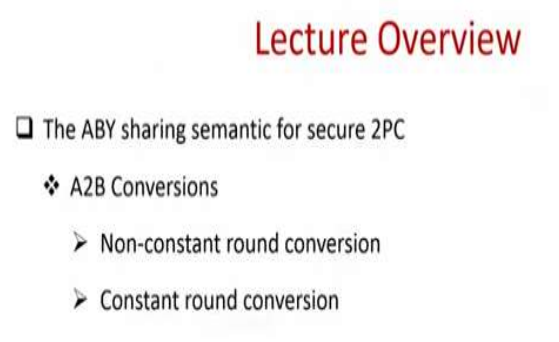
Secure Computation: Part I
Prof. Ashish Choudhury
International Institute of Information Technology, Bengaluru

Lecture - 56
The ABY Conversions Continued.

(Refer Slide Time: 00:33)

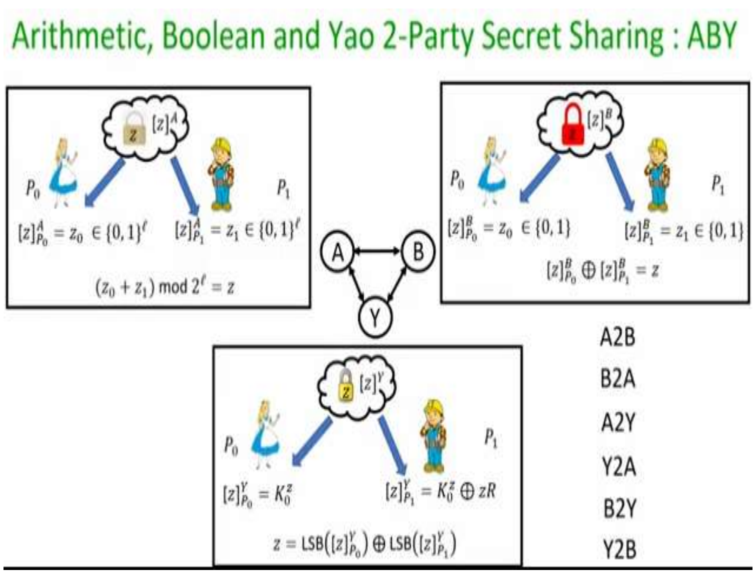


Lecture Overview

- ❑ The ABY sharing semantic for secure 2PC
  - ❖ A2B Conversions
    - ➢ Non-constant round conversion
    - ➢ Constant round conversion

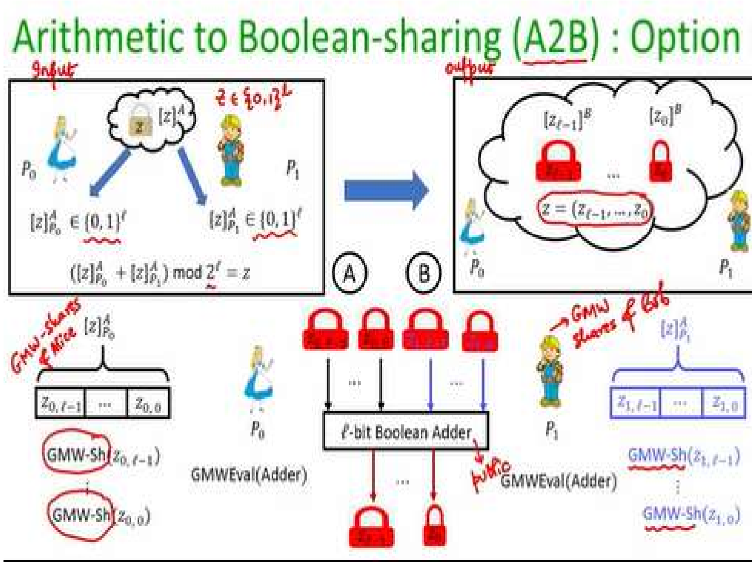
Hello everyone. Welcome to this lecture. So, we will continue our discussion on ABY conversions. And, we will see in today's lecture the conversion from the arithmetic world to the Boolean world and we will see two versions of the conversion, one which requires a constant round of interaction and one which requires a non-constant round of interaction.

(Refer Slide Time: 00:55)


Arithmetic, Boolean and Yao 2-Party Secret Sharing : ABY
P_0
P_1
$[z]^A$
$[z]^A_{P_0} = z_0 \in \{0,1\}^\ell$
$[z]^A_{P_1} = z_1 \in \{0,1\}^\ell$
$(z_0 + z_1) \bmod 2^\ell = z$
$[z]^B$
$[z]^B_{P_0} = z_0 \in \{0,1\}$
$[z]^B_{P_1} = z_1 \in \{0,1\}$
$[z]^B_{P_0} \oplus [z]^B_{P_1} = z$
A
B
Y
$[z]^Y$
$[z]^Y_{P_0} = K_0^z$
$[z]^Y_{P_1} = K_0^z \oplus zR$
$z = \text{LSB}([z]^Y_{P_0}) \oplus \text{LSB}([z]^Y_{P_1})$
A2B
B2A
A2Y
Y2A
B2Y
Y2B

So, just to recap, we have identified three different representations in which the values can be secret shared and computations can be performed and we are interested to find mechanisms to switch between this sharing semantics, secret Sharing semantics without disclosing the actual shared values.

(Refer Slide Time: 01:22)



And we had already seen three conversions; we will now try to go through the remaining conversions. So, the focus for this lecture will be the arithmetic to Boolean conversion. So, this is the input. Where there is an l bit number, which is arithmetic shared as per the GMW semantics, namely the share for Alice is an l bit number, the share for Bob is a l bit number. Such that, if we add those l bit numbers modular 2 power l, the value is Z.

And output that we desire here is the GMW sharing for the individual bits of Z. So, the bit representation of Z is Z_0 to Z_{l-1} from LSB to MSB, we require that without disclosing the bits of Z. The individual bits should be now available in a secret shared fashion between Alice and Bob as per the GMW representation. So, let us first see the conversion which requires a non-constant number of rounds.

So, what Alice can do is the following. So, these are the bits of her GMW shares and in the same way these are the GMW shares of Bob. And we know that there is an l bit Boolean adder circuit publicly known, which can take the inputs as Alice's GMW share and Bob's GMW share and bit by bit it can add them and produce the bits of Z as output. We know that, this public adder circuit is, this adder circuit is publicly no.

So, the idea behind this conversion is to let Alice and Bob evaluate this circuit as per the GMW sharing semantic, without disclosing their inputs. So, their inputs are the GMW shares, we want that they should not disclose their GMW shares to each other, but still, they should evaluate this circuit securely and obtain the output of this circuit in a secret shared fashion. How this can be done?

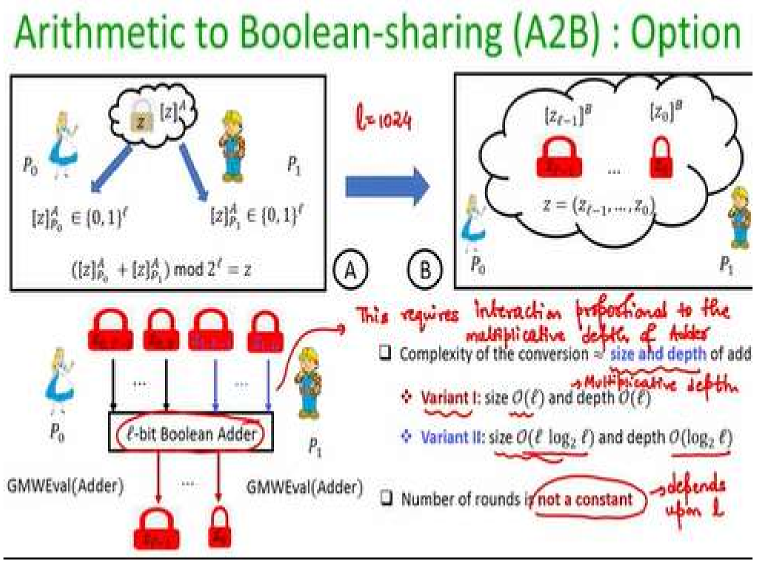
Well, we can ask Alice to act as a dealer invokes instances of GMW secret sharing to secret share her bits of GMW shares. And remember assuming a PRF key setup or AES keys setup the secret sharing protocols or instances can be executed in a non-interactive fashion. Namely, even without talking to Bob, what Alice can ensure is that her bits of the GMW shares are available in a secret shared fashion as per the GMW Boolean semantic.

And in the same way Bob can do the following, he can act as a dealer and execute instances of GMW sharing protocol to secret share the bits of his GMW arithmetic shares. Again, assuming an AES key setup, these instances of GMW sharing protocol can be executed in a non-interactive fashion. And now, the inputs for this l bit Boolean adder circuit are available in a secret shared presentation.

What Alice and Bob can do is you can evaluate this adder circuit over the secret shared inputs as per the GMW procedure. And that will provide the output of this adder circuit in a secret shared presentation. And what is output of this circuit? The output of the circuit is nothing but the bits of Z, which are now made available to the two parties in the GMW Boolean sharing semantics.

So, in essence this is the same conversion which we had seen for converting an arithmetic secret shared value into your secret shared value, except that now, the bits of GMW shares are secret shared as per the, except that the bits of the GMW shares are, shared using the Boolean sharing protocol and not yao sharing protocol and the adder circuit is evaluated in the GMW domain and not in the yao domain.

(Refer Slide Time: 06:15)



Arithmetic to Boolean-sharing (A2B) : Option

This requires Interaction proportional to the multiplicative depth of Adder

- ❑ Complexity of the conversion ≈ size and depth of add
  - ❖ Variant I: size $O(\ell)$ and depth $O(\ell)$ → Multiplicative depth
  - ❖ Variant II: size $O(\ell \log_2 \ell)$ and depth $O(\log_2 \ell)$
- ❑ Number of rounds is not a constant → depends upon ℓ

$[z]^A$
P_0
P_1
$[z]^A_{P_0} \in \{0,1\}^\ell$
$[z]^A_{P_1} \in \{0,1\}^\ell$
$([z]^A_{P_0} + [z]^A_{P_1}) \bmod 2^\ell = z$
$\ell = 1024$
A
B
$[z_{\ell-1}]^B$
$[z_0]^B$
$z = (z_{\ell-1}, \dots, z_0)$
ℓ-bit Boolean Adder
GMWEval(Adder)
GMWEval(Adder)

Now, what will be the complexity of this conversion? Well, the complexity of the conversion depends upon the size and depth of this adder circuit, this is a Boolean adder circuit. It is not a simple arithmetic adder circuit; it is a Boolean adder circuit operates over the bits of GMW shares of Alice and Bob. Now, there are two variants of this Boolean adder circuit available. Variant one which has order l number of gates and whose multiplicative depth is also order of l.

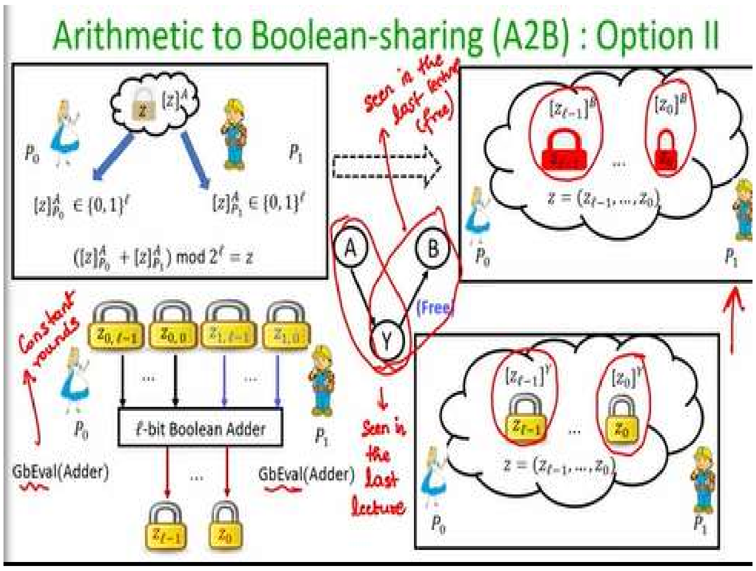
Whereas, we have another variant of the same circuit where the multiplicative depth is no longer order of l, but it is proportional to logarithm of l, but now the size of the circuit is large. So, depending upon whether you want to go for small size circuit or larger size circuit, but with less multiplicative depth, you can either use variant 1 circuit or variant 2 circuit in this conversion. But whatever variant we use, since we are now evaluating this Boolean adder circuit in the GMW world.

This requires interaction proportional to know multiplicative depth of the circuit. Say for instance, if Alice 1024, then if we use variant 1 circuit adder circuit, then the depth will be order of 1024 and it will require Alice and Bob to interact 1024 times to do this conversion whereas, if we use variant to then the size the depth of the circuit will be proportional to 10. So, roughly it will require Alice and Bob to interact 10 times.

But now, the number of gates will be more, which Alice and Bob have to securely evaluate when they are doing this conversion. But irrespective of what variant to use, the number of times they have to interact is not a constant, it depends upon l. If l is large, then it requires an enormous amount of communication, it requires the number of interactions will be more. Communication will be less.

Because, the advantage of GMW circuit evaluation is that parties do not need to communicate too much. But, the number of times they have to communicate is significantly larger, it is not constant.

(Refer Slide Time: 09:30)



So, if you are looking for a constant round conversion, then there is another way to do the arithmetic to Boolean conversion. And this is through a transitive method or transitive rule or transitive route or indirect method. What we can do is we can first do arithmetic to yao conversion? So, you can imagine that this is like an intermediate step, where the arithmetic shares of Z are used to convert them into to Yao shares of Z are bits of Z.

And we know how to do this arithmetic to yao conversion, we had already seen in the last lecture. We have Alice and Bob have to act as dealer and Yahoo share their bits of the GMW shares, and then the same adder circuit, which we are right now considering will be evaluated in the yao domain. Now, evaluating this circuit in the yao domain, it will require a constant number of rounds.

Because in the yao's domain evaluating any circuit is going to require a constant number of rounds, it does not matter how large is your l, and how deep is your arithmetic circuit. It will require only a fixed number of rounds of interaction between Alice and Bob. But, the amount of communication will be large because, now the circuit is evaluated in the yao domain, so Alice has to communicate the garbled circuits and so on.

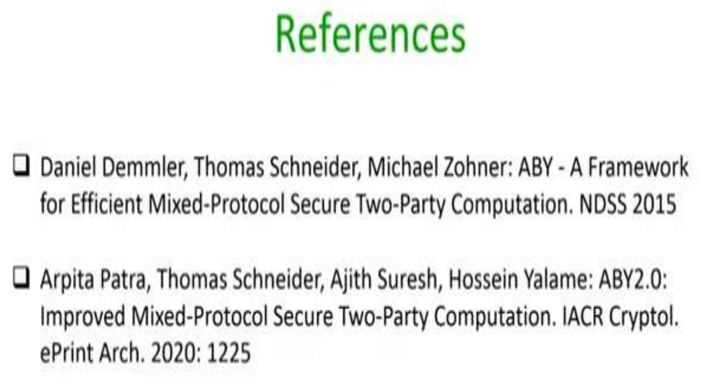
And Bob's computation also will be heavy because, he has to decrypt the gates while evaluating this Boolean adder circuit, using the point and permute optimization. But that is not full conversion; we have converted an arithmetic secret shared value into yao secret shared bits. We want actually the Boolean representation or the Boolean sharing output for the bits of Z. And for that, we observed that going from the yao's sharing of the bits of Z to the Boolean sharing of bits of Z is completely free.

Because, the yao to Boolean conversion we had seen in the earlier lecture and this is completely free. We had already seen that. Namely, if I consider say for instance the yao representation, a yao sharing for the bit Z_0, and if we want to extract GMW Boolean sharing of Z_0, what Alice and Bob have to do is they just have to set the permutation bits of their respective yao shares of Z_0 as their respective Boolean shares.

And the same computation if they do it for all other remaining bits as well. They easily go from the yao's secret sharing of the bits of Z to the Boolean secret sharing of the bits of Z and in the process nothing about Z is closed. So, that is the constant round version of the arithmetic to the Boolean conversion, but even though it is constant round the communication will be anonymous. So, you have two options, if number of rounds is very critical then go for option two.

When you are doing arithmetic 2 Boolean conversion, if communication is critical, then go for option number one.

(Refer Slide Time: 13:24)



References

- Daniel Demmler, Thomas Schneider, Michael Zohner: ABY - A Framework for Efficient Mixed-Protocol Secure Two-Party Computation. NDSS 2015
- Arpita Patra, Thomas Schneider, Ajith Suresh, Hossein Yalame: ABY2.0: Improved Mixed-Protocol Secure Two-Party Computation. IACR Cryptol. ePrint Arch. 2020: 1225

So, with that I end this lecture. So, again the conversions which are used for, which I discussed in today's lecture they are taken from the original ABY paper. You have more efficient conversions available in the latest version of the ABY conversions. Thank you.