An *l*-Span Generalized Secret Sharing Scheme

Lein Harn and Hung-Yu Lin

Computer Science Telecommunications Program
University of Missouri - Kansas City
Kansas City, MO 64110

Abstract. For some secret sharing applications, the secret reconstructed is not revealed to the participants, and therefore, the secret/shadows can be repeatedly used without having to be changed. But for other applications, in which the secret reconstructed is revealed to participants, a new secret must be chosen and its corresponding shadows must be regenerated and then secretly distributed to participants again, in order to enforce the same secret sharing policy. This is inefficient because of the overhead in the generation and distribution of shadows. In this paper, an *l*-span secret sharing scheme for the general sharing policy is proposed to solve the secret/shadows regeneration problem by extending the life span of the shadows from *1* to *l*, i. e., the shadows can be repeatedly used for *l* times to generate *l* different secrets.

I. Introduction

A secret sharing scheme is a method of hiding a secret among multiple shadows such that the secret can be retrieved by some subsets of these shadows but not by the others according to a given secret sharing policy. For example, Shamir's well-known (m,n)-threshold scheme [1] realizes the secret sharing policy in which any m, or more than m shadows, can reconstruct the secret. This sharing policy is far too simple for many applications because,

E.F. Brickell (Ed.): Advances in Cryptology - CRYPTO '92, LNCS 740, pp. 558-565, 1993.

implicitly, it assumes that every participant has equal privilege to the secret or every participant is equally trusted. Complicated sharing policies, in which participants have different privileges, can also be realized by other generalized secret sharing schemes [2, 3, 4]. One common feature among almost all secret sharing schemes is that once the reconstructed secret is exposed, a new secret must be chosen and its corresponding shadows must be regenerated and then secretly distributed to participants again, in order to enforce the same secret sharing policy. From life span aspects of the shadows, these traditional schemes are called *1*-span secret sharing schemes.

Depending on applications, the secret can be reconstructed in a tamper-free device without revealing it to the participants. For such applications, the secret/shadows can be repeatedly used. But, for other applications, in which the secret reconstructed is revealed to participants, a new secret must be chosen and its corresponding shadows are then generated in order to enforce the same secret sharing policy. Such regeneration process is inefficient because of the overhead in the generation and distribution of shadows.

One previous work which tries to solve the shadow regeneration problem can be found in [5], but it deals with only traditional threshold schemes and the threshold value is decreased in proportion to the number of different secrets which have been revealed. In this paper, an *l*-span secret sharing scheme for the general sharing policy will be proposed to solve the secret/shadows regeneration problem by extending the life span of the shadows from *1* to *l*., i. e., the shadows can be repeatedly used for *l* times to generate *l* different secrets. Section II gives some definitions and Section III briefly reviews the scheme on which the proposed *l*-span generalized secret sharing scheme is

based. The l-span generalized secret sharing scheme and an example are included in Section IV.

II. Definitions

Suppose a secret key k is to be shared according to a given secret sharing policy by a group of m participants U = $\{u_1, u_2, \ldots, u_m\}$. Each participant may be designated with a different privilege. A generalized secret sharing scheme is a method of breaking k into m pieces $k_1, k_2, \ldots, k_m$, with k_i secretly distributed to u_i such that

(1) if $\mathcal{A} \subseteq \mathcal{U}$ is a qualified subset of participants, called *positive access instance*, according to the secret sharing policy, then k can be reconstructed from shadows $\{k_i \mid u_i \in \mathcal{A}\}$.

(2) if $\mathcal{A} \subseteq \mathcal{U}$ is not a qualified subset of participants, called *negative access instance*, according to the secret sharing policy, then k cannot be reconstructed from $\{k_i \mid u_i \in \mathcal{A}\}$.

The set F of all positive access instances is called the positive access structure of the secret sharing policy and the set N of all negative access instances is called the negative access structure of the secret sharing policy. Suppose the positive access structure of a given sharing policy is F. The corresponding negative access structure is $N = 2^{\mathcal{U}} - F$.

III. Lin and Harn's Generalized Secret Sharing Scheme

The dealer first secretly selects two large primes, p and q, and publishes their product n=p*q. Then it assigns a distinct prime p_j to each negative access instance N_j of $\mathfrak{M}(N)$ and computes the tag t_i associated with participant u_i as

$$t_i = \prod_{u_i \in N_j} p_j,$$

where $\mathfrak{M}(N)$ is the maximum set of the negative access structure.

The shadows assigned to the participants are computed as

$k_i = k^{t_i} \mod n$, for $i = 1, 2, \ldots, m$, where k is the secret.

Each shadow k_i is then secretly distributed to participant u_i.

The dealer also publishes one pair of check values, t_c and k_c where

$$t_c = \prod_{N_j \in \mathcal{M}(N)} p_j.$$

and $k_c = k^{t_c} \pmod n$ for users' verification of the correctness of their received shadows.

The secret k can be reconstructed by any positive access instance according to the THEOREM 1 in [4]:

THEOREM 1. Given k_1, k_2, e_1, and e_2 such that $k_1 = k^{e_1} \mod n$ and $k_2 = k^{e_2} \mod n$, $k^r \mod n$ can be easily computed if $\gcd(e_1, e_2) = r$.

IV. The *l*-Span Generalized Secret Sharing Scheme

In this l-span secret sharing scheme, the generation of tags associated with participants is the same as mentioned above. However, since there are multiple secrets corresponding to the same set of shadows, the choice of the secrets and the generation of shadows need to be modified.

First, the secret, k, is replaced by a sequence of secrets, s_j's, where

$s_j = k^{t_c^{l-j}} \mod n$, for $j = 1, 2, \ldots, l$.

Note that each secret should be used only once to enforce the secret sharing policy and participants should reconstruct the secrets, $s_1, s_2, \ldots, s_l$, accordingly in order to obtain the maximum life span of k.

Then the shadows assigned to participants are computed as

$k_i = k^{t_i^l} \mod n$, for $i = 1, 2, \ldots, m$.

Now suppose a positive access instance $\mathcal{A}$ wants to reconstruct secret s_j. Each participant $u_i \in \mathcal{A}$ computes

$$\begin{aligned} k_{i,j} &= (k_i)^{(t_c/t_i)^{l-j}} && \bmod n \\ &= (k^{t_i^{\,l}})^{(t_c/t_i)^{l-j}} && \bmod n \\ &= (k)^{t_c^{l-j} t_i^{\,j}} && \bmod n \\ &= (k^{t_c^{l-j}})^{t_i^{\,j}} && \bmod n \\ &= (s_j)^{t_i^{\,j}} && \bmod n, \end{aligned}$$

and then submits it, instead of his shadow, k_i.

THEOREM 2. Any positive access instance $\mathcal{A}$ can reconstruct s_j, for j = 1, 2,..., l.

<Proof> The greatest common divisor of t_i^j's, for $u_i \in \mathcal{A}$, is 1, so s_j can be derived from $k_{i,j}$'s by Theorem 1.

Lemma 1. s_j's and $k_{i,j}$'s, $i = 1$ to m, can be derived from s_r, if $j < r \le l$.

<proof> Since $r-j > 0$, $s_j = k^{t_c^{l-j}} \bmod n$, $s_r = k^{t_c^{l-r}} \bmod n$, and modular exponentiation is an one-way function, we can derive s_j from s_r as

$$s_j = (s_r)^{t_c^{r-j}} \bmod n.$$

Similarly, we can derive $k_{i,j}$'s from s_r.

Lemma 2. s_j's and $k_{i,j}$'s, $i = 1$ to m, cannot be derived with knowledge of s_r, if $r < j \le l$

<proof> From RSA assumption in [4], i.e., the modular exponentiation is an one-way function.

THEOREM 3. No negative access instance can derives s_j unless some s_r, with $j < r$, have been revealed.

<proof> This theorem can be proved from Lemma 1, Lemma 2, and THEOREM 5 in [4].

Here we give an example to illustrate our idea.

EXAMPLE. Suppose there are four members in the system, Alice, Bob, Cathy, and David. The secret sharing policy is that either Alice and Bob working together, or Bob and Cathy working together, or Alice, Cathy, and David working together can reconstruct the secret. The positive access structure of this sharing policy can be represented as

$$F = (AB) \cup (BC) \cup (ACD).$$

The negative access structure is therefore the complement of the positive access structure and can be represented as

$$N = (AB'C'D') \cup (AB'C'D) \cup (AB'CD') \cup (A'BC'D') \cup (A'BC'D)$$
$$\cup (A'B'C'D') \cup (A'B'C'D) \cup (A'B'CD) \cup (A'B'CD').$$

By LEMMA 2-5 in reference [4], we can derive

$$\mathfrak{M}(N) = \mathfrak{M}(\mathfrak{M}(B'C') \cup \mathfrak{M}(B'D') \cup \mathfrak{M}(A'B') \cup \mathfrak{M}(A'C'))$$
$$= \mathfrak{M}(B'C') \cup \mathfrak{M}(B'D') \cup \mathfrak{M}(A'B') \cup \mathfrak{M}(A'C')$$
$$= \{\{\text{Alice, David}\}\ \{\text{Alice, Cathy}\}\ \{\text{Cathy, David}\}\ \{\text{Bob, David}\}\}.$$

This maximum set of the negative access structure tells that the secret key cannot be reconstructed either by Alice and David alone, or by Alice and Cathy alone, or by Cathy and David alone, or by Bob and David alone.

Now, the trusted key center selects two secret large primes, p and q, and publishes their product $n = p*q$. Then it selects p_1, p_2, p_3, and p_4 as the public primes. These prime numbers can be chosen as small as possible. A secret key, k, is chosen from $[1, n-1]$. According to this l-span generalized secret sharing scheme, the secret keys to be shared are chosen as

$$s_j = k^{t_c^{l-j}} \bmod n, \text{ for } j = 1, 2, \ldots, l, \text{ where } t_c = p_1p_2p_3p_4,$$

and the tags and the corresponding shadows associated with users are computed as

$t_{Alice} = p_1 p_2$, $\quad k_{Alice} = k^{p_1^{l} p_2^{l}} \bmod n$,

$t_{Bob} = p_4$, $\quad k_{Bob} = k^{p_4^{l}} \bmod n$,

$t_{Cathy} = p_2 p_3$, $\quad k_{Cathy} = k^{p_2^{l} p_3^{l}} \bmod n$, and

$t_{David} = p_1 p_3 p_4$, $\quad k_{David} = k^{p_1^{l} p_2^{l} p_4^{l}} \bmod n$.

Now suppose Alice and Bob, which combination is a positive access instance, want to reconstruct s_j. Alice will present her shadow as

$$k_{Alice,j} = (k_{Alice})^{p_3^{l-j} p_4^{l-j}} = k^{p_1^{l} p_2^{l} p_3^{l-j} p_4^{l-j}} \bmod n$$

and Bob will present his shadow as

$$k_{Bob,j} = (k_{Bob})^{p_1^{l-j} p_2^{l-j} p_3^{l-j}} = k^{p_1^{l-j} p_2^{l-j} p_3^{l-j} p_4^{l}} \bmod n.$$

By Euclid algorithm, since

$$\gcd(p_1^{l} p_2^{l} p_3^{l-j} p_4^{l-j}, p_1^{l-j} p_2^{l-j} p_3^{l-j} p_4^{l}) = p_1^{l-j} p_2^{l-j} p_3^{l-j} p_4^{l-j},$$

an integer pair (a, b) can be found such that

$$a * (p_1^{l} p_2^{l} p_3^{l-j} p_4^{l-j}) + b * (p_1^{l-j} p_2^{l-j} p_3^{l-j} p_4^{l}) = p_1^{l-j} p_2^{l-j} p_3^{l-j} p_4^{l-j}.$$

Therefore, the secret s_j can be reconstructed by computing

$$(k_{Alice,j})^{a} * (k_{Bob,j})^{b} \bmod n$$
$$= (k)^{p_1^{l-j} p_2^{l-j} p_3^{l-j} p_4^{l-j}} \bmod n$$
$$= s_j$$

V. Conclusion

An l-span generalized secret sharing scheme is proposed in this paper. It allows secrets to be shared in a more efficient way in which same set of shadows can be used to reconstruct l different secrets. For applications in which the reconstructed secret must be revealed and the same secret sharing policy must still be enforced, it alleviates the overhead in the generation and distribution of shadows.

References

[1] A. Shamir, "How to Share a Secret", Communication ACM 22, 11, Nov. 1979, 612-613.

[2] M. Ito, A. Saito, and T. Nishizeki, "Secret Sharing Scheme Realizing General Access Structure", Proc. Glob. Com(1987).

[3] J. Benaloh, and J. Leichter, "Generalized Secret Sharing and Monotone Functions", Proc. Crypto '88, Springer-Verlag, 27-35.

[4] H. Y. Lin and L. Harn, "A Generalized Secret Sharing Scheme with Cheater Detection", Proc. Asiacrypt '91, Nov. 1991, Japan.

[5] C. S. Laih, L. Harn, J. Y. Lee and T. Hwang, "Dynamic Threshold Scheme Based on the Definition of Cross-Product in an N-Dimensional Linear Space", Proc. Crypto '89, Springer-Verlag, 286-297.