Expressive Power of the Schematic Protection Model*

Ravi S. Sandhu

Center for Secure Information Systems
and
Department of Information and Software Systems Engineering
George Mason University, Fairfax, VA 22030-4444

September 27, 1991

*This manuscript has been accepted for publication in the *Journal of Computer Security*. An early version of this paper was presented at the *Computer Security Foundations Workshop*, Franconia, New Hampshire, June 12-15, 1988. An extended abstract appears in the proceedings (MITRE technical report M88-37, pages 188-193).

Abstract

In this paper we show that the Schematic Protection Model (SPM) subsumes several well-known protection models as particular instances. We show this for a diverse collection of models including the Bell-LaPadula multi-level security model, take-grant models, and grammatical protection systems. Remarkably SPM subsumes these models within its known efficiently decidable cases for safety analysis (i.e., the determination or whether or not a given privilege can possibly be acquired by a particular subject). Therefore SPM subsumes these models not only in terms of its expressive power but also in terms of safety analysis. This is in sharp contrast to the Harrison-Ruzzo-Ullman (HRU) access-matrix model. HRU does subsume all the models discussed in this paper in terms of expressive power. However, all known constructions of these models in HRU require multi-conditional commands (i.e., commands whose conditions have two or more terms), whereas safety is undecidable in HRU even for bi-conditional commands (i.e., commands whose conditions have exactly two terms).

1 INTRODUCTION

Access controls for protection and sharing of information and physical resources are an essential component of any multi-user computer system. For this purpose, these systems are typically viewed as consisting of subjects and objects. Subjects are generally the active entities such as users or processes, while objects are passive entities such as text files. Protection is enforced by ensuring that subjects can execute only those operations for which they are authorized.

Access controls are useful to the extent they meet the user community's needs. They need to be flexible so that individual users can specify access of other users to the objects they control. At the same time the discretionary power given to individual users must be constrained to meet the overall objectives and policies of an organization. For example, members of a project team might be allowed to freely share project documents with each other but only the project leader is authorized to allow non-members to read project documents.

The *protection state* of a system is defined by the *privileges*[*] in subjects' *domains* at a given moment. Hereafter, we understand state to mean protection state. *Inert privileges* authorize operations that do not modify the state, e.g., reading or writing a file. *Control privileges* authorize control operations that modify the protection state, e.g., user X authorizes user Y to read file Z. Control privileges define the dynamics of authorization. Once the initial state has been established,[†] the protection state evolves by the autonomous actions of subjects constrained by control privileges. The challenge is to ensure that all reachable states conform with the policy that the security administrator wishes to implement.

A protection model provides a framework for specifying the dynamics of the protection state. This is usually done by stating rules which prescribe the authorization for making incremental changes in the state. We call such a collection of rules an *authorization scheme*, often abbreviated simply as *scheme*. To understand the implications of a scheme it must be possible to determine the cumulative effect of authorized incremental changes in the protection state. The incremental state changes authorized by a scheme may appear innocent enough in isolation, although their cumulative effect turns out to be undesirable. So for a given initial state and authorization scheme, we need to characterize protection states that are reachable.

This problem was first identified in [14] where it is called the *safety problem*. In its most basic form, the safety question asks: is there a reachable state in which a particular subject possesses a particular privilege which it did not previously possess?

[*]We view a privilege as an undefined primitive concept. For the most part, privileges can be treated as synonymous to access rights. However, there are privileges such as security level and type which are usually represented as attributes of subjects and objects rather than as access rights.

[†]The initial state is established by the security administrator at the moment of system generation. The mechanics of this procedure are inherently implementation dependent, and therefore not modeled within SPM.

It is the fundamental question which a protection model must confront. Since subjects are usually authorized to create new subjects and objects, the system is unbounded and it is not certain that such analysis will be decidable, let alone tractable, without sacrificing generality.

Safety analysis becomes particularly complex when control privileges can themselves be dynamically acquired. To illustrate the need for propagating control privileges, consider the example above where only the project leader is authorized to allow non-members to read project documents. This policy can be enforced by giving the project leader a special control privilege not available to ordinary project members. Suppose in addition we wish to allow the project leader to delegate this special authority to a project member, say while the project leader is absent on a business trip. This can be achieved by allowing the project leader to grant the special control privilege to project members.

Analysis issues were first formalized by Harrison, Ruzzo and Ullman [14] in the context of the well-known access matrix model [13, 17]. The matrix has a row for each subject and a column for each subject or object. The [X,Y] cell of the matrix contains symbols called rights which authorize subject X to perform operations on entity Y. An authorization scheme is defined by a set of commands. Each command has a condition part and a body. The condition specifies the rights that are required to exist in the matrix before the body can be executed for its actual arguments. The body consists of a sequence of primitive operations. The primitive operations enter or delete a right from a cell of the matrix, create a new row or column, or destroy an existing row or column. This model is hereafter called HRU.

In the general HRU setting safety is undecidable [14]. Furthermore safety remains undecidable even if the condition part of each command has at most two terms and there are no delete or destroy operations in the body [15]. The very weak assumptions from which undecidability follows are most disappointing. There does not appear to be any natural and useful special case of this model for which safety is efficiently decidable [14, 15]. Specifically, safety in HRU is known to be decidable for mono-conditional monotonic commands [15] (i.e., commands whose condition part has only one term) but is undecidable even for bi-conditional monotonic commands [15] (i.e., commands whose condition part has exactly two terms). Most practical systems require multi-conditional commands (i.e., commands whose condition part has two or more terms).

The schematic protection model (SPM) [34] was developed in response to this situation. SPM provides considerably more structure than HRU. It classifies subjects and objects into protection types. The dynamic component of a protection state in SPM consists of tickets (capabilities). The key idea is that the authorization scheme is specified in terms of protection types. In particular, subject creation is authorized by a can-create binary relation on types. Safety is decidable provided this relation is acyclic, and in certain cases even if it has cycles of length one [34]. On the other

hand with arbitrary cycles in can-create, safety is undecidable [38]. Fortunately, it appears that SPM schemes of practical interest satisfy the decidability constraints, as demonstrated by the constructions of this paper and the examples of [32, 33, 34, 36].

Our objective in this paper is to demonstrate that SPM subsumes several well-known protection models as special cases. Specifically, we show that the Bell-LaPadula multi-level security model [3], take-grant models [16, 21, 41] and grammatical protection systems [10, 22] are particular instances of SPM.

Remarkably in all our constructions the resulting SPM scheme has efficient safety analysis. Therefore SPM subsumes these models not only in terms of its expressive power but also in terms of safety analysis. This is in sharp contrast to HRU. HRU can simulate all the models discussed in this paper. However, because of its very weak safety properties HRU is unable to subsume these models within its known decidable classes for safety.[‡] However, all known constructions of these models in HRU require multi-conditional commands (i.e., commands whose conditions have two or more terms), whereas safety is undecidable in HRU even for bi-conditional commands (i.e., commands whose conditions have exactly two terms) [15]. In SPM, on the other hand, these models are all simulated by acyclic attenuating schemes which are known to have decidable safety [34].

Taken collectively our constructions demonstrate that the safety results for SPM subsume a diversity of published safety results for protection models. In and of itself each construction is also notable for the following reasons.

- The construction for multi-level security models shows that the traditional black-and-white distinction between mandatory and discretionary controls in the Bell-LaPadula model has an alternate expression in SPM in terms of constraints on the propagation of access rights. The SPM viewpoint has the advantage of providing explicit machinery for formulating policies "in between" these two extremes.

- The construction for theft in take-grant models shows a great advantage of SPM whereby assumptions about the behavior of subjects are easily specified as part of a scheme. SPM, therefore, gives us a powerful framework for investigating the consequences of assumptions about trusted behavior.

- Finally, the construction for grammatical systems demonstrates the ability of SPM to simulate models whose control operations are at first sight quite contrary to SPM control operations. It also completes the simulation of take-grant analysis within SPM, since aspects of this analysis require a combination of our constructions for theft and grammatical systems.

[‡]This statement is true if we require that each model be simulated by a "behaviorally equivalent" HRU system. For weaker notions of equivalence it is always possible to construct an "equivalent" HRU system for any model that has a decidable safety problem. This issue is discussed in Section 4.

The rest of the paper is organized as follows. We begin by reviewing SPM and its analysis results respectively in Sections 2 and 3. Section 4 discusses the concept of equivalence among systems and subsumption among models. Section 5 shows that the Bell-LaPadula multi-level security model [3] and several variations of it are expressible as SPM schemes. Section 6 considers how several variations of the take-grant model [16, 21, 41] can be simulated in SPM. Section 7 shows that grammatical protection systems [10, 22] are particular instances of SPM. The constructions of Sections 5, 6 and 7 are largely independent of each other and can be read in any order. Section 8 concludes the paper.

2 THE SCHEMATIC PROTECTION MODEL

In this Section we review the definition of SPM. Our review is necessarily brief. Motivational details for various components of the model are given in [34]. The only difference in notation with respect to [34] is in describing the create-rules.

SPM recognizes two kinds of *entities*, called *subjects* and *objects*, in a system. Subjects can possess privileges and may or may not be active agents in the system. Objects, on the other hand, are purely containers of information. They do not possess privileges and are inherently passive. SPM regards subjects and objects as mutually exclusive. In the literature, subjects are often defined to be a subset of objects. This amounts to calling what SPM calls entities as objects and coining some other term (say, pure objects) for entities which are not subjects. In SPM terminology a process is a subject, and operations (e.g., signal) can be performed on subjects. In the literature, a process is often called a subject when we talk about the process executing operations on other objects, whereas it is called an object when we talk about operations performed on that process. These nuances of terminology are quite straightforwad, but it is important to keep them in mind while reading the paper.

The key notion in SPM is that all entities are instances of protection types. Instances of the same protection type are treated uniformly by control privileges. Hereafter, we understand *type* to mean protection type. SPM assumes strong typing in that every entity is created to be of a specific type which cannot change thereafter. The domain of an SPM subject has two parts: a static type-dependent part defined by the *scheme* and a dynamic part consisting of *tickets* (capabilities). The scheme is defined in terms of types by the security administrator when a system is first set-up and thereafter does not change. Major policy decisions are built into the scheme while details are reflected in the initial distribution of tickets.

Tickets are privileges of the form Y/x. Y/x is an ordered pair where Y identifies some unique entity and the right symbol x authorizes the possessor of this ticket to perform some operation(s) on Y. Tickets are unforgeable and cannot be generated at will by a subject. They can be acquired only in accordance with specific rules which

comprise the scheme. The assumption that a ticket carries only one right symbol simplifies the formal framework without loss of generality. Capabilities with multiple right symbols then correspond to sets of tickets. We often abbreviate sets of tickets in this manner, e.g., Y/uvw denotes the set of tickets {Y/u, Y/v, Y/w}.

2.1 TYPES AND RIGHT SYMBOLS

The first step in defining a scheme is to specify the disjoint sets of object types TO and subject types TS. Their union T is the entire set of entity types. Types identify classes of entities which have common properties for security purposes. For subjects this may be membership in a department or a particular position of authority in a group, such as project leader. For objects this may be a classification such as an internal document or a public document. Types are usually named in lower case italics and entities in upper case roman script. Similarly italics and roman script are used to name sets, functions and relations whose domains involve types and entities respectively. The type of entity Y is denoted by *type*(Y).

The next (or perhaps concurrent) step is to define the right symbols carried by tickets. For this purpose the set of right symbols R is partitioned into two disjoint subsets: the inert rights RI and the control rights RC. Examples of inert rights are the typical read, write, execute and append access rights for a file. The interpretation of control rights will be discussed shortly. We emphasize that SPM does not interpret the inert rights, but rather treats them as abstract symbols. The security administrator is free to define RI as the collection of symbols appropriate for the particular system of interest. The interpretation of these symbols, e.g., that r means read and w means write, is informal and is not specified in the SPM scheme. On the other hand, the interpretation of the control rights is specified in the SPM scheme in terms of link predicates and filter functions.

Every right symbol x comes in two variations x and xc, where c is the copy flag. The only difference between Y/x and Y/xc is that the former ticket cannot be copied from one subject to another whereas the latter may be, provided certain additional conditions to be defined shortly are true. It follows that presence of Y/xc in a domain implies presence of Y/x but not vice versa. We use x:c to denote x or xc with the understanding that multiple occurrences of x:c in the same context are either all read as x or all as xc. When used with multiple right symbols on a ticket the copy flag applies to each symbol, that is Y/uvwc denotes {Y/uc, Y/vc, Y/wc}.

We denote the type of a ticket Y/x:c by *type*(Y/x:c) and define it to be the ordered pair *type*(Y)/x:c. That is, the type of a ticket is determined by the type of entity it addresses and the right symbol it carries. Conventions for representing tickets, especially regarding the copy flag, extend in an obvious way to ticket types. In particular *type*(Y/x) and *type*(Y/xc) are different. This is an important distinction because of the role of the copy flag. The entire set of ticket types is T×R.

The remaining components of a scheme are defined in terms of functions and relations involving the sets TS, T and T×R. SPM requires that T and R be finite, so a scheme is defined by finite sets, relations and functions. SPM recognizes two operations that change the protection state: copy and create.[§]

2.2 THE COPY OPERATION

The copy operation moves a copy of a ticket from the domain of one subject to the domain of another, leaving the original ticket intact. We often speak of copying a ticket from one subject to another although technically a ticket is copied from one subject's domain to another's domain.

The copy operation requires three independent pieces of authorization. A formal statement of these three conditions is given in Section 2.2.2.

1. The original ticket in the source subject's domain must carry the copy flag.

2. There must be a link from the source subject to the destination subject. In general an SPM scheme defines a collection of link predicates $\{\text{link}_i\}$ for this purpose. A link_i is said to exist from one subject to another provided the predicate link_i evaluates to true in a given state.

3. Finally the filter function f_i associated with the link predicate link_i must also authorize the operation.

The subscript i is used to distinguish one link predicate from another as well as to maintain the association between link predicates and filter functions. The symbol i is usually chosen to have some mnemonic significance with respect to the control rights which establish the link or the purpose of the link. We now formally define links and filter functions.

2.2.1 LINK PREDICATES

In an SPM scheme a finite collection of link predicates is defined. Each predicate takes two subjects, say U and V, as arguments and evaluates to true or false. If true, it establishes a connection from U to V which can be used to copy tickets from U to V. The definition of each link predicate is in terms of the presence of some combination of control tickets for U and V in the domains of U and V. The idea is that link predicates can therefore be evaluated by examining the domains of the two subjects of concern and that too only with respect to presence of control tickets for these two subjects. That the definition should depend only on the presence and not

[§]In its original formulation SPM included a third operation called demand. Demand is not used in the constructions of this paper and is known to be formally redundant [37].

the absence of tickets is a well-known principle for protection [31]. As a special case we also allow a link predicate which is always true to be defined. Formally we have the following definition.

Definition 1 *Let dom(U) be the set of tickets possessed by subject U. A* local link predicate $\text{link}_\text{i}(\text{U,V})$ *with U and V as formal parameters is defined as a conjunction or disjunction, but not negation, of the following terms for any z∈RC: U/z∈dom(U), U/z∈dom(V), V/z∈dom(U), V/z∈dom(V), and true.*

For a given state if $\text{link}_\text{i}(\text{A,B})$ is true we say there is a link_i from A to B. We emphasize the existence of a link is necessary but not sufficient for copying tickets from A to B. Examples of local link predicates from the literature are listed below.

1. $\text{link}_{\text{tg}}(\text{U,V}) \equiv \text{U/t}\in\text{dom(V)} \vee \text{V/g}\in\text{dom(U)}$
2. $\text{link}_{\text{t}}(\text{U,V}) \equiv \text{U/t}\in\text{dom(V)}$
3. $\text{link}_{\text{g}}(\text{U,V}) \equiv \text{V/g}\in\text{dom(U)}$
4. $\text{link}_{\text{sr}}(\text{U,V}) \equiv \text{U/r}\in\text{dom(V)} \wedge \text{V/s}\in\text{dom(U)}$
5. $\text{link}_{\text{b}}(\text{U,V}) \equiv \text{U/b}\in\text{dom(U)}$
6. $\text{link}_{\text{p}}(\text{U,V}) \equiv \text{V/p}\in\text{dom(V)}$
7. $\text{link}_{\text{bp}}(\text{U,V}) \equiv \text{U/b}\in\text{dom(U)} \wedge \text{V/p}\in\text{dom(V)}$
8. $\text{link}_{\text{u}}(\text{U,V}) \equiv \textbf{true}$

The first example is from the take-grant model [21] where the t and g control rights are respectively read as take and grant. The next two examples each retain just one of these privileges [23]. The fourth example is from the send-receive mechanism [28, 32] where the s and r control rights are respectively read as send and receive. The first four cases are defined in terms of control tickets for U in V's domain or vice versa. The next three cases are quite different and are defined in terms of a control ticket for U in U's domain or similarly for V. The last case is unique in that it requires no tickets for a link to exist. There are other interesting possibilities for defining link predicates. We anticipate that simple predicates of the kind defined above will suffice in practice, although the model does allow for arbitrarily complex ones.

Since SPM is a model and not an implementation, the precise mechanics by which a link is evaluated to be true or false are deliberately left unspecified. The conservative approach would be to evaluate a link on every occasion that a copy operation is attempted using that link. It is possible to have implementations where the link is evaluated once and "cached" to enable several copy operations. Similarly it is left unspecified whether subjects have to explicitly identify which link to use in a

copy operation or whether the operating system will search for the existence of a suitable link. At this level of detail there are numerous alternatives consistent with the abstract SPM model.

2.2.2 FILTER FUNCTIONS

The final condition required for authorizing a copy operation is defined by the filter functions f_i: TS×TS → $2^{T\times R}$, one for each predicate $link_i$. The interpretation is that Y/x : c can be copied from dom(U) to dom(V) if and only if all of the following are true for some $link_i$.

1. Y/xc∈dom(U)
2. $link_i$(U,V) evaluates to true
3. y/x : c∈$f_i(u,v)$ where U, V and Y are of type u, v and y respectively

Some possible values of $f_i(u,v)$ are T×R, TO×RI and ϕ respectively authorizing all tickets, inert tickets and no tickets to be copied from a subject of type u to a subject of type v over a $link_i$.

The copy flag, link predicates and filter functions together authorize a copy operation in this manner. The first two conditions depend on the protection state whereas the third depends only on the scheme. Note that Y/xc is required in dom(U) for copying either of Y/xc or Y/x. The filter function determines whether or not the copied ticket can have the copy flag. Selectivity in copying is controlled by the filter function entirely in terms of types.

We emphasize that there is a different filter function f_i for each predicate $link_i$. Also, the value of a filter function f_i can specify a different set of ticket types for each pair of its argument subject types. Filter functions are a powerful tool for specifying policies. They impose non-discretionary controls which are inviolable and confine the discretionary behavior of individual subjects.

SPM imposes no assumptions regarding the role of U and V in a copy operation from U to V. For worst-case analysis it is equally acceptable that copying take place at the initiative of U or V alone or require both to cooperate. In this respect SPM is similar to HRU, which also does not specify which of the many subjects involved in a command are regarded as initiators of the command.

2.3 THE CREATE OPERATION

The create operation introduces new subjects and objects in the system. There are two issues here: what types of entities can be created and which tickets are introduced as the immediate result of a create operation.

2.3.1 The Can-Create Function

Authorization for creation is specified in a scheme by the can-create function $cc : \mathrm{TS} \rightarrow 2^{\mathrm{T}}$. The interpretation is that subjects of type u are authorized to create entities of type v if and only if $v \in cc(u)$. For example $cc(user)=\{file\}$ authorizes users to create files. Similarly, $cc(security\text{-}officer)=\{user\}$ authorizes security officers to create users.

2.3.2 Create Rules

The tickets introduced by a create operation are specified by a create-rule for every pair (u,v) such that $v \in cc(u)$. The create-rules are local in that the only tickets introduced are for the parent and child entities in the domains of these two entities. The motivation is that creation should immediately have only a local incremental impact on the state. We emphasize there is a different create-rule for each pair $v \in cc(u)$.

Let subject U of type u create entity V of type v, so U is the parent and V the child. If V is an object the create-rule is specified as follows, where *child* is a special symbol signifying the created object.

$$cr_p(u,v) \subseteq \{child/\mathrm{x:c} |\ \mathrm{x:c} \in \mathrm{RI}\}$$

The interpretation is that the parent U gets V/x:c if and only if $\mathrm{x:c} \in cr_p(u,v)$. For example, $cr_p(user, file) = child/\mathrm{rwc}$ specifies that the creator of a file gets copiable read and write tickets for it.

If V is a subject the situation is more complex since the create-rule must also specify tickets to be placed in V's domain. So if v is a subject type the create-rule has two components as follows.

$$cr_p(u,v) \subseteq \{child/\mathrm{x:c},\ parent/\mathrm{x:c} |\ \mathrm{x:c} \in \mathrm{R}\}$$
$$cr_c(u,v) \subseteq \{child/\mathrm{x:c},\ parent/\mathrm{x:c} |\ \mathrm{x:c} \in \mathrm{R}\}$$

These respectively specify tickets to be placed in the parent and child domains. Tickets for the parent and child are identified by the special symbols *parent* and *child* respectively. The interpretation is the parent U gets U/x:c provided $parent/\mathrm{x:c} \in cr_p(u,v)$ and V/x:c provided $child/\mathrm{x:c} \in cr_p(u,v)$. Similarly the child V gets U/x:c provided $parent/\mathrm{x:c} \in cr_c(u,v)$ and V/x:c provided $child/\mathrm{x:c} \in cr_c(u,v)$. The motivation for allowing a create-rule to introduce tickets for the parent in the parent's own domain is discussed at length in [34].

The following example from the take-grant model [21] specifies that the parent subject gets copiable take and grant tickets for its child, while the child is created with an empty domain: $cr_p(s,s) = child/\mathrm{rwc}$ and $cr_c(s,s) = \phi$.

2.4 SUMMARY OF SPM

In summary, SPM requires the security administrator to specify an authorization scheme by defining the following components.

1. A finite set of types T partitioned into disjoint sets of subject types TS and object types TO.
2. A finite set of rights R partitioned into disjoint sets of inert rights RI and control rights RC.
3. A finite collection of local link predicates $\{\text{link}_\text{i}\}$.
4. A filter function f_i: $\text{TS}\times\text{TS} \rightarrow 2^{\text{T}\times\text{R}}$ for each predicate link_i.
5. A can-create function $cc : \text{TS} \rightarrow 2^{\text{T}}$.
6. A local create-rule for each (u,v) such that $v \in cc(u)$.

A system is specified by defining a scheme and the initial protection state, i.e., the initial set of entities and the initial distribution of tickets. Thereafter the state evolves by copy and create operations.

2.5 REVOCATION

SPM is monotonic in that it lacks facilities for revocation of tickets and deletion of entities. In any real system there must, of course, be mechanisms for revocation and deletion. Similarly, any implementation of SPM would also provide these mechanisms. Fortunately it turns out that under rather general assumptions revocation and deletion can be ignored for safety analysis in the worst case.

Revocation can be ignored in a worst-case scenario provided the effect of revocation can be undone. We call this the *restoration principle*, i.e., whatever can be revoked can be restored [34]. In SPM, if a ticket obtained by a copy or demand operation is revoked it is easily restored by repeating the operation. However if a ticket introduced by a create operation is revoked, it may not be restorable by repeating the operation since each created entity is unique. Also tickets distributed in the initial state may not be restorable. If we assume tickets distributed in the initial state or introduced by create-rules are irrevocable, the restoration principle does not entail any loss of generality in context of SPM. The need for a restoration principle is also demonstrated by the lost object problem. With unrestricted revocation it is possible that all tickets for an object disappear. If tickets for this object cannot be generated on demand, the object thereby becomes inaccessible.

The situation regarding deletion of entities is similar. Here the restoration principle requires that an entity which can be deleted should be replaceable by an equivalent

entity. In general this rules out deletion of entities present in the initial state. Regarding deletion of entities created subsequently, it is always possible to re-create an entity of the same type as was deleted. In other words the individuality of created entities is not significant for analysis of the safety problem whereas the individuality of entities in the initial state may be significant.

To summarize, revocation and deletion policies which are consistent with the restoration principle can be ignored for analysis of the safety problem in a worst-case scenario.

2.6 AN EXAMPLE

We close this Section with a simple example of an SPM scheme based on the well-known concept of ownership. A user is regarded as the owner of all files created by him and has complete discretion regarding access to these files. The following scheme specifies this policy in SPM.

Scheme 1 *Basic owner-based policy.*

1. TS = {*user*}, TO = {*file*}
2. RI = {x:c}, RC = ϕ
3. link_u(X,Y) $\equiv$ **true**
4. $f_u(user, user) = \{file/\text{xc}\}$
5. $cc(user) = file$
6. $cr_p(user, file) = \{file/\text{xc}\}$

The types *user* and *file* correspond to users and files respectively. For simplicity, a single inert right x:c provides access to files. This suffices so long as the policy regarding the dynamics of different inert rights, such as the typical read, write, execute and append, remains the same. There are no control rights so only the universal link predicate is defined. Tickets for files, with or without the copy flag, can be copied across universal links. Users can create files and get a copiable ticket for each created file.

Note that the specification $f_u(user, user) = \{file/\text{x}\}$ would give us a very different behavior. In this case tickets given by the owner of a file to other users cannot carry the copy flag. Consequently the owner is the only one who can ever grant privileges for an owned file to other users.

3 SAFETY IN SPM

In this Section we briefly review the safety analysis of SPM with respect to propagation of access rights. For a system, whose initial state and scheme are given, the safety problem asks whether or not there is a state reachable from the initial state, by the rules of the scheme, with V/x : c in dom(U). The complication in analyzing this problem arises from the create operation. If the scheme does not allow creation safety is easily determined by a polynomial time algorithm [34]. We simply keep executing copy operations until the state stabilizes in the sense that further copy operations do not change any subject's domain. We call this stable state the *no-creates maximal state*. Our approach to dealing with creation is to break the analysis problem into two phases, as follows.

I. From the initial state construct an *augmented state* by create operations alone.

II. Compute the no-creates maximal state from the augmented state of phase I.

This strategy works provided we have a method for constructing a suitable augmented state. We need to prove somehow that subjects and objects in the augmented state account for the potentially unbounded set of entities which can be created.

There is a very natural restriction under which the above strategy can be proved correct. Define the *cc*-digraph to be the directed graph with vertices T and an edge from u to v if and only if $v \in cc(u)$. We say *cc* is *acyclic* if this graph is acyclic. For acyclic *cc* we compute the augmented state in phase I as follows.

procedure augment
mark all subjects in the initial state to be open;
while there exists an open subject U **do**
    forall $v \in cc(type(\mathrm{U}))$ **do**
        let U create an entity of type v;
        mark this created entity to be open;
    end
    mark U to be closed;
end

It is obvious that this procedure terminates if and only if *cc* is acyclic. In [34] it is shown that the no-creates maximal state obtained from this augmented state correctly answers the safety question. The reason for this is quite simple. If a subject creates two entities of the same type, there is no difference between them as far as the scheme is concerned. So for purpose of safety analysis it suffices to create just one of them.

Of course if *cc* has cycles the above procedure will not terminate. Indeed it has been shown that with arbitrary cycles in *cc* safety is undecidable [38]. So there is no algorithm for computing a suitable augmented state in general. For the most part it

appears that cycles in *cc* do not arise in practice. In our experience cycles in can-create occur only in the very special, but also very important, case where a subject is authorized to create new subjects of its own type. Such cycles are called loops and show up in the form $u \in cc(u)$. In other words loops are cycles of length one in the *cc*-digraph. The augmenting construction for *cc* with loops is as follows.

procedure augment with loops
eliminate loops from *cc*;
perform the augment procedure;
restore the loops in *cc*;
forall subjects U such that $type(\mathrm{U}) \in cc(type(\mathrm{U}))$ **do**
    let U create a subject of type $type(\mathrm{U})$;
end

In [34] it is proved that the no-creates maximal state obtained from this augmented state correctly answers the safety question provided the create-rules for loops in *cc* satisfy the following restriction.

Definition 2 *A create-rule for a loop in* cc *is said to be* attenuating *if*

1. $cr_c(u,u) \subseteq cr_p(u,u)$
2. $child/\mathrm{x} : \mathrm{c} \in cr_p(u,u) \Rightarrow parent/\mathrm{x} : \mathrm{c} \in cr_p(u,u)$

The motivation and justification for this definition are discussed at length in [34].

For our purpose in this paper, it is important to demonstrate that models subsumed by SPM are indeed subsumed by acyclic attenuating schemes, which are defined as follows:

Definition 3 *An SPM scheme is* acyclic attenuating *if its* cc*-digraph is acyclic or has loops with attenuating create-rules.*

To summarize, safety is decidable for acyclic attenuating schemes. Moreover, the decision procedures given above are efficient unless the *cc*-digraph is very dense. This completes our review of SPM.

4 EQUIVALENCE OF SYSTEMS

Our main objective in this paper is to show that SPM subsumes three well-known protection models as special cases. In order to do this we must of course define what we mean by subsumption. We do so as follows:

Definition 4 *We say that SPM* subsumes *a model M provided that for every system S which can be specified in M we can construct an* equivalent *SPM system S′.*

To complete this definition we need to define the meaning of "equivalent." The simplest definition of equivalence is perhaps the following one.

Definition 5 *Two systems S and S′ are said to be* equivalent *provided we can construct a mapping σ such that subject s can have access r to object o in S if and only if subject σ(s) can have access σ(r) to object σ(o) in S′.*

By this definition systems are equivalent if they have equivalent worst case behavior. Note that there may be substantial differences between the details of system S and S'. For example:

1. Subject $\sigma(s)$ may have to go through far more convoluted actions to acquire $\sigma(r)$ access to object $\sigma(o)$, as compared to the actions of subject s in acquiring access r to object o.

2. There may be additional subjects, objects and rights in S' that have no direct counterpart in S, but are present due to bookkeeping details in the simulation of S in S'.

3. The mapping σ may be extremely complex (although it must be computable).

Nevertheless, from a perspective of worst-case safety analysis, the two systems are equivalent. In other words if both systems are assumed to be infested with cooperating Trojan Horses who are determined to propagate access rights as far as possible, the net accesses in both systems will be identical. Therefore, safety analysis of S reduces to safety analysis of S' and vice versa.

The constructions of this paper show that the Bell-LaPadula multi-level security model [3], take-grant models [16, 21, 41] and grammatical protection systems [10, 22] are all equivalent to acyclic attenuating schemes in SPM. Therefore, the safety analysis results of SPM also apply to these models. We reiterate that these three models, and SPM itself, are all subsumed by monotonic HRU. However, all known constructions of these models within HRU require multi-conditional commands (i.e., commands whose conditions have two or more terms), whereas safety is undecidable in HRU even for bi-conditional commands (i.e., commands whose conditions have exactly two terms).

The actual constructions given in this paper establish equivalence in a stronger sense than definition 5. It is beyond the scope of this paper to give a formal definition of "stronger" in this context. The intuition is that "stronger" means "behavioral equivalence." That is, every state transition in S, say from state α to state β, can be mimicked by one or more state transitions in S' which applied to state $\sigma(\alpha)$ result in state $\sigma(\beta)$. In other words, it is not only the states of S which are being simulated

in S' but also the individual transitions. It will be evident that this requirement is easily satisfied by the constructions of this paper. Most equivalence results in computer science are actually behavioral equivalence results. For example, the familiar equivalence between classes of automata and formal grammars is of the behavioral variety.

Consider an example to make this intuition clearer. The take-grant model has decidable safety, therefore it is trivial to give an equivalent HRU system S' for a given take-grant system S. We simply run the safety algorithm of take-grant, as part of the σ mapping, and construct the worst-case state as the target HRU system. These two systems are therefore worst-case equivalent with respect to safety. However, they are not behaviorally equivalent. In behavioral equivalence we are looking for simulation of actual behavior, so that what transpires in one system is accurately mimicked in the other. In other words behavioral equivalence requires equivalence of actual behavior, whereas definition 5 only requires equivalence of worst-case behavior.

5 MULTILEVEL SECURITY MODELS

The Bell-LaPadula (BLP) model [3] is a well-known access control model for multilevel security policies, most often applied in the military. In this Section we show how BLP is subsumed by SPM. This construction demonstrates that the traditional black-and-white distinction between mandatory and discretionary controls in the Bell-LaPadula model, has an alternate expression in SPM in terms of constraints on the propagation of access rights. The SPM viewpoint has the advantage of providing explicit machinery for formulating policies "in between" these two extremes.¶

There has been recent controversy about exactly what the rules of BLP are [5, 24, 25, 27]. Moreover, since its original publication the model has been modified and reformulated in several ways in its application to specific design and implementation projects [18, 19]. Nevertheless most versions of the model are closely related and there is a clearly identifiable common core.

The key component in all versions of BLP is a lattice of *security levels*, usually derived from the military classification system [12, 18]. Each subject and object is assigned a level from this lattice. Access rights are represented in an access matrix, and the model specifies rules by which this matrix can be modified. The rules are open ended, in the sense there is no formal constraint on what a rule might be (other than that it requires authorization by the current access matrix and transforms the access matrix to a new state). In practice the rules typically involve constraints on the relative security levels of subjects and objects pertaining to that operation.

¶BLP can be extended to accommodate these "in between" policies, such as done in [26]. The point is that BLP needs to be extended for this purpose, whereas SPM already has the necessary mechanisms.

The controversy about the model stems from the open-ended nature of the rules, since rules that are intuitively insecure can be defined [24]. Rules which change the security levels of subjects and objects are particularly troublesome in this respect. Versions of the model in which these security levels are constant and cannot be changed are said to satisfy the *tranquility* requirement. Most practically used versions of the model do require tranquility. Sometimes the tranquility requirement is slightly relaxed to allow changes in security levels to be effected by some designated security officer. We shall examine how to accommodate such relaxations of tranquility within SPM at the end of this Section.

In this Section we consider two versions of the BLP model, both with strong tranquility requirements. Our first version, defined below, is adapted from Pittelli [30] who showed that the BLP model he considered is an instance of the access matrix model as formalized by Harrison, Ruzzo and Ullman [14]. We show that with tranquility this model can be expressed as an SPM scheme.

Definition 6 *The BLP model with tranquility defines a system as follows.*

1. $\Sigma = \{S_1, \ldots, S_m\}$, the set of subjects.
2. $\Omega = \{O_1, \ldots, O_n\}$, the set of objects where $\Sigma \cap \Omega = \phi$.
3. $\Lambda = \{\lambda_1, \ldots, \lambda_o\}$, the lattice of security levels with dominance relation $\sqsupseteq$.
4. $\lambda : \Sigma \cup \Omega \rightarrow \Lambda$, the current security level of subjects and objects.
5. $\lambda_{max} : \Sigma \rightarrow \Lambda$, the maximum security level of subjects (i.e., $\lambda_{max}(S_i) \sqsupseteq \lambda(S_i)$).
6. $\lambda_{max}(S_i)$, $\lambda(O_j)$ are constants (tranquility).
7. $R = \{r, w, o\}$, the set of access rights (read, write, own).
8. An m×n discretionary access matrix M, with $M[i,j] \subseteq R$ specifying the discretionary access rights of S_i to O_j.
9. Subject S_i can create object O_j with arbitrary $\lambda(O_j)$. Immediately after creation $o \in M[i,j]$, that is the creator is the owner of the object.
10. The owner of an object can give read and write access to that object to another subject. That is, if $o \in M[i,j]$ then S_i can enter r or w in M[k,j] for any k.
11. An m×n current access matrix B, with $B[i,j] \subseteq \{r,w\}$, specifying the current access rights of S_i to O_j determined for r and w as follows.[‖]

[‖] In the original BLP formulation B[i,j] is a subset of B[i,j] as defined here. This is because rights are entered in B only as per the actual accesses attempted by subjects. In other words the "if and only if" ($\Leftrightarrow$) in the two conditions enumerated here is actually an "only if" ($\Rightarrow$) in the original BLP model [3]. The "if and only if" formulation we have chosen is slightly simpler to deal with, although it is possible to simulate the original BLP "only if" formulation if so desired.

$$\begin{array}{rcll} r{\in}B[i,j] & \Leftrightarrow & r{\in}M[i,j] \wedge \lambda(S_i) \sqsupseteq \lambda(O_j) & \text{Simple security} \\ w{\in}B[i,j] & \Leftrightarrow & w{\in}M[i,j] \wedge \lambda(S_i) \sqsubseteq \lambda(O_j) & \text{Star-property} \end{array}$$

There is a fixed set of subjects. The current security level of a subject is given by λ and can change so long as it is dominated by the subject's maximum security level given by λ_{max}. Subjects are allowed to create objects, and on doing so the creator becomes the owner of the created object. Each object has a security level assigned at the time of creation and given by λ. Tranquility implies that this level cannot change. Versions of BLP without tranquility usually have a security officer subject who can change λ for objects and possibly λ_{max} for subjects. The potential dangers of unrestrained non-tranquility are demonstrated in [24]. The owner of an object has discretion regarding who may access that object. However, access can be exercised only if it is consistent with simple security and the star-property. The star-property is also called the confinement property. Sometimes append and execute rights are also defined. We have dropped these for simplicity since these could be handled in much the same way as read and write in our construction.

Before proceeding further it is worth clarifying a point of terminology. What we are calling "security levels" or simply "levels," that is the elements of the security lattice, are often called "classifications" or "clearances" in the literature. The term "classification" is typically used for objects while subjects have "clearances." Moreover, military classifications are derived by combining a linearly ordered level and a set of compartments or caveats. The security lattice is derived by combining the linear ordering on levels with the subset relation on compartments. In the formalism it is irrelevant how the elements of the lattice are derived, so we can simply begin with a given lattice whose elements we call levels. Actually, the levels need not even constitute a lattice. For access control models it suffices that the levels be partially ordered.

One difficulty in constructing an SPM scheme equivalent to BLP is that the current security level of a BLP subject can change, resulting in changes in the current access matrix B. Lowering $\lambda(S_i)$ shrinks the set of objects that S_i can read while expanding the set of objects which S_i can write. Similarly raising $\lambda(S_i)$ expands the set of objects that S_i can read while shrinking the set of objects which S_i can write. This non-monotonic behavior implies that a BLP subject cannot be modeled as a single SPM subject. We circumvent this problem by mapping a BLP subject with varying λ to a set of SPM subjects, each with a fixed security level. Specifically the BLP subject S_i is mapped to the set $\{S_i\lambda_l |\ \lambda_{max}(S_i) \sqsupseteq \lambda_l\}$ of SPM subjects. Each SPM subject $S_i\lambda_l$ has the fixed security level λ_l and is of the SPM type $\sigma\lambda_l$. The idea is that $S_i\lambda_l$ simulates the BLP subject S_i when $\lambda(S_i){=}\lambda_l$. SPM subjects derived from the same BLP subject in this manner are said to be *cohorts*. The connection between cohorts is maintained by setting up cohort links between every pair of cohorts. This allows ownership of an object created by a BLP subject to be shared among the SPM cohorts for that BLP subject.

In our construction each BLP object is mapped to an SPM subject. Specifically, a BLP object with current security level λ_l is mapped to an SPM subject of type $o\lambda_l$. These SPM subjects are passive entities which cannot initiate any operations. The reason they are subjects is they possess tickets with the rc and wc rights for themselves. These "self tickets" are useful at various places in the simulation. They allow us, for instance, to conveniently specify that the SPM cohorts that own an object can obtain read and write tickets for that object, provided simple security and the star-property are not violated.

The second difficulty in our construction arises from the ability of a BLP subject S_i to create a BLP object O_j with arbitrary $\lambda(O_j)$. Now consider what happens if $\lambda_{max}(S_i) \not\supseteq \lambda(O_j)$. By simple security the creator S_i cannot read the created object O_j. However by virtue of being the owner, S_i has the ability to give read access for O_j to other subjects. In our simulation at least one of the SPM cohorts of S_i, say $S_i\lambda_l$, should be able to give read access to O_j to cohorts of other BLP subjects. But this requires that $S_i\lambda_l$ possess the O_j/rc ticket and thereby have read access to O_j, contrary to simple security. This situation appears to indicate an inherent limitation of the SPM copy operation, i.e., SPM requires a subject to possess a right before that right can be given to another subject. Similar problems arise in the following situations where S_i is the creator of O_j.

1. Let $\lambda_{max}(S_i) \sqsupseteq \lambda(O_j)$ where $\lambda(O_j)$ is not the bottom element of the lattice. By simple security, if $\lambda(O_j) \sqsupset \lambda(S_i) = \lambda_l$, the SPM cohort $S_i\lambda_l$ should not be able to read O_j. However as the owner $S_i\lambda_l$ should have the ability to give read access for O_j to other subjects.

2. Let $\lambda_{max}(S_i) \sqsupset \lambda(O_j)$. By the star-property, if $\lambda(S_i) = \lambda_l \sqsupset \lambda(O_j)$, the SPM cohort $S_i\lambda_l$ should not be able to write O_j. However as the owner $S_i\lambda_l$ should have the ability to give write access for O_j to other subjects.

There is an elegant, and quite general, technique by which SPM gets around this apparent limitation of its copy operation, by means of links and filter functions. The key idea is to introduce new right symbols $\hat{r}:c$ and $\hat{w}:c$, which control the ability to give read and write access respectively to other subjects. These right symbols are converted to r and w respectively if so allowed by simple security and the star-property.

These considerations lead us to define the following scheme.

Scheme 2 *BLP with tranquility.*

1. TS = $\{\sigma\lambda_i, o\lambda_i | \ \lambda_i \in \Lambda\}$, TO = ϕ

   SPM subjects of type $\sigma\lambda_i$ simulate BLP subjects with current level λ_i. SPM subjects of type $o\lambda_i$ simulate BLP objects with current level λ_i.

2. RI = {r : c, w : c}, RC = {o : c, k : c, $\hat{\text{r}}$: c, $\hat{\text{w}}$: c}

   r, w, and o are the original BLP rights; k, $\hat{\text{r}}$ and $\hat{\text{w}}$ are artifacts of the simulation

3. $\text{link}_{\text{u}}(\text{U,V}) \equiv \textbf{true}$
   $\text{link}_{\text{o}}(\text{U,V}) \equiv \text{U/o} \in \text{dom(V)}$
   $\text{link}_{\text{k}}(\text{U,V}) \equiv \text{U/k} \in \text{dom(V)} \wedge \text{V/k} \in \text{dom(U)}$
   $\text{link}_{\hat{\text{r}}}(\text{U,V}) \equiv \text{U/}\hat{\text{r}} \in \text{dom(V)}$
   $\text{link}_{\hat{\text{w}}}(\text{U,V}) \equiv \text{U/}\hat{\text{w}} \in \text{dom(V)}$

   The subscripts on these links have the following mnemonic significance: u for universal, o for owner, k for cohort, $\hat{\text{r}}$ for discretionary read access, and $\hat{\text{w}}$ for discretionary write access.

4. Undefined values of the filter functions are assumed by default to be ϕ.

   $f_u(\sigma\lambda_{\text{i}}, \sigma\lambda_{\text{j}}) = \{o\lambda_{\text{l}}/\hat{\text{r}}\hat{\text{w}} |\ \lambda_{\text{l}} \in \Lambda\}$

   $f_k(\sigma\lambda_{\text{i}}, \sigma\lambda_{\text{j}}) = \{o\lambda_{\text{l}}/\text{oc} |\ \lambda_{\text{l}} \in \Lambda\}$

   $f_o(o\lambda_{\text{i}}, \sigma\lambda_{\text{j}}) = o\lambda_{\text{i}}/\hat{\text{r}}\hat{\text{w}}\text{c}$

   $f_{\hat{\text{r}}}(o\lambda_{\text{i}}, \sigma\lambda_{\text{j}}) = \textbf{if } \lambda_{\text{i}} \sqsubseteq \lambda_{\text{j}} \textbf{ then } o\lambda_{\text{i}}/\text{r} \textbf{ else } \phi$

   $f_{\hat{\text{w}}}(o\lambda_{\text{i}}, \sigma\lambda_{\text{j}}) = \textbf{if } \lambda_{\text{i}} \sqsupseteq \lambda_{\text{j}} \textbf{ then } o\lambda_{\text{i}}/\text{w} \textbf{ else } \phi$

5. $cc(\sigma\lambda_{\text{i}}) = \{o\lambda_{\text{j}} |\ \lambda_{\text{j}} \in \Lambda\}$

6. $cr_p(\sigma\lambda_{\text{i}}, o\lambda_{\text{j}}) = \textit{child}/\text{oc}$,
   $cr_c(\sigma\lambda_{\text{i}}, o\lambda_{\text{j}}) = \textit{child}/\text{rw}\hat{\text{r}}\hat{\text{w}}\text{c}$

The simple security and star properties of BLP are respectively enforced by $f_{\hat{\text{r}}}$ and $f_{\hat{\text{w}}}$. Discretionary control over access to a created entity is enforced by f_k, f_o, f_u and the create-rules. Note that cc is acyclic and sparse so safety for this scheme is decidable in polynomial time by the technique of Section 3. This is in contrast to Pittelli's instantiation of BLP in HRU [30] where the resulting HRU system does not fall within the decidable cases of [14] (because Pittelli's construction uses multi-conditional HRU commands).

To complete the construction we define the initial state to be as follows, where SUB is the initial set of subjects in the SPM system.

1. $\text{SUB} = \Omega \cup \{\text{S}_{\text{i}}\lambda_{\text{l}} \in \Sigma \times \Lambda |\ \lambda_{\max}(\text{S}_{\text{i}}) \sqsupseteq \lambda_{\text{l}}\}$

2. $\textit{type}(\text{O}_{\text{j}}) = o\lambda_{\text{l}}$, where $\lambda(\text{O}_{\text{j}}) = \lambda_{\text{l}}$
   $\textit{type}(\text{S}_{\text{i}}\lambda_{\text{l}}) = \sigma\lambda_{\text{l}}$, where $\lambda_{\max}(\text{S}_{\text{i}}) \sqsupseteq \lambda_{\text{l}}$

3. $\text{dom}(\text{O}_{\text{i}}) = \text{O}_{\text{i}}/\text{rw}\hat{\text{r}}\hat{\text{w}}\text{c}$

$$\begin{aligned}\mathrm{dom}(S_i\lambda_j) = \{&S_i\lambda_l/k|\ S_i\lambda_l \in SUB\} \cup \{O_l/o\hat{r}\hat{w}c|\ o \in M[i,l]\} \cup \\ &\{O_l/\hat{r}|\ r \in M[i,l]\} \cup \{O_l/\hat{w}|\ w \in M[i,l]\} \cup \\ &\{O_l/r|\ r \in B[i,l]\} \cup \{O_l/w|\ w \in B[i,l]\}\end{aligned}$$

Each subject gets the k right for all its cohorts. The entries of the BLP M matrix are represented by the o, $\hat{r}$ and $\hat{w}$ rights. The entries of the BLP B matrix are represented by the r and w rights.

The correspondence between the BLP model of definition 6 and its realization by the above SPM system is almost self-evident. Scheme 2 expresses the BLP rules in a natural and intuitive manner without straining the SPM notation. Thus BLP is an instance of SPM not merely in some obscure theoretical sense, but actually in an intuitively meaningful manner. We now establish the following result.

Theorem 1 *The BLP system of definition 6 and the SPM scheme 2 with its specified initial state are equivalent.*

Proof: To establish equivalence between the two systems we first show how each control operation in the BLP system is simulated by a sequence of control operations in the SPM system, which have the same net effect regarding distribution of read, write and owner rights. There are four control operations in BLP shown in table 1 along with the SPM operations which simulate these. The general idea is that any operation executed by the BLP subject S_i with $\lambda(S_i)=\lambda_p$ is simulated by a sequence of operations initiated by the SPM cohort $S_i\lambda_p$. Creation of object O_j by subject S_i is simulated by letting $S_i\lambda_p$ create O_j. Ownership of O_j is transferred to the other cohorts of S_i using $link_k$'s. These cohort links are established in constructing the initial state and are static, since the cohort right k cannot be copied. So it is exactly the SPM cohorts of S_i who can share ownership of O_j. Ownership in BLP implies the right to give read or write access to other subjects. This is simulated by letting each SPM cohort obtain $O_j/\hat{r}\hat{w}c$ from $\mathrm{dom}(O_j)$ using $link_o$. In this manner each SPM cohort of S_i has the authorization to give read and write access for its owned objects to other subjects. The BLP operation of S_i giving S_k read access to O_j is simulated in SPM by letting $S_i\lambda_p$ copy $O_j/\hat{r}$ from its own domain to every SPM cohort of S_k. Cohorts of S_k whose level dominates $\lambda(O_j)$ can then copy O_j/r from $\mathrm{dom}(O_j)$ using $link_{\hat{r}}$. Cohorts of S_k whose level does not dominate $\lambda(O_j)$ are denied this ability by $f_{\hat{r}}$. So simple security is enforced in the SPM simulation by $f_{\hat{r}}$. The BLP operation of S_i giving S_k write access to O_j is similarly simulated in SPM with the star-property enforced by $f_{\hat{w}}$. The BLP operation of subject S_i changing its current level is ignored in SPM. In BLP this operation has the effect of modifying row i of the current access matrix B to preserve simple security and the star-property. This is accounted for in the SPM simulation by the manner in which the SPM cohorts of S_i at different levels obtain read and write tickets.

To complete the demonstration of equivalence it remains to show the converse property that each SPM control operation can be simulated by a BLP operation

Operation	BLP System [Let $\lambda(S_i)=\lambda_p$, $\lambda(O_j)=\lambda_q$]	SPM Simulation
1	S_i creates O_j	$S_i\lambda_p$ creates O_j of type $o\lambda_q$ and copies O_j/oc to its cohorts $S_i\lambda_l$ over $link_k$. Each cohort copies $O_j/\hat{r}\hat{w}c$ from O_j over $link_o$.
Net Effect	$o\in M[i,j]$	$O_j/o\hat{r}\hat{w}c\in dom(S_i\lambda_l)$ for all λ_l
2	S_i enters r in M[k,j]	$S_i\lambda_p$ copies $O_j/\hat{r}$ over $link_u$ to all cohorts of S_k. Each cohort $S_k\lambda_l$ copies O_j/r from O_j over $link_{\hat{r}}$ if allowed by $f_{\hat{r}}$.
Net Effect	$r\in M[k,j]$ $r\in B[k,j]$ if $\lambda(S_k)\sqsupseteq\lambda_q$	$O_j/\hat{r}\in dom(S_k\lambda_l)$ for all λ_l $O_j/r\in dom(S_k\lambda_l)$ if $\lambda_l\sqsupseteq\lambda_q$
3	S_i enters w in M[k,j]	$S_i\lambda_p$ copies $O_j/\hat{w}$ over $link_u$ to all cohorts of S_k. Each cohort $S_k\lambda_l$ copies O_j/w from O_j over $link_{\hat{w}}$ if allowed by $f_{\hat{w}}$.
Net Effect	$w\in M[k,j]$ $w\in B[k,j]$ if $\lambda(S_k)\sqsubseteq\lambda_q$	$O_j/\hat{w}\in dom(S_k\lambda_l)$ for all λ_l $O_j/w\in dom(S_k\lambda_l)$ if $\lambda_l\sqsubseteq\lambda_q$
4	S_i changes $\lambda(S_i)$	Ignored.
Net Effect	Row i of B is changed to preserve simple-security and the star-property.	Accounted for in simulating operations 2 and 3.

Table 1: Simulation of BLP Operations in SPM

Operation	SPM System	BLP Simulation
1	$S_i \lambda_p$ creates O_j of type $o\lambda_m$	S_i creates O_j with $\lambda(O_j)=\lambda_m$
Net Effect	$O_j/oc \in dom(S_i \lambda_p)$ $dom(O_j)=O_j/rw\hat{r}\hat{w}c$	$o \in M[i,j]$
2	$S_i \lambda_p$ copies $O_j/o:c$ from itself to its cohort $S_i \lambda_q$ over $link_k$	Ignored
Net Effect	$O_j/o:c \in dom(S_i \lambda_q)$	Accounted for in operation 1
3	$S_i \lambda_p$ copies $O_j/\hat{r}:c$ from O_j to itself over $link_o$	S_i enters r in M[i,j]
Net Effect	$O_j/\hat{r}:c \in dom(S_i \lambda_p)$	$r \in M[i,j]$; $r \in B[i,j]$ if $\lambda(S_i) \sqsupseteq \lambda(O_j)$
4	$S_i \lambda_p$ copies $O_j/\hat{w}:c$ from O_j to itself over $link_o$	S_i enters w in M[i,j]
Net Effect	$O_j/\hat{w}:c \in dom(S_i \lambda_p)$	$w \in M[i,j]$; $w \in B[i,j]$ if $\lambda(S_i) \sqsubseteq \lambda(O_j)$
5	$S_i \lambda_p$ copies $O_j/\hat{r}$ from itself to $S_k \lambda_q$ over $link_u$	S_i enters r in M[k,j]
Net Effect	$O_j/\hat{r} \in dom(S_k \lambda_q)$	$r \in M[k,j]$; $r \in B[k,j]$ if $\lambda(S_k) \sqsupseteq \lambda(O_j)$
6	$S_i \lambda_p$ copies $O_j/\hat{w}$ from itself to $S_k \lambda_q$ over $link_u$	S_i enters w in M[k,j]
Net Effect	$O_j/\hat{w} \in dom(S_k \lambda_q)$	$w \in M[k,j]$; $w \in B[k,j]$ if $\lambda(S_k) \sqsubseteq \lambda(O_j)$
7	$S_i \lambda_p$ copies O_j/r from O_j to itself over $link_{\hat{r}}$	Ignored
Net Effect	$O_j/r \in dom(S_i \lambda_p)$	Accounted for in operations 3 and 5
8	$S_i \lambda_p$ copies O_j/w from O_j to itself over $link_{\hat{w}}$	Ignored
Net Effect	$O_j/w \in dom(S_i \lambda_p)$	Accounted for in operations 4 and 6

Table 2: Simulation of SPM Operations in BLP

having the same net effect. There are eight control operations in the SPM system as enumerated in table 2. Of these only operations 1, 3, 4, 5 and 6, i.e., object creation and copying of $\hat{r}$ and $\hat{w}$, are explicitly simulated in BLP; respectively as object creation and giving of read and write access. SPM operation 2 establishes shared ownership of an object between SPM cohorts and needs no simulation. Operation 7 and 8 respectively convert $\hat{r}$ and $\hat{w}$ rights to r and w. These are accounted for in BLP by the current access matrix B which is automatically changed to preserve simple security and the star-property whenever the current security level of a subject changes. □

Next we consider a version of BLP with subjects who are given more power than allowed by the simple security and star properties of definition 6. This model is based on the security properties of [4] in the so-called network interpretation of multi-level security. The basic idea is to allow subjects to violate simple security and star properties in a controlled manner. This is achieved by associating a pair of security levels $\lambda_{\mathrm{vmax}}(\mathrm{S_i})$ and $\lambda_{\mathrm{amin}}(\mathrm{S_i})$ with each subject $\mathrm{S_i}$, with the subscripts respectively read as view-maximum and alter-minimum. It is required that $\lambda_{\mathrm{vmax}}(\mathrm{S_i}) \sqsupseteq \lambda_{\mathrm{amin}}(\mathrm{S_i})$. The range of a subject is the set of levels bounded by λ_{vmax} and λ_{amin} as follows.

$$range(\mathrm{S_i}) = \{\lambda_{\mathrm{k}} |\ \lambda_{\mathrm{vmax}}(\mathrm{S_i}) \sqsupseteq \lambda_{\mathrm{k}} \sqsupseteq \lambda_{\mathrm{amin}}(\mathrm{S_i})\}$$

Subject $\mathrm{S_i}$ is allowed to read and write objects whose security levels are in $range(\mathrm{S_i})$. Outside this range requirements similar to simple security and the star-property are stipulated as follows.

1. $\mathrm{S_i}$ can read $\mathrm{O_j}$ only if $\lambda_{\mathrm{amin}}(\mathrm{S_i}) \sqsupseteq \lambda(\mathrm{O_j})$.
2. $\mathrm{S_i}$ can write $\mathrm{O_j}$ only if $\lambda(\mathrm{O_j}) \sqsupseteq \lambda_{\mathrm{vmax}}(\mathrm{S_i})$.

Since $\lambda_{\mathrm{vmax}} \sqsupseteq \lambda_{\mathrm{amin}}$ we can view these requirements as respectively generalizing simple security to require that all levels in $range(\mathrm{S_i})$ dominate $\lambda(\mathrm{O_j})$, and the star-property to require that $\lambda(\mathrm{O_j})$ dominates all levels in $range(\mathrm{S_i})$. Within $range(\mathrm{S_i})$ simple security and the star-property are not enforced.

The SPM simulation in this case is actually simpler than the previous one since there is no notion of a changing current security level for subjects. So, there is no need for SPM cohorts. Each BLP subject is mapped to a single SPM subject whose type is determined by λ_{vmax} and λ_{amin} of the BLP subject. We have the following scheme.

Scheme 3 *BLP network interpretation with tranquility.*

1. $\mathrm{TS} = \{\sigma\lambda_{\mathrm{i}}\lambda_{\mathrm{j}} |\ \lambda_{\mathrm{i}} \sqsupseteq \lambda_{\mathrm{j}}\} \cup \{o\lambda_{\mathrm{k}} |\ \lambda_{\mathrm{k}} \in \Lambda\}$, $\mathrm{TO} = \phi$

   SPM subjects of type $\sigma\lambda_{\mathrm{i}}\lambda_{\mathrm{j}}$ model BLP subjects with $\lambda_{\mathrm{vmax}} = \lambda_{\mathrm{i}}$ and $\lambda_{\mathrm{amin}} = \lambda_{\mathrm{j}}$. SPM subjects of type $o\lambda_{\mathrm{i}}$ simulate BLP objects with current level λ_{i}.

2. $\mathrm{RI} = \{\mathrm{r:c},\ \mathrm{w:c}\}$, $\mathrm{RC} = \{\mathrm{o:c},\ \hat{\mathrm{r}}\mathrm{:c},\ \hat{\mathrm{w}}\mathrm{:c}\}$

   r, w, and o are the original BLP rights; $\hat{\mathrm{r}}$ and $\hat{\mathrm{w}}$ are artifacts of the simulation

3. $\mathrm{link_u(U,V)} \equiv \mathbf{true}$
   $\mathrm{link_o(U,V)} \equiv \mathrm{U/o}\in\mathrm{dom(V)}$
   $\mathrm{link}_{\hat{\mathrm{r}}}\mathrm{(U,V)} \equiv \mathrm{U}/\hat{\mathrm{r}}\in\mathrm{dom(V)}$
   $\mathrm{link}_{\hat{\mathrm{w}}}\mathrm{(U,V)} \equiv \mathrm{U}/\hat{\mathrm{w}}\in\mathrm{dom(V)}$

   The subscripts on these links have the following mnemonic significance: u for universal, o for owner, $\hat{\mathrm{r}}$ for discretionary read access, and $\hat{\mathrm{w}}$ for discretionary write access.

4. Undefined values of the filter functions are assumed by default to be ϕ.

   $f_u(\sigma\lambda_\mathrm{i}\lambda_\mathrm{j},\ \sigma\lambda_\mathrm{k}\lambda_\mathrm{l}) = \{o\lambda_\mathrm{m}/\hat{\mathrm{r}}\hat{\mathrm{w}}|\ \lambda_\mathrm{m}\in\Lambda\}$

   $f_o(o\lambda_\mathrm{i},\ \sigma\lambda_\mathrm{j}\lambda_\mathrm{k}) = o\lambda_\mathrm{i}/\hat{\mathrm{r}}\hat{\mathrm{w}}\mathrm{c}$

   $f_{\hat{\mathrm{r}}}(o\lambda_\mathrm{i},\ \sigma\lambda_\mathrm{j}\lambda_\mathrm{k}) = \mathbf{if}\ \lambda_\mathrm{j}\sqsupseteq\lambda_\mathrm{i}\sqsupseteq\lambda_\mathrm{k} \vee \lambda_\mathrm{k}\sqsupseteq\lambda_\mathrm{i}\ \mathbf{then}\ o\lambda_\mathrm{i}/\mathrm{r}\ \mathbf{else}\ \phi$

   $f_{\hat{\mathrm{w}}}(o\lambda_\mathrm{i},\ \sigma\lambda_\mathrm{j}\lambda_\mathrm{k}) = \mathbf{if}\ \lambda_\mathrm{j}\sqsupseteq\lambda_\mathrm{i}\sqsupseteq\lambda_\mathrm{k} \vee \lambda_\mathrm{i}\sqsupseteq\lambda_\mathrm{j}\ \mathbf{then}\ o\lambda_\mathrm{i}/\mathrm{w}\ \mathbf{else}\ \phi$

5. $cc(\sigma\lambda_\mathrm{i}\lambda_\mathrm{j}) = \{o\lambda_\mathrm{k}|\ \lambda_\mathrm{k}\in\Lambda\}$

6. $cr_p(\sigma\lambda_\mathrm{i}\lambda_\mathrm{j},\ o\lambda_\mathrm{j}) = \mathit{child}/\mathrm{oc}$,
   $cr_c(\sigma\lambda_\mathrm{i}\lambda_\mathrm{j},\ o\lambda_\mathrm{j}) = \mathit{child}/\mathrm{rw}\hat{\mathrm{r}}\hat{\mathrm{w}}\mathrm{c}$

The initial state for the SPM system is as follows.

1. $\mathrm{SUB} = \Omega \cup \{\mathrm{S_i}\lambda_\mathrm{j}\lambda_\mathrm{k}|\ \mathrm{S_i}\in\Sigma \wedge \lambda_\mathrm{vmax}(\mathrm{S_i})=\lambda_\mathrm{j} \wedge \lambda_\mathrm{amin}(\mathrm{S_i})=\lambda_\mathrm{k}\}$

2. $\mathit{type}(\mathrm{O_j}) = o\lambda_\mathrm{l}$, where $\lambda(\mathrm{O_j})=\lambda_\mathrm{l}$
   $\mathit{type}(\mathrm{S_i}\lambda_\mathrm{j}\lambda_\mathrm{k}) = \sigma\lambda_\mathrm{j}\lambda_\mathrm{k}$

3. $\mathrm{dom(O_i)} = \mathrm{O_i/rw}\hat{\mathrm{r}}\hat{\mathrm{w}}\mathrm{c}$

   $$\begin{aligned}\mathrm{dom}(\mathrm{S_i}\lambda_\mathrm{j}\lambda_\mathrm{k}) = {} & \{\mathrm{O_l}/o\hat{\mathrm{r}}\hat{\mathrm{w}}\mathrm{c}|\ o\in\mathrm{M[i,l]}\} \cup \\ & \{\mathrm{O_l}/\hat{\mathrm{r}}|\ r\in\mathrm{M[i,l]}\} \cup \{\mathrm{O_l}/\hat{\mathrm{w}}|\ w\in\mathrm{M[i,l]}\} \cup \\ & \{\mathrm{O_l}/\mathrm{r}|\ r\in\mathrm{B[i,l]}\} \cup \{\mathrm{O_l}/\mathrm{w}|\ w\in\mathrm{B[i,l]}\}\end{aligned}$$

We can establish equivalence between the BLP and SPM systems as was done in theorem 1 for the construction of scheme 2. Because of the absence of cohorts the proof will be simpler in this case.

Next consider the Biba integrity model [6] which is the exact dual of BLP with the aim of controlling unauthorized modification of information rather than unauthorized disclosure. Its definition is obtained from definition 6 by making the following replacements.

$$\begin{array}{ccc} \sqsupseteq & \leftrightarrow & \sqsubseteq \\ \sqsupset & \leftrightarrow & \sqsubset \\ \lambda_{max} & \leftrightarrow & \lambda_{min} \end{array}$$

The lattice of security levels is replaced by a lattice of integrity levels. Each subject has a minimum integrity level designated by λ_{min}. The current integrity level of the subject must dominate λ_{min} at all times. Simple security and the star-property are replaced by the following.

$$\begin{array}{rcll} r{\in}B[i,j] & \Leftrightarrow & r{\in}M[i,j] \wedge \lambda(S_i) \sqsubseteq \lambda(O_j) & \text{Simple integrity} \\ w{\in}B[i,j] & \Leftrightarrow & w{\in}M[i,j] \wedge \lambda(S_i) \sqsupseteq \lambda(O_j) & \text{Integrity star-property} \end{array}$$

That is a subject is only allowed to read objects of the same or higher integrity as itself and to write objects of the same or lower integrity. Given the constructions for BLP it is no surprise that the Biba model can be instantiated in SPM. We simply need to reverse the dominance relations in $f_{\hat{r}}$ and $f_{\hat{w}}$ of scheme 2. Similarly by reversing the dominance relation of the BLP construction we obtain the proper initial state. Lee [20] and Schockley [39] formulate integrity models which are duals of the BLP network interpretation. These can be expressed in SPM as the dual of scheme 3. It is also possible for the Biba and BLP models to coexist in a single system. If the same lattice is used for both models, their coexistence implies that a subject can read or write only at its current level. More generally the two models can coexist with independent lattices, so each subject and object has a security level and an independent integrity level. Such coexistence can be easily modeled in SPM by combining the rules of the two models. We conclude that the mandatory controls of the BLP model for non-disclosure and of the Biba model for integrity are special cases of the more general mandatory controls of SPM.

Finally it is worth considering what kind of non-tranquility can be accommodated in a monotonic manner in SPM. To be specific consider scheme 2. Non-tranquility in BLP is usually specified by including a security officer subject who has the authority to enroll new subjects, change λ_{max} of existing subjects, and change λ of existing objects. In SPM we can define a new subject type *sec-off* with one instance in the initial state to model the security officer. Creation of new subjects can be simulated by allowing the security officer to create a collection of SPM cohorts and giving him the ability to connect them by $link_k$'s. Changing $\lambda_{max}(S_i)$ from λ_p to λ_q can be similarly modeled so long as $\lambda_q \sqsupseteq \lambda_p$. The security officer simply introduces new cohorts of S_i at levels dominated by λ_q but not by λ_p, and connects these to the existing cohorts of S_i and each other by $link_k$'s. If $\lambda_q \not\sqsupseteq \lambda_p$ we need to delete some of the existing cohorts. Since this might delete some cohorts which existed in the initial state we would need to treat this as being a different SPM system. Changing the security level of an object requires revocation of read and write privileges to preserve simple security and the star-property, and would again have to be treated as a transition to a different SPM system. Note that with this kind of unrestricted power given to a security officer

there really is no safety in the system, unless we assume the security officer does not change security levels arbitrarily. So for purpose of safety analysis one does assume some form of tranquility.

6 TAKE-GRANT MODELS

Of all the models discussed in this paper, take-grant is closest in viewpoint to SPM. Its SPM simulation is, therefore, a very natural one. Take-grant derives its name from its two control rights t (take) and g (grant). Several papers have been published on this model, including [16, 21, 40]. Inevitably there are slight differences in the precise definition of the model in these papers. Our presentation follows Snyder's review [40] of the model most closely. Several variations of take-grant have also been proposed [9, 23]. These variations are also easily specified in SPM.

The SPM simulation of the basic take-grant model is given as schemes VI through VIII of [34]. In this paper we extend the construction to accommodate analysis of theft in take-grant [41]. This shows how assumptions about behavior are easily expressed in SPM.

Transfer of information in the take-grant model has been analyzed by Bishop and Snyder [7]. The control operations used for this purpose are a special case of grammatical protection systems which are modeled in SPM in the next Section. Analysis of combined authority and information transfer and theft [8] can be accommodated in SPM by combining the constructions of this Section with those of the following Section. In this way we are able to cover the analysis results of take-grant within the analysis framework of Section 3 for SPM.

Let us briefly review scheme VIII of [34] which is equivalent** to the so called subject-object version of take-grant [16]. In this scheme there are two types of subjects: *as* for active subjects and *ps* for passive subjects. A passive subject cannot execute operations and is merely a repository for tickets. In SPM terms, the take-grant model defines two link predicates as follows, where the subscripts have obvious mnemonic significance.

$$\text{link}_g(U,V) \equiv V/g{\in}\text{dom}(U)$$
$$\text{link}_t(U,V) \equiv U/t{\in}\text{dom}(V)$$

A link_g requires a grant capability at the source while a link_t requires a take capability

**The equivalence is not absolute since, strictly speaking, the take-grant model does not allow a subject to possess tickets for itself. It appears this restriction cannot be specified in SPM, without some drastic step such as declaring each subject to be of a distinct type unique to itself. However, as observed by Snyder [40] this is not a fundamental feature of take-grant. Moreover, the restriction is unnecessarily restrictive in that we often want a subject to possess rights for itself so as to give these to other subjects at that subject's discretion. We also need to assume that all tickets in the SPM initial state are copiable.

at the destination. A link can be exercised only if authorized by a ticket in the domain of an active subject, i.e., link_g(U,V) can be exercised only if U is active whereas link_t(U,V) can be exercised only if V is active. There is otherwise no selectivity in the copy operation. Passive subjects are not allowed to create subjects whereas active subjects can create both passive and active subjects. All this is easily specified in SPM as follows.

Scheme 4 *The take-grant model with passive subjects.*

1. TS = $\{as, ps\}$, TO = ϕ
2. RC = {t:c, g:c}, RI = some finite set disjoint from RC
3. link_g(U,V) $\equiv$ V/g$\in$dom(U)
   link_t(U,V) $\equiv$ U/t$\in$dom(V)
4. $f_g(as, [as|ps]) = \text{T}\times\text{R}$
   $f_g(ps, [as|ps]) = \phi$
   $f_t([as|ps], as) = \text{T}\times\text{R}$
   $f_t([as|ps], ps) = \phi$
5. $cc(as) = \{as, ps\}$
   $cc(ps) = \phi$
6. There is a uniform create-rule with cr_p=*child*/R and cr_c=ϕ

Here we introduce abbreviated notation to keep the scheme compact. The interpretation of $[as|ps]$ is that it is an abbreviation for all combinations of the bracketed terms. For example in the above case the verbose definition of f_g is understood to be as follows.

$$\begin{aligned} f_g(as, as) &= \text{T}\times\text{R} \\ f_g(as, ps) &= \text{T}\times\text{R} \\ f_g(ps, as) &= \phi \\ f_g(ps, ps) &= \phi \end{aligned}$$

In addition to its compactness this notation is useful in highlighting the similarities and differences between types.

The create-rule in scheme 4 is not attenuating and there is a loop in cc due to $as \in cc(as)$. So as it stands the scheme is not acyclic attenuating and thereby does not fall within the known decidable cases of SPM. However scheme 4 can easily be modified to be attenuating by the technique described in [34] of distinguishing the initial set of subjects from those created subsequently. The remaining schemes of this

Section are all non-attenuating in the same way as scheme 4. This technique can be used in all constructions of this paper to obtain an equivalent attenuating version.

We now show how the notion of theft as defined by Snyder [41] for the take-grant model can be specified by an SPM scheme. This notion assumes that certain subjects will not carry out particular operations even though they are authorized to do so. That is these subjects are trusted not to cooperate in some specific way for propagating tickets. There are numerous assumptions about behavior that one could make. Snyder analyzes a particular set of assumptions, but would need to carry out similar and perhaps more complicated analysis if these assumptions are changed. One of the great advantages of SPM is that assumptions about the behavior of subjects can be easily specified as part of a scheme. To demonstrate this we show how the specific assumptions used by Snyder are stated in SPM.

Snyder's concept of theft is that a ticket Y/x is stolen by a subject U provided the following conditions hold.

1. U does not possess Y/x in the initial state.
2. Subjects who possess Y/xc in the initial state do not grant Y/x to any other subject, i.e., subjects possessing Y/xc are trusted not to give it away.
3. There is a reachable state with Y/x $\in$ dom(U).

In other words theft is said to occur if U is able to obtain Y/x, even if subjects possessing Y/xc in the initial state do not give it away to anybody. We can model these assumptions in SPM by distinguishing different types of subjects. First we distinguish trusted subjects from untrusted ones. Since passive subjects cannot exercise the grant operation, this distinction applies only to active subjects. Next we need to distinguish entities that are confidential from those that are non-confidential. The assumed behavior of trusted subjects applies only to confidential entities. They are free to grant tickets for non-confidential entities, but are constrained by their behavior in granting tickets for confidential entities. The notions of trusted and confidential are independent attributes of subjects, so we need to define subjects types for all possible combinations of these as given in the top four rows of table 3. Passive subjects are unable to exercise the grant privilege, so they are inherently trusted. This gives us the two bottom rows of table 3.

Finally we identify the rights RT which will not be granted by trusted subjects for confidential entities. That is trusted subjects are assumed not to grant tickets of type $\{tcas,ucas,cps\}\times$RT even if authorized to so, but may grant tickets of type $\{tcas,ucas,cps\}\times$(R$\Leftrightarrow$RT). This is specified by setting the value of f_g from trusted subjects to all other subjects to be (T$\times$R)$\Leftrightarrow$($\{tcas,ucas,cps\}\times$RT). The value of f_g from untrusted subjects to all other subjects remains unchanged as T$\times$R. All values of f_t remain unchanged.

This results in the following scheme.

TYPE	TRUSTED	CONFIDENTIAL	ACTIVE
tcas	Yes	Yes	Yes
tnas	Yes	No	Yes
ucas	No	Yes	Yes
unas	No	No	Yes
cps	—	Yes	No
nps	—	No	No

Table 3: SPM Subject Types for Modeling Theft in Take-Grant

Scheme 5 *Theft of rights in the take-grant model.*

1. TS = $\{tcas, tnas, ucas, unas, cps, nps\}$, TO = ϕ
   Let CS = $\{tcas, ucas, cps\}$ be the set of confidential types

2. RC = {t:c, g:c}, RI = some finite set disjoint from RC
   Let RT $\subseteq$ R be the set of rights whose theft we are analyzing

3. $\text{link}_g(U,V) \equiv V/g \in \text{dom}(U)$
   $\text{link}_t(U,V) \equiv U/t \in \text{dom}(V)$

4. Let $[TS] \equiv [tcas|tnas|ucas|unas|cps|nps]$
   $f_g([tcas|tnas], [TS]) = (T \times R) \Leftrightarrow (CS \times RT)$
   $f_g([ucas|unas], [TS]) = T \times R$
   $f_g([cps|nps], [TS]) = \phi$

   $f_t([TS], [tcas|tnas|ucas|unas]) = T \times R$
   $f_t([TS], [cps|nps]) = \phi$

5. $cc([tcas|tnas|ucas|unas]) = \{unas, nps\}$
   $cc([cps|nps]) = \phi$

6. There is a uniform create-rule with cr_p=*child*/R and $cr_c = \phi$

We can as easily model a different notion of theft in which say the trusted subjects do not grant any take rights in addition to the above restriction. We simply need to change f_g as follows.

$$f_g([tcas|tnas], [TS]) = (T \times R) \Leftrightarrow (CS \times RT \cup TS \times \{t:c\})$$

Or perhaps the assumption that trusted subjects do not grant any take rights except to other trusted subjects, specified by modifying f_g as follows.

$$f_g([tcas|tnas],\ [ucas|unas|cps|nps]) = (\mathrm{T}\times\mathrm{R}) \Leftrightarrow (\mathrm{CS}\times\mathrm{RT} \cup \mathrm{TS}\times\{\mathrm{t{:}c}\})$$
$$f_g([tcas|tnas],\ [tcas|tnas]) = (\mathrm{T}\times\mathrm{R}) \Leftrightarrow (\mathrm{CS}\times\mathrm{RT})$$

The structure of SPM gives us a powerful framework for investigating the consequences of such assumptions about behavior. In the take-grant framework each of these separate notions of theft would require a separate analysis along the lines of [7, 8, 16, 21, 41]. In SPM these alternate notions require separate schemes, but the same analysis algorithm of Section 3 can be used in all cases. Moreover in the SPM framework ad hoc assumptions about behavior can be accommodated quite easily. In the limit each individual user may be treated separately for this purpose.

7 GRAMMATICAL PROTECTION SYSTEMS

Grammatical protection systems (GPS) were defined by Lipton and Budd [22] and shown to have a close relation to context-free grammars. Safety in these systems is reduced to a parsing problem which is decidable in polynomial time. The subsumption of GPS by SPM demonstrates the ability of SPM to simulate models whose control operations are at first sight quite contrary to SPM control operations. It is also significant because, in combination with the constructions of the previous Section, it allows us to accommodate notions of information and authority transfer and theft in take-grant within SPM.

There is no create operation in GPS so the system has a fixed set of subjects. The protection state of the system is visualized as a graph in which there is a directed edge labeled α from U to V, shown as U $\stackrel{\alpha}{\Leftrightarrow}$ V, if U possesses the set of rights α for V. In other words, U $\stackrel{\alpha}{\Leftrightarrow}$ V if and only if subject U possesses the tickets V/α.

The rules for changing the protection state are expressed in one of the forms indicated in table 4 where α, β and γ are non-empty sets of rights. The interpretation of these rules is straightforward. A class I rule says that if U possesses V/α and V possesses W/β then U can acquire W/γ. Similarly a class II rule says that if U possesses V/α and W possesses V/β then U can acquire W/γ. Class III and IV rules are similarly interpreted.

The relation of these systems to context-free grammars is strongest when α, β and γ are singleton sets. However the safety analysis algorithms are applicable to the more general case where they are arbitrary non-empty sets [10]. The rules for modeling transfer of information in the take-grant model [7] are actually instances of GPS rules as shown in table 5. Here r and w are the standard read and write privileges, while r′ is a pseudo-privilege denoting implicit read. So we do have a realistic interpretation for each of the rule classes. Moreover the simulation of grammatical protection systems in SPM thereby also subsumes the information transfer analysis of the take-grant model. Combined analysis of authority and information transfer and theft in take-grant can be accommodated in SPM by combining the constructions of this Section

Rule Class	Given	Add
I	$U \overset{\alpha}{\Rightarrow} V \overset{\beta}{\Rightarrow} W$	$U \overset{\gamma}{\Rightarrow} W$
II	$U \overset{\alpha}{\Rightarrow} V \overset{\beta}{\Leftarrow} W$	$U \overset{\gamma}{\Rightarrow} W$
III	$U \overset{\alpha}{\Leftarrow} V \overset{\beta}{\Rightarrow} W$	$U \overset{\gamma}{\Rightarrow} W$
IV	$U \overset{\alpha}{\Leftarrow} V \overset{\beta}{\Leftarrow} W$	$U \overset{\gamma}{\Rightarrow} W$

Table 4: Rules in Grammatical Protection Systems

Rule Class	Rule	Given	Add
I	Spy	$U \overset{r}{\Rightarrow} V \overset{r}{\Rightarrow} W$	$U \overset{r'}{\Rightarrow} W$
II	Post	$U \overset{r}{\Rightarrow} V \overset{w}{\Leftarrow} W$	$U \overset{r'}{\Rightarrow} W$
III	Pass	$U \overset{w}{\Leftarrow} V \overset{r}{\Rightarrow} W$	$U \overset{r'}{\Rightarrow} W$
IV	Find	$U \overset{w}{\Leftarrow} V \overset{w}{\Leftarrow} W$	$U \overset{r'}{\Rightarrow} W$

Table 5: Take-Grant Information Transfer Rules in GPS

with those of the previous one.

For future reference we define grammatical protection systems as follows.

Definition 7 *A grammatical protection system is defined by specifying a fixed set of subjects, a fixed set of rights and a fixed collection of rules of the form indicated in table 4.*

The general definition of grammatical protection systems actually includes the notion of subject types. The rules are typed in that U, V and W are required to be of specific types as specified for each rule. However GPS with multiple types can be reduced to GPS with a single type by introducing new right symbols which effectively encode the type information [10]. So it suffices to consider GPS without types.

GPS rules appear in many way contrary to SPM operations and offer a significant challenge for the expressive power of SPM. The major problem is that the rights introduced by a GPS rule may not be present in any subject's domain prior to applying the rule. For instance in a class I or III rule, V is required to possess W/β but U ends up with W/γ where β and γ may not be related in any way. In class II and IV rules there may be no tickets at all for W and yet U acquires W/γ.

We are able to get around this problem by simulating each GPS subject by a number of SPM subjects of different types. The collection of SPM subjects which simulate the GPS subject X are said to be cohorts of X and of each other. This

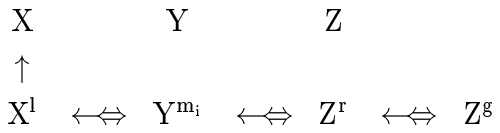

Figure 1: Simulation of GPS rule i in SPM

general idea of using several cohorts to simulate a single subject is similar to our use of cohorts in simulating the multilevel security models of Section 5. Of course the connections between the cohorts and the role they play in the simulation are now quite different. In this case the construction is much more intricate. We now explain the underlying intuition.

For a GPS system with n rules, numbered 1 through n, we define the following subject types in SPM.

$$\text{TS} = \{s,\ sg,\ sl,\ sr,\ sm_1,\ sm_2,\ \ldots,\ sm_n\}$$

The intention is that a GPS subject X be simulated by a set of SPM cohorts which has one member of each of these types, as shown below.

$$\text{SPM cohorts of X} = \{\text{X},\ \text{X}^g,\ \text{X}^l,\ \text{X}^r,\ \text{X}^{m_1},\ \ldots,\ \text{X}^{m_n}\}$$

By convention the type of each SPM cohort is s concatenated with the superscript on the cohort's name. So X^g is of type sg, X^l of type sl, and so on. If the superscript is missing, as in X, its type is simply s. The type s SPM subjects are the ones which simulate the actual domain of GPS subjects. We will establish that $\text{Y}/\alpha \in \text{dom(X)}$ in the SPM system if and only if $\text{X} \overset{\alpha}{\Leftrightarrow} \text{Y}$ in the GPS system. However dom(X) may contain tickets for subjects of type other than s, which are used for the simulation.

We regard the type s SPM cohort to be the one which "truly" simulate each GPS subject. SPM subjects of type other than s are an artifact of the simulation. The SPM cohorts X^l and X^r respectively simulate the role of the GPS subject X when a rule with X at the left end or right end is invoked. When a rule with the GPS subject X in the middle is invoked we have a different cohort of X for each rule. The role of the GPS subject X as the middle subject in rule i is simulated by its cohort X^{m_i}. In these cases the superscripts have obvious mnemonic significance. It remains to consider SPM subjects of type sg. In our construction the cohort X^g serves as a source or generator of tickets for the GPS subject X, when a rule with X at the right end is invoked (note that in table 4 it is always the subject at the left end which acquires rights for the subject at the right end). Each generator cohort X^g possesses all tickets for itself and for X, that is $\text{dom}(\text{X}^g)$ contains X^g/R as well as X/R.

Invocation of rule i in the GPS system with X, Y and Z as the left, middle and right subjects respectively is simulated as depicted in figure 1 (where each directed

edge denotes an SPM link). A sequence of links is established from Z^g to Z^r to Y^{m_i} to X^l to X. The SPM scheme ensures that the links from Z^r to Y^{m_i} and from Y^{m_i} to X^l, in this sequence, can be established if and only if rule i is authorized in the GPS system. Let X obtain Z/γ as a result of invoking rule i in the GPS system. The links and filter functions in the SPM scheme are defined so it is possible to copy exactly $Z^g/\gamma c$ from dom(Z^g) to dom(X) using the above sequence of links. Finally by virtue of possessing Z^g/γ, X is allowed to obtain $Z/\gamma c$ from dom(Z^g).

Let RG be the set of rights in the GPS system, extended to occur with and without the SPM copy flag. Let k be a symbol, denoting cohort, which does not occur in RG. We define the rights in the SPM simulation to be RG $\cup$ {k : c}. Since GPS has no copy flag, we will make sure that all rights in RG which are in dom(X) have the SPM copy flag. We say that tickets of type $s/r:c$ for r$\in$RG are GPS tickets. All other types of tickets are said to be non-GPS tickets.

The SPM cohorts of X are connected to X by placing X/k in their domains. This sets up a cohort link from X to each of its cohorts. Note that the cohort link is authorized by a k ticket at the destination. We also define the inverse-cohort link by requiring the k ticket at the source. A cohort link is therefore always accompanied by an inverse-cohort link in the opposite direction. The formal definitions are respectively as follows.

$$\text{link}_k(U,V) \equiv U/k \in \text{dom}(V)$$
$$\text{link}_{\hat{k}}(U,V) \equiv V/k \in \text{dom}(U)$$

We define f_k from s to sl, sr and sm_i to be s/RG. This has the effect that every GPS ticket in dom(X) can be copied to X^l, X^r and X^{m_i}. Moreover this is the only way that these cohorts of X can acquire GPS tickets. A further connection between X and its X^l, X^r and X^{m_i} cohorts is that X possesses all tickets for these cohorts. These non-GPS tickets play a crucial role in the simulation as explained below.

To simulate the GPS rules we define σ and inverse-σ links for each set of rights σ which occurs as α, β or γ in a GPS rule, respectively as follows.

$$\text{link}_\sigma(U,V) \equiv U/\sigma \in \text{dom}(V) \equiv (\forall p \in \sigma)\ U/p \in \text{dom}(V)$$
$$\text{link}_{\hat{\sigma}}(U,V) \equiv V/\sigma \in \text{dom}(U) \equiv (\forall p \in \sigma)\ V/p \in \text{dom}(U)$$

Now consider a class I GPS rule. Let this rule be invoked in the GPS system with X, Y and Z respectively as the left, middle and right subjects as indicated in part I of table 6. The SPM counterpart of the given state is that $Z/\beta c \in$dom(Y) and $Y/\alpha c \in$dom(X). To simulate the rule we copy Z/β from Y to Y^{m_i} using $\text{link}_k(Y,Y^{m_i})$. The resulting $\text{link}_\beta(Z,Y^{m_i})$ is then used to copy Z^r/β from Z to Y^{m_i}. This sets up $\text{link}_\beta(Z^r,Y^{m_i})$. Similarly we copy Y/α from X to X^l using $\text{link}_k(X,X^l)$. The resulting $\text{link}_\alpha(Y,X^l)$ is used to copy Y^{m_i}/α from Y to X^l, setting up $\text{link}_\alpha(Y^{m_i},X^l)$. So at this point we have $\text{link}_\beta(Z^r,Y^{m_i})$ and $\text{link}_\alpha(Y^{m_i},X^l)$. For class II rules we have a similar

Class of rule i	GPS System X obtains Z/γ using rule i	SPM Simulation X obtains Z/γc as follows			
		Copy	From	To	Using
I	Given X $\overset{\alpha}{\Leftrightarrow}$ Y $\overset{\beta}{\Leftrightarrow}$ Z Obtain X $\overset{\gamma}{\Leftrightarrow}$ Z	Z/β	Y	Y^{m_i}	$link_k(Y,Y^{m_i})$
		Z^r/β	Z	Y^{m_i}	$link_\beta(Z,Y^{m_i})$
		Y/α	X	X^l	$link_k(X,X^l)$
		Y^{m_i}/α	Y	X^l	$link_\alpha(Y,X^l)$
II	Given X $\overset{\alpha}{\Leftrightarrow}$ Y $\overset{\beta}{\Leftarrow\!\Leftrightarrow}$ Z Obtain X $\overset{\gamma}{\Leftrightarrow}$ Z	Y/β	Z	Z^r	$link_k(Z,Z^r)$
		Y^{m_i}/β	Y	Z^r	$link_\beta(Y,Z^r)$
		Y/α	X	X^l	$link_k(X,X^l)$
		Y^{m_i}/α	Y	X^l	$link_\alpha(Y,X^l)$
III	Given X $\overset{\alpha}{\Leftarrow\!\Leftrightarrow}$ Y $\overset{\beta}{\Leftrightarrow}$ Z Obtain X $\overset{\gamma}{\Leftrightarrow}$ Z	Z/β	Y	Y^{m_i}	$link_k(Y,Y^{m_i})$
		Z^r/β	Z	Y^{m_i}	$link_\beta(Z,Y^{m_i})$
		X/α	Y	Y^{m_i}	$link_k(Y,Y^{m_i})$
		X^l/α	X	Y^{m_i}	$link_\alpha(X,Y^{m_i})$
IV	Given X $\overset{\alpha}{\Leftarrow\!\Leftrightarrow}$ Y $\overset{\beta}{\Leftarrow\!\Leftrightarrow}$ Z Obtain X $\overset{\gamma}{\Leftrightarrow}$ Z	Y/β	Z	Z^r	$link_k(Z,Z^r)$
		Y^{m_i}/β	Y	Z^r	$link_\beta(Y,Z^r)$
		X/α	Y	Y^{m_i}	$link_k(Y,Y^{m_i})$
		X^l/α	X	Y^{m_i}	$link_\alpha(X,Y^{m_i})$
All	Common suffix	Z^g/γc	Z^g	Z^r	$link_{\hat{k}}(Z^g,Z^r)$
		Z^g/γc	Z^r	Y^{m_i}	$link_\beta(Z^r,Y^{m_i})$ Class I, III $link_{\hat{\beta}}(Z^r,Y^{m_i})$ Class II, IV
		Z^g/γc	Y^{m_i}	X^l	$link_\alpha(Y^{m_i},X^l)$ Class I, II $link_{\hat{\alpha}}(Y^{m_i},X^l)$ Class III, IV
		Z^g/γc	X^l	X	$link_{\hat{k}}(X^l,X)$
		Z/γc	Z^g	X	$link_\gamma(Z^g,X)$

Table 6: Simulation of GPS Operations in SPM

simulation indicated in part II of table 6. We establish $\text{link}_\alpha(\text{Y}^{\text{m}_\text{i}},\text{X}^\text{l})$ as in the class I case. However the given state now requires that $\text{Y}/\beta\text{c}\in\text{dom}(\text{Z})$. So in the simulation we now copy Y/β from Z to Z^r using $\text{link}_\text{k}(\text{Z},\text{Z}^\text{r})$, using the resulting $\text{link}_\beta(\text{Y},\text{Z}^\text{r})$ to copy $\text{Y}^{\text{m}_\text{i}}/\beta$ from Y to Z^r. This sets up $\text{link}_{\hat{\beta}}(\text{Z}^\text{r},\text{Y}^{\text{m}_\text{i}})$. Class III and IV rules are similarly simulated as shown in table 6.

The net effect, with respect to figure 1, is that for rules with a left to right α edge we establish $\text{link}_\alpha(\text{Y}^{\text{m}_\text{i}},\text{X}^\text{l})$, while for rules with a right to left α edge we establish $\text{link}_{\hat{\alpha}}(\text{Y}^{\text{m}_\text{i}},\text{X}^\text{l})$. Similarly for rules with a left to right β edge we establish $\text{link}_\beta(\text{Z}^\text{r},\text{Y}^{\text{m}_\text{i}})$, while for rules with a right to left β edge we establish $\text{link}_{\hat{\beta}}(\text{Z}^\text{r},\text{Y}^{\text{m}_\text{i}})$. To ensure that these links can be established if and only if the corresponding GPS rule is authorized we define f_σ from a type s subject to be as follows.

$$\begin{array}{lcl} f_\sigma(s,\ sl) & = & \{sm_\text{i}/\sigma|\text{ for every class I or II rule i with } \alpha{=}\sigma\} \\ f_\sigma(s,\ sr) & = & \{sm_\text{i}/\sigma|\text{ for every class II or IV rule i with } \beta{=}\sigma\} \\ f_\sigma(s,\ sm_\text{i}) & = & \{sl/\sigma|\text{ if rule i is of class III or IV with } \alpha{=}\sigma\} \cup \\ & & \{sr/\sigma|\text{ if rule i is of class I or III with } \beta{=}\sigma\} \end{array}$$

To continue the simulation of the GPS rules we somehow have to get the $\text{Z}/\gamma\text{c}$ ticket in dom(X). This is achieved by the "common suffix" portion of table 6. Recall that Z^g possesses the tickets Z/R and Z^g/R. We connect Z^g to Z^r by an inverse-cohort link, by placing Z^r/k in $\text{dom}(\text{Z}^\text{g})$. By defining $f_{\hat{\text{k}}}(sg,sr)$ to be sg/R we allow Z^r to acquire Z^g/R. We allow $\text{Z}^\text{g}/\gamma\text{c}$ to be copied from Z^r to $\text{Y}^{\text{m}_\text{i}}$ to X^l provided rule i of the GPS system lets X obtain Z/γ. We achieve this by the following definitions.

$$\begin{array}{lcl} f_\sigma(sr,\ sm_\text{i}) & = & \textbf{if rule i is of class I or III with } \beta{=}\sigma \textbf{ then } sg/\gamma\text{c} \textbf{ else } \phi \\ f_{\hat{\sigma}}(sr,\ sm_\text{i}) & = & \textbf{if rule i is of class II or IV with } \beta{=}\sigma \textbf{ then } sg/\gamma\text{c} \textbf{ else } \phi \\ f_\sigma(sm_\text{i},\ sl) & = & \textbf{if rule i is of class I or II with } \alpha{=}\sigma \textbf{ then } sg/\gamma\text{c} \textbf{ else } \phi \\ f_{\hat{\sigma}}(sm_\text{i},\ sl) & = & \textbf{if rule i is of class III or IV with } \alpha{=}\sigma \textbf{ then } sg/\gamma\text{c} \textbf{ else } \phi \end{array}$$

The effect of all this so far is to enable X^l to acquire $\text{Z}^\text{g}/\gamma\text{c}$, if X can obtain Z/γ in the GPS system. Now there is an inverse-cohort link from X^l to X, so by defining $f_{\hat{\text{k}}}(sl,s)$ to be sg/R, X is able to acquire Z^g/γ in our simulation. The final step is to allow X to obtain $\text{Z}/\gamma\text{c}$ from $\text{dom}(\text{Z}^\text{g})$ over this $\text{link}_\gamma(\text{Z}^\text{r},\text{X})$. This is easily achieved by defining $f_\gamma(sg,s)$ to be $s/\gamma\text{c}$.

The construction is now almost complete. We could in fact stop at this point and simply construct the initial state of the SPM system from the initial state of the GPS system as follows. For each subject X in the GPS system introduce the SPM cohorts X, X^g, X^l, X^r, X^{m_1}, ..., $\text{X}^{\text{m}_\text{n}}$ with their initial domains as given below.

$$
\begin{array}{lcl}
\mathrm{dom(X)} & = & \{\mathrm{Y}/\sigma\mathrm{c}|\ \mathrm{X} \overset{\sigma}{\Leftrightarrow} \mathrm{Y} \text{ in the GPS system}\} \cup \\
 & & \{\mathrm{X^l},\ \mathrm{X^r},\ \mathrm{X^{m_1}},\ \ldots,\ \mathrm{X^{m_n}}\} \times \mathrm{R} \\
\mathrm{dom(X^l)} & = & \mathrm{X/k} \\
\mathrm{dom(X^r)} & = & \mathrm{X/k} \\
\mathrm{dom(X^{m_i})} & = & \mathrm{X/k,\ i{=}1 \ldots n} \\
\mathrm{dom(X^g)} & = & \mathrm{X/R} \cup \mathrm{X^g/R} \cup \mathrm{X^r/k}
\end{array}
$$

This would suffice since GPS systems have no create operation. On the other hand GPS can be easily extended to include create operations. In fact to simulate combined authority and information transfer and theft in the take-grant model, we need to combine the constructions of this Section with those of the previous one. In doing so we would need to allow creation of subjects and their cohorts. It is therefore important to show that the cohorts with appropriate domains can actually be realized by SPM create operations. For the most part this is straightforward and simply requires suitable definition of *cc* and the create-rules. It seems proper to authorize creation by letting the type *s* subjects create the other types of cohorts. That is,

$$cc(s) = \{sg,\ sl,\ sr,\ sm_1,\ sm_2,\ \ldots,\ sm_\mathrm{n}\}$$

All other values of *cc* are empty. The initial state can then be defined to simply consist of a type *s* subject X for each GPS subject X with

$$\mathrm{dom(X)} = \{\mathrm{Y}/\sigma\mathrm{c}|\ \mathrm{X} \overset{\sigma}{\Leftrightarrow} \mathrm{Y} \text{ in the GPS system}\}$$

Tickets relating X to its cohorts can then be introduced by the create-rules. A minor complication arises from the requirement that $\mathrm{X^r/k} \in \mathrm{dom(X^g)}$. We can achieve this by copying $\mathrm{X^r/k}$ from X to $\mathrm{X^g}$, for which purpose we define $f_\mathrm{k}(s,sg)$ to be sr/k. With this set up we can easily extend the construction to allow subject creation by placing *s* in *cc*.

The above discussion results in the following scheme.

Scheme 6 *Grammatical protection systems with n rules numbered 1 ... n.*

1. $\mathrm{TS} = \{s,\ sg,\ sl,\ sr,\ sm_1,\ sm_2,\ \ldots,\ sm_\mathrm{n}\}$, $\mathrm{TO} = \phi$

   The type *s* subjects simulate GPS subjects. The other subject types respectively have the following roles: generator cohort, left cohort, right cohort, and middle cohort for GPS rules 1 ...n.

2. $\mathrm{R} = \mathrm{RG} \cup \{\mathrm{k:c}\}$, where RG is the set of rights in the grammatical protection system extended to occur with and without the copy flag and $\mathrm{k} \notin \mathrm{RG}$

3. $cc(s) = \{sg,\ sl,\ sr,\ sm_1,\ sm_2,\ \ldots,\ sm_\mathrm{n}\}$

4. $cr_p(s,\ sl) = child/\mathrm{R}$, $cr_c(s,\ sl) = parent/\mathrm{k}$
   $cr_p(s,\ sm_\mathrm{i}) = child/\mathrm{R}$, $cr_c(s,\ sm_\mathrm{i}) = parent/\mathrm{k}$
   $cr_p(s,\ sr) = child/\mathrm{R}$, $cr_c(s,\ sr) = parent/\mathrm{k}$
   $cr_p(s,\ sg) = \phi$, $cr_c(s,\ sg) = child/\mathrm{R} \cup parent/\mathrm{R}$

5. $\mathrm{link_k(U,V)} \equiv \mathrm{U/k}{\in}\mathrm{dom(V)}$
   $\mathrm{link_{\hat{k}}(U,V)} \equiv \mathrm{V/k}{\in}\mathrm{dom(U)}$

   For every σ which occurs as α, β or γ in any GPS rule,

   $\mathrm{link}_\sigma\mathrm{(U,V)} \equiv \mathrm{U}/\sigma{\in}\mathrm{dom(V)} \equiv (\forall \mathrm{p} \in \sigma)\ \mathrm{U/p}{\in}\mathrm{dom(V)}$
   $\mathrm{link}_{\hat{\sigma}}\mathrm{(U,V)} \equiv \mathrm{V}/\sigma{\in}\mathrm{dom(U)} \equiv (\forall \mathrm{p} \in \sigma)\ \mathrm{V/p}{\in}\mathrm{dom(U)}$

   The subscripts on these links have the following mnemonic significance: k for cohort, $\hat{\mathrm{k}}$ for inverse cohort, σ for sigma, and $\hat{\sigma}$ for inverse sigma.

6. Undefined values of the filter functions are assumed by default to be ϕ.

   $f_\mathrm{k}(s,\ sl) = s/\mathrm{RG}$
   $f_\mathrm{k}(s,\ sm_\mathrm{i}) = s/\mathrm{RG}$
   $f_\mathrm{k}(s,\ sr) = s/\mathrm{RG}$
   $f_\mathrm{k}(s,\ sg) = sr/\mathrm{k}$

   $f_{\hat{\mathrm{k}}}(sg,\ sr) = sg/\mathrm{R}$
   $f_{\hat{\mathrm{k}}}(sl,\ s) = sg/\mathrm{R}$

   $f_\sigma(s,\ sl) = \{sm_\mathrm{i}/\sigma |$ for every class I or II rule i with $\sigma{=}\alpha\}$
   $f_\sigma(s,\ sr) = \{sm_\mathrm{i}/\sigma |$ for every class II or IV rule i with $\sigma{=}\beta\}$
   $f_\sigma(s,\ sm_\mathrm{i}) = \{sl/\sigma |$ if rule i is of class III or IV with $\sigma{=}\alpha\} \cup$
   $\{sr/\sigma |$ if rule i is of class I or III with $\sigma{=}\beta\}$

   $f_\sigma(sr,\ sm_\mathrm{i}) =$ **if** rule i is of class I or III with $\sigma{=}\beta$ **then** $sg/\gamma\mathrm{c}$ **else** ϕ
   $f_{\hat{\sigma}}(sr,\ sm_\mathrm{i}) =$ **if** rule i is of class II or IV with $\sigma{=}\beta$ **then** $sg/\gamma\mathrm{c}$ **else** ϕ
   $f_\sigma(sm_\mathrm{i},\ sl) =$ **if** rule i is of class I or II with $\sigma{=}\alpha$ **then** $sg/\gamma\mathrm{c}$ **else** ϕ
   $f_{\hat{\sigma}}(sm_\mathrm{i},\ sl) =$ **if** rule i is of class III or IV with $\sigma{=}\alpha$ **then** $sg/\gamma\mathrm{c}$ **else** ϕ

   $f_\sigma(sg,\ s) = s/\sigma\mathrm{c}$

In the rest of this Section we prove the correctness of our construction.

Theorem 2 *The GPS system of definition 7 and the SPM scheme 6 with its specified initial state are equivalent.*

Proof: We prove equivalence in two steps. Firstly we need to show that if there is a GPS state with $\mathrm{X} \overset{\gamma}{\Leftrightarrow\!\!\!\rightarrow} \mathrm{Z}$ then there is a sequence of operations in the SPM system by which $\mathrm{Z}/\gamma\mathrm{c}{\in}\mathrm{dom(X)}$. This is achieved by simulating each GPS operation as shown in table 6. It is apparent from our discussion leading up to scheme 6 that the SPM operations in this simulation are authorized. Secondly we need to show the converse property given below.

$$Z/r:c \in dom(X) \Rightarrow \text{there exists a GPS state in which } X \overset{\gamma}{\Leftrightarrow} Z \text{ with } r \in \gamma$$

By the results of [34] as discussed in Section 3, we can ignore create operations for safety analysis by assuming that each SPM subject of type s creates one instance of each of the remaining types. So from this augmented state we need to consider only copy operations.

We prove the above assertion by induction on the number of copy operations in the SPM system. For the basis case let this number be 0 and the assertion follows trivially from construction of the initial state. Assume the assertion is true for states derived by less than n copy operations. If the n-th operation is other than copying $Z/r:c$ to X, the assertion follows by induction hypothesis. Otherwise by inspection of the scheme it is evident that $Z/r:c$ can be copied to dom(X) only from $dom(Z^g)$ over some $link_\sigma$ where $r \in \sigma$. This requires $Z^g/\sigma \in dom(X)$. It is further evident that Z^g/σ can be copied to dom(X) only from $dom(X^l)$ over $link_\hat{k}$, which can be established only by the create-rules. Now subjects of type *sl* can obtain tickets of type $sg/\sigma c$ only from subjects of type sm_i. So there must exist some Y^{m_i} from which $Z^g/\sigma c$ was copied to X^l. There are four cases to consider. Let rule i be of class I. To copy $Z^g/\sigma c$ from Y^{m_i} to X^l we require $link_\alpha(Y^{m_i},X^l)$ and $sg/\sigma \in f_\alpha(sm_i,sl)$. That is $r \in \sigma \subseteq \gamma$. By definition $link_\alpha(Y^{m_i},X^l)$ implies $Y^{m_i}/\alpha \in dom(X^l)$. By inspection of the scheme, X^l can obtain Y^{m_i}/α only by copying it from Y over $link_\alpha(Y,X^l)$. This in turn requires $Y/\alpha \in dom(X^l)$. Subjects of type *sl* can obtain tickets of type s/α only from subjects of type s over a $link_k$. Such $link_k$'s can be established only by the create-rules, so X^l must have obtained Y/α from dom(X), which requires that $Y/\alpha c \in dom(X)$. Therefore by induction hypothesis there is a GPS state with $X \overset{\alpha}{\Leftrightarrow} Y$.

Next consider the requirement that $Z^g/\sigma c \in dom(Y^{m_i})$. By similar arguments we can conclude that there is a GPS state with $Y \overset{\beta}{\Leftrightarrow} Z$.

Since GPS systems are monotonic it follows that if there is a GPS state with $X \overset{\alpha}{\Leftrightarrow} Y$ and a GPS state with $Y \overset{\beta}{\Leftrightarrow} Z$ then there is a GPS state with $X \overset{\alpha}{\Leftrightarrow} Y \overset{\beta}{\Leftrightarrow} Z$. In this GPS state rule i is authorized so X can obtain γ rights for Z, where $r \in \gamma$. This completes the induction step when rule i is of class I. For the other cases where rule i is of class II, III or IV the induction step can be similarly proved. □

8 CONCLUSION

In this paper we have shown how versions of Bell-LaPadula multi-level security model [3], take-grant models [16, 21, 41] and grammatical protection systems [10, 22] can be specified as SPM schemes. This work complements our earlier efforts in demonstrating the modeling power of SPM by considering specific policies of practical interest [32, 33, 34, 36]. It is encouraging that SPM offers a unified framework in which these diverse models and policies can be expressed. It is moreover remarkable, that in

all these cases the SPM schemes satisfy the acyclic attenuating assumption required for safety analysis [34].

The results of this paper are in sharp contrast to results for the Harrison-Ruzzo-Ullman (HRU) access-matrix model [14]. HRU does subsume all the models discussed in this paper in terms of expressive power. However, all known constructions of these models within HRU require multi-conditional commands (i.e., commands whose conditions have two or more terms), whereas safety is undecidable in HRU even for bi-conditional commands (i.e., commands whose conditions have exactly two terms).

Our construction for multilevel models establishes that the traditional label-based mandatory controls of multilevel security have an alternate expression in terms of the type-based constraints on propagation of access rights imposed by SPM. The SPM viewpoint has the advantage of providing explicit machinery for formulating policies "in between" the two extremes of mandatory and discretionary policies in the Bell-LaPadula model.

The construction for theft in take-grant models emphasizes that a protection model with the generality of SPM is useful, even for a system with very specific control operations. This is because assumptions about behavior can be modeled in SPM. Most systems implement very specific control operations, and safety can be guaranteed only with such additional assumptions.

Our construction for grammatical protection systems demonstrates the ability of SPM to simulate models whose control operations appear to be contrary to SPM operations. GPS rules are particularly troublesome in this regard, since they actually allow new privileges to be created. This indicates that SPM has abstracted some essential properties of control operations in protection models. This abstraction is probably more fundamental than the viewpoint which lead us to develop the SPM rules in the first place. It is also significant, because in combination with the take-grant constructions it allows us to accommodate the take-grant notions of information and authority transfer and theft within SPM.

We conjecture that SPM is in some sense equivalent to the monotonic access matrix, in which delete and destroy operations are not allowed [15]. Some kind of equivalence is inevitable since both models have undecidable safety in general and both are monotonic. The interesting question is whether or not SPM has behavioral equivalence to monotonic HRU, in the sense discussed in Section 4. Resolution of this question will provide a significant advance in our understanding of protection models. It has recently been shown by Ammann and Sandhu [1, 2] that extending SPM to have a multi-parent joint create operation gives us equivalence to monotonic HRU. It has also been conjectured that SPM is actually less expressive than monotonic HRU (under the terms of behavioral equivalence). The precise relationship of the expressive power of SPM with respect to extended SPM or monotonic HRU remains an important open question.

Finally, we are well aware that SPM is a monotonic model and the question of

extending it to include some non-monotonic features such as transfer-only privileges and mutually exclusive privileges is an important research issue. It appears that some aspects of the integrity policies considered by Clark and Wilson [11] and others [29, 35, 42] will need such features. However as demonstrated by Budd [10] it does not take very much to get into intractable analysis problems with such non-monotonic privileges. Developing a suitable model which includes non-monotonic privileges and has tractable safety analysis is an important and difficult research problem. Our work on SPM provides a basis for this research.

ACKNOWLEDGMENT

The author acknowledges several insightful comments of the anonymous referees which have led to a much improved manuscript. The author also acknowledges the support and encouragement of Sylvan Pinsky and Howard Stainer in conducting this research. Finally, the author thanks Richard Lipton of Princeton University and Paul Ammann of George Mason University for discussions on the meaning of equivalence among models.

References

[1] Ammann, P.E. and Sandhu, R.S. "Extending the Creation Operation in the Schematic Protection Model." *Proc. Sixth Annual Computer Security Applications Conference*, Tucson, Arizona, December 1990, pages 340-348.

[2] Ammann, P.E. and Sandhu, R.S. "Safety Analysis for the Extended Schematic Protection Model." *Proc. IEEE Symposium on Research in Security and Privacy*, Oakland, California, May 1991, pages 87-97.

[3] Bell, D.E. and LaPadula, L.J. "Secure Computer Systems: Unified Exposition and Multics Interpretation." MTR-2997, MITRE Corporation, Bedford, Mass. (1975).

[4] Bell, D.E. "Secure Computer Systems: A Network Interpretation." *Third Aerospace Computer Security Applications Conference*, 32-39 (1987).

[5] Bell, D.E. "Concerning "Modeling" of Computer Security." *IEEE Symposium on Security and Privacy*, 8-13 (1988).

[6] Biba, K.J. "Integrity Considerations for Secure Computer Systems." MTR-3153, MITRE Corporation, Bedford, Mass. (1977).

[7] Bishop, M. and Snyder, L. "The Transfer of Information and Authority in a Protection System." *7th ACM Symposium on Operating Systems Principles*, 45-54 (1979).

[8] Bishop, M. "Theft of Information in the Take-Grant Protection Model." *Computer Security Foundations Workshop*, 194-218 (1988).

[9] Biskup, J. "Some Variants of the Take-Grant Protection Model." *Information Processing Letters* 19(3):151-156 (1984).

[10] Budd, T.A. "Safety in Grammatical Protection Systems." *International Journal of Computer and Information Sciences* 12(6):413-431 (1983).

[11] Clark, D.D. and Wilson, D.R. "A Comparison of Commercial and Military Computer Security Policies." *IEEE Symposium on Security and Privacy*, 184-194 (1987).

[12] Denning, D.E. "A Lattice Model of Secure Information Flow." *Communications of ACM* 19(5):236-243 (1976).

[13] Graham, G.S. and Denning, P.J. "Protection - Principles and Practice." *AFIPS Spring Joint Computer Conference* 40:417-429 (1972).

[14] Harrison, M.H., Ruzzo, W.L. and Ullman, J.D. "Protection in Operating Systems." *Communications of ACM* 19(8):461-471 (1976).

[15] Harrison, M.H. and Ruzzo, W.L. "Monotonic Protection Systems." In DeMillo, R.A., Dobkin, D.P., Jones, A.K. and Lipton, R.J. (Editors). *Foundations of Secure Computations.* Academic Press (1978).

[16] Jones, A.K., Lipton, R.J. and Snyder, L., "A Linear Time Algorithm for Deciding Security." *17th IEEE Symposium on the Foundations of Computer Science*, 337-366 (1976).

[17] Lampson, B.W. "Protection." *5th Princeton Symposium on Information Science and Systems*, 437-443 (1971). Reprinted in *ACM Operating Systems Review* 8(1):18-24 (1974).

[18] Landwehr, C.E. "Formal Models for Computer Security." *ACM Computing Surveys* 13(3):247-278 (1981).

[19] Landwehr, C.E. "The Best Available Technologies for Computer Security." *IEEE Computer* 16(7):86-100 (1983).

[20] Lee, T.M.P. "Using Mandatory Integrity to Enforce "Commercial" Security." *IEEE Symposium on Security and Privacy*, 140-146 (1988).

[21] Lipton, R.J. and Snyder, L. "A Linear Time Algorithm for Deciding Subject Security." *Journal of ACM* 24(3):455-464 (1977).

[22] Lipton, R.J. and Budd, T.A. "On Classes of Protection Systems." In DeMillo, R.A., Dobkin, D.P., Jones, A.K. and Lipton, R.J. (Editors). *Foundations of Secure Computations.* Academic Press (1978).

[23] Lockman, A. and Minsky, N. "Unidirectional Transport of Rights and Take-Grant Control." *IEEE Transactions on Software Engineering* SE-8(6):597-604 (1982).

[24] McLean, J. "A Comment on the 'Basic Security Theorem' of Bell and LaPadula." *Information Processing Letters* 20(2):67-70 (1985).

[25] McLean, J. "Reasoning About Security Models." *IEEE Symposium on Security and Privacy*, 123-131 (1987).

[26] McLean, J. "The Algebra of Security." *IEEE Symposium on Security and Privacy*, 2-7 (1988).

[27] McLean, J. "Specifying and Modeling Computer Security." *IEEE Computer* 23(1):9-16 (1990).

[28] Minsky, N. "Selective and Locally Controlled Transport of Privileges." *ACM Transactions on Programming Languages and Systems* 6(4):573-602 (1984).

[29] Moffett, J.D. and Sloman, M.S. "The Source of Authority for Commercial Access Control." *Computer* 21(2):59-69 (1988).

[30] Pittelli, P. "The Bell-LaPadula Computer Security Model Represented as a Special Case of the Harrison-Ruzzo-Ullman Model." *NBS-NCSC National Computer Security Conference*, 118-121 (1987).

[31] Saltzer, J.H. and Schroeder, M.D. "The Protection of Information in Computer Systems." *Proceedings of IEEE* 63(9):1278-1308 (1975).

[32] Sandhu, R.S., "The SSR Model for Specification of Authorization Policies: A Case Study in Project Control." *8th IEEE International Computer Software and Applications Conference*, 482-491 (1984).

[33] Sandhu, R.S. and Share, M.E. "Some Owner Based Schemes with Dynamic Groups in the Schematic Protection Model." *IEEE Symposium on Security and Privacy*, 61-70 (1986).

[34] Sandhu, R.S. "The Schematic Protection Model: Its Definition and Analysis for Acyclic Attenuating Schemes." *Journal of ACM* 35(2):404-432 (1988).

[35] Sandhu, R.S. "Transaction-Control Expressions for Separation of Duties." *Fourth Aerospace Computer Security Applications Conference*, 282-286 (1988).

[36] Sandhu, R.S. "Transformation of Access Rights." *IEEE Symposium on Security and Privacy*, 259-268 (1989).

[37] Sandhu, R.S. "The Demand Operation in the Schematic Protection Model." *Information Processing Letters* 32(4):213-219 (1989).

[38] Sandhu, R.S. "Undecidability of the Safety Problem for The Schematic Protection Model with Cyclic Creates." *Journal of Computer and System Sciences*, in press.

[39] Schockley, W.R. "Implementing the Clark/Wilson Integrity Policy Using Current Technology," *NIST-NCSC National Computer Security Conference*, 29-37 (1988).

[40] Snyder, L. "Formal Models of Capability-Based Protection Systems." *IEEE Transactions on Computers* C-30(3):172-181 (1981).

[41] Snyder, L. "Theft and Conspiracy in the Take-Grant Model." *Journal of Computer and Systems Sciences* 23(3):337-347 (1981).

[42] *Report of the Invitational Workshop on Integrity Policy in Computer Information Systems (WIPCIS)*, (Katzke, S.W. and Ruthberg, Z.G., editors), National Institute of Standards and Technology, Special Publication 500-160, January 1989.