CSA E0 312: Secure Computation 26th August 2015

[Lecture 6]

Instructor: Arpita Patra *Submitted by: Marilyn George*

In this lecture, we continue to discuss the properties of the (n,t) Shamir Secret Sharing and move to the BGW protocol for secure arithmetic circuit evaluation and its proof of its security. We first show that (n,t) Shamir Secret Sharing satisfies linearity property. That is, given Shamir sharing of two secrets among a set of n parties, the parties can locally add their shares to obtain (n,t) Shamir sharing of the sum of the secrets. A similar result for the operator 'multiplication by constant' is also shown. No communication is thus required to perform linear combination of shared secrets. Only local computation is enough. BGW protocol will use the linearity properties of (n,t) Shamir sharing.

1 Linearity Properties of (n,t) Shamir Secret Sharing

1.1 Linearity of addition

Given players $P_1, P_2, \ldots, P_n$, the publicly known points $\alpha_1, \alpha_2, \ldots, \alpha_n$ and the Shamir shares of two secrets, say a and b using polynomials, say $f_a(x)$ and $f_b(x)$ of degree at most t, adding the corresponding shares of both would give the shares of the secret $a+b$, using a new polynomial $f_{a+b}(x)$ which is same as the sum polynomial $f_a(x) + f_b(x)$. To prove this, we consider the following:

$$\begin{aligned} f_a(x) = &\quad a + a_1 \cdot x + a_2 \cdot x^2 + \quad ... + a_t \cdot x^t \\ f_b(x) = &\quad b + b_1 \cdot x + b_2 \cdot x^2 + \quad ... + b_t \cdot x^t \end{aligned}$$

For any party P_i, his shares of $f_a(x)$ and $f_b(x)$ will consist of $f_a(\alpha_i)$ and $f_b(\alpha_i)$ respectively, as per Shamir Secret Sharing. Now, P_i computes the sum:

$$f_a(\alpha_i) + f_b(\alpha_i) = (a+b) + (a_1 + b_1) \cdot \alpha_i + (a_2 + b_2) \cdot \alpha_i^2 + .. + (a_t + b_t) \cdot \alpha_i^t$$

The sum is now the value of a polynomial $f_{a+b}(x) = (a+b) + (a_1+b_1)x + \ldots, (a_t+b_t)x^t$ with constant term equal to the secret $(a+b)$ evaluated at α_i. Since the coefficients of both the original polynomials are random, their sums are also random. Additionally the resulting random polynomial also has degree at most t. This implies that the sum shares define *valid* (n,t) Shamir sharing of the secret $(a+b)$. Hence each party can *locally* compute the shares of the sum of the secrets. Finding (n,t) Shamir sharing of sum of two shared secrets is absolutely *communication free.*

1.2 Linearity of constant multiplication

In a similar setting, multiplication by a publicly-known constant c is also linear. We consider the Shamir shared polynomial $f_a(x)$ of degree at most t, as in the previous example. Now for any party P_i, his share of $f_a(x)$ will be $f_a(\alpha_i)$. On multiplying the share with the constant c, P_i gets:

$$c \cdot f_a(\alpha_i) = c \cdot a + c \cdot a_1 \cdot \alpha_i + c \cdot a_2 \cdot \alpha_i^2 + ... + c \cdot a_t \cdot \alpha_i^t$$

which corresponds to the polynomial $f_{ca}(x) = (c \cdot a) + (c \cdot a_1)x, \ldots, (c \cdot a_t)x^t$ (with the constant term equal to the secret $(c \cdot a)$) evaluated at α_i.

Since the coefficients of $f_a(x)$ are random, the products obtained on multiplying with a fixed constant are also random. Additionally the resulting random polynomial also has degree at most t. This implies that the c times the shares are *valid* (n, t) Shamir shares of the secret $(c \cdot a)$. Hence each party can *locally* compute the shares of the product of the secret with a publicly-known constant. Finding (n, t) Shamir sharing of c times a shared secret is absolutely *communication free.*

From the above properties we have that the linear combination of shared secrets with publicly-known constants can be computed locally by the parties, and hence is also *communication free.* i.e. If $r_1, r_2, ..r_n$ are publicly-known constants and $a_1, a_2, ..a_n$ are secret shared, then the parties can compute the shares of $r_1 \cdot a_1 + r_2 \cdot a_2 + .. + r_n \cdot a_n$ by local computation.

1.3 Non-linearity of multiplication

In the same scenario, however the product of the shares of two secrets do not result in a valid (n, t) Shamir sharing of the product of the secrets. The product of the shares are

$$g(\alpha_i) = f_a(\alpha_i) \cdot f_b(\alpha_i)$$

for some polynomial $g(x)$. We will have $n = 2t + 1$ points (corresponding to $n \geq 2t + 1$ parties) which will define a polynomial of degree at most $2t$. Now consider the product polynomial $p(x) = f_a(x) \cdot f_b(x)$. We know that the degree of $p(x)$ is at most $2t$, since both $f_a(x)$ and $f_b(x)$ are of degree at most t. We also have that $p(x)$ and $g(x)$ coincide at the n points, $\alpha_1, \alpha_2, \ldots, \alpha_n$. Since these two polynomials of degree at most $2t$ coincide at $2t + 1$ points, they are the same polynomial, or $g(x) =$ the product polynomial. This polynomial will have constant term ab. Although the constant term is what we require, note that the resulting polynomial $g(x)$ is always a *reducible polynomial* over the field i.e. it always has non-constant polynomial factors. This implies that the distribution of the product polynomial $g(x)$ does not induce a random distribution of polynomial of degree at most $2t$ over F_p. The reason is the former distribution excludes the irreducible polynomials. Additionally $g(x)$ has degree at most $2t$, and hence the product shares of the parties does not induce an (n, t) Shamir sharing of the product secret. We now have that multiplication of two Shamir shared secrets is not free.

We show a specific example where the multiplication of the shares of two Shamir shared secrets do not produce the shares of the Shamir shared product. Consider $n = 3$ and $t = 1$ over field F_5, which satisfies the condition $p > n$.

- Let $\alpha_1 = 1$, $\alpha_2 = 3$, $\alpha_3 = 4$.
- Let $a = 3$ and $f_a(x) = 2x + 3$. Similarly $b = 2$ and $f_b(x) = x + 2$. Both these polynomials have degree at most $t = 1$.

- We assume that P_2 is corrupt, and P_1, P_3 are honest parties. Then the shares for each player for secret a are: $P_1 \to 0$, $P_2 \to 4$ and $P_3 \to 1$. Similarly for b: $P_1 \to 3$, $P_2 \to 0$ and $P_3 \to 1$

We consider only the adversary's view at the present situation.

- Consider the share received of a, 4. From this share and the fact that $\alpha_2 = 3$, the adversary can compute that the possible polynomials chosen of degree at most 1 are: $\{4, x+1, 2x+3, 3x, 4x+2\}$. From this we see that each of the secrets in the field $\{0,1,2,3,4\}$ are equally likely to be the constant term.

- Similarly with the share of b, the possible polynomials are $\{0, x+2, 2x+4, 3x+1, 4x+3\}$. Again all the secrets are equally likely.

Hence, given the (n,t) Shamir shares so far, all the secrets are equally likely by the property of Shamir sharing. Now we consider that each party multiplies their shares. Then P_2's share of the product polynomial is 0. He can compute the 25 possible polynomials with degree at most 2 which could give rise to this share. They are:

$0x^2+0x+0$	x^2+0x+1	$2x^2+0x+2$	$3x^2+0x+3$	$4x^2+0x+4$
$0x^2+1x+2$	x^2+1x+3	$2x^2+1x+4$	$3x^2+1x+0$	$4x^2+1x+1$
$0x^2+2x+4$	x^2+2x+0	$2x^2+2x+1$	$3x^2+2x+2$	$4x^2+2x+3$
$0x^2+3x+1$	x^2+3x+2	$2x^2+3x+3$	$3x^2+3x+4$	$4x^2+3x+0$
$0x^2+4x+3$	x^2+4x+4	$2x^2+4x+0$	$3x^2+4x+1$	$4x^2+4x+2$

Now he calculates the polynomials of degree at most 2 which are products of the possible polynomials that he had for secrets a and b. i.e. Products of one polynomial from $\{4, x+1, 2x+3, 3x, 4x+2\}$ with one from $\{0, x+2, 2x+4, 3x+1, 4x+3\}$. This list of products is then:

$0x^2+0x+0$	x^2+0x+1	$2x^2+0x+2$	$3x^2+0x+3$	$4x^2+0x+4$
$0x^2+1x+2$	x^2+1x+3	$2x^2+1x+4$	$3x^2+1x+0$	$------$
$0x^2+2x+4$	x^2+2x+0	$2x^2+2x+1$	$------$	$4x^2+2x+3$
$0x^2+3x+1$	x^2+3x+2	$------$	$3x^2+3x+4$	$4x^2+3x+0$
$0x^2+4x+3$	$------$	$2x^2+4x+0$	$3x^2+4x+1$	$4x^2+4x+2$

Due to the absence of 4 polynomials in this list we find that the probability of the secret being {0, 1, 2, 3, 4} respectively is now $\{\frac{5}{21}, \frac{4}{21}, \frac{4}{21}, \frac{4}{21}, \frac{4}{21}\}$. Then the secrets are no longer equally likely, and the adversary has gained some additional information by being able to see just one share. In this manner, using the product polynomial is not truly random and will leak information to the adversary. Hence, this cannot be used as a valid Shamir secret sharing.

2 BGW Protocol for Secure Circuit Evaluation

We now move on to BGW protocol for secure arithmetic circuit evaluation with information theoretic security and with n parties tolerating a semi-honest adversary who may corrupt t

out of the n parties. An arithmetic circuit consists of gates, each having two inputs and one output. For simplicity, we assume ith input to the circuit is the secret input value supplied by the party P_i. As shown in Figure 1, each gate represents a certain operation, and the output the circuit is the output of the computation, y.

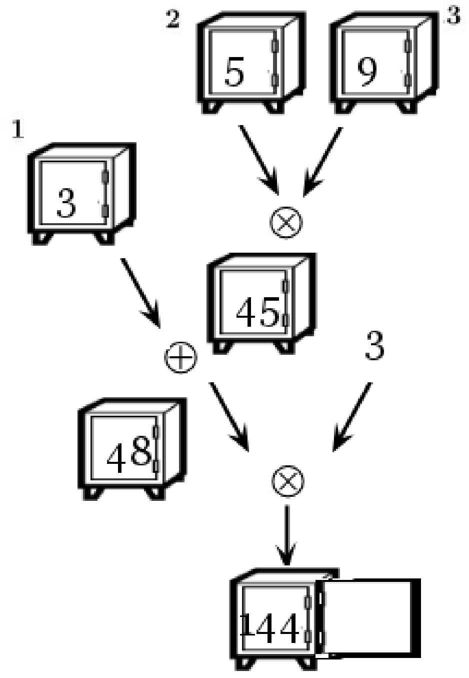

Figure 1: An example arithmetic circuit with $n = 3$, $t = 1$.

- *Addition gate:* It adds the two inputs and outputs the sum.
- *Multiply by constant gate:* One of its inputs is a predefined public constant, and the output is the product of the other input with the known constant.
- *Multiplication gate:* It multiplies the two inputs and output the product.

The procedure for secure arithmetic circuit evaluation is as follows:

1. Each party P_i (n,t)-secret-shares its input, say x_i. This sharing is denoted by the closed boxes in the diagram.
2. The parties together find (n,t) secret sharing of each intermediate gate output value. That is, the following invariant is maintained for every gate. If the inputs to the gates are (n,t)-secret-shared, then the gate is evaluated in a way that the output is remains (n,t)-secret-shared.
3. When the (n,t) secret sharing of the final gate output value is computed, the parties together reconstruct the output using the reconstruction phase of Shamir sharing i.e. by exchanging the shares with each other. This is denoted by the open box, since the final output is visible to all parties.

Specifically, given the secret input values of the parties, our goal is to output the final result of the computation, y and nothing beyond must be leaked during the computation. As described above, the computation of the circuit is done in such a way that every circuit wire is assigned values where the values themselves remain (n, t)-shared. We can assume any topological ordering for the circuit, and proceed to evaluate the gates in that order. A topological ordering will ensure that during the evaluation of a gate its input wires are have been assigned their values. If both the inputs of the gate have already been assigned or computed, we can perform the required operation and assign the answer to the output of the gate. We continue till the final output has been computed. Note that all the operations are performed on the *shared secrets*. We maintain the invariant that *if the inputs of a gate are* (n, t) *Shamir shared, the output will also be* (n, t) *Shamir shared*. We have seen in the previous sections that addition and multiplication by a constant of shared secrets is free and locally computable by each of the parties. Hence the 'linear' gates are non-interactive due to the linearity of Shamir Sharing. However, the multiplication gates are non-linear and will require an interactive technique called *degree reduction* for secure evaluation.

2.1 Secure Multiplication Gate Evaluation

We have n parties $P_1, P_2, .., P_n$ who have shares of the secrets a and b with respect to the polynomials $f_a(x)$ and $f_b(x)$. Then as we have seen before each party can compute shares of the product polynomial $g(x)$ by multiplying their shares of $f_a(x)$ and $f_b(x)$. Let each of these shares be $z_i = f_a(\alpha_i) \cdot f_b(\alpha_i) = g(\alpha_i)$ for the party P_i. This polynomial has constant term ab, but it is not random or of degree t. Now, we know that any point on a polynomial of degree $2t$ can be expressed as a linear combination of $2t + 1$ points on the polynomial, given the public recombination vector. Each P_i shares their z_i with all the other parties according to (n, t) Shamir Sharing. Now, the parties can perform a linear recombination on these $n = 2t + 1$ points of the polynomial to generate an (n, t) Shamir sharing of $g(0) = ab$ as follows.

$$g(x) = \sum_{i=1}^{n} z_i \cdot \delta_i(x), \text{ where } \delta_i(x) = \prod_{j \in [1,\dots,n], j \neq i} \frac{x - j}{i - j}$$

$$ab = g(0) = \sum_{i=1}^{n} z_i \cdot \delta_i(0) = \sum_{i=1}^{n} z_i \cdot r_i$$

where $(r_1, r_2, .., r_n)$ is the public *recombination vector*. Hence, the multiplication protocol is complete. Note that this protocol is secure because our adversary is semi-honest and cannot actively interfere with the protocol steps. This technique is called *degree reduction* because its starts with the product secret ab shared using a non-random polynomial of degree at most $2t$ and generates its (n, t)-sharing using a random polynomial of degree at most t.

2.2 Proof of Security

The circuit output will always be computed correctly since the adversary is semi-honest. The privacy of the circuit evaluation is intuitive because:

- No inputs of the honest parties are leaked.

- No intermediate value is leaked, as the computations are always performed on Shamir-shared secrets.

We will formalize this notion of security using our Real World / Ideal World paradigm and constructing a simulator SIM. The adversary's real world view is as follows:

1. **At the outset:** Input and random coins.
2. **Input-sharing and multiplication gate computation:** The adversary sees t shares of the inputs of the honest parties. Additionally he sees the t shares of the honest parties product-share during the computation of the multiplication gate. He sees t values distributed uniformly at random in F_p^t, irrespective of the values shared, as we have seen in the previous lecture. Since none of the other gates require interaction, he learns no information during their computation.
3. **Output Reconstruction:** He sees the shares of the honest parties corresponding to the correct circuit output y.

We have to prove that this view of the adversary, denoted by the random variable $\{\mathsf{View}_i^{\mathsf{Real}}\}_{P_i \in C}$, leaks nothing beyond the inputs/outputs of the corrupted parties i.e. 'the allowed values' with respect to the ideal world scenario where a Trusted Third Party does all the computation. We construct the Simulator, which behaves as follows during the above three phases:

1. **At the outset:** The simulator has the input, output (of the corrupted parties) and random coins.
2. **Input-sharing and multiplication gate computation:** Sample t random shares and give them to the adversary on behalf of the honest parties.
3. **Output Reconstruction:** The simulator knows the shares of the corrupted parties and the output y. This gives it a total of $t+1$ shares of a polynomial of degree at most t. It can reconstruct the polynomial, compute the correct shares of $t+1$ honest parties and send them to the adversary.

This simulator generates the view of the adversary in the ideal world - $\{\mathsf{View}_i^{\mathsf{Ideal}}\}_{P_i \in C}$, using the inputs/outputs of the corrupted parties. We now see that the simulation is perfectly indistinguishable from the adversary's view in the real world:

- The honest parties do not interact with the adversary for Step 1, neither does the simulator.
- In Step 2, the adversary sees t random values from F_p^t, as discussed above, in both cases.
- Given the output and t shares of the corrupted parties, the polynomial and hence the shares of the honest parties are uniquely determined. Note that the points of evaluation of the function α_is, which are used to compute the shares, are publicly known.

Since the Steps 1-3 are perfectly indistinguishable in the real and simulated view for a semi-honest adversary with unbounded power, we have proved that the protocol is information theoretically secure in semi-honest adversary setting.