CMSC 858K — Introduction to Secure Computation November 15, 2013

Lecture 30

Lecturer: Jonathan Katz *Scribe(s): Xiong Fan*

1 Verifiable Secret Sharing (VSS)

Dealer specifies a degree t polynomial g, and parties P_i gets $g(i)$. The description of functionality $F_{vss}^{subshare}$ is:

After a sharing of some value a in set, i.e., each party holds $a_i = f(i)$ for some f of degree w with $f(0) = a$, share a_i.

The properties of *Reed-Solomon* codes are following: Assume the distance between $a' \in \mathbb{F}^n$ and a codeword is less than t:

- There is a linear transformation τ of a' that compute the syndrome of a'.
- From the syndrome $s \in \{0,1\}^{2t}$, it is possible to compute an error vector $e \in \mathbb{F}^n$, such that $a' - e$ is a codeword.

Protocol for $F_{vss}^{subshare}$:

1. Each P_i invokes VSS on their shares a_i using a degree t polynomial g_i.
2. Each party P_i applies linear transformation T locally to $g_1(i), g_2(i), ..., g_n(i)$ to get $s_1(i), ..., s_{2t}(i)$.
3. Each party sends $s_1(i), ..., s_{2t}(i)$ to all other parties.
4. For each $j = 1, ..., 2t$, use shares $s_j(1), ..., s_j(n)$ and Reed-Solomon decoding to recover $s_j(a) = s_j$.
5. Each party locally uses s to compute e.
6. For indices j, where e is a_j, P_i outputs $g_j(i)$.
7. For indices j, where e is non-zero, all parties send $g_j(i)$ to each other, and use Reed-Solomon decoding to recover $g_j(0)$, then output $g_j(0) = e_j$.

In the beginning of the protocol, each party P_i holds a share a_i of a, while at the end of the protocol, each party P_i holds the values $a_{1i}, a_{2i}, ..., a_{ni}$, such that $a_{ij} = g_i(j)$, for $g_i(0) = a_i$. We use notation (a) to denote the shares of a. Roughly speaking, the process of the protocol is the following:

- First, the distance between $a'_1, ..., a'_n$ and codewords $a_1, ..., a_n$ are less than t.
- Compute the shares for $a'_1, ..., a'_n$, i.e., $(a'_1), ..., (a'_n)$.
- Using linear transformation to compute the syndrome $(s_1), ..., (s_{2t})$.
- Exchange shares of

2 Evaluation

Protocol of F^k_{eval}:

1. All parties have shares a_i of some value a (i.e., $a_i = f(i)$ for f of degree t such that $f(0) = a$)
2. Compute $f(k)$, where $f(k)$ is a linear function of $a_1, ..., a_n$.
3. All parties invoke F^{share}_{vss}, so now parties have shares $(a_1), ..., (a_n)$.
4. All parties locally apply a linear transformation to their shares, i.e., $(f(k))$.
5. All parties exchange their shares of $(f(k))$.
6. Decode using Reed-Solomon decoding to get $f(k)$.

The functionality of F^{mult}_{vss} is that all parties have shares $(a), (b)$ for a, b known to some dealer P_1, and parties end up with $(a \cdot b)$.

Protocol for F^{mult}_{vss}:

1. P_1 knows $A(x), B(x)$ used to share a, b.
2. P_1 computes $D(x) = A(x) \cdot B(x)$ $(degD = 2t)$.
3. P_1 chooses $D_1(x), ..., D_t(x)$ $(degD_i = t)$ random polynomials subject to $L(x) = D(x) - \sum_{k=1}^{t} D_i(x) \cdot x^i$ has degree t.
4. P_i uses F_{vss} to share $L(x), D_1(x), ..., D_t(x)$.
5. Each party P_i checks if $C(i) = a_i \cdot b_i - \sum_{k=1}^{t} i^k D_k(i)$, if not, broadcast complaint.
6. If there was a complaint by P_j, use F^j_{eval} to reconstruct $a_j, b_j, C(j), D_1(j), ..., D_t(j)$.
7. If any complaint was justified, parties reconstruct a, b.
8. Otherwise, parties output shares (c_i).