Fachhochschule Aachen

Campus Jülich

Fachbereich: Medizintechnik und Technomathematik

Studiengang: Technomathematik

Secure Multi-Party Computation for Decentralized Distributed Systems

Masterarbeit von Frederic Klein

Diese Arbeit wurde betreut von:

1. Prüfer: Prof. Dr. rer. nat. Alexander Voß
2. Prüfer: Dr. Stephan Jonas

Aachen, Dezember, 2016

Diese Arbeit ist von mir selbständig angefertigt und verfasst. Es sind keine anderen als die angegebenen Quellen und Hilfsmittel benutzt worden.

Frederic Klein .....
Unterschrift

Abstract

In recent years gamification has become a part in many areas of our daily routine. In regard to our personal life, companies like Amazon or Runtastic can base their gamification approach on publicly sharing personal achievements and statistics to improve user commitment. In contrast, gamification concerning our work life has to satisfy much higher privacy demands. Since comparison is a key component for gamification, privacy protecting computations of system wide statistical values (for example minimum and maximum) are needed. The solution comes in the form of secure multi-party computation (SMPC), a subfield of cryptography. Existing frameworks for SMPC utilize the Internet Protocol, though access to the Internet or even a local area network (LAN) cannot be provided in all environments. Facilities with sensible measuring systems, e.g. medical devices in hospitals, often avoid Wi-Fi to reduce the risk of electromagnetic interference. To be able to utilize SMPC in environments with Wi-Fi restrictions, this thesis studies the characteristics of mobile ad hoc networks (MANET) and proposes the design of a SMPC framework for MANET, especially based on Bluetooth technology, and the implementation as a C library.

Since MANETs have a high probability for network partition, a centralized architecture for the computation and data preservation is unfavorable. Therefor a blockchain based distributed database is implemented in the framework. Typical problems of distributed systems are addressed with the implementation of algorithms for clock synchronization and coordinator election as well as protocols for the detection of computation partners and data distribution. Since the framework aims to provide distributed computations of comparable values, protocols for secure addition and secure comparison are implemented, enabling the computation of minimum, maximum and average.

bad word high acceptable here?

Devices of diverse computational power will be used to verify the applicability for wearables and Internet of Things (IoT) grade devices. Also field-tests with a smart phone ad hoc network (SPAN)(20-50 nodes) will be conducted to evaluated real life use cases. In contrast, the security of the framework and attack scenarios will be discussed. In summary, this thesis proposes a framework for SMPC for decentralized, distributed systems.

Contents

5-10%, including motivation, general audience

10-15%; thorough review of the state of the art; informed audience

15-20%; explains complete processing chain; explains what methods are used; for someone that wants to know what was done in detail

15-20%; details on the implementation; for someone who wants to continue the work

5-15%; outcome; how was it tested; for supervisor

5-15%; outcome for a design-reader

5-10%; outcome for a introduction-reader

List of Figures

List of Tables

List of Acronyms

2PC secure two-party computation.

DDoS Distributed Denial of Service.

IoT Internet of Things.

LAN local area network.

LSB least significant bit.

MANET mobile ad hoc networks.

MSB most significant bit.

PRNG pseudo-random number generator.

SMPC secure multi-party computation.

SPAN smart phone ad hoc network.

Chapter 1

Introduction

5-10%, including motivation, general audience

In the last couple of years gamification has found it's way into many areas of our daily life. In regard to our personal life, companies like Amazon or Runtastic can base their gamification approach on publicly sharing personal achievements and statistics to improve user commitment. In contrast, gamification concerning our work life can have much higher privacy demands. Since comparison is a key component for the gamification approach, privacy protecting computations of system wide statistical values (for example minimum and maximum) are needed. The solution comes in the form of SMPC, a subfield of cryptography.

Existing frameworks for SMPC utilize the Internet protocol, though access to the Internet or even a LAN cannot be provided in all environments. Especially many hospitals tend to avoid Wi-Fi to reduce the risk of electromagnetic interference with medical devices.

To be able to utilize SMPC in environments with Wi-Fi restrictions, this thesis studies the characteristics of mesh-networks and proposes describes the design of a SMPC framework for mesh-networks.

Context

mention IoT problems (DDoS and botnets), to emphasis usefulness of connected but not online

Restatement of the problem

Restatement of the response

Roadmap

1.1 Case Study: "The Hygiene Games"

Gamification

Wireless Networks in Hospitals

Chapter 2

Background

10-15%; thorough review of the state of the art; informed audience

In this chapter a general understanding of SMPC and the key features of MANETs is established.

First the general idea for SMPC is introduced in 2.1 Secure Multi-Party Computation. Since secret sharing is used for the development of SMPC protocols, Shamir's secret sharing scheme is presented in 2.1.1 Secret Sharing. Protocols for secure addition and secure comparison with passive security are introduced in 2.1.2 and 2.1.3 and existing frameworks for SMPC are discussed in 2.1.4.

To be able to define requirements for the new framework, the key features of MANETs are identified in 2.2 Mobile Ad Hoc Networks, with a focus on the wireless technology standards Bluetooth and Wi-Fi and the differences to similar network types like mesh networks.

Since the SMPC protocols expect a secure communication channel, while a pairing-less connection doesn't provide security by default, public key cryptography is needed. The key generation, as well as Shamir's secret sharing scheme, requires random numbers. Computer systems can generate pseudo-random numbers and the randomness of such an pseudo-random number generator (PRNG) is discussed in2.3 Pseudo-Random Numbers.

2.1 Secure Multi-Party Computation

general idea

SMPC is a subfield of cryptography. The target of SMPC is to run computations over inputs from multiple parties while keeping these inputs secret. In 1982 Yao described the problem of two millionaires trying to find out, which one is wealthier, without giving each

other information about their actual capital (Yao 1982). Yao's solution for this secure two-party computation (2PC) is considered to be the basis for general SMPC protocols. Since the target group for the protocols used in this thesis

resume

use-cases to describe it better

cite Clifton et al. (2002): data mining;

For SMPC two types of adversaries have to be considered: semi-honest adversaries and malicious adversaries. Semi-honest adversaries "follow the protocol specification, yet may attempt to learn additional information by analyzing the transcript of messages received during the execution" (Aumann and Lindell 2007). Malicious adversaries "are not bound in any way to following the instructions of the specified protocol" (Aumann and Lindell 2007). SMPC protocols that can tolerate semi-honest parties (up to a specific threshold) provide semi-honest or passive security. SMPC protocols that are secure against malicious adversaries achieve malicious or active security. Cramer, Damgard, and Nielsen (2015, p. 82) also differentiate between unconditional or perfect security and computational security: if security can be proven for an adversary with unlimited computation power a protocol has unconditional security. In contrast, computational security can only be proven for a polytime adversary.

only for honest parties, what is for other settings?

discuss passive and active security

discuss the situation for gamification: what can be gained? -¿ this thesis focuses on honest but curious "players" -¿ basic sanity checks (see ...)

simple example

2.1.1 Secret Sharing

Cramer, Damgard, and Nielsen (2015, p. 32) describe secret sharing schemes as the main tool to build a SMPC protocol with passive security. In 1979 Adi Shamir described a (k, n) threshold scheme for sharing secret data D: "Our goal is to divide D into n pieces D_i, ..., D_n in such a way that: (1) knowledge of any k or more D_i pieces makes D easily computable; (2) knowledge of any $k-1$ or fewer D_i pieces leaves D completely undetermined (in the sense that all its possible values are equally likely)." (Shamir 1979) Shamir's secret sharing scheme is based on polynomials of degree $k-1$ with $a_0 = D$ (compare 2.1).

$$q(x) = D + a_1 \cdot x + ... + a_{k-1} \cdot x^{k-1} \tag{2.1}$$

To divide D into n pieces the polynomial is evaluated: $D_i = q(i),\ i = 1, ..., n$.

For cryptographic protocols it is not practical to work with real arithmetic, instead a finite field is used. Shamir (1979) specifies that modular instead of real arithmetic is used. A prime p with $p > D, p > n$ is selected and used to define the set $[0, p)$. "The coefficients $a_1, ..., a_{k-1}$ in $q(x)$ are randomly chosen from a uniform distribution over the integers in $[0, p)$, and the values $D_1, ..., D_n$ are computed modulo p." (Shamir 1979, p. 613) (compare 2.2)

$$q(x) = D + a_1 \cdot x + ... + a_{k-1} \cdot x^{k-1} \mod p \qquad D,\ a_i \in [0, p)\ , \quad p \in \mathbb{P} \tag{2.2}$$

Cramer, Damgard, and Nielsen (2015, p. 7) declare the set restricted by p as $\mathbb{Z}_p = \{0, 1, ..., p-1\}$. They also use the notion *secret* S for the data to be shared and *shares* s_i for the computed pieces of the secret.

describe number off messages, usage of threshold as trade-off between security and performance -¿ START

The reconstruction of a secret S can be done using Lagrange interpolation (compare 2.3).

$$S = \sum_i s_i \prod_{i \neq j} \frac{-x_j}{x_i - x_j} \mod p \tag{2.3}$$

describe number off messages, usage of threshold as trade-off between security and performance -¿ END

k shares s_i are needed to reconstruct S, so only the associated values for i are used in the Lagrange interpolation.

Example Computation

Consider the following task: a secret $S = 8$ is supposed to be shared among $n = 4$ parties $P_i, \quad i = 0, ..., 3$. The threshold for the number of needed shares for the reconstruction of the secret shall be $k = 3$ (public).

First a prime p has to be chosen, which has to be larger than the secret ($p > S$) and the number of parties ($p > n$): $p = 17$ (public)

Since $k = 3$, the polynomial has a degree of $k - 1 = 2$ (compare 2.4).

$$f(x) = S + a_1 \cdot x + a_2 \cdot x^2 \mod p \tag{2.4}$$

The coefficients are selected randomly uniformly out of $\mathbb{Z}_p = \{0, 1, ..., p-1\} =$

$\{0, 1, ..., 16\}$: $a_1 = 13$ and $a_2 = 4$ and the shares s_i are computed (compare 2.5).

$$
\begin{aligned}
f(x) &= 8 + 13 \cdot x + 4 \cdot x^2 \mod 17 \qquad (2.5)\\
&\Downarrow \\
f(x_1) &= f(1) = 25 \mod 17 = 8 = s_1 \\
f(x_2) &= f(2) = 50 \mod 17 = 16 = s_2 \\
f(x_3) &= f(3) = 83 \mod 17 = 15 = s_3 \\
f(x_4) &= f(4) = 124 \mod 17 = 5 = s_4
\end{aligned}
$$

If parties P_2, P_3 and P_4 pool their shares, they can reconstruct the secret S using Lagrange interpolation (using also the public information: $p = 17$):

$$
\begin{aligned}
S &= \sum_i s_i \prod_{i \neq j} \frac{-x_j}{x_i - x_j} \mod 17 \qquad \textit{with } i, j \in \{2, 3, 4\} \qquad (2.6)\\
&= s_2 \cdot \frac{-x_3}{x_2 - x_3} \cdot \frac{-x_4}{x_2 - x_4} + s_3 \cdot \frac{-x_2}{x_3 - x_2} \cdot \frac{-x_4}{x_3 - x_4} + s_4 \cdot \frac{-x_2}{x_4 - x_2} \cdot \frac{-x_3}{x_4 - x_3} \mod 17 \\
&= 16 \cdot \frac{-3}{2 - 3} \cdot \frac{-4}{2 - 4} + 15 \cdot \frac{-2}{3 - 2} \cdot \frac{-4}{3 - 4} + 5 \cdot \frac{-2}{4 - 2} \cdot \frac{-3}{4 - 3} \mod 17 \\
&= 96 - 120 + 15 \mod 17 \\
&= -9 \mod 17 \qquad (2.7)\\
&= 8
\end{aligned}
$$

Note: in cryptography $a \mod n$ for $a < 0$ (negative dividend) is calculated by adding a multiple of n, so that $m \cdot n + a > 0$: e.g. $-9 \mod 17 = (\underbrace{1 \cdot 17 - 9}_{>0}) \mod 17$ (compare 2.7).

Write about differential privacy?

2.1.2 Secure Addition Protocol

For an environment with honest parties there are simple SMPC protocols to compute the sum over shares. Clifton et al. (2002) describe a ring based method, where the initializing party adds a random number R to the secret input s_1 before passing it to the next node. Each node then adds its secret until the first party receives the result. By removing R the party can than reconstruct the sum over all secret inputs:

visualize as svg figure

This method is efficient ($2n$ messages for computation and announcing the sum in a n-node ring) but if parties collude, party P_i only needs the output of P_{i+1} as received by party P_{i+2} to reconstruct the secret input of P_{i+1}. Clifton et al. (2002) propose using shares in combination with permutation of the ring order, so neighbors change in each iteration and the number of parties in need to pool their data increases. This approach was extended by Sheikh, Kumar, and Mishra (2009) for their "k-Secure Sum Protocol". Using Shamir's secret sharing a In 2.1.1 it was demonstrated how a secret can be reconstructed from the shares using Lagrange interpolation. It is also possible to reconstruct the sum of secrets by using the sums of shares for a Lagrange interpolation.

Proof:

n shares for m secrets s_l:

$$s_{l,i} = f_l(x_i) = s_l + \sum_{i=1}^{k-1} \alpha_{l,i} x_i^i \mod p \tag{2.8}$$

$$\Leftrightarrow \begin{cases} s_{1,i} = f_1(x_i) = & s_1 + \alpha_{1,1}x_i + \alpha_{1,2}x_i^2 + ... + \alpha_{1,k-1}x_i^{k-1} \mod p \\ & \vdots \\ s_{m,i} = f_n(x_i) = & s_n + \beta_{n,1}x_i + \beta_{n,2}x_i^2 + ... + \beta_{n,k-1}x_i^{k-1} \mod p \end{cases}$$

$$\text{with } \{l \in \mathbb{N} \mid 1 \leq l \leq m\},\ \{i \in \mathbb{N} \mid 1 \leq i \leq n\},\ \{p \in \mathbb{P} \mid p > \sum_l s_l\},$$

$$\{\alpha \in \mathbb{N} \mid 0 \leq \alpha \leq p\},\ \{k \in \mathbb{N} \mid 2 < k \leq n\}$$

Lagrange-interpolation for secret s_l:

$$s_l = \sum_{i=1}^{n} s_{l,i} \prod_{i \neq j} \frac{-x_j}{x_i - x_j} \mod p \tag{2.9}$$

Sum s over secrets s_l:

$$s = \sum_{l=1}^{m} s_l \overset{\text{with } 2.9}{=} \sum_{l=1}^{m} \sum_{i=1}^{n} s_{l,i} \prod_{i \neq j} \frac{-x_j}{x_i - x_j} \mod p \tag{2.10}$$

$$\text{with} \sum_{i=1}^{n}\sum_{j=1}^{m} a_{ij} = \sum_{j=1}^{m}\sum_{i=1}^{n} a_{ij} \text{follows for 2.10}$$

$$\underbrace{s = \sum_{i=1}^{n} \underbrace{\sum_{l=1}^{m} s_{l,i}}_{\text{sum over shares}} \prod_{i \neq j} \frac{-x_j}{x_i - x_j} \mod p}_{\text{Lagrange-interpolation for sum over shares}} \tag{2.11}$$

Example Computation

Public information: $n = 3$, $p = 67$, $k = 3$

Secrets: $s_1 = 13$, $s_2 = 27$, $s_3 = 17$, $s_4 = 1$

Target computation: sum s over secrets $s = \sum_{i=1}^{4} s_i = 58$ without revealing ones secret to another party.

$$s_{1,i} = f_1(x_i) = 13 + 35x + 22x^2 + 7x^3 \mod 67 \tag{2.12}$$

$$s_{2,i} = f_2(x_i) = 27 + 3x + 19x^2 \mod 67 \tag{2.13}$$

$$s_{3,i} = f_3(x_i) = 17 + 9x^2 + 27x^3 \mod 67 \tag{2.14}$$

$$s_{4,i} = f_4(x_i) = 1 + 13x + 31x^2 + 40x^3 \mod 67 \tag{2.15}$$

with $x_1 = 1$, $x_2 = 2$, $x_3 = 3$, $x_4 = 4$ follows

$$\begin{aligned}
&\overset{2.12}{\Rightarrow} s_{1,1} = 10 && s_{1,2} = 26 && s_{1,3} = 36 && s_{1,4} = 15 \\
&\overset{2.13}{\Rightarrow} s_{2,1} = 49 && s_{2,2} = 42 && s_{2,3} = 6 && s_{2,4} = 8 \\
&\overset{2.14}{\Rightarrow} s_{3,1} = 53 && s_{3,2} = 1 && s_{3,3} = 23 && s_{3,4} = 13 \\
&\overset{2.15}{\Rightarrow} s_{4,1} = 18 && s_{4,2} = 2 && s_{4,3} = 59 && s_{4,4} = 27 \\
\Rightarrow &\sum_{l} s_{l,1} = 130 && \sum_{l} s_{l,2} = 71 && \sum_{l} s_{l,3} = 124 && \sum_{l} s_{l,4} = 63
\end{aligned}$$

Lagrange-interpolation:

$$\begin{aligned} s =& \sum_{i=1}^{4}\sum_{l=1}^{4} s_{l,i} \prod_{i\neq j} \frac{-x_j}{x_i - x_j} \mod 67 \\ =& 130\frac{-2}{1-2}\frac{-3}{1-3}\frac{-4}{1-4} + 71\frac{-1}{2-1}\frac{-3}{2-3}\frac{-4}{2-4} \\ &+ 124\frac{-1}{3-1}\frac{-2}{3-2}\frac{-4}{3-4} + 63\frac{-1}{4-1}\frac{-2}{4-2}\frac{-3}{4-3} \mod 67 \\ =& 527 \mod 67 = 58 = \sum_{i=1}^{4} s_i \end{aligned} \tag{2.16}$$

As expected, the result of the Lagrange-interpolation for the sum over shares is equal to the sum over the initial secrets (compare 2.16).

Protocol Description

Assumptions:

- number of parties $n > 2$
- secure communication channel
- no malicious adversaries
- upper bound of sum $s \leq b$ can be estimated, so a prime $p > b$ can be chosen

The secure addition protocol, as used in this thesis, consists of six phases:

1. The coordinator announces the number of parties for the computation and the indexation of each party.
2. Each party j sends shares $s_{j,i}$ of the secret input s_j to the other parties.
3. Each party i computes the sum over the received shares $s_{j,i}$.
4. Each party sends the computed sum to the coordinator.
5. The coordinator reconstructs the sum over the inputs using Lagrange-interpolation.
6. The coordinator broadcasts the reconstructed sum.

In total $(n+3)\cdot(n-1) = n^2+2n-3$ messages are exchanged, so the traffic increases with the number of parties squared. Selecting a lower threshold for the secret reconstruction $\frac{n}{2} \leq k < n$ lowers the total messages by $\Delta_{\text{messages}} = n^2 - n(k-1)$.

For a secure channel this protocol is information-theoretically secure: independent from computation power an adversary with $m_{\text{leaked}} < k$ shares will gain no information regarding the inputs.

secure addition with verification (Cramer, Damgard, and Nielsen 2015); number of messages

2.1.3 Secure Comparison Protocol

The secure comparison protocol compares the secret inputs and provides the minimum and the maximum in a set without revealing the the inputs or the parties holding the minimum or the maximum.

The general idea: the secure comparison protocol utilizes the secure addition protocol: in iterations the secure-sum for the bits $(0 \vee 1)$ of the secrets multiplied with a random value are computed, starting with the most significant bit (MSB) lower than a predefined upper bound till the least significant bit (LSB). The announced sum gives each party the information if at least one party has this bit set, if the sum is unequal zero. If a party has this bit not set itself it has a lower value and commits only zeros to the following iterations. Storing the result of each iteration, the parties can reconstruct the maximum. For finding the minimum the inputs are negated (using the binary operation NOT), making the minimum in the set the largest value. Afterwards the maximum is determined as described above. Finally the found maximum is negated again to reconstruct the minimum in the set.

Example Computation

Public information: $n = 3$, $p = 67$, $\mathbb{Z}_p = \{1, ..., p-1\}$, $k = 3$, $s_i < b = 64$ (upper bound for secret value range)
Secrets: $s_1 = 13$, $s_2 = 27$, $s_3 = 17$
Target computation: $\min(s_i) = 13$, $\max(s_i) = 27$

Since $64_{10} = 1000000_2$ is defined as upper bound for the secret values the MSB is the sixth bit (second column in table 2.1).

Table 2.1: Binary representation of secrets s_i

Decimal $s_{i,10}$	Binary $s_{i,2}$					
13	0	0	1	1	0	1
27	0	1	1	0	1	1
17	0	1	0	0	0	1

Each party multiplies each bit with a random within $\mathbb{Z}_l$:

Table 2.2: Randomized binary representation of secrets

Decimal $s_{i,10}$	Binary $s_{i,2}$						Randomized					
13	0	0	1	1	0	1	0	0	45	61	0	57
27	0	1	1	0	1	1	0	12	31	0	5	15
17	0	1	0	0	0	1	0	24	0	0	0	9

There are six bits, therefore six rounds of secure addition ($\sum_{secure}$) are computed:

$$1^{st} \text{ round:} \quad \sum_{secure} = 0 \quad \Rightarrow \quad 6^{th} \text{ bit of the maximum is } \quad 0$$

$$2^{nd} \text{ round:} \quad \sum_{secure} = 36 > 0 \quad \Rightarrow \quad 5^{th} \text{ bit of the maximum is } \quad 1$$

Party p_1 disqualifies itself as the maximum (see table 2.3)

$$3^{rd} \text{ round:} \quad \sum_{secure} = 31 > 0 \quad \Rightarrow \quad 4^{th} \text{ bit of the maximum is } \quad 1$$

Party p_3 disqualifies itself as the maximum (see table 2.4)

$$4^{th} \text{ round:} \quad \sum_{secure} = 0 \quad \Rightarrow \quad 3^{rd} \text{ bit of the maximum is } \quad 0$$

$$5^{th} \text{ round:} \quad \sum_{secure} = 5 > 0 \quad \Rightarrow \quad 2^{nd} \text{ bit of the maximum is } \quad 1$$

$$6^{th} \text{ round:} \quad \sum_{secure} = 15 > 0 \quad \Rightarrow \quad 1^{st} \text{ bit of the maximum is } \quad 1$$

Table 2.3: Party p_1 disqualifies itself as maximum in 2^{nd} round

Decimal $s_{i,10}$	Randomized					
13	0	$\underline{0}$	$\not{45}^{0}$	$\not{61}^{0}$	0	$\not{57}^{0}$
27	0	12	31	0	5	15
17	0	24	0	0	0	9

Table 2.4: Party p_3 disqualifies itself as maximum in 2^{nd} round

Decimal $s_{i,10}$	Randomized					
13	0	0	0	0	0	0
27	0	12	31	0	5	15
17	0	24	$\underline{0}$	0	0	$\not{9}^{0}$

In total, each party has the bits 0|1|1|0|1|1 stored and can reconstruct the correct

maximum $\max(s_i) = 27$.

Using the negation of the binary representation, the order of the corresponding values in decimal numeral system is inverted (compare table 2.5). The computation is then the same as for the maximum search. The reconstructed maximum is finally negated to result in $\min(s_i)$.

Table 2.5: Negation of binary representation for minimum determination

Decimal $s_{i,10}$	Binary $s_{i,2}$						Negated					
13	0	0	1	1	0	1	1	1	0	0	1	0
27	0	1	1	0	1	1	1	0	0	1	0	0
17	0	1	0	0	0	1	1	0	1	1	1	0

example computation

Protocol Description

Assumptions:

- number of parties $n > 2$
- secure communication channel
- no malicious adversaries
- upper bound of sum $s \leq b$ can be estimated, so a prime $p > b$ can be chosen

The secure comparison protocol, as used in this thesis, consists of the phases for secure addition within iterations for the bitwise length of a predefined upper bound for the inputs:

1. The coordinator announces the number of parties for the computation and the indexation of each party.
2. For minimum-search: each party negates the secret input.
3. For each bit in the secret input starting from MSB to LSB each party runs through iterations:
   (a) If input is flagged as lower than maximum, then use $s_j = 0$ as the input. Otherwise multiply actual bit b with a random value R: $s_j = b \cdot R$.
   (b) Each party j sends shares $s_{j,i}$ of the input s_j to the other parties.
   (c) Each party i computes the sum over the received shares $s_{j,i}$.
   (d) Each party sends the computed sum to the coordinator.
   (e) The coordinator reconstructs the sum over the inputs using Lagrange-interpolation.

(f) The coordinator broadcasts the reconstructed sum.

(g) Each party stores if the sum for the bit was equal 0 (set bit 0) or unequal 0 (set bit 1).

(h) Each party compares if bit from the computed sum is greater than own bit. If so input is flagged as lower than maximum.

4. For minimum-search: each party negates the stored sum-result.

Note: the assumption $n > 2$ for the secure addition and secure comparison protocols is not strict enough, if sum, min and max are computed for the same parties, since for $n = 3$ the secret between minimum and maximum can be restored (the mapping of values to parties is still secure though).

2.1.4 Existing Frameworks

SMCL discontinued -¿ VIFF discontinued -¿ SPDZ

maybe: mention FairplayMP, but also abandoned, issue from 2015 unanswered

SPDZ Software

link

SPDZ Software

SEPIA

link

SEPIA Java lib

MpcLib

link

MpcLib: no documentation

Sharemind

link

sharemind

2.2 Mobile Ad Hoc Networks

differences between mesh and MANET

- continuously self-configuring

- self-forming
- self-healing
- infrastructure-less
- peer-to-peer
- Difference to mesh: mobility of nodes

Example: firechat in SPAN

2.2.1 Comparison to Wi-Fi Direct

- SPAN support multi-hop relays
- Wi-Fi Direct since Android 4.0
- Wi-Fi Direct: Soft AP

2.2.2 Bluetooth Based MANET

2.2.3 Wi-Fi Based MANET

mention https://github.co manet-manager kernel changes needed for manet

2.3 Pseudo-Random Numbers

random numbers important for cryptography: selection of coefficients in secret sharing, public key generation, ...; RNG in different environments; entropy

lib will require a callback for random number generator -¿ maybe mention with outlook for requirements

Chapter 3

Design

15-20%; explains complete processing chain; explains what methods are used; for someone that wants to know what was done in detail

3.1 Requirements

use cases, process description, resulting requirements

3.2 Distributed Computing

Coordinator Election

exponential back-off + heartbeat

Clock Synchronization

berkeley algorithm

Distributed Databases

3.3 Applicability of SMPC Protocols in MANETs

Analysis of Key Factors: Computing Power, Network Data Rates and Duration of Connection

Effectiveness of SMPC Protocols in Sparse Networks

Maintaining anonymity

Strategies for Aggregation of Participants in Sparse Networks

3.4 Architecture

UML; module structure

Chapter 4

Implementation

15-20%; details on the implementation; for someone who wants to continue the work

4.1 Communication Layer

Pairing-less Connection

Secure Channel

https://developer

4.2 SMPC Module

4.3 Data Storage and Distribution

4.4 Interfacing the Library

Configuration

Usage in C

Usage in Android

Chapter 5

Evaluation

5-15%; outcome; how was it tested; for supervisor

5.1 Testing Tools

Unity (Unit test for C); JUnit; Simulation?

centralized client-server test application for android: trigger test runs, report results (measured execution time, correctness)

5.2 Examination of Computation Time Dependent on Computing Power

5.3 Examination of Computation Time Dependent on Number of Participants

Chapter 6

Discussion

5-15%; outcome for a design-reader

extend protocol: implement merging of results, to reduce probability of not finding computation partner; implement alternative protocols

implement optional verification for addition, if performance is good enough for real life application

Chapter 7

Conclusion

5-10%; outcome for a introduction-reader

outlook: bt 5.0, mesh network

References

Aumann, Yonatan and Yehuda Lindell (2007). "Security Against Covert Adversaries: Efficient Protocols for Realistic Adversaries". In: *Theory of Cryptography: 4th Theory of Cryptography Conference, TCC 2007, Amsterdam, The Netherlands, February 21-24, 2007. Proceedings*. Ed. by Salil P. Vadhan. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 137–156. ISBN: 978-3-540-70936-7. DOI: 10.1007/978-3-540-70936-7_8. URL: http://dx.doi.org/10.1007/978-3-540-70936-7_8.

Clifton, Chris et al. (2002). "Tools for Privacy Preserving Distributed Data Mining". In: *SIGKDD Explor. Newsl.* 4.2, pp. 28–34. ISSN: 1931-0145. DOI: 10.1145/772862.772867. URL: http://doi.acm.org/10.1145/772862.772867.

Cramer, Ronald, Ivan Bjerre Damgard, and Jesper Buus Nielsen (2015). *Secure Multiparty Computation and Secret Sharion*. Cambridge University Press.

Shamir, Adi (1979). "How to Share a Secret". In: *Communications of the ACM*.

Sheikh, Rashid, Beerendra Kumar, and Durgesh Kumar Mishra (2009). "Privacy Preserving k Secure Sum Protocol". In: *CoRR* abs/0912.0956. URL: http://arxiv.org/abs/0912.0956.

Yao, Andrew C. (1982). "Protocols for Secure Computations". In: *Proceedings of the 23rd Annual Symposium on Foundations of Computer Science*. SFCS '82. Washington, DC, USA: IEEE Computer Society, pp. 160–164. DOI: 10.1109/SFCS.1982.88. URL: http://dx.doi.org/10.1109/SFCS.1982.88.

Appendix A

Some name

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.