COMS W4261: Introduction to Cryptography.
Instructor: Prof. Tal Malkin

Lecture Notes on Secret Sharing

Abstract

These are lecture notes from the first two lectures in Fall 2016, focusing on technical material we saw for secret sharing (including a proof by reduction using the hybrid proof technique). The notes do not include the general introduction to the course and modern crypto, as well as some discussion and motivation for secret sharing. Also not covered here is Shamir's secret sharing, which we saw in the third lecture.

1 Definition

Intuitively, a secret sharing scheme allows a *dealer* to share a secret s among n parties $P_1, \ldots, P_n$, such that any *authorized* subset of parties can use all their shares to reconstruct the secret, while any other (non-authorized) subset learns nothing about the secret from their shares.

We discussed motivating scenarios for secret sharing in class, and it is also a useful tool in many larger cryptographic systems (notably, secure computation).

Secret sharing can be defined with respect to any access structure that specifies the set of authorized subsets, as long as that access structure is monotone (namely, if a subset is authorized, any larger subset should also be authorized). In class, we gave a definition for the common case of *t-out-of-n* (or *threshold* secret sharing, where authorized subsets are all those of size at least t, while sets of size less than t are not authorized.

Definition 1 (t-out-of-n secret sharing syntax and correctness)**.** *A* t-out-of-n secret sharing scheme *over message space $\mathcal{M}$ is a pair of algorithms* $(\texttt{Share}, \texttt{Reconstruct})$ *such that:*

- `Share` *is a randomized algorithm that on any input $m \in \mathcal{M}$ outputs a n-tuple of shares $(s_1, \ldots, s_n)$.*
- `Reconstruct` *is a deterministic algorithm that given a t-tuple of shares outputs a message in $\mathcal{M}$*

and satisfying the following correctness *requirement:*
$\forall m \in \mathcal{M}, \forall S = \{i_1, \ldots, i_t\} \subseteq \{1, \ldots, n\}$ *of size t,*

$$\Pr_{\texttt{Share}(m) \to (s_1,\ldots,s_n)}[\texttt{Reconstruct}(s_{i_1}, \ldots, s_{i_t}) = m] = 1$$

We discussed how we should go about defining security for such a t-out-of-n secret sharing scheme. Clearly, we need to require that any subset of size less than t cannot reconstruct the secret from its shares. But this is not sufficient: we want to require that such a subset cannot, for example figure out the first bit of the secret, or the parity (xor of all bits), or be able to use their shares together with some prior information on what the secret may be in order to deduce even more information.[1]

[1]This is a theme we will see again and again in cryptography: we prefer our definition to hold against adversaries who may or may not have arbitrary side information. This is important to make our definitions of security stronger, and also so that the security does not depend on the context or application scenarios where the primitive is being used as a component.

We ended up with two definitions, which we claimed (though didn't proof) are equivalent. These definitions capture the idea that the shares of any unauthorized subset contain no information at all about the secret. Indeed, the definitions require that these shares look exactly the same whether they came from secret m or from secret m', for any possible secrets m, m'.

Definition 2 (secret sharing security via identical distributions)**.** *A t-out-of-n secret sharing scheme* $(\mathtt{Share}, \mathtt{Reconstruct})$ *over M is perfectly secure if:*
$\forall m, m' \in \mathcal{M}$, $\forall S \subseteq \{1, \ldots, n\}$ s.t. $|S| < t$, the following distributions are identical:

$$\{(s_i | i \in S) : (s_1, \ldots, s_n) \leftarrow \mathtt{Share}(m)\}$$

$$\{(s'_i | i \in S) : (s'_1, \ldots, s'_n) \leftarrow \mathtt{Share}(m')\}$$

Recall that two distributions are identical if they give exactly the same probability for every possible value. Thus, the above definition can be restated as follows: $\forall m, m' \in \mathcal{M}$, $\forall S \subseteq \{1, \ldots, n\}$ s.t. $|S| < t$, and for any set $\alpha = (\alpha_1, \ldots, \alpha_{|S|})$, we have that

$$\Pr_{\mathtt{Share}(m) \to (s_1, \ldots, s_n)}[(s_i | i \in S) = \alpha] = \Pr_{\mathtt{Share}(m') \to (s'_1, \ldots, s'_n)}[(s'_i | i \in S) = \alpha]$$

The second (and equivalent) definition we gave is in terms of an adversarial algorithm A trying to learn information about the secret. Intuitively, we require that even if the adversary is trying to learn just *one* bit of information to differentiate between shares of m or m' (for any pair of secrets m, m'), it will fail.

Definition 3 (secret sharing security with adversaries)**.** *A t-out-of-n secret sharing scheme* $(\mathtt{Share}, \mathtt{Reconstruct})$ *over M is perfectly secure if:*
$\forall m, m' \in \mathcal{M}$, $\forall S \subseteq \{1, \ldots, n\}$ s.t. $|S| < t$, $\forall A$,

$$\Pr_{\mathtt{Share}(m) \to (s_1, \ldots, s_n)}[A((s_i | i \in S)) = 1] = \Pr_{\mathtt{Share}(m') \to (s'_1, \ldots, s'_n)}[A((s'_i | i \in S)) = 1]$$

2 A 2-out-of-2 Secret Sharing Scheme

Consider the following simple 2-out-of-2 secret sharing scheme for ℓ-bit secrets.

$\mathtt{Share}_{2-2}$: On input $m \in \{0,1\}^\ell$,

- select $s_0 \in \{0,1\}^\ell$ uniformly at random.
- set $s_1 = s_0 \oplus m$
- output (s_0, s_1)

$\mathtt{Reconstruct}_{2-2}$: On input $(s_0, s_1) \in \{0,1\}^\ell \times \{0,1\}^\ell$,

- output $s_0 \oplus s_1$

Note that in this scheme, the size of each share is the size of the secret (ℓ bits). It can be shown that this size is required (there's no scheme where each share is shorter).

Theorem 1. *The scheme* $(\texttt{Share}_{2-2}, \texttt{Reconstruct}_{2-2})$ *above is a correct and secure 2-out-of-2 secret sharing scheme over* $\mathcal{M} = \{0,1\}^\ell$.

Proof. Correctness follows immediately from the way $\texttt{Share}_{2-2}$ and $\texttt{Reconstruct}_{2-2}$ were defined: $\forall m \in \{0,1\}^\ell$,

$$\Pr_{\texttt{Share}(m)\to(s_0,s_1)}[\texttt{Reconstruct}(s_0,s_1) = m] = \Pr_{\texttt{Share}(m)\to(s_0,s_1)}[s_0 \oplus s_0 \oplus m = m] = 1$$

Security follows intuitively since each share individually is distributed uniformly at random, so does not contain any information about the secret. We prove it formally using Definition 2.

Fix any $m, m' \in \{0,1\}^\ell$, and consider a subset of one share.

In case it is the first share, since it is chosen uniformly, we have that $\forall \alpha \in \{0,1\}^\ell$,

$$\Pr_{\texttt{Share}(m)\to(s_0,s_1)}[s_0 = \alpha] = \frac{1}{2^\ell} = \Pr_{\texttt{Share}(m')\to(s'_0,s'_1)}[s'_0 = \alpha]$$

In case it is the second share, we have that $\forall \alpha \in \{0,1\}^\ell$,

$$\Pr_{\texttt{Share}(m)\to(s_0,s_1)}[s_1 = \alpha] = \Pr_{\texttt{Share}(m)\to(s_0,s_1)}[s_0 \oplus m = \alpha] = \Pr_{\texttt{Share}(m)\to(s_0,s_1)}[s_0 = m \oplus \alpha] = \frac{1}{2^\ell}$$

Similarly,

$$\Pr_{\texttt{Share}(m')\to(s'_0,s'_1)}[s'_1 = \alpha] = \Pr_{\texttt{Share}(m')\to(s'_0,s'_1)}[s'_0 = m' \oplus \alpha] = \frac{1}{2^\ell}$$

and again both probabilities are equal. This completes the proof. □

t-out-of-t Secret Sharing Scheme

The above scheme is sometimes referred to as "additive secret sharing", since the xor operation is addition mod 2. We note that additive secret sharing can easily be extended to any t-out-of-t threshold secret sharing. The sharing algorithm chooses t strings uniformly at random subject to the requirement that their xor is the secret (this can be done by choosing $t-1$ strings uniformly at random, and setting the last string to be the xor of all previous ones and the secret). The reconstruction algorithm simply xors the shares. We leave the proof of correctness and security as an exercise.

3 A 2-out-of-n Secret Sharing Scheme

We will see next time a direct way to get any t-out-of-n secret sharing. Here, we will build a 2-out-of-n secret sharing from any 2-out-of-2 secret sharing. This will allow us to show the first instance of a reduction, and also allow us to introduce the hybrid proof techniques – both useful tools that we will see many more time in this class.

3.1 Reductions in Cryptography: General Comments

The notion of *reduction* is a central notion in cryptography, which we describe here informally. In general, to show that $P \Rightarrow Q$ for some cryptographic primitives P, Q, we typically show a construction of Q that calls P as a subroutine (so, we assume P is given, and we construct Q out of it). We then prove correctness and security of our construction of Q, assuming the correctness and security of the underlying P.

The security proof typically has the following structure. To prove that security of P implies security of Q, we prove the contrapositive: If Q is not secure, then we show P is not secure either. So, we start by assuming we are given an adversary A breaking the security of our constructed Q. We use it to construct an adversary B, which can call A as a subroutine, and breaks P.

We will now see how to apply this general approach to our current example.

3.2 Building 2-out-of-n Secret Sharing from 2-out-of-2 Secret Sharing

Suppose we are given a 2-out-of-2 secret sharing scheme ($\texttt{Share}_{2-2}, \texttt{Reconstruct}_{2-2}$) (e.g., the one we showed above). We want to use it to construct a 2-out-of-n secret sharing scheme, namely sharing the secret among n parties, so that any two of them can reconstruct, but any single party learns nothing about the secret.

A first idea is to just share the secret via a fresh 2-out-of-2 sharing for every possible pair of parties. The new share of each party consists of $n-1$ 2-out-of-2 shares (one for each of the other parties). When two parties get together, they can use the two corresponding 2-out-of-2 shares to reconstruct the secret.

The above idea works (satisfies correctness and security). However, the size of the share for each party is $(n-1)$ times the 2-out-of-2 share size, so at least $(n-1)\ell$ for ℓ bit secrets. Can we do better?

In the above, we shared the secret afresh n times. The idea for the improved 2-out-of-n scheme is to have a smaller number of 2-out-of-2 sharings, as long as we have the property that any pair of parties has at least one complete pair of two corresponding shares (or "two halves" from the same sharing). We can achieve this property with only $\log n$ 2-out-of-2 sharings, where each party gets one half of each of the $\log n$ pairs of shares, corresponding to the binary representation of the party's index. The resulting share size will be $\log n$ times the 2-out-of-2 share size, so we can do this with $(\log n)\ell$ share size for ℓ bit secrets. The scheme is specified below.

$\texttt{Share}_{2-n}$: On input $m \in \{0,1\}^\ell$,

- For $k = 1$ to $\log n$
  - Run $\texttt{Share}_{2-2}(m) \to (s_0^k, s_1^k)$
- For $i = 0$ to $n-1$, if binary representation of i is $i_1 \dots i_{\log n}$, set $S_i = (i, s_{i_1}^1, \dots, s_{i_{\log n}}^{\log n})$
- Output $(S_0, \dots, S_{n-1})$.

$\texttt{Reconstruct}_{2-n}$: On input (S_i, S_j),

- Consider the binary representations of the indices $i = i_1 \dots i_{\log n}$ and $j = j_1 \dots j_{\log n}$.

- Find a bit position k where they differ, namely $i_k \neq j_k$ (thus, $s^k_{i_k} = s^k_0, s^k_{j_k} = s^k_1$ or vice versa).
- Run $\texttt{Reconstruct}_{2-2}(s^k_0, s^k_1)$ and output the same.

Theorem 2. *If* $(\texttt{Share}_{2-2}, \texttt{Reconstruct}_{2-2})$ *is a correct and secure 2-out-of-2 secret sharing scheme over* $\mathcal{M} = \{0,1\}^\ell$*, then the scheme* $(\texttt{Share}_{2-n}, \texttt{Reconstruct}_{2-n})$ *specified above is a correct and secure 2-out-of-n secret sharing scheme over* $\mathcal{M}$*.*

Proof. Correctness: $\forall m \in \{0,1\}^\ell$, $\forall i \neq j \in \{0,\dots,n-1\}$,

$$\Pr_{\texttt{Share}_{2-n}(m)\to(S_0,\dots,S_{n-1})}[\texttt{Reconstruct}_{2-n}(S_i,S_j) = m] =$$
$$= \Pr_{\texttt{Share}_{2-2}(m)\to(s^k_0,s^k_1)}[\texttt{Reconstruct}_{2-2}(s^k_0,s^k_1) = m] = 1$$

where the first equality follows from the construction and the fact that any indices $i \neq j$ differ in at least one position k, and the second equality follows from the correctness of $(\texttt{Share}_{2-2}, \texttt{Reconstruct}_{2-2})$.

For security, we could use the definition via identical distributions (Definition 2) fairly easily. However, for pedagogical reasons we will prove security via the adversary based definition (Definition 3). As discussed, this will allow us to introduce a new proof technique and common proof structure.

As described in Section 3.1, we start by assuming that the 2-out-of-n scheme above is not secure. This means (by negation of Definition 3) that $\exists m, m' \in \{0,1\}^\ell$, $\exists i \in \{0,\dots,n-1\}$, $\exists A$, such that

$$\Pr_{\texttt{Share}_{2-n}(m)\to(S_1,\dots,S_n)}[A(S_i) = 1] \neq \Pr_{\texttt{Share}_{2-n}(m')\to(S'_1,\dots,S'_n)}[A(S'_i) = 1] \tag{1}$$

Our goal is to use this to construct an adversary B that breaks the 2-out-of-2 scheme.

In Equation (1) above, we have two distributions such that when A is called on one it outputs 1 with a different probability than when it's called on the other. Recall that in the lefthandside distribution (which we will name H^0), we have $S_i = (s^1_{i_1}, \dots, s^{\log n}_{i_{\log n}})$, where each (s^k_0, s^k_1) is the output of $\texttt{Share}_{2-2}(m)$. In the righthandside distribution (which we will name $H^{\log n}$), we have $S'_i = (s'^1_{i_1}, \dots, s'^{\log n}_{i_{\log n}})$, where each (s'^k_0, s'^k_1) is the output of $\texttt{Share}_{2-2}(m')$.

We will define intermediate distributions H^j ("hybrids") where for each share, the first j components are taken from m', while the rest are taken from m. That is, for every $j \in \{0,\dots,\log n\}$, we define

$$H^j = \{\forall k \in \{1,\dots,\log n\}\ \texttt{Share}^k_{2-2}(m) \to (s^k_0, s^k_1),\ \texttt{Share}^k_{2-2}(m') \to (s'^k_0, s'^k_1) : \\ (s'^1_{i_1}, \dots, s'^j_{i_j}, s^{j+1}_{i_{j+1}}, \dots, s^{\log n}_{i_{\log n}})\}$$

Using this notation, Equation (1) can be written as: $\Pr[A(H^0) = 1] \neq \Pr[A(H^{\log n}) = 1]$. It follows that there must exist a $j \in \{1,\dots,\log n\}$ such that

$$\Pr[A(H^{j-1}) = 1] \neq \Pr[A(H^j) = 1] \tag{2}$$

(otherwise, if all adjacent hybrids produce equal probabilities, the end hybrids would also have equal probabilities).

Equation (2) says that A outputs 1 with different probabilities when applied to

$$(s'^{1}_{i_1}, \ldots, s'^{j-1}_{i_{j-1}}, s^{j}_{i_j}, s^{j+1}_{i_{j+1}}, \ldots, s^{\log n}_{i_{\log n}})$$

vs. when applied to

$$(s'^{1}_{i_1}, \ldots, s'^{j-1}_{i_{j-1}}, s'^{j}_{i_j}, s^{j+1}_{i_{j+1}}, \ldots, s^{\log n}_{i_{\log n}})$$

We now have two "adjacent" hybrids, differing in only one location (j), that A behaves differently on. We are finally ready to define B, that will attack the 2-out-of-2 sharing, by "plugging it in" that location. Specifically, we define B as follows (where i, j, m, m' are all hard-coded into B):

B: on input $\mathbf{s}$,

- For $k = 1, \ldots, j-1$, run $\texttt{Share}_{2-2}(m') \to (s'^{k}_0, s'^{k}_1)$.
- For $k = j+1, \ldots, \log n$, run $\texttt{Share}_{2-2}(m) \to (s^{k}_0, s^{k}_1)$
- Set $S_i = (s'^{1}_{i_1}, \ldots, s'^{j-1}_{i_{j-1}}, \mathbf{s}, s^{j+1}_{i_{j+1}}, \ldots, s^{\log n}_{i_{\log n}})$
- Run $A(S_i)$ and output the same.

Now it is not hard to see that for the same m, m', i, j from above we have that

$$\Pr_{\texttt{Share}_{2-2}(m) \to (s_0, s_1)}[B(s_{i_j}) = 1] = \Pr[A(H^{j-1}) = 1]$$

while

$$\Pr_{\texttt{Share}_{2-2}(m') \to (s'_0, s'_1)}[B(s'_{i_j}) = 1] = \Pr[A(H^{j}) = 1]$$

By Equation 2 these two are not equal, and thus B breaks the security of the 2-out-of-2 secret sharing scheme (as per Definition 3).

We have shown that if the security of our 2-out-of-n is broken, then so is the security of the underlying 2-out-of-2 scheme (contradiction). This completes the proof.

$\square$