This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the ICC 2007 proceedings.

Distributed Privacy-Preserving Policy Reconciliation

Ulrike Meyer*, Susanne Wetzel†, and Sotiris Ioannidis†
*Siemens Networks GmbH & Co, München, Germany
†Stevens Institute of Technology, Hoboken, NJ, USA

***Abstract*— Organizations use security policies to regulate how they share and exchange information, *e.g.,* under what conditions data can be exchanged, what protocols are to be used, who is granted access, *etc.* Agreement on specific policies is achieved though *policy reconciliation*, where multiple parties, with possibly different policies, exchange their security policies, resolve differences, and reach a consensus. 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 security policy reconciliation. We introduce new protocols that meet the privacy requirements of the organizations and allow parties to find a common policy rule which *maximizes* their individual preferences.**

I. Introduction

Sharing of sensitive information between organizations is a very challenging task. The difficulty lies not only in the structure of the resources that parties want to share, but also in the security policy rules that govern *what* data can be shared and *how* it can be shared. To control such interactions, (two or more) participants have to accept a set of rules that they will follow for the duration of the interaction. Each party has to express their desired security rules in a common format and follow a protocol with the other parties. The process will determine which rules control their future interactions. This protocol is called a *reconciliation protocol* and its output is a security policy that is consistent with the security 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 requirements and repeat the protocol.

	Wireless Provider	User
Requirement	Security, compat. with customer	Battery life, compat. with network
High preference	Strong Crypto	No Crypto
Medium preference	Medium Crypto	Medium Crypto
Low preference	No Crypto	Strong Crypto

Fig. 1. Requirements for the wireless provider and the network user. The wireless provider wants to maintain a secure network but still provide compatibility and service to customers with older devices, while the user wants to achieve maximum battery life and still get service.

The importance of security policy reconciliation is tied to the importance of communication in general. One prominent field of application is mobile communication. In mobile communication networks, users must form associations with different network providers as they roam from one network to another. To form such associations, the user and the network provider have to agree on a *set of attributes* that will govern their interaction (authentication, type of connection, underlying security, *etc.*). Both the provider and the user can represent their requirements in form of a security policy that captures their individual needs. It is possible—and likely—that each party will have a wide range of policies that satisfy their individual communication requirements, ranked in some order of preference. Obviously, participants may have conflicting preferences. For example, the network provider may want to use strong encryption whenever possible, but might still be willing to support certain weaker encryption mechanisms, or no encryption at all, so that it can provide service to older mobile devices. The mobile user, on the other hand, might prefer to maximize the battery life of his mobile device and therefore prefer not to use encryption whenever possible (see Figure 1).

Revealing their policies to the other party may not negatively impact the reconciliation process itself. It, however, can pose a major security risk since any disclosure of security policy can be misused by an attacker to compromise the system. Continuing our example, an attacker may exploit the knowledge that the wireless provider is willing to provide service without requiring "strong crypto" to mount a series of attacks [1]. Of course, in our example, the parties can easily discover each other's security policies, since they only involve a single attribute, *i.e.,* the use of encryption. We deliberately keep our example simple to illustrate why preserving privacy during policy reconciliation is crucial. One can easily envision scenarios between organizations that will require the reconciliation of policies that include a large number of attributes (types of users, types of data, protocols, *etc.*). Obviously, meeting the privacy objective of each organization in those cases is much more important. In this paper we show that it can effectively be met by means of privacy-preserving policy reconciliation.

In general, a security policy is a collection of rules that express which actions are permitted and disallowed in systems [2]. It can take the form of specific access control rules [3], policy credentials [4], or security policy language statements [5]. 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 security needs of all parties are met.

The problem of policy reconciliation, in its most general form, is intractable [6], [7]. For this reason, previous work has focused on policy reconciliation for specific representations of a security policy [8], or by using heuristic algorithms to achieve tractability. While such approaches are effective in

1-4244-0353-7/07/$25.00 ©2007 IEEE

reconciling policies, they do not take into account *privacy requirements* of the participants. Ideally, upon termination of a reconciliation protocol, the parties should have only revealed the part of their policy that enables them to form the association and nothing more. Our work addresses privacy-preservation in policy reconciliation. In particular, as the main contribution of this paper we introduce new policy reconciliation protocols that meet the privacy objective while maximizing the preferences of all parties.

II. Related Work

Security Policy Reconciliation: Gong and Qian in [6] analyze the complexity of secure interoperation between systems with heterogeneous access control structures. They proved that composing authorization policies is NP-complete. McDaniel and Prakash defined the Ismene policy language [9] which permits the specification and reconciliation of group security requirements. In [7], they show that general purpose provisioning policy reconciliation is intractable. Wang *et al.* demonstrate that it is possible to structure security policies in such a way that policy reconciliation becomes tractable [8]. Zao *et al.* propose the Security Policy System (SPS) which resolves IPsec security associations between domains of communication [10]. Reconciliation is achieved by intersecting sets of policy values. Dinsmore *et al.* [11] 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 not on the reconciliation algorithm.

Automated Trust Negotiation: First proposed by Winsborough *et al.* [12], it is the procedure where two parties exchange digitally signed credentials that contain attribute information to establish trust and make access control decisions. In environments like the Internet where there may be few or no pre-existing relationships, parties that seek to form a trust relationship might be unwilling to release sensitive credentials [13], [14]. To overcome this difficulty, a number of schemes have been proposed recently that use cryptography or multiple rounds of negotiations to protect protecting credentials and attributes [15], [16], [17]. In automated trust negotiation, participants specify access control policies for the disclosure of credentials. The negotiation phase consists of a sequence of exchanges that are controlled by the access control policies defined for the credentials. At each round, parties gain higher levels of mutual trust, permitting access control policies for more sensitive credentials to be satisfied, which in turn enable these credentials to be exchanged.

In security policy reconciliation, there are *no additional access control policies* put in place to protect the policies that the parties try to reconcile. Instead, each participant presents its individual security policies and subsequently the cryptographic protocol determines the subset of policies that satisfy all parties.

	Wireless Provider			User		
	3DES	*DES*	*None*	*3DES*	*DES*	*None*
Rule 1	1	0	0	0	0	1
Rule 2	0	1	0	0	1	0
Rule 3	0	0	1	1	0	0

Fig. 2. Policy rules expressed as bit-strings and ranked in order of preference.

Privacy-Preserving Protocols: Freedman *et al.* [18] consider the problem of computing the intersection of private datasets of two parties and analyze their protocol under a number of threat models. Their solution is based on representing sets as roots of polynomials. Kissner and Song [19], [20] extend the results of [18] to utilize properties of polynomials beyond evaluation at given points. Their work provides privacy-preserving operations for union, intersection, and element reduction. The main contribution of our paper is that we build on the privacy-preserving operations results in [18] to construct two new and more complex privacy-preserving protocol for security policy reconciliation with *preferences* for *two different notions of fair reconciliation*. In addition, we 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.

III. 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 IV and the Appendix.

A. 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 [6], [7]. 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, security policies are represented as bitstrings, which can be easily manipulated and operated on.

We illustrate our policy representation by continuing the example from Section I. The first step is for the participants to agree on a set of attributes that will be used to define security policy rules. In our example, we have settled for three attributes defining the underlying use of encryption, *3DES*, *DES*, and *None*. Then, each participant represents its requirements by defining rules in order of preference. In our example, the wireless provider prefers increased security but is willing to accommodate older devices, whereas the user want to maximize battery life, but is willing to compromise and use encryption if that is required to access the network.

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 $P_A = (a_{ij})_{1 \leq i \leq k, 1 \leq j \leq n} \in \{0,1\}^{k \times n}$. 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 2, *3DES*, *DES* and *None* are the attributes defined in the security policy. Rule a_1 for the wireless provider is the bit-string $(1,0,0)$ specifying that he requires the use of *3DES*.

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 $A_j = B_j$ for $j = 1, \ldots, u$.

Definition 3.2: A policy preference order of party A on its policy P_A is a total order $\leq_A$ of its policy rules.

Definition 3.3: Let $\{a_1, \ldots, a_k\}$ be the set of policy rules of A in decreasing order of $\leq_A$. The ranking of a_i corresponding to $\leq_A$ is defined as $\text{rank}_A(a_i) := k - i + 1$ $(i = 1, \ldots, k)$.

Definition 3.4: A policy preference order composition scheme takes two policy preference orders $\leq_A$ on a policy P_A and $\leq_B$ on P_B as input and combines them to a joint (partial or total) order $\leq_{AB}$ on $P_A \cap P_B$.
Examples for preference order composition schemes are:

Definition 3.5: The **sum of ranks** composition scheme combines $\leq_A$ on P_A and $\leq_B$ on P_B to the joint total preference order $\leq_{AB}$ defined as follows: If $x, y \in P_A \cap P_B$, then $x \leq_{AB} y :\Leftrightarrow \text{rank}_A(x) + \text{rank}_B(x) \leq \text{rank}_A(y) + \text{rank}_B(y)$.

Definition 3.6: The **maximized minimum of ranks** composition scheme combines $\leq_A$ on P_A and $\leq_B$ on P_B to the joint total preference order $\leq_{AB}$ as follows: If $x, y \in P_A \cap P_B$, then $x \leq_{AB} y :\Leftrightarrow \min\{\text{rank}_A(x), \text{rank}_B(x)\} \leq \min\{\text{rank}_A(y), \text{rank}_B(y)\}$.

B. Protocols for Privacy-Preserving Policy Reconciliation

Drawing on the previous definitions, we now introduce the protocols for privacy-preserving policy reconciliation.

Policy reconciliation in general has the single 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.

While on one hand, privacy-preservation allows for naturally extending the objective for policy reconciliation, it on the other hand also leads to distinguishing different levels of privacy. In particular, introducing privacy-preservation to the objective of policy reconciliation will prevent parties from disclosing any information to the other party aside from the policies they have in common. The disclosure of information can be further limited (thus increasing the level of privacy) by not revealing the actual policies they share but instead disclosing only the number of policies the parties have in common. Recognizing preferences on policies as additional sensitive information, the primary objective can be extended to having the two parties not learn anything about the other party other than the policy that maximizes both parties' combined preferences.

Keeping these different privacy considerations in mind, we define the three different protocols for privacy-preserving 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 private 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 which policy rules they have in common. I.e., if a PCP takes P_A and P_B as input, then A and B learn $P_A \cap P_B$.

A PCP scheme thus allows two parties to reveal nothing but their *common* policy rules to each other during reconciliation.

Protocol 2: A **privacy-preserving common policy cardinality** scheme (PC^2) is a two-party protocol between parties A and B. Their respective private 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 private policies but the number of policy rules they have in common. I.e., if a PC^2 takes P_A and P_B as input, then A and B learn $|P_A \cap P_B|$ and nothing else.

A PC^2 thus allows two parties A and B to determine how many policy rules they have in common *without* revealing any other information 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.

Protocol 3: A **privacy-preserving preference-maximizing policy reconciliation** scheme for a preference order composition scheme $\mathcal{C}$ ($3\text{PR}^{\mathcal{C}}$) 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 one common policy rule max that maximizes the combined preference order $\leq_{AB}$ of $\leq_A$ and $\leq_B$ under $\mathcal{C}$ as well as in which round of the protocol max was determined. I.e., if $3\text{PR}^{\mathcal{C}}$ takes P_A, P_B, $\leq_A$, and $\leq_B$ as input, then A and B learn $\max_{x \in P_A \cap P_B; \leq_{AB}}\{x\}$.
Note that in a $3\text{PR}^{\mathcal{C}}$ 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 the two parties that they 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.

C. 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 according to their prescribed actions in the protocol.[1] A PCP, a PC^2, or a 3PR$^{\mathcal{C}}$ protocol is **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 output of the protocol and its 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 (see [21] for the formal definition of privacy-preservation in the semi-honest model). 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 [21], 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 run or the output of the protocol.[2] 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.

D. Privacy-Preserving Tools

Privacy-Preserving Set Intersection: In [18], Freedman *et al.* present a protocol to compute the intersection of two data sets in a privacy-preserving way. Their private matching scheme 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 server 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 protocols 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 [22]. 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 $f(X)$.

Private Cardinality Matching: In [18], Freedman *et al.* also present 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.[3] For data sets of size $O(k)$, the protocols results in a communication overhead of $O(k)$ and computational overhead of $O(k \ln \ln k)$.

IV. Privacy-Preserving Reconciliation with Preferences

Building on the work of Freedman *et al.* it is possible to build protocols for PC^2 and PCP in a straight-forward manner (see Appendix). However, implementing a 3PR$^{\mathcal{C}}$ schemes requires some sophisticated protocol design. The main contribution of the 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 the privacy-preserving set intersection protocol by Freedmen *et al.* as building block.

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 $\{0,1\}^n$ (Remark 3.1). In both 3PR$^{\mathcal{C}}$ protocols we assume that P_A and P_B have

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

[2] As a consequence of the three above mentioned unpreventable actions a malicious party can take, a PCP scheme, for example, can never 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.

[3] It is important to note that in the Freedman *et al.* schemes the server learns the size of X_C. Other schemes avoid this leakage at the cost of efficiency (*e.g.,* [23] in the case of private cardinality matching and [24] for private set intersection).

the same number of policy rules, that is $k = l$. Furthermore, P_A and P_B are in decreasing order of $\leq_A$ and $\leq_B$.

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

The objective of the following 3PR$^{\mathcal{C}}$ protocol is for the two parties A and B to determine one common policy rule max in a privacy-preserving manner that maximizes the joint preference order on $P_A \cap P_B$ defined by

$$x \leq_{AB} y \Leftrightarrow \text{rank}_A(x) + \text{rank}_B(x) \leq \text{rank}_A(y) + \text{rank}_B(y).$$

3PR$^{\mathcal{C}}$ 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:

STEP 1: A sends $E_A^{(1)}$ to B. B chooses a random $r_{B,1}^{(1)}$, computes $c_{B,1}^{(1)} = E_A\Big(r_{B,1}^{(1)} \cdot f_A^{(1)}(b_1) + b_1\Big)$, and sends $E_B^{(1)}$, $E_B^{(2)}$, as well as $c_{B,1}^{(1)}$ to A. A decrypts $c_{B,1}^{(1)}$ under D_A and compares the result to a_1. If $a_1 = D_A(c_{B,1}^{(1)})$, then $a_1 = b_1$, that is, A has found $max := a_1 = b_1$ and continues with the *Match Confirmation Phase*. Otherwise, A continues with *Step 2* of the *Commitment Phase*.

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

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

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 $c_{A,m}^{(i-m+1)}$ 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

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

and sends them together with $E_B^{(i+1)}$ 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 *Step* $i+1$ of the *Commitment Phase*.

MATCH CONFIRMATION PHASE: If it was A who found $max = a_{i-m+1} = b_m$ in *Step* i 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 $a_{i-m+1} = b_m = max$.

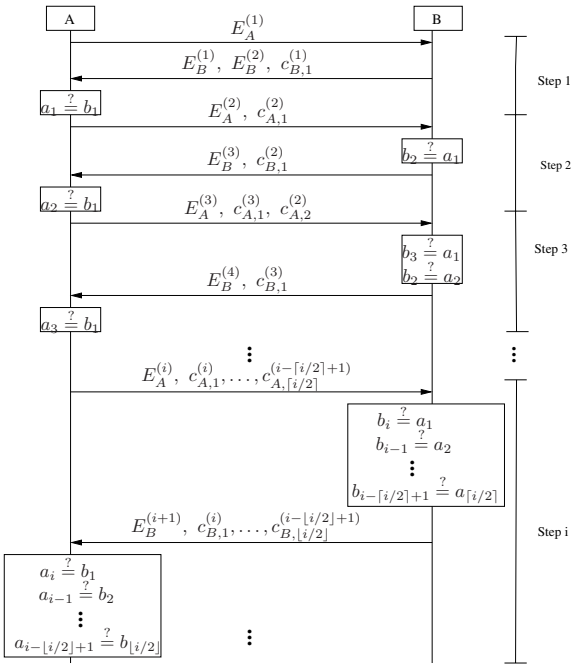

Fig. 3. Commitment Phase of the sum of ranks 3PR$^{\mathcal{C}}$. In Step i, A for $j = 1, \ldots, \lfloor i/2 \rfloor$ compares a_{i-j+1} with b_j and B for $j = 1, \ldots, \lceil i/2 \rceil$ compares b_{i-j+1} with a_j without obtaining knowledge of these values (unless they are equal), that is, in a privacy-preserving manner. Note that in Step i all policy rules are compared that result in the same sum of ranks.

If it was B who found $max = b_{i-m+1} = a_m$ in *Step* i 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 $a_m = b_{i-m+1} = max$.

Remark 4.1: Unlike in PCP and PC^2 we have assumed for the 3PR$^{\mathcal{C}}$ scheme for the sum of ranks composition scheme 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 policy rules containing only zeros for every attribute. A matching zero policy as result of 3PR$^{\mathcal{C}}$ then indicates that A and B do not share a single valid policy rule.

Discussion: For $P_A \cap P_B \neq \emptyset$, the *Commitment Phase* of the 3PR$^{\mathcal{C}}$ protocol always terminates with one of the parties finding a match. This is due to the fact that for each pair (l, s) $1 \leq l \leq k$, $1 \leq s \leq k$, a_l is compared with b_s in *Step* $l+s-1$. 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 Step $l+s-1$. Let $max = a_l = b_s$ be a match found in *Step* $l + s - 1$. Then no match was found in any previous step 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 Step $r + v - 1$ of the *Commitment Phase*. This obviously contradicts the assumption.

The 3PR$^{\mathcal{C}}$ scheme uses one round of the private intersection

protocol of Freedman *et al.* in each round. If A (B) would learn anything besides preference-maximizing policy rules at the end of the protocol, A (B) would have had to learn this in one of the rounds. As each round is privacy-preserving in the semi-honest model, A (B) learns nothing in any round unless a match is found. If a match is found by A (B), A (B) may learn more than one preference maximizing policy rule.[4] This is due to the fact that more than one pair of policy rules is evaluated by A (B) in each round. Upon termination of the protocol in *Step* i both parties also learn about the correlation of their preferences of max to i. Obviously, the protocol with the additional subrounds is privacy-preserving in the semi-honest model.[5] 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.

Following the analysis of Freedman *et al.*, for policies of size $O(k)$, the communication overhead of the protocol is bounded by $O(k^2)$ and the computational overhead is bounded by $O(k^2 \ln \ln k)$. A more detailed analysis can be found in the journal version of this paper. The performance of this protocol is evaluated in [25].

B. 3PR$^{\mathcal{C}}$ protocol for the maximized minimum of ranks compostion scheme

The objective of the following 3PR$^{\mathcal{C}}$ protocol is for the two parties A and B to determine a common policy rule max in a privacy-preserving manner that maximizes the joint preference order on $P_A \cap P_B$ defined by

$$x \leq_{AB} y \Leftrightarrow \min(\text{rank}_A(x), \text{rank}_B(x)) \leq \min(\text{rank}_A(y), \text{rank}_B(y)).$$

3PR$^{\mathcal{C}}$ Protocol for Maximized Minimum Composition: 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:

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

STEP $i \geq 2$: 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 $c_{A,m}^{(i)}$ 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 together with $E_B^{(i+1)}$ 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 $c_{B,m}^{(i)}$ decrypts to a_i, A has found $max := a_i = b_l$ and continues with the *Match Confirmation Phase*. Otherwise, A continues with *Step* $i+1$ of the *Commitment Phase*.

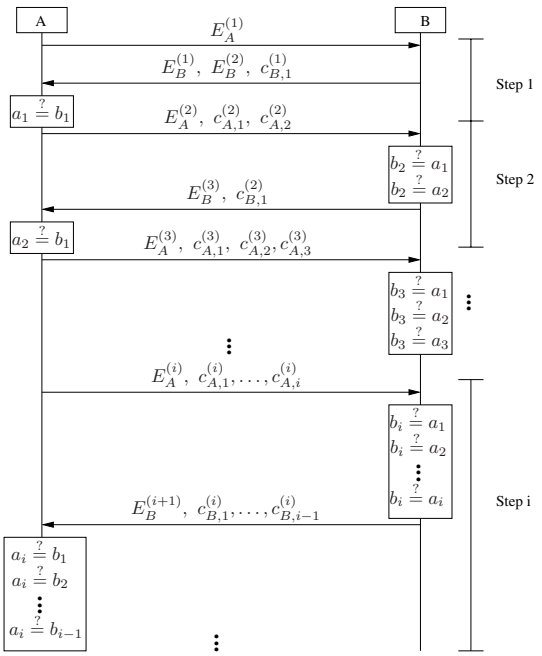

Fig. 4. Commitment Phase of 3PR$^{\mathcal{C}}$ for the maximized minimum of ranks composition scheme. In Step i, A compares a_i with $b_1, \ldots, b_{i-1}$ and B compares b_i with $a_1, \ldots, a_i$ without obtaining knowledge of these values, that is, in a privacy-preserving manner.

MATCH CONFIRMATION PHASE: If it was A who found max in *Step* 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 *Step* 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$^{\mathcal{C}}$ scheme for the sum of ranks composition scheme, the 3PR$^{\mathcal{C}}$ scheme for the maximized minimum of ranks composition scheme allows the same

[4]However, A (B) does not learn anything about policy rules of B (A) that do not maximize the composed preferences.

[5]This is due to the fact that up to the point where a match is found, A (B) receives only ciphertexts from B (A) that decrypt to an arbitrary value. As soon as a match is found by one of the honest party's, this party acknowledges the match to the other party and the protocol terminates.

rank to be assigned to multiple policy rules assuming that A and B still have the same number k of policy rules.

Discussion: For $P_A \cap P_B \neq \emptyset$, the *Commitment Phase* of the 3PR$^{\mathcal{C}}$ protocol for the maximized minimum of ranks 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 *Step* 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 $\max(\min(\text{rank}_A(max), \text{rank}_B(max))) = \text{rank}_A(max)$. 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

$$\min(\text{rank}_A(p), \text{rank}_B(p)) > \text{rank}_A(max)$$

This implies that $\text{rank}_A(p) > \text{rank}_A(max)$ and $\text{rank}_B(p) > \text{rank}_A(max)$ 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.

The 3PR$^{\mathcal{C}}$ for the maximized minimum of ranks scheme uses one round of the private intersection protocol of Freedman *et al.* in each round. If A (B) would learn anything besides preference-maximizing policies at the end of the protocol, A (B) would have had to learn this in one of the rounds. As each round is privacy-preserving in the semi-honest model, A (B) learns nothing in any round unless a match is found. As argued before, this match always is a preference-maximizing policy rule. However, as in the 3PR$^{\mathcal{C}}$ for the sum of ranks composition scheme, A (B) may find more than one preference-maximizing policy rule. This is due to the fact that in each round of the *Commitment Phase* A (B) evaluates several pairs of policy rules for which the maximized minimum of ranks is the same. Upon termination of the protocol in *Step* i both parties also learn about the correlation of their preferences of max to i. Consequently, the protocol is privacy-preserving in the semi-honest model.[6] 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.

Following the analysis of Freedman *et al.*, for policies of size $O(k)$, the communication overhead of the protocol is bounded by $O(k^2)$ and the computational overhead is bounded by $O(k^2 \ln \ln k)$ (more details are provided in the journal version of this paper). A performance evaluation of this protocol can be found in [25].

[6]This is due to the fact that up to the point where a match is found, A (B) receives only ciphertexts from B (A) that decrypt to an arbitrary value. As soon as a match is found by one of the honest parties, this party acknowledges the match to the other party and the protocol terminates.

V. Conclusions

In this paper we addressed the problem of distributed privacy-preserving policy reconciliation, in particular, how multiple parties reconcile their security policy while only revealing the parts of their policy that are required to achieve a communication association. Using our two 3PR$^{\mathcal{C}}$ protocols, parties can rank their policies according to their preferences. The first of our new protocols then returns the policy that maximizes the sum of the ranks of the individual preferences. The second of our new protocols returns the policy that maximizes the minimum of the ranks of the individual preferences. We accomplish this, for policies of size $O(k)$ with communication overhead bounded by $O(k^2)$ and computational overhead bounded by $O(k^2 \ln \ln k)$.

References

[1] Meyer, U., Wetzel, S.: On the Impact of GSM Encryption and Man-in-the-Middle Attacks on the Security of Interoperating GSM/UMTS Networks. In: IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC2004). (2004)

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

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

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

[5] Ribeiro, C., Zuquete, A., Ferreira, P., Guedes, P.: SPL: An Access Control Language for Security Policies with Complex Constraints. In: Proceedings of Network and Distributed System Security Symposium (NDSS). (2001)

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

[7] McDaniel, P., Prakash, A.: Methods and Limitations of Security Policy Reconciliation. In: Proceedings of the IEEE Symposium of Security and Privacy (S&P'02). (2002)

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

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

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

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

[12] Winsborough, W.H., Seamons, K.E., Jones, V.E.: Automated Trust Negotiation. In: Proceedings of DARPA Information Survuvability Conference and Exposition, Volume I, IEEE Press. (2000)

[13] Winsborough, W., Li, N.: Towards Practical Automated Trust Negotiation. In: Proceedings of IEEE 3rd International Workshop on Policies for Distributed Systems and Networks (Policy'02). (2002)

[14] Irwin, K., Yu, T.: Preventing Attribute Information Leakage in Automated Trust Negotiation. In: Proceedings of the ACM Conference on Computer and Communications Security (CCS). (2005)

[15] Winsborough, W.H., Li, N.: Protecting Sensitive Attributes in Automated Trust Negotiation. In: Proceedings of the 2002 ACM Workshop on Privacy in the Electronic Society (WPES'02). (2002)

[16] Holt, J.E., Bradshaw, R.W., Seamons, K.E., Orman, H.: Hidden Credentials. In: WPES'03: Proceedings of the 2003 ACM Workshop on Privacy in the Electronic Society. (2003)

[17] Seamons, K., Winslett, M., Yu, T.: Limiting the Disclosure of Access Control Policies during Automated Trust Negotiation. In: Proceedings of Network and Distributed System Security Symposium. (2001)
[18] Freedman, M.J., Nissim, K., Pinkas, B.: Efficient Private Matching and Set Intersection. In: Proceedings of EUROCRYPT'04. (2004)
[19] Kissner, L., Song, D.: Private and Threshold Set-Intersection. In: Proceedings of CRYPTO'05. (2005)
[20] Kissner, L., Song, D.: Private and Threshold Set-Intersection. Technical Report CMU-CS-05-113, Carnegie Mellon University (2005)
[21] Goldreich, O.: Secure Multi-Party Computation. http://www.wisdom.weizmann. ac.il/ oded/pp.html (2002)
[22] Paillier, P.: Public-key Cryptosystems Based on Composite Degree Residuosity Classes. In: Proceedings of EUROCRYPT'99. (1999)
[23] Atallah, M., Du, W.: Secure Multi-Party Computational Geometry. In: Proceedings of the Seventh International Workshop on Algorithms and Data Structures. (2001)
[24] Yao, A.: Protocols for Secure Computation. In: Proceedings of the IEEE (FOCS)'82. (1982)
[25] Voris, J., Ioannidis, S., Wetzel, S., Meyer, U.: Performance Evaluation of a Privacy-Preserving Policy Reconciliation Protocol. In: In Proceedings of IEEE Workshop on Policies for Distributed Systems and Networks (Policy'07). (2007)

APPENDIX

A. Privacy-Preserving Policy Reconciliation

In this section we provide protocols for PC^2 and PCP introduced in Section III-B. 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.*

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 $(i = 1, \ldots, k)$ 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 $(i = 1, \ldots, l)$ 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 [18], 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).

As in the protocol of Freedman *et al.* for policies of size $O(k)$, the protocol results in a communication overhead of $O(k)$ and computational overhead of $O(k \ln \ln k)$.

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 $(i = 1, \ldots, k)$ 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 $(i = 1, \ldots, l)$ 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 [18]. Following the detailed proof in [18], 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.

As in the protocol of Freedman *et al.* for policies of size $O(k)$, the protocol results in a communication overhead of $O(k)$ and computational overhead of $O(k \ln \ln k)$.