Verifiable Secret Redistribution for Threshold Sharing Schemes

Abstract

We present a new protocol for the verifiable redistribution of secrets from (m,n) to (m',n') access structures for threshold sharing schemes. Our protocol enables the addition or removal of shareholders and also guards against mobile adversaries that cause permanent damage. We observe that existing protocols either cannot be readily extended to allow redistribution between different access structures, or have vulnerabilities that allow faulty old shareholders to corrupt the shares of new shareholders. Our primary contribution is that, in our protocol, new shareholders can verify the validity of their shares after redistribution between different access structures.

Keywords: threshold secret sharing, verification, proactive security.

1 Introduction

Threshold sharing schemes provide fundamental building blocks for the safeguarding of secrets and secure distributed computation. Since its invention, many enhancements to threshold schemes have been proposed. *Proactive secret sharing* (PSS) schemes [FGMY97a, FGMY97b, GJKR96, HJJ$^+$97, Rab98], for example, provide enhanced protection against *mobile adversaries* [OY91] by updating the shares periodically in a distributed fashion. In general, PSS schemes retain the same shareholders and access structure across updates. A more general proactive problem is the redistribution of shares between different (possibly disjoint) sets of shareholders and different access structures, hereafter referred to as *secret redistribution*. Secret redistribution has been studied by Desmedt and Jajodia [DJ97] and Frankel *et al.* [FGMY97a]. In this paper, we identify weaknesses in previous work, and propose a new protocol that performs *verifiable secret redistribution* (VSR) between different shareholders and access structures. We prove the security of our scheme with an information-theoretic security proof.

The development of our new protocol is motivated by work on a secure, distributed storage system [XXX] that stores shares of files (or long-term encryption keys) on a distributed set of servers. For system management and security purposes (such as load balancing or server compromises), the system needs to generate new shares and invalidate old shares. In general, the ability to redistribute shares of secrets between different sets of shareholders is useful for a wide range of applications. Consider the following examples:

Multiparty signature schemes: Business organizations may use digital signature schemes to sign legal documents they exchange with counterparties. Such schemes are typically asymmetric: an organization generates signatures with a *private* key known only to itself, and the counterparties verify signatures with a corresponding *public* key. To prevent a single rogue agent from signing documents without proper authorization, the organization may require multiple agents to generate signatures with a multiparty signature scheme [FGMY97a, FGMY97b, GJKR96, HJJ$^+$97, Rab98] that distributes shares of the private key to the agents. Over time, the organization will need to give shares of the private key to agents who join, and invalidate the shares of agents who leave. Changing the private key each time agents join or leave would require revocation of the well-known public key. A better solution would be to redistribute shares of the private key in a way that invalidates old shares and obviates the need for public key revocation.

Distributed key servers: Recent distributed storage systems, such as CFS [DKK$^+$01], FarSite [BDET00], PASIS [WBS$^+$00, WBP$^+$01] and PAST [RD01], use disk space on (potentially) untrusted storage devices to store data. Clients may encrypt data before handing it off to the storage system. One way for clients to store their encryption keys is to employ threshold sharing schemes to distribute shares of the keys to a set of *key servers*. Of course, since clients must store keys for as long as they store the encrypted data, a mobile adversary may have a large window of opportunity to compromise multiple key servers, and thus obtain enough shares to reconstruct the keys. To counter the adversary, the uncompromised key servers could periodically redistribute shares of the keys to new, uncompromised servers. The adversary would then need to restart the process of compromising servers, assuming that old shares cannot be combined with new shares to reconstruct the secret.

Both of these applications must support dynamic shareholder membership, and protect secrets from mobile adversaries. In the multiparty signature system, agents may join or leave the organization, while in the storage system, key servers may be added or removed for maintenance or security purposes. It may also be advantageous to change the threshold value of the underlying sharing scheme to accommodate new policies. In both applications, the system needs to retain the original secrets when generating new shares and invalidating old shares. More importantly, to prevent faulty old shareholders from corrupting the shares of new shareholders, new shareholders must be able to verify the *validity* of their shares after redistribution (i.e., that their shares can be used to reconstruct the secret).

Desmedt and Jajodia propose a protocol to redistribute secret shares between different (possibly disjoint) sets of shareholders with different access structures [DJ97]. They postulate that a straightforward extension of their protocol with a *verifiable secret sharing* (VSS) scheme allows them to tolerate faulty old shareholders and verify the validity of new shares. We show that such a naïve extension fails, since it still allows faulty old shareholders to corrupt the shares of new shareholders.

Frankel *et al.* propose a proactive threshold sharing scheme for RSA [FGMY97a] that uses a *poly-to-sum* redistribution from a polynomial sharing scheme to an additive sharing scheme, and a *sum-to-poly* redistribution from the additive scheme back to a polynomial scheme. They suggest that changes in threshold value and number of shareholders can be accommodated in the poly-to-sum redistribution. However, their scheme relies on public information distributed in the preceding round to verify the validity of new shares. If secret redistribution is performed among the same set of shareholders, verification can be achieved because all shareholders retain the information from the preceding round. However, if redistribution is performed to new shareholders who do not possess the necessary public information, faulty old shareholders could corrupt redistribution. We will discuss this point further in Section 4.

Our key observations are that:

- PSS schemes cannot be readily extended to allow "updates" between different sets of shareholders with different access structures. Thus, these schemes cannot accommodate the permanent addition or removal of shareholders.

- Redistribution protocols have vulnerabilities that allow faulty old shareholders to corrupt redistribution and cause new shareholders to generate invalid shares.

- For verification purposes, old shareholders in a secret redistibution protocol must pass additional information to new shareholders. This information can be a commitment to the original secret, or commitments to the shares of all old shareholders.

- Pinpoint identification and elimination of faulty old shareholders are not immediately possible if redistribution is to occur between two disjoint sets of shareholders. In the worst case, for redistribution

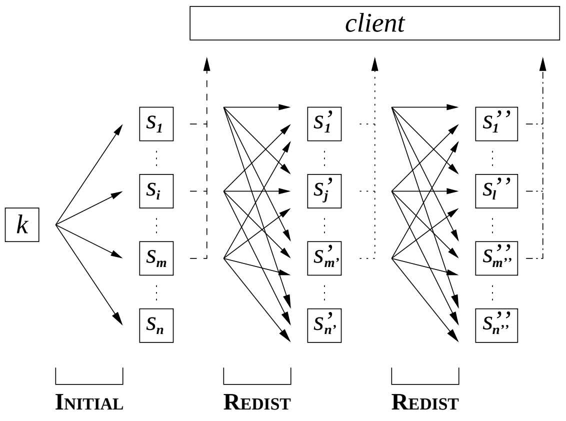

Figure 1: Initial threshold scheme distribution of a secret k with an (m,n) access structure, followed by redistribution to an (m',n') access structure. The INITIAL phase of our VSR protocol guarantees that the shares s_1 ... s_n are valid. The REDIST phase of our protocol guarantees that the shares s'_1 ... $s'_{n'}$ are valid. The dashed (dotted) lines represent a client contacting servers holding s_1 ... s_m (s'_1 ... $s'_{m'}$). We can execute REDIST an arbitrary number of times.

from an (m,n) access structure, $\sum_{i=1}^{m-1} \binom{m}{i} \binom{n-m+1}{m-i}$ restarts are required to eliminate faulty shareholders and complete redistribution.

We present a new verifiable secret redistribution protocol for Shamir's threshold sharing scheme [Sha79] in which we redistribute secrets from an (m,n) to (m',n') access structure. We base our protocol on Desmedt and Jajodia's redistribution protocol, in which new shareholders generate shares from *subshares* of old shares. We extend their protocol to enable new shareholders to verify the validity of the shares they generate. We prove that the new shareholders can generate valid new shares if they can both verify the validity of the old shares and that of the subshares. We also prove that an adversary who obtains less than m old shares and less than m' new shares cannot reconstruct the secret.

We summarize the operation of our VSR protocol in Figure 1. Returning to our example applications, suppose that we have distributed shares of a key, k, to n shareholders, as shown in the INITIAL phase. A counterparty wishing to obtain the signature for a document, or a client wishing to retrieve an encryption key, can do so by contacting m of the n shareholders (the dashed lines). When agents join or leave, or when key servers are added or taken offline, our VSR protocol redistributes k to a new set of shareholders, as shown in the REDIST phases. Upon the completion of redistribution, a client can perform the same distributed operations by contacting m' of the n' new servers (the dotted lines). The applications can execute the REDIST phase as often as necessary to ensure the security and availability of the shared secrets.

2 Related work

Blakley and Shamir invented threshold sharing schemes independently [Bla79, Sha79]. In Blakley's scheme, the intersection of m of n vector spaces yields a one-dimensional vector that corresponds to the secret. In Shamir's scheme, the interpolation of an $m-1$ degree polynomial through m of n points yields a constant term in the polynomial that corresponds to the secret. Desmedt surveys other sharing schemes [Des97].

Chor *et al.* present a VSS scheme in which the dealer and shareholders perform an interactive secure distributed computation [CGMA85]. Benaloh [Ben87], Gennaro and Micali [GJKR96, GM95], Goldreich *et al.* [GMW87], and Rabin and Ben-Or [Rab94, RBO89] propose schemes in which the dealer and shareholders participate in an interactive zero-knowledge proof of validity; the scheme of Gennaro and Micali, and that of Rabin and Ben-Or, is information-theoretically secure. Feldman and Pedersen [Fel87, Ped91] present

VSS schemes in which the dealer broadcasts a non-interactive zero-knowledge proof to the shareholders. Beth *et al.* [BKO93] present a VSS scheme for monotone access structures based on finite geometries. Our VSR protocol differs from previous VSS schemes in that the multiple "dealers" of the new shares (the old shareholders) do not have the secret, and must use other information to generate a proof for the new shareholders. Also, each new shareholder verifies the validity of the subshares distributed by the old shareholders, and verifies the validity of the shares used by the old shareholders to generate the subshares.

Frankel *et al.* [FGMY97b, FMY99, FMY01] and Rabin [Rab98] propose threshold PSS schemes in which each shareholder periodically distributes a subshare of its share to all the other members. Each shareholder then combines the subshares to generate a new share. A drawback of these protocols is that the shareholders rely on commitments received during the initial distribution of the secret to verify the validity of the new shares, and thus one cannot redistribute between disjoint sets of shareholders. Also, the commitments depend on (m,n), and thus one cannot redistribute between different access structures.

Desmedt and Jajodia present a secret redistribution protocol that does not require the intermediate reconstruction of the original secret [DJ97]. We present the details of their protocol in Section 3.2. Their protocol allows redistribution between different (possibly disjoint) sets of shareholders with different access structures. Unfortunately, a faulty old shareholder can undetectably distribute "subshares" of some random value instead of subshares of a valid old share, and thus cause new shareholders to generate invalid shares.

Frankel *et al.* propose a proactive threshold sharing scheme for RSA private keys [FGMY97a]. The protocol uses a poly-to-sum redistribution from an (m,n) to (m,m) sharing scheme, and a sum-to-poly redistribution back to an (m,n) scheme. During redistribution, each old shareholder broadcasts a commitment to its share, which new shareholders use to verify the validity of their generated share. Unfortunately, during redistribution to a disjoint set of shareholders, it is not enough for the old shareholders to broadcast the commitment to their respective shares, since a faulty shareholder can broadcast a random "commitment." There are two potential remedies for this problem. One is for the old shareholders to broadcast a commitment to the original secret, which can be used to verify the consistency of commitments to shares. The alternative is for each old shareholder to keep and broadcast all share commitments. We opt for the former in our protocol because it is both space and time efficient.

Other researchers present secret redistribution protocols that do not involve the physical redistribution of shares. Blakley *et al.* consider threshold schemes that *disenroll* (remove) shareholders from the access structure with broadcast messages [BBCM92]; the new shareholders are a subset of the old ones. Cachin proposes a secret sharing scheme that *enrolls* (adds) shareholders in the access structure after the initial sharing [Cac95]; the new shareholders are a superset of the old ones. Blundo *et al.* presents a scheme in which the dealer uses broadcast messages to activate different, possibly disjoint, authorized subsets [BCSV96]. 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 alters the access structure by physical redistribution of shares, and allows new shareholders to verify that they have valid shares.

Ostrovsky and Yung introduce the concept of mobile adversaries [OY91] that corrupt participants in a distributed protocol at a constant rate. Canetti and Herzberg use mobile adversaries to motivate their development of a distributed proactive pseudorandom number generator [CH94]. Herzberg *et al.* [HJKY95, HJJ$^+$97] propose a PSS scheme for Shamir's sharing scheme [Sha79] in which each shareholder periodically distributes *update shares* to all other shareholders. Zhou, Schneider, and van Renesse propose a PSS scheme for asynchronous, wide-area networks, and employ it in an on-line certification authority [ZSvR00]. Our VSR protocol, unlike these PSS schemes, can redistribute shares to arbitrary access structures. However, we assume that there exist reliable broadcast channels among all participants and private channels between every pair of participants in our protocol, which Zhou *et al.* avoid in their asynchronous protocol.

We note that our VSR protocol, in contrast to the earlier threshold PSS schemes, can guard against mobile adversaries that cause permanent damage (i.e., that cannot be undone with a reboot operation). Of course, we still require that at any given point of time, the number of faulty shareholders in the current set of shareholders is less than the threshold value.

3 Cryptographic building blocks

In this section, we outline the cryptographic protocols that form the building blocks for our VSR protocol. We first recap Shamir's threshold sharing scheme [Sha79], and then summarize Desmedt and Jajodia's secret redistribution protocol [DJ97] and Feldman's VSS scheme [Fel87].

3.1 Shamir's threshold sharing scheme

Shamir's threshold sharing scheme is based on polynomial interpolation [Sha79]. A secret k is in $\mathbb{Z}_p$, where p is prime and $p > n$; shares of k are also in $\mathbb{Z}_p$. Authorized subsets, A, of the set of shareholders, P, are in the access structure $\mathcal{A}_P^{(m,n)}$, where $|P| = n$ and $|A| = m$.

To distribute k to the access structure, $\mathcal{A}_P^{(m,n)}$, we select an $m-1$ degree polynomial $a(x)$ with constant term k and random coefficients a_1 ... $a_{m-1} \in \mathbb{Z}_p$, and generate shares s_i for each shareholder $i \in P$:

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

To reconstruct k, we retrieve m pairs (i, s_i) from $i \in A$, and compute k by Lagrange interpolation:

$$k = \sum_{i \in A} b_i s_i \quad \text{where} \quad b_i = \prod_{j \in A \setminus \{i\}} \frac{j}{(j-i)} \tag{2}$$

3.2 Desmedt and Jajodia's secret redistribution protocol

Desmedt and Jajodia present a protocol for the redistribution of shares of secrets distributed with threshold sharing schemes, which does not require the intermediate reconstruction of the secret [DJ97]. We present a specialization of their protocol for Shamir's scheme in Appendix B. Suppose we have distributed a secret k to the access structure $\mathcal{A}_P^{(m,n)}$, and wish to redistribute k to the new access structure $\mathcal{A}_{P'}^{(m',n')}$. To achieve this, we select an authorized subset $A \in \mathcal{A}_P^{(m,n)}$. Each shareholder $i \in A$ uses Shamir's scheme to distribute subshares $\hat{s}_{ij}$ of its share s_i to $\mathcal{A}_{P'}^{(m',n')}$. Each new shareholder $j \in P'$ receives $\hat{s}_{ij}$ from each i, and generates a new share s'_j by Lagrange interpolation:

$$s'_j = \sum_{i \in A} b_i \hat{s}_{ij} \quad \text{where} \quad b_i = \prod_{x \in A \setminus \{i\}} \frac{x}{(x-i)} \tag{3}$$

3.3 Feldman's VSS scheme

Feldman presents a VSS scheme for shareholders of a secret to verify the validity of their shares [Fel87]. We present a specialization for Shamir's scheme in Appendix C. Herzberg *et al.* present a similar treatment [HJKY95].

The application of Feldman's VSS scheme to Shamir's scheme takes advantage of the homomorphic properties of exponentiation, and of the assumption that the computation of discrete logs in a finite field is intractable. Suppose we have field $\mathbb{Z}_p$ and ring $\mathbb{Z}_r^*$, such that p and r are prime and $r = pq + 1$ (where q is a non-negative integer), and suppose we have a generator g for $\mathbb{Z}_r^*$. We first use Shamir's scheme with polynomial $a(x)$ to distribute a secret $k \in \mathbb{Z}_p$ to the access structure $\mathcal{A}_P^{(m,n)}$. Then, in addition to sending

Verifiable Secret Redistribution protocol for Shamir's sharing scheme:
INITIAL: To distribute a secret $k \in \mathbb{Z}_p$ to the access structure $\mathcal{A}_P^{(m,n)}$:

1. Use the polynomial $a(i) = k + a_1 i + \ldots + a_{m-1} i^{m-1}$ to compute the shares s_i of k, and send s_i to the corresponding $i \in P$ over private channels.
2. Use generator g to compute $g^k, g^{a_1} \ldots g^{a_{m-1}}$, and send them to all $i \in P$ over the broadcast channel.
3. For each $i \in P$, verify that:

$$g^{s_i} = g^k \prod_{l=1}^{m-1} (g^{a_l})^{i^l}$$

   If the condition holds, i broadcasts a "commit" message. Otherwise, i broadcasts an "abort" message.
4. If all $i \in P$ agree to commit, each i stores s_i and g^k. Otherwise, they abort the protocol.

REDIST: To redistribute $k \in \mathbb{Z}_p$ from an $\mathcal{A}_P^{(m,n)}$ to $\mathcal{A}_{P'}^{(m',n')}$ access structure, using the authorized subset $A \in \mathcal{A}_P^{(m,n)}$:

1. For each $i \in A$, use the polynomial $a'_i(j) = s_i + a'_{i1} j + \ldots + a'_{i(m'-1)} j^{m'-1}$ to compute the subshares $\hat{s}_{ij}$ of s_i, and send $\hat{s}_{ij}$ to the corresponding $j \in P'$ over private channels.
2. For each $i \in A$, use g to compute $g^{s_i}, g^{a'_{i1}} \ldots g^{a'_{i(m'-1)}}$, and send them and g^k to all $j \in P'$ over the broadcast channel.
3. For each $j \in P'$, verify that:

$$\forall i \in A : g^{\hat{s}_{ij}} = g^{s_i} \prod_{l=1}^{m'-1} (g^{a'_{il}})^{j^l}$$

   and:

$$g^k = \prod_{i \in A} (g^{s_i})^{b_i} \quad \text{where} \quad b_i = \prod_{l \in A \setminus \{i\}} \frac{l}{(l-i)}$$

   If the conditions hold, j broadcasts a "commit" message. Otherwise, j broadcasts an "abort" message.
4. If all $j \in P'$ agree to commit, each j generates a new share s'_j:

$$s'_j = \sum_{i \in A} b_i \hat{s}_{ij} \quad \text{where} \quad b_i = \prod_{l \in A \setminus \{i\}} \frac{l}{(l-i)}$$

   and stores s'_j and g^k. Otherwise, they abort the protocol.

Figure 2: Protocol for the verifiable redistribution of shares of a secret from an $\mathcal{A}_P^{(m,n)}$ to $\mathcal{A}_{P'}^{(m',n')}$ access structure, for Shamir's threshold sharing scheme.

the shares $s_i \in \mathbb{Z}_p$ to shareholders $i \in P$, we broadcast commitments to k and the coefficients a_1 ... a_{m-1} of $a(x)$ of the form g^k and g_1^a ... $g^{a_{m-1}}$. Each i may then verify that s_i is a valid share of k:

$$g^{s_i} = g^k (g^{a_1})^i \ldots (g^{a_{m-1}})^{i^{m-1}} \tag{4}$$

which is the exponentiation of $a(x)$ (Equation (1)). Assuming that the computation of discrete logs is intractable, no i can learn k or a_1 ... a_{m-1} from the commitments.

4 The VSR protocol

We present our verifiable secret redistribution protocol for secrets distributed with Shamir's scheme. The protocol receives shares of a secret distributed to the access structure $\mathcal{A}_P^{(m,n)}$, and outputs shares of the secret distributed to a new access structure $\mathcal{A}_{P'}^{(m',n')}$. We assume that the computation of discrete logs in a finite field is intractable, and that there exist reliable broadcast channels among all participants and private channels between every pair of participants. We also assume that there are at least m non-faulty old shareholders, at most $m-1$ faulty old shareholders, and n' non-faulty new shareholders.

In the initial distribution phase (INITIAL in Figure 2), the dealer of secret k distributes shares s_i to each shareholder $i \in P$ with the polynomial $a(i)$ (INITIAL step 1). The dealer also broadcasts commitments g^k and g^{a_1} ... $g^{a_{m-1}}$, which each i uses to verify the validity of s_i (Equation (4), INITIAL steps 2 and 3). If verification passes, i stores s_i and g^k (INITIAL step 4).

In the redistribution phase (REDIST in Figure 2), each i in an authorized subset $A \in \mathcal{A}_P^{(m,n)}$ uses Shamir's scheme (with the polynomial $a'_i(j)$) to distribute subshares $\hat{s}_{ij}$ of its share s_i to $\mathcal{A}_{P'}^{(m',n')}$ (REDIST step 1). Each shareholder $j \in P'$ receives $\hat{s}_{ij}$ from each i, and generates a new share s'_j (Equation (3), REDIST step 4). We may redistribute k 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, we require that two conditions, SHARES-VALID and SUBSHARES-VALID, hold. When all $i \in A$ redistribute s_i to each $j \in P'$, all s_j are valid shares of k if

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

SUBSHARES-VALID:
$\forall i \in A, A' \in \mathcal{A}_{P'}^{(m',n')} : s_i = \sum_{j \in A'} b'_j \hat{s}_{ij}$

We define a NEW-SHARES-VALID condition, which holds if new shareholders have valid shares of the secret. We prove in Section 4.5 that NEW-SHARES-VALID holds if SHARES-VALID and SUBSHARES-VALID hold. The definition of NEW-SHARES-VALID follows from Equation (2) for a secret distributed to $\mathcal{A}_{P'}^{(m',n')}$:

NEW-SHARES-VALID:
$\forall A' \in \mathcal{A}_{P'}^{(m',n')} : k = \sum_{j \in A'} b'_j s'_j$

We use Feldman's VSS scheme [Fel87] to verify that SUBSHARES-VALID holds. Each $i \in A$ broadcasts commitments to its share and the coefficients of $a'_i(j)$ (g^{s_i} and $g^{a_{i1}}$ ... $g^{a_{i(m-1)}}$), which each j uses to verify the validity of $\hat{s}_{ij}$ (REDIST step 2).

To allow the new shareholders to verify that SHARES-VALID holds, which together with SUBSHARES-VALID verifies that NEW-SHARES-VALID holds, the old shareholders in our protocol broadcast a commitment to the original secret. Each $i \in A$ therefore stores g^k (received during INITIAL) and later broadcasts it to all $j \in P'$. Recall that each j receives g^{s_i} from each i to verify that SUBSHARES-VALID holds. Once each j receives g^k, it verifies that s_i is a valid share of k:

$$g^k = \prod_{i \in A} g^{b_i s_i} \tag{5}$$

Equation (5) follows from Equation (2) and the homomorphic properties of exponentiation. Assuming that the computation of discrete logs is intractable, no j can learn k from g^k.

4.1 Discussion

The key insight in our VSR protocol is that a naïve extension of Desmedt and Jajodia's protocol with Feldman's VSS scheme [DJ97, Fel87] does not in itself allow the new shareholders to verify that NEW-SHARES-VALID holds. The difficulty arises because the VSS scheme only verifies that SUBSHARES-VALID holds, which in the absence of SHARES-VALID is insufficient to verify that NEW-SHARES-VALID holds. Although Desmedt and Jajodia claim that the linear properties of their protocol and the VSS scheme ensure that each new shareholder j generates valid shares, they implicitly assume that each shareholder $i \in A$ distributes subshares of valid share s_i. The VSS scheme only allows i to prove that it distributed valid subshares of some value. However, i may have distributed "subshares" of some random value instead of subshares of s_i. The same difficulty exists if one extends Desmedt and Jajodia's protocol with Pedersen's VSS scheme [Ped91] in the same simple manner.

Our insight also applies to the proactive scheme presented by Frankel *et al.* [FGMY97a]. Their verification checks ensure that both SUBSHARES-VALID and SHARES-VALID hold during redistribution to the same set of shareholders. However, during redistribution to new shareholders, their checks only ensure that SUBSHARES-VALID holds. Their "proper secret" check does not ensure that SHARES-VALID holds because it relies on a "witness" ($g^{s_i L^2}$ in their paper) computed from information distributed in the preceding round. A faulty shareholder can thus distribute spurious information to the new shareholders and ultimately cause them to accept a false witness value.

To allow new shareholders to verify that both SHARES-VALID and SUBSHARES-VALID hold, which are sufficient to guarantee that NEW-SHARES-VALID holds, additional information tying the shares back to the original secret must be passed to the new shareholders. In our protocol, this information is the commitment to the original secret, g^k. Each old shareholder participating in the redistribution broadcasts g^k to the new shareholders. Then g^k is used to check that SHARES-VALID holds (Equation (5)).

We could augment Frankel's PSS scheme in the same way. Each old shareholder could pass a commitment to the original private key, g^d, to the new shareholders, who then verify that

$$g^d \equiv g^P \prod_{i \in \Lambda} g^{s_i z_{i,\Lambda}} \pmod{n}$$

holds, where s_i are shares, and P, $z_{i,\Lambda}$ are publicly computable (see page 5 of their paper).

As an alternative to broadcasting the commitment to the original secret, g^k, each shareholder could retain and broadcast the commitments to all shares, g^{s_1} ... g^{s_m}. This would also allow new shareholders to verify that SHARES-VALID holds. Any discrepancy in the commitment values would indicate the presence of faulty shareholders. We choose to use g^k for efficiency reasons.

4.2 Detecting faulty shareholders

During redistribution from an $\mathcal{A}_P^{(m,n)}$ to $\mathcal{A}_{P'}^{(m',n')}$ access structure with our VSR protocol, we assume that at least m of the n shareholders in P and all n' of the shareholders in P' are non-faulty, and that up to $m-1$ shareholders in P may be faulty. We denote faulty shareholders, and the values they distribute, with over-bars. A non-faulty shareholder $i \in P$ distributes valid subshares $\hat{s}_{ij}$ of its share s_i to all shareholders $j \in P'$ and broadcasts g^k corresponding to secret $k \in \mathbb{Z}_p$. A faulty shareholder $\bar{i} \in P$ may distribute invalid subshares $\overline{\hat{s}_{\bar{i}j}}$ or broadcast $\overline{g^k}$ not corresponding to k.

In order to check that the verification conditions hold, we require that certain information be made available to the new shareholders. In the redistribution protocol of Desmedt and Jajodia [DJ97], this information is commitments g^k, g^{s_i}, and $g^{a_{i1}}$ ... $g^{a_{i(m-1)}}$. In the PSS scheme of Frankel *et al.* [FGMY97a], this information is the value $g^{s_i L^2}$ and g^d. In the absence of a trusted information repository, the new members must rely on the old shareholders to deliver this information. It is this process that proves to be problematic for the pinpoint identification of faulty shareholders.

Consider redistribution from $\mathcal{A}_P^{(m,n)}$ to $\mathcal{A}_{P'}^{(m',n')}$. Assume that we start with a random authorized subset $A \in \mathcal{A}_P^{(m,n)}$, and recall that $|A| = m$. It is possible that some subset of the old shareholders in A (at most $m-1$) are faulty, and will attempt to broadcast $\overline{g^k}$ and $\overline{\hat{s}_{\bar{i}j}}$. If the faulty shareholders conspire to broadcast the same $\overline{g^k}$, the new shareholders will detect the discrepancy in the m broadcast values, but cannot pinpoint the faulty shareholders. The new shareholders cannot use majority voting since the majority of the old shareholders in A may be faulty.

Assuming that up to m–1 shareholders may be faulty, any randomly selected authorized subset of m old shareholders must contain at least one non-faulty shareholder. If the new shareholders detect discrepancies in the commitments broadcast by the old shareholders, they can restart the redistribution protocol with another authorized subset until all values are consistent and all verification conditions hold. For $\mathcal{A}_P^{(m,n)}$, the number of times we must restart the redistribution protocol is bounded in the worst case by

$$\binom{n}{m} - \binom{n-m+1}{m} = \sum_{i=1}^{m-1} \binom{m}{i} \binom{n-m+1}{m-i} \tag{6}$$

which is simply the number of sets of size m containing at least one faulty shareholder.

The requirement that all n' shareholders in P' are non-faulty is reasonable if we view the purpose of our VSR protocol as one of detecting faulty behavior by shareholders in 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.

4.3 Computational cost

The computational cost for each new shareholder of verification in our VSR protocol (REDIST Step 3 in Figure 2) is $\mathrm{O}(mm')$ multiplications and $\mathrm{O}(mm')$ exponentiations, exclusive of the cost of computing the commitments. Consider redistribution from an $\mathcal{A}_P^{(m,n)}$ to $\mathcal{A}_{P'}^{(m',n')}$ access structure. Each new shareholder $j \in P'$ performs m–1 multiplications ($A \in \mathcal{A}_P^{(m,n)}$; $|A| = m$) and m exponentiations to verify that SHARES-VALID holds (Equation (5)), for a total cost of $\mathrm{O}(m)$; we do not include the (small) cost of computing the powers of i. Each j also performs $m'-1$ multiplications ($A' \in \mathcal{A}_{P'}$; $|A'| = m'$) and $m'-1$ exponentiations for m old shareholders $i \in A$ to verify that SUBSHARES-VALID holds (Equation (4)), for a total cost of $\mathrm{O}(mm')$. Thus, the total cost for each j to verify that both conditions hold is $\mathrm{O}(mm')$ multiplications and $\mathrm{O}(mm')$ exponentiations, exclusive of the cost of computing the commitments. In the worst case, the number of times we must restart the redistribution protocol is bounded by Equation (6).

4.4 Generalization to linear threshold sharing schemes

We can generalize our VSR protocol for application to *linear* threshold sharing schemes other than Shamir's scheme [Sha79]. Let K denote the secret set, and S_i the share value set for shareholder i. Suppose we have distributed shares of a secret $k \in K$ with a linear scheme to the access structure $\mathcal{A}$. k is then a linear combination of the shares $s_i \in S_i$ of i in an authorized subset $A \in \mathcal{A}$:

$$k = \sum_{i \in A} \psi_i(s_i)$$

where ψ_i is a homomorphism from S_i to K.

For the general case, we require a homomorphic commitment function $C(x)$ that is hard to invert. We also require that there exist reliable broadcast channels among all participants and private channels between every pair of participants. We then use the general form of Feldman's VSS scheme [Fel87] to verify that SUBSHARES-VALID holds, and

$$C(k) = \prod_{i \in A} C\left(\psi_i(s_i)\right)$$

to verify that SHARES-VALID holds.

4.5 Proof of correctness

We prove that NEW-SHARES-VALID holds after redistribution if SHARES-VALID and SUBSHARES-VALID hold. We also show that Equations (4) and (5) verify that SUBSHARES-VALID and SHARES-VALID hold.

Lemma 1 SUBSHARES-VALID *holds if Equation (4) holds.*

PROOF: Proved by Feldman [Fel87]. □

Lemma 2 SHARES-VALID *holds if Equation (5) holds.*

PROOF: Assume that Equation (5) holds. It then follows that SHARES-VALID holds from Equation (2) and the homomorphic properties of exponentiation. □

Theorem 1 (VSR correctness) *For the verifiable redistribution of shares of a secret from an $\mathcal{A}_P^{(m,n)}$ to $\mathcal{A}_{P'}^{(m',n')}$ access structure for Shamir's threshold sharing scheme [Sha79], for all secrets $k \in \mathbb{Z}_p$, and for all authorized subsets $A \in \mathcal{A}_P^{(m,n)}$, $A' \in \mathcal{A}_{P'}^{(m',n')}$,* NEW-SHARES-VALID *holds after redistribution of k with the VSR protocol if* SHARES-VALID *and* SUBSHARES-VALID *hold.*

PROOF: Assume that both SHARES-VALID and SUBSHARES-VALID hold. Then:

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

□

Our correctness proof mirrors that for Desmedt and Jajodia's secret redistribution protocol [DJ97].

4.6 Proof of security

We prove that an adversary cannot reconstruct a secret from a combination of shares distributed with Shamir's scheme to an $\mathcal{A}_P^{(m,n)}$ access structure and shares distributed to an $\mathcal{A}_{P'}^{(m',n')}$ access structure. In particular, we show that an adversary who has obtained $m-1$ old shares and $m'-1$ new shares of a secret k cannot reconstruct k (it then trivially follows that an adversary with less than $m-1$ old shares and less than $m'-1$ new shares cannot reconstruct k). In the proof, we make use of lemmas from linear algebra (summarized in Appendix A).

Theorem 2 (VSR security) *For the verifiable redistribution of shares of a secret from an $\mathcal{A}_P^{(m,n)}$ to $\mathcal{A}_{P'}^{(m',n')}$ access structure for Shamir's threshold sharing scheme [Sha79], and for all secrets $k \in \mathbb{Z}_p$, the shares s_i of shareholders i in any non-authorized subset $\overline{A} \notin \mathcal{A}_P^{(m,n)}$ cannot be used with the shares s'_j of shareholders j in any non-authorized subset $\overline{A}' \notin \mathcal{A}_{P'}^{(m',n')}$ to uniquely determine k.*

PROOF: Assume there is a unique solution for k from the shares of shareholders in $\overline{A}$ and $\overline{A}'$, where $|\overline{A}| = m-1$ and $|\overline{A}'| = m'-1$. We show that this assumption leads to a contradiction. Suppose that we have s_i of $i \in \overline{A}$ and s'_j of $j \in \overline{A}'$. We use Equation (1) to construct the system of equations

$$\begin{bmatrix} 1 & 1 & \cdots & 1^{m-1} & 0 & \cdots & 0 \\ \vdots & \vdots & \cdots & \vdots & \vdots & & \vdots \\ 1 & i & \cdots & i^{m-1} & \vdots & \ddots & \vdots \\ \vdots & \vdots & \cdots & \vdots & \vdots & & \vdots \\ 1 & (m-1) & \cdots & (m-1)^{m-1} & 0 & \cdots & 0 \\ 1 & 0 & \cdots & 0 & 1 & \cdots & 1^{m'-1} \\ 1 & \vdots & & \vdots & \vdots & \cdots & \vdots \\ 1 & \vdots & \ddots & \vdots & j & \cdots & j^{m'-1} \\ 1 & \vdots & & \vdots & \vdots & \cdots & \vdots \\ 1 & 0 & \cdots & 0 & (m'-1) & \cdots & (m'-1)^{m'-1} \end{bmatrix} \begin{bmatrix} k \\ a_1 \\ \vdots \\ a_{m-1} \\ a'_1 \\ \vdots \\ a'_{m'-1} \end{bmatrix} = \begin{bmatrix} s_1 \\ \vdots \\ s_i \\ \vdots \\ s_{m-1} \\ s'_1 \\ \vdots \\ s'_j \\ \vdots \\ s'_{m'-1} \end{bmatrix} \tag{7}$$

Let $\mathbf{M}$ denote the left-hand matrix in Equation (7), $\mathbf{a}$ the coefficient vector k, a_1 ... $a'_{m'-1}$, and $\mathbf{s}$ the share vector. The maximum possible value for $\operatorname{rank}(\mathbf{M})$ is the number of rows ($m+m'-2$, by Lemma 3 in Appendix A), which is less than the number of values in $\mathbf{a}$ ($m+m'-1$). Also, $\operatorname{rank}(\mathbf{M}) = \operatorname{rank}([\mathbf{M}|\mathbf{s}])$ since $\mathbf{s}$ is a linear combination of the columns of $\mathbf{M}$ (by the method of share generation). Thus, we have infinitely many solutions for $\mathbf{a}$ in Equation (7) (by Lemma 4 in Appendix A). We arrive at the same conclusion with any $\overline{A} \notin \mathcal{A}_P^{(m,n)}$ such that $|\overline{A}| < m-1$, and any $\overline{A}' \notin \mathcal{A}_{P'}^{(m',n')}$ such that $|\overline{A}'| < m'-1$.

Assuming that there is a unique solution for k, we can re-write Equation (7) as

$$\begin{bmatrix} 1 & \cdots & 1^{m-1} & 0 & \cdots & 0 \\ \vdots & \cdots & \vdots & \vdots & & \vdots \\ i & \cdots & i^{m-1} & \vdots & \ddots & \vdots \\ \vdots & \cdots & \vdots & \vdots & & \vdots \\ (m-1) & \cdots & (m-1)^{m-1} & 0 & \cdots & 0 \\ 0 & \cdots & 0 & 1 & \cdots & 1^{m'-1} \\ \vdots & & \vdots & \vdots & \cdots & \vdots \\ \vdots & \ddots & \vdots & j & \cdots & j^{m'-1} \\ \vdots & & \vdots & \vdots & \cdots & \vdots \\ 0 & \cdots & 0 & (m'-1) & \cdots & (m'-1)^{m'-1} \end{bmatrix} \begin{bmatrix} a_1 \\ \vdots \\ a_{m-1} \\ a'_1 \\ \vdots \\ a'_{m'-1} \end{bmatrix} = \begin{bmatrix} s_1 - k \\ \vdots \\ s_i - k \\ \vdots \\ s_{m-1} - k \\ s'_1 - k \\ \vdots \\ s'_j - k \\ \vdots \\ s'_{m'-1} - k \end{bmatrix} \tag{8}$$

Let $\mathbf{M_k}$ denote the left-hand matrix in Equation (8), and $\mathbf{a_k}$ the coefficient vector a_1 ... $a'_{m'-1}$. Let $\mathbf{M_k^{UL}}$ and $\mathbf{M_k^{LR}}$ denote the upper-left and lower-right square sub-matrices of $\mathbf{M_k}$,

$$\mathbf{M}_\mathbf{k}^{\mathbf{UL}} = \begin{bmatrix} 1 & \cdots & 1^{m-1} \\ \vdots & \cdots & \vdots \\ i & \cdots & i^{m-1} \\ \vdots & \cdots & \vdots \\ (m-1) & \cdots & (m-1)^{m-1} \end{bmatrix} \quad \text{and} \quad \mathbf{M}_\mathbf{k}^{\mathbf{LR}} = \begin{bmatrix} 1 & \cdots & 1^{m'-1} \\ \vdots & \cdots & \vdots \\ j & \cdots & j^{m'-1} \\ \vdots & \cdots & \vdots \\ (m'-1) & \cdots & (m'-1)^{m'-1} \end{bmatrix}$$

We can express $\det(\mathbf{M}_\mathbf{k}^{\mathbf{UL}})$ as

$$\det(\mathbf{M}_\mathbf{k}^{\mathbf{UL}}) = 1 \cdots i \cdots (m-1) \begin{vmatrix} 1 & \cdots & 1^{m-2} \\ \vdots & \cdots & \vdots \\ i & \cdots & i^{m-2} \\ \vdots & \cdots & \vdots \\ (m-1) & \cdots & (m-1)^{m-2} \end{vmatrix}$$

Since the rightmost term for $\det(\mathbf{M}_\mathbf{k}^{\mathbf{UL}})$ is a non-zero Vandermonde determinant (all of its elements are non-zero and pair-wise unique), and the factor $1 \cdots i \cdots (m-1)$ is also non-zero, $\det(\mathbf{M}_\mathbf{k}^{\mathbf{UL}})$ is non-zero; likewise, $\det(\mathbf{M}_\mathbf{k}^{\mathbf{LR}})$ is non-zero. Thus, $\det(\mathbf{M}_\mathbf{k})$ is non-zero since it is simply the product of $\det(\mathbf{M}_\mathbf{k}^{\mathbf{UL}})$ and $\det(\mathbf{M}_\mathbf{k}^{\mathbf{LR}})$ (by Lemma 6 in Appendix A).

If $\det(\mathbf{M}_\mathbf{k})$ is non-zero, then Equation (8) has a unique solution for $\mathbf{a}_\mathbf{k}$ (by Lemma 5 in Appendix A). If Equation (8) has a unique solution for $\mathbf{a}_\mathbf{k}$, then Equation (7) has a unique solution for $\mathbf{a}$ (since we know k). But we have already established that we have infinitely many solutions for $\mathbf{a}$, and our assumption that we have a unique solution for k has led to a contradiction. Thus, we cannot uniquely determine k with the shares of shareholders in $\overline{A}$ and $\overline{A}'$. $\square$

5 Summary

We have presented a protocol to verifiably redistribute shares of secrets from an (m,n) to (m',n') access structure for Shamir's threshold sharing scheme. A generalization of our protocol to linear sharing schemes is also presented. We identified a vulnerability in Desmedt and Jajodia's redistribution protocol and proved that two conditions, SHARES-VALID and SUBSHARES-VALID, are sufficient to guarantee that new shareholders have valid shares after redistribution. We also proved that an adversary cannot combine old shares and new shares to reconstruct the secret, provided that the adversary has less than m old shares and m' new shares. Our redistribution protocol can tolerate up to $m-1$ faulty old shareholders (provided that there are at least m non-faulty old shareholders).

In contrast to proactive secret sharing in which redistribution occurs within the same set of shareholders, verifiable secret redistribution achieves flexible secret management through redistribution of shares to different shareholders with a different access structure. We identified that additional verification information must be passed to successive sets of shareholders. We pointed out that identification and removal of faulty shareholders is not immediately possible if the new members must rely on the old shareholders to distribute verification information. In the worst case, the number of times we must restart the redistribution protocol to eliminate faulty shareholders is bounded by Equation (6).

The primary contribution of our work is that in our protocol, new shareholders can verify the validity of their shares after redistribution from old to new access structures.

We have implemented a simple prototype of our protocol that uses Castro and Liskov's Byzantine fault-tolerance library for broadcast communications [CL99], and are currently incorporating the protocol into a survivable storage system [XXX] to evaluate its performance costs.

References

[BBCM92] B. Blakley, G. R. Blakley, A. H. Chan, and J. L. Massey. Threshold schemes with disenrollment. In *Proc. of CRYPTO 1992, the 12th Ann. Intl. Cryptology Conf.*, vol. 740 of *Lecture Notes in Computer Science*, pp. 540–548. Aug. 1992.

[BCSV96] C. Blundo, A. Cresti, A. D. Santis, and U. Vaccaro. Fully dynamic secret sharing schemes. *Theoretical Computer Science*, 165(2):407–440, Oct. 1996.

[BDET00] W. J. Bolosky, J. R. Douceur, D. Ely, and M. Theimer. Feasibility of a serverless distributed file system deployed on an existing set of desktop PCs. In *Proc. of SIGMETRICS 2000, the Intl. Conf. on Measurement and Modeling of Computing Systems*, pp. 34–43. June 2000.

[Bea65] R. A. Beaumont. *Linear algebra*. Harcourt, Brace & World, Inc., 1965.

[Ben87] J. C. Benaloh. Secret sharing homomorphisms: Keeping shares of a secret secret. In *Proc. of CRYPTO 1986, the 6th Ann. Intl. Cryptology Conf.*, vol. 263 of *Lecture Notes in Computer Science*, pp. 213–222. 1987.

[BKO93] T. Beth, H.-J. Knobloch, and M. Otten. Verifiable secret sharing for monotone access structures. In *Proc. of the 1st ACM Intl. Conf. on Computer and Communications Security*, pp. 189–194. Nov. 1993.

[Bla79] G. R. Blakley. Safeguarding cryptographic keys. In *Proc. of the Natl. Computer Conf.*, vol. 48 of *American Federation of Information Processing Societies Proceedings*, 1979.

[Cac95] C. Cachin. On-line secret sharing. In *Proc. of the 5th IMA Conf. on Cryptography and Coding*, vol. 1025 of *Lecture Notes in Computer Science*, pp. 90–198. Dec. 1995.

[CGMA85] 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*, pp. 383–395. Oct. 1985.

[CH94] R. Canetti and A. Herzberg. Maintaining security in the presence of transient faults. In *Proc. of CRYPTO 1994, the 14th Ann. Intl. Cryptology Conf.*, vol. 839 of *Lecture Notes in Computer Science*, pp. 425–438. Aug. 1994.

[CL99] M. Castro and B. Liskov. Practical Byzantine fault tolerance. In *Proc. of the 3nd Symp. on Operating Systems Design and Implementation*, pp. 173–186. Feb. 1999.

[Des97] Y. Desmedt. Some recent research aspects of threshold cryptography. In *Proc. of the 1st Intl. Information Security Workshop*, vol. 1396 of *Lecture Notes in Computer Science*, pp. 158–173. Sept. 1997.

[DJ97] 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.

[DKK$^+$01] F. Dabek, M. F. Kaashoek, D. Karger, R. Morris, and I. Stoica. Wide-area cooperative storage with CFS. In *Proc. of the 18th Symp. on Operating Systems Principles*, pp. 202–215. Oct. 2001.

[Fel87] P. Feldman. A practical scheme for non-interactive verifiable secret sharing. In *Proc. of the 28th IEEE Ann. Symp. on Foundations of Computer Science*, pp. 427–437. Oct. 1987.

[FGMY97a] 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*, pp. 384–393. Oct. 1997.

[FGMY97b] Y. Frankel, P. Gemmell, P. D. MacKenzie, and M. Yung. Proactive RSA. In *Proc. of CRYPTO 1997, the 17th Ann. Intl. Cryptology Conf.*, vol. 1294 of *Lecture Notes in Computer Science*, pp. 440–454. Aug. 1997.

[FMY99] Y. Frankel, P. D. MacKenzie, and M. Yung. Adaptively-secure optimal-resilience proactive RSA. In *Proc. of ASIACRYPT1999, the 5th Intl. Conf. on the Theory and Application of Cryptology and Information Security*, vol. 1716 of *Lecture Notes in Computer Science*, pp. 180–194. Nov. 1999.

[FMY01] Y. Frankel, P. D. MacKenzie, and M. Yung. Adaptive security for the additive-sharing based proactive RSA. In *Proc. of PKC 2001, the 4th Intl. Workshop on Practice and Theory in Public Key Cryptography*, vol. 1992 of *Lecture Notes in Computer Science*, pp. 240–263. Febraury 2001.

[GJKR96] R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin. Robust threshold DSS signatures. In *Proc. of EUROCRYPT 1996, the Intl. Conf. on the Theory and Application of Cryptographic Techniques*, vol. 1070 of *Lecture Notes in Computer Science*, pp. 354–371. May 1996.

[GM95] R. Gennaro and S. Micali. Verifiable secret sharing as secure computation. In *Proc. of EUROCRYPT 1995, the Intl. Conf. on the Theory and Application of Cryptographic Techniques*, vol. 921 of *Lecture Notes in Computer Science*, pp. 168–182. May 1995.

[GMW87] O. Goldreich, S. Micali, and A. Wigderson. How to prove all NP statements in zero-knowledge and a methodology of cryptograhpic protocol design. In *Proc. of CRYPTO 1986, the 6th Ann. Intl. Cryptology Conf.*, vol. 263 of *Lecture Notes in Computer Science*, pp. 171–185. 1987.

[HJJ$^+$97] 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*, pp. 100–110. Apr. 1997.

[HJKY95] A. Herzberg, S. Jarekci, H. Krawczyk, and M. Yung. Proactive secret sharing or: How to cope with perpetual leakage. In *Proc. of CRYPTO 1995, the 15th Ann. Intl. Cryptology Conf.*, vol. 963 of *Lecture Notes in Computer Science*, pp. 339–352. Aug. 1995.

[Kos82] A. I. Kostrikin. *Introduction to algebra*. Springer-Verlag, 1982.

[OY91] R. Ostrovsky and M. Yung. How to withstand mobile virus attacks. In *Proc. of the 10th Ann. ACM Symp. on Principles of Distributed Computing*, pp. 51–59. Aug. 1991.

[Ped91] T. P. Pedersen. Non-iteractive and information-theoretic secure verifiable secret sharing. In *Proc. of CRYPTO 1991, the 11th Ann. Intl. Cryptology Conf.*, vol. 576 of *Lecture Notes in Computer Science*, pp. 129–140. Aug. 1991.

[Rab94] T. Rabin. Robust sharing of secrets when the dealer is honest or cheating. *Journal of the ACM*, 41(6):1089–1109, Nov. 1994.

[Rab98] T. Rabin. A simplified approach to threshold and proactive RSA. In *Proc. of CRYPTO 1998, the 18th Ann. Intl. Cryptology Conf.*, vol. 1462 of *Lecture Notes in Computer Science*, pp. 89–104. Aug. 1998.

[RBO89] 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*, pp. 73–85. May 1989.

[RD01] A. Rowstron and P. Druschel. Storage management and caching in PAST, a large-scale, persistent peer-to-peer storage utility. In *Proc. of the 18th Symp. on Operating Systems Principles*, pp. 188–201. Oct. 2001.

[Sha79] A. Shamir. How to share a secret. *Communications of the ACM*, 22(11):612–613, Nov. 1979.

[WBP$^+$01] 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. Tech. Rep. CMU-CS-01-120, Sch. of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213, May 2001.

[WBS$^+$00] J. J. Wylie, M. W. Bigrigg, J. D. Strunk, G. R. Ganger, H. Kiliççöte, and P. K. Khosla. Survivable information storage systems. *IEEE Computer*, pp. 61–68, Aug. 2000.

[XXX] Removed for blind submission.

[ZSvR00] L. Zhou, F. B. Schneider, and R. van Renesse. COCA: A secure distributed on-line certification authority. Tech. Rep. TR2000-1828, Dept. of Computer Science, Cornell University, Ithaca, NY 14853, Dec. 2000.

A Linear algebra lemmas

To complete the security proof, we require some lemmas (presented by Beaumont [Bea65] and Kostrikin [Kos82]) for systems of u linear equations in v unknowns of the form

$$\begin{array}{c} m_{11}x_1 + m_{12}x_2 + \cdots + m_{1v}x_v = b_1 \\ m_{21}x_1 + m_{22}x_2 + \cdots + m_{2v}x_v = b_2 \\ \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ m_{u1}x_1 + m_{u2}x_2 + \cdots + m_{uv}x_v = b_2 \end{array} \tag{9}$$

Let $\mathbf{M}$ and $\mathbf{x}$ denote the coefficient matrix and unknown vector

$$\mathbf{M} = \begin{bmatrix} m_{11} & \cdots & m_{1v} \\ \vdots & \ddots & \vdots \\ m_{u1} & \cdots & m_{uv} \end{bmatrix} \quad , \quad \mathbf{x} = \begin{bmatrix} x_1 \\ \vdots \\ x_v \end{bmatrix} \quad , \quad \mathbf{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_u \end{bmatrix}$$

let $[\mathbf{M}|\mathbf{y}]$ denote the *augmented matrix*

$$[\mathbf{M}|\mathbf{y}] = \begin{bmatrix} m_{11} & \cdots & m_{1v} & y_1 \\ \vdots & \ddots & \vdots & \vdots \\ m_{u1} & \cdots & m_{uv} & y_u \end{bmatrix}$$

let $\operatorname{rank}(\mathbf{M})$ denote the rank of $\mathbf{M}$ (number of linearly independent columns in $\mathbf{M}$), and let $\det(\mathbf{M})$ denote the determinant of $\mathbf{M}$.

Lemma 3 $\operatorname{rank}(\mathbf{M}) = \operatorname{rank}(\mathbf{M}^T)$.

Lemma 4 (Kronecker-Capelli theorem) *If (and only if)* $\operatorname{rank}(\mathbf{M}) = \operatorname{rank}([\mathbf{M}|\mathbf{y}])$*, then Equation (9) has a solution for* $\mathbf{x}$*. Furthermore, if* $\operatorname{rank}(\mathbf{M}) < v$*, then Equation (9) has infinitely many solutions for* $\mathbf{x}$.

Lemma 5 (Cramer's rule) *If* $u = v$ *and* $\det(\mathbf{M}) \neq 0$*, then Equation (9) has a unique solution for* $\mathbf{x}$.

Lemma 6 *For* $u \times u$ *matrix* $\mathbf{A}$*,* $v \times v$ *matrix* $\mathbf{B}$*, and* $u \times v$ *matrix* $\mathbf{C}$*:*

$$\det\left(\begin{bmatrix} \mathbf{A} & \mathbf{C} \\ \mathbf{0} & \mathbf{B} \end{bmatrix}\right) = \det(\mathbf{A})\det(\mathbf{B})$$

PROOF: Presented by Kostrikin [Kos82]. □

B Summary of Desmedt and Jajodia's secret redistribution protocol

To redistribute a secret k, $k \in \mathbb{Z}_p$, from an $\mathcal{A}_P^{(m,n)}$ to $\mathcal{A}_{P'}^{(m',n')}$ access structure, using the authorized subset $A \in \mathcal{A}_P^{(m,n)}$:

1. For each $i \in A$, use the polynomial $a'_i(j) = s_i + a'_{i1}j + \ldots + a'_{i(m'-1)}j^{m'-1}$ to compute the subshares $\hat{s}_{ij}$ of s_i, and send $\hat{s}_{ij}$ to the corresponding $j \in P'$.
2. For each $j \in P'$, generate a new share s'_j by Lagrange interpolation:

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

b_i are interpolation constants that may be precomputed.

C Summary of Feldman's VSS scheme

1. Use the polynomial $a(i) = k + a_1 i + \ldots + a_{m-1} i^{m-1}$ to compute the shares s_i of k, and send s_i to the corresponding $i \in P$ over private channels.

2. Use generator g to compute $g^k, g^{a_1} \ldots g^{a_{m-1}}$, and broadcast them to all $i \in P$.

3. For each $i \in P$, verify that:

$$g^{s_i} = g^k \prod_{l=1}^{m-1} (g^{a_l})^{i^l}$$

   If the condition holds, i broadcasts a "commit" message. Otherwise, i broadcasts an "abort" message.