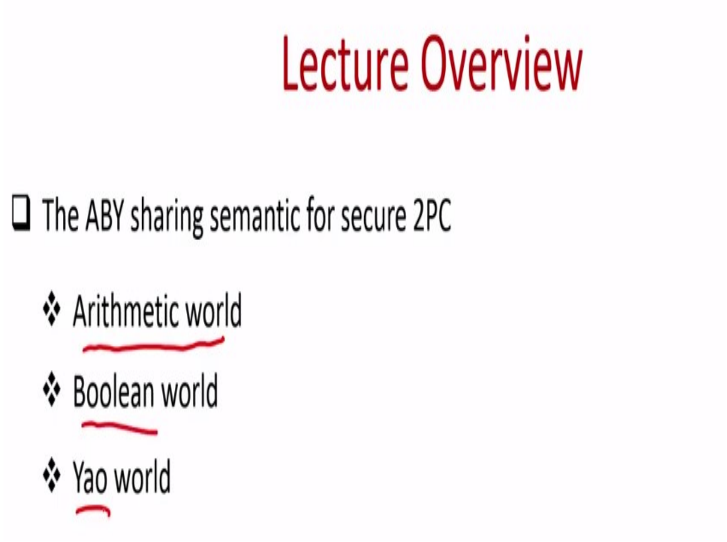
**Secure Computation – Part 1**
**Prof. Ashish Choudhury**
**Indian Institute of Technology – Bangalore**

**Lecture – 54**
**The Arithmetic, Boolean and Yao Sharing for Secure 2PC**

**(Refer Slide Time: 00:30)**

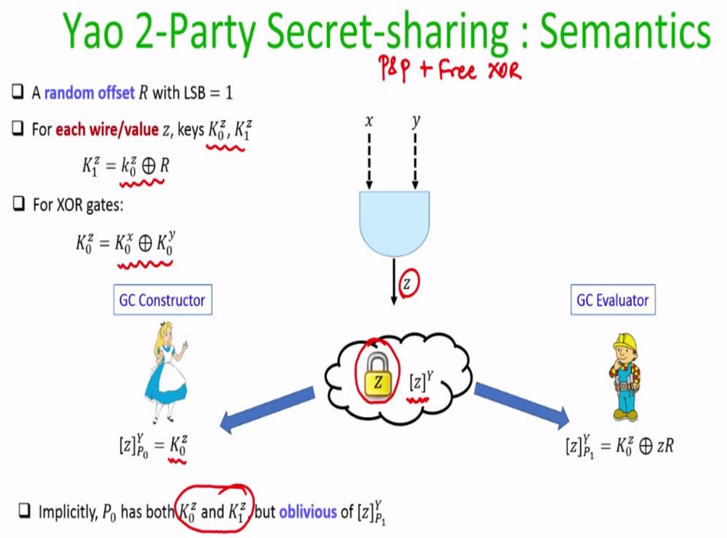


## Lecture Overview

- ❑ The ABY sharing semantic for secure 2PC
  - ❖ Arithmetic world
  - ❖ Boolean world
  - ❖ Yao world

Hello everyone, welcome to this lecture. So, in this world, we will continue our discussion on ABY sharing, ABY semantics of framework for doing mixed world secure 2PC computation. And in today's lecture we will focus on the sharing semantics in the arithmetic world, Boolean world and the Yao world.

**(Refer Slide Time: 00:53)**



## Yao 2-Party Secret-sharing : Semantics

P&P + Free XOR

- ❑ A random offset $R$ with LSB = 1
- ❑ For each wire/value $z$, keys $K_0^z, K_1^z$

$$K_1^z = k_0^z \oplus R$$

- ❑ For XOR gates:

$$K_0^z = K_0^x \oplus K_0^y$$

GC Constructor

GC Evaluator

$[z]^Y_{P_0} = K_0^z$

$[z]^Y_{P_1} = K_0^z \oplus zR$

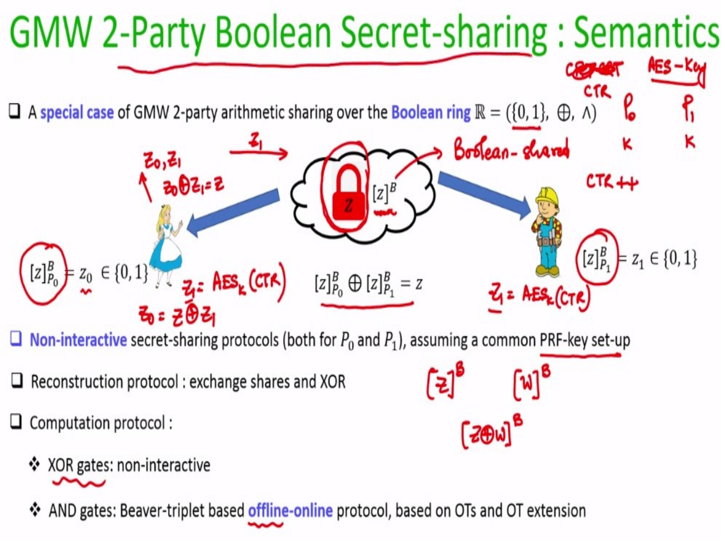
- ❑ Implicitly, $P_0$ has both $K_0^z$ and $K_1^z$ but oblivious of $[z]^Y_{P_1}$

So, the Yao's 2-party secret sharing semantics we had already discussed. Let me quickly go through it. So, we will follow the point and permute optimization along with free XOR technique where there will be a random offset or random global offset whose LSB is 1 known to the constructor. If a value z is supposed to be Yao's secret shared, then corresponding to this value z there will be two keys where the key corresponding to the value 1 will be related to the 0 key as per the global offset.

And we have an additional constraint that if this value z is outcome of an XOR gate, then its 0 key is not randomly chosen, but it is said to be the XOR of the 0 key for x and the 0 key for y. And a share for the value z available with Alice will be the 0 key, the share available with Bob for the bit Z will be the actual key corresponding to the actual value of Z. Namely if Z = 0 Bob will have the 0 key.

If Z = 1, Bob will have the 1 key. But Alice share will only always be the 0 key and implicitly Alice will have both the keys corresponding to z, but as per the sharing semantic we will say her share is the 0 key.

**(Refer Slide Time: 02:27)**



GMW 2-Party Boolean Secret-sharing : Semantics

- ❑ A special case of GMW 2-party arithmetic sharing over the Boolean ring $\mathbb{R} = (\{0,1\}, \oplus, \wedge)$

$[z]^B_{P_0} = z_0 \in \{0,1\}$

$[z]^B_{P_0} \oplus [z]^B_{P_1} = z$

$[z]^B_{P_1} = z_1 \in \{0,1\}$

- ❑ Non-interactive secret-sharing protocols (both for $P_0$ and $P_1$), assuming a common PRF-key set-up
- ❑ Reconstruction protocol : exchange shares and XOR
- ❑ Computation protocol :
  - ❖ XOR gates: non-interactive
  - ❖ AND gates: Beaver-triplet based offline-online protocol, based on OTs and OT extension

Now, let us focus on the GMW 2-party secret sharing semantic and it depends upon whether we are considering the Boolean sharing or the arithmetic sharing. So, if we are considering the Boolean sharing, then we have to assume that the values are shared over a Boolean ring consisting of only two possible values 0 and 1. The plus operation is to XOR operation, the dot operation or the product operation is the add operation.

And now I will be using this notation. This lock in a different colour compared to the lock which we are using for the Yao secret sharing. So, for the Yao secret sharing the lock was this and a value inside the lock means that the value is a secret shared or in terms of mathematical notation this was the symbol used for a bit secret shared as per the Yao semantic.

For GMW two parties, Boolean sharing semantic I will be using this picture, this lock with a different colour to denote that bit z is Boolean shared or in terms of mathematical notation this will be the notation or square bracket inside that bit Z and in the superscript we have B, B denotes that it is a Boolean shared and square bracket Z means no one, neither Alice nor Bob can in isolation find out what is the value of Z just based on its respective share.

But their combined share can allow them to find out the values. So, the GMW secret sharing is very simple. Alice share will be bit Z 0, Bob's share will be bit Z 1, and they are related by this relation their XOR of the XOR of the shares available with Alice and Bob will be the actual bit Z. So, this representation denotes the Alice share for Z, this denotes Bob's share for Z. Now, if I assume a PRF key setup what does that mean?

So, that means that if I assume that we have P 0 and P 1 and there is say AES key, AES is one of the practical indentations for pseudorandom function, say both of them are common key which has been set up which can be used for polynomially many instances. Then the secret sharing protocol both for Alice as well as for Bob can be made noninteractive, how so? So, we can imagine that there is a value of counter available both with Alice and Bob.

And now, suppose Alice wants to secret share a value, so what she can do is in the actual GMW secret sharing protocol, Alice would have picked Z 0, Z 1 randomly such that their XOR is equal to Z and then she would have transferred the shared Z 1 to Bob that is what Alice would have done if she is the owner of the value Z which she wants to secret share and this would require interaction.

What I am saying is assuming this AES key setup, we can run the counter mode of operation for AES and make the secret sharing protocol noninteractive, namely the communication of Z 1 from Alice to Bob we can get rid of that interaction, that communication, how? So, remember Z 1 is a share which is known both to Alice and Bob and which is randomly chosen independent of the value of Z because Z 0 and Z 1 are shares, they are random shares such that they are XOR is Z.

So, Alice would have picked Z 0 randomly and then she would have said Z 1 to be the XOR of Z and Z 0. Instead, what Alice and Bob can do is both of them can set Z 1 to be say the value of AES on the value of this counter and then Alice can set Z 0 to be the XOR of Z and Z 1 that serves the same purpose. Now, this does not require Z 1 to be communicated to Bob because both Alice and Bob will generate the same Z 1 because they have the same AES key and they have the agreement on the value of the counter.

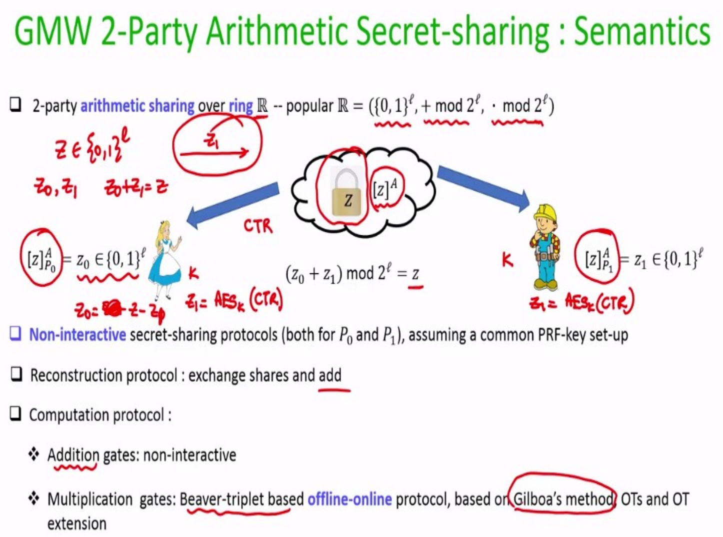
And once they have generated a sharing of Z, they have to increment the value of counter. So, that again if Alice has a next value which she wants to secret share, they can run the same procedure and Alice did not have to communicate her share to Bob. So, these kinds of tricks we can use in practice when implementing the MPC protocol. So, assuming that we have this kind of AES key set up, the communication of Alice share to Bob can be completely avoided in secret sharing instances where Alice is playing the role of the dealer.

And similar steps you can imagine can be executed if Bob wants to secret share a value. So, I am not writing down those specific steps. What will be the reconstruction protocol in the Yao domain, Yao world? So, if there is a bit Z which is secret shared as per this process and Alice and Bob want to reconstruct the bit Z, they just have to exchange their shares and then they can XOR the XOR both the shares and get back the value Z that is a simple reconstruction protocol.

How computations are performed in the GMW, Boolean world XOR gates since they are linear gates, they can be evaluated in a noninteractive fashion, namely if Z is secret shared as per Yao's Boolean representation and W is another bit which is also secret shared as per Yao's Boolean representation, then to get the secret sharing of Z XOR W as per the Yao's Boolean representation, both the parties have to just locally XOR their respective shares of Z and W.

Whereas AND gates can be evaluated based on the Beaver-triplet mechanism. And Beaver-triplet can be generated in the offline phase based on OTs and OT extension. So that is how the computations are handled in the GMW 2-party Boolean secret sharing representation.

**(Refer Slide Time: 09:48)**



GMW 2-Party Arithmetic Secret-sharing : Semantics

- ❑ 2-party arithmetic sharing over ring $\mathbb{R}$ -- popular $\mathbb{R} = (\{0,1\}^{\ell}, + \bmod 2^{\ell}, \cdot \bmod 2^{\ell})$

$z \in \{0,1\}^{\ell}$
$z_0, z_1$ $z_0 + z_1 = z$
$z_1$
$z$
$[z]^A$
CTR
$[z]^A_{P_0} = z_0 \in \{0,1\}^{\ell}$
K
$z_0 = z - z_1$
$z_1 = AES_K(CTR)$
$(z_0 + z_1) \bmod 2^{\ell} = z$
K
$[z]^A_{P_1} = z_1 \in \{0,1\}^{\ell}$
$z_1 = AES_K(CTR)$

- ❑ Non-interactive secret-sharing protocols (both for $P_0$ and $P_1$), assuming a common PRF-key set-up
- ❑ Reconstruction protocol : exchange shares and add
- ❑ Computation protocol :
  - ❖ Addition gates: non-interactive
  - ❖ Multiplication gates: Beaver-triplet based offline-online protocol, based on Gilboa's method, OTs and OT extension

Now let us see the GMW symmetric secret sharing representation where the values are over ring and the popular ring which is used in practice is the ring of all l-bit integers, where the addition operation is addition modular to power l, so that the result of the addition is still an l-bit number. And if it exceeds more than l-bits, then you do a wrap up, so that is why modular 2 power l. And in the same way multiplication operation is multiplication modular 2 power l.

Namely, you multiply two l-bit numbers, you again obtain a result built which is an l-bit number and to do that, we perform a modular 2 power l operation. In terms of the lock representation, now I will be using this lock to represent that there is an l-bit number which is secret shared between Alice and Bob. And this will be the mathematical notation. A in the superscript denotes that l-bit number Z is arithmetic secret shared.

Where the shares of Alice and Bob are two l-bit numbers such that if we sum them modular 2 power l, then we get the l-bit number Z. So, Alice's share will be represented by this notation, Bob's share will be represented by this notation. Again, assuming a PRF-key setup, the secret sharing protocols in the GMW arithmetic secret sharing world can be made noninteractive, say for instance if Alice wants to secret share a value Z which is an l-bit number.

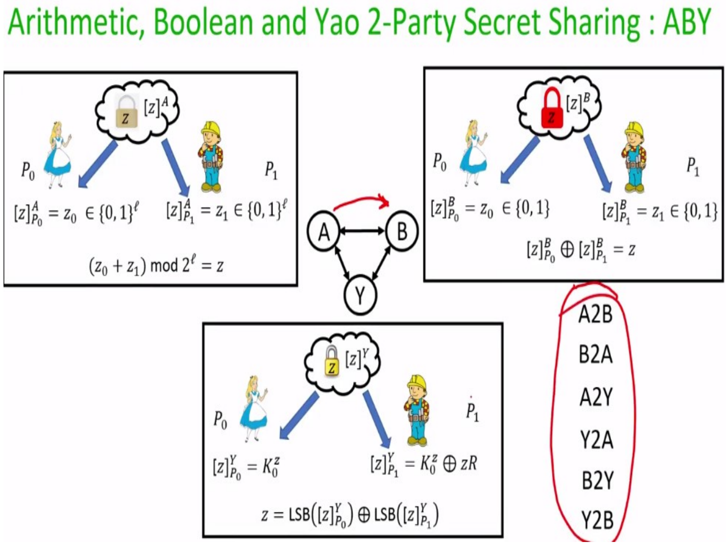
Then actually as per the GMW secret sharing protocol, she would have randomly picked Z 0, Z 1 such that their sum is Z and she would have communicated Z 1 namely the share of Bob to Bob. We can completely remove this interaction by assuming that there is an AES PRF setup available to Alice and Bob where both of them hold the value of the AES key, common AES key and the value of a common counter.

Now, what Alice and Bob can do is they can run the AES whenever Alice is supposed to secret share a value on the value of the counter that will be Bob's share and hence it need not be communicated from Alice to Bob. And once Bob's share is fixed, Alice can set her share, she has to set the Z 0 to be her share for Z to be this value. Similar step Bob can do execute to make his part of secret sharing noninteractive.

Reconstruction protocol is very simple. If an l-bit number needs to be publicly reconstructed which is secret share, then both the parties have to exchange their shares and then they have to simply add the shares to get back the l-bit number. When I say add, I mean to say addition modular 2 power l because that is the underlying plus operation. Computation protocols in this world are as follows.

So, there is an addition gate which needs to be evaluated where the gate inputs are GMW arithmetic secret shared and to get the gate output in GMW arithmetic secret sharing representation, the parties do not require any interaction, they just have to add their respective shares of the gate input. Whereas multiplication gates can be evaluated using the Beaver-triple method in the offline-online paradigm, where the triplets can be generated in the offline phase based on the Gilboa’s method using OTs and OT extension.

**(Refer Slide Time: 14:11)**



Arithmetic, Boolean and Yao 2-Party Secret Sharing : ABY

$[z]^A$
$z$
$P_0$
$P_1$
$[z]^A_{P_0} = z_0 \in \{0,1\}^\ell$
$[z]^A_{P_1} = z_1 \in \{0,1\}^\ell$
$(z_0 + z_1) \bmod 2^\ell = z$
$[z]^B$
$z$
$P_0$
$P_1$
$[z]^B_{P_0} = z_0 \in \{0,1\}$
$[z]^B_{P_1} = z_1 \in \{0,1\}$
$[z]^B_{P_0} \oplus [z]^B_{P_1} = z$
A
B
Y
$[z]^Y$
$z$
$P_0$
$P_1$
$[z]^Y_{P_0} = K^z_0$
$[z]^Y_{P_1} = K^z_0 \oplus zR$
$z = \mathrm{LSB}([z]^Y_{P_0}) \oplus \mathrm{LSB}([z]^Y_{P_1})$
A2B
B2A
A2Y
Y2A
B2Y
Y2B

So, these are our three different worlds; arithmetic world, Boolean world, Yao world. In the arithmetic world, the values are l-bit numbers. In the Boolean world and the Yao world the values are bits. And now what we want is a switching mechanism, namely if we have say a value which is secret shared as per the arithmetic representation, without disclosing that value we would like a mechanism to switch to the Boolean representation as per the GMW 2-party secret sharing for the same value Z.

Similarly, you can interpret the other switching as well. So since we have three worlds, we have to now come up with protocols for six possible switching mechanisms, which we are now going to see in the next lecture.

**(Refer Slide Time: 15:04)**

## 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, these are the references used for today's lecture to explain the arithmetic sharing semantic, Boolean sharing semantic and Yao sharing semantics. Thank you.