1. Introduction

Secure multi-party computation (MPC) is one of the subfield of cryptography. The aim of MPC is providing input privacy and correctness while a set of parties compute a function of their inputs[1]. The computation could occur between trusted parties, between untrusted parties or even between competitors. Usually in many computations one must know all inputs from all participants; however if nobody is trusted enough then privacy becomes primary concern. There are many applications where privacy of input is primary concern. For example:

Two financial organizations may plan to work on the same project together for their mutual benefit. Each organization would like to satisfy their own requirements. However, since their requirements include some proprietary data such as the interests, inflation rates, economic statistics etc. they will not want to disclose their requirements[2].

The privacy of this two financial organizations' inputs, their requirements in that example indicated above, will be primary concern. We can also simply explain MPC concepts by millionaire problem which is suggested by Andrew C. Yao in a paper in 1982[3]. Simply by considering millionaire problem by using MPC two millionaires can compute which one is richer, but keeping privacy of net worth of each.

The aim of this report is giving technical information about a simple mobile application that exemplifies SMC using addition and multiplication protocols. Project also gives short explanation about secret sharing scheme that was used.

The layout of this report is as follows. In section 2 the idea behind this project and point of it are briefly explained. Section 3 will discuss technologies those are used for this project, while Section 4 explains how the system works. Section 5 explains the use cases and system requirements (functional and non-functional) of the project and Section 6 will show the problems encountered during the development of the app and some important points that should be carefully considered. Finally we outline our conclusions in Section 7.

2. Description of the Concept

The mobile application implemented for this project uses secure multiparty computation approach which is secret sharing based. Secret sharing uses $n$ computation parties which are independent and each of them can compute an agreed function of their inputs in a secure way, where security means guaranteeing the privacy of the parties' inputs. The app makes bluetooth connection for communication and shares secret data.
In this project 3 computation parties were used. Every party runs Android application developed by this project which ensures each of them has just one role:

1. Initiator party (IP): The initial owner of the secret data. Initiator party initiates the application and chooses other two parties for computation. The initiator also chooses the function either addition or multiplication.
2. Helper party (HP): The party who helps to evaluate a function on secret data.
3. Guesser party (GP): The party who is interested the result of the computation. Guesser is the one just who is able to see the result and try to find inputs.

Purpose of this project is developing an educational mobile app to demonstrate secure multiparty computation. The app aims providing privacy of inputs by showing GP cannot deduce anything from the shares it gets. Since the connection channel is neither authorized nor encrypted it is important to know that the app achieve security only for a passive adversary setting, without the possibility to enhance it to active adversary settings. The system provide one secret sharing scheme: additive which uses following protocols:

Protocol for addition Diagram 1:

![Diagram 1: Addition](image)

We will denote secret shared value $x$ as $[[x]]$. Let us have two secret shared values $t$ and $s$ shared as $[[s]] = (s1; s2; s3)$ and $[[t]] = (t1; t2; t3)$,

where shares $s_i$ is generated by P1 (IP) and $t_i$ is generated by P2 (HP) ($i \in \{1,2,3\}$).

Addition of those two values needs the following computations:

$$[[s]] + [[t]] = (s1 + s2 + s3) + (t1 + t2 + t3) = (s1 + t1) + (s2 + t2) + (s3 + t3)$$

where the additions computed locally by the corresponding parties if shares and result of the addition can be shared as it seen in Diagram 1.

Protocol for multiplication Diagram 2:
In the diagram above we will denote secret shared value $x$ as $\langle x \rangle$. Let us have two secret shared values $s$ and $t$ shared as $\langle s \rangle = (s_1; s_2; s_3)$ and $\langle t \rangle = (t_1; t_2; t_3)$, where shares $s_i$ is generated by $P_1$ (IP) and $t_i$ is generated by $P_2$ (HP) ($i \in \{1, 2, 3\}$). Multiplication of those two values needs the following computations:

$$\langle s \rangle \cdot \langle t \rangle = (s_1 + s_2 + s_3)(t_1 + t_2 + t_3)$$

$$= s_1t_1 + s_1t_2 + s_1t_3 + s_2t_1 + s_2t_2 + s_2t_3 + s_3t_1 + s_3t_2 + s_3t_3$$

$$= (s_1t_1 + s_3t_1 + s_1t_3) + (s_2t_2 + s_2t_1 + s_1t_2) + (s_3t_3 + s_3t_2 + s_2t_3)$$

where the three products in the brackets can be computed locally by the corresponding parties if shares and result of the product can be shared as it seen in diagram 2.

In both function as it can be seen easily just $P_3$ (GP) calculate the result. The other parties just enter their secret and compute agreed function by using their shares.

3. Technology Overview

- **Nexus 5**
  Nexus 5 is an Android device which was announced in October 2013 by LG and developed by LG and Google[4]. For this application three Nexus 5 were used. There is no special reason to use these devices.

- **Android**
  Android is one of the open source operating system developed by Google. It is based on Linux Kernel[5]. Since it is free and it is open source which means it is also possible to modify and customize the OS, Android was chosen for this project to get benefit of economical and flexibility of the Android. This application supports Android 4.3 version (API Level 18) and above which means the device that wants to run this mobile application should have at least Android 4.3 version as an operating system.
• **Java**  
Java is concurrent, class based, object oriented language that is originally developed by James Gosling and released in 1995. Java provides platform independency, which means code that is written in Java and runs on one platform does not need to be edited to run on another[6][7]. Since the mobile app was implemented in native (no framework is used) Java was preferred as a language.

• **Eclipse**  
Eclipse is a Java based open source platform that provides a multi-language software development environment to the developers. Since Eclipse has also special platform just for Android development and it is strictly recommended, it is used for that project to take advantage of its utilities.

• **Bluetooth**  
Bluetooth is low-cost, short range, low power consuming wireless networking technology that is used for communications and also sharing file/data between parties. Since Bluetooth is much more cheaper and use less power than WiFi it was used for development of this project.

• **Android SDK**  
Android SDK is set of Android development tools and API libraries that is necessary to build, test, debug apps for Android powered devices.

4. **How The System Works**  
First of all, all participants should have the app on their phone which has to have at least 4.3 Android version and which also should have Bluetooth support. The initiator party (IP) starts the process. The system works as follows:

**Step 1: Start application and choose other participants**

![Diagram of System](figure1.png)

*Figure 1*
IP should start the app and ask the app for searching devices as it seen in Figure 1. The app searches all of the possible participants who have been paired before or who are discoverable by the others and lists all of the devices that it found with their address and also their bluetooth device name.

**Step 2: Choose other participants**

![Figure 2](image)

As it is seen in Figure 2, in Step 2 initiator has to choose two more participants those are found and listed in Step 1 by the system. If he chooses more or less than two, the system will not allowed him to continue. In this case the system will give warning message and two options: either to return to main app window or to try again.

**Step 3: Choose the function: Addition or Multiplication**

![Figure 3](image)

At this step it is time to start the multiparty computation. As it is shown in Figure 3 after the IP chooses two other participants system offers three options to the IP: addition, multiplication or cancel. The IP should make a choice for calculation.

If he chooses “Cancel” system will turn back to initial state and if he wants make calculation again later he will have to repeat previous two steps (he had already started the app in Step 1 he will just have to ask the system to search devices again in Step 1. Step 2 and Step 3 will be completely repeated). If he chooses either addition or multiplication the system will continue to Step 4.
Step 4: Enter the input

In that step the system asks for input to the IP that he wants to use it either for addition or for multiplication. The IP should enter the input (let call it S) and press OK (see Figure 4). After the IP completes his task in that step it comes the system turn. From this step on no more interaction from the IP is required.

Step 5: Create secrets and define HP and GP

In that step we are still in IP side but without any interaction with IP anymore. In Step 5 the system will create three shares, two of which are generated randomly and the third is calculated as shown in the following Java code:

```java
s1 = ran.nextInt (100);
s2 = ran.nextInt (100);
s3 = (s - s1 - s2) % 100;
if (s3 < 0) s3 += 100;
```

This algorithm creates shares of IP. Note that \((s1+s2+s3)\%100=s\). In this application we have chosen to perform computations modulo 100 so that the shares can be easily recognized by a person and they fit on a smartphone screen. If it is addition S2 will be sent to HP and S3 to GP, while S1 stays in IP for the computation. If it is multiplication S2 and S1 will be used with HP, S3 and S1 will be shared with GP.

It is time to define the role of other participants. The roles of other participants are chosen by system randomly in this project. Even there is no reason for randomness here system uses the following algorithm to choose the HP and GP.

```java
Random ran = new Random ();
hpLocationInSelectedDeviceList = ran.nextInt (2);
if (hpLocationInSelectedDeviceList == 0){
gpLocationInSelectedDeviceList = 1;
}
else{
gpLocationInSelectedDeviceList = 0;
}
```

Since there security of protocol does not require random selection of roles, the report will not discuss whether the Random() class provided in java.util generates secure random numbers.
Step 6: Connect with HP and send a formatted message

In this step the message exchange starts. The system has to be sure that every participant knows their roles. Therefore, it should inform other participants about their roles and also about selected scheme.

Since more connection means less efficiency and more expenses, system first connect HP. If it is addition he sends his secret S2 (which is generated for sharing it with HP) and get HP’s secret T1 (which is generated in Step 7 for sharing it with IP) by using the same bluetooth connection (see Figure 5 and Figure 7).

If it is multiplication he sends his secret S2 and S1 and get HP’s secrets T1 and T3 by using the same bluetooth connection (see Figure 5 and Figure 7).

The message that is sent by IP to the HP has two following format (these formats is also used for every message except two messages between parties):

For Addition:

```java
//isHelper, isInitiator, isGuesser, initiatorAddress, theGuesserAddress, theSecret, isAddition, result, isSenderInitiator
message="1,0,0,"+mBluetoothAdapter.getAddress()+"","
+devices.get (watcherDevice) +","+String.valueOf (s2)+"","
+"1"+","+String.valueOf (0)+"","+"1";
```

For Multiplication:

```java
//isHelper, isInitiator, isGuesser, initiatorAddress, theGuesserAddress, theSecret, isAddition, result, isSenderInitiator, secondValue
message="1,0,0,"+mBluetoothAdapter.getAddress ()
+"","+devices.get (watcherDevice)
+"","+String.valueOf (s2)+"","+"0"+","+
+String.valueOf (0)+"","+"1"+","+String.valueOf (s1);
```

All of the attributes has feasible reason:

- **isHelper, isInitiator, isGuesser**: These attributes define the participants their roles.
- **initiatorAddress, theGuesserAddress**: After the Step 1 these attributes will be used initiator address by GP. The guesser address by HP.
- **theSecret**: This attribute includes the secret for the other participants S2 for HP S3 for GP those are sent by IP and also T1 for IP and T3 for GP those are sent by
HP to other participants (check step 7 for generating HP's secrets).

- **isAddition**: This attribute inform the other participants about scheme whether it is additive or not. The scheme is additive if it is zero and multiplicative if it is one. It is initialized by IP.
- **result**: This attributes include the result of the calculation which is done by HP and IP by used selected scheme: addition or multiplication. This value is sent by HP and IP to only GP. The messages those are exchanged by other parties have “0” as a result. attribute
- **isSenderInitiator**: This attributes tell the GP if the sender is IP or not. HP set this value 0 and IP set 1 and GP take action by using this value (this action will be explained in following steps)
- **secondValue**: This value is used just for multiplication for second value that is sent just for multiplication (check Diagram 2).

After HP gets the message of IP, it parses the message and check for attributes those are indicated above to set the system to appropriate role. And whenever the system sees the participant's role as a HP it continues to next step. All of the attributes in message are kept in memory of the HP by the system.

Step 7: **Enter the input as a HP and sent a share to IP by using ongoing connection**

After that there will be no interaction between system and HP.

It is time to generate the secrets of HP. The system of HP will create three secrets two for sharing one for using by using the input that the HP entered. It will use following algorithm that is implemented in Java:

\[
\begin{align*}
t_1 &= \text{ran.nextInt}(100); \\
t_2 &= \text{ran.nextInt}(100); \\
t_3 &= (t_1 - t_2) \% 100; \\
\text{if} \ (t_3 < 0) \ t_3 &=+ 100;
\end{align*}
\]

This algorithm creates shares of HP. Check that \((t_1 + t_2 + t_3) \% 100 = t\). T1 will be used with IP and T3 will be shared with GP while the HP will use T2 for his side calculation.

As it was explained in Step 6 there are economical and efficiency reasons to select HP first instead of GP. The system of HP can use ongoing bluetooth connection that is initiated by IP to send the secret T1 to the IP. The system send a formatted message to the IP and cut the connection (Figure 7).
The message that is send to the IP can be in two formats as follows (please check Step 6 to understand fields):

For Addition:

```java
//isHelper, isInitiator, isGuesser, initiatorAddress, theGuesserAddress, thesecret, isAddition, result, isSenderInitiator
message="0,1,0,"+mChatService.getInitiatorAddress ()
+","+mChatService.getWatcherAddress ()+","+String.valueOf (t1)+"","+"1"","+String.valueOf (0)+"","+"0";
```

For Multiplication:

```java
//isHelper, isInitiator, isGuesser, initiatorAddress, theGuesserAddress, thesecret, isAddition, result, isSenderInitiator, secondValue
message="0,1,0,"+mChatService.getInitiatorAddress ()
+","+mChatService.getWatcherAddress ()+","+String.valueOf (t1)+"","+"0"","+String.valueOf (0)+"","+"0"","+String.valueOf (t3);
```

IP will save the fields and wait for its turn.

**Step 8: Connect with GP as a HP and send a formatted messages**

It is time to get GP to the game. In Step 6 HP get some information like guesser address and S2 from IP if agreed function is addition otherwise, he gets U1 and U2 if it is multiplication. After that the system generates his secrets: T1, T2 and T3. Therefore, he is ready to make calculation in that point. First of all HP makes a bluetooth connection (Figure 8) with GP and checks the selected secret based scheme to decide calculation that he is supposed to do.

**If the function that was chosen in Step 3 is addition then HP will calculate:**

\[
\text{u2} = \text{(S2+T2)} \mod 100
\]

and prepare his formatted message to share information with GP. HP will prepare a
message for \( GP \) as follows:

```java
//isHelper,is initiator, isGuesser, initiator address, the guesser address, the secret, isAddition, result, isSenderInitiator
messages="0,0,1,"+mChatService.getInitiatorAddress () +","+mChatService.getWatcherAddress ()+","+String.valueOf (t3) +","+"1"+,","+String.valueOf (u2)+","+"0";
```

**If the function that was chosen in Step 3 is multiplication then HP will calculate:**

\[
w_2 = (S_2 T_2 + S_2 T_1 + S_1 T_2)
\]

and prepare his formatted message to share information with \( GP \). HP will prepare a message for \( GP \) as follows:

```java
//isHelper,is initiator, isGuesser, initiator address, the guesser address, the secret, isAddition, result, isSenderInitiator, secondValue
messages="0,0,1,"+mChatService.getInitiatorAddress () +","+mChatService.getWatcherAddress ()+","+String.valueOf (t3)+","+"0"+,","+String.valueOf (w2)+","+"0"+,","+String.valueOf (t2);
```

\( GP \) will parse the message and will set its system as a \( GP \) (isGuesser has value of 1 which tells the system it has role of \( GP \)). The system of \( GP \) will check isSenderInitiator value is zero or not. Since it is zero (means it should connect to other party; \( IP \)) it will store all attributes, send one message to \( HP \) to tell him his mission is completed and reset his system automatically to the initial state (Figure 9), get the address of the initiator and connect with \( IP \) to get shares (\( S_3 \) if addition, \( S_3 \) and \( S_2 \) if multiplication) and also calculation result of \( IP \) which is derived from addition or multiplication.

![Figure 9](image)

---

**Figure 9**
Step 9: Connect with IP as a GP and get shares and sum or multiplication result

As it is seen Figure 10, GP will connect IP to get shares (S3 if addition, S3 and S2 if multiplication) and also calculation result of IP which is derived from addition or multiplication. GP first makes connection and send message to the IP to inform him about it is time for GP. The message that GP sends to the IP is one length message and it has just the value of 1 just to spark IP. IP checks the selected secret based scheme to decide calculation that he is supposed to do.

If the function that was chosen in Step 3 is addition then IP will calculate:

\[ u_1 = (s_1 + t_1) \mod 100 \]

and prepare a formatted message to the GP as follows:

```java
//isHelper, isInitiator, isGuesser, initiatorAddress, theguesserAddress, thesecret, isAddition, result, isSenderInitiator
messages="0,0,1,"+mBluetoothAdapter.getAddress ()
+","+devices.get (watcherDevice)
+","+String.valueOf (s3)+","+"1"+","+String.valueOf (u1)+","+"1";
```

If the function that was chosen in Step 3 is multiplication then IP will calculate:

\[ w_1 = (s_1 \times t_1 + s_1 \times t_3 + s_3 \times t_1) \mod 100 \]

and prepare a formatted message to the GP as follows:

```java
//isHelper, isInitiator, isGuesser, initiatorAddress, theguesserAddress, thesecret, isAddition, result, isSenderInitiator, secondValue
messages="0,0,1,"+mBluetoothAdapter.getAddress ()
+","+devices.get (watcherDevice)
+","+String.valueOf (s3)+","+"0"+","+String.valueOf (w1)+","+"1"+","+String.valueOf (s2);
```

The GP will parse the message and check if isSenderInitiator is one or not. If it is one it will send message to IP to tell him his mission is completed and reset his system automatically to the initial state and also cut the connection. In that step the system of GP is ready to make calculation by checking selected scheme.

If the scheme that was chosen in Step 3 is addition then GP will calculate:

\[ u_3 = (s_3 + t_3) \mod 100 \]
\[ s+t = (u_1 + u_2 + u_3) \mod 100 \]

If the scheme that was chosen in Step 3 is multiplication then GP will calculate:

\[ w_3 = (s_3 \times t_3 + s_3 \times t_2 + s_2 \times t_3) \mod 100 \]
\[ s.t = (w_1 + w_2 + w_3) \mod 100 \]

GP is ready for guessing. System will continue to next step.
Step 10: Guess the input of IP and HP as a GP

This is the last step for the system. In the end of this step privacy of inputs will be proved. The system will show a transcript to the GP and will ask him to guess S and T.

If the function that was chosen in Step 3 is addition then:
The transcript will contain s3, t3, u1, u2, u3, s+t and for every guess the system will mix the numbers as following algorithm:

```java
Random ran = new Random ();
int s=ran.nextInt (sum_of_sAndt);
s1 = ran.nextInt (100);
s2=ran.nextInt (100);
s3= (s-s1-s2)%100;
if (s3<0) s3 += 100;
int t=sum_of_sAndt-s;
t1 = ran.nextInt (100);
t2=ran.nextInt (100);
t3= (t-t1-t2)%100;
if (t3<0) t3 += 100;
u2 = (t2+s2)%100;
u1 = (t1+s1)%100;
u3 = (s3+t3)%100;
```

In that algorithm s is choosen randomly between zero and sum of s and t however the value of t depends on the value of the new s and sum of s and t. The result of sum will not change even s and t will change. The GP will see the different value of s3, t3, u1, u2, u3 however S+T will not change.

If the function that was chosen in Step 3 is multiplication then:
The transcript will contain s3, s2, t2, t3, w1, w2, w3, w*t and for every guess the system will mix the numbers as following algorithm:

```java
Random ran = new Random ();
int s=-1;
while (true){
    int temp_s=ran.nextInt (allResult);
    if (temp_s!=0 & allResult%temp_s==0){
        s=temp_s;
        break;
    }
}

s1 = ran.nextInt (100);
s2=ran.nextInt (100);
s3= (s-s1-s2)%100;
if (s3<0) s3 += 100;
int t=allResult/s;
t1 = ran.nextInt (100);
t2=ran.nextInt (100);
t3= (t-t1-t2)%100;
if (t3<0) t3 += 100;
w1 = (s1*t1+s1*t3+s3*t1)%100;
w2 = (s2*t2+s2*t1+s1*t2)%100;
```

In that algorithm s is choosen randomly between zero and multiplication of s and t however the value of t depends on the value of the new s and multiplication of s and t. The result of
multiplication will not change even s and t will change. The GP will see the different value of s3, s2, t2, t3, w1, w2, w3, however S*T will not change.

In both functions even GP's guess will be right since the system's of all parties will know the values just about their own there will be no clue about his guess's correctness. The system will let him to guess 3 times to show there are many possibilities for S and T and there is no way to disclosing privacy of the inputs. After three guesses system will generate last warn to inform the GP about game is over.

5. **Use Cases and System Requirements**

This section gives uses cases with screenshots. Resolution of the texts in pictures are quite low. However, on device quality is much more higher.

**Use Case 1: Show the Main Page:**
Whenever user starts application he will see the Picture 1 if his device's bluetooth is not open. Otherwise, Picture 2 will be seen.

![Picture 1](image1.png)

![Picture 2](image2.png)

The app will not allow user to choose “Deny” option. If he wants to continue he must
choose “Allow” option.

Use Case 2: Make the Device Discoverable:
If the participants didn't pair with IP before. They should be visible to IP. The app provides this if the user press “Make discoverable” button the user will see Picture 3. If he allows to make his device visible to the others he will see Picture 4 otherwise he will see Picture 2 again.
Use Case 3: See Paired and Non-paired Devices:
Please see Section 4 ---> Step 1
Use Case 4: Select **GP** and **HP**:
Please see Section 4 ---> Step 2
Use Case 5: Select function and Enter the input as an Initiator:
Please see Section 4 ---> Step 3, Step 4
Use Case 7: Enter the input as a HP
Please see Section 4 --> Step 7
Use Case 8: Guess the input of IP and HP as a GP for addition
Please see Section 4 ---> Step 10
Use Case 9: Guess the input of IP and HP as a GP for multiplication
Please see Section 4 ---> Step 10

System Requirements

Purpose
Purpose of this project is developing an educational mobile app to demonstrate secure multiparty computation. The app aims providing privacy and correctness of inputs by showing GP cannot deduce anything from the shares it gets. (For understanding how it is working please see the use cases)

Objectives and Success Criteria of the Project
Application provides privacy of the inputs.

Non-Functional Requirements

Usability
Since the aim of this project is developing an educational mobile app to demonstrate
secure multiparty, the app has user friendly interface and it is easy to use it.

**Reliability**
Since the application does not have to provide security and integrity of the users' inputs, reliability of the app relies on reliability of bluetooth connection channel.

**Performance**
Since the app does not have any complicated functions, performance of the app relies on the quality of the bluetooth devices that parties have on their mobile phone.

6. **Problems and Important Points**

The application is developed on Linux, version 14.04, Eclipse is used as a platform and Bluetooth connection as a connection channel. The security of the application relies on Bluetooth channel's security. However, since the security of Bluetooth channel is out of our scope and also security is not concern of this project, availability of Bluetooth almost in every devices were main reasons to select Bluetooth connection as a channel.

This application is developed for Android which actually has still many problem. The development started by using one Nexus 7 and two Nexus 5, however, since some of the Nexus 7 has bluetooth problem it is replaced with other Nexus 5. It also was planned to use multiple connections simultaneously, however even it is possible to have multiple connections simultaneously by using multiple threads, it was increasing the expenses: time for development and also resources that are allocated for all connections. Management of multiple threads are also quite difficult. Furthermore, even Bluetooth is cheaper for users because of the time it is vice versa in development side.

Android Wifi -Direct is one of the alternative for connection channel. Even it is faster, more secure comparing to bluetooth channel and also has many advantage over bluetooth channel, its cost and power consumption which should be considered carefully for mobile phones is much more higher. Even also latency which is lower in bluetooth makes the process more complicated since data that is being exchanged in that project is not so big and speed is not main concern, bluetooth the connection is best choice for this project. However, wifi-direct may be used for future works which need higher speed, security, latency and bit-rate.

7. **Conclusion**

This application can be used as an educational mobile app to demonstrate secure multiparty. Although its reliability and security relies on bluetooth communication channel, it provides privacy for parties' inputs.
References

1. Cramer, Ronald and Damgård, Ivan and Maurer, Ueli “General Secure Multi-Party Computation from any Linear Secret-Sharing Scheme” eprint 2000/037
3. Yao, Andrew “Protocols for secure computations” FOCS 1982
4. Nexus 5, Available at: http://www.google.com/nexus/5/
5. Android, Available at: http://www.techterms.com/definition/Android
6. Programming Language Popularity, Available at: http://www.langpop.com/