New Advances on Privacy-Preserving Policy Reconciliation

Ulrike Meyer[1], Susanne Wetzel[2], and Sotiris Ioannidis[3]

[1]RWTH Aachen University, Germany [2]Stevens Institute of Technology, Hoboken, US
[3]Foundation for Research and Technology, Hellas (FORTH)

July 30, 2010

Abstract

Entities define their own set of rules under which they are willing to collaborate, e.g., interact, share and exchange resources or information with others. Typically, these individual *policies differ for different parties. Thus, collaboration requires the resolving of differences and reaching a consensus. This process is generally referred to as* policy reconciliation.

Current solutions for policy reconciliation do not take into account the privacy concerns of reconciliating parties. This paper addresses the problem of preserving privacy during policy reconciliation. We introduce new protocols that meet the privacy requirements of the organizations and allow parties to find a common policy rule which *optimizes* their individual preferences.

1 Introduction

Enabling collaboration between organizations is a very challenging task. The main difficulty lies in determining the policy rules that should govern this process. That is, the (two or more) participants must accept a set of rules that they will follow for the duration of the collaboration. Each party has to express their desired rules in a common format and follow a protocol with the other parties. The process will determine which rules control their future collaboration. This protocol is called a *reconciliation protocol* and its output is a policy that is consistent with the requirements of all the participants. If such a policy exists, the participants can continue their collaboration. Otherwise, they can decide not to collaborate or they can decide to modify their individual requirements and repeat the protocol.

Consider the following example: A company has posted a job opening. Multiple applications have been received and HR would like to schedule interviews with some of the applicants. To successfully schedule an interview, both HR and the respective applicant need to reach an agreement on the date and time for the interview. I.e., they need to reconcile their policies, which—in this case—regulate the dates and times when the individual parties are available for the interview.

Obviously, this is a simple example with respect to policy reconciliation and it is very unlikely that the two parties will make use of any sophisticated reconciliation methods for scheduling such an interview. However, we believe this example highlights some of the shortcomings of current policy reconciliation processes. To date, policy reconciliation does not consider privacy. Continuing with our example, either party may want certain information not to be disclosed to the other party during the reconciliation process, i.e., when trying to find a date/time for the interview. For example, HR may not want to disclose its schedule to the applicants in order not to allow for the applicants to infer the number of applicants interviewed for the opening. Similarly, an applicant may not want to disclose its schedule to the HR department of its potential new employer as the applicant may not want them to learn certain aspects of his personal circumstances, e.g., regularly scheduled doctor's appointments or a second job. Yet, state-of-the-art policy reconciliation mechanisms require that at least one of the parties discloses all the information in order to allow for the reconciliation process to work.

It is in this context that this paper introduces preferences and privacy into the policy reconciliation processes and as such provides for a major advancement in this area. In particular, with respect to privacy, we propose new two-party protocols that guarantee that parties participating in the reconciliation process learn nothing about the other party's policies other than the policy they will eventually agree on. In our initial example above, both the applicant and the HR department will learn nothing about each other's schedules other than the date and time at which the interview of this particular applicant will take place (or the fact that there is no mutually agreeable time and date if such a match cannot be found). In addition, our work is the first to introduce mechanisms that consider the privacy requirements and still permit optimizing individual preferences that the parties may have. For example, HR may have preferences on interviewing applicants based on its expectation to hear back from other applicants to which they have already made an offer. If a higher ranked applicant accepts, this will eliminate the need to interview other applicants and thus result in a saving of time. In this paper, we introduce a fair mechanism for parties to reconcile their policies such that the preferences of all parties are optimized while none of the parties will have to disclose any of its preferences.

Outline: The remainder of the paper is organized as follows: In the next section, we discuss previous work in the area of policy reconciliation and privacy preserving set intersection. In Section 3, we provide details on the policy representation, define the new privacy objectives, and review existing privacy-preserving tools. The main contributions of this paper are in Section 4 where we present and discuss our new privacy-preserving policy reconciliation protocols which allow the optimization of the preferences of the individual parties. In Appendix A, we additionally outline some privacy-preserving policy reconciliation protocols without considering preferences.

2 Related Work

In general, a policy is a collection of rules that express which actions are permitted and disallowed in a system [22, 2, 29, 27]. They can take the form of specific access control rules [28], policy credentials [4, 3, 11, 10], or security policy language statements [8, 26]. In this paper, we address policies that govern collaboration and communication between a number of parties. In particular, these policies specify the algorithms, protocols, and any other parameters that must be agreed upon to ensure that the conditions and requirements of all parties are met.

Policy Reconciliation In [15] Gong and Qian analyze the complexity of secure interoperation between systems with heterogeneous access control structures. They prove that composing authorization policies is NP-complete. McDaniel and Prakash define the Ismene policy language [23] which permits the specification and reconciliation of group security requirements. Policy rules are expressed as conditionals which contain zero or more predicates that need to be satisfied for the policy to hold. In [24] Mc Daniel *et al.* specify an algorithm for efficient two-policy reconciliation and show that, in the worst-case, reconciliation of three or more policies is intractable.

Wang *et al.* demonstrate that it is possible to structure security policies in such a way that policy reconciliation becomes tractable [30]. Specifically, they use a graphical policy representation. A graphical policy is a series of policy operations represented in a directed acyclic graph with a single root. One reads policies starting from the root node. The authors present a recursive algorithm for generating the conjunction of graphical policies. The conjunction results in the reconciled policy. Their scheme also allows for specifying preferences. In that case, the authors rank the graphical policies in preferential order before running their conjunction algorithm.

Zao *et al.* propose the Security Policy System (SPS) which resolves IPsec security associations between domains of communication [32]. Reconciliation is achieved by intersecting sets of policy values. Dinsmore *et al.* [9] present the Dynamic Cryptographic Context Management project in which security policies are negotiated between dynamic groups of participants. Their protocol involves multiple rounds of negotiation between the participants, eventually producing a common policy that all of them agree upon.

Our work focuses on ensuring privacy during policy reconciliation and on taking preferences into account. Previous work on policy reconciliation focused on how to make policy reconciliation tractable with respect to performance under various conditions without taking privacy concerns of the participants into account. For this purpose, McDaniel *et al.* [24] introduced a tree-based policy representation. In this paper, we represent a policy as a matrix and each policy rule as a row in the matrix. This is very simple and straightforward to use, especially when optimizing for privacy and not for performance.

Privacy-Preserving Protocols Freedman *et al.* [12] proposed the first protocol specifically designed for private intersection of datasets based on representing set elements as roots of polynomials. The proposed protocol is a two-party protocol, which, in its simplest version, is secure in the semi-honest model. This seminal work was extended by Kissner *et al.* [19, 20] to multiple parties and more set operations on multisets. In particular they propose privacy-preserving protocols for set intersection, set union, and element reduction of multisets. In order to reach security in the malicious model Kissner *et al.* propose the use of a zero-knowledge construction. Hohenberger *et al.* [17] constructed a more efficient two-party version of set intersection that is secure against malicious provers. Camenisch *et al.* in [5] suggest to make use of a trusted third party that certifies the sets of both parties in order to protect against either party being malicious and in order to ensure that sets cannot be changed on several subsequent interactions. All of the approaches mentioned so far are based on the oblivious polynomial evaluation construction introduced by Freedman *et al.* [12].

Another approach to private set intersection is based on oblivious pseudo-random function evaluation. This approach was first introduced by Hazay *et al.* [16] and subsequently further improved by Jarecki *et al.* [18]. The two-party protocol suggested in [16] is more efficient than the protocols based on oblivious polynomial evaluation. However, it is secure only in a relaxed version of the malicious model. Jarecki *et al.* [18] improved this protocol to provide security in the standard malicious model with the help of requiring the parties to commit on their input prior to the private set intersection. The computational effort of the suggested protocol is linear in the cardinality of the input sets.

Finally, and most recently, a private set intersection variant based on blind RSA signature schemes was proposed by Cristofaro *et al.* in [7]. The proposed construction was first introduced in [6] for privacy-preserving policy-based information transfer. The two-party private set intersection protocol proposed in [7] is linear in the maximum of the cardinalities of the two private input sets and thus as efficient as the oblivious pseudo-random function construction by Jarecki *et al.* [18]. Cristofaro *et al.* show that using pre-computation their protocol can outperform all previously suggested protocols. Unfortunately, in practice, pre-computation is not always usable. In particular, in the case of policy reconciliation, the private set of each party may change over time and depends on the context in which the parties interact.

The main contribution of our paper is that we build on the privacy-preserving set intersection results to construct two new and more complex privacy-preserving protocols for policy reconciliation with *preferences* for *two different notions of fair reconciliation*. While any of the aforementioned private intersection protocols can be used in our new construction, we detail our approach for the set intersection protocol introduced by Freedman *et al.* in [12]. As we will show later on, this particular set intersection protocol allows for a reduction of the communicational overhead in our construction. In the appendix, we additionally demonstrate how the existing results can be applied to the problem of security policy reconciliation in the absence of preferences in a straight-forward manner.

Kursawe *et al.* [21] address a similar problem of reconciling privacy policies in a privacy-preserving manner. Their solution is based on modeling the function to evaluate as boolean circuits and evaluating it on inputs encrypted with a threshold homomorphic cryptosystem. As opposed to this, we use privacy-preserving set intersections [12] for our reconciliation protocols. While Kursawe *et al.* recognize the objective of privacy-preserving preference-maximizing policy negotiation, their proposed solution is restricted to the problem of finding the intersection of two sets of policy rules. As opposed to this, we provide solutions for both, the privacy-preserving common policy as well as the preference-maximizing objective. To the best of our knowledge our protocols are the first solutions proposed for the preference-maximizing objective.

	Applicant					HR				
	Mo	*Tue*	*Wed*	*Thu*	*Fri*	*Mo*	*Tue*	*Wed*	*Thu*	*Fri*
Rule 1	1	0	0	0	0	0	0	1	0	0
Rule 2	0	1	0	0	0	0	1	0	0	0
Rule 3	0	0	1	0	0	1	0	0	0	0
Rule 4	0	0	0	1	0	0	0	0	1	0
Rule 5	0	0	0	0	1	0	0	0	0	1

Figure 1: Policy rules expressed as bit strings and ranked in order of preference.

3 Preliminaries

In this section, we summarize the definitions, notation, and objectives used throughout this paper. Furthermore, we briefly review existing privacy-preserving primitives that we use as tools in our new privacy-preserving policy reconciliation protocols introduced in Section 4.

3.1 Policy Representation

To reconcile their policies, organizations must first agree on a common representation. This allows them to compare their policies and determine whether a common subset exists. Otherwise the problem of policy reconciliation becomes intractable [15, 24]. In this work, we represent a policy as a matrix. The rows represent individual policy rules and the columns are the attributes for each rule. Consequently, policies are represented as bit strings, which can be easily manipulated and operated on.

We illustrate our policy representation by continuing the example from Section 1. The first step is for the participants to agree on a set of attributes that will be used to define policy rules. In our example, we have settled for five attributes defining the day of the week on which the interview could take place, *Monday*, *Tuesday*, *Wednesday*, *Thursday*, and *Friday.* (Obviously, in reality this would be much more fine grained, i.e., one attribute could be *Monday: 8:00-9:00AM.*)

In some scenarios finding common policy rules will be the main objective of the participants. However, in some scenarios, the participants may not only want to express which combinations of attributes are acceptable for them but may also want to order these combinations according to their preferences. In these cases the objective of the participants is to find a common policy rule that maximizes their preferences.

Taking preferences into account, each participant represents its requirements by defining rules in order of preference. In our example, the applicant prefers to do the interview on Monday, but could do any other day with lower preference. In turn, HR prefers to do the interview on Wednesday but can accommodate any of the other days as well.

Definition 3.1 *A policy rule $a_i = (a_{i1}, \ldots, a_{in}) \in \{0,1\}^n$ is a bit string of length n indicating whether a field $a_{ij},\ j = 1, \ldots, n$ is defined or not. A policy P_A is a set of rules $a_1, \ldots, a_k$ represented as a matrix $P_A = (a_{ij})_{1 \le i \le k, 1 \le j \le n} \in \{0,1\}^{k \times n}$. Attribute $\mathcal{A}_j$ is a string of characters that uniquely represents the j-th field of the policy rules.*

In our example from Figure 1, *Monday*, *Tuesday*, *Wednesday*, *Thursday*, and *Friday* are the attributes defined in the policy. Rule a_1 for the applicant is the bit string $(1, 0, 0, 0, 0)$ specifying that his top priority is to do the interview on a *Monday.*

Remark 3.1 *Assuming parties A and B with policies P_A and P_B, attributes $\mathcal{A}_1, \ldots, \mathcal{A}_u$ and $\mathcal{B}_1, \ldots, \mathcal{B}_v$, policy reconciliation for these parties requires $u = v$ and $\mathcal{A}_j = \mathcal{B}_j$ for $j = 1, \ldots, u$. I.e., the policy rules of both parties have the same length and both parties associate the same attribute with each bit in the policy rule.*

Definition 3.2 *A pre-order on a set X is a reflexive, transitive, binary relation R on X. A pre-order is total if all $x_1, x_2 \in X$ are comparable, i.e.,$x_1 R x_2$ or $x_2 R x_1$.*

Remark 3.2 *Let X be a set and f be a real-valued function. Then f induces a total pre-order R on X, by xRy iff $f(x_1) \leq f(x_2)$.*

Remark 3.3 *If f is a bijective function, then f induces a total order on X.*

Throughout this paper we assume that a party A is able to totally order the policy rules in its policy P_A according to its preferences. We denote $P_A = \{a_1, \dots, a_k\}$, where a_1 is the rule that is most preferred and a_k is the rule that is preferred least. That is the policy rules are listed in decreasing order $a_k \leq_A a_{k-1} \leq_A \cdots \leq_A a_1$. In other words we assume that the policy rules in P_A are numbered such that $f(a_i) := k - i + 1$ ($i = 1, \dots, k$) is a bijective function that induces a total order $\leq_A$ on P_A. In the sequel, we refer to this function $f(a_i)$ as $\mathrm{rank}_A(a_i)$, *i.e.,* the rank which induces a so-called simple preference order $\leq_A$ on the set of policy rules P_A.

Definition 3.3 *Let $\leq_A$ be the simple preference order induced on the set of policy rules P_A (by $rank_A$) and $\leq_B$ be the simple preference order induced on the set of policy rules P_B (by $rank_B$). A combined preference order $\leq_{AB}$ is a total pre-order induced on the set of policy rules $P_A \cap P_B$ by a the real-valued function f—in the sequel referred to as preference order composition scheme.*

We focus on two specific preference order composition schemes:

Definition 3.4 *The* sum of ranks *composition scheme combines the simple preference orders $\leq_A$ induced by $rank_A$ on P_A and $\leq_B$ induced by $rank_B$ on P_B to the combined preference order $\leq_{AB}$ induced by f on $P_A \cap P_B$ defined as $f_{SoR}(c) := rank_A(c) + rank_B(c)$ for $c \in P_A \cap P_B$. I.e., if $x, y \in P_A \cap P_B$, then $x \leq_{AB} y :\Leftrightarrow rank_A(x) + rank_B(x) \leq rank_A(y) + rank_B(y)$.*

Definition 3.5 *The* maximized minimum of ranks *composition scheme combines the simple preference orders $\leq_A$ induced by $rank_A$ on P_A and $\leq_B$ induced by $rank_B$ on P_B to the combined preference order $\leq_{AB}$ induced by f on $P_A \cap P_B$ defined as $f_{MMR}(c) := \min\{rank_A(c), rank_B(c)\}$ for $c \in P_A \cap P_B$. I.e., if $x, y \in P_A \cap P_B$, then $x \leq_{AB} y :\Leftrightarrow \min\{rank_A(x), rank_B(x)\} \leq \min\{rank_A(y), rank_B(y)\}$.*

Remark 3.4 *The functions $f_{SoR}(c) := rank_A(c) + rank_B(c)$ and $f(c)_{MMR} := \min\{rank_A(c), rank_B(c)\}$ are not bijective. Consequently, the induced combined preference orders are total* pre-orders *but not total* orders.

3.2 Protocols for Privacy-Preserving Policy Reconciliation

Drawing on the previous definitions, we now introduce the protocols for policy reconciliation and define when we call these protocols privacy-preserving.

Policy reconciliation in general has the objective of two parties determining those policy rules they have in common. In the course of conventional reconciliation protocols, at least one of the two parties learns more of the other party's policies than just what the two parties have in common. As discussed previously, the latter constitutes a major problem in situations where privacy is of primary concern. The objective of privacy-preserving policy reconciliation is not to reveal any more information about a party's policy rules to the other party than the common policy rules.

Let us now assume that two parties have total preference orders on their policies and have agreed upon a preference order composition scheme. Then preference-maximizing policy reconciliation for a composition scheme has the objective of determining a policy rule that maximizes the combined preference order. Again, this problem could be solved by one party providing its policy as well as its total preference order to the other party. However, in this case, the other party could chose a policy rule that only maximizes its own preferences. The objective of privacy-preserving common policy reconciliation is to find a policy rule that maximizes the combined preference order without revealing any information about a party's policy rules and preferences to the other party than the maximizing policy rule.

In both cases, with and without considering preferences, a trusted third party could be used to solve the privacy problem. However, in practice such a trusted third party barely ever exists. Thus, the goal in secure multi-party computation in general, and here in particular, is to define protocols without a trusted third party that provide the same view to each of the parties as if a trusted third party was present.

As we will see in the course of this paper, it is comparatively easy to implement privacy-preserving policy reconciliation schemes that do not take preferences into account. We illustrate this with the help of the private set intersection techniques introduced in [12]. Kursawe *et al.* reach the same goal with the help of a boolean circuit and using efficient two-party computation techniques based on threshold homomorphic cryptosystems [21]. However, implementing privacy-preserving preference-maximizing policy reconciliation seems not quite that straight forward.[1] Here, we introduce the slightly weaker notion of a privacy-preserving *multi-round* preference-maximizing policy reconciliation scheme. In addition to a preference-maximizing policy rule, such a scheme reveals the rank the other party has assigned to the resulting policy rule. Note that the additional leakage of the rank of the result in the other parties preference order seems acceptable in many situations as at the time of leakage the agreement has already been reached such that the other party cannot use the additional information to influence the result anymore.

Keeping these considerations in mind, we define three different protocols for policy reconciliation as follows:

Protocol 1 *A* privacy-preserving common policy *scheme (PCP) is a two-party protocol between parties A and B. Their respective policies P_A and P_B are drawn from the same domain of policy rules (see Remark 3.1). Upon completion of the protocol, A and B learn nothing else about each other's private policies but what can be deduced from the policy rules they have in common.*

In our example, this means that the parties determine all dates and times that would work for an interview for both HR and the applicant.

Protocol 2 *A* privacy-preserving common policy cardinality *scheme (PC^2) is a two-party protocol between parties A and B. Their respective P_A and P_B are drawn from the same domain of policy rules (see Remark 3.1). Upon completion of the protocol, A and B learn nothing else about each other's private policies but what can be deduced from the number of policy rules they have in common.*

A PC^2 thus allows two parties A and B to determine how many policy rules they have in common *without* revealing these policy rules to each other. A PC^2 protocol could, for example, be used by A and B to first determine the number of policies they have in common *before* reconciling their policies through a PCP protocol.

In our previous example, this would translate into the parties determining how many dates and times would work for both of them.

Protocol 3 *A* privacy-preserving preference-maximizing multi-round policy reconciliation *scheme for a preference order composition scheme $\mathcal{C}$ ($3PR^{\mathcal{C}}$) is a multi-round two-party protocol between parties A and B. Their respective policies P_A and P_B are drawn from the same domain of policy rules (see Remark 3.1).*

Upon completion of the protocol, A and B learn nothing about each other's policies but what can be deduced from one common policy rule max that maximizes the combined preference order $\leq_{AB}$ (of $\leq_A$ and $\leq_B$ under $\mathcal{C}$) and its respective rank under $\leq_{AB}$.

Note that in a $3\mathrm{PR}^{\mathcal{C}}$ scheme, the two parties should not learn anything about any other policy rules of the respective other party and nothing about the other party's preferences other than what can be deduced from the run or the output of the protocol. Furthermore, the scheme must enforce the output to maximize the combined preference order for the composition scheme in question. This guarantees that the two parties can reconcile their preferences in a fair way, i.e., according to an agreed upon preference composition scheme. Consequently, none of the parties has the power to set its preferences over the other party's preferences. The different preference composition schemes can hereby be used to express different notions of fairness in the reconciliation process.

In our example, the use of a $3\mathrm{PR}^{\mathcal{C}}$ scheme ensures that the interview will take place at a time and date that optimizes, i.e., best meets the preferences of both the HR department and the applicant.

[1]Note that in [21] the problem of taking preferences into account is recognized but no solution for this case is proposed.

3.3 Adversary Model and Privacy Requirements

We now describe the adversary models used in the remainder of this paper. Furthermore, we define the security and privacy requirements for PCP, PC^2, and 3PR$^{\mathcal{C}}$ protocols.

Semi-Honest Model. In the semi-honest model, all parties act honestly, but may be curious. That is, all parties act according to their prescribed actions in the protocol[2]. Yet, the participants may try to learn as much as possible in the process of the protocol and may perform any additional arbitrary polynomial-time computations apart from the prescribed protocol actions [14]. In particular, any party may store and use any intermediate result of the protocol. Intuitively, a two-party protocol is said to be *privacy-preserving* in the *semi-honest model*, if no party gains information about the other party's private input other than what can be deduced from the desired output of the protocol and the party's own private input. In other words, for each party there exists a simulator that produces an output distribution which is computationally indistinguishable from the other party's view executing the real protocol [13].[3]Note that while for a PCP and a PC^2 scheme the private inputs are the policies P_A and P_B only, the private input in the 3PR$^{\mathcal{C}}$ additionally includes the preference orders of A and B.

Malicious Model. In the malicious model, each party may behave arbitrarily. As detailed in [13], one cannot prevent a party from (1) refusing to participate in the protocol; (2) substituting its private input at the beginning of the protocol; or (3) aborting the protocol before completion. Intuitively, a two-party protocol is said to be *privacy-preserving* in the *malicious model*, if apart from the unavoidable deviations no other deviation of one malicious party leads to information leakage about the other (honest) party's private input, other than what can be deduced from the output of the protocol.[4] Similarly, we say that a 3PR$^{\mathcal{C}}$ scheme is *preference-maximizing in the malicious model*, if apart from the unavoidable deviations no other deviation of one malicious party can lead to an output the malicious party prefers over the one that maximizes the combined preference order.

4 Privacy-Preserving Reconciliation with Preferences

The main contribution of this paper is the introduction of a new 3PR$^{\mathcal{C}}$ scheme for the *sum of ranks* composition scheme and a new 3PR$^{\mathcal{C}}$ scheme for the *maximized minimum of ranks* composition scheme. These protocols consist of several rounds. Each round uses a privacy-preserving set intersection protocol (PPSI) as building block. In each round, the two parties commit to those policy rules that lead to the same rank in their combined preference order. As soon as a match is found, the protocol terminates. The order in which the parties commit to their policy rules thus guarantees the optimization of the combined preference order.

We recall that the objective of the sum of ranks composition scheme is for the two parties A and B to determine one common policy rule *max* in a privacy-preserving manner that maximizes the combined preference order on $P_A \cap P_B$ defined by $x \leq_{AB} y \Leftrightarrow \mathrm{rank}_A(x) + \mathrm{rank}_B(x) \leq \mathrm{rank}_A(y) + \mathrm{rank}_B(y)$. Using a PPSI as a building block, the protocol can generically be implemented as illustrated in Figure 2 on an example where both parties A and B have four policies each, sorted in decreasing order of their individual preferences. In order to determine a policy that maximizes the preferences of both parties A and B, the protocol progresses in a number of rounds. Each round consists of a number of PPSIs (each indicated by a line in Figure 2). For example, in Round 1, party A carries out one PPSI to determine whether there is a match between its top policy and that of party B. If so, the protocol terminates as it found a common policy rule that maximizes the combined preference order. Otherwise, the protocol moves on to the next

[2]Note that the parties may keep a record of all their intermediate computations.

[3]As generally is the case with secure multi-party computation, we assume the channel over which messages are exchanged to be an authenticated channel.

[4]As a consequence of the three above mentioned unpreventable actions a malicious party can take, a PCP scheme, for example, without any further external protection mechanism cannot prevent a malicious party from extracting the other parties policy: the malicious party simply substitutes its own input with a policy containing all possible policy rules and consequently learns the honest parties policy as output of the PCP scheme. Similarly, a PC^2 scheme cannot protect the cardinality of an honest party's private input policy against a malicious party.

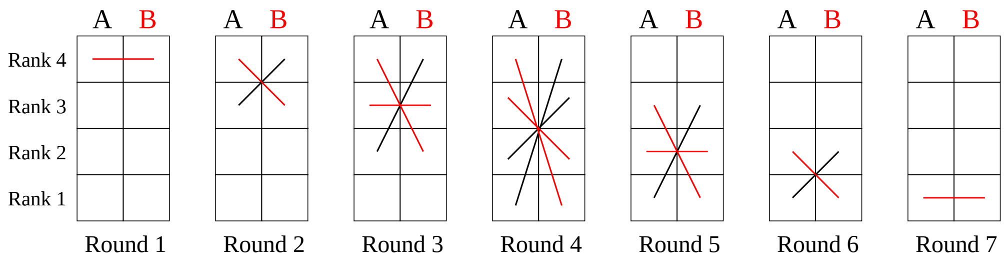

Figure 2: Sum-of-ranks: order in which A and B compare policy rules in case of 4 rules

round. All PPSIs carried out in one round result in the same sum of ranks. In particular, the sum of ranks in Round 1 is eight (in the example, or $2k$ in case of both parties holding k policy rules). In each round, the sum decreases by one, i.e., Round i corresponds to a sum of ranks of $2k - i + 1$ thus guaranteeing that the protocol will in fact always maximize the sum of ranks. In the worst case, the protocol will carry out $2k - 1$ rounds of PPSIs.

For the maximized minimum of ranks composition scheme, the order of executing PPSIs to obtain the $3\text{PR}^{\mathcal{C}}$ is slightly different as the two parties A and B strive to determine a common policy rule *max* in a privacy-preserving manner such that it maximizes the combined preference order on $P_A \cap P_B$ defined by $x \leq_{AB} y \Leftrightarrow \min(\text{rank}_A(x), \text{rank}_B(x)) \leq \min(\text{rank}_A(y), \text{rank}_B(y))$. Figure 3 illustrates the generic implementation of the multi-round protocol using some PPSI as a building block. Unlike before, the focus in each

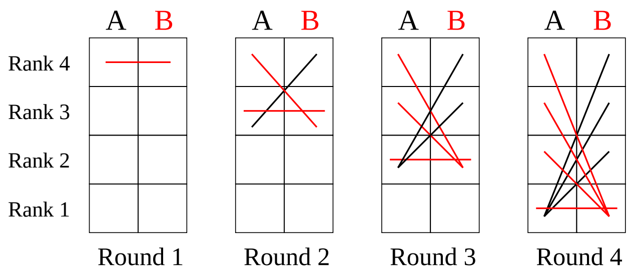

Figure 3: Maximized-minimum-of-ranks: order in which A and B compare policy rules in case of 4 rules

round is now on the policy rule with the smaller preference as the goal is to maximize that. Consequently, the respective $3\text{PR}^{\mathcal{C}}$ requires fewer rounds and executes the PPSI in a different fashion. In fact, the number of protocol rounds never exceeds the number of policy rules the parties hold. As before, both parties A and B each carry out a certain number of PPSIs in each round of the $3\text{PR}^{\mathcal{C}}$. For example, in Round 3 in Figure 3, party B will carry out up to two PPSIs, that is, the PPSI between its top policy and party A's third policy and possibly a PPSI with its second policy and A's third policy. Obviously, the minimum preference of all policies considered in this round is two. If no intersection is found, the protocol will move on to the next round which is characterized by the fact that the minimum of the preferences of the policy rules considered is decreased by one. This procedure ensures that the protocol will determine a common policy rule that maximizes the combined preference order.

Assuming that the PPSI is privacy-preserving in the semi-honest model automatically implies that the new $3\text{PR}^{\mathcal{C}}$ protocols are privacy-preserving in the semi-honest model—irrespective of the PPSI and combined preference order used. This is due to the fact that in the semi-honest model both parties will follow the protocol. Consequently, if the PPSI used is privacy-preserving in the semi-honest model, then a party will not learn anything about any policies unless a common policy is determined through the respective PPSI. Obviously, an analogous statement for the malicious model does not hold true as malicious behavior in the $3\text{PR}^{\mathcal{C}}$ protocols may come into play both within a specific round (including multiple PPSIs) as well as in between different rounds.

While any PPSI may be used as building block for our $3\mathrm{PR}^{\mathcal{C}}$ protocols, in the following we will focus our attention on the PPSI introduced in [12]. In particular, we will detail and optimize our protocols using the PPSI constructions by Freedman *et al.* which is based on private polynomial evaluation. While any other PPSI can be employed in a generic straight-forward manner as explained above, we expect that the use of some specific PPSIs (see Section 2) can lead to protocols that are more efficient than others.

4.1 Privacy-Preserving Tools by Freedman *et al.*

Privacy-Preserving Set Intersection The PPSI of [12] is a two-party protocol between a chooser C and a sender S. At the beginning of the protocol, both parties have private data sets (Z_C and Z_S) drawn from some common domain. At the conclusion of the protocol, the chooser learns the intersection $Z_C \cap Z_S$, but nothing about any other data in Z_S. The sender learns nothing about any data in Z_C. That is, Freedman *et al.* prove that their protocol is privacy-preserving in the semi-honest model. For data sets of size $O(k)$, the protocol results in a communication overhead of $O(k)$ and computational overhead of $O(k \ln \ln k)$.

The Freedman protocol is based on a semantically secure homomorphic encryption scheme: If E is a public encryption function of a homomorphic encryption scheme, then, given the ciphertexts $c_1 = E(m_1)$ and $c_2 = E(m_2)$, the ciphertext $c^* = E(m_1 + m_2)$ can be computed efficiently without knowledge of the private key. Similarly, given $c = E(m)$ and some r from the group of plaintexts, then $c^* = E(rm)$ can be computed efficiently without knowledge of the private key. A public encryption function E is semantically secure if it is computationally infeasible for an attacker to derive significant information about a plaintext given only its ciphertext and the public encryption key. An example of a semantically secure homomorphic encryption scheme is Paillier's cryptosystem [25]. The homomorphic property of an encryption function E implies that anyone in possession of the encrypted coefficients of a polynomial $f(X)$ can compute a valid encryption of $f(y)$ for any y from the group of plaintexts without the knowledge of the private key or the coefficients. In particular, for any known plaintexts y_1, y_2 and any known constant r, a valid encryption $E(rf(y_1) + y_2)$ can be computed without the knowledge of the private key or the coefficients of $f(X)$. The property of the encryption scheme to be semantically secure is crucial for the protocols to be privacy-preserving. That is, if the encryption scheme was not semantically secure, then it would be computationally feasible to violate privacy, i.e., determine the plaintexts—which, in the context of Freedman's work, are the elements of the private data sets of the parties—without knowledge of the private key. This would be possible by simply encrypting all candidate plaintexts using the party's respective public key and checking the results against the publicly known ciphertexts.

Private Cardinality Matching [12] also presents a protocol to compute the cardinality of the intersection of two data sets that is privacy-preserving in the semi-honest model. In fact, this protocol is a variant of the set intersection protocol. The chooser learns nothing about the data sets of the server, except for the cardinality of the intersection of the chooser's and the server's data sets.[5] For data sets of size $O(k)$, the protocols results in a communication overhead of $O(k)$ and computational overhead of $O(k \ln \ln k)$.

As illustrated in Appendix A, building on the work of Freedman *et al.* it is possible to build protocols for PC^2 and PCP in a straight-forward manner. However, implementing a $3\mathrm{PR}^{\mathcal{C}}$ schemes requires some sophisticated protocol design.

In the following, we assume that prior to the execution of any of the protocols introduced in this section the two parties A and B agree upon a semantically secure homomorphic encryption scheme. The parties choose their public and private key pairs for the encryption scheme and exchange their public keys. We denote the public encryption functions of A and B with E_A and E_B and their private decryption functions with D_A and D_B. Parties A and B have policies $P_A = (a_1, \ldots, a_k)$ and $P_B = (b_1, \ldots, b_l)$ consisting of policy rules drawn from the same domain $\{0,1\}^n$ (Remark 3.1). In addition, in both $3\mathrm{PR}^{\mathcal{C}}$ protocols we assume

[5]It is important to note that in the Freedman *et al.* schemes the server learns the size of Z_C. Other schemes avoid this leakage at the cost of efficiency (*e.g.,* [1] in the case of private cardinality matching and [31] for private set intersection).

that P_A and P_B have the same number of policy rules, that is $k = l$. Furthermore, the policy rules in P_A and P_B are enumerated in decreasing order of $\leq_A$ and $\leq_B$ such that a_1 is A's favorite policy rule and a_k is A's least favorite policy rule.

4.2 3PR$^{\mathcal{C}}$ Protocol for the Sum of Ranks Composition Scheme

In this section we detail the 3PR$^{\mathcal{C}}$ protocol for the sum of ranks (SoR) composition scheme. The basic idea of the multi-round protocol using any private set intersection protocol was illustrated in Figure 2. In the following, we formally describe the protocol messages for the case that the private set intersection protocol introduced by Freedman *et al.* is used.

3PR^{SoR} Protocol: For each policy rule $a_i \in P_A$ $(i = 1, \ldots, k)$ we define $f_A^{(i)} := (X - a_i)$. The polynomial with coefficients encrypted under E_A is denoted by $E_A^{(i)}$. Similarly, for each $b_i \in P_B$ $(i = 1, \ldots, k)$ we define $f_B^{(i)}$ and $E_B^{(i)}$.

Commitment Phase:

Round 0: B sends $E_B^{(1)}$ to A.

Round 1: A chooses a random $r_{A,1}^{(1)}$, computes $c_{A,1}^{(1)} = E_B\left(r_{A,1}^{(1)} \cdot f_B^{(1)}(a_1) + a_1 \right)$, and sends it together with $E_A^{(1)}$ to B. B decrypts $c_{A,1}^{(1)}$ under D_B and compares the result to b_1. If $b_1 = D_B(c_{A,1}^{(1)})$, then $b_1 = a_1$, that is, B has found $max := b_1 = a_1$ and continues with the *Match Confirmation Phase*. Otherwise, B sends $E_B^{(2)}$ to A and A continues with *Round 2* of the *Commitment Phase*.

Round $2 \leq i \leq k$: For $j = 1, \ldots, \lceil i/2 \rceil$, A chooses random $r_{A,j}^{(i-j+1)}$ and computes the ciphertexts

$$c_{A,j}^{(i-j+1)} := E_B\left(r_{A,j}^{(i-j+1)} \cdot f_B^{(i-j+1)}(a_j) + a_j \right)$$

and sends them as well as $E_A^{(i)}$ to B. For $j = 1, \ldots, \lceil i/2 \rceil$ B decrypts $c_{A,j}^{(i-j+1)}$ with D_B and checks whether the decrypted value equals b_{i-j+1}. If for some $j =: m$ the ciphertext $c_{A,m}^{(i-m+1)}$ decrypts to b_{i-m+1}, B has found $max := b_{i-m+1} = a_m$ and continues with the *Match Confirmation Phase*. Otherwise, for $1 \leq j \leq \lfloor i/2 \rfloor$, B chooses random $r_{B,j}^{(i-j+1)}$ and computes the ciphertexts

$$c_{B,j}^{(i-j+1)} := E_A\left(r_{B,j}^{(i-j+1)} \cdot f_A^{(i-j+1)}(b_j) + b_j \right)$$

and sends them to A. For $2 \leq i < k$ B additionally sends $E_B^{(i+1)}$ to A. For $1 \leq j \leq \lfloor i/2 \rfloor$, A decrypts $c_{B,j}^{(i-j+1)}$ with D_A and checks whether the decrypted value equals a_{i-j+1}. If $c_{B,m}^{(i-m+1)}$ decrypts to a_{i-m+1}, A has found $max := a_{i-m+1} = b_m$ and continues with the *Match Confirmation Phase*. Otherwise, A continues with *Round* $i+1$ of the *Commitment Phase*.

Round $k < i < 2k - 1$: For $j = 1, \ldots, k - \lfloor i/2 \rfloor$, A chooses random $r_{A,i-k+j}^{(k-j+1)}$ and computes the ciphertexts

$$c_{A,i-k+j}^{(k-j+1)} := E_B\left(r_{A,i-k+j}^{(k-j+1)} \cdot f_B^{(k-j+1)}(a_{i-k+j}) + a_{i-k+j} \right)$$

and sends them to B. For $j = 1, \ldots, k - \lfloor i/2 \rfloor$ B decrypts $c_{A,i-k+j}^{(k-j+1)}$ with D_B and checks whether the decrypted value equals b_{k-j+1}. If for some $j =: m$ the ciphertext $c_{A,i-k+m}^{(k-m+1)}$ decrypts to b_{k-m+1}, B has found $max := b_{k-m+1} = a_{i-k+m}$ and continues with the *Match Confirmation Phase*. Otherwise, for $1 \leq j \leq k - \lceil i/2 \rceil$, B chooses random $r_{B,i-k+j}^{(k-j+1)}$ and computes the ciphertexts

$$c_{B,i-k+j}^{(k-j+1)} := E_A\left(r_{B,i-k+j}^{(k-j+1)} \cdot f_A^{(k-j+1)}(b_{i-k+j}) + b_{i-k+j} \right)$$

and sends them to A. For $1 \leq j \leq k - \lceil i/2 \rceil$, A decrypts $c_{B,i-k+j}^{(k-j+1)}$ with D_A and checks whether the decrypted value equals a_{k-j+1}. If $c_{B,i-k+m}^{(k-m+1)}$ decrypts to a_{k-m+1}, A has found $max := a_{k-m+1} = b_{i-k+m}$ and continues with the *Match Confirmation Phase*. Otherwise, A continues with *Round* $i+1$ of the *Commitment Phase*.

Round $i = 2k-1$: A chooses random $r_{A,k}^{(k)}$ and computes the ciphertext

$$c_{A,k}^{(k)} := E_B\left(r_{A,k}^{(k)} \cdot f_B^{(k)}(a_k) + a_k \right)$$

and sends it to B. B decrypts $c_{A,k}^{(k)}$ with D_B and checks whether the decrypted value equals b_k. If $c_{A,k}^{(k)}$ decrypts to b_k, B has found $max := b_k = a_k$ and continues with the *Match Confirmation Phase*. Otherwise, B sends a message to A indicating that no match was found and aborts the protocol.

Match Confirmation Phase: If B found the match $b_1 = a_1 = max$ in the first round of the *Commitment Phase*, then B computes $c_{B,1}^{(1)}$ and sends it to A. A decrypts the received ciphertext with D_A to a_1. Thus A and B both know that $max = a_1 = b_1$.

If B found $max = b_{i-m+1} = a_m$ in *Round* $2 \leq i \leq k$ of the *Commitment Phase*, then B computes $c_{B,i-m+1}^{(m)}$ and sends it to A. A decrypts the received value under D_A to a_m. Thus, A and B both know that $a_m = b_{i-m+1} = max$.

If A found $max = a_{i-m+1} = b_m$ in *Round* $2 \leq i \leq k$ of the *Commitment Phase*, A computes $c_{A,i-m+1}^{(m)}$ and sends it to B. B decrypts $c_{A,i-m+1}^{(m)}$ with D_B and checks that $D_B(c_{A,i-m+1}^{(m)})$ decrypts to a value $b_m \in \{b_1, \ldots, b_{\lfloor i/2 \rfloor}\}$. Thus, both parties A and B know that $a_{i-m+1} = b_m = max$.

If B found $max = b_{k-m+1} = a_{i-k+m}$ in *Round* $k < i < 2k-1$ of the *Commitment Phase*, then B computes $c_{B,k-m+1}^{(i-k+m)}$ and sends it to A. A decrypts the received value under D_A to a_{i-k+m}. Thus, A and B both know that $a_{i-k+m} = b_{k-m+1} = max$.

If A found $max = a_{k-m+1} = b_{i-k+m}$ in *Round* $k < i < 2k-1$ of the *Commitment Phase*, A computes $c_{A,k-m+1}^{(i-k+m)}$ and sends it to B. B decrypts $c_{A,k-m+1}^{(i-k+m)}$ with D_B and checks that $D_B(c_{A,k-m+1}^{(i-k+m)})$ decrypts to a value $b_{i-k+m} \in \{b_{i-k+1}, \ldots, b_{\lfloor i/2 \rfloor}\}$. Thus, both parties A and B know that $a_{k-m+1} = b_{i-k+m} = max$.

If B found $max = b_k = a_k$ in *Round* $i = 2k-1$ of the *Commitment Phase*, then B computes $c_{B,k}^{(k)}$ and sends it to A. A decrypts the received value with D_A to a_k. Thus A and B both know that $a_k = b_k = max$.

Remark 4.1 *Unlike in PCP and PC^2 we have assumed for the 3PR^{SoR} protocol that A and B have the same number k of policy rules and each rank is assigned to exactly one policy rule. However, this does not limit A and B to define less than k valid policy rules, as A and B can simply augment their valid policies to contain k rules by appending special rules agreed upon to be dummy rules. E.g., policy rules containing only zeros for every attribute could be interpreted as such dummy rules. A matching zero policy as result of 3PR^{SoR} then indicates that A and B do not share a single valid policy rule.*

Intuitively, the protocol works as follows: The homomorphic encryption function allows a party A to encrypt a polynomial corresponding to one of its own policy rules in a way that the encrypted polynomial does not reveal any information to party B. In addition, if B evaluates the encrypted polynomial on one of its own policy rules, then its policy rule is blinded and encrypted. From this blinded ciphertext A learns nothing about B's policy rule, unless the policy rule A used for the creation of the polynomial and the policy rule on which B evaluated the polynomial coincide. This is because, if and only if the policy rules coincide, the blinded ciphertext decrypts to A's policy rule. A and B can thus determine a common policy rule without revealing rules to each other that they do not have in common. Our protocols make use of this basic procedure in multiple rounds.

In each round, parties A and B exchange polynomials and blinded ciphertexts corresponding to previously received polynomials. They each decrypt the blinded ciphertext and compare the result to the policy rules corresponding to the previously sent polynomials. The order in which the information is exchanged

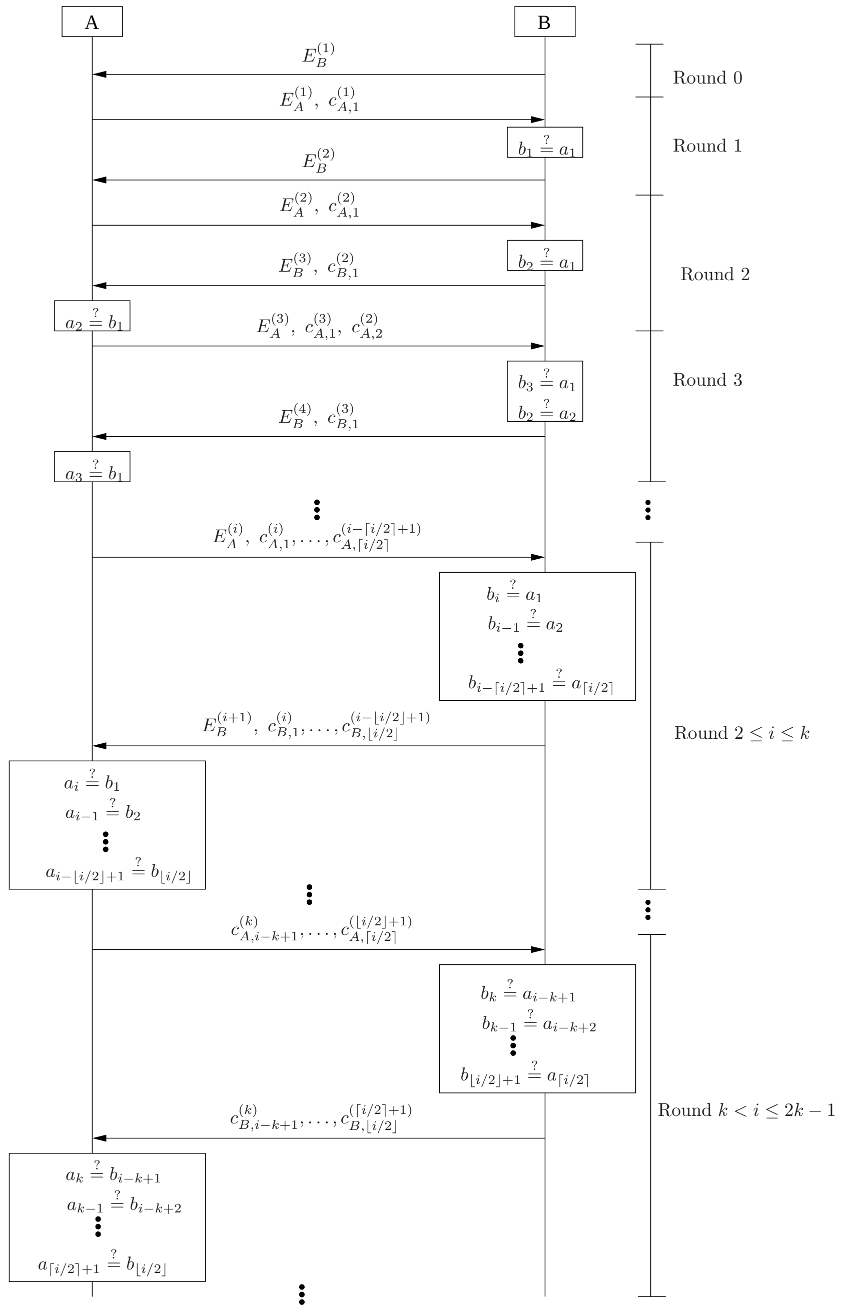
A
B
Round 0
Round 1
Round 2
Round 3
Round $2 \leq i \leq k$
Round $k < i \leq 2k-1$
$E_B^{(1)}$
$E_A^{(1)},\ c_{A,1}^{(1)}$
$b_1 \overset{?}{=} a_1$
$E_B^{(2)}$
$E_A^{(2)},\ c_{A,1}^{(2)}$
$b_2 \overset{?}{=} a_1$
$E_B^{(3)},\ c_{B,1}^{(2)}$
$a_2 \overset{?}{=} b_1$
$E_A^{(3)},\ c_{A,1}^{(3)},\ c_{A,2}^{(2)}$
$b_3 \overset{?}{=} a_1$
$b_2 \overset{?}{=} a_2$
$E_B^{(4)},\ c_{B,1}^{(3)}$
$a_3 \overset{?}{=} b_1$
$E_A^{(i)},\ c_{A,1}^{(i)},\ldots,c_{A,\lceil i/2\rceil}^{(i-\lceil i/2\rceil+1)}$
$b_i \overset{?}{=} a_1$
$b_{i-1} \overset{?}{=} a_2$
$b_{i-\lceil i/2\rceil+1} \overset{?}{=} a_{\lceil i/2\rceil}$
$E_B^{(i+1)},\ c_{B,1}^{(i)},\ldots,c_{B,\lfloor i/2\rfloor}^{(i-\lfloor i/2\rfloor+1)}$
$a_i \overset{?}{=} b_1$
$a_{i-1} \overset{?}{=} b_2$
$a_{i-\lfloor i/2\rfloor+1} \overset{?}{=} b_{\lfloor i/2\rfloor}$
$c_{A,i-k+1}^{(k)},\ldots,c_{A,\lceil i/2\rceil}^{(\lfloor i/2\rfloor+1)}$
$b_k \overset{?}{=} a_{i-k+1}$
$b_{k-1} \overset{?}{=} a_{i-k+2}$
$b_{\lfloor i/2\rfloor+1} \overset{?}{=} a_{\lceil i/2\rceil}$
$c_{B,i-k+1}^{(k)},\ldots,c_{B,\lfloor i/2\rfloor}^{(\lceil i/2\rceil+1)}$
$a_k \overset{?}{=} b_{i-k+1}$
$a_{k-1} \overset{?}{=} b_{i-k+2}$
$a_{\lceil i/2\rceil+1} \overset{?}{=} b_{\lfloor i/2\rfloor}$

Figure 4: Commitment Phase of the 3PR^{SoR} protocol. For example, in Round $2 \leq i \leq k$, A for $j = 1,\ldots,\lfloor i/2\rfloor$ compares a_{i-j+1} with b_j and B for $j = 1,\ldots,\lceil i/2\rceil$ compares b_{i-j+1} with a_j without obtaining knowledge of these values (unless they are equal), that is, in a privacy-preserving manner. Note that in each Round $1 \leq i \leq 2k-1$ all policy rules are compared that result in the same sum of ranks $i+1$.

guarantees that the protocol terminates when a common policy rule is found that maximizes the sum of ranks of the common policies of A and B. For example, in the first round of the protocol, A can check whether the two most preferred policy rules of both parties are the same ($a_1 \overset{?}{=} b_1$). If this is not the case, A

does not learn anything about b_1. As a general rule, in Round i of the protocol, A and B compare all policy rules that result in the same sum of ranks in the combined preference order of A and B.

Discussion:

For $P_A \cap P_B \neq \emptyset$, the *Commitment Phase* of the 3PR^{SoR} protocol always terminates with one of the parties finding a match. This is due to the fact that for each pair (l, s) $1 \leq l, s \leq k$, a_l is compared with b_s in *Round* $l + s - 1$. To be precise, for $l < s$, a_l is compared with b_s by A and for $s \leq l$, b_s is compared with a_l by B in Round $l + s - 1$. Let $max = a_l = b_s$ be a match found in *Round* $l + s - 1$. Then no match was found in any previous round of the *Commitment Phase*. We claim that then max maximizes the sum of the ranks of all elements in the intersection $P_A \cap P_B$. If this was not the case, then there would be a pair (a_r, b_v) with $a_r = b_v \in P_A \cap P_B$ and $r + v < s + l$. As a consequence, a_r would have been compared to b_v already in Round $r + v - 1$ of the *Commitment Phase*. This obviously contradicts the assumption.

In order to prove that the 3PR^{SoR} protocol is privacy-preserving in the semi-honest model (see Protocol 3), it is necessary to show that the information that either party can deduce from the 3PR^{SoR} protocol is equivalent to either party knowing nothing else but what can be de deduced from knowing the matching rule max (maximizing the sum of ranks composition scheme) and its rank $f_{SoR}(max)$ in the combined preference order $\leq_{AB}$. As discussed before, finding a match with the 3PR^{SoR} protocol implies that both parties know the policy rule max which maximizes the combined preference order as well as the round in which it was found. The latter corresponds to the rank of the matching rule under the sum of ranks composition scheme.[6] In turn, knowing max itself and its rank $f_{SoR}(max)$ allows party A (B) to determine $rank_B(max)$ ($rank_A(max)$). Furthermore, since max maximizes $f_{SoR}(x)$ with $x \in P_A \cap P_B$, it holds that $\nexists\ a \in P_A, b \in P_B$ such that $a = b$ and $f_{SoR}(a) = f_{SoR}(b) > f_{SoR}(max)$. Since $f_{SoR}(max)$ directly corresponds to the round in which the 3PR^{SoR} protocol would find the match, this implies that the operations in Rounds i with $i < f_{SoR}(max)$ in the 3PR^{SoR} protocol should yield no information other than that the respective policy rules do not match. Using Freedman's PPSIs in each round, the 3PR^{SoR} protocol achieves this by construction as Freedman's PPSI is privacy-preserving in the semi-honest model.

It is important to note that while deviations from the protocol always imply privacy violations, a malicious party will not necessarily profit from deviations in terms of maximizing its preferences. This is due to the fact that the combined preference order depends on the preference order of the honest party. In order to profit from deviations a malicious party would need to know the honest party's preference order at the time of initiation of reconciliation. As a consequence, in situations where A and B reconcile their policies only once and have to follow the result later on, deviations from the protocol are unattractive for either party.

We evaluate the worst case performance of the protocol by counting (1) the number of times A and B have to compare a received decrypted policy rule of the other party with one of their policy rules, (2) the number of decryptions A and B have to perform, (3) the number of encryptions of coefficients of polynomials, (4) the number of ciphertext A and B have to compute to commit to their policy rules, (5) the number of messages exchanged between A and B, as well as (6) the overall size of the payload of all messages exchanged, counted in the number of ciphertexts included in all messages.

We count the above numbers for three different relations between the overall number of n of policy rules and the number k of policy rules A and B choose from the overall set of policy rules.

In the first case, we assume that n equals k. In this case, A and B share all policy rules and differ only in their preferences. The worst case occurs if the policy rules of A and B are in exactly the opposite order. In this case, the match is found in Round k of the protocol. Table 1 shows the results.

For example, the number of comparisons is counted as follows: In the worst case, the match is found in Round k by one comparison and confirmed in the confirmation phase by yet another comparison. The preceding $k - 1$ rounds are unsuccessful. In the $i - th$ round ($i = 1, \ldots, k - 1$), i comparisons take place. Therefore the overall number of comparisons is $1 + 2 + \cdots + k - 1 + 2 = \sum_{r=1}^{k-1} r + 2 = \frac{k(k-1)}{2} + 2$.

[6]It is important to note that if a match is found by A (B), A (B) may learn more than one preference maximizing policy rule. This is due to the fact that more than one pair of policy rules may be evaluated by A (B) in each round. This asymmetry can be prevented by adding subrounds to the *Commitment Phase* such that each party commits to only one policy rule in each subround of each round in the *Commitment Phase*. This, however, would result in a higher communication overhead.

$k = n$	
# Comparisons	$\frac{k(k-1)}{2} + 2$
# Encryption of coefficients	$4k - 3$
# Decryptions	$\frac{k(k-1)}{2} + 2$
# Commitment of rules	$\frac{k^2+2}{2}$ (k even); $\frac{k^2+3}{2}$ (k odd)
# Messages	$2k + 1$
Size of payload in # ciphertexts	$k^2/2 + 4k - 2$

Table 1: Sum-of-ranks: Worst case performance for $n = k$, where n is the overall number of policy rules A and B choose from and k is the number of policy rules A and B choose.

In the second case, we assume that $k \leq n \leq 2k-1$. That is A and B differ in at least one policy rule and have at least one policy rule in common. In this case, the worst case occurs if A and B share only exactly one policy rule and this policy rules is the least preferred rule of both parties. In this case, the match is found in Round $2k-1$ of the protocol. The results are shown in Table 2.

$k \leq n \leq 2k-1$	
# Comparisons	$k^2 + 1$
# Encryption of coefficients	$4k$
# Decryptions	$k^2 + 1$
# Commitment of rules	$k^2 + 1$
# Messages	$4k - 1$
Size of payload in # ciphertexts	$k^2 + 4k + 1$

Table 2: Sum-of-ranks: Worst case performance for $k \leq n \leq 2k-1$

Finally, in the third case, we assume that $n > 2k-1$. In this case, the policies of A and B may be disjunct, which is also the worst case for this relation between n and k. A and B realize that they do not have a policy rule in common when no match is found after $2k-1$ rounds. Table 3 shows the results for this case.

$n > 2k-1$	
# Comparisons	k^2
# Encryption of coefficients	$4k$
# Decryptions	k^2
# Commitment of rules	k^2
# Messages	$4k - 1$
Size of payload in # ciphertexts	$k^2 + 4k$

Table 3: Sum-of-ranks: Worst case performance for $n > 2k-1$

Overall we can conclude that the computational overhead as well as the communication overhead of the protocol are bounded by $O(k^2)$.

4.3 3PR$^{\mathcal{C}}$ Protocol for the Maximized Minimum of Ranks Composition Scheme

In this section we detail the 3PR$^{\mathcal{C}}$ protocol for the maximized minimum of ranks (MMR) composition scheme. The main difference to the 3PR$^{\mathcal{C}}$ protocol for the sum of ranks composition scheme is the order in which private set intersections are executed (see Figure 3). In the following, we formally describe the protocol messages and processing for the case that the private set intersection protocol introduced by Freedman *et al.* is used.

3PR^{MMR} Protocol: Within this protocol description, we reuse the definitions of $f_A^{(i)}, f_B^{(i)}, E_A^{(i)}, E_B^{(i)}, c_{A,j}^{(i)}$, and $c_{B,j}^{(i)}$ introduced in the previous section.

Commitment Phase:

Round 0: B sends $E_B^{(1)}$ to A.

Round 1: A sends $E_A^{(1)}$ and $c_{A,1}^{(1)}$ to B. B decrypts $c_{A,1}^{(1)}$ under D_B and compares the result to b_1. If $b_1 = D_B(c_{A,1}^{(1)})$, then $b_1 = a_1$, that is, B has found $max := a_1 = b_1$ and continues with the *Match Confirmation Phase*. Otherwise B sends $E_B^{(2)}$ to A and A continues with *Round 2* of the *Commitment Phase*.

Round $2 \leq i \leq k$: For $j = 1, \ldots, i$, A computes the ciphertexts $c_{A,j}^{(i)}$ and sends them as well as $E_A^{(i)}$ to B. B decrypts $c_{A,1}^{(i)}, \ldots, c_{A,i}^{(i)}$ with D_B and checks whether any decrypted value equals b_i. If for some $j =: m$ the ciphertext $c_{A,m}^{(i)}$ decrypts to b_i, B has found $max := b_i = a_m$ and continues with the *Match Confirmation Phase*. Otherwise, B computes the ciphertexts $c_{B,j}^{(i)}$ for $1 \leq j \leq i-1$ and sends them to A. For $1 \leq i < k$ B additionally sends $E_B^{(i+1)}$ to A. A decrypts $c_{B,1}^{(i)}, \ldots, c_{B,i-1}^{(i)}$ with D_A and checks whether any decrypted value equals a_i. If for some $j =: m$ the ciphertext $c_{B,m}^{(i)}$ decrypts to a_i, A has found $max := a_i = b_l$ and continues with the *Match Confirmation Phase*. Otherwise, if $i + 1 \leq k$, A continues with *Round* $i + 1$ of the *Commitment Phase*. If $i + 1 > k$, the protocol terminates without a match being found.

Match Confirmation Phase: If it was A who found *max* in *Round* i of the *Commitment Phase*, A computes $c_{A,i}^{(m)}$ and sends it to B. B decrypts $c_{A,i}^{(m)}$ with D_B and checks that $D_B(c_{A,i}^{(m)})$ decrypts to a value $b_m \in \{b_1, \ldots, b_i\}$. Thus, both parties A and B know that $a_i = b_m = max$.

If it was B who found *max* in *Round* i of the *Commitment Phase*, then B computes $c_{B,i}^{(m)}$ and sends it to A. A decrypts the received value under D_A to a_m. Thus, A and B both know that $a_m = b_i = max$.

Remark 4.2 *Unlike in the 3PR^{SoR} protocol for, the 3PR^{MMR} protocol allows the same rank to be assigned to multiple policy rules assuming that A and B still have the same number k of policy rules.*

Intuitively speaking, the difference between this and the previously described protocol is that the order in which A and B exchange information and compare policies is changed. As a general rule, in Round i of the protocol A and B compare all policy rules that lead to the same maximum of minimum of ranks.

Discussion:

For $P_A \cap P_B \neq \emptyset$, the *Commitment Phase* of the 3PR^{MMR} protocol always terminates with one of the parties finding a match. This is due to the fact that for each $1 \leq i \leq k$, a_i is compared with $b_1, \ldots, b_{i-1}$ by A in *Round* i. Similarly, for each $1 \leq i \leq k$, b_i is compared with $a_1, \ldots a_i$ by B.

Let *max* be the first match found in Round i of the *Commitment Phase* and let without loss of generality $\max(\min(\mathrm{rank}_A(max), \mathrm{rank}_B(max))) = \mathrm{rank}_A(max)$. Then, *max* maximizes the minimum of the ranks of all policies in the intersection $P_A \cap P_B$. If this was not the case, then there was a policy rule $p \in P_A \cap P_B$ with $\min(\mathrm{rank}_A(p), \mathrm{rank}_B(p)) > \mathrm{rank}_A(max)$.

This implies that $\mathrm{rank}_A(p) > \mathrm{rank}_A(max)$ and $\mathrm{rank}_B(p) > \mathrm{rank}_A(max)$ and A and B would have committed to p in a round prior to Round i. As a consequence, p would have been found in a prior round. This obviously contradicts the assumption.

In order to prove that the 3PR^{MMR} protocol is privacy-preserving in the semi-honest model (see Protocol 3), it is necessary to show that the information that either party can deduce from the 3PR^{MMR} protocol

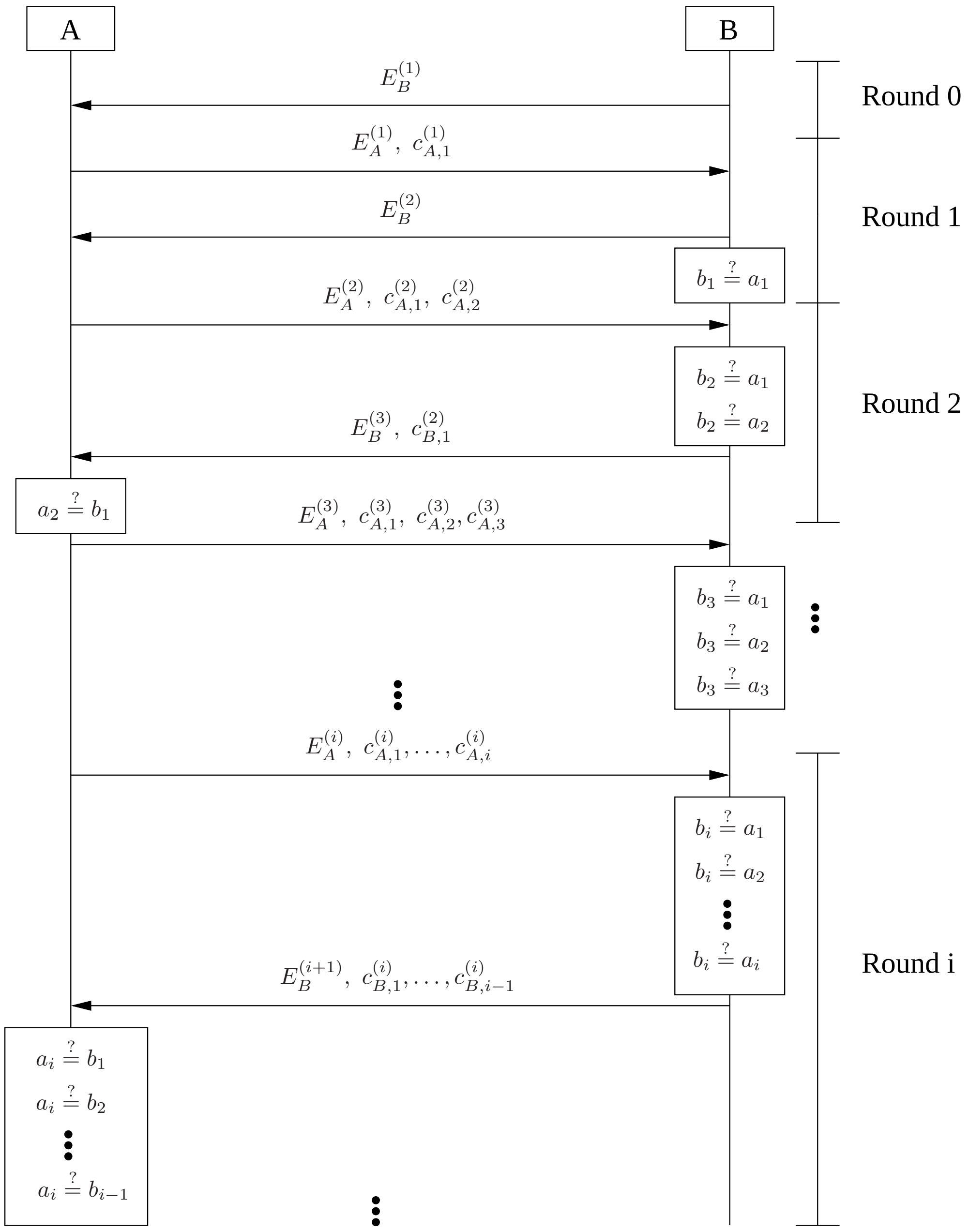

Figure 5: Commitment Phase of the 3PR^{MMR} protocol. In Round i, A compares a_i with $b_1, \ldots, b_{i-1}$ and B compares b_i with $a_1, \ldots, a_i$ without obtaining knowledge of these values, that is, in a privacy-preserving manner.

is equivalent to either party knowing nothing else but what can be de deduced from knowing the matching rule *max* (maximizing the minimum of ranks composition scheme) and its rank $f_{MMR}(max)$ in the combined preference order $\leq_{AB}$. As discussed before, finding a match with the 3PR^{MMR} protocol implies that both parties know the policy rule *max* which maximizes the combined preference order as well as the round in which it was found. The latter corresponds to the rank of the matching rule under the maximized minimum of ranks composition scheme.[7] In turn, knowing max itself and its rank $f_{MMR}(max)$ allows party A (B) to determine $rank_B(max)$ if $rank_A(max) > f_{MMR}(max)$ ($rank_A(max)$ if $rank_B(max) > f_{MMR}(max)$).

Furthermore, since max maximizes $f_{MMR}(x)$ with $x \in P_A \cap P_B$, it holds that $\nexists\ a \in P_A, b \in P_B$ such that $a = b$ and $f_{MMR}(a) = f_{MMR}(b) > f_{MMR}(max)$. Since $f_{MMR}(max)$ directly corresponds to the round in which the 3PR^{MMR} protocol would find the match, this implies that the operations in Rounds i with $i < f_{MMR}(max)$ in the 3PR^{MMR} protocol should yield no information other than that the respective policy rules do not match. Using Freedman's PPSIs in each round, the 3PR^{MMR} protocol achieves this by construction as Freedman's PPSI is privacy-preserving in the semi-honest model.

It is important to note that while deviations from the protocol always imply privacy violations, a malicious party will not necessarily profit from deviations in terms of maximizing its preferences. This is due to the fact that the combined preference order depends on the preference order of the honest party. In order to profit from deviations, a malicious party would need to know the honest party's preference order at the time of initiation of reconciliation.

As in the sum of ranks case, we evaluate the worst case performance of the protocol by counting (1) the number of comparisons, (2) decryptions, (3) encryptions of coefficients, (4) ciphertext commitments, (5) messages exchanged, as well as (6) the overall size of the exchanged messages counted in number of ciphertexts. Again, we consider three different relations between the overall number of n of policies and the number k of policy rules A and B choose from the overall set of policies.

In the first case, we assume that n equals k. In this case, A and B share all policy rules and differ only in their preferences. The worst case occurs if the policy rules of A and B are in exactly the opposite order. In this case, the match is found in Round $\lfloor k/2 \rfloor + 1$ of the protocol. Table 4 shows the results.

$k = n$	k odd	k even
# Comparisons	$\frac{k^2+1}{4} + 2$	$\frac{k(k+2)}{4} + 1$
# Encryption of coefficients	$2k + 2$	$2k + 4$
# Decryptions	$\frac{k^2+1}{4} + 2$	$\frac{k(k+2)}{4} + 1$
# Commitment of rules	$\frac{k-1}{2}(\frac{k-1}{2} + 1) + 2$	$\frac{k}{2}(\frac{k}{2} + 1) + 1$
# Messages	$k + 2$	$k + 3$
Size of payload in # ciphertexts	$2k + 4 + \frac{k-1}{2}(\frac{k-1}{2} + 1)$	$2k + 5 + \frac{k}{2}(\frac{k}{2} + 1)$

Table 4: Maximized-minimum-of-ranks: Worst case performance for $n = k$

In the second case, we assume that $k \leq n \leq 2k - 1$. In this case, A and B have at least one policy rule in common but may differ greatly in their preferences. In this case, the worst case occurs if A and B share only exactly one policy rule and this policy rules is the least preferred rule of both parties. In this case, the match is found in Round k of the protocol. The results are shown in Table 5.

Finally, in the third case, we assume that $n > 2k - 1$. In this case, the policies of A and B may be disjunct, which is also the worst case for this relation between n and k. A and B realize that they do not have a policy rule in common when no match is found after k rounds. Table 6 shows the results for this case.

Overall we can conclude that the computational overhead of the protocol is bounded by $O(k^2)$ and that the communicational overhead is bounded by $O(k^2)$.

5 Conclusions

In this paper we introduced a new paradigm—preference-based privacy-preserving policy reconciliation. We proposed a new general multi-round construction that makes use of a set intersection protocol and

[7] It is important to note that if a match is found by A (B), A (B) may learn more than one preference maximizing policy rule. This is due to the fact that more than one pair of policy rules may be evaluated by A (B) in each round. This asymmetry can be prevented by adding subrounds to the *Commitment Phase* such that each party commits to only one policy rule in each subround of each round in the *Commitment Phase*. This, however, would result in a higher communication overhead.

$k \leq n \leq 2k-1$	
# Comparisons	k^2
# Encryption of coefficients	$4k$
# Decryptions	k^2
# Commitment of rules	k^2
# Messages	$2k+1$
Size of payload in # ciphertexts	k^2+4k

Table 5: Maximized-minimum-of-ranks: Worst case performance for $k \leq n \leq 2k-1$

$n > 2k-1$	
# Comparisons	k^2
# Encryption of coefficients	$4k$
# Decryptions	k^2
# Commitment of rules	k^2
# Messages	$2k+1$
Size of payload in # ciphertexts	$4k+k^2$

Table 6: Maximized-minimum-of-ranks: Worst case performance for $n > 2k-1$

maximizes the combined preferences of two parties reconciling their policies. We focused on two approaches for combining individual preferences to combined preferences: computing the sum of ranks assigned by each party or using the minimum of ranks assigned by the two parties. We showed that the new general construction is privacy-preserving in the semi-honest model as long as the set intersection protocol is privacy-preserving in the semi-honest model. We furthermore detailed our new construction for the private set intersection protocol based on oblivious polynomial evaluation introduced by Freedman *et al.* for two different notions of fairness and analytically determined their performance.

A Appendix: Privacy-Preserving Policy Reconciliation in the Absence of Preferences

For reason of completeness, in this appendix, we provide the straight forward protocols for PC^2 and PCP introduced in Section 3.2. As a main component we use the protocols of Freedman *et al.* PC^2 and PCP are *straight-forward* adaptations of the protocols of Freedman *et al.* to the problem of privacy-preserving policy reconciliation in the absence of preferences. The description and discussion of these protocols are included here to provide a complete picture of how the privacy-preserving policy reconciliation problem can be tackled. As pointed out previously, the main contribution of this paper is, however, the design and analysis of protocols in the presence of preferences which is provided in Section 4.

We assume that prior to the execution of any of our protocols the two parties A and B agree upon a semantically secure homomorphic encryption scheme. The parties choose their public and private key pairs for the encryption scheme and exchange their public keys. We denote the public encryption functions of A and B with E_A and E_B and their private decryption functions with D_A and D_B. Parties A and B have policies $P_A = (a_1, \ldots, a_k)$ and $P_B = (b_1, \ldots, b_l)$ consisting of policy rules drawn from the same domain $\{0,1\}^n$ (see Remark 3.1).

A.1 Privacy-Preserving Common Policy Cardinality

The objective of the following protocol is for the two parties A and B to determine the number of policy rules they have in common in a privacy-preserving manner.

PC^2 Protocol: The parties A and B agree on some constant γ. A computes the polynomial

$$f_A(X) = (X - a_1)(X - a_2)\cdots(X - a_k) =: \sum_{i=0}^{k} \alpha_i X^i,$$

encrypts its coefficients α_i ($i = 0, \ldots, k$) under E_A, and sends them to B. Similarly, B computes the polynomial

$$f_B(X) = (X - b_1)(X - b_2)\cdots(X - b_l) =: \sum_{i=0}^{l} \beta_i X^i$$

and encrypts its coefficients β_i ($i = 0, \ldots, l$) under E_B. Then, for each $b_i \in P_B$ ($i = 1, \ldots, l$), B chooses a random r_i and computes $c_{B,i} = E_A(r_i \cdot f_A(b_i) + \gamma)$. B sends the ciphertexts $c_{B,i}$ and the encrypted coefficients $E_B(\beta_i)$ to A. A decrypts the received ciphertexts under D_A. For each $b_i \in P_A$, the ciphertext $c_{B,i}$ will decrypt to γ, while for any other b_i the decryption of the ciphertext $c_{B,i}$ under D_A will yield some arbitrary value. Consequently, the cardinality of the intersection $P_A \cap P_B$ corresponds to the number of ciphertexts that decrypt to γ. Then, for each $a_i \in P_A$ ($i = 1, \ldots, k$), A chooses a random r'_i and computes $c_{A,i} = E_B(r'_i \cdot f_B(a_i) + \gamma)$. A sends these ciphertexts to B. For each $a_i \in P_B$, the ciphertext $c_{A,i}$ will decrypt to γ, while for any $a_i \notin P_B$ the corresponding ciphertext will decrypt to some arbitrary value. Consequently, B also determines the cardinality of the intersection $P_A \cap P_B$ as the number of ciphertexts that decrypt to γ.

Discussion: If parties A and B follow the PC^2 protocol, both will learn the cardinality of the intersection of their policies. Following the detailed proof in [12], each direction of our PCP is privacy-preserving in the semi-honest model. This is due to the fact that ciphertexts sent from A (B) to B (A) that do not correspond to a common policy rule of both parties decrypt to some arbitrary value. As the encryption function is semantically secure, the ciphertexts $B(A)$ receives from A (B) are for two inputs of A (B) indistinguishable for B (A).

We evaluate the complexity of the PC^2 protocol by counting (1) the number of comparisons A and B have to conduct, (2) the number of polynomial coefficients they need to encrypt, (3) the number of decryptions, and (4) the number of commitments of rules exchanged. The result is shown in Table 7.

# Comparisons	$l + k$
# Encryption of Coefficients	$l + k + 2$
# Decryptions	$l + k$
# Commitment of rules	$l + k$
# Messages	2

Table 7: Performance of PC^2, where l and k are the number of policy rules of A and B.

A.2 Privacy-Preserving Common Policy

The objective of the following protocol is for the two parties A and B to determine the policy rules they have in common in a privacy-preserving manner.

PCP Protocol: A computes the polynomial

$$f_A(X) = (X - a_1)(X - a_2)\cdots(X - a_k) =: \sum_{i=0}^{k} \alpha_i X^i,$$

encrypts the coefficients α_i $(i = 0, \ldots, k)$ under E_A, and sends the encrypted coefficients to B. Similarly, B computes the polynomial

$$f_B(X) = (X - b_1)(X - b_2) \cdots (X - b_l) =: \sum_{i=0}^{l} \beta_i X^i$$

and encrypts the coefficients β_i $(i = 0, \ldots, l)$ under E_B. Then, for each $b_i \in P_B$ $(i = 1, \ldots, l)$, B chooses a random r_i and computes the ciphertexts $c_{B,i} = E_A(r_i \cdot f_A(b_i) + b_i)$. B sends the ciphertexts $c_{B,i}$ $(i = 1, \ldots, l)$ and the encrypted coefficients $E_B(\beta_i)$ to A. A decrypts the received ciphertexts $c_{B,i}$ $(i = 1, \ldots, l)$ under D_A. For each $b_i \in P_A \cap P_B$, the ciphertext $c_{B,i}$ will decrypt to b_i (matching some $a_j \in P_A$). For each $b_i \notin P_A \cap P_B$, $c_{B,i}$ will decrypt under D_B to some arbitrary value. A thus learns all elements of the intersection, but nothing about any other rule in P_B. Then, A computes $c_{A,i} = E_B(r_i \cdot f_B(a_i) + a_i)$ $(i = 1, \ldots, k)$ and sends these ciphertexts to B. B decrypts the received ciphertexts under D_B. For each $a_i \in P_A \cap P_B$, $c_{A,i}$ will decrypt to a_i (matching some $b_j \in P_B$). For each $a_i \notin P_A \cap P_B$, $c_{A,i}$ will decrypt under D_B to some arbitrary value. B thus also learns all elements in the intersection $P_A \cap P_B$.

Discussion: If parties A and B follow the PCP protocol, both will learn the intersection of their policies. The PCP protocol simply uses the private cardinality protocol of [12]. Following the detailed proof in [12], each direction of the protocol is privacy-preserving in the semi-honest model. This is due to the fact that ciphertexts sent from A (B) to B (A) that do not correspond to a common policy rule of both parties decrypt to some arbitrary value. Moreover, the views of B (A) for two inputs of A (B) are indistinguishable due to the semantical security of the encryption scheme used.

We again evaluate the complexity of the PCP protocol by counting (1) the number of comparisons A and B have to conduct, (2) the number of polynomial coefficients they need to encrypt, (3) the number of decryptions, and (4) the number of commitments of rules exchanged. The result are the same as shown in Table 7.

References

[1] M. Atallah and W. Du. Secure Multi-Party Computational Geometry. In *Proceedings of the Seventh International Workshop on Algorithms and Data Structures*, 2001.

[2] D. Bell and L. LaPadula. Secure Computer Systems: Mathematical Foundations. Technical Report MTR-2547, Vol. I – III, MITRE Corporation, Bedford, MA, November 1973.

[3] M. Blaze, J. Feigenbaum, J. Ioannidis, and A. D. Keromytis. The KeyNote Trust Management System Version 2. RFC 2704, 1999.

[4] M. Blaze, J. Feigenbaum, and J. Lacy. Decentralized Trust Management. In *Proceedings of the 17th IEEE Symposium on Security and Privacy*, 1996.

[5] Jan Camenisch and Gregory M. Zaverucha. Private intersection of certified sets. In *Financial Cryptography*, pages 108–127, 2009.

[6] Emiliano De Cristofaro, Stanislaw Jarecki, Jihye Kim, and Gene Tsudik. Privacy-Preserving Policy-Based Information Transfer. In *Privacy Enhancing Technologies*, pages 164–184, 2009.

[7] Emiliano De Cristofaro and Gene Tsudik. Practical Private Set Intersection Protocols with Linear Complexity. In *In Proceedings of Financial Cryptography and Data Security*, 2010.

[8] Nicodemos Damianou, Naranker Dulay, Emil Lupu, and Morris Sloman. The Ponder Policy Specification Language. In *Proceedings of Policy Worshop*, 2001.

[9] P. Dinsmore, D. Balenson, M. Heyman, P. Kruus, C. Scace, and A. Sherman. Policy-Based Security Management for Large Dynamic Groups: A Overview of the DCCM Project. In *Proceedings of DARPA Information Survivability Conference and Exposition*, 2000.

[10] C. Ellison, B. Frantz, B. Lampson, R. Rivest, B. Thomas, and T. Ylonen. SPKI Certificate Theory. Request for Comments (Proposed Standard) 2693, Internet Engineering Task Force, September 1999.

[11] Carl Ellison. SPKI Requirements. Request for Comments (Proposed Standard) 2692, Internet Engineering Task Force, September 1999.

[12] M. J. Freedman, K. Nissim, and B. Pinkas. Efficient Private Matching and Set Intersection. In *Proceedings of EUROCRYPT'04*, 2004.

[13] O. Goldreich. Foundations of Cryptography - Volume 2, 2004.

[14] O. Goldreich, S. Micali, and A. Wigderson. How to Play ANY Mental Game. In *STOC '87: Proceedings of the Nineteenth Annual ACM Conference on Theory of Computing*. ACM, 1987.

[15] L. Gong and X. Qian. The Complexity and Composability of Secure Interoperation. In *Proceedings of the IEEE Symposium of Security and Privacy (S&P'94)*, 1994.

[16] Carmit Hazay and Yehuda Lindell. Efficient protocols for set intersection and pattern matching with security against malicious and covert adversaries. Cryptology ePrint Archive, Report 2009/045, 2009. `http://eprint.iacr.org/`.

[17] Susan Hohenberger and Stephen A. Weis. Honest-verifier private disjointness testing without random oracles. In *Privacy Enhancing Technologies*, pages 277–294, 2006.

[18] Stanislaw Jarecki and Xiaomin Liu. Efficient oblivious pseudorandom function with applications to adaptive ot and secure computation of set intersection. In *TCC*, pages 577–594, 2009.

[19] Lea Kissner and Dawn Song. Private and Threshold Set-Intersection. In *Proceedings of CRYPTO'05*, August 2005.

[20] Lea Kissner and Dawn Song. Private and Threshold Set-Intersection. Technical Report CMU-CS-05-113, Carnegie Mellon University, February 2005.

[21] K. Kursawe, G. Neven, and P. Tuyls. Private Policy Negotiation. In *Proceedings of Financial Cryptography'06*, 2006.

[22] B.W. Lampson. Protection. In *Proceedings of the 5th Princeton Conference on Information Science and Systems*, 1971.

[23] P. McDaniel and A. Prakash. Ismene: Provisioning and Policy Reconciliation in Secure Group Communication. Technical Report CSE-TR-438-00, University of Michigan, 2000.

[24] Patrick McDaniel and Atul Prakash. Methods and Limitations of Security Policy Reconciliation. In *Proceedings of the IEEE Symposium of Security and Privacy (S&P'02)*, 2002.

[25] P. Paillier. Public-key Cryptosystems Based on Composite Degree Residuosity Classes. In *Proceedings of EUROCRYPT'99*, 1999.

[26] Carlos Ribeiro, Andre Zuquete, Paulo Ferreira, and Paulo Guedes. SPL: An Access Control Language for Security Policies with Complex Constraints. In *Proceedings of Network and Distributed System Security Symposium (NDSS)*, February 2001.

[27] P. Samarati and S Vimercati. Access Control: Policies, Models, and Mechanisms. *Foundations of Security Analysis and Design*, 2000.

[28] Ravi S. Sandhu and P. Samarati. Access Control: Principles and Practice. *IEEE Communications Magazine*, 32(9), 1994.

[29] M. Sloman. Policy Driven Management for Distributed Systems. *Journal of Network and Systems Management*, 2(4), 1994.

[30] H.B. Wang, S. Jha, P. McDaniel, and M. Livny. Security Policy Reconciliation in Distributed Computing Environments. In *Proceedings of the IEEE Workshop on Policies for Distributed Systems and Networks (Policy'04)*, 2004.

[31] A. Yao. Protocols for Secure Computation. In *Proceedings of the IEEE (FOCS)'82*, 1982.

[32] J. Zao, L. Sanchez, M. Condell, C. Lynn, M. Fredette, P. Helinek, P. Krishnan, A. Jackson, D. Mankins, M. Shepard, and S. Kent. Domain Based Internet Security Policy Management. In *Proceedings of DARPA Information Survivability Conference and Exposition*, 2000.