Merging Heterogeneous Security Orderings

P.A. Bonatti[1], M.L. Sapino[1] and V.S. Subrahmanian[2*]

[1] Università di Torino
{bonatti,mlsapino}@di.unito.it
[2] University of Maryland
vs@cs.umd.edu

Abstract. The problem of integrating multiple heterogeneous legacy databases is an important problem. Many papers [7, 9, 3] to date on this topic have assumed that all the databases comprising a mediated/federated system share the same security ordering. This assumption is often not true as the databases may have been developed independently by different agencies at different points in time. In this paper, we present techniques by which we may merge multiple security orderings into a single unified ordering that preserves the security relationships between orderings. We present a logic programming based approach, as well as a graph theoretical approach to this problem.

Keywords: Theoretical Foundations of Security, Heterogeneous Mediated/Federated Systems.

1 Introduction

Complex applications in today's rapidly changing world require the ability to access a wide variety of distributed, heterogeneous data sources. In order to assist the *author* of such complex applications, Wiederhold [14, 15] has proposed the important concept of a *mediator* as a paradigm for integrating heterogeneous data and software. Major efforts towards the construction of mediators are currently underway at many universities and companies [12, 1, 4, 10].

One of the fundamental problems faced by each and every one of these efforts is the problem of *security*. Besides the standard problems related to security in databases and information systems, one has to tackle specific problems raised by the heterogeneous nature of data sources. In particular, it is not reasonable to assume that all packages participating in a mediated system and the mediator itself use the same security orderings.

* Partially supported by the Army Research Office under grant DAAH-04-95-10174, by the Air Force Office of Scientific Research under grant F49620-93-1-0065, by ARPA/Rome Labs contract Nr. F30602-93-C-0241 (Order Nr. A716), by NSF Young Investigator award IRI-93-57756, by NSF award IRI-93-14905, and by the Army Research Laboratory under Cooperative Agreement DAAL01-96-2-0002 Federated Laboratory ATIRP Consortium.

For example, a Supply Mediator might use an **inventory** relation, created by an Army Logistics agency, and a GIS database, developed by a defense contractor, and it may very well be the case that these two groups used different security orderings in their design. A mediator must be able to *automatically combine the security orderings used by the different packages in a semantically meaningful and security-preserving way.* Such a combination will result in a new security level ordering that can be used by the mediator. In this paper, we will present methods by which different security orderings used by individual packages may be neatly *combined* into a new security ordering that captures the essential properties of the constituent orderings. Furthermore, we will show that this combination may be elegantly implemented using well known logic programming techniques and graph algorithms, and is solvable in polynomial time.

In the next section, we introduce the basic principles of our approach to the combination of heterogeneous security orderings. In Sec. 3, we introduce an axiomatization of the combinability problem based on a logic program P, and show how a global security ordering can be computed by querying P. In Sec. 4 we introduce an alternative approach, based on a graph representation of the problem, which leads to more efficient algorithms. We conclude with a section on related work.

2 Combining Heterogeneous Security Orderings

A *security ordering* is a partial order $(S, \leq)$, where S is a set of security levels and $\leq$ is a partial order over S. Intuitively, in an individual data source, each user is assigned a security level that determines the user's capabilities; higher security levels correspond to higher clearance. Informally speaking, we would like to construct a security ordering $(S, \leq)$ that takes into account and semantically merges the security orderings $(S_1, \leq_1), \ldots, (S_z, \leq_z)$ that are used by the different packages participating in a mediated system. Users may then be assigned security levels from the merged ordering $(S, \leq)$.

Example 1. Figure 1 shows two different security orderings used by two relational DBs, **db1** and **db2**. The first database uses the ordering **u** (unclassified), **s** (secret), **ts** (top-secret), **ts-sci1**, and **ts-sci2**, where **ts-sci** stands for top-secret special compartmented information. The classification levels of **db2** may be read similarly. For now, the reader can ignore the dotted line in Figure 1. There are several things to note, however. As **db1** and **db2** may have been created independently, it is entirely possible that these orderings refer to different things. For example, it may well be the case that the classification level **ts** in $(S_1, \leq_1)$ is equal to the classification level **s** in $(S_2, \leq_2)$. Later (Example 2) we will show how this may be captured within our framework. □

The aim of this section is to find a way of *combining* a given set of security orderings $(S_1, \leq_1), \ldots, (S_z, \leq_z)$. In our framework, we will merge the given security orderings into a new ordering in such a way that certain constraints are preserved. These constraints express relationships between security levels in

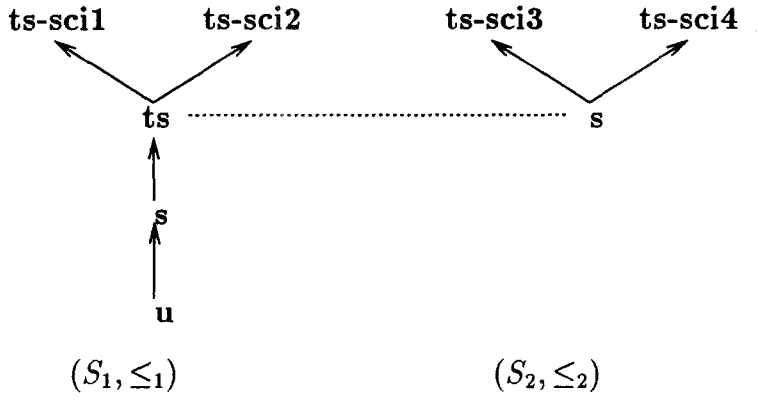

Fig. 1. Two different Security Orderings

different security orderings, thus providing us with information on how these different security orderings are linked. In order to avoid confusion, we will write $x : i$ to denote the security level x of the ordering $(S_i, \leq_i)$.

Definition 1. An *interoperation security constraint set* is a set of statements of the form $x : i \preceq y : j$ or $x : i \not\preceq y : j$ where $x \in S_i$ and $y \in S_j$ and $i \neq j$. Sometimes, when an interoperation security constraint set contains both the constraints $x : i \preceq y : j$ and $y : j \preceq x : i$, we will replace them by the single constraint $x : i = y : j$. Similarly, when both $x : i \preceq y : j$ and $y : j \not\preceq x : i$ belong to an interoperation security constraint set, we will sometimes replace them with $x : i \prec y : j$.

Example 2. Returning to the security orderings of Figure 1 (cf. also Example 1), it may very well be the case that ts of ordering $(S_1, \leq_1)$ and s of ordering $(S_2, \leq_2)$ are identical, and so are **ts-sci1** and **ts-sci4**. In this case, we have *two* members in our interoperation security constraint set; these are:

$$\textbf{ts} : 1 = \textbf{s} : 2; \quad \textbf{ts-sci1}:1 = \textbf{ts-sci4}:2.$$

□

Given the security orderings $(S_1, \leq_1) \ldots (S_z, \leq_z)$, and a set of interoperation security constraints ISC, we have to find a new, global security ordering $(S, \preceq)$ and a family of functions $\psi_i : S_i \rightarrow S$ $(1 \leq i \leq z)$, translating the given security levels into the new ones,[3] such that:

(C1) the original orderings are preserved, i.e., for all x, y in any given S_i,

$$\psi_i(x) \preceq \psi_i(y) \leftrightarrow x \leq_i y;$$

[3] In other words, $\psi_i(x)$ is the new security level, corresponding to the given security level x of S_i.

(C2) the interoperation constraints are satisfied, that is,

$$(x : i \preceq y : j) \in \mathsf{ISC} \rightarrow \psi_i(x) \preceq \psi_j(y)\,,$$
$$(x : i \not\preceq y : j) \in \mathsf{ISC} \rightarrow \psi_i(x) \not\preceq \psi_j(y)\,.$$

Definition 2. A set of security orderings $\mathcal{H} = \{(S_1, \leq_1), \ldots, (S_z, \leq_z)\}$ is said to be *combinable* w.r.t. a set of interoperation security constraints, ISC, iff there exist a partially ordered set $(S, \preceq)$ and a family of *translation functions* ψ_i : $S_i \rightarrow S$ $(1 \leq i \leq z)$, such that conditions (C1) and (C2) hold. In this case, we say that $(S, \preceq), \psi_1, \ldots, \psi_z$ constitute a *witness* to the combinability of $\mathcal{H}$.

Example 3. Let us return to the security ordering of Figure 1 and the interoperation security constraints of Example 2. The partial ordering shown in Figure 2 constitutes a witness to the combinability of such orderings, together with the following translation functions: $\psi_1 : S_1 \rightarrow S$ is the identity function, while $\psi_2 : S_2 \rightarrow S$ is the map:

$$\mathbf{s} \mapsto \mathbf{ts}; \quad \mathbf{ts\text{-}sci3} \mapsto \mathbf{ts\text{-}sci3}; \quad \mathbf{ts\text{-}sci4} \mapsto \mathbf{ts\text{-}sci1}.$$

□

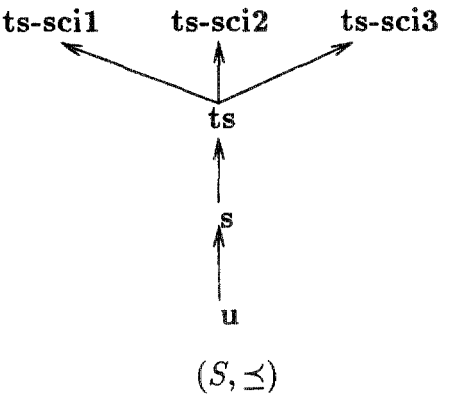

Fig. 2. A security ordering that merges $(S_1, \leq_1)$ and $(S_2, \leq_2)$

Example 4. Let us return to the security orderings of Figure 1 and consider the situation when the interoperability constraint set contains the statements:

$$\mathbf{s} : 1 = \mathbf{ts\text{-}sci3} : 2, \quad \mathbf{ts} : 1 = \mathbf{s} : 2.$$

These orderings are not combinable with respect to the above interoperation constraints. In fact, **ts:1** = **s:2** $\preceq$ **t-sci3:2** = **s:1**. This violates the principle of preservation of the given orderings, (C1), because **ts:1** is not smaller than **s:1** in the given ordering. □

Remark. Condition (C1) implies that each ψ_i must be *order-preserving* and *injective* (cf. [2]), that is, $x \leq_i y \rightarrow \psi_i(x) \preceq \psi_i(y)$, and $x \neq y \rightarrow \psi_i(x) \neq \psi_i(y)$. These conditions, however, are not equivalent to (C1) (essentially, the "only if" part of (C1) is not entailed, because every partial order can be linearized). We omit the details here, because they are not relevant in this context.

Given $(S_1, \leq_1), \ldots, (S_z, \leq_z)$ and ISC, there are several natural problems to solve. First, we have to check combinability. Then, if possible, a "conservative" witness should be computed; by "conservative", we mean that security levels belonging to different orderings should not be equated or made to satisfy an inequality in the merged ordering unless doing so is absolutely necessary to satisfy the combinability conditions (C1) and (C2). If the given orderings are not combinable w.r.t. ISC, then we need to find a minimal relaxation of the constraints that allows for combinability. In the following sections, we will show how we can solve these problems by exploiting logic programming techniques and standard algorithms on graphs.
But first, we show that conservative witnesses are essentially unique; therefore, it does not really matter which witness we choose, as far as it enjoys conservativity. For this purpose, we formalize the notion of conservativity.

Definition 3. A witness $(S^*, \preceq^*), \psi_1^*, \ldots, \psi_z^*$ of the combinability of $\mathcal{H}$ w.r.t. ISC is *maximally conservative* iff, for all other such witnesses $(S, \preceq), \psi_1, \ldots, \psi_z$,

$$\psi_i^*(x) \preceq^* \psi_j^*(y) \rightarrow \psi_i(x) \preceq \psi_j(y).$$

Besides conservativity, we are naturally interested in witnesses that do not contain "useless" security levels, in the sense that each level in the witness corresponds to some level in one of the given security orderings. These witnesses are called *parsimonious*.

Definition 4. Let $\mathcal{H} = \{ (S_1, \leq_1), \ldots, (S_z, \leq_z) \}$ be a set of security orderings. A witness $(S^*, \preceq^*), \psi_1^*, \ldots, \psi_z^*$ of the combinability of $\mathcal{H}$ is *parsimonious* iff for all $x^* \in S^*$, there exists $x : i$ such that $x^* = \psi_i^*(x)$.

In the next section, we will show that each combinable $\mathcal{H}$ has a maximally conservative and parsimonious witness (cf. Theorem 9 and Corollary 10). Here we prove that all such witnesses are isomorphic to each other, and hence, roughly speaking, they are essentially the same object.

Theorem 5. *Let $(S^*, \preceq^*), \psi_1^*, \ldots, \psi_z^*$ and $(S', \preceq'), \psi_1', \ldots, \psi_z'$ be witnesses of the combinability of $\mathcal{H}$ w.r.t.* ISC. *If they are parsimonious and maximally conservative, then they are isomorphic. Moreover, there is an isomorphism $f : S^* \rightarrow S'$ such that $f \circ \psi_i^* = \psi_i'$, that is, the diagram in Fig. 3 commutes.*

Proof. To show that the two witnesses are isomorphic, we have to prove that (i) there is a bijection $f : S^* \rightarrow S'$, and (ii) f preserves the orderings, that is, $x^* \preceq^* y^*$ iff $f(x^*) \preceq' f(y^*)$.

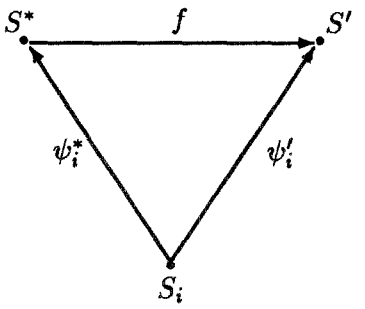

Fig. 3. This diagram commutes, for all $i = 1, \ldots, z$

(i) The function f is constructed as follows. For each element $x^* \in S^*$, consider the set $X = \{ x : i \mid \psi_i^*(x) = x^* \}$ (X is not empty because $(S^*, \preceq^*), \psi_1^*, \ldots, \psi_z^*$ is parsimonious), and define $f(x^*) = \psi_i'(x)$, for an arbitrary $x : i \in X$.

To see that f is well defined, we have to show that the choice of $x : i$ does not influence the value of f. For this purpose, it suffices to prove

(i') for all $x : i$, $y : j$ in X, $\psi_i'(x) = \psi_j'(y)$.

Note that, for all pairs of elements $x : i$, $y : j$ in X, $\psi_i^*(x) = x^* = \psi_j^*(y)$ holds; furthermore, $\psi_i^*(x) = \psi_j^*(y)$ iff $\psi_i^*(x) \preceq^* \psi_j^*(y)$ and $\psi_j^*(y) \preceq^* \psi_i^*(x)$, iff $\psi_i'(x) \preceq' \psi_j'(y)$ and $\psi_j'(y) \preceq' \psi_i'(x)$ (since the two witnesses are maximally conservative), iff $\psi_i'(x) = \psi_j'(y)$. This completes the proof that f is well defined.

Next we prove that f is a bijection. To see that f is surjective, note that for all $x' \in S'$, there exists $x : i$ such that $\psi_i'(x) = x'$ (because $(S', \preceq'), \psi_1', \ldots, \psi_z'$ is parsimonious); then, by definition of f, $f(\psi_i^*(x)) = x'$. To see that f is injective, assume that $f(x^*) = f(y^*)$; we have to show $x^* = y^*$. By definition of f, there exist $x : i$ and $y : j$ such that

$$\psi_i^*(x) = x^* \tag{1}$$

$$f(x^*) = \psi_i'(x) \tag{2}$$

$$\psi_j^*(y) = y^* \tag{3}$$

$$f(y^*) = \psi_j'(y) \tag{4}$$

From $f(x^*) = f(y^*)$, (2) and (4), it follows that $\psi_i'(x) = \psi_j'(y)$; this implies $\psi_i^*(x) = \psi_j^*(y)$, because the two witnesses are maximally conservative (see the proof of (i') above). From this fact, (1) and (3), derive $x^* = y^*$. This completes the proof that f is injective, and the proof that f is a bijection.

(ii) Consider two elements $x : i$, and $y : j$ such that $\psi_i^*(x) = x^*$ and $\psi_j^*(y) = y^*$ (they exist because $(S^*, \preceq^*), \psi_1^*, \ldots, \psi_z^*$ is parsimonious). Clearly, $x^* \preceq^* y^*$ iff $\psi_i^*(x) \preceq^* \psi_j^*(y)$, iff $\psi_i'(x) \preceq' \psi_j'(y)$ (maximal conservativity of the two witnesses) iff $f(x^*) \preceq' f(y^*)$ (because $f(x^*) = \psi_i'(x)$ and $f(y^*) = \psi_j'(y)$, by definition of f).

Finally, note that f satisfies $f \circ \psi_i^* = \psi_i'$ by definition. □

Remark. If a witness $\mathcal{W}$ is maximally conservative but not parsimonious, then it contains a parsimonious, maximally conservative witness, which coincides with the restriction of $\mathcal{W}$ to the range of the translation functions. The proof is easy and is omitted here.

3 Logic Programming Approach

In this section we prove that the combinability problem can be solved in polynomial time by exploiting a suitable logic program. In the rest of the paper, without loss of generality, we will assume that each ordering $(S_i, \leq_i)$ is represented as a finite Hasse diagram (as shown in Figure 1).
The combinability problem can be represented in a natural way through a logic program P, that encodes (a) the given security orderings, (b) the standard properties that must be satisfied by the ordering $\preceq$ of the witness, and (c) the interoperation constraints ISC.

Definition 6. The Herbrand Universe U_P (i.e. the set of ground terms) of the program P consists of all the terms $x : i$ where $x \in S_i$, $(1 \leq i \leq z)$. The rules of P comprise:

1. the axiomatization of the reflexive, antisymmetric and transitive properties:
   $X \leq X$
   $X \approx Y \leftarrow X \leq Y, Y \leq X$
   $X \leq Y \leftarrow X \leq Z, Z \leq Y$
   Note that above, $\leq$ and $\approx$ are binary predicate symbols.
2. the facts
   $x : i \leq y : i$
   such that (x,y) is a link in the Hasse diagram of S_i (principle of autonomy);
3. the facts
   $x : i \leq y : j.$
   such that $(x : i \preceq y : j)$ is a constraint of ISC;
4. the rules
   $non\_combinable \leftarrow x : i \leq y : i$
   whenever x is not smaller than y in S_i (principle of security);
5. the rules
   $non\_combinable \leftarrow x : i \leq y : j$
   such that $(x : i \not\preceq y : j)$ is a constraint of ISC.

Remark. The relation $\approx$ in P is an *equivalence relation.*

Remark. For all given security orderings $(S_i, \leq_i)$,

$$x \leq_i y \text{ implies } P \models x : i \leq y : i .$$

Example 5. Suppose we return to the two partial orderings of Fig. 1. The logic program P associated with the problem of combining these two security orderings is given in Fig. 4. Note that rule (a) is redundant, because if $\mathbf{ts} : \mathbf{1} \leq \mathbf{u} : \mathbf{1}$ then,

$$
\begin{array}{rl}
X \leq X \leftarrow & \\
X \leq X \leftarrow & X \leq Z, Z \leq Y \\
X \approx X \leftarrow & X \leq Y, Y \leq X \\
\mathbf{u} : \mathbf{1} \leq \mathbf{s} : \mathbf{1} \leftarrow & \\
\mathbf{s} : \mathbf{1} \leq \mathbf{ts} : \mathbf{1} \leftarrow & \\
\mathbf{ts} : \mathbf{1} \leq \mathbf{ts\_sci1} : \mathbf{1} \leftarrow & \\
\mathbf{ts} : \mathbf{1} \leq \mathbf{ts\_sci2} : \mathbf{1} \leftarrow & \\
\mathbf{s} : \mathbf{2} \leq \mathbf{ts\_sci3} : \mathbf{2} \leftarrow & \\
\mathbf{s} : \mathbf{2} \leq \mathbf{ts\_sci4} : \mathbf{2} \leftarrow & \\
\mathbf{ts} : \mathbf{1} \leq \mathbf{s} : \mathbf{2} \leftarrow & \\
\mathbf{s} : \mathbf{2} \leq \mathbf{ts} : \mathbf{1} \leftarrow & \\
\mathbf{ts\_sci1} : \mathbf{1} \leq \mathbf{ts\_sci4} : \mathbf{2} \leftarrow & \\
\mathbf{ts\_sci4} : \mathbf{2} \leq \mathbf{ts\_sci1} : \mathbf{1} \leftarrow & \\
\mathit{non\_combinable} \leftarrow & \mathbf{s} : \mathbf{1} \leq \mathbf{u} : \mathbf{1} \\
\mathit{non\_combinable} \leftarrow & \mathbf{ts} : \mathbf{1} \leq \mathbf{u} : \mathbf{1} \qquad (a) \\
\mathit{non\_combinable} \leftarrow & \mathbf{ts} : \mathbf{1} \leq \mathbf{s} : \mathbf{1} \qquad (b) \\
\mathit{non\_combinable} \leftarrow & \mathbf{ts\_sci1} : \mathbf{1} \leq \mathbf{u} : \mathbf{1} \qquad (*) \\
\mathit{non\_combinable} \leftarrow & \mathbf{ts\_sci1} : \mathbf{1} \leq \mathbf{s} : \mathbf{1} \qquad (*) \\
\mathit{non\_combinable} \leftarrow & \mathbf{ts\_sci1} : \mathbf{1} \leq \mathbf{ts} : \mathbf{1} \\
\mathit{non\_combinable} \leftarrow & \mathbf{ts\_sci1} : \mathbf{1} \leq \mathbf{ts\_sci2} : \mathbf{1} \\
\mathit{non\_combinable} \leftarrow & \mathbf{ts\_sci2} : \mathbf{1} \leq \mathbf{u} : \mathbf{1} \qquad (*) \\
\mathit{non\_combinable} \leftarrow & \mathbf{ts\_sci2} : \mathbf{1} \leq \mathbf{s} : \mathbf{1} \qquad (*) \\
\mathit{non\_combinable} \leftarrow & \mathbf{ts\_sci2} : \mathbf{1} \leq \mathbf{ts} : \mathbf{1} \\
\mathit{non\_combinable} \leftarrow & \mathbf{ts\_sci2} : \mathbf{1} \leq \mathbf{ts\_sci1} : \mathbf{1} \\
\mathit{non\_combinable} \leftarrow & \mathbf{ts\_sci3} : \mathbf{2} \leq \mathbf{s} : \mathbf{2} \\
\mathit{non\_combinable} \leftarrow & \mathbf{ts\_sci4} : \mathbf{2} \leq \mathbf{s} : \mathbf{2} \\
\mathit{non\_combinable} \leftarrow & \mathbf{ts\_sci3} : \mathbf{2} \leq \mathbf{ts\_sci4} : \mathbf{2} \\
\mathit{non\_combinable} \leftarrow & \mathbf{ts\_sci4} : \mathbf{2} \leq \mathbf{ts\_sci3} : \mathbf{2}
\end{array}
$$

Fig. 4. The program P for the orderings in Fig. 1

by transitivity, $\mathbf{ts} : \mathbf{1} \leq \mathbf{s} : \mathbf{1}$, and hence *non_combinable* can be derived through (b). Similarly, the rules marked with $(*)$ can be eliminated. In general, among the lower bounds of a security level, only the maximal ones need to be considered.

The following lemma shows that the above logic program is a sound formalization of the combinability problem, in the sense that the relations which are entailed by P must hold in every possible witness[4].

Lemma 7. *For all witnesses* $(S, \preceq), \psi_1, \ldots, \psi_z$ *of the combinability of* $\mathcal{H}$,

$$P \models x : i \leq y : j \text{ implies } \psi_i(x) \preceq \psi_j(y) .$$

Proof. If $x : i = y : j$, then the property holds trivially. Otherwise, if $P \models x : i \leq y : j$, then $(x : i \leq y : j) \in T_P^\omega$, since P is a Horn program, and therefore there exists n such that $(x : i \leq y : j) \in T_P^n - T_P^{n-1}$.

By induction on n: if $x : i \leq y : j \in T_P^1$, then $x : i \leq y : j \in P$, which means that either $(x : i \preceq y : j) \in \mathsf{ISC}$, and then $\psi_i(x) \preceq \psi_j(y)$ by definition of witness, or $i = j$, and $x \leq_i y$, and then $\psi_i(x) \preceq \psi_i(y)$, by definition of witness . Assume that the lemma holds for all the facts $z : l \leq t : k \in T_P^n$, and consider $(x : i \leq y : j) \in T_P^{n+1}$. Since $(x : i \leq y : j) \in T_P^{n+1}$, there must be a ground instance $x : i \leq y : j \leftarrow x : i \leq z : l, z : l \leq y : j$ of a clause in P such that both $x : i \leq z : l$ and $z : l \leq y : j$ belong to T_P^n. Therefore, both $x : i \leq z : l$ and $z : l \leq y : j$ satisfy the inductive hypothesis: $\psi_i(x) \preceq \psi_l(z)$ and $\psi_l(z) \preceq \psi_j(y)$ From the transitivity of $\preceq$, $\psi_i(x) \preceq \psi_j(y)$. □

Intuitively, if the given orderings are combinable, we can build a witness by querying the program P. The new security levels are obtained by collapsing all the pairs $x : i$ and $y : j$ such that P entails $x : i \approx y : j$. Technically speaking, this is obtained by taking the quotient $(U_P / \approx)$ as the witness domain. Similarly, the ordering is obtained by querying P, as specified below.

Definition 8. The *P-canonical security ordering* associated with $\mathcal{H}$ is $(S^*, \preceq^*)$, where

- S^* is the quotient set $(U_P / \approx)$;
- $X \preceq^* Y$ holds iff $P \models x : i \leq y : j$, for some $x : i \in X$ and $y : j \in Y$.

The translation functions are defined as $\psi_i^*(x) = [x : i]$, where $[x : i]$ denotes the equivalence class of $x : i$.

Theorem 9. $\mathcal{H}$ *is combinable iff* P *does not entail non_combinable* .

Proof. First we prove the "only if" part. Let $(S, \preceq), \psi_1, \ldots, \psi_z$ be a witness of the combinability of $\mathcal{H}$. Assume that $P \models$ *non_combinable*. This means that there exists n such that *non_combinable* $\in T_P^n - T_P^{n-1}$. Therefore, either $x : i \leq$

[4] The proof of this lemma, as well as proofs of some others, require the use of an operator called T_P well known in logic programming [11] that may be associated with a logic program P. Due to space restrictions, we briefly state the definition here. T_P takes as input, a set I of ground atoms. $T_P(I)$ is the set $\{A \mid A \leftarrow B_1, \ldots, B_n$ is a ground instance of a rule in P and $\{B_1, \ldots, B_n\} \subseteq I\}$. T_P may be iteratively applied as follows. $T_P^0 = \emptyset$; $T_P^{i+1} = T_P(T_P^i)$. $T_P^\omega = \bigcup_{i \geq 0} T_P^i$. It is well known [11] that T_P^ω is identical to the set of all ground atoms that are logically entailed by P.

$y : i \in T_P^{n-1}$, and $x \not\leq_i y$, or $x : i \leq y : j \in T_P^{n-1}$, and $(x : i \not\preceq y : i) \in \mathsf{ISC}$. In the first case $P \models x : i \leq y : i$, therefore, by Lemma 7, $\psi_i(x) \preceq \psi_i(y)$ and, by definition of witness $x \leq_i y$; a contradiction. Consider now the second case. $P \models x : i \leq y : j$, and therefore, by Lemma 7, $\psi_i(x) \preceq \psi_j(y)$. This contradicts the hypothesis $(x : i \not\preceq y : j) \in \mathsf{ISC}$, since, by definition of witness, it would imply $\psi_i(x) \not\preceq \psi_j(y)$. This means that also the second case is impossible, and therefore it cannot be the case that $P \models$ *non_combinable*.

We are left to prove that if $P \not\models$ *non_combinable*, then $\mathcal{H}$ is combinable. For this purpose, we show that the canonical security ordering $(S^*, \preceq^*)$, and the canonical translations $\psi_1^*, \ldots, \psi_z^*$, constitute a witness of its combinability.

Condition (C1) is satisfied. Indeed, assume $\psi_i^*(x) \preceq^* \psi_i^*(y)$, i.e., $[x : i] \preceq [y : i]$. Then, by definition, $P \models x : i \leq y : i$, and therefore it must be $x \leq_i y$ (otherwise, P would entail *non_combinable*). On the other hand, if $x \leq_i y$, then $P \models x : i \leq y : i$ (Remark 3), and then, by definition, $[x : i] \preceq^* [y : i]$, i.e., $\psi_i^*(x) \preceq \psi_i^*(y)$.

Condition (C2) is satisfied. Indeed, if $(x : i \preceq y : j) \in \mathsf{ISC}$, then the fact $x : i \leq y : j$ belongs to the program P. By definition, $P \models x : i \leq y : j$ implies $[x : i] \preceq [y : j]$, i.e., $\psi_i^*(x) \preceq \psi_j^*(y)$.

On the other hand, if $(x : i \not\preceq y : j) \in \mathsf{ISC}$, then $P \not\models x : i \leq y : j$, otherwise P would entail *non_combinable*. From $P \not\models x : i \leq y : j$ it follows, by definition, $[x : i] \not\preceq [y : j]$, and then $\psi_i^*(x) \not\preceq^* \psi_j^*(y)$. □

As a corollary of Theorem 9 and Lemma 7, we derive that the canonical security ordering and the canonical translations integrate the given security orderings $(S_i, \leq_i)$ by introducing the least possible number of dependencies, thereby achieving a maximally conservative composition.

Corollary 10. *If $\mathcal{H}$ is combinable then $(S^*, \preceq^*), \psi_1^*, \ldots, \psi_z^*$ is a maximally conservative and parsimonious witness.*

Since deduction from Horn clauses can be done in polynomial time [5], also the combinability check and the construction of a witness can be performed in polynomial time.

Theorem 11. *Let $n = \sum_{i=1}^{z} |S_i|$. Checking combinability and building the canonical ordering with the LP method can be done in time and space $O(n^3)$, that is, the size of the ground instantiation of the program.*

Proof. To check that $\mathcal{H}$ is combinable, it suffices to

- build the ground instance of the program P associated with $\mathcal{H}$, say P_{ground}. This can be done in time and space $O(n^3)$, since the axiomatization of the transitive property is the only rule in P in which three variables occur (the other rules contain at most two variables);
- compute the least model of P_{ground}. P_{ground} is a ground Horn program, and its least Herbrand model can be found in time and space linear in the size of P_{ground} [5], that is, $O(n^3)$;

- verify that *non_combinable* does not belong to the least model of P_{ground}. The size of the least model is linear in the size of the Herbrand Universe of P, that is, $O(n)$; checking if *non_combinable* belongs to such a model can be done in $O(n)$.

The total cost is therefore $O(n^3)$. □

The canonical ordering can be computed by a naive algorithm in time $O(n^4)$, by repeatedly scanning the least model of P (whose size is $O(n^2)$, and which can be computed in time $O(n^3)$, cf. the proof of the above theorem). The naive algorithm can be improved, but the cost is unlikely to drop below $O(n^3)$, as far as general query methods for function-free logic programs are used. In fact, the best general query methods have worst case complexity linear in the size of the ground instantiation of the program, that is $O(n^3)$ for P. We omit the details here, because these complexity bounds can be improved by adopting a different, more efficient technique, based on standard graph algorithms.

4 Composing Security Orderings Through Graph Algorithms

In this section we reformulate the notion of canonical security ordering in terms of the graph defined below. The final goal is obtaining more efficient algorithms for solving the various problems related to combinability.

Definition 12. The *graph associated with* $\mathcal{H} = \{(S_1, \leq_1), \ldots, (S_z, \leq_z)\}$ and ISC is $G = (V, E)$, where V (the set of vertices) is the set of all $x : i$ such that $x \in S_i$ $(1 \leq i \leq z)$, and where E (the set of edges) is the set of all ordered pairs $(x : i, y : j)$ such that either

(1) $i = j$ and there is an edge from x to y in the Hasse diagram of S_i, or
(2) $i \neq j$ and $(x : i \preceq y : j) \in \mathsf{ISC}$.

The basic idea is that if two security levels belong to a cycle in G, then the two levels must be identified to satisfy (C1) and (C2). The rest of the section expands on this idea.

Definition 13. The *G-canonical security ordering* associated to $\mathcal{H}$ is $(S^+, \preceq^+)$, where S^+ is the set of strongly connected components[5] of the graph G associated to $\mathcal{H}$, and $x \preceq^+ y$ holds iff $x = y$ or there is a directed path in G from a member of x to a member of y. The *canonical translations* ψ_i^+ are defined by

$$\psi_i^+(x) = [x : i],$$

where $[x : i]$ denotes the strongly connected component containing $x : i$.

[5] Here, with a slight abuse of notation, we identify the strongly connected components of a directed graph with the maximal sets C of vertices such that each member of C can be reached from any other member of C through a directed path.

The following is a soundness lemma, stating that the dependencies that hold in the canonical security ordering must hold also in every witness.

Lemma 14. *Let $(S^+, \preceq^+)$ be the canonical security ordering of $\mathcal{H}$. Then, for all witnesses $(S, \preceq), \psi_1, \ldots, \psi_z$ of the combinability of $\mathcal{H}$,*

$$[x : i] \preceq^+ [y : j] \rightarrow \psi_i(x) \preceq \psi_j(y) .$$

Proof. Assume that $[x : i] \preceq^+ [y : j]$. By definition, it follows that either $x : i = y : j$ or there must be a path from $x : i$ to $y : j$ in G. The first case is trivial, so we focus on the latter. Let the path be

$$x : i = x_1 : i_1,\ x_2 : i_2,\ \ldots,\ x_n : i_n = y : j .$$

By definition of the edges of G, we have that for all $k = 1, \ldots, n-1$ one of the following conditions holds:

(1) $i_k = i_{k+1}$ and $x_k \leq_{i_k} x_{k+1}$. In this case, by (C1), we have $\psi_{i_k}(x_k) \preceq \psi_{i_k}(x_{k+1}) = \psi_{i_{k+1}}(x_{k+1})$.
(2) $i_k \neq i_{k+1}$ and $(x_k : i_k \preceq x_{k+1} : i_{k+1}) \in \mathsf{ISC}$. Also in this case, we have $\psi_{i_k}(x_k) \preceq \psi_{i_{k+1}}(x_{k+1})$, by (C2).

It follows that $\psi_i(x) = \psi_{i_1}(x_1) \preceq \ldots \preceq \psi_{i_n}(x_n) = \psi_j(y)$, so, by transitivity, $\psi_i(x) \preceq \psi_j(y)$. □

This result, however, does not imply that the canonical ordering and the canonical translations constitute a witness, nor that $\mathcal{H}$ is combinable. For this purpose we prove the next theorems.

Theorem 15. *$\mathcal{H}$ is combinable iff the corresponding canonical ordering $(S^+, \preceq^+)$ and the canonical translations $\psi_1^+, \ldots, \psi_z^+$, constitute a witness to the combinability of $\mathcal{H}$.*

Proof. The "if" part is trivial. To prove the "only if" part, assume that $\mathcal{H}$ is combinable, and let $(S, \preceq), \psi_1, \ldots, \psi_z$ be a witness to the combinability of $\mathcal{H}$. We have to show that $(S^+, \preceq^+)$ and the canonical translations constitute a witness, i.e., that they satisfy the conditions (C1) and (C2). To prove the "if" part of (C1), assume $x \leq_i y$; then there is a path from x to y in the Hasse diagram of the security ordering S_i; by definition of G, there is a corresponding path from $x : i$ to $y : i$ in G; by definition of canonical ordering, it follows that $[x : i] \preceq^+ [y : i]$; then, by definition of ψ_i^+, $\psi_i^+(x) \preceq^+ \psi_i^+(y)$. This proves the "if" part of (C1). To prove the "only if part", assume that $\psi_i^+(x) \preceq^+ \psi_i^+(y)$; then, by definition of ψ_i^+, $[x : i] \preceq^+ [y : i]$; by Lemma 14, it follows that $\psi_i(x) \preceq \psi_i(y)$, and hence, by (C1), $x \leq_i y$. This completes the proof that (C1) holds.

We are left to prove that also (C2) is satisfied. For each constraint $(x : i \preceq y : j) \in \mathsf{ISC}$, G contains an edge from $x : i$ to $y : j$; therefore $[x : i] \preceq^+ [y : j]$, and hence $\psi_i^+(x) \preceq^+ \psi_j^+(y)$. This proves that all positive constraints are satisfied. Now assume that some negative constraint $(x : i \not\preceq y : j) \in \mathsf{ISC}$ is not satisfied by the canonical ordering, that is, $\psi_i^+(x) \preceq^+ \psi_j^+(y)$. This is equivalent to $[x : i] \preceq^+ [y : j]$; by Lemma 14, it follows that $\psi_i(x) \preceq \psi_j(y)$. But then, the witness $(S, \preceq), \psi_1, \ldots, \psi_n$ would violate (C2), which is absurd. □

As a corollary of the above results, we derive that the new notion of canonical security ordering constitutes a maximally conservative witness.

Corollary 16. *If $\mathcal{H}$ is combinable then $(S^+, \preceq^+), \psi_1^+, \ldots, \psi_z^+$ is a maximally conservative and parsimonious witness.*

Therefore, by Theorem 5, the G-canonical witness is isomorphic to the P-canonical witness, i.e. the two approaches are equivalent, from a semantic point of view. On the contrary, there are some differences in efficiency. All the basic problems related to combinability outlined in the previous sections can be solved in polynomial time, with some improvements w.r.t. the logic programming approach.

Theorem 17. *Let n be the number of vertices of G, that is, $n = \sum_{i=1}^{z} |S_i|$.*

(i) *Deciding combinability can be done in time $O(n^2 \log n)$.*
(ii) *The canonical ordering $(S^+, \preceq^+)$ can be computed in time $O(n^2)$.*
(iii) *A maximal satisfiable subset of* ISC *can be computed in time $O(n^4 \log n)$.*

Proof. (*i*) To check that $\mathcal{H}$ is combinable, it suffices to verify that the canonical ordering and the canonical translations constitute a witness. This can be done through the following steps:

(a) Construct the associated graph G (time: $O(|G|)$).
(b) Compute the strongly connected components of G (time: $O(|G|)$, cf. [13]).
(c) Verify condition (C1). This can be done by constructing the set $L_i^+ = \{ (x, y) \mid [x : i] \preceq^+ [y : i] \}$ and checking that it coincides with the set $L_i = \{ (x, y) \mid x \leq_i y \}$ $(1 \leq i \leq z)$. This can be done as follows:
  (c1) Construct L_i^+ and L_i as lists (time: $O(n^2)$).
  (c2) Sort L_i^+ and L_i under lexicographic order (time: $O(n^2 \log n^2)$).
  (c3) Verify that $L_i^+ = L_i$ (time: $O(n^2)$).
  Since the cost of (c2) dominates the cost of (c1) and (c3), the total cost of step (c) is $O(n^2 \log n^2)$.
(d) Verify condition (C2). Clearly, the positive constraints of ISC are satisfied by construction (cf. proof of Theorem 15), so we only have to check the negative constraints. This can be done by
  (d1) constructing the set $N = \{ (x : i, y : j) \mid (x : i \not\preceq y : j) \in \text{ISC} \}$ and the set $L^+ = \{ (x : i, y : j) \mid [x : i] \preceq^+ [y : j] \}$ $(1 \leq i \leq z)$; (time: $O(n^2)$);
  (d2) verifying that the intersection of L^+ and N is empty; this can be done in time $O(n^2 \log n^2)$ by sorting L^+ and N and then visiting them one time in parallel, in ascending order.

The total cost of step (d) is $O(n^2 \log n^2)$. Now, since the size of $|G|$ grows as n^2, in the worst case, we have that the cost of (a) and (b) is dominated by the cost of (c) and (d); therefore, the overall complexity of this algorithm is $O(n^2 \log n^2)$. Moreover, $n^2 \log n^2 = 2n^2 \log n$, and hence the asymptotic time complexity is $O(n^2 \log n)$.

(*ii*) Note that the canonical ordering can be constructed by performing steps (a) and (b), and by computing the set L^+ (cf. step (d1)) which yields the relation

$\preceq^+$. It follows immediately that the canonical ordering can be constructed in time $O(n^2)$.

(iii) Let $\chi_1, \ldots, \chi_k$ be any enumeration of ISC. For $i = 1, \ldots, k$ define

$$\mathsf{ISC}_0 = \emptyset,$$
$$\mathsf{ISC}_i = \begin{cases} \mathsf{ISC}_i & \text{if } \mathcal{H} \text{ is not combinable w.r.t. } \mathsf{ISC}_i \cup \{\chi_i\} \\ \mathsf{ISC}_i \cup \{\chi_i\} & \text{otherwise.} \end{cases}$$

Clearly, $\mathcal{H}$ is combinable w.r.t. ISC_0 (the given orderings remain unrelated), therefore, by construction, ISC_k is a maximal satisfiable subset of ISC. By *(i)*, each iteration can be computed in time $O(n^2 \log n)$; moreover, $k \leq n^2$, therefore ISC_k can be computed in time $O(n^4 \log n)$. □

Remark. Point *(iii)* considers the worst case in which the size of ISC grows as n^2. However, in real applications, it seems reasonable to assume that the set of constraints is sparse. Clearly, if we assume that the cardinality of ISC is proportional to n, then the algorithm for computing a maximal satisfiable subset of ISC runs in time $O(n^3 \log n)$.

5 Related Work

Gong and Qian [6] studied the complexity of secure interoperation in a framework that is similar to ours in several respects. Their principles of autonomy and security are equivalent to our combinability condition (C1). The major difference between the two approaches lies in the treatment of negative constraints (*restricted access* relation in [6]). In the extended version of their paper, Gong and Qian note that a negative constraint can be violated by the transitive closure of the permitted accesses (permitted by the individual secure systems, or by the positive constraints, which they call *permitted access relation*). On the contrary, in our approach, negative constraints are satisfied by the transitive closure as well. Our combinability checking algorithm (which runs in time $O(n^2 \log n)$), is slightly more efficient than their security evaluation algorithm, whose complexity is $O(n^3)$. By using our methods for comparing sets of edges (cf. steps (c) and (d) in the proof of Theorem 17), the complexity of their algorithm can be reduced to $O(n^2 \log n)$, as well. When the interoperation constraints are not satisfiable, Gong and Qian consider appealing forms of constraint relaxation (e.g. which maximize the cardinality of the satisfiable subset) which we do not consider here; they show intractability results and characterize tractable subclasses for these problems.

Jones and Winslett [8] consider *role based* secure interoperation (as opposed to the clearance level approach) in object-oriented databases. In that framework, roles with identical attributes are identified; on the contrary, in the approaches based on security levels, different levels should be identified only if the interoperation constraints force them to coincide. In [8] no negative constraints are considered.

Acknowledgements. We thank Xiaolei Qian for her comments concerning the relationship between our algorithm and that of Gong and Qian.

References

1. Y. Arens, C.Y.Chee, C.N. Hsu and C. Knoblock. Retrieving and Integrating Data from Multiple Information Sources, *Intl. J. of Intelligent Cooperative Info. Systems*, 2, 2, pp. 127–158, 1994.
2. G. Birkhoff. *Lattice Theory*, American Math. Society, Providence, 1967.
3. K.S. Candan, S. Jajodia and V.S. Subrahmanian. Secure Mediated Databases, to appear in:*Proc. 1996 IEEE Conf. on Data Engineering.*
4. S. Chawathe, H. Garcia-Molina, J. Hammer, K. Ireland, Y. Papakonstantinou, J. Ullman and J. Widom. (1994) *The TSIMMIS Project: Integration of Heterogeneous Information Sources*, Proc. IPSJ Conf., Tokyo, Japan, Oct. 1994.
5. W.F. Dowling, J.H. Gallier. Linear-time algorithms for testing satisfiability of propositional Horn formulae. *Journal of Logic Programming*, 3:267-284, (1984).
6. L. Gong and X. Qian. (1996) Computational Issues in Secure Interoperation, *IEEE Trans. on Software Engineering*, 22, 1, pp. 43–52.
7. N. B. Idris, W. A. Gray and R. F. Churchhouse, Providing Dynamic Security Control in a Federated Database, *Proc. 1994 Intl. Conf. on Very Large Databases*, pp. 13–23.
8. V.E. Jones and M. Winslett. (1993) Secure Database Interoperation via Role Translation, in "Security for Object Oriented Systems (eds. B. Thuraisingham, R. Sandhu amd T.C. Ting), Springer Verlag.
9. D. Jonscher and K. R. Diittrich, An approach for building secure database federations, *Proc. 20th VLDB Conf.*, 1994.
10. Laks V.S. Lakshmanan, F. Sadri and I.N. Subramanian, On the logical foundations of schema integration and evolution in Heterogeneous Database Systems, *Proc. DOOD-93*, Phoenix, Arizona, 1993.
11. J.W. Lloyd. (1987) *Foundations of Logic Programming*, Springer.
12. V.S. Subrahmanian, et al. (1995) *HERMES: A Heterogeneous Reasoning and Mediator System*, submitted for publication.
13. R. Tarjan. Depth-first search and linear graph algorithms. *SIAM J. of Computing*, 1(2):146-160 (1972)
14. G. Wiederhold, Mediators in the Architecture of Future Information Systems, *IEEE Computer*, pp. 38–49, March 1992.
15. G. Wiederhold, Intelligent Integration of Information, *Proceedings of the ACM Conference on Management of Data*, pp. 434–437, 1993.