{P}'$ during redistribution. Each j would use Equation (7) to verify the validity of each s_i, and generate s'_j. However, since each j generates s'_j by interpolation (step 4 of REDISTRIBUTE in Figure 6) instead of using a polynomial $a'(x)$, it has no coefficients $a'_1 \ldots a'_{m'-1}$ to broadcast during a subsequent redistribution to another set $\mathcal{P}''$ of shareholders. Other VSS schemes (such as Pederson's scheme [18]) have similar difficulties.

We can verify that SHARES-VALID is true by taking advantage of the homomorphic properties of exponentiation. If we exponentiate both sides of Equation (3), we obtain the SHARES-VALID verification check:

$$g^k = \prod_{i \in \mathcal{B}} (g^{s_i})^{b_i} \tag{8}$$

Thus, if each j in $\mathcal{P}'$ receives g^k and g^{s_i} from all i in $\mathcal{B}$, they can verify that all s_i were valid shares of k. Each j accomplishes verification without learning s_i, given our assumption about discrete logs.

4.1 Assumptions about faulty shareholders

When we redistribute the secret k in $\mathbb{Z}_p$ from the scheme $\{\Gamma_{\mathcal{P}}, \mathbb{Z}_p, \{\mathbb{Z}_p\}, \psi_{\mathcal{P}}\}$ to the scheme $\{\Gamma_{\mathcal{P}'}, \mathbb{Z}_p, \{\mathbb{Z}_p\}, \psi_{\mathcal{P}'}\}$ with our VSR protocol, we assume at least m of the n shareholders in $\mathcal{P}$ and all n' of the shareholders in $\mathcal{P}'$ are non-faulty, and up to $n - m$ of the remaining shareholders in $\mathcal{P}$ may be faulty. We denote faulty shareholders, and the values they distribute, with over-bars. A non-faulty shareholder i in $\mathcal{P}$ distributes valid subshares $\hat{s}_{ij}$ of its share s_i to all shareholders j in $\mathcal{P}'$ and broadcasts g^k corresponding to k. A faulty shareholder $\bar{i}$ in $\mathcal{P}$ may distribute invalid subshares $\overline{\hat{s}_{\bar{i}j}}$ or broadcast $\overline{g^k}$ not corresponding to k.

We also assume we do not know which m of the n shareholders in $\mathcal{P}$ are non-faulty. Suppose we include a faulty shareholder $\bar{i}$ in our selection of $\mathcal{B}$ in $\Gamma_{\mathcal{P}}$ to participate in redistribution (REDISTRIBUTE in Figure 6). However, if $\bar{i}$ distributes $\overline{\hat{s}_{\bar{i}j}}$, one of the j will detect the presence of $\bar{i}$ since one of the verification checks in Equations (7) or (8) will fail. Alternatively, if $\bar{i}$ broadcasts $\overline{g^k}$, all j will detect the discrepancy when non-faulty old shareholders broadcast g^k. Thus, $\bar{i}$ must participate in the protocol without fault or risk detection. If we detect the presence of $\bar{i}$, we must restart redistribution with another set of m old shareholders. Unfortunately, we cannot identify $\bar{i}$ with our protocol.

The assumption that we do not know which m shareholders in $\mathcal{P}$ are non-faulty bounds the relative values of m and n. We assume we can detect discrepancies between $\overline{g^k}$ and g^k broadcast by faulty and non-faulty shareholders in $\mathcal{P}$ respectively. However, if we were to select a group of m faulty shareholders $\bar{i}$ inadvertently, then we would be unable to detect discrepancies if all $\bar{i}$ broadcast $\overline{g^k}$. We therefore require that $m > \frac{n}{2}$ so each authorized subset $\mathcal{B}$ in $\Gamma_{\mathcal{P}}$ has at least one non-faulty shareholder; if $m \leq \frac{n}{2}$, $n - m$ faulty shareholders in $\mathcal{P}$ could conspire to reconstruct k.

The requirement that all n' shareholders in $\mathcal{P}'$ are non-faulty is reasonable if we view the purpose of our VSR protocol as one of detecting faulty behavior by shareholders in $\mathcal{P}$. This is analogous to one of the assumptions underlying Feldman's VSS scheme, in which the shareholders are implicitly trusted to store valid shares (and reject invalid shares) of a secret.

4.2 Correctness

We prove that if the SHARES-VALID and SUBSHARES-VALID conditions are true after the share redistribution, then the new shareholders have valid shares of the original secret. We also show that Equations (7) and (8) check that the two conditions are true.

Lemma 1 *If the check in Equation (8) is true, then* SHARES-VALID *is true.*

PROOF: Assume the check in Equation (8) is true. It then follows that SHARES-VALID is true from Equation (3) and the homomorphic properties of exponentiation. □

Lemma 2 *If the check in Equation (7) is true, then* SUBSHARES-VALID *is true.*

PROOF: Proved by Feldman [9]. □

Theorem 1 (VSR theorem) *For Shamir's (m, n) threshold secret sharing scheme $\{\Gamma_{\mathcal{P}}, \mathbb{Z}_p, \{\mathbb{Z}_p\}, \psi_{\mathcal{P}}\}$ and the (m', n') scheme $\{\Gamma_{\mathcal{P}'}, \mathbb{Z}_p, \{\mathbb{Z}_p\}, \psi_{\mathcal{P}'}\}$, for all secrets $k \in \mathbb{Z}_p$, and for all authorized subsets $\mathcal{B} \in \Gamma_{\mathcal{P}}$, if* SHARES-VALID *and* SUBSHARES-VALID *are true after the execution of the* REDISTRIBUTION *step (Figure 6) of the VSR protocol, then all shareholders j in all authorized subsets $\mathcal{B}' \in \Gamma_{\mathcal{P}'}$ hold valid shares of k.*

PROOF: Assume both SHARES-VALID and SUBSHARES-VALID are true. Then:

$$
\begin{aligned}
k &= \sum_{i \in \mathcal{B}} b_i s_i \qquad (\text{SHARES-VALID}) \\
&= \sum_{i \in \mathcal{B}} \left(b_i \sum_{j \in \mathcal{B}'} b'_j \hat{s}_{ij} \right) \qquad (\text{SUBSHARES-VALID}) \\
&= \sum_{i \in \mathcal{B}} \sum_{j \in \mathcal{B}'} b_i b'_j \hat{s}_{ij} \qquad (x(y+z) = xy + xz) \\
&= \sum_{i \in \mathcal{B}} \sum_{j \in \mathcal{B}'} b'_j b_i \hat{s}_{ij} \qquad (xy = yx) \\
&= \sum_{j \in \mathcal{B}'} \sum_{i \in \mathcal{B}} b'_j b_i \hat{s}_{ij} \qquad (x + y = y + x) \\
&= \sum_{j \in \mathcal{B}'} \left(b'_j \sum_{i \in \mathcal{B}} b_i \hat{s}_{ij} \right) \qquad (xy + xz = x(y+z)) \\
&= \sum_{j \in \mathcal{B}'} b'_j s'_j \qquad (\text{Equation (3)})
\end{aligned}
$$

□

5 Summary and future work

We have presented a protocol for the verifiable redistribution of secrets distributed with Shamir's secret sharing scheme [22]. We have proven that new shareholders have valid shares after redistribution if the SHARES-VALID and SUBSHARES-VALID conditions are true, and have given the corresponding verification checks. We have shown that our protocol guards against faulty behavior by up to $n - m$ of the old shareholders provided that $m > \frac{n}{2}$. In our presentation, we have assumed that the computation of discrete logs in a finite field is intractable, and that there exist reliable broadcast communication channels among all participants and private channels between every pair of participants.

As part of our future work, we will investigate ways to identify faulty old shareholders during redistribution, and to relax the bounds on the number of non-faulty new shareholders. We also plan to implement our protocol to evaluate its performance costs over non-verifiable redistribution protocols.

6 Acknowledgements

We would like to thank Michael Reiter and Chenxi Wang for their technical input and support.

References

[1] J. C. Benaloh. Secret sharing homomorphisms: Keeping shares of a secret secret. In A. M. Odlyzko, editor, *Proc. of CRYPTO 1986, the 6th Ann. Intl. Cryptology Conf.*, volume 263 of *Lecture Notes in Computer Science*, pages 213–222. Intl. Assoc. for Cryptologic Research, Springer-Verlag, 1987.

[2] G. R. Blakely. Safeguarding cryptographic keys. In *Proc. of the Natl. Computer Conf.*, volume 48 of *American Federation of Information Processing Societies Proceedings*, 1979.

[3] B. Blakley, G. R. Blakley, A. H. Chan, and J. L. Massey. Threshold schemes with disenrollment. In E. F. Brickell, editor, *Proc. of CRYPTO 1992, the 12th Ann. Intl. Cryptology Conf.*, volume 740 of *Lecture Notes in Computer Science*, pages 540–548. Intl. Assoc. for Cryptologic Research, Springer-Verlag, August 1992.

[4] C. Blundo, A. Cresti, A. D. Santis, and U. Vaccaro. Fully dynamic secret sharing schemes. *Theoretical Computer Science*, 165(2):407–440, October 1996.

[5] C. Cachin. On-line secret sharing. In C. Boyd, editor, *Proc. of the 5th IMA Conf. on Cryptography and Coding*, volume 1025 of *Lecture Notes in Computer Science*, pages 90–198. The Inst. of Mathematics and its Applications, Springer-Verlag, December 1995.

[6] B. Chor, S. Goldwasser, S. Micali, and B. Awerbuch. Verifiable secret sharing and achieving simultaneity in the presence of faults (Extended abstract). In *Proc. of the 26th IEEE Ann. Symp. on Foundations of Computer Science*, pages 383–395. IEEE, October 1985.

[7] Y. Desmedt. Some recent research aspects of threshold cryptography. In E. Okamoto, G. Davida, and M. Mambo, editors, *Proc. of the 1st Intl. Information Security Workshop*, volume 1396 of *Lecture Notes in Computer Science*, pages 158–173. Springer-Verlag, September 1997.

[8] Y. Desmedt and S. Jajodia. Redistributing secret shares to new access structures and its applications. Technical Report ISSE TR-97-01, George Mason University, Fairfax, VA, July 1997.

[9] P. Feldman. A practical scheme for non-interactive verifiable secret sharing. In *Proc. of the 28th IEEE Ann. Symp. on Foundations of Computer Science*, pages 427–437. IEEE, October 1987.

[10] Y. Frankel, P. Gemmell, P. D. MacKenzie, and M. Yung. Optimal resilience proactive public-key cryptosystems. In *Proc. of the 38th IEEE Ann. Symp. on Foundations of Computer Science*, pages 384–393. IEEE, October 1997.

[11] Y. Frankel, P. Gemmell, P. D. MacKenzie, and M. Yung. Proactive RSA. In B. S. Kaliski Jr, editor, *Proc. of CRYPTO 1997, the 17th Ann. Intl. Cryptology Conf.*, volume 1294 of *Lecture Notes in Computer Science*, pages 440–454. Intl. Assoc. for Cryptologic Research, Springer-Verlag, August 1997.

[12] R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin. Robust threshold DSS signatures. In U. M. Maurer, editor, *Proc. of EUROCRYPT 1996, the Intl. Conf. on the Theory and Application of Cryptographic Techniques*, volume 1070 of *Lecture Notes in Computer Science*, pages 354–371. Intl. Assoc. for Cryptologic Research, Springer-Verlag, May 1996.

[13] R. Gennaro and S. Micali. Verifiable secret sharing as secure computation. In L. C. Guillou and J.-J. Quisquater, editors, *Proc. of EUROCRYPT 1995, the Intl. Conf. on the Theory and Application of Cryptographic Techniques*, volume 921 of *Lecture Notes in Computer Science*, pages 168–182. Intl. Assoc. for Cryptologic Research, Springer-Verlag, May 1995.

[14] O. Goldreich, S. Micali, and A. Wigderson. How to prove all NP statements in zero-knowledge and a methodology of cryptograhpic protocol design. In A. M. Odlyzko, editor, *Proc. of CRYPTO 1986, the 6th Ann. Intl. Cryptology Conf.*, volume 263 of *Lecture Notes in Computer Science*, pages 171–185. Intl. Assoc. for Cryptologic Research, Springer-Verlag, 1987.

[15] A. Herzberg, M. Jakobsson, S. Jarecki, H. Krawczyk, and M. Yung. Proactive public key and signature systems. In *Proc. of the 4th ACM Intl. Conf. on Computer and Communications Security*, pages 100–110. Assoc. for Computing Machinery, April 1997.

[16] A. Herzberg, S. Jarekci, H. Krawczyk, and M. Yung. Proactive secret sharing or: How to cope with perpetual leakage. In D. Coppersmith, editor, *Proc. of CRYPTO 1995, the 15th Ann. Intl. Cryptology Conf.*, volume 963 of *Lecture Notes in Computer Science*, pages 339–352. Intl. Assoc. for Cryptologic Research, Springer-Verlag, August 1995.

[17] R. Ostrovsky and M. Yung. How to withstand mobile virus attacks. In *Proc. of the 10th Ann. ACM Symp. on Principles of Distributed Computing*, pages 51–59. ACM SIGACT and ACM SIGOPS, August 1991.

[18] T. P. Pederson. Non-iteractive and information-theoretic secure verifiable secret sharing. In J. Feigenbaum, editor, *Proc. of CRYPTO 1991, the 11th Ann. Intl. Cryptology Conf.*, volume 576 of *Lecture Notes in Computer Science*, pages 129–140. Intl. Assoc. for Cryptologic Research, Springer-Verlag, August 1991.

[19] T. Rabin. Robust sharing of secrets when the dealer is honest or cheating. *Journal of the ACM*, 41(6):1089–1109, November 1994.

[20] T. Rabin. A simplified approach to threshold and proactive RSA. In H. Krawczyk, editor, *Proc. of CRYPTO 1998, the 18th Ann. Intl. Cryptology Conf.*, volume 1462 of *Lecture Notes in Computer Science*, pages 89–104. Intl. Assoc. for Cryptologic Research, Springer-Verlag, August 1998.

[21] T. Rabin and M. Ben-Or. Verifiable secret sharing and multiparty protocols with honest majority. In *Proc. of the 21st Symp. on the Theory of Computing*, pages 73–85. Assoc. for Computing Machinery, May 1989.

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

[23] J. J. Wylie, M. Bakkaloglu, V. Pandurangan, M. W. Bigrigg, S. Oguz, K. Tew, C. Williams, G. R. Ganger, and P. K. Khosla. Selecting the right data distribution scheme for a survivable storage system. Technical Report CMU-CS-01-120, Sch. of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213, May 2001.

[24] J. J. Wylie, M. W. Bigrigg, J. D. Strunk, G. R. Ganger, H. Kiliççöte, and P. K. Khosla. Survivable information storage systems. *IEEE Computer*, pages 61–68, August 2000.

[25] L. Zhou, F. B. Schneider, and R. van Renesse. COCA: A secure distributed on-line certification authority. Technical Report TR2000-1828, Dept. of Computer Science, Cornell University, Ithaca, NY 14853, December 2000.