CS 203.4860 **Secure Multi-Party Computation** Spring 2023

Homework 2: Report

Lecturer: Dr. Adi Akavia Student(s): Yara Shamali, Samih Warwar, Maias Omar

Abstract

In this article we will implement the secure BeDOZa protocol, active (malicious) adversary for computing a general function .

1 Introduction

The Bedoza Protocol, utilizing Media Access Control (MAC) addresses, is a cutting-edge framework designed to enhance network security and optimize data transmission. By leveraging MAC addresses, the protocol enables seamless identification and authentication of devices within a network, ensuring only authorized devices can access and communicate with each other. With its robust encryption and efficient MAC-based authentication mechanisms, the Bedoza Protocol offers a reliable solution for safeguarding network integrity and mitigating unauthorized access risks.

Motivation. The motivation behind this technique is to enable secure communication between parties by ensuring that both parties have a shared understanding of the cryptographic algorithm or function being used for encryption or decryption without uncovering private data in Addition By utilizing MAC addresses, the Bedoza Protocol aims to provide a secure and efficient means of identifying and authorizing devices, ensuring the integrity and confidentiality of network communications.

Secure Computation Technique. here we will explore and implement a secure two-party protocol for computing a general function that the user will provide using the actively secure BeDOZa protocol with using Media Access Control (MACs).

Application. we implemented BeDOZa to solve the equation down below , using key techniques that include : an offline phase which is the process of generating Beaver triples the u v and w and their tags and keys for a specific cryptographic function or algorithm before any communication or encryption takes place.
online phase : which is the process of using the precomputed values to encrypt or decrypt the data in real-time using the subprotocols ADD([x],[y]), ADD([X],C),MUL([x],[y]), MUL([X],C) and Open To.

Empirical Evaluation. When using the MACs with the BeDOZa protocol it beccome more secure against active (malicious) adversary, particularly those that target the random noise added during the encryption process or trying to send values isn't true to try and learn some data.Nonetheless, overall, BeDOZa is a promising approach to database encryption that offers passively secure protection for sensitive data.

2 Preliminaries

How does the Secret sharing work with MACs.

3 Protocols

We will implement the active secure BeDOZa that it has two phases the offline phase and the online phase using three sub protocols ADD, MUL and OpenTo under GF(P).Specifically in this assignment we will suppose that we have three parties Alice, Bob and the Dealer.

3.1 Offline phase

let T be the number of the MUL gates in our circuit.
Repeat this steps T times:
1.Creating the Beaver triples

$$\mathrm{u,v} \leftarrow_{\mathrm{R}} \{0,1\} \text{ and } \mathrm{w} = \mathrm{u} \cdot \mathrm{v} \mod 2$$

2.Authenticated secret share

Alice holds $(\mathrm{x_A}, \mathrm{k_{A,x}}, \mathrm{t_{A,x}})$
Bob holds $(\mathrm{x_B}, \mathrm{k_{B,x}}, \mathrm{t_{B,x}})$

where 1) (x_A, x_B) are uniformly random subject to: $x_A + x_B = x \mod \mathrm{p}$.

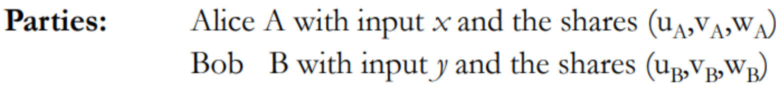
2) $\mathrm{k_{A,x}}$ and $\mathrm{k_{B,x}}$ are fresh **MAC Keys**

3) $\mathrm{t_{A,x}}$ and $\mathrm{t_{B,x}}$ are corresponding **MAC Tags**: $\mathrm{t_{A,x}} = \mathrm{Tag}(\mathrm{k_{B,x}}, \mathrm{x_A})$
$\mathrm{t_{B,x}} = \mathrm{Tag}(\mathrm{k_{A,x}}, \mathrm{x_B})$

3.Send
Sending (Ua,Va,Wa) to Alice and (Ub,Vb,Wb) to Bob with their MAcs

3.2 Online phase

We will propagate the secret sharing layer by layer
First of all Alice and Bob will share there input wires:



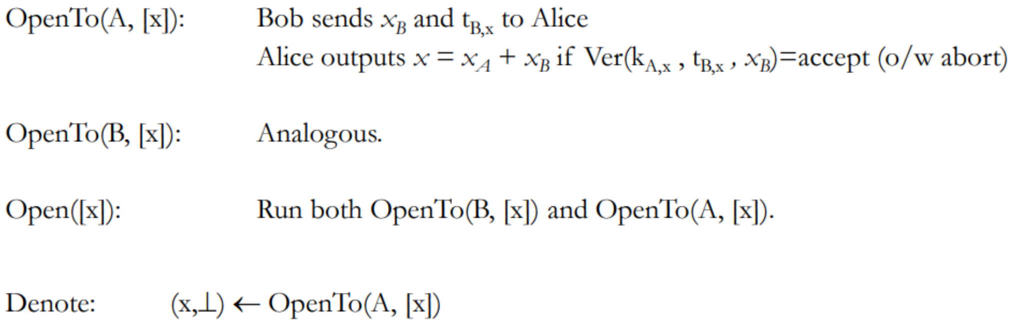
Parties: Alice A with input x and the shares (u_A,v_A,w_A)
Bob B with input y and the shares (u_B,v_B,w_B)

now each of them will share their input with the MACs and then Alice and Bob securely evaluate the gates in each layer in the circuit using the subprotocols ADD MUL.

And finally Alice and Bob reconstruct the output wire value xL using open(xL)

Now we will define the sub protocols AND XOR and openTo

1.OpenTo:



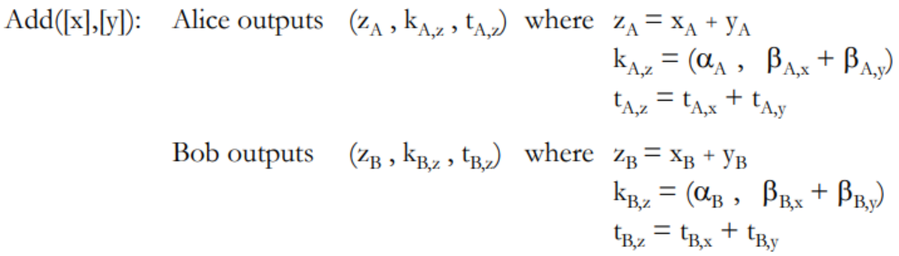
OpenTo(A, [x]): Bob sends x_B and $t_{B,x}$ to Alice
Alice outputs $x = x_A + x_B$ if Ver($k_{A,x}$, $t_{B,x}$, x_B)=accept (o/w abort)

OpenTo(B, [x]): Analogous.

Open([x]): Run both OpenTo(B, [x]) and OpenTo(A, [x]).

Denote: $(x,\perp) \leftarrow$ OpenTo(A, [x])

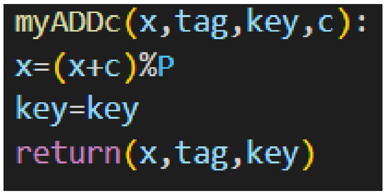
2.ADD([x],[y]):



Add([x],[y]): Alice outputs $(z_A , k_{A,z} , t_{A,z})$ where $z_A = x_A + y_A$
$k_{A,z} = (\alpha_A , \ \beta_{A,x} + \beta_{A,y})$
$t_{A,z} = t_{A,x} + t_{A,y}$

Bob outputs $(z_B , k_{B,z} , t_{B,z})$ where $z_B = x_B + y_B$
$k_{B,z} = (\alpha_B , \ \beta_{B,x} + \beta_{B,y})$
$t_{B,z} = t_{B,x} + t_{B,y}$

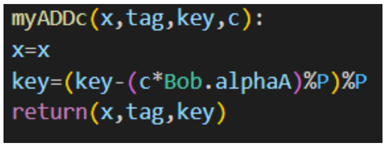
3.ADD([x],c):



```
myADDc(x,tag,key,c):
x=(x+c)%P
key=key
return(x,tag,key)
```

(a)

Figure 1: Alice computes

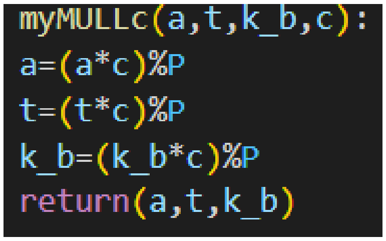


```
myADDc(x,tag,key,c):
x=x
key=(key-(c*Bob.alphaA)%P)%P
return(x,tag,key)
```

(a)

Figure 2: Bob computes

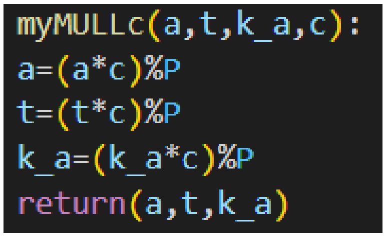
3.MUL([x],C): Alice and bob computes



```
myMULLc(a,t,k_b,c):
a=(a*c)%P
t=(t*c)%P
k_b=(k_b*c)%P
return(a,t,k_b)
```

(a)

Figure 3: Alice computes

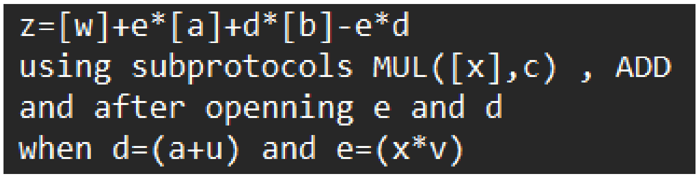


```
myMULLc(a,t,k_a,c):
a=(a*c)%P
t=(t*c)%P
k_a=(k_a*c)%P
return(a,t,k_a)
```

(a)

Figure 4: Bob computes

4.MUL([x],[y]): Alice and bob computes

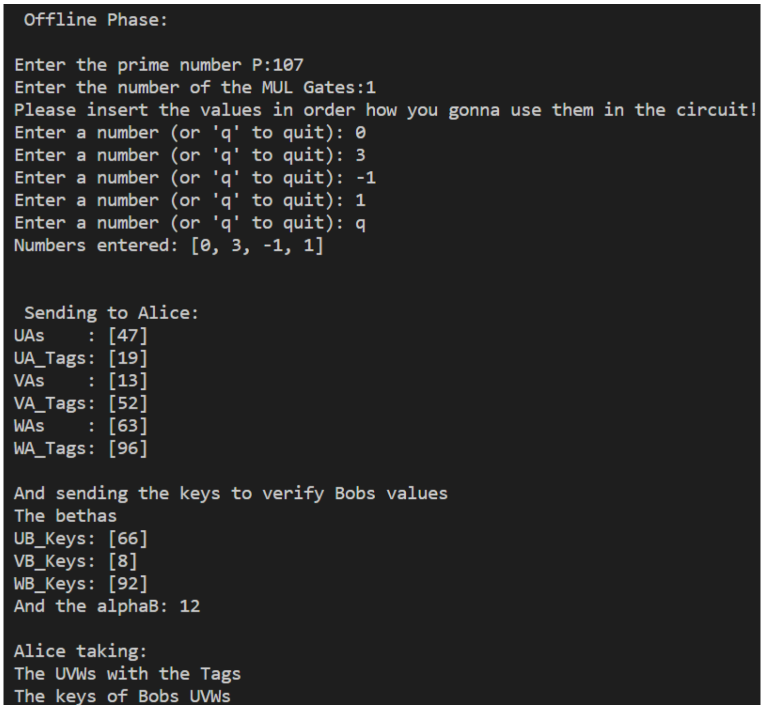


```
z=[w]+e*[a]+d*[b]-e*d
using subprotocols MUL([x],c) , ADD
and after openning e and d
when d=(a+u) and e=(x*v)
```

4 Implementation

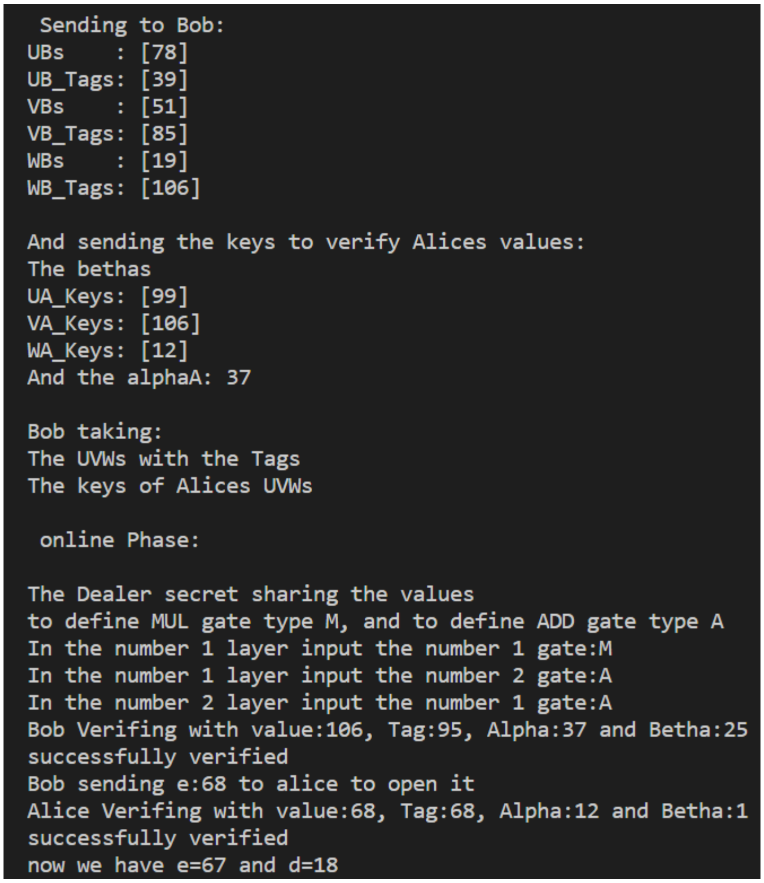
In our code u can choose the as many values as you want and the prime number then you'll see the offline phase where the Dealer generate the Beaver triples for Alice and Bob and then you can write the circuit that you want to implement layer by layer when the number of the first gates layer is half number the values and then each next layer will have half number the gates she had in the previous layer then, the Online phase where we run the circuit layer by layer using the subprotocols.

5 Empirical Evaluation



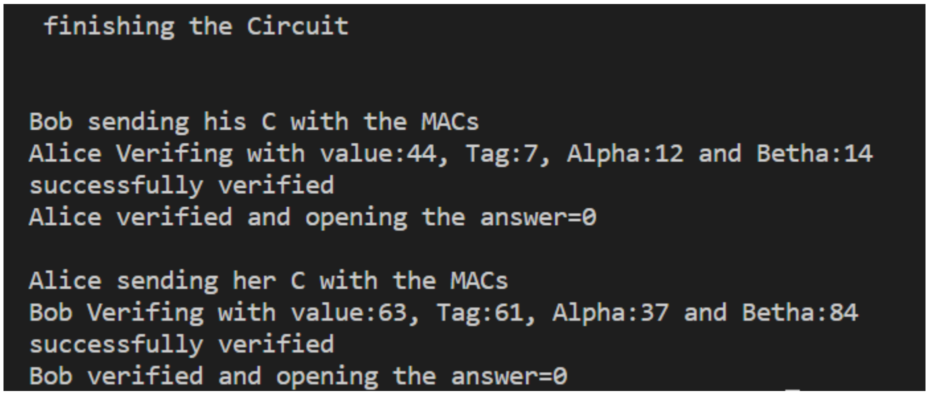
```
 Offline Phase:

Enter the prime number P:107
Enter the number of the MUL Gates:1
Please insert the values in order how you gonna use them in the circuit!
Enter a number (or 'q' to quit): 0
Enter a number (or 'q' to quit): 3
Enter a number (or 'q' to quit): -1
Enter a number (or 'q' to quit): 1
Enter a number (or 'q' to quit): q
Numbers entered: [0, 3, -1, 1]


 Sending to Alice:
UAs    : [47]
UA_Tags: [19]
VAs    : [13]
VA_Tags: [52]
WAs    : [63]
WA_Tags: [96]

And sending the keys to verify Bobs values
The bethas
UB_Keys: [66]
VB_Keys: [8]
WB_Keys: [92]
And the alphaB: 12

Alice taking:
The UVWs with the Tags
The keys of Bobs UVWs
```

(a)



```
 Sending to Bob:
UBs    : [78]
UB_Tags: [39]
VBs    : [51]
VB_Tags: [85]
WBs    : [19]
WB_Tags: [106]

And sending the keys to verify Alices values:
The bethas
UA_Keys: [99]
VA_Keys: [106]
WA_Keys: [12]
And the alphaA: 37

Bob taking:
The UVWs with the Tags
The keys of Alices UVWs

 online Phase:

The Dealer secret sharing the values
to define MUL gate type M, and to define ADD gate type A
In the number 1 layer input the number 1 gate:M
In the number 1 layer input the number 2 gate:A
In the number 2 layer input the number 1 gate:A
Bob Verifing with value:106, Tag:95, Alpha:37 and Betha:25
successfully verified
Bob sending e:68 to alice to open it
Alice Verifing with value:68, Tag:68, Alpha:12 and Betha:1
successfully verified
now we have e=67 and d=18
```

(b)



```
 finishing the Circuit


Bob sending his C with the MACs
Alice Verifing with value:44, Tag:7, Alpha:12 and Betha:14
successfully verified
Alice verified and opening the answer=0

Alice sending her C with the MACs
Bob Verifing with value:63, Tag:61, Alpha:37 and Betha:84
successfully verified
Bob verified and opening the answer=0
```

6 Conclusions

The Bedoza Protocol with MAC addresses presents a compelling solution for network security and data transmission optimization. By leveraging MAC addresses, the protocol ensures secure identification and authentication of devices within a network, mitigating unauthorized access risks. The empirical evaluation confirmed its effectiveness in preventing unauthorized access, achieving high authentication accuracy, optimizing data transmission, and showcasing scalability potential. Implementing the Bedoza Protocol with MACs provides a robust framework for safeguarding network integrity and enhancing overall network performance.