Secret Sharing

As portrayed in Hollywood movies, launching a nuclear missile requires the president to issue the launch command and two launch administrators to authorize the launch (usually by turning two keys simultaneously). Although this activation scheme could be implemented by having the missile check for the presence of the three launch commands, a better scheme would render the missile completely inoperative until the president and the two launch administrators provide some input to the system. A physical implementation of this idea would distribute the uranium in the warhead between the three authorities, and they would have to place their share of the uranium in the warhead for it to be dangerous.

Can we develop a digital analog of this scheme? Given a secret, x, can we distribute x among three parties so that all three parties must cooperate to recover x?

Idea Each party, i, generates a public/private key pair (p_i, s_i) and we broadcast to all users $E(p_3, E(p_2, E(p_1, x)))$. This scheme works but is not unconditionally secure.

Idea Choose random x_1, x_2, x_3 such that $x_1 \oplus x_2 \oplus x_3 \oplus x = 0$, and give x_i to party i. Then $x_i \oplus x_j$ is uniformly distributed for all i and j, so two of the parties cannot collude to recover any information about x. Hence the scheme is unconditionally secure.

The above schemes are "3-out-of-3" schemes: the secret is divided among 3 people, and 3 people are required to reconstruct the secret. We can create a trivial 1-out-of-3 scheme by giving a copy of x to each party. We can also do 2-out-of-3 secret sharing.

Idea Divide the secret into x_1, x_2, x_3 as above, and give party i shares x_i and x_{i+1}.

Idea Execute a 2-out-of-2 sharing scheme for each pair of parties in the scheme. In other words, pick $x_1 \oplus x_2 = y_1 \oplus y_2 = z_1 \oplus z_2 = x$, and give party 1 shares x_1 and z_2, party 2 shares z_1 and y_2, and party 3 y_1 and x_2.

These schemes have a *threshold* of 2: 2 users can recover x. The latter scheme obviously generalizes to w-out-of-n sharing schemes by using $\binom{n}{w}$ w-out-of-w sharing schemes, but the size of each share is proportional to $\binom{n}{w}$. This suggests a notion of efficiency for sharing schemes.

Definition 1 *The rate of a sharing scheme is the ratio of the size of the secret to the size of each share.*

For unconditionally secure schemes, rate 1 is optimal. To see why, suppose a set of users, $B = \{i_1, \ldots, i_k\}$, is allowed to reconstruct the secret x and each user's share of the secret is of length $\ell < |x|$. Then users $i_1, \ldots, i_{k-1}$ can guess user i_k's share and recover x. Rates greater than 1 can be achieved by allowing fuzzy thresholds, but we won't discuss those in this note.

We should also set down a precise definition of a secret sharing scheme.

Definition 2 *A probabilistic polynomial-time computable function $f : \mathcal{K} \times \{1, \dots, n\} \to X$ is an unconditionally secure w-out-of-n secret sharing scheme if*

- *(robustness) For every k and $B \subseteq \{1, \dots, n\}$ with $|B| \geq w$, $f(k, B)$ can be used to recover k in polynomial time.*
- *(security) For all $B \subseteq \{1, \dots, n\}$ with $|B| < w$, k is independent of $f(k, B)$.*

Note that this definition only considers security against passive attackers; we'll discuss active attacks later. This definition also assumes the existence of a dealer which initially knows the secret, applies the function f, and distributes the shares to the parties $1, \dots, n$. We also assume the dealer can communicate securely with each party, and that each pair of party members have a secure channel between them.

1 Shamir's w-out-of-n secret sharing scheme

This scheme was invented by Shamir[1]. Pick a field $\mathbb{F}$ such that $|\mathbb{F}| > |\mathcal{K}|$ and $|\mathbb{F}| > n$. Let $f_0 = x \in \mathbb{F}$.

Dealer To distribute the secret x, the dealer chooses $f_1, \dots, f_{w-1} \in \mathbb{F}$, which defines a polynomial $f(t) = f_0 + f_1 t + f_2 t^2 + \cdots + f_{w-1} t^{w-1}$. Share i is $x_i = f(i)$.

Recovery Recall that any degree ℓ polynomial is uniquely determined by $\ell + 1$ points. So if a set, B, of w users cooperate they can use their shares to recover f. They can then compute the secret $x = f(0)$. The polynomial f can be recovered using the formula

$$f(t) = \sum_{i \in B} x_i \prod_{j \in B, j \neq i} \frac{t - i}{i - j}$$

so

$$f(0) = \sum_{i \in B} x_i \prod_{j \in B, j \neq i} \frac{-i}{i - j}.$$

Note that the coefficients $c_i = \prod_{j \in B, j \neq i} \frac{-i}{i-j}$ are independent of the secret and choices of f_i, so they can be computed once and for all. With this optimization, the recovery scheme is simply $x = \sum_{i \in B} c_i x_i$.

This scheme is unconditionally secure because, for any choice of $x, x_1, \dots, x_{w-1}$, there is exactly one polynomial f such that $f(0) = x$ and $f(i) = x_i$. Since each share is an element of $\mathbb{F}$, this scheme has rate 1.

Proposition 1
Let $(x_1, \dots, x_n)$ be a Shamir w-out-of-n sharing of x and $(x'_1, \dots, x'_n)$ be a Shamir w-out-of-n sharing of x'. Then $(x_1 + x'_1, \dots, x_n + x'_n)$ is a Shamir w-out-of-n sharing of $x + x'$.

Proof: This follows immediately from the linearity of the reconstruction algorithm. □

This observation has several applications. For example, if $\ell < w$ shares are compromised, the dealer can freshen all party's shares by sending them a share of 0. Each user then sums their old and new shares and destroys their old share. At this point, the compromised shares are useless. To see why, observe that any shares the attacker learns after the update are from an independent sharing of x. Thus an attacker who learns $w-1$ original shares and $w-1$ new shares is left holding $2w-2$ independently chosen random values.

We can also consider more complicated access control rules.

DEFINITION 3 *Let $\Gamma \subseteq \mathcal{P}(\{1,\ldots,n\})$ be monotone, i.e. if $B \in \Gamma$ and $B \subseteq B'$, then $B' \in \Gamma$. Then $f : \mathcal{K} \times \{1,\ldots,n\} \to X$ is a secret sharing scheme for access structure Γ if f satisfies*

- *(robustness) For every k and $B \in \Gamma$, $f(k,B)$ can be used to recover k in polynomial time.*
- *(security) For all $B \notin \Gamma$, k is independent of $f(k,B)$.*

THEOREM 2
Every monotone access structure has a secret sharing scheme.
PROOF: Let Γ be a monotone access structure. Then there exists a monotone circuit C of n inputs such that C recognizes Γ. Since C is monotone, C can be implemented using only 2-input AND and OR gates. We build the scheme inductively on the size of the circuit. If Γ is recognized by a single AND gate, then we can use a 2-out-of-2 sharing scheme. If Γ is recognized by a single OR gate, then a 1-out-of-2 sharing scheme works.

Now suppose Γ is recognized by a k gate circuit, and the last gate is an AND gate. Let a and b be the inputs to the last AND gate. Then, by induction, there is a secret sharing scheme, f' for the access structure Γ' recognized by circuit C' which outputs a, and a scheme, f'', for the structure Γ'' recognized by circuit C'' outputting b. Note $\Gamma = \Gamma' \cap \Gamma''$. So divide x into shares x' and x'' using a 2-out-of-2 sharing scheme. Then user i's share is $(f'(x',i), f''(x'',i))$. This scheme obviously allows any valid set of users to reconstruct x. To see that is is secure, suppose $B \notin \Gamma$. WLOG, we can assume $B \in \Gamma' - \Gamma''$. Then the users in B may reconstruct x' but, by induction, x'' is independent of the shares held by users in B. Thus $x = x' \oplus x''$ is independent of their shares, too.

The proof when the last gate is an OR gate is similar, except we use a 1-out-of-2 sharing scheme. $\Box$

The scheme constructed above is not guaranteed to be efficient: share sizes can be exponential in the size of the circuit. By using other monotonic primitive gates, such as a w-out-of-n gate, better schemes may be achieved.

The schemes presented so far all have problems. When the users get together to reconstruct the secret, one party may sabotage the process by submitting a bogus share. The dealer can distribute bogus shares, and this will only be detected when the users try to recover the secret. The dealer also must be trusted not to reveal the secret. Finally, the schemes can only be used once. Afterwards, the secret is revealed. They do not permit sharing of a secret that we might want to use more than once, such as a private signing key. Error correcting codes can be used to detect saboteurs, but the technique below prevents both cheating parties and cheating dealers, so we present it instead.

2 Verifiable secret sharing

This scheme is due to Feldman[2]. Pick a prime $p = 2q + 1$, where q is also prime, and $g \in (\mathbb{Z}/p\mathbb{Z})^*$ such that $o(g) = q$.

Dealer The dealer picks a degree $w - 1$ polynomial $f(t) = \sum_{i=0}^{w-1} f_i t^i \in \mathbb{F}_q$ such that $f(0) = x$. User i's share is $x_i = f(i)$. The dealer also broadcasts g^{f_i} for all i.

Verification Each user can verify his share by checking that $g^{x_i} = \Pi(g^{f_i})^{i^j}$.

Recovery Same as in the Shamir scheme.

This scheme is not unconditionally secure: if someone can solve discrete logs, they can recover $x = f_0$ from g^{f_0}. Furthermore, a user can implicate the dealer by destroying his valid share and claiming the dealer sent him a bogus share instead. The latter problem can probably be solved by having the dealer sign the shares she distributes. Any user which claims to have received a bogus share must present a valid signature on the bogus share. However, the actual solution could not be settled on by the end of class.

References

[1] A. Shamir, *How to Share a Secret*. Communications of the ACM, 22, 1979, pp. 612–613.

[2] P. Feldman, *A practical scheme for non-interactive verifiable secret sharing*. 28th IEEE Symposium on Foundations of Computer Science, 1987, 427–437.