Secret Sharing

CPT, 2006

Version 3

1 Introduction

In all secure systems that use cryptography in practice, keys have to be protected by encryption under other keys when they are stored in a physically insecure location. But the keys used for protection have to be protected themselves, so no matter what we do, we cannot avoid having one or more keys in our system that are only protected because they are stored in a physically secure way. These are typically very high priority keys, such as the secret key that a certification authority (CA) uses to create certificates. Precisely because such a key is so important, it would be a disaster if it was revealed to an adversary. But it would be equally bad if the key was lost and could not be retrieved. In other words, there is a big need to keep such keys *secret* and *available* at the same time.

This seemingly puts designers of security systems in a rather difficult dilemma: to make sure that a key is not revealed to anyone, one is inclined to store it only in a single, very secure location; while the need to make sure the key is always available seems to imply that you should store the key in as many different locations as possible. Secret sharing is a technique that allows us to nevertheless address both of these concerns at the same time.

2 The Concept

A *threshold secret sharing scheme* is defined by a probabilistic algorithm $\mathcal{S}$. It takes as input a *secret* s chosen from some finite set S, and it outputs n *shares*, i.e., bit strings $s_1, ..., s_n$. Finally, the secret sharing scheme comes with a *threshold* t, a number with $0 < t < n$. The idea is that if at most t

shares are known, then this reveals nothing about s, whereas any set of at least $t+1$ shares determine s uniquely. More precisely, we want:

Privacy: Take any subset I of the indices $\{1, 2, ..., n\}$ of size at most t, and run $\mathcal{S}$ on input some $s \in S$. Then the probability distribution of $\{s_i |\ i \in I\}$ is independent of s.

Correctness: Take any subset J of the indices $\{1, 2, ..., n\}$ of size at least $t+1$, and run $\mathcal{S}$ on input some $s \in S$. Then s is uniquely determined by $\{s_i |\ i \in J\}$, and in fact there is an efficient algorithm that computes s from $\{s_i |\ i \in J\}$.

This concept was introduced by Shamir in 78, who also proposed the implementation we describe below. If we had such a scheme, we could use it to store one of the important keys we discussed earlier, by letting the key be the secret, and store the n shares in different locations. An adversary would have to get hold of at least $t+1$ shares to steal the key, and on the other hand, as long as we loose no more than $n-t-1$ shares, there will still be enough information to reconstruct the key. So this is a solution that is at the same time robust against loss of information and information leakage.

3 An implementation

Assume we set $S = \mathbb{Z}_p$ for some prime p, where $p > n$, and t is the threshold value we want. Then we can describe the algorithm $\mathcal{S}$ proposed by Shamir:

1. Choose elements $a_1, ..., a_t \in \mathbb{Z}_p$ at random, and let $f(x)$ be the polynomial $f(x) = s + a_1 x + a_2 x2 + ... + a_t x^t$. In other words: choose a random polynomial $f(x)$ over $\mathbb{Z}_p$ of degree at most t, such that $f(0) = s$.

2. Let the shares be defined by $s_i = f(i) \bmod p$ for $i = 1, ..., n$.

This scheme has the properties we outlined above, simply because of a classical results on so called Lagrange interpolation:

PROPOSITION 1 *For any field F, and any set of pairs $(x_1, y_1), ..., (x_{t+1}, y_{t+1}) \in F \times F$ where the x_i's are distinct, there exists exactly one polynomial $g(x)$ over F of degree at most t, such that $g(x_i) = y_i$ for $i = 1...t+1$. All coefficients of this polynomial can be efficiently computed from $(x_1, y_1), ..., (x_{t+1}, y_{t+1})$.*

PROOF. Note that the polynomial

$$g_i(x) = \frac{(x_1 - x)(x_2 - x)\cdots(x_{i-1} - x)(x_{i+1} - x)\cdots(x_{t+1} - x)}{(x_1 - x_i)(x_2 - x_i)\cdots(x_{i-1} - x_i)(x_{i+1} - x_i)\cdots(x_{t+1} - x_i)}$$

satisfies $g_i(x_i) = 1, g_i(x_j) = 0$ for $i \neq j$, and has degree at most t. It follows that

$$g(x) = y_1 g_1(x) + ... + y_{t+1} g_{t+1}(x)$$

has the right properties. It follows directly by construction that g can be efficiently computed. There can be only one solution, since if two different polynomials $g(x), g'(x)$ were both solutions, then $g(x) - g'(x)$ would be a non-zero polynomial of degree at most t with $t+1$ roots, which cannot exist. △

This fact immediately implies correctness. It also implies privacy: we will prove that any set of t shares give no information on the secret, this is sufficient since of course less than t shares give even less information. So fix any index set I of size t and any secret s. Let us call a polynomial *relevant* if it evaluates to s in 0 and has degree at most t. Any relevant polynomial f leads to a set of shares with indices in I, namely $\{f(i)|\ i \in I\}$. Conversely, consider *any* potential set of shares $A = \{s_i|\ i \in I\}$. Could this set result from sharing s? the answer is clearly yes, since by Lagrange interpolation, there is exactly one relevant polynomial $f_A(x)$ that satisfies $f_A(0) = s$ and $f_A(i) = s_i$ for $i \in I$. Since there are exactly p^t relevant polynomials, and also p^t potential sets of shares with indices in I, we have a 1-1 correspondence between the two. So since $\mathcal{S}$ chooses randomly between all relevant polynomials, we see that sharing s results in all potential sets of shares A being equally likely. This is true for any s and I, so privacy follows.

4 Some more general facts on secret sharing

Shamir's scheme is very efficient in the sense that all shares have the same size as the secret, measured in the number of bits you need to store them. This is in fact optimal, by the following

LEMMA 1 *For every probability distribution with which the secret s is chosen, and for any secret sharing scheme, the entropy of every share is at least the entropy of the secret. In particular, an optimal encoding requires at least as many bits to write down a share as to write down the secret.*

PROOF. Suppose the entropy of the secret is l bits. Without loss of generality, assume we look at the first share s_1. Take some set of shares A such that A is insufficient to determine s, but $A \cup \{s_1\}$ *is* sufficient. Then, by the privacy property, given A you have 0 bits of information on s. But given A and s, you have l bits of information. Thus by being told s_1, you learn l bits of information, so the entropy of s_1 must be at least that large. $\triangle$

Sometimes you want a more general solution than what threshold secret sharing can provide. Suppose you are protecting a password s that gives access to executing a particular critical operation in some system, and you have 4 persons in the game, A, B, C and D. Suppose further that you want that in order to carry out the operation, you want agreement from at least $\{A, B\}$ or $\{B, C\}$ or $\{C, D\}$ (but $\{A, D\}$ is not considered sufficient). The sets we listed here as approved, plus all larger sets are called the *qualified* sets. Shamir's solution cannot handle this, because it would give access to all pairs of players. What we can do, however, is to share $s \in \mathbb{Z}_p$ "independently" in each of the minimal qualified sets: we choose s_1, s_1', s_2, s_2',s_3, s_3' randomly in $\mathbb{Z}_p$ such that $s_i + s_i' = s$ for $i = 1, 2, 3$. Then we give s_1 to A, s_1', s_2 to B, s_2', s_3 to C and s_3' to D. This clearly satisfies that the qualified sets can easily find s, but unqualified sets have no information.

This is a special case of general secret sharing: we are given a so called *access structure* Γ, namely a family of subsets of $\{1, ..., n\}$. Subsets in Γ correspond to subsets of shares we want to be sufficient to find the secret, so for this to make sense, such a family must be *monotone*, that is, $A \in \Gamma, A \subset B$ implies $B \in \Gamma$: if players in A together know enough information to find the secret, then of course players in a larger set also know enough. A *perfect secret sharing scheme* $\mathcal{S}$ for Γ satisfies:

Privacy: Take any subset $I \notin \Gamma$ and run $\mathcal{S}$ on input some $s \in S$. Then the probability distribution of $\{s_i |\ i \in I\}$ is independent of s.

Correctness: Take any subset $J \in \Gamma$ and run $\mathcal{S}$ on input some $s \in S$. Then s is uniquely determined by $\{s_i |\ i \in J\}$, and in fact there is an efficient algorithm that computes s from $\{s_i |\ i \in J\}$.

Of course, threshold secret sharing as above is a special case. Using the idea we saw of sharing the secret independently for each (minimal) qualified set, one can easily prove:

THEOREM 1 *There exists a perfect secret sharing scheme for every monotone access structure.*

However, the idea behind the proof does not lead to an efficient scheme, since there may be a very large number of qualified sets: if for instance, we define a set to be qualified if it has at least $n/2$ elements, then the number of such sets is exponential in n, and each "shareholder" would receive an exponential number of values as his share, namely one value for each set he is a member of. By contrast, since this is a threshold situation, we could instead have used Shamir's idea and every share would have been the same size as the secret. One can show that it is not possible to handle all access structures efficiently in the sense that the shares are at most a polynomial factor larger than the secret, but it is an open question to characterize those structures that can be handled efficiently.

5 Exercises

EXERCISE 1 *Assume that a secret $x \in \mathbb{Z}_p$ has been shared with threshold t using Shamir's secret sharing scheme, as $(x_1, \ldots, x_n)$. I.e. there are n servers $S_1, \ldots, S_n$ and server S_i holds the share x_i, and there exists a polynomial $f(X) \in \mathbb{Z}_p[X]$ of degree at most t such that $f(0) = x$ and $f(i) = x_i$ for $i = 1, \ldots, n$. Let $a \in \mathbb{Z}_p$ be some value known by all servers.*

Prove that if each server S_i locally computes $z_i = ax_i \bmod p$, then $(z_1, \ldots, z_n)$ is a Shamir secret sharing with threshold t of the secret $z = ax \bmod p$. I.e. prove that there exists a polynomial $h(X) \in \mathbb{Z}_p[X]$ of degree at most t such that $h(0) = z$ and $h(i) = z_i$ for $i = 1, \ldots, n$.

EXERCISE 2 *Assume that secrets $x, y \in \mathbb{Z}_p$ has been shared with threshold t using Shamir's secret sharing scheme, as $(x_1, \ldots, x_n)$ respectively $(y_1, \ldots, y_n)$. I.e. server S_i holds the shares x_i and y_i and there exist polynomials $f(X)$, $g(X) \in \mathbb{Z}_p[X]$, each of degree at most t, such that $f(0) = x$ and $g(0) = y$ and $f(i) = x_i$ and $g(i) = y_i$ for $i = 1, \ldots, n$.*

Prove that if each server S_i locally computes $z_i = x_i + y_i \bmod p$, then $(z_1, \ldots, z_n)$ is a Shamir secret sharing with threshold t of the secret $z = x + y \bmod p$. I.e. prove that there exists a polynomial $h(X) \in \mathbb{Z}_p[X]$ of degree at most t such that $h(0) = z$ and $h(i) = z_i$ for $i = 1, \ldots, n$.

EXERCISE 3 *Assume that secrets $x, y \in \mathbb{Z}_p$ has been shared with threshold t using Shamir's secret sharing scheme, as in the Exercise 2, and assume now in addition that $t < n/2$.*

Prove that if each server S_i locally computes $z_i = x_i y_i \bmod p$, then $(z_1, \ldots, z_n)$ is a Shamir secret sharing with threshold $2t$ of the secret $z = xy \bmod p$. I.e. prove that there exists a polynomial $h(X) \in \mathbb{Z}_p[X]$ of degree at most $2t$ such that $h(0) = z$ and $h(i) = z_i$ for $i = 1, \ldots, n$.