An Implementation of Secure Two-Party Computation for Smartphones with Application to Privacy-Preserving Interest-Cast

Gianpiero Costantino, Fabio Martinelli, Paolo Santi, Dario Amoruso
IIT-CNR, Pisa, Italy
Email: name.surname@iit.cnr.it

1. GOAL

For this demo, we present for the first time a *feasible* implementation of a cryptographic framework for Secure Multi-Party computation (the FairPlay framework proposed in [1]) on the Android mobile platform. Secure Multi-Party computation is a general framework that can be used for privacy-preserving computation of a variety of functions.

Our application was developed with the aim to: 1) find people in the user's (Alice) neighbourhood through a Bluetooth scan operation, 2) connect to another user (Bob) and discover whether Bob and Alice have similar interest profiles without disclosing sensitive information, and 3) share messages between Alice's and Bob's devices only if their profiles are similar.

2. FAIRPLAY PROJECT

FairPlay [1] is a framework for secure two-party and multi-party [2] computation that allows users to write and run secure functions. In particular, a user writes high-level procedures that are compiled by FairPlay into optimised boolean circuits. Since in mobile environments, and especially in OppNets, interactions are mostly pair-wise, in the following we focus on the two-party version of FairPlay, which is sufficient to our purposes.

3. MOBILEFAIRPLAY

Two main issues have to be addressed in porting FairPlay to a mobile environment such as Android. First, the Java object computed as outcome of the FairPlay compilation phase has to be made compatible with the JavaVM used in the mobile phone, which is different from the standard one. In particular, Android phones use the DalvikVM. The first step in our porting process has then been translating a Java object as produced by FairPlay into a `.dex` file executable on the DalvikVM. Second, FairPlay uses TCP/IP for communication between parties, which is not suitable for setting up and operating a direct radio communication between two smartphones. Instead, we used the Bluetooth interface for communications between the two parties. In particular, when Alice wants to communicate with Bob, she hooks to the Bluetooth socket, and then sends a Bluetooth request to Bob. Finally, the communication between the two parties can start once Bob has accepted Alice's connection request.

4. INTEREST-CASTING IN OPPNETS

User interests can be modeled as an m-dimensional vector in a common m-dimensional *interest space*, where the number m of interest is typically much smaller than the number n of nodes in the network. More formally, the *interest profile* of user A is defined as:

$$I_A = (a_1, \ldots, a_m) \ ,$$

where $a_i \in [1, max]$ is an integer representing A's interest in the i-th topic of the interest space. Note that interests are expressed as integers in the range $[1, max]$, with 1 representing no interest and max (an arbitrary integer > 0) representing maximum interest[1].

Let S be a user denoted as the message *source*. According to the definition of interest-casting, the message M generated by S (which can be thought of as a piece of information node S wants to share within the network) should be delivered to all nodes in the set $\mathcal{D}(S, \gamma)$, where

$$\mathcal{D}(S, \gamma) = \{U \in \mathcal{N} | sim(U, S) \geq \gamma\} \ ,$$

where $sim(U, S)$ is a similarity metric used to express similarity between a node U and S's interest profiles, with relatively higher similarity values representing relatively more similar interests, and γ is the *relevance threshold* (set by S). Set $\mathcal{D}(S, \gamma)$ is called the set of *relevant destinations*, and in principle it is not known to node S. Instead, set $\mathcal{D}(S, \gamma)$ is implicitly defined by S's interest profile, and by the relevance threshold γ.

In the interest-cast implementation presented in the following, we consider the vector-component-wise (vcw) similarity metric defined in [4], which we recall here. Let

[1]The notion of interest profile can be straightforwardly extended to represent also information about a user's habits, such as living in a certain neighborhood, working in a certain place, and so on. For details, see [3].

$S = (s_1, \ldots, s_m)$ and $U = (u_1, \ldots, u_m)$ be the interest profiles of users S and U, respectively. We have:

$$vcw(U, S, \lambda) = \begin{cases} 1 & \text{if } \forall i \in \{1, \ldots, m\},\ |u_i - s_i| \leq \lambda \\ 0 & \text{otherwise} \end{cases},$$

where $\lambda \in [0, max]$ is an integer parameter used to narrow/widen the scope of the interest-cast. More specifically, by setting $\gamma = 1$, we have that $\mathcal{D}(S, 1)$ corresponds to the set of all nodes in the network if $\lambda = max$, while $\mathcal{D}(S, 1) = \{S\}$ if $\lambda = 0$.

5. INTEREST-CASTING WITH MOBILEFAIRPLAY

In the developed application, and a user can:

1. set up his own profile regarding different topics;
2. start a new connection with another user and checking if they have similar interests;
3. wait for incoming connections.

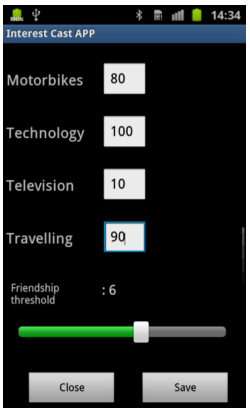

Figure 1: User's profile window

5.1 User profile

When the application is run for the first time, the preference window is shown to the user, see Fig. 1. He/she must insert a value for each topic in the window. The possible values that can be inserted are between 1 and 100, where the lowest value means no interest, and the highest value, maximum interest for a topic. Finally, the user sets the value of λ by means of a slide-bar, and accepts values between 0 and 10.

5.2 Privacy-preserving hand-shaking

This part represents the main section of our App. We consider the case in which Alice is carrying her smartphone, and starts to discovers people around her. Alice decides to connect to another device, which is owned by Bob. Our app manages both the discovery and connection phase with the Bluetooth interface, hence Alice can covers a range of ten-twenty meters around her. Thus, while Bob is waiting for an incoming connection, Alice tries to connect with Bob. After that both devices have been paired, the App works as described in Fig. 2.

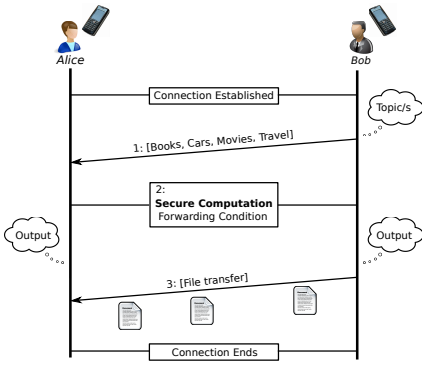

Figure 2: Protocol steps

5.2.1 *Topic/s selection*

When Alice and Bob have been established a new connection, Bob, who received the connection, randomly selects different topics to verify the forwarding condition. As soon as Alice receives the packet containing the topics selected by Bob, they start verifying the forwarding condition in a secure manner.

5.2.2 *Secure Computation of the forwarding condition*

When the Bluetooth connection is established, Bob and Alice are ready to run the secure function. Bob is the first that runs the function and waits for Alice. She uses the Bluetooth socket that has just established to connect with Bob. At this point, they start to run the secure function according to the secure steps implemented in MobileFairPlay. During this execution, both Bob and Alice use their own value of the selected topic, extracted from the interest profile, to compute vcw_e. However, these values are not sent to the other participant in plain, but they are encoded in the garbled Boolean circuits exchanged through MobileFairPlay. This way, at the end of the hand-shaking phase Alice and Bob only knows the result of jointly computing vcw_e, without knowing the specific interest values of the other party.

5.2.3 *Files Transfer*

Once the hand-shaking phase has established that Alice and Bob have similar interests, Bob sends his files to her. Our application is developed to exchange files of any kind and any extension; in fact, raw bytes are exchanged during this phase, allowing users to transfer files of arbitrary format. In our tests, we have successfully transferred text files (`.txt`), pdf files, image files (`.jpg`), etc.

Smartphone	CPU	RAM	Bluetooth Ver.	Android O.S.
Samsung Galaxy S2	Dual-core 1228 MHz	1 GB	3.0	2.3.6
Samsung Galaxy S-Plus	Single-core 1443 MHz	512 MB	3.0	2.3.5
Samsung Galaxy S	Single-core 1024 MHz	512 MB	3.0	2.3.3
Lg Optimus Dual	Dual-core 1024 MHz	512 MB	2.1	2.3.4
Htc Desire	Single-core 1024 MHz	576 MB	2.1	2.2.3

Table 1: Smartphones used for testing the interest-cast application

Smartphone	One Topic (ms)
Samsung Galaxy S2	382
Samsung Galaxy S-Plus	446
Samsung Galaxy S	492
Lg Optimus Dual	449
Htc Desire	489

Table 2: Compilation time required for one topic function

6. COMPUTATIONAL-TIME EVALUATION

Here, we present the results —in terms of execution-time— of two main studies, regarding 1) compilation of the SFDL code, and 2) running of the secure function (hand-shaking phase).

Table 1 reports the specification of the models considered in our evaluation.

6.1 Compiling the secure function

The current version of the APP required that when the compilation phase is run for the first time. In Table 2, the time that each device requires to compile the secure function is shown.

6.2 Running the secure function (hand-shaking)

In this section we evaluate the time that a user, for instance Bob, needs to run the secure function, i.e., the duration of the hand-shaking phase.

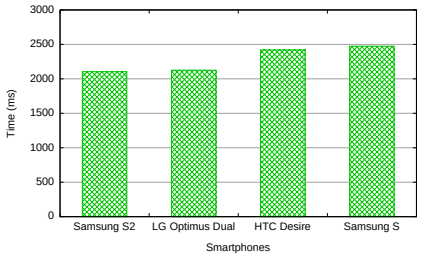

Figure 3: One topic: Bob running time of the secure function

Fig. 3 shows the time needed to run the secure function in the case of one topic comparison. This time includes the time needed to exchange the garbled boolean circuits through the Bluetooth interface, and to compute the output of vcw_e. The results are obtained considering the case in which the role of Alice is kept constant. In fact, while Bob is run four times with four different smartphones, Alice is always run on the Samsung Galaxy S-Plus.

7. OTHER INFO

7.1 Equipment to be used for the demo

For the Demo we have to use at least two smartphones. We may take with us two or three smartphones.

7.2 Space needed and setup time required

A desktop to keep our smartphones. Set up time required is 15 minutes

7.3 Additional facilities needed

We need the power for our smartphones. Internet is optional.

8. REFERENCES

[1] D. Malkhi, N. Nisan, B. Pinkas, and Y. Sella, "Fairplay;a secure two-party computation system," in *Proc. of the 13th conf. on USENIX Security Symposium*, Berkeley, CA, USA, 2004, pp. 20–20.

[2] A. Ben-David, N. Nisan, and B. Pinkas, "Fairplaymp: a system for secure multi-party computation," in *Proc. of the 15th ACM conf. on Comp. and communications security*, ser. CCS '08. ACM, 2008, pp. 257–266.

[3] A. Mei, G. Morabito, P. Santi, and J. Stefa, "Social-aware stateless forwarding in pocket switched networks," in *IEEE Infocom*, 2011.

[4] G. Costantino, F. Martinelli, and P. Santi, "Privacy-preserving interest-casting in opportunistic networks," in *IEEE Wireless Communication and Net.Conf. (WCNC)*, 2012.