Access control with prohibitions and obligations

Philippe Balbiani, Fatme Harb, Ali Kaafarani

Access control with prohibitions and obligations

Philippe Balbiani
Université Paul Sabatier, Irit-CNRS
31062 Toulouse Cedex 9, France

Fatima Harb
Université Paul Sabatier, Irit-CNRS
31062 Toulouse Cedex 9, France

Ali Kaafarani
Université Paul Sabatier, Irit-CNRS
31062 Toulouse Cedex 9, France

Abstract

Most of access control mechanisms use the matrix model to represent protection states of computer systems. Firstly, we present a variant of the access control matrix model obtained by incorporating explicit prohibitions saying, e.g., that "it is not permitted that subject s performs action a on object o". Secondly, we present a variant of the access control matrix model obtained by incorporating explicit obligations saying, e.g., that "it is obligatory that subject s performs action a on object o". We then turn to the question whether the expressive power of the matrix model grows when enriching access control with explicit prohibitions or explicit obligations. In connection with these enriched models, we also discuss the solvable and unsolvable cases of one of the major themes of computer security, namely the classical safety problem for access control matrices.

1 Introduction

The matrix model was first defined by Lampson [13]. It is based on the idea of associating with each subject s and each object o of a protection system a set $M[s, o]$ of actions, the relationship $a \in M[s, o]$ being read "subject s has permission to perform action a on object o". Most of access control mechanisms use the matrix model to represent protection states of computer systems. We shall focus our attention on the HRU model introduced by Harrison, Ruzzo, and Ullman [12] because this model summarizes in the majority of cases the design features of such-and-such protection system. Within the HRU model, protection systems consist of commands. As subjects execute commands, the protection state of the system, i.e. its access control matrix, changes. The original classical safety problem for the HRU model can be stated in the following way: given a protection system Π, an action a, and a protection state Q, determine if there is at least one protection state Q' containing a and reachable by Π in a finite number of steps from Q. The safety question is undecidable for generic protection systems but it becomes decidable if protection systems are restricted in some way. Can the borderline between solvable and unsolvable cases of the safety problem be drawn sharply and on the basis of which criteria? This matter is analysed in a number of books [5; 6; 15] and papers [10; 11; 12] partly or entirely devoted to HRU.

Additional topics related to the HRU model include results concerning a number of interesting variants obtained by extending HRU in various ways. Revisiting the results obtained so far, Sandhu [16] and Soshi [19] expanded the HRU model by typing subjects and objects. The papers [7; 9; 17] formulated a model for access control within which the permission for a subject to have legal access to an object depends both on the roles assigned to the subject and on the permissions allocated to the object; in this connection see [8]. Role-based access control has recently attracted a great deal of attention. However, nothing is known about role-based protection systems for which the safety problem is decidable. An interesting extensions of HRU is HRU with explicit prohibitions saying, e.g., that "it is not permitted that subject s performs action a on object o". The essential ingredients of this variant of the HRU model have been introduced by Sandhu and Ganta [18]. Nevertheless, nothing is known about protection systems with explicit prohibitions for which the safety problem is decidable. The description of HRU-like models incorporating explicit obligations has been suggested in several places including [2]. The safety problem for access control matrix models allowing obligations saying, e.g., that "it is obligatory that subject s performs action a on object o" has never been considered.

The bulk of this paper is devoted to the problem of trying to characterize the borderline between decidable and undecidable cases of the safety problem for HRU with explicit prohibitions or explicit obligations. On the one hand, when we add explicit prohibitions, decidability of safety remains open for mono-operational protection systems and monoconditional monotonic protection systems. Hence, we present various fragments of HRU with explicit prohibitions mainly defined in terms of additional restrictions on the syntactic structure of commands. On the other hand, we demonstrate that safety for HRU with explicit obligations becomes decidable for mono-operational protection systems and monoconditional monotonic protection systems. We follow here the approach developed by Harrison, Ruzzo, and Ullman [10; 11; 12] for mono-operational HRU protection systems and monoconditional monotonic HRU protection systems. Let us review briefly the contents of the paper. The main part of section 2 is devoted to the original classical safety problem for the HRU model which we explain from scratch. In sections 3 and 4, we present our variants of the access control matrix model obtained by incorporating explicit prohibitions or explicit obligations. We conclude the paper with a number of open matters.

2 Access control matrix model

A typical feature of the model introduced by Harrison, Ruzzo, and Ullman [12] is that protection systems are ordered pairs $\Pi = (A, C)$ where A is a finite set of actions and C is a finite set of commands.

Example The actions of a protection system correspond, for instance, to those of the Unix system: $read$, $write$, etc.

Commands are expressions of the form:

command $\alpha(X_1, \ldots, X_i, Y_1, \ldots, Y_j)$ is
$(a_1, X_{u_1}, Y_{v_1}) \ldots (a_k, X_{u_k}, Y_{v_k}) \Rightarrow \pi_1; \ldots \pi_n;$

where X_1, ..., X_i are variables of type subject, Y_1, ..., Y_j are variables of type object, a_1, ..., a_k denote actions in A, u_1, ..., u_k are integers between 1 and i, v_1, ..., v_k are integers between 1 and j, and π_1, ..., π_n are atomic programs, i.e. expressions of the form "enter a into (X, Y)", "create subject X", "create object Y", "delete a from (X, Y)", "destroy subject X", and "destroy object Y". The command $\alpha(X_1, \ldots, X_i, Y_1, \ldots, Y_j)$ denotes the conditional:

if "a_1 is in $M[X_{u_1}, Y_{v_1}]$" ... "a_k is in $M[X_{u_k}, Y_{v_k}]$"
then begin $\pi_1; \ldots \pi_n$; end

The number of its elementary conditions is k, a non-negative integer, and the number of its atomic programs is n, a positive integer. Commands are used to update protection states, i.e. ordered triples $Q = (S, O, M)$ where S is a finite set of subjects, O is a finite set of objects, and M is a function assigning to each subject s in S and each object o in O a finite set $M[s, o]$ of actions. For technical reasons, we assume that S and O have no common elements. Q's subjects are active entities, such as human beings, whereas Q's objects are passive entities, such as files, the relationship $a \in M[s, o]$ being read "subject s has permission to perform action a on object o".

Q	o_0	o_1
s_0	a_1	a_0, a_1
s_1	a_0, a_1	a_1
s_2	a_1	a_1

Q'	o_0	o_1	o_2
s_0	a_1	a_0, a_1	
s_1	a_0, a_1	a_1	
s_2	a_1	a_1	a_0, a_1

Q''	o_0	o_1	o_2
s_0	a_1	a_0, a_1	a_2
s_1	a_0, a_1	a_1	
s_2	a_1	a_1	a_0, a_1

Q'''	o_0	o_1	o_2
s_0	a_1	a_0, a_1	a_2, a_3
s_1	a_0, a_1	a_1	
s_2	a_1	a_1	a_0, a_1

Table 1. Protection states Q, Q', Q'', and Q'''.

Example Table 1 illustrates protection states presented in a matrix form. The entries in the matrices specify the actions that each subject has permission to perform on each object.

If all variables in α are replaced by names of concrete entities, that is, subjects s_1, ..., s_i for variables X_1, ..., X_i and objects o_1, ..., o_j for variables Y_1, ..., Y_j, then we shall say that protection state $Q = (S, O, M)$ makes command $\alpha(s_1, \ldots, s_i, o_1, \ldots, o_j)$ possible iff $a_1 \in M[s_{u_1}, o_{v_1}]$, ..., $a_k \in M[s_{u_k}, o_{v_k}]$. Let $\pi_1^\star$, ..., $\pi_n^\star$ be the atomic programs of $\alpha(s_1, \ldots, s_i, o_1, \ldots, o_j)$. If $Q_{z-1} = (S_{z-1}, O_{z-1}, M_{z-1})$ and $Q_z = (S_z, O_z, M_z)$ are protection states then we shall say that Q_z is derivable from Q_{z-1} in one step using $\pi_z^\star$, $Q_{z-1} \vdash_{\pi_z^\star} Q_z$, iff one of the following conditions is satisfied:

- $\pi_z^\star$ is "enter a into (s, o)", $s \in S_{z-1}$, $o \in O_{z-1}$, and the difference between Q_{z-1} and Q_z is that $M_z[s, o] = M_{z-1}[s, o] \cup \{a\}$,
- $\pi_z^\star$ is "create subject s", $s \notin S_{z-1}$, and the difference between Q_{z-1} and Q_z is that $S_z = S_{z-1} \cup \{s\}$ whereas for all $o \in O_z$, $M_z[s, o] = \emptyset$,
- $\pi_z^\star$ is "create object o", $o \notin O_{z-1}$, and the difference between Q_{z-1} and Q_z is that $O_z = O_{z-1} \cup \{o\}$ whereas for all $s \in S_z$, $M_z[s, o] = \emptyset$,
- $\pi_z^\star$ is "delete a from (s, o)", $s \in S_{z-1}$, $o \in O_{z-1}$, and the difference between Q_{z-1} and Q_z is that $M_z[s, o] = M_{z-1}[s, o] \setminus \{a\}$,

- $\pi_z^\star$ is "destroy subject s", $s \in S_{z-1}$, and the difference between Q_{z-1} and Q_z is that $S_z = S_{z-1} \setminus \{s\}$,
- $\pi_z^\star$ is "destroy object o", $o \in O_{z-1}$, and the difference between Q_{z-1} and Q_z is that $O_z = O_{z-1} \setminus \{o\}$.

The effects describing each "create" program reflect the fact that no permission is granted to created subjects and no permission is granted on created objects. We say that protection state $Q = (S, O, M)$ yields protection state $Q' = (S', O', M')$ under command $\alpha(s_1, \ldots, s_i, o_1, \ldots, o_j)$, $Q \vdash_{\alpha(s_1,\ldots,s_i,o_1,\ldots,o_j)} Q'$, iff Q makes $\alpha(s_1, \ldots, s_i, o_1, \ldots, o_j)$ possible and there are protection states $Q_0, Q_1, \ldots, Q_n$ such that:

- $Q_0 = Q$,
- $Q_{z-1} \vdash_{\pi_z^\star} Q_z$ for all integers z in $\{1, \ldots, n\}$,
- $Q_n = Q'$.

We shall write $Q \vdash_\alpha Q'$ iff there are concrete entities s_1, …, s_i and o_1, …, o_j such that $Q \vdash_{\alpha(s_1,\ldots,s_i,o_1,\ldots,o_j)} Q'$. Also we shall write $Q \vdash_\Pi Q'$ iff there is a command α in Π such that $Q \vdash_\alpha Q'$. Define on protection states the binary relations $\vdash_\Pi^n$, for all non-negative integers n, and $\vdash_\Pi^\star$ as follows:

- $Q \vdash_\Pi^0 Q'$ iff $Q = Q'$,
- $Q \vdash_\Pi^{n+1} Q'$ iff there is a protection state Q'' such that $Q \vdash_\Pi^n Q''$ and $Q'' \vdash_\Pi Q'$,
- $\vdash_\Pi^\star = \bigcup\{\vdash_\Pi^n : n$ is a non-negative integer$\}$.

Example If Q, Q', Q'', and Q''' are the protection states defined in example 2 and α', α'', and α''' are the following commands:

command $\alpha'(X, Y)$ is
$\Rightarrow$ "create object Y"; "enter a_0 into (X, Y)"; "enter a_1 into (X, Y)";

command $\alpha''(X, X', Y)$ is
$(a_0, X, Y) \Rightarrow$ "enter a_2 into (X', Y)";

command $\alpha'''(X, X', Y)$ is
$(a_0, X, Y) \Rightarrow$ "enter a_3 into (X', Y)";

then $Q \vdash_{\alpha'(s_2,o_2)} Q'$, $Q' \vdash_{\alpha''(s_2,s_0,o_2)} Q''$, and $Q'' \vdash_{\alpha'''(s_2,s_0,o_2)} Q'''$. Hence $Q \vdash_\Pi^\star Q'''$.
Unlike Harrison, Ruzzo, and Ullman, we go on the assumption that $S \cap O = \emptyset$ rather than the assumption that $S \subseteq O$. It is easy to see that this slight change does not affect any of the results if the concept of safety is defined in the following way. Given non-negative integer n, protection system $\Pi = (A, C)$, action a in A, and protection state $Q = (S, O, M)$, we say that Π n-leaks a from Q iff there is a protection state $Q' = (S', O', M')$ such that $Q \vdash_\Pi^n Q'$ and there are s in S' and o in O' such that $a \in M'[s, o]$. We shall say that Π leaks a from Q iff there is a non-negative integer n such that Π n-leaks a from Q. We also say that Q is unsafe for Π and a.
Example If Q is the protection state defined in example 2 and Π contains the commands α', α'', and α''' defined in example 2 then Q is unsafe for Π, and both a_2 and a_3.
We say that protection state $Q' = (S', O', M')$ covers protection state $Q = (S, O, M)$, $Q \sqsubseteq Q'$, iff $S \subseteq S'$, $O \subseteq O'$, and for all s in S and for all o in O, $M[s, o] \subseteq M'[s, o]$. We shall say that command α is monotonic iff it does not contain an atomic program of the form "delete" or "destroy". Now we prove a simple lemma about monotonic.

Lemma 1 *Let α be a monotonic command and Q and Q' be protection states. If $Q \vdash_\alpha Q'$ then $Q \sqsubseteq Q'$.*

Proof The result follows from the fact that α does not contain an atomic program of the form "delete" or "destroy". $\dashv$
A protection system is monotonic iff all its commands are monotonic. We shall say that command α is monoconditional iff it does not contain more than 1 elementary condition. A protection system is monoconditional iff all its commands are monoconditional. We shall say that command α is mono-operational iff it does not contain more than 1 atomic program. A protection system is mono-operational iff all its commands are mono-operational. Let $\mathcal{C}(-, -)$ be the class of all protection systems, $\mathcal{C}^+(-, -)$ be the class of all monotonic protection systems, $\mathcal{C}(1, -)$ be the class of all monoconditional protection systems, $\mathcal{C}^+(1, -)$ be the class of all monotonic monoconditional protection systems, $\mathcal{C}(-, 1)$ be the class of all mono-operational protection systems, and $\mathcal{C}^+(-, 1)$ be the class of all monotonic mono-operational protection systems.
Example For example, the protection system Π containing the commands α', α'', and α''' defined in example 2 is in the class $\mathcal{C}^+(1, -)$. It specifies the ways in which protection states could be modified when subjects create new objects and give themselves rights a_0 and a_1 or when subjects grant rights a_2 and a_3.
Given a class $\mathcal{C}$ of protection systems, the most basic problem on protection systems in $\mathcal{C}$ is the following decision problem:

Problem: SAFETY($\mathcal{C}$),

Input: A protection system $\Pi = (A, C)$ in $\mathcal{C}$, an action a in A, and a protection state $Q = (S, O, M)$,

Output: Determine if Q is unsafe for Π and a.

The safety question is undecidable for generic protection systems and it becomes decidable when protection systems are restricted in some way.

Theorem 1 *1. SAFETY($\mathcal{C}(-,-)$) is undecidable,*

2. SAFETY($\mathcal{C}^+(-,-)$) is undecidable,

3. SAFETY($\mathcal{C}^+(1,-)$) is decidable,

4. SAFETY($\mathcal{C}(-,1)$) is decidable,

5. SAFETY($\mathcal{C}^+(-,1)$) is decidable.

Proof See [10; 11; 12]. $\dashv$

It is not known at present whether SAFETY($\mathcal{C}(1,-)$)) is decidable or not.

3 Explicit prohibitions

A simple way to strengthen the HRU model is to relax the limitation on formation of elementary conditions. At present we can formalize certain positive conditions but not their denials — we accept, for instance, "a is in $M[X,Y]$" but not "a is not in $M[X,Y]$" — and there are a number of protection systems whose expression requires us to allow negative conditions to occur within commands; in this connection see [18]. A distinctive feature of our treatment of explicit prohibitions will be to allow negative conditions to occur within commands' elementary conditions. A command is now an expression of the form:

command $\alpha(X_1,\ldots,X_i,Y_1,\ldots,Y_j)$ is
$+(a_1,X_{u_1},Y_{v_1})\ldots+(a_k,X_{u_k},Y_{v_k})$
$-(a_{k+1},X_{u_{k+1}},Y_{v_{k+1}})\ldots-(a_{k+l},X_{u_{k+l}},Y_{v_{k+l}})$
$\Rightarrow \pi_1;\ldots\pi_n;$

denoting the conditional:

if "a_1 is in $M[X_{u_1},Y_{v_1}]$" ... "a_k is in $M[X_{u_k},Y_{v_k}]$" "a_{k+1} is not in $M[X_{u_{k+1}},Y_{v_{k+1}}]$" ... "$a_{k+l}$ is not in $M[X_{u_{k+l}},Y_{v_{k+l}}]$"
then begin $\pi_1;\ldots\pi_n$; end

where π_1, ..., π_n are atomic programs of the form "enter", "create", "delete", or "destroy". It has k positive elementary conditions, l negative elementary conditions, and n atomic programs. The method we adopt for handling explicit prohibitions is well established in the theory of deductive databases with negation [1] but does not appear to have been considered in the context of protection systems before. The definition of protection states making commands possible must be modified in the following way. If all variables in α are replaced by names of concrete entities, that is, subjects s_1, ..., s_i and objects o_1, ..., o_j, then we shall say that protection state $Q=(S,O,M)$ makes command $\alpha(s_1,\ldots,s_i,o_1,\ldots,o_j)$ possible iff $a_1\in M[s_{u_1},o_{v_1}]$, ..., $a_k\in M[s_{u_k},o_{v_k}]$, $a_{k+1}\notin M[s_{u_{k+1}},o_{v_{k+1}}]$, ..., $a_{k+l}\notin M[s_{u_{k+l}},o_{v_{k+l}}]$. On this account the binary relations $\vdash_{\pi_z^\star}$, $\vdash_{\alpha(s_1,\ldots,s_i,o_1,\ldots,o_j)}$, $\vdash_\alpha$, $\vdash_\Pi$, $\vdash_\Pi^n$, and $\vdash_\Pi^\star$ are defined just as above in section 2.

Example If Q, Q', Q'', and Q''' are the protection states presented in a matrix form by table 1 and α', α'', and α''' are the following commands:

command $\alpha'(X,Y)$ is
$\Rightarrow$ "create object Y"; "enter a_0 into (X,Y)"; "enter a_1 into (X,Y)";

command $\alpha''(X,X',Y)$ is
$+(a_0,X,Y)$ $-(a_3,X',Y)$ $\Rightarrow$ "enter a_2 into (X',Y)";

command $\alpha'''(X,X',Y)$ is
$+(a_0,X,Y)$ $-(a_2,X',Y)$ $\Rightarrow$ "enter a_3 into (X',Y)";

then $Q\ \vdash_{\alpha'(s_2,o_2)}\ Q'$, $Q'\ \vdash_{\alpha''(s_2,s_0,o_2)}\ Q''$, but $Q''\not\vdash_{\alpha'''(s_2,s_0,o_2)} Q'''$. Moreover $Q\not\vdash_\Pi^\star Q'''$.

We generalize the definitions of n-leaks, leaks, and unsafe to protection systems with explicit prohibitions. If we define the concepts of covers and monotonic just as above in section 2 then:

Lemma 2 *Let α be a monotonic command and Q and Q' be protection states. If $Q\vdash_\alpha Q'$ then $Q\sqsubseteq Q'$.*

Proof The result follows from the fact that α does not contain an atomic program of the form "delete" or "destroy". $\dashv$

We extend the definitions of monoconditional and monooperational to protection systems with explicit prohibitions. If we define the safety problem just as above in section 2 then:

Theorem 2 *1. SAFETY($\mathcal{C}(-,-)$) is undecidable,*

2. SAFETY($\mathcal{C}^+(-,-)$) is undecidable.

Proof By items 1 and 2 of theorem 1. $\dashv$

It would be nice to be able to prove decidability of SAFETY($\mathcal{C}^+(1,-)$), SAFETY($\mathcal{C}(-,1)$), or SAFETY($\mathcal{C}^+(-,1)$), but this is not known. Also, the decidability of SAFETY($\mathcal{C}(1,-)$) is not known. What has been proved, however, is that the safety question is decidable if protection systems are restricted in the following way. We shall say that command α is positive iff it contains neither a negative elementary condition nor an atomic program of the form "create", command α is neutral iff it contains no elementary conditions, and command α is negative iff it contains neither a positive elementary condition nor an atomic program of the form "create". The most significant result that emerges from the concepts of positive, neutral, and negative is the following:

Lemma 3 *Let α and α' be monotonic commands and Q and Q' be protection states. If α is positive, α' is neutral or negative, and $Q\vdash_\alpha\circ\vdash_{\alpha'} Q'$ then $Q\vdash_{\alpha'}\circ\vdash_\alpha Q'$.*

Proof The result follows from the fact that α contains neither a negative elementary condition nor an atomic program of the form "create", "delete", or "destroy" and α' contains neither a positive elementary condition nor an atomic program of the form "delete" or "destroy". $\dashv$

We will make heavy use, usually without explicit comment, of the following:

- If command α is monotonic and positive then none of its elementary conditions is negative and none of its atomic programs is of the form "create", "delete", or "destroy",
- If command α is monotonic and neutral then it contains no elementary conditions and none of its atomic programs is of the form "delete", or "destroy",
- If command α is monotonic and negative then none of its elementary conditions is positive and none of its atomic programs is of the form "create", "delete", or "destroy".

A protection system is pure iff all its commands are positive, neutral, or negative. Let $\mathcal{C}_p(-,-)$ be the class of all pure protection systems, $\mathcal{C}_p^+(-,-)$ be the class of all monotonic pure protection systems, $\mathcal{C}_p(1,-)$ be the class of all pure monoconditional protection systems, $\mathcal{C}_p^+(1,-)$ be the class of all monotonic pure monoconditional protection systems, $\mathcal{C}_p(-,1)$ be the class of all pure mono-operational protection systems, and $\mathcal{C}_p^+(-,1)$ be the class of all monotonic pure mono-operational protection systems. The reason for the apparently unnatural choice of the concepts of positive, neutral, and negative will soon become clear.

Theorem 3 *SAFETY($\mathcal{C}_p^+(1,-)$) is decidable.*

Proof Let $\Pi = (A, C)$ be a protection system in $\mathcal{C}_p^+(1,-)$, Π^+ be the set of all positive commands in Π, Π^0 be the set of all neutral commands in Π, and Π^- be the set of all negative commands in Π. Define a binary relation $\prec$ on A by $a' \prec a''$ iff there is a command α in Π^+ such that a' occurs in α's atomic programs and a'' occurs in α's elementary conditions. Define on actions the binary relations $\prec^n$, for all non-negative integers n, and $\prec^\star$ as follows:

- $a \prec^0 a'$ iff $a = a'$,
- $a \prec^{n+1} a'$ iff there is an action a'' such that $a \prec^n a''$ and $a'' \prec a'$,
- $\prec^\star = \bigcup\{\prec^n$: n is a non-negative integer$\}$.

It is easy to check that $\prec^\star$ is decidable. Let a be an action in A and $Q = (S, O, M)$ be a protection state. If Q is unsafe for Π and a then there is a protection state $Q' = (S', O', M')$ such that $Q \vdash_\Pi^\star Q'$ and there are s in S' and o in O' such that $a \in M'[s, o]$. Let p be a non-negative integer and $Q_0 = (S_0, O_0, M_0)$, $Q_1 = (S_1, O_1, M_1)$, ..., $Q_p = (S_p, O_p, M_p)$ be protection states. Suppose $Q_0 \vdash_{\alpha_1} Q_1 \ldots \vdash_{\alpha_p} Q_p$ is a minimal length computation between Q and Q' using commands in Π. By lemma 3, we may assume that there is an integer q between 0 and p such that α_1, ..., α_q are neutral or negative and α_{q+1}, ..., α_p are positive. We have to consider 3 cases.

$q = p$**.** Hence Q is unsafe for $\Pi^0 \cup \Pi^-$ and a.

$q < p$ **and** α_{q+1} **contains no elementary conditions.** Hence there are a command α' in Π^+ and an action a' in A such that α' contains no elementary conditions, $a \prec^\star a'$, a' occurs in α''s atomic programs, and there is a protection state $Q' = (S', O', M')$ such that $Q \vdash^\star_{\Pi^0 \cup \Pi^-} Q'$, $S' \neq \emptyset$, and $O' \neq \emptyset$.

$q < p$ **and** α_{q+1} **contains elementary conditions.** Hence there are a command α' in Π^+ and an action a' in A such that α' contains elementary conditions, $a \prec^\star a'$, a' occurs in α''s elementary conditions, and Q is unsafe for $\Pi^0 \cup \Pi^-$ and a'.

We are now almost through with the proof of theorem 3. All we have to do is show the following:

Claim It is decidable to determine, given a protection system $\Pi = (A, C)$ in $\mathcal{C}_p^+(1,-)$, an action a in A, and a protection state $Q = (S, O, M)$, if Q is unsafe for $\Pi^0 \cup \Pi^-$ and a.

By definition of unsafe, Q is unsafe for $\Pi^0 \cup \Pi^-$ and a iff one of the following conditions is satisfied:

- There are s in S and o in O such that $a \in M[s, o]$.
- There is a command α in Π^0 such that a occurs in α's atomic programs and there is a protection state $Q' = (S', O', M')$ such that $Q \vdash^\star_{\Pi^0} Q'$, $S' \neq \emptyset$, and $O' \neq \emptyset$,
- There are a command α in Π^- and an action a' in A such that a occurs in α's atomic programs, a' occurs in α's elementary conditions, and there is a protection state $Q' = (S', O', M')$ such that $Q \vdash^\star_{\Pi^0} Q'$ and there are s in S' and o in O' such that $a' \notin M'[s, o]$.

It is easy to check that the conditions above are decidable. This completes the proof of the claim, and hence of the theorem. $\dashv$

Are SAFETY($\mathcal{C}_p(-,-)$), SAFETY($\mathcal{C}_p^+(-,-)$), SAFETY($\mathcal{C}_p(1,-)$), SAFETY($\mathcal{C}_p(-,1)$), and SAFETY($\mathcal{C}_p^+(-,1)$) decidable?

4 Explicit obligations

In this section we enrich the HRU model by introducing explicit obligations. In what follows, protection states are

Q	o_0	o_1
s_0	$(a_1, 0)$	$(a_0, +1), (a_1, 0)$
s_1	$(a_0, +1), (a_1, 0)$	$(a_1, 0)$
s_2	$(a_1, 0)$	$(a_1, 0)$

Q'	o_0	o_1	o_2
s_0	$(a_1, 0)$	$(a_0, +1), (a_1, 0)$	
s_1	$(a_0, +1), (a_1, 0)$	$(a_1, 0)$	
s_2	$(a_1, 0)$	$(a_1, 0)$	$(a_0, +1), (a_1, 0)$

Q''	o_0	o_1	o_2
s_0	$(a_1, 0)$	$(a_0, +1), (a_1, 0)$	$(a_2, 0)$
s_1	$(a_0, +1), (a_1, 0)$	$(a_1, 0)$	
s_2	$(a_1, 0)$	$(a_1, 0)$	$(a_0, +1), (a_1, 0)$

Q'''	o_0	o_1	o_2
s_0	$(a_1, 0)$	$(a_0, +1), (a_1, 0)$	$(a_2, 0), (a_3, 0)$
s_1	$(a_0, +1), (a_1, 0)$	$(a_1, 0)$	
s_2	$(a_1, 0)$	$(a_1, 0)$	$(a_0, +1), (a_1, 0)$

Table 2. Protection states Q, Q', Q'', and Q'''.

ordered triples $Q = (S, O, M)$ where M is a function assigning to each subject s in S and each object o in O a total mapping $M[s, o]$ from the set of all actions to the rational numbers in $[-1, +1]$, the relationship $M[s, o](a) = x$ being read "subject s has permission of degree x to perform action a on object o". Degrees between -1 and $+1$ represent permission levels. The lower a negative degree, the less recommended the action; the higher a positive degree, the greater the recommendation to execute the action:

- $M[s, o](a) = -1$ means that s is forbidden to perform a on o,
- $-1 < M[s, o](a) < 0$ means that a on o is inadvisable to s,
- $M[s, o](a) = 0$ permits s both to do and not to do a on o,
- $0 < M[s, o](a) < +1$ means that a on o is advisable to s,
- $M[s, o](a) = +1$ means that s is obliged to perform a on o.

Example Table 2 illustrates protection states presented in a matrix form. The entries in the matrices specify the degrees of the actions that each subject has permission to perform on each object. Note that degrees equal to -1 are not represented.

The following atomic programs are used to modify protection states: "increase a of Δ in (X, Y)" where Δ is in $[0, 1]$, "create subject X", "create object Y", "decrease a of Δ in (X, Y)" where Δ is in $[0, 1]$, "destroy subject X", and "destroy object Y". These atomic programs can be combined into commands, i.e. expressions of the form:

command $\alpha(X_1, \ldots, X_i, Y_1, \ldots, Y_j)$ is
$(a_1, X_{u_1}, Y_{v_1}) \geq \delta_1 \ldots (a_k, X_{u_k}, Y_{v_k}) \geq \delta_k \Rightarrow \pi_1; \ldots \pi_n;$

denoting the conditional:

if "$M[X_{u_1}, Y_{v_1}](a_1) \geq \delta_1$" … "$M[X_{u_k}, Y_{v_k}](a_k) \geq \delta_k$"
then begin π_1; … π_n; end

where δ_1, …, δ_k are in $[-1, +1]$ and π_1, …, π_n are atomic programs of the form "increase", "create", "decrease", or "destroy". It has k elementary conditions and n atomic programs. Replacing variables in α by names of concrete entities, that is, subjects s_1, …, s_i and objects o_1, …, o_j, we shall say that protection state $Q = (S, O, M)$ makes command $\alpha(s_1, \ldots, s_i, o_1, \ldots, o_j)$ possible iff $M[s_{u_1}, o_{v_1}](a_1) \geq \delta_1$, …, $M[s_{u_k}, o_{v_k}](a_k) \geq \delta_k$. On this account the binary relations $\vdash_{\alpha(s_1, \ldots, s_i, o_1, \ldots, o_j)}$, $\vdash_\alpha$, $\vdash_\Pi$, $\vdash_\Pi^n$, and $\vdash_\Pi^\star$ are defined just as above in section 2 whereas if $\pi_1^\star$, …, $\pi_n^\star$ are the atomic programs of $\alpha(s_1, \ldots, s_i, o_1, \ldots, o_j)$ and $Q_{z-1} = (S_{z-1}, O_{z-1}, M_{z-1})$ and $Q_z = (S_z, O_z, M_z)$ are protection states then we shall say that Q_z is derivable from Q_{z-1} in one step using $\pi_z^\star$, $Q_{z-1} \vdash_{\pi_z^\star} Q_z$, iff one of the following conditions is satisfied:

- $\pi_z^\star$ is "increase a of Δ in (s, o)", $s \in S_{z-1}$, $o \in O_{z-1}$, and the difference between Q_{z-1} and Q_z is that $M_z[s, o](a) = M_{z-1}[s, o](a) \times (1 - \Delta) + \Delta$,
- $\pi_z^\star$ is "create subject s", $s \notin S_{z-1}$, and the difference between Q_{z-1} and Q_z is that $S_z = S_{z-1} \cup \{s\}$ whereas for all $o \in O_z$, $M_z[s, o](a) = -1$ for each action a,
- $\pi_z^\star$ is "create object o", $o \notin O_{z-1}$, and the difference between Q_{z-1} and Q_z is that $O_z = O_{z-1} \cup \{o\}$ whereas for all $s \in S_z$, $M_z[s, o](a) = -1$ for each action a,
- $\pi_z^\star$ is "decrease a of Δ in (s, o)", $s \in S_{z-1}$, $o \in O_{z-1}$, and the difference between Q_{z-1} and Q_z is that $M_z[s, o] = M_{z-1}[s, o] \times (1 - \Delta) - \Delta$,
- $\pi_z^\star$ is "destroy subject s", $s \in S_{z-1}$, and the difference between Q_{z-1} and Q_z is that $S_z = S_{z-1} \setminus \{s\}$,
- $\pi_z^\star$ is "destroy object o", $o \in O_{z-1}$, and the difference between Q_{z-1} and Q_z is that $O_z = O_{z-1} \setminus \{o\}$.

The effects describing each "create" program reflect the fact that no permission is granted to created subjects and no permission is granted on created objects.

Example If Q, Q', Q'', and Q''' are the protection states defined in example 2 and α', α'', and α''' are the following commands:

command $\alpha'(X, Y)$ is
$\Rightarrow$ "create object Y"; "increase a_0 of 1.0 in (X, Y)"; "increase a_1 of 0.5 in (X, Y)";

command $\alpha''(X, X', Y)$ is
$(a_0, X, Y) \geq 1 \Rightarrow$ "increase a_2 of 0.5 in (X', Y)";

command $\alpha'''(X, X', Y)$ is
$(a_0, X, Y) \geq 1 \Rightarrow$ "increase a_3 of 0.5 in (X', Y)";

then $Q \vdash_{\alpha'(s_2,o_2)} Q'$, $Q' \vdash_{\alpha''(s_2,s_0,o_2)} Q''$, and $Q'' \vdash_{\alpha'''(s_2,s_0,o_2)} Q'''$. Hence $Q \vdash^\star_\Pi Q'''$.
This justifies the following definition of the concept of safety. Given non-negative integer n, protection system $\Pi = (A, C)$, action a in A, rational number x in $[-1, +1]$, and protection state $Q = (S, O, M)$, we say that Π n-leaks a with degree x from Q iff there is a protection state $Q' = (S', O', M')$ such that $Q \vdash^n_\Pi Q'$ and there are s in S' and o in O' such that $M'[s, o](a) \geq x$. We shall say that Π leaks a with degree x from Q iff there is a non-negative integer n such that Π n-leaks a with degree x from Q. We also say that Q is unsafe for Π and a with degree x. We say that protection state $Q' = (S', O', M')$ covers protection state $Q = (S, O, M)$, $Q \sqsubseteq Q'$, iff $S \subseteq S'$, $O \subseteq O'$, and for all s in S and for all o in O, $M[s, o](a) \leq M'[s, o](a)$ for all actions a. We shall say that command α is monotonic iff it does not contain an atomic program of the form "decrease" or "destroy". Now we prove a simple lemma about monotonic.

Lemma 4 *Let α be a monotonic command and Q and Q' be protection states. If $Q \vdash_\alpha Q'$ then $Q \sqsubseteq Q'$.*

Proof The result follows from the fact that α does not contain an atomic program of the form "decrease" or "destroy". $\dashv$
A protection system is monotonic iff all its commands are monotonic. We generalize the definitions of monoconditional and mono-operational to protection systems with explicit obligations. If we define the safety problem in the following way:

Problem: SAFETY($\mathcal{C}$),

Input: A protection system $\Pi = (A, C)$ in $\mathcal{C}$, an action a in A, a rational number x in $[-1, +1]$, and a protection state $Q = (S, O, M)$,

Output: Determine if Q is unsafe for Π and a with degree x,

then:

Theorem 4 1. *SAFETY($\mathcal{C}(-, -)$) is undecidable,*

2. *SAFETY($\mathcal{C}^+(-, -)$) is undecidable.*

Proof Consider a HRU protection system Π. If Π' is the protection system with explicit obligations obtained from Π by modifying its commands as follows:

- Replace each elementary condition (a, X, Y) by the positive elementary condition $(a, X, Y) \geq 0$,
- Replace each atomic program "enter a into (X, Y)" by the atomic program "increase a of 0.5 in (X, Y)",
- Replace each atomic program "delete a from (X, Y)" by the atomic program "decrease a of 1 in (X, Y)",

then, obviously, Π and Π' leak the same actions. By items 1 and 2 of theorem 1, SAFETY($\mathcal{C}(-, -)$) and SAFETY($\mathcal{C}^+(-, -)$) are undecidable for HRU protection systems. Hence SAFETY($\mathcal{C}(-, -)$) and SAFETY($\mathcal{C}^+(-, -)$) are undecidable for protection systems with explicit obligations. $\dashv$
It turns out that:

Theorem 5 *SAFETY($\mathcal{C}^+(1, -)$) is decidable.*

Proof Let $\Pi = (A, C)$ be a protection system in $\mathcal{C}^+(1, -)$, Π^i be the set of all commands in Π containing an atomic program of the form "increase", and Π^c be the set of all commands in Π not containing an atomic program of the form "increase". Define a binary relation $\prec$ on A by $a' \prec a''$ iff there is a command α in Π^i such that a' occurs in α's atomic programs and a'' occurs in α's elementary conditions. Define on actions the binary relations $\prec^n$, for all non-negative integers n, and $\prec^\star$ as follows:

- $a \prec^0 a'$ iff $a = a'$,
- $a \prec^{n+1} a'$ iff there is an action a'' such that $a \prec^n a''$ and $a'' \prec a'$,
- $\prec^\star = \bigcup\{\prec^n \colon n \text{ is a non-negative integer}\}$.

It is easy to check that $\prec^\star$ is decidable. Let a be an action in A, x be a rational number in $[-1, +1]$, and $Q = (S, O, M)$ be a protection state. If Q is unsafe for Π and a with degree x then there is a protection state $Q' = (S', O', M')$ such that $Q \vdash^\star_\Pi Q'$ and there are s in S' and o in O' such that $M'[s, o](a) \geq x$. Let p be a non-negative integer and $Q_0 = (S_0, O_0, M_0)$, $Q_1 = (S_1, O_1, M_1)$, ..., $Q_p = (S_p, O_p, M_p)$ be protection states. Suppose $Q_0 \vdash_{\alpha_1} Q_1 \ldots \vdash_{\alpha_p} Q_p$ is a minimal length computation between Q and Q' using commands in Π. Following the line of reasoning suggested by Harrison and Ruzzo [11], we may assume that there is an integer q between 0 and p such that α_1, ..., α_q do not contain an atomic program of the form "increase" and α_{q+1}, ..., α_p contain an atomic program of the form "increase". We have to consider 3 cases.

$q = p$**.** Hence Q is unsafe for Π^c and a.

$q < p$ **and** α_{q+1} **contains no elementary conditions.** Hence there are a command α' in Π^i and an action a' in A such that α' contains no elementary conditions, $a \prec^\star a'$, a' occurs in α''s atomic programs, and there is a protection state $Q' = (S', O', M')$ such that $Q \vdash^\star_{\Pi^c} Q'$, $S' \neq \emptyset$, and $O' \neq \emptyset$.

$q < p$ **and** α_{q+1} **contains elementary conditions.** Hence there are a command α' in Π^i and an action a' in A such that α' contains elementary conditions, $a \prec^\star a'$, a' occurs in α''s elementary conditions, and Q is unsafe for Π^c and a'.

It is easy to check that the conditions above are decidable. This completes the proof of the theorem. $\dashv$

Are SAFETY$(\mathcal{C}(-,1))$ and SAFETY$(\mathcal{C}^+(-,1))$ decidable? To answer this question we need the following:

Lemma 5 *Let α be a mono-operational command and Q and Q' be protection states. If α is not monotonic and $Q \vdash_\alpha Q'$ then $Q \sqsupseteq Q'$.*

Proof The result follows from the fact that α is a mono-operational command containing an atomic program of the form "decrease" or "destroy". $\dashv$

Lemma 6 *Let α be a mono-operational command and Q and Q' be protection states. If α is monotonic and $Q \sqsupseteq \circ \vdash_\alpha Q'$ then $Q \vdash_\alpha \circ \sqsupseteq Q'$.*

Proof The result follows from the fact that α is a mono-operational command containing an atomic program of the form "increase" or "create". $\dashv$

With this established, we now prove the following:

Theorem 6 *1. SAFETY$(\mathcal{C}(-,1))$ is decidable,*

2. SAFETY$(\mathcal{C}^+(-,1))$ is decidable.

Proof Let $\Pi = (A, C)$ be a protection system in $\mathcal{C}(-,1)$, Π^i be the set of all commands in Π containing an atomic program of the form "increase", and Π^c be the set of all commands in Π not containing an atomic program of the form "increase". Define a binary relation $\prec$ on A by $a' \prec a''$ iff there is a command α in Π^i such that a' occurs in α's atomic programs and a'' occurs in α's elementary conditions. Define on actions the binary relations $\prec^n$, for all non-negative integers n, and $\prec^\star$ as follows:

- $a \prec^0 a'$ iff $a = a'$,
- $a \prec^{n+1} a'$ iff there is an action a'' such that $a \prec^n a''$ and $a'' \prec a'$,
- $\prec^\star = \bigcup\{\prec^n : n$ is a non-negative integer$\}$.

It is easy to check that $\prec^\star$ is decidable. Let a be an action in A, x be a rational number in $[-1, +1]$, and $Q = (S, O, M)$ be a protection state. If Q is unsafe for Π and a with degree x then there is a protection state $Q' = (S', O', M')$ such that $Q \vdash^\star_\Pi Q'$ and there are s in S' and o in O' such that $M'[s, o](a) \geq x$. Let p be a non-negative integer and $Q_0 = (S_0, O_0, M_0)$, $Q_1 = (S_1, O_1, M_1)$, ..., $Q_p = (S_p, O_p, M_p)$ be protection states. Suppose $Q_0 \vdash_{\alpha_1} Q_1 \ldots \vdash_{\alpha_p} Q_p$ is a minimal length computation between Q and Q' using commands in Π. By lemmas 5 and 6, we may assume that $\alpha_1, \ldots, \alpha_p$ are monotonic. Following the line of reasoning suggested by Harrison, Ruzzo, and Ullman [12], we may also assume that there is an integer q between 0 and p such that α_1, ..., α_q do not contain an atomic program of the form "increase" and $\alpha_{q+1}, \ldots, \alpha_p$ contain an atomic program of the form "increase". We have to consider 3 cases.

$q = p$**.** Hence Q is unsafe for Π^c and a.

$q < p$ **and** α_{q+1} **contains no elementary conditions.** Hence there are a command α' in Π^i and an action a' in A such that α' contains no elementary conditions, $a \prec^\star a'$, a' occurs in α''s atomic programs, and there is a protection state $Q' = (S', O', M')$ such that $Q \vdash^\star_{\Pi^c} Q'$, $S' \neq \emptyset$, and $O' \neq \emptyset$.

$q < p$ **and** α_{q+1} **contains elementary conditions.** Hence there are a command α' in Π^i and an action a' in A such that α' contains elementary conditions, $a \prec^\star a'$, a' occurs in α''s elementary conditions, and Q is unsafe for Π^c and a'.

It is easy to check that the conditions above are decidable. This completes the proof of the theorem. $\dashv$

At this point, we do not know whether SAFETY$(\mathcal{C}(1,-))$ is decidable or not.

5 Conclusion

This paper has had as its goal the formulation of a framework for access control with prohibitions and obligations. We demonstrate that SAFETY, the most basic problem on protection systems, is decidable for monotonic pure protection systems with explicit prohibitions, monotonic mono-conditional protection systems with explicit obligations, and mono-operational protection systems with explicit obligations. As for future work, we plan to deal with the definition of timed protection systems and the safety issues involved in their use. Other models incorporate the notion of time in specifying access control requirements. We should consider, for instance, the model introduced by [4] within the context of role-based access control. The temporal constraints specified there can be used to implement conditions like "subject s has either permission to perform action a_1 or permission to perform action a_2 on object o". The intensive study of the safety issues relating to the support of such conditions in timed protection systems is still to be done.

Acknowledgments

We would like to thank Fahima Cheikh and Yannick Chevalier for their support throughout the writing of this paper. Thanks also to the project "Développement de systèmes informatiques par raffinement des contraintes sécuritaires" of the action "Sécurité informatique" for partly financing our research.

References

[1] Abiteboul, S., Hull, R., Vianu, V.: Foundations of Databases. Addison-Wesley (1995).

[2] Abou El Kalam, A., El Baida, R., Balbiani, P., Benferhat, S., Cuppens, F., Deswarte, Y., Miège, A., Saurel, C., Trouessin, G.: Organization based access control. In: IEEE 4th International Workshop on Policies for Distributed Systems and Networks. IEEE Computer Society Press (2003).

[3] Balbiani, P., Cheikh, F.: Safety problems in access control with temporal constraints. In V. Gorodetsky, I. Kotenko, V. Skormin (editors): Computer Network Security. Springer-Verlag (2005).

[4] Bertino, E., Bonatti, A., Ferrari, E.: TRBAC: a temporal role-based access control model. ACM Transactions on Information and System Security **4** (2001) 65–104.

[5] Bishop, M.: Computer Security: Art and Science. Addison-Wesley (2003).

[6] Denning, D.: Cryptography and Data Security. Addison-Wesley (1982).

[7] Ferraiolo, D., Barkley, J., Kuhn, D.: A role-based access control model and reference implementation within a corporate intranet. ACM Transactions on Information And System Security **2** (1999) 34–64.

[8] Ferraiolo, D., Kuhn, D., Chandramouli, R.: Role-Based Access Control. Artech House (2003).

[9] Ferraiolo, D., Sandhu, R., Gavrila, S., Kuhn, D., Chandramouli, R.: Proposed NIST standard for role-based access control. ACM Transactions on Information And System Security **4** (2001) 224–274.

[10] Harrison, M.: Theoretical issues concerning protection in operating systems. Advances in Computers **24** (1985) 61–100.

[11] Harrison, M., Ruzzo, W.: Monotonic protection systems. In DeMillo, R., Dobkin, D., Jones, A., Lipton, R. (Editors): Foundations of Secure Computation. Academic Press (1978) 337–363.

[12] Harrison, M., Ruzzo, W., Ullman, J.: Protection in operating systems. Communications of the ACM **19** (1976) 461–471.

[13] Lampson, B.: Protection. Operating Systems Review **8** (1974) 18–24.

[14] Lipton, R., Snyder, L.: On synchronization and security. In DeMillo, R., Dobkin, D., Jones, A., Lipton, R. (Editors): Foundations of Secure Computation. Academic Press (1978) 367–385.

[15] Pieprzyk, J., Hardjono, T., Seberry, J.: Fundamentals of Computer Security. Springer-Verlag (2003).

[16] Sandhu, R.: The typed access matrix model. In: IEEE 13th Symposium on Security and Privacy. IEEE Computer Society Press (1992).

[17] Sandhu, R., Coyne, E., Feinstein, H., Youman, C.: Role-based access control models. Computer **29** (1996) 38–47.

[18] Sandhu, R., Ganta, S: On testing for the absence of rights in access control models. In: IEEE 6th Computer Security Foundations Workshop. IEEE Computer Society Press (1993).

[19] Soshi, M.: Safety analysis of the dynamic-typed access matrix model. In Cuppens, F., Deswarte, Y., Gollmann, D., Waidner, M. (Editors): Computer Security — ESORICS 2000. Springer-Verlag, Lecture Notes in Computer Science **1895** (2000) 106–121.