Architectural Refinement and Notions of Intransitive Noninterference*

Ron van der Meyden
School of Computer Science and Engineering,
University of New South Wales
meyden@cse.unsw.edu.au

May 23, 2012

Abstract

This paper deals with architectural designs that specify components of a system and the permitted flows of information between them. In the process of systems development, one might refine such a design by viewing a component as being composed of subcomponents, and specifying permitted flows of information between these subcomponents and others in the design. The paper studies the soundness of such refinements with respect to a spectrum of different semantics for information flow policies, including Goguen and Meseguer's purge-based definition, Haigh and Young's intransitive purge-based definition, and some more recent notions TA-security, TO-security and ITO-security defined by van der Meyden. It is shown that all these definitions support the soundness of architectural refinement, for both a state- and an action-observed model of systems. A notion of systems refinement in which the information content of observations is reduced is also studied. It is also shown that refinement preserves weak access control structure, an implementation mechanism that ensures TA-security.

1 Introduction

Architectural design is a high level of systems specification, concerned with identifying the components of a system and the patterns of their interaction. In this paper, we consider the relationship between information flow security policies and the architecture development process. We use ideas from the literature

*Author version of a paper to appear in Formal Aspects of Computing. The original publication is available at springerlink.com. Version of May 23, 2012. Work supported by Australian Research Council Discovery grants DP0451529 and DP0987769. An extended abstract of this work appeared in International Symposium on Engineering Secure Software and Systems, February 04-06, 2009 Leuven, Belgium, Springer LNCS No 5429, pp. 60-74. The present version adds proofs and the content of Sections 6 and 7.

on information flow security to give semantics to architectures, and study how such semantics support a systems development process that refines high level architectural designs to more detailed architectures.

The type of information flow security policies that form the basis of this work place constraints on the permitted flows of information, or causal effects, between system components, and are referred to in the literature as *non-interference* policies. These policies can be represented as a binary relation on the set of components. For classical multi-level security policies, this relation is transitive. It has been proposed that extensions to multi-level security, such as downgraders, require that the policy be intransitive [HY87, Rus92]. An architectural interpretation of intransitive noninterference policies is gaining increased prominence through such efforts as the MILS (Multiple Independent Levels of Security and Safety) approach to high-assurance systems design [AFHOT06, VBC+05], which envisages the utilization of recent advances in the efficiency of separation kernels to increase the degree of componentization of systems, enabling secure systems to be built from a mix of small, trusted and more complex, untrusted components [RR83], with global security properties assured from the separation property and a verification effort focussed on the trusted components.

During the process of system design, one may *refine* an architectural diagram by specifying internal structure for some of the systems components, breaking them down into sub-components, and specifying the permitted interferences of these sub-components with each other and with other components in the design. This leads to the following question: if one now builds a system according to the refined architecture, is it guaranteed to be compliant to the original architectural diagram? This property needs to hold in order for the process of architectural refinement to be sound for designs of information flow secure systems.

Our contribution in this paper is to answer this question for a range of meanings of the notion of a system being compliant with an architectural design. For the meaning of architectures, we consider several different semantics for security with respect to intransitive noninterference policies that have previously been proposed in the literature. The first of these (which we call here P-security) is Goguen and Meseguer's original semantics [GM82] for transitive policies, which has generally been felt to be inappropriate for intransitive policies, although such application is not without its adherents [RG99]. An alternate semantics for intransitive noninterference policies (called here IP-security) was proposed by Haigh and Young [HY87], and propounded by Rushby [Rus92]. We have argued recently that this definition is flawed: it considers some systems to be secure that have flows of information that are contrary to the intuitions for intransitive policies [Mey08]. In response, we have defined [Mey08] several alternate semantics for noninterference policies — TO-security, ITO-security and TA-security — that are better behaved, and all equivalent to the classical definition in case the policy is transitive. Moreover, TA-security can be shown [Mey08] to correspond in a precise sense to Rushby's "unwinding" proof technique for intransitive noninterference. We remark that under certain circumstances, the definitions given by Roscoe and Goldsmith [RG99] correspond either to P-security or to ITO-

security— for details, we refer the reader to [Mey07], which is concerned with a detailed comparison of the above definitions of security in a number of different semantic frameworks.

The answer to our question turns out to be that a common notion of architectural refinement is sound with respect to *all* these semantics for intransitive noninterference policies. Indeed, we show this for two types of systems models, one in which observations are made at a state, and the other in which observations take the form of outputs returned on the invocation of an action.

In order to deal with the action-observed model, it proves to be convenient to first consider also a notion of refinement, on systems rather than on architectural designs, in which the amount of information in observations is reduced. This notion is of some independent interest since it is intuitive that security of a system should be preserved when we reduce the amount of information in the observations that agents make in the system. In fact, we show that while the intuition holds for P-security, IP-security and TA-security, not all of the semantics of intransitive policies are preserved by this notion of systems refinement. The ones that do not, TO-security and ITO-security, state both upper and lower bounds on permitted information. Our results in this regard help to clarify the content of these semantics. Nevertheless, we do identify a useful sufficient condition under which all semantics are preserved under reduction of observation content, and this condition aids in the derivation of the results on architectural refinement with respect to the action-observed model.

In the next step of systems development beyond architectural design, one might apply specific mechanisms to enforce the constraints imposed by the architecture. One implementation mechanism that is known to support the satisfaction of information flow policies is the use of a type of access control structure, in which domains are restricted with respect to the objects that they may read and write. A simple statically checkable condition on these restrictions entails IP-security [Rus92] as well as the stronger notion of TA-security [Mey08]. We show that a notion of refinement, closely related to architectural refinement, can also be defined for access control structure, and that refinements at the architectural level can be satisfied by refinements at the level of the access control implementation. Moreover, we define a more expressive notion of access control structure than has previously been considered in the context of intransitive policies, in which read and write constraints may apply differentially to individual actions performed within a domain, rather than uniformly across the domain. We show how such a policy at the action level can implement a policy at the domain level.

Taken together, these results provide the justification for a systems development process that proceeds from high level architectural design, via refinements, though to implementations using access control mechanisms. We give an example that illustrates the process.

The structure of the paper is as follows. In Section 2 we give a formal model for architectures and define architectural refinement. Section 3 defines our state-observed system model, noninterference policies, and the spectrum of different semantics for these policies. In Section 4 we show that architectural re-

finement preserves all the policy semantics for the state-observed model. Next, in Section 5, we consider access control as a mechanism for the implementation of architectures, and show that access control structure is also preserved under architectural refinement. The example of a refinement from architecture through to access control implementation is presented in this section. Section 6 defines a notion of systems refinement on state-observed systems, in which the amount of information in observations is reduced, and identifies conditions under which this preserves the architectural semantics we consider. Section 7 deals with architectural refinement in action-observed systems. Related literature is discussed in section 8, and we make some concluding remarks in Section 9.

2 Architectural Refinement

We begin by formalising the notion of architecture, and defining a relation of refinement between architectures. In this section, we leave the question of the formal semantics of architectures open. In the following sections, we will show that architectural refinement is sound with respect to several different semantic interpretations of architectures.

The notion of architecture that we consider expresses only the highest levels of a system design. Intuitively, an architecture specifies that the system is comprised of a set of components that generate and hold information, and constrains the permitted flows of information between these components. Using terminology from the security literature, we refer to these components as domains.

Define an *architecture* to be a pair $\mathcal{A} = (D, \rightarrowtail)$, where D is a set of domains and $\rightarrowtail$ is a reflexive binary relation over D. We call the relation $\rightarrowtail$ the *information flow policy* of the architecture. Several related intuitions may be associated with this relation. One is that information is permitted to flow from a domain u to a domain v only if $u \rightarrowtail v$. Another is that actions in domain u may have directly observable effects in domain v only if $u \rightarrowtail v$. The relation is assumed to be reflexive because information flow from a domain to itself can never be prevented. The security literature has frequently assumed information flow policies to be transitive. The following example illustrates a case where this assumption is not desirable.

Example 1: Consider the architecture $\mathcal{A} = (\{H, D, L\}, \rightarrowtail)$ where $\rightarrowtail$ is the downgrader policy depicted in the upper part of Figure 1 (we omit reflexive edges). Here H represents the high security domain, L the low security domain and D the downgrader. Information may flow from L to H, but any flow in the other direction needs to be mediated by the downgrader (so we would not want transtivity of $\rightarrowtail$). Information flow from L to D is permitted, to allow for requests by L to D for H information, e.g., freedom-of-information requests. □

The process of systems design may take a component in a high level design, and specify that it is to be implemented as the composition of a set of lower level

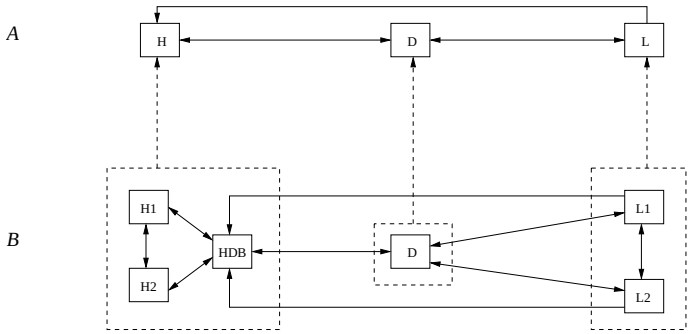

Figure 1: Refinement of a downgrader architecture

components. In this design step, the permitted flows of information between the lower level components, and others in the design, should also be specified. One way to understand this process is to view the design step as establishing a relationship of *refinement* between low level and high level architectures.

Example 2: The architecture $\mathcal{B} = (\{H_1, H_2, HDB, D, L_1, L_2\}, \rightarrowtail)$ (where the policy $\rightarrowtail$ is depicted in the lower part of Figure 1) represents a refinement of $\mathcal{A}$. We refine H into three components, two high level users H_1 and H_2 and a database HDB. Similarly, we refine L into two low level users L_1 and L_2. We assume that these may transmit information to each other, the high database, and D, but that all information flow from H_1 and H_2 to L_1 and L_2 is mediated by HDB and D. □

To formalise the notion of architectural refinement, let $\mathcal{A}_1 = (D_1, \rightarrowtail_1)$ and $\mathcal{A}_2 = (D_2, \rightarrowtail_2)$ be architectures. A *refinement mapping* from $\mathcal{A}_1$ to $\mathcal{A}_2$ is a function $r : D_1 \to D_2$ such that

1. r is *onto* D_2, and
2. for all $u, v \in D_1$, if $u \rightarrowtail_1 v$ then $r(u) \rightarrowtail_2 r(v)$.

In this case, we write $\mathcal{A}_1 \leq_r \mathcal{A}_2$, and say that $\mathcal{A}_1$ is a *refinement* of $\mathcal{A}_2$.

The requirement that r be surjective captures that intuition that if a design calls for the existence of a component, then a more detailed design should include an implementation of that component. Intuitively, each domain u in the high level design $\mathcal{A}_2$ is implemented by the set of domains in the preimage $r^{-1}(\{u\})$ in the detailed design. (For brevity, we henceforth write $r^{-1}(u)$ for $r^{-1}(\{u\})$.) The second condition expresses that the lower level policy should not permit information flow between two subdomains that was prohibited between their superdomains. That is, if in a high level design, information is not permitted to flow directly from component U to a component V, it should be incorrect to implement U and V using lower level components u and v, respectively, with information permitted to flow from u directly to v. (Note that the refinement mapping in Figure 1 satisfies this condition.)

We note that the definition of refinement is transitive in the following sense: if $\mathcal{A}_1 \leq_r \mathcal{A}_2$ and $\mathcal{A}_2 \leq_s \mathcal{A}_3$, then $\mathcal{A}_1 \leq_{s \circ r} \mathcal{A}_3$. This is a key property usually considered to be desirable in theories of refinement, since it permits the development of an architectural design to proceed in a number of stages, each of which refines the previous, by guaranteeing that the final design is a refinement of the original design.

Our main result in this paper is that the process of replacing a high level architecture by a lower level refinement is a sound design step, in the sense that any concrete system that implements the refined architecture also implements the higher level architecture. In order to make this claim formally precise, we first need to define what counts as a concrete implementation of an architecture. As a step towards this, we first consider a range of possible semantic interpretations of information flow policies. We use these in the semantics of architectures in Section 4.

3 Semantics of Information Flow Policies

To give semantics to architectures, we recall in this section several classical semantics for information flow policies [GM82, HY87, Rus92], and several new definitions proposed by van der Meyden [Mey08]. These definitions can be given for both state- and action-observed machines. We consider here the state-observed versions. The action-observed variants are discussed in a later section. The content of this section is largely definitional, and drawn from [Mey08].

The *state-observed* machine model [Rus92] for these definitions consists of deterministic machines of the form $\langle S, s_0, A, \mathtt{step}, \mathtt{obs}, \mathtt{dom}\rangle$, where S is a set of states, $s_0 \in S$ is the *initial state*, A is a set of actions, $\mathtt{dom} : A \to D$ associates each action to an element of the set D of security domains, $\mathtt{step} : S \times A \to S$ is a deterministic transition function, and $\mathtt{obs} : S \times D \to O$ maps states to an observation in some set O, for each security domain. We write $s \cdot \alpha$ for the state reached by performing the sequence of actions $\alpha \in A^*$ from state s, defined inductively by $s \cdot \epsilon = s$, and $s \cdot \alpha a = \mathtt{step}(s \cdot \alpha, a)$ for $\alpha \in A^*$ and $a \in A$. Here ϵ denotes the empty sequence.

Transitive information flow policies have been given a formal semantics using a definition based on a "purge" function [GM82]. Given a set $E \subseteq D$ of domains and a sequence $\alpha \in A^*$, we write $\alpha \upharpoonright E$ for the subsequence of all actions a in α with $\mathtt{dom}(a) \in E$. Given a policy $\rightarrowtail$, define the function $\mathtt{purge} : A^* \times D \to A^*$ by

$$\mathtt{purge}(\alpha, u) = \alpha \upharpoonright \{v \in D \mid v \rightarrowtail u\}.$$

For clarity, we may use subscripting of domain arguments of functions, writing e.g., $\mathtt{purge}(\alpha, u)$ as $\mathtt{purge}_u(\alpha)$.

Definition 1 *A system M is* P-secure *with respect to a policy $\rightarrowtail$ if for all sequences $\alpha, \alpha' \in A^*$ such that $\mathtt{purge}_u(\alpha) = \mathtt{purge}_u(\alpha')$, we have $\mathtt{obs}_u(s_0 \cdot \alpha) = \mathtt{obs}_u(s_0 \cdot \alpha')$.*

This can be understood as saying that domain u's observation depends only on the sequence of interfering actions that have been performed. By idempotence of $\mathtt{purge}_u$, this definition is equivalent to the classical formulation, according to which the system M is secure when for all $\alpha \in A^*$ and domains $u \in D$, we have $\mathtt{obs}_u(s_0 \cdot \alpha) = \mathtt{obs}_u(s_0 \cdot \mathtt{purge}_u(\alpha))$. This formulation can be understood as saying that each domain's observations are as if only interfering actions had been performed. We note that we will apply P-security to intransitive policies as well as the transitive policies for which it was originally intended.

While P-security is well accepted for transitive policies, it has been felt to be inappropriate for intransitive policies, since it does not permit L to *ever* learn about H actions if the policy is $H \rightarrowtail D \rightarrowtail L$, even if D is intended to be a trusted downgrader of H information. To address this deficiency, Haigh and Young [HY87] proposed to generalise the definition of the purge function to intransitive policies. (We follow the presentation of [Rus92].) Intuitively, the intransitive purge of a sequence of actions with respect to a domain u is the largest subsequence of actions that could form part of a causal chain of effects (permitted by the policy) ending with an effect on domain u. More formally, the definition makes use of a function $\mathtt{sources} : A^* \times D \to \mathcal{P}(D)$ defined inductively by $\mathtt{sources}(\epsilon, u) = \{u\}$ and

$$\mathtt{sources}(a\alpha, u) = \mathtt{sources}(\alpha, u) \cup \{\mathtt{dom}(a) \mid \exists v \in \mathtt{sources}(\alpha, u)(\mathtt{dom}(a) \rightarrowtail v)\}$$

for $a \in A$ and $\alpha \in A^*$. Intuitively, $\mathtt{sources}(\alpha, u)$ is the set of domains v such that there exists a sequence of permitted interferences from v to u within α. The *intransitive purge* function $\mathtt{ipurge} : A^* \times D \to A^*$ is then defined inductively by $\mathtt{ipurge}(\epsilon, u) = \epsilon$ and

$$\mathtt{ipurge}(a\alpha, u) = \begin{cases} a \cdot \mathtt{ipurge}(\alpha, u) & \text{if } \mathtt{dom}(a) \in \mathtt{sources}(a\alpha, u) \\ \mathtt{ipurge}(\alpha, u) & \text{otherwise} \end{cases}$$

for $a \in A$ and $\alpha \in A^*$. An alternative, equivalent formulation that we will find useful is the following: given a set $X \subseteq D$, define $\mathtt{ipurge}_X(\alpha)$ inductively by $\mathtt{ipurge}_X(\epsilon) = \epsilon$ and

$$\mathtt{ipurge}_X(\alpha a) = \begin{cases} \mathtt{ipurge}_{X \cup \{\mathtt{dom}(a)\}}(\alpha) \cdot a & \text{if } \exists u \in X(\mathtt{dom}(a) \rightarrowtail u) \\ \mathtt{ipurge}_X(\alpha) & \text{otherwise} \end{cases}$$

Then $\mathtt{ipurge}_u(\alpha)$ is identical to $\mathtt{ipurge}_{\{u\}}(\alpha)$. Haigh and Young's definition can then be formulated as the following variant of P-security in which we use the `ipurge` function in place of the `purge` function.

Definition 2 *M is IP-secure with respect to a policy $\rightarrowtail$ if for all $u \in D$ and all sequences $\alpha, \alpha' \in A^*$ with $\mathtt{ipurge}_u(\alpha) = \mathtt{ipurge}_u(\alpha')$, we have $\mathtt{obs}_u(s_0 \cdot \alpha) = \mathtt{obs}_u(s_0 \cdot \alpha')$.*

It can be seen that $\mathtt{ipurge}_u(\alpha) = \mathtt{purge}_u(\alpha)$ when $\rightarrowtail$ is transitive, so IP-security is in fact a generalisation of the definition of security for transitive policies.

These definitions are critiqued in [Mey08], where it is shown that IP-security sometimes allows quite unintuitive flows of information. In response, several alternative definitions are proposed. Each is based on a concrete model of the maximal amount of information that a domain may have after some sequence of actions has been performed, and states that a domain's observation may not give it more than this maximal amount of information. The definitions differ in the modelling of the maximal information, and take the view that a domain increases its information either by performing an action or by receiving information transmitted by another domain.

In the first model of the maximal information, what is transmitted when an domain performs an action is information about the actions performed by other domains. The following definition expresses this in a weaker way than the ipurge function.

Given sets X and A, let the set $\mathcal{T}(X, A)$ be the smallest set $\mathcal{T}$ containing X and such that if $x, y \in \mathcal{T}$ and $z \in A$ then $(x, y, z) \in \mathcal{T}$. Intuitively, the elements of $\mathcal{T}(X, A)$ are binary trees with leaves labelled from X and interior nodes labelled from A.

Given a policy $\rightarrowtail$, define, for each domain $u \in D$, the function $\mathtt{ta}_u : A^* \to \mathcal{T}(\{\epsilon\}, A)$ inductively by $\mathtt{ta}_u(\epsilon) = \epsilon$, and, for $\alpha \in A^*$ and $a \in A$,

1. if $\mathtt{dom}(a) \not\rightarrowtail u$, then $\mathtt{ta}_u(\alpha a) = \mathtt{ta}_u(\alpha)$,
2. if $\mathtt{dom}(a) \rightarrowtail u$, then $\mathtt{ta}_u(\alpha a) = (\mathtt{ta}_u(\alpha), \mathtt{ta}_{\mathtt{dom}(a)}(\alpha), a)$.

Intuitively, $\mathtt{ta}_u(\alpha)$ captures the maximal information that domain u may, consistently with the policy $\rightarrowtail$, have about the past actions of other domains. (The nomenclature is intended to be suggestive of *transmission* of information about *actions*.) Initially, a domain has no information about what actions have been performed. The recursive clause describes how the maximal information $\mathtt{ta}_u(\alpha)$ permitted to u after the performance of α changes when the next action a is performed. If a may not interfere with u, then there is no change, otherwise, u's maximal permitted information is increased by adding the maximal information permitted to $\mathtt{dom}(a)$ at the time a is performed (represented by $\mathtt{ta}_{\mathtt{dom}(a)}(\alpha)$), as well the fact that a has been performed. Thus, this definition captures the intuition that a domain may only transmit information that it is permitted to have, and then only to domains with which it is permitted to interfere.

Definition 3 *A system M is TA-secure with respect to a policy $\rightarrowtail$ if for all domains u and all $\alpha, \alpha' \in A^*$ such that $\mathtt{ta}_u(\alpha) = \mathtt{ta}_u(\alpha')$, we have $\mathtt{obs}_u(s_0 \cdot \alpha) = \mathtt{obs}_u(s_0 \cdot \alpha')$.*

Intuitively, this says that each domain's observations provide the domain with no more than the maximal amount of information that may have been transmitted to it, as expressed by the functions $\mathtt{ta}$.

The notion of TA-security can be shown to be a better fit to the intended applications and theory of IP-security. On the other hand, it may still be too weak for some applications. For example, it considers to be secure a system where a downgrader transmits to L an email attachment that it received from H,

without opening the attachment first (so that it does not know what information it is transmitting!) The second of van der Meyden's definitions is intended to address this potential deficiency.

The definition uses the following notion of *view*. The definition uses an absorptive concatenation function $\circ$, defined over a set X by, for $s \in X^*$ and $x \in X$, by $s \circ x = s$ if $s \neq \epsilon$ and x is equal to the final element of s, and $s \circ x = s\,x$ (ordinary concatenation) otherwise. Represent the view of domain u with respect to a sequence $\alpha \in A^*$ using the function $\mathtt{view}_u : A^* \to O(A \cup O)^*$ (where O is the set of observations in the system), defined inductively by

$$\begin{array}{rcl} \mathtt{view}_u(\epsilon) & = & \mathtt{obs}_u(s_0), \text{ and} \\ \mathtt{view}_u(\alpha a) & = & \begin{cases} \mathtt{view}_u(\alpha)\, a\, \mathtt{obs}_u(s_0 \cdot \alpha) & \text{if } \mathtt{dom}(a) = u \\ \mathtt{view}_u(\alpha) \circ \mathtt{obs}_u(s_0 \cdot \alpha) & \text{otherwise} \end{cases} \end{array}$$

That is, $\mathtt{view}_u(\alpha)$ is the sequence of all observations and actions of domain u in the run generated by α, compressed by the elimination of stuttering observations. Intuitively, $\mathtt{view}_u(\alpha)$ is the complete record of information available to domain u in the run generated by the sequence of actions α. The absorptive concatenation is intended to capture that the system is asynchronous, with domains not having access to a global clock. Thus, two periods of different length during which a particular observation obtains are not distinguishable to the domain.

Given a policy $\rightarrowtail$, for each domain $u \in D$, define the function $\mathtt{to}_u : A^* \to \mathcal{T}(O(A \cup O)^*, A)$ by $\mathtt{to}_u(\epsilon) = \mathtt{obs}_u(s_0)$ and

$$\mathtt{to}_u(\alpha a) = \begin{cases} \mathtt{to}_u(\alpha) & \text{when } \mathtt{dom}(a) \not\rightarrowtail u, \\ (\mathtt{to}_u(\alpha), \mathtt{view}_{\mathtt{dom}(a)}(\alpha), a) & \text{otherwise.} \end{cases}$$

Intuitively, this definition takes the model of the maximal information that an action a may transmit after the sequence α to be the fact that a has occurred, together with the information that $\mathtt{dom}(a)$ *actually* has, as represented by its view $\mathtt{view}_{\mathtt{dom}(a)}(\alpha)$. By contrast, TA-security uses in place of this the maximal information that $\mathtt{dom}(a)$ *may* have. (The nomenclature '$\mathtt{to}$' is intended to be suggestive of *transmission* of information about *observations*.)

Another variant of these definitions is mentioned in [Mey08] because of its relationship to a definition of Roscoe and Goldsmith [RG99]. Given a policy $\rightarrowtail$, for each domain $u \in D$, define the function $\mathtt{ito}_u : A^* \to \mathcal{T}(O(A \cup O)^*, A)$ by $\mathtt{ito}_u(\epsilon) = \mathtt{obs}_u(s_0)$ and

$$\mathtt{ito}_u(\alpha a) = \begin{cases} \mathtt{ito}_u(\alpha) & \text{when } \mathtt{dom}(a) \not\rightarrowtail u, \\ (\mathtt{ito}_u(\alpha), \mathtt{view}_{\mathtt{dom}(a)}(\alpha), a) & \text{if } \mathtt{dom}(a) = u. \\ (\mathtt{ito}_u(\alpha), \mathtt{view}_{\mathtt{dom}(a)}(\alpha a), a) & \text{otherwise.} \end{cases}$$

This definition is just like that of $\mathtt{to}$, with the difference that the information that may be transmitted by an action a to domains u such that $\mathtt{dom}(a) \rightarrowtail u$ but $\mathtt{dom}(a) \neq u$, includes the observation $\mathtt{obs}_{\mathtt{dom}(a)}(s_0 \cdot \alpha a)$ produced by the action a. Intuitively, the definition of security based on this notion will allow that the

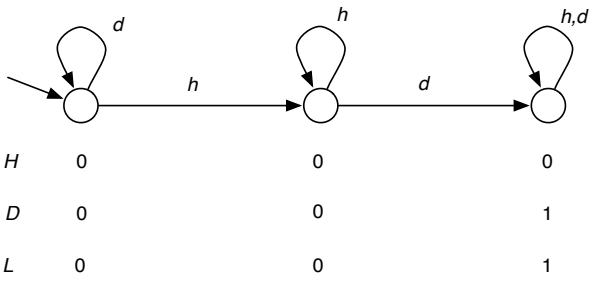

Figure 2: A system for the downgrader policy

action a transmits not just the information observable to $\mathtt{dom}(a)$ at the time that it is invoked, but also the new information that it computes and makes observable in $\mathtt{dom}(a)$. This information is not included in the value $\mathtt{ito}_{\mathtt{dom}(a)}(\alpha)$ itself, since the definition of security will state that the new observation may depend only on this value. The nomenclature in this case is intended to be suggestive of *immediate* transmission of information about observations.

We may now base the definition of security on either the function `to` or `ito` rather than `ta`.

Definition 4 *The system M is TO-secure with respect to $\rightarrowtail$ if for all domains $u \in D$ and all $\alpha, \alpha' \in A^*$ with $\mathtt{to}_u(\alpha) = \mathtt{to}_u(\alpha')$, we have $\mathtt{obs}_u(s_0 \cdot \alpha) = \mathtt{obs}_u(s_0 \cdot \alpha')$.*

The system M is ITO-secure with respect to $\rightarrowtail$ if for all domains $u \in D$ and all $\alpha, \alpha' \in A^*$ with $\mathtt{ito}_u(\alpha) = \mathtt{ito}_u(\alpha')$, we have $\mathtt{obs}_u(s_0 \cdot \alpha) = \mathtt{obs}_u(s_0 \cdot \alpha')$.

We remark that under certain circumstances, the definitions given by Roscoe and Goldsmith [RG99] correspond either to P-security or to ITO-security— see [Mey07] for details. The following result shows how these definitions are related:

Theorem 1 ([Mey08]) *With respect to a given policy $\rightarrowtail$, P-security implies TO-security implies ITO-security implies TA-security implies IP-security.*

Examples showing that all these notions are distinct are presented in [Mey08]. We give just one example here to illustrate how the definitions work.

Example 3: Figure 2 illustrates a system for the downgrader architecture $\mathcal{A}$ of Figure 1. There are two actions h, d, with $\mathtt{dom}(h) = H$ and $\mathtt{dom}(d) = D$. Transitions on taking an action are indicated by edge labels, and the observation made at a state is indicated below the state for each domain. For example, at the initial (leftmost) state s_0 we have $\mathtt{obs}_L(s_0) = 0$, and at the (rightmost) state t reached from s_0 by the sequence of actions hd we have $\mathtt{obs}_D(t) = 1$.

This system is not TO-secure (hence also not P-secure). To see this, consider

the sequences $\alpha = d$ and $\beta = hd$. We have

$$\begin{aligned} \mathtt{to}_L(d) &= (\mathtt{to}_L(\epsilon), \mathtt{view}_D(\epsilon), d) \\ &= (\epsilon,\ 0,\ d) \\ &= (\mathtt{to}_L(\epsilon), \mathtt{view}_D(h), d) \\ &= \mathtt{to}_L(hd). \end{aligned}$$

but $\mathtt{obs}_L(s_0 \cdot d) = 0$ and $\mathtt{obs}_L(s_0 \cdot hd) = 1$, so this is a violation of TO-security. On the other hand, these sequences do not yield a violation of ITO-security, because

$$\begin{aligned} \mathtt{ito}_L(d) &= (\mathtt{ito}_L(\epsilon), \mathtt{view}_D(d), d) \\ &= (\epsilon,\ 0d0,\ d)\ . \end{aligned}$$

and

$$\begin{aligned} \mathtt{ito}_L(hd) &= (\mathtt{ito}_L(h), \mathtt{view}_D(hd), d) \\ &= (\mathtt{ito}_L(\epsilon), \mathtt{view}_D(hd), d) \\ &= (\epsilon,\ 0d1,\ d)\ . \end{aligned}$$

so $\mathtt{ito}_L(d) \neq \mathtt{ito}_L(hd)$. It can be shown that the system is in fact ITO-secure (and therefore also TA-secure and IP-secure. Intuitively, L learns that the H action has been performed, but this (according to ITO-security) is considered secure because D also acquires this information (at the same time), and D is in fact responsible for having transmitted the information to L. TO-security involves the more stringent requirement that D *know* (by seeing in its view) that h has been performed *before* D performs the action d that transmits the information to L. On the other hand TA-security is more liberal: the system would remain TA-secure even if we were to replace the final D observation of 1 by the observation 0, so that D never learns whether H has performed h. Intuitively, TA-security requires just that D be causally involved in any transmission of information about H activity to L. □

Since definitions of security like those introduced above quantify over the infinite set of sequences of actions, it is desirable to have a proof technique for security that avoids having to examine an infinite number of possibilities. One approach that is prominent in the literature is the use of "unwinding" relations [GM84]. Rushby [Rus92] discusses the following unwinding relations for intransitive noninterference

Definition 5 *A* weak unwinding *for a system M with respect to a policy $\rightarrowtail$ is an indexed collection of equivalence relations $\{\sim_u\}_{u \in D}$ on the states of M satisfying the following conditions:*

OC: If $s \sim_u t$ then $\mathtt{obs}_u(s) = \mathtt{obs}_u(t)$. *(Output Consistency)*

WSC: If $s \sim_u t$ and $s \sim_{dom(a)} t$ then $s \cdot a \sim_u t \cdot a$.
(Weak Step Consistency)

LR: If $\mathtt{dom}(a) \not\rightarrowtail u$ then $s \sim_u s \cdot a$. *(Left Respect)*

Rushby shows that this notion provides a sound proof technique for IP-security. The following result of van der Meyden [Mey08] shows that it is in fact also a sound proof technique for the stronger notion TA-security.

Theorem 2 *Suppose that there exists a weak unwinding for M with respect to $\rightarrowtail$. Then M is TA-secure with respect to $\rightarrowtail$.*

The converse implication does not quite hold, but it turns out that unwinding is in fact a complete proof technique for TA-security if we allow consideration of a bisimilar variant of M. Given a system $M = \langle S, s_0, \mathtt{step}, \mathtt{obs}, \mathtt{dom}\rangle$ with actions A, define the "unfolded" system $\mathtt{uf}(M) = \langle S', s'_0, \mathtt{step}', \mathtt{obs}', \mathtt{dom}\rangle$ with actions A having the same domains as in M, by $S' = A^*$, $s'_0 = \epsilon$, $\mathtt{step}'(\alpha, a) = \alpha a$, and $\mathtt{obs}'_u(\alpha) = \mathtt{obs}_u(s_0 \cdot \alpha)$, where $s_0 \cdot \alpha$ is computed in M. Intuitively, this construction unfolds the graph of M into an infinite tree. It is easy to see that $\mathtt{uf}(M)$ is bisimilar to M in an obvious sense. The following result of van der Meyden [Mey08] provides a sense in which unwindings are a complete proof technique for TA-security.

Theorem 3 *The system M is TA-secure with respect to $\rightarrowtail$ iff there exists a weak unwinding on $\mathtt{uf}(M)$ with respect to $\rightarrowtail$.*

We use this characterization in the results that follow.

4 Soundness of Architectural Refinement

We are now in a position to assign a formal meaning to architectures, and to state precisely the main result of the paper.

Let X-security be a notion of security for information flow policies, like those discussed in the previous section. We say that a system M is *X-compliant* with an architecture $\mathcal{A} = (D, \rightarrowtail)$, where X is a definition of security, if

1. D is the set of domains of M, and
2. M is X-secure with respect to $\rightarrowtail$.

Intuitively, M is X-compliant with an architecture if it has an implementation for each of the components required by the architecture, and the information flows between these implementation components is consistent with the architecture's policy (with consistency defined by X-security.)

Consider now the definition of architectural refinement introduced in Section 2. In a system design process we may have started with a high level architecture $\mathcal{A}$, refined this to a lower level architecture $\mathcal{B}$, and then constructed a system that implements $\mathcal{B}$. In what sense have we then implemented the architecture $\mathcal{A}$ that formed the highest level specification of the system? For this, we need to be able to view the system from the perspective of the set of domains of the higher level architecture $\mathcal{A}$ rather than those of $\mathcal{B}$. This is the intent of the following definition.

Let $M = \langle S, s_0, A, \mathtt{step}, \mathtt{obs}^1, \mathtt{dom}^1\rangle$ be a system with set of domains D_1, and suppose $r : D_1 \rightarrow D_2$ is surjective. Then we may construct a system $r(M) = \langle S, s_0, A, \mathtt{step}, \mathtt{obs}^2, \mathtt{dom}^2\rangle$ as follows:

1. the actions A, set of states S, initial state s_0, and transition function $\texttt{step}$ of $r(M)$ are exactly as in M,

2. the set of domains of $r(M)$ is D_2

3. $\texttt{dom}^2(a) = r(\texttt{dom}^1(a))$ for each $a \in A$,

4. for $u \in D_2$ and $s \in S$, the observation $\texttt{obs}^2_u(s)$ is the function $f : r^{-1}(u) \to O$, given by $f(v) = \texttt{obs}^1_v(s)$ for $v \in r^{-1}(u)$.

Intuitively, each domain $u \in D_2$ is viewed by the refinement mapping as being broken down into the collection of subdomains $r^{-1}(u)$. An action of a subdomain in M is interpreted in $r(M)$ as belonging to its superdomain. The observation of a superdomain in $r(M)$ is taken to be the collection of all observations of its subdomains.

We can now give a formal meaning to soundness of architectural refinement. Say that a notion of compliance X is *preserved under architectural refinement* if whenever $r : \mathcal{A}_1 \to \mathcal{A}_2$ is a refinement mapping, if M is a system that is X-compliant with $\mathcal{A}_1$, then $r(M)$ is X-compliant with $\mathcal{A}_2$. That, is, if the goal of design was to construct a system that complies with architecture $\mathcal{A}_2$, we may do so by building a system M that complies with architecture $\mathcal{A}_1$ and viewing this system from the perspective of $\mathcal{A}_2$ through the mapping r.

Our main result is the following. (We give the proof later.)

Theorem 4 *The following notions are all preserved under architectural refinement: P-compliance, TO-compliance, ITO-compliance, TA-compliance and IP-compliance.*

One might also view a system to be secure if it has a weak unwinding, although there are some difficulties with this in view of the fact that weak unwinding is not preserved under bisimulation [Mey08]. Nevertheless, the following result is useful for the proof of Theorem 4, and is of independent interest in that it shows that a sound proof technique for security is preserved by architectural refinement.

Theorem 5 *Let $(D_1, \rightarrowtail_1) \leq_r (D_2, \rightarrowtail_2)$ and suppose that M is a system with set of domains D_1. If there exists a weak unwinding on M then there exists a weak unwinding on $r(M)$.*

Proof: Let $\{\sim_u\}_{u \in D}$ be a weak unwinding on M. Define the family of binary relations $\{\approx_u\}_{u \in D}$ on the states of $r(M)$ by $s \approx_u t$ if $s \sim_v t$ for all $v \in r^{-1}(u)$. We show that this is a weak unwinding on $r(M)$, i.e., that it satisfies OC, WSC and LR.

OC: If $s \approx_u t$ then $s \sim_v t$ for all $v \in r^{-1}(u)$. Since $\sim$ satisfies OC, this means that $\texttt{obs}^M_v(s) = \texttt{obs}^M_v(t)$ for all $v \in r^{-1}(u)$. This in turn implies that $\texttt{obs}^{r(M)}_u(s) = \texttt{obs}^{r(M)}_u(t)$, as required for OC.

WSC: Suppose $s \approx_u t$ and $s \approx_{\texttt{dom}^{r(M)}(a)} t$. Note that $\texttt{dom}^M(a) \in r^{-1}(\texttt{dom}^{r(M)}(a))$, so the latter implies $s \sim_{\texttt{dom}^M(a)} t$. The former gives that for all $v \in r^{-1}(u)$, we

have $s \sim_v t$. Since $\sim$ satisfies WSC, we conclude that $s \cdot a \sim_v t \cdot a$ for all $v \in r^{-1}(u)$. This yields that $s \cdot a \approx_u t \cdot a$.

LR: Suppose $\mathtt{dom}^{r(M)}(a) \not\rightarrow_2 u$. Then also $\mathtt{dom}^{M}(a) \not\rightarrow_1 v$, for all $v \in r^{-1}(u)$. By LR for $\sim$, we obtain that $s \cdot a \sim_v s$ for all $v \in r^{-1}(u)$. Thus, $s \cdot a \approx_u s$. □

To prove Theorem 4, we first prove a number of technical lemmas. The following lemma captures a uniform pattern of explanation for the proof of Theorem 4. Note that the definitions of security are expressed in terms of mappings X taking as arguments a system M, a domain u of this system, a sequence of actions α in this system, and an information flow policy $\rightarrowtail$ on the set of domains of the system. (In our notation above, the parameters M and $\rightarrowtail$ have generally been left implicit). The system is classified as secure according to the notion X with respect to policy $\rightarrowtail$ if for all systems M, domains u and sequences of actions α, α', if $X(M, u, \alpha, \rightarrowtail) = X(M, u, \alpha', \rightarrowtail)$ then $\mathtt{obs}_u(s_0 \cdot \alpha) = \mathtt{obs}_u(s_0 \cdot \alpha')$.

Lemma 1 *Let $(D_1, \rightarrowtail_1) \leq_r (D_2, \rightarrowtail_2)$ and suppose that M is a system with set of domains D_1. Suppose M is X-compliant with $(D_1, \rightarrowtail_1)$ and that for all domains $v \in D_1$ and all sequences of actions α, α', if*

$$X(r(M), r(v), \alpha, \rightarrowtail_2) = X(r(M), r(v), \alpha', \rightarrowtail_2)$$

then

$$X(M, v, \alpha, \rightarrowtail_1) = X(M, v, \alpha', \rightarrowtail_1).$$

Then $r(M)$ is X-compliant with $(D_2, \rightarrowtail_2)$.

Proof: Suppose M is X-compliant with $(D_1, \rightarrowtail_1)$. To show that $r(M)$ is X-compliant with $(D_2, \rightarrowtail_2)$, suppose that α, α' are sequences of actions of $r(M)$ (hence also of M), and u a domain of $r(M)$, such that

$$X(r(M), u, \alpha, \rightarrowtail_2) = X(r(M), u, \alpha', \rightarrowtail_2).$$

We need to show that $\mathtt{obs}_u^{r(M)}(s_0 \cdot \alpha) = \mathtt{obs}_u^{r(M)}(s_0 \cdot \alpha')$. By definition of $r(M)$, this amounts to showing that for all $v \in r^{-1}(u)$, we have $\mathtt{obs}_v^{M}(s_0 \cdot \alpha) = \mathtt{obs}_v^{M}(s_0 \cdot \alpha')$. For this, note that we have, by the assumed property, that

$$X(M, v, \alpha, \rightarrowtail_1) = X(M, v, \alpha', \rightarrowtail_1)$$

for all $v \in r^{-1}(u)$, Since M is X-compliant with $(D_1, \rightarrowtail_1)$, this implies that $\mathtt{obs}_v^{M}(s_0 \cdot \alpha) = \mathtt{obs}_v^{M}(s_0 \cdot \alpha')$, as required. □

In practice, we will often establish the condition of the lemma by constructing for all domains $v \in D_1$ a mapping F_v such that

$$X(M, v, \alpha, \rightarrowtail_1) = F_v(X(r(M), r(v), \alpha, \rightarrowtail_2))$$

for all sequences of actions α of M_1.

The following lemma is useful when showing preservation of TO-security and ITO-security.

Lemma 2 *Let $r : D_1 \to D_2$ and suppose that M is a system with set of domains D_1. Then for all sequences of actions α, α' and domains v of M, if $\mathtt{view}^{r(M)}_{r(v)}(\alpha) = \mathtt{view}^{r(M)}_{r(v)}(\alpha')$ then $\mathtt{view}^M_v(\alpha) = \mathtt{view}^M_v(\alpha')$.*

Proof: Note that views with respect to $r(v)$ in $r(M)$ are sequences consisting of actions a such that $\mathtt{dom}^{r(M)}(a) = r(v)$, and observations which are mappings f from $r^{-1}(r(v))$ to observations in M. Define the function F_v mapping such sequences with respect to $r(v)$ in $r(M)$ to views with respect to v in M by the following induction: $F_v(\epsilon) = \epsilon$,

$$F_v(\sigma a) = \begin{cases} F_v(\sigma)a & \text{if } \mathtt{dom}^M(a) = v \\ F_v(\sigma) & \text{otherwise} \end{cases}$$

when a is an action, and $F_v(\sigma f) = F_v(\sigma) \circ f(v)$ when f is an observation. We claim that for all sequences of actions α, we have $F_v(\mathtt{view}^{r(M)}_{r(v)}(\alpha)) = \mathtt{view}^M_v(\alpha)$. The result then follows.

The proof is by induction on the length of α. In case $\alpha = \epsilon$, we have $F_v(\mathtt{view}^{r(M)}_{r(v)}(\alpha)) = F_v(\mathtt{obs}^{r(M)}_{r(v)}(s_0)) = \mathtt{obs}^M_v(s_0) = \mathtt{view}^M_v(\alpha)$. For the inductive case of a sequence αa where a is an action and the result holds for α, we consider two cases, depending on whether $\mathtt{dom}^M(a) = v$.

Suppose first that $\mathtt{dom}^M(a) = v$. Then $\mathtt{dom}^{r(M)}(a) = r(v)$. Hence

$$\begin{aligned} F_v(\mathtt{view}^{r(M)}_{r(v)}(\alpha a)) &= F_v(\mathtt{view}^{r(M)}_{r(v)}(\alpha)\, a\, \mathtt{obs}^{r(M)}_{r(v)}(s_0 \cdot \alpha a)) \\ &= F_v(\mathtt{view}^{r(M)}_{r(v)}(\alpha))\, a\, \mathtt{obs}^M_v(s_0 \cdot \alpha a) \\ &= \mathtt{view}^M_v(\alpha)\, a\, \mathtt{obs}^M_v(s_0 \cdot \alpha a) \\ &= \mathtt{view}^M_v(\alpha a) \end{aligned}$$

Next, suppose that $\mathtt{dom}^M(a) \neq v$. Here we consider two subcases, depending on whether $\mathtt{dom}^{r(M)}(a) = r(v)$. If $\mathtt{dom}^{r(M)}(a) = r(v)$ then we have

$$\begin{aligned} F_v(\mathtt{view}^{r(M)}_{r(v)}(\alpha a)) &= F_v(\mathtt{view}^{r(M)}_{r(v)}(\alpha)\, a\, \mathtt{obs}^{r(M)}_{r(v)}(s_0 \cdot \alpha a)) \\ &= F_v(\mathtt{view}^{r(M)}_{r(v)}(\alpha))\, a\, \mathtt{obs}^M_v(s_0 \cdot \alpha a) \\ &= \mathtt{view}^M_v(\alpha) \circ \mathtt{obs}^M_v(s_0 \cdot \alpha a) \\ &= \mathtt{view}^M_v(\alpha a) \end{aligned}$$

If $\mathtt{dom}^M(a) \neq v$ and $\mathtt{dom}^{r(M)}(a) \neq r(v)$ then we consider two further subcases, depending on whether $\mathtt{obs}^{r(M)}_{r(v)}(s_0 \cdot \alpha a) = \mathtt{obs}^{r(M)}_{r(v)}(s_0 \cdot \alpha)$. If this condition holds, then it follows that $\mathtt{obs}^M_v(s_0 \cdot \alpha a) = \mathtt{obs}^M_v(s_0 \cdot \alpha)$, hence

$$\begin{aligned} F_v(\mathtt{view}^{r(M)}_{r(v)}(\alpha a)) &= F_v(\mathtt{view}^{r(M)}_{r(v)}(\alpha)) \\ &= \mathtt{view}^M_v(\alpha) \\ &= \mathtt{view}^M_v(\alpha a). \end{aligned}$$

On the other hand, if $\mathtt{obs}_{r(v)}^{r(M)}(s_0 \cdot \alpha a) \neq \mathtt{obs}_{r(v)}^{r(M)}(s_0 \cdot \alpha)$, then we have

$$\begin{aligned} F_v(\mathtt{view}_{r(v)}^{r(M)}(\alpha a)) &= F_v(\mathtt{view}_{r(v)}^{r(M)}(\alpha)\,\mathtt{obs}_{r(v)}^{r(M)}(s_0 \cdot \alpha a)) \\ &= F_v(\mathtt{view}_{r(v)}^{r(M)}(\alpha)) \circ \mathtt{obs}_v^M(s_0 \cdot \alpha a) \\ &= \mathtt{view}_v^M(\alpha) \circ \mathtt{obs}_v^M(s_0 \cdot \alpha a) \\ &= \mathtt{view}_v^M(\alpha a). \end{aligned}$$

Hence, in all cases we have the desired equation. □

We can now give the proof of Theorem 4.

Proof: Let $(D_1, \rightarrowtail_1) \leq_r (D_2, \rightarrowtail_2)$ and suppose that M is a system with set of domains D_1. We show in each case that the conditions of Lemma 1 hold. If σ is a sequence and S is a set, write $\sigma \restriction S$ for the subsequence of σ consisting of all elements in S.

For v a domain of a system M, and $\rightarrowtail$ a policy, let $I(M, \rightarrowtail, v)$ be the set of actions a of M with $\mathtt{dom}(a) \rightarrowtail v$. Note that $I(M, \rightarrowtail_1, v) \subseteq I(r(M) \rightarrowtail_2, r(v))$. For, if $a \in I(M, \rightarrowtail_1, v)$ then $\mathtt{dom}_1(a) \rightarrowtail_1 v$, so by the definition of refinement, $\mathtt{dom}_2(a) = r(\mathtt{dom}_1(a)) \rightarrowtail_2 r(v)$, hence $a \in I(r(M), \rightarrowtail_2, r(v))$.

P-security: For $v \in D_1$, define $F_v(\alpha) = \alpha \restriction I(M, \rightarrowtail_1, v)$. To see that this function has the required property, note that

$$\begin{aligned} F_v(\mathtt{purge}_{r(v)}^{r(M)}(\alpha)) &= \mathtt{purge}_{r(v)}^{r(M)}(\alpha)) \restriction I(M, \rightarrowtail_1, v) \\ &= (\alpha \restriction I(r(M), \rightarrowtail_2, r(v))) \restriction I(M, \rightarrowtail_1, v) \\ &= \alpha \restriction I(M, \rightarrowtail_1, v) \\ &= \mathtt{purge}_v^M(\alpha)) \end{aligned}$$

where we use the fact that $I(M, \rightarrowtail_1, v) \subseteq I(r(M) \rightarrowtail_2, r(v))$ in the third step.

TO-security: For TO-security we show that for all sequences of actions α, α', and $v \in D_1$, if $\mathtt{to}_{r(v)}^{r(M)}(\alpha) = \mathtt{to}_{r(v)}^{r(M)}(\alpha')$ then $\mathtt{to}_v^M(\alpha) = \mathtt{to}_v^M(\alpha')$. The proof is by induction on the combined length of α and α'. The base case of $\alpha = \alpha' = \epsilon$ is trivial.

Consider sequences αa and α', where a is an action, a domain $v \in D_2$, and suppose that $\mathtt{to}_{r(v)}^{r(M)}(\alpha a) = \mathtt{to}_{r(v)}^{r(M)}(\alpha')$. We consider two cases, depending on whether $\mathtt{dom}^{r(M)}(a) \rightarrowtail_2 r(v)$.

If $\mathtt{dom}^{r(M)}(a) \not\rightarrowtail_2 r(v)$, then $\mathtt{to}_{r(v)}^{r(M)}(\alpha) = \mathtt{to}_{r(v)}^{r(M)}(\alpha a) = \mathtt{to}_{r(v)}^{r(M)}(\alpha')$. Hence, by induction, we have $\mathtt{to}_v^M(\alpha) = \mathtt{to}_v^M(\alpha')$. Since $\mathtt{dom}^{r(M)}(a) \not\rightarrowtail_2 r(v)$, also $\mathtt{dom}^M(a) \not\rightarrowtail_1 v$. Hence $\mathtt{to}_v^M(\alpha a) = \mathtt{to}_v^M(\alpha)$. It follows that $\mathtt{to}_v^M(\alpha a) = \mathtt{to}_v^M(\alpha')$.

Alternately, if $\mathtt{dom}^{r(M)}(a) \rightarrowtail_2 r(v)$, then it follows from $\mathtt{to}_{r(v)}^{r(M)}(\alpha a) = \mathtt{to}_{r(v)}^{r(M)}(\alpha')$ that α' is not ϵ. Let $\alpha' = \beta b$, where b is an action. If $\mathtt{dom}^{r(M)}(b) \not\rightarrowtail_2 r(v)$, then we may apply the previous case with the roles of αa and α' swapped. We may assume, therefore, that $\mathtt{dom}^{r(M)}(b) \rightarrowtail_2 r(v)$. It then follows that $a = b$ and $\mathtt{to}_{r(v)}^{r(M)}(\alpha) = \mathtt{to}_{r(v)}^{r(M)}(\beta)$ and $\mathtt{view}_{\mathtt{dom}^{r(M)}(a)}^{r(M)}(\alpha) = \mathtt{view}_{\mathtt{dom}^{r(M)}(a)}^{r(M)}(\beta)$. From the former, we have by induction that $\mathtt{to}_v^M(\alpha) = \mathtt{to}_v^M(\beta)$. From the latter, we

have by Lemma 2 that $\mathtt{view}^{r(M)}_{\mathtt{dom}^M(a)}(\alpha) = \mathtt{view}^{r(M)}_{\mathtt{dom}^M(a)}(\beta)$. In both the case that $\mathtt{dom}^M(a) \rightarrowtail_1 v$ and the case that $\mathtt{dom}^M(a) \not\rightarrowtail_1 v$, it follows from these that $\mathtt{to}^M_v(\alpha a) = \mathtt{to}^M_v(\beta a)$.

ITO-security: In the case of ITO-security, we show that if M is ITO-secure, then $\mathtt{ito}^{r(M)}_{r(v)}(\alpha) = \mathtt{ito}^{r(M)}_{r(v)}(\alpha')$ implies $\mathtt{ito}^M_v(\alpha) = \mathtt{ito}^M_v(\alpha')$. The proof is by induction on the combined length of α and α'. The base case of $\alpha = \alpha' = \epsilon$ is trivial. Consider sequences αa and α', where a is an action, a domain $v \in D_2$, and suppose that $\mathtt{ito}^{r(M)}_{r(v)}(\alpha a) = \mathtt{ito}^{r(M)}_{r(v)}(\alpha')$. The case of $\mathtt{dom}^{r(M)}(a) \rightarrowtail_2 r(v)$ proceeds exactly along the lines of the argument for TO-security, and is left to the reader.

In case $\mathtt{dom}^{r(M)}(a) \rightarrowtail_2 r(v)$, it follows from $\mathtt{ito}^{r(M)}_{r(v)}(\alpha a) = \mathtt{ito}^{r(M)}_{r(v)}(\alpha')$ that α' is not ϵ. Let $\alpha' = \beta b$, where b is an action. If $\mathtt{dom}^{r(M)}(b) \not\rightarrowtail_2 r(v)$, then we may apply the previous case with the roles of αa and α' swapped. We may assume, therefore, that $\mathtt{dom}^{r(M)}(b) \rightarrowtail_2 r(v)$. It then follows that $a = b$ and $\mathtt{ito}^{r(M)}_{r(v)}(\alpha) = \mathtt{ito}^{r(M)}_{r(v)}(\beta)$ and either

1. $\mathtt{view}^{r(M)}_{\mathtt{dom}^{r(M)}(a)}(\alpha a) = \mathtt{view}^{r(M)}_{\mathtt{dom}^{r(M)}(a)}(\beta a)$ (in case $\mathtt{dom}^{r(M)}(a) \neq r(v)$) or
2. $\mathtt{view}^{r(M)}_{\mathtt{dom}^{r(M)}(a)}(\alpha) = \mathtt{view}^{r(M)}_{\mathtt{dom}^{r(M)}(a)}(\beta)$. (in case $\mathtt{dom}^{r(M)}(a) = r(v)$).

By induction, we have that $\mathtt{ito}^M_v(\alpha) = \mathtt{ito}^M_v(\beta)$. Note that in the case that $\mathtt{dom}^{r(M)}(a) = r(v)$, we also have by the induction hypothesis that $\mathtt{ito}^M_{\mathtt{dom}^M(a)}(\alpha) = \mathtt{ito}^M_{\mathtt{dom}^M(a)}(\beta)$. Hence

$$\begin{aligned}
\mathtt{ito}^M_{\mathtt{dom}^M(a)}(\alpha a) &= (\mathtt{ito}^M_{\mathtt{dom}^M(a)}(\alpha), , \mathtt{view}^M_{\mathtt{dom}^M(a)}(\alpha), a) \\
&= (\mathtt{ito}^M_{\mathtt{dom}^M(a)}(\beta), , \mathtt{view}^M_{\mathtt{dom}^M(a)}(\beta), a) \\
&= \mathtt{ito}^M_{\mathtt{dom}^M(a)}(\beta a).
\end{aligned}$$

By ITO-security of M, it then follows that $\mathtt{obs}_{\mathtt{dom}^M(a)}(s_0 \cdot \alpha a) = \mathtt{obs}_{\mathtt{dom}^M(a)}(s_0 \cdot \beta a)$. Together with $\mathtt{view}^{r(M)}_{\mathtt{dom}^{r(M)}(a)}(\alpha) = \mathtt{view}^{r(M)}_{\mathtt{dom}^{r(M)}(a)}(\beta)$, this gives the conclusion that $\mathtt{view}^{r(M)}_{\mathtt{dom}^{r(M)}(a)}(\alpha a) = \mathtt{view}^{r(M)}_{\mathtt{dom}^{r(M)}(a)}(\beta a)$. Thus, we in fact have $\mathtt{view}^{r(M)}_{\mathtt{dom}^{r(M)}(a)}(\alpha a) = \mathtt{view}^{r(M)}_{\mathtt{dom}^{r(M)}(a)}(\beta a)$ irrespective of whether $\mathtt{dom}^{r(M)}(a) = r(v)$.

Thus, in both the case that $\mathtt{dom}^M(a) \rightarrowtail_1 v$ and the case that $\mathtt{dom}^M(a) \not\rightarrowtail_1 v$, it follows from these that $\mathtt{ito}^M_v(\alpha a) = \mathtt{ito}^M_v(\beta a)$.

TA-security: By Theorem thm:ta-unwind-equiv, a system N is TA-secure if there exists a weak unwinding on the unfolded system $\mathtt{uf}(N)$. It is easy to verify that $\mathtt{uf}(r(M)) = r(\mathtt{uf}(M))$. Hence if M is TA-secure then there exists a weak unwinding on $\mathtt{uf}(M)$. By Theorem 5, there exists a weak unwinding on $r(\mathtt{uf}(M))$, hence on $\mathtt{uf}(r(M))$. It follows that $r(M)$ is TA-secure.

IP-security: We claim that if $X \subseteq D_1$ and $Y \subseteq D_2$ are sets of domains such that $r(X) \subseteq Y$, then $\mathtt{ipurge}^M_X(\alpha) = \mathtt{ipurge}^M_X(\mathtt{ipurge}^{r(M)}_Y(\alpha))$. Taking $Y =$

$\{r(v)\}$ and $X = \{v\}$, we then obtain $\mathtt{ipurge}_v^M(\alpha) = \mathtt{ipurge}_v^M(\mathtt{ipurge}_{r(v)}^{r(M)}(\alpha))$, so we may take $F_v(\alpha) = \mathtt{ipurge}_v^M(\alpha)$.

To prove the claim, we proceed by induction on the length of α. If S is set of domains and u a domain, we write $u \rightarrowtail S$ if there exists $v \in S$ such that $u \rightarrowtail v$. The base case is trivial. Consider a sequence of actions αa, where the claim holds for α, and suppose $r(X) \subseteq Y$. We consider a number of cases, depending on whether $\mathtt{dom}^{r(M)}(a) \rightarrowtail_2 Y$.

Suppose first that $\mathtt{dom}^{r(M)}(a) \not\rightarrowtail_2 Y$. Note that if we were to have $\mathtt{dom}^M(a) \rightarrowtail_1 v \in X$, then by the definition of refinement we would have $\mathtt{dom}^{r(M)}(a) = r(\mathtt{dom}^M(a)) \rightarrowtail_2 r(v) \in r(X)$, hence $\mathtt{dom}^{r(M)}(a) \rightarrowtail_2 Y$. Thus, we must in fact have $\mathtt{dom}^M(a) \not\rightarrowtail_1 X$. Hence

$$\begin{array}{lll}
\mathtt{ipurge}_X^M(\mathtt{ipurge}_Y^{r(M)}(\alpha a)) & & \\
\quad = \quad \mathtt{ipurge}_X^M(\mathtt{ipurge}_Y^{r(M)}(\alpha)) & & \text{since } \mathtt{dom}^{r(M)}(a) \not\rightarrowtail_2 Y \\
\quad = \quad \mathtt{ipurge}_X^M(\alpha) & & \text{by induction} \\
\quad = \quad \mathtt{ipurge}_X^M(\alpha a) & &
\end{array}$$

This proves the claim for the sequence αa.

Next, suppose that $\mathtt{dom}^{r(M)}(a) \rightarrowtail_2 Y$. We further break this case into two subcases, depending on whether $\mathtt{dom}^M(a) \rightarrowtail_1 X$. Suppose first that $\mathtt{dom}^M(a) \not\rightarrowtail_1 X$. Note that $r(X) \subseteq Y$ implies $r(X) \subseteq Y \cup \{\mathtt{dom}^M(a)\}$. Hence

$$\begin{array}{lll}
\mathtt{ipurge}_X^M(\mathtt{ipurge}_Y^{r(M)}(\alpha a)) & & \\
\quad = \quad \mathtt{ipurge}_X^M(\mathtt{ipurge}_{Y\cup\{\mathtt{dom}^{r(M)}(a)\}}^{r(M)}(\alpha)a) & & \text{since } \mathtt{dom}^{r(M)}(a) \rightarrowtail_2 Y \\
\quad = \quad \mathtt{ipurge}_X^M(\mathtt{ipurge}_{Y\cup\{\mathtt{dom}^{r(M)}(a)\}}^{r(M)}(\alpha)) & & \text{since } \mathtt{dom}^M(a) \not\rightarrowtail_1 X \\
\quad = \quad \mathtt{ipurge}_X^M(\alpha) & & \text{by induction} \\
\quad = \quad \mathtt{ipurge}_X^M(\alpha a) & & \text{since } \mathtt{dom}^M(a) \not\rightarrowtail_1 X.
\end{array}$$

Alternately, suppose that $\mathtt{dom}^M(a) \rightarrowtail_1 X$. Note that $r(X \cup \{\mathtt{dom}^M(a)\}) \subseteq Y \cup \{r(\mathtt{dom}^M(a))\} = Y \cup \{\mathtt{dom}^{r(M)}(a)\}$. Hence

$$\begin{array}{lll}
\mathtt{ipurge}_X^M(\mathtt{ipurge}_Y^{r(M)}(\alpha a)) & & \\
\quad = \quad \mathtt{ipurge}_X^M(\mathtt{ipurge}_{Y\cup\{\mathtt{dom}^{r(M)}(a)\}}^{r(M)}(\alpha)a) & & \text{since } \mathtt{dom}^{r(M)}(a) \rightarrowtail_2 Y \\
\quad = \quad \mathtt{ipurge}_{X\cup\{\mathtt{dom}^M(a)\}}^M(\mathtt{ipurge}_{Y\cup\{\mathtt{dom}^{r(M)}(a)\}}^{r(M)}(\alpha))a & & \text{since } \mathtt{dom}^M(a) \rightarrowtail_1 X \\
\quad = \quad \mathtt{ipurge}_{X\cup\{\mathtt{dom}^M(a)\}}^M(\alpha)a & & \text{by induction} \\
\quad = \quad \mathtt{ipurge}_X^M(\alpha a) & & \text{since } \mathtt{dom}^M(a) \rightarrowtail_1 X.
\end{array}$$

This completes the proof of the claim. □

We remark that it can be shown independently of other assumptions that if X is one of `purge`, `ipurge`, `ta`, or `to` then (with all parameters made explicit), if

$$X(r(M), r(v), \alpha, \rightarrowtail_2) = X(r(M), r(v), \alpha', \rightarrowtail_2)$$

then

$$X(M, v, \alpha, \rightarrowtail_1) = X(M, v, \alpha', \rightarrowtail_1).$$

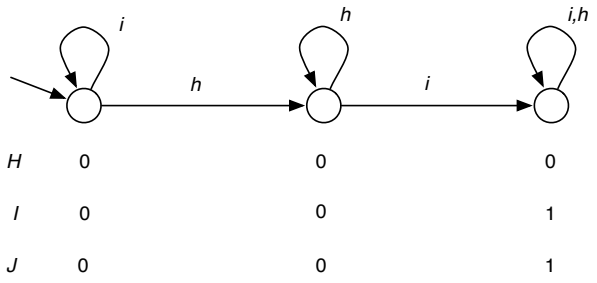

Figure 3: `ito` not preserved under refinement

The following example shows this is not true for $X = \texttt{ito}$, but the proof of Theorem 4 handles this case by showing that it does hold under the further assumption that M is ITO-secure.

Example 4: Consider system in Figure 3, where there are domains H, I, J with actions h of domain H and i of domain I. Observations in the domains are given under the states in the order H, I, J. Intuitively, action i is used to test whether there has been an occurrence of h. If so, this fact is made observable to both I and J simultaneously.

Let $\rightarrowtail_1$ be the reflexive closure on $\{H, I, J\}$ of the fact $I \rightarrowtail_1 J$, and let r be given by $r(H) = H$ and $r(I) = r(J) = K$. Let $\rightarrowtail_2$ be the smallest reflexive relation on $\{H, K\}$. Then r is a refinement mapping. Take $\alpha = hi$ and $\beta = i$. Then

$$\begin{array}{rcl} \texttt{ito}_K^{r(M)}(\alpha) & = & (\texttt{ito}_K^{r(M)}(h),\ \texttt{view}_K^{r(M)}(h), i) \\ & = & (\texttt{ito}_K^{r(M)}(\epsilon),\ (0^I 0^J),\ i) \\ & = & (\epsilon,\ (0^I 0^J),\ i) \\ & = & (\texttt{ito}_K^{r(M)}(\epsilon),\ \texttt{view}_K^{r(M)}(\epsilon), i) \\ & = & \texttt{ito}_K^{r(M)}(\beta) \end{array}$$

where $(0^I 0^J)$ denotes the function mapping I to 0 and J to 0. However,

$$\begin{array}{rcl} \texttt{ito}_J^{M}(\alpha) & = & (\texttt{ito}_J^{r(M)}(h),\ \texttt{view}_I^{r(M)}(h), i) \\ & = & (\texttt{ito}_J^{M}(\epsilon),\ 0i1,\ i) \end{array}$$

and

$$\begin{array}{rcl} \texttt{ito}_J^{M}(\beta) & = & (\texttt{ito}_J^{M}(\epsilon),\ \texttt{view}_I^{r(M)}(i), i) \\ & = & (\texttt{ito}_J^{M}(\epsilon),\ 0i0,\ i) \end{array}$$

so although $r(J) = K$, we do not have that $\texttt{ito}_J^M(\alpha) = \texttt{ito}_J^M(\beta)$. □

5 Access Control

The interpretations of policies discussed above are somewhat abstract. We now consider a further, more concrete interpretation, that is closer to the level of *mechanisms* for control of information flow. One of the key mechanisms for implementation of secure systems is access control. Rushby [Rus92] defined the notion of access control system and showed that access control systems are IP-secure with respect to a policy $\rightarrowtail$ if they are consistent with the policy in an appropriate sense. We present here a variant of Rushby's definitions due to van der Meyden, that strengthens Rushby's result (see [Mey08] for an explanation of how this variant improves on Rushby's.)

Define an *access control structure* for a machine $\langle S, s_0, A, \mathtt{step}, \mathtt{obs}, \mathtt{dom}\rangle$ with domains D to be a tuple $\mathtt{AC} = (N, V, \mathtt{contents}, \mathtt{alter}, \mathtt{observe})$, where

1. N is a set, of *names*,
2. V is a set, of *values*,
3. $\mathtt{contents} : S \times N \rightarrow V$, with $\mathtt{contents}(s, n)$ interpreted as the value of object n in state s,
4. $\mathtt{observe} : D \rightarrow \mathcal{P}(N)$, with $\mathtt{observe}(u)$ interpreted as the set of objects that domain u may observe, and
5. $\mathtt{alter} : D \rightarrow \mathcal{P}(N)$, with $\mathtt{alter}(u)$ interpreted as the set of objects whose values domain u is permitted to alter.

We call the pair $(M, \mathtt{AC})$ a system with structured state. For a system with structured state, define for each domain $u \in D$, an equivalence relation on the states S of M, of *observable content equivalence*, by $s \sim_u^{\mathtt{oc}} t$ if $\mathtt{contents}(s, n) = \mathtt{contents}(t, n)$ for all $n \in \mathtt{observe}(u)$. That is, two states are related for u if they are identical with respect to the values of objects that u may observe.

The following conditions are a variant of Rushby's "Reference Monitor" conditions.

> WAC1. If $s \sim_u^{\mathtt{oc}} t$ then $\mathtt{obs}_u(s) = \mathtt{obs}_u(t)$.
>
> WAC2. For all actions a, states s, t and objects $n \in \mathtt{alter}(dom(a))$, if $s \sim_{\mathtt{dom}(a)}^{\mathtt{oc}} t$ and $\mathtt{contents}(s, n) = \mathtt{contents}(t, n)$ then $\mathtt{contents}(s \cdot a, n) = \mathtt{contents}(t \cdot a, n)$.
>
> WAC3. If $\mathtt{contents}(s \cdot a, n) \neq \mathtt{contents}(s, n)$ then $n \in \mathtt{alter}(dom(a))$.

Intuitively, WAC1-WAC3 capture the conditions under which the machine operates in accordance with the intuitive interpretations of the structure $\mathtt{AC}$. WAC1 and WAC3 are identical to Rushby's RM1 and RM3, respectively. WAC1 says that a domain's observation depends only on the values of the objects observable to it. WAC2 (a modification of Rushby's condition RM2) says that if an action in domain u is permitted to change the value of an object n, then the

new value of n depends only on its old value and the values of objects of domain u. WAC3 says that if an action can change the value of an object, then the domain of that action is in fact permitted to alter that object. We call the pair $(M, \mathtt{AC})$ a *weak access control system* if it satisfies WAC1-WAC3. We say that M *admits a weak access control interpretation* if there exists an access control structure $\mathtt{AC}$ such that $(M, \mathtt{AC})$ is a weak access control system.

Plainly, if there is an object that domain u may alter and domain v may observe, then information flow from domain u to v cannot be prevented. We say that a policy $\rightarrowtail$ is *consistent* with a weak access control system if the following condition is satisfied:

AOI. If $\mathtt{alter}(u) \cap \mathtt{observe}(v) \neq \emptyset$ then $u \rightarrowtail v$.

Access control interpretations are closely related to TA-security in a way that is similar to the connection between weak unwinding and TA-security already noted above in Theorem 3. The following result is shown in [Mey08].

Theorem 6 *The following are equivalent*

1. *M is TA-secure with respect to $\rightarrowtail$,*
2. *$\mathtt{uf}(M)$ admits a weak access control interpretation consistent with $\rightarrowtail$,*

Moreover, if M admits a weak access control interpretation consistent with $\rightarrowtail$, then so does $\mathtt{uf}(M)$ (hence M is TA-secure).

This result shows that TA-security captures, in a precise sense, the notion of information flow security that may be enforced by access control mechanisms.

We now consider the interaction of refinement and access control structure. Suppose that M is a system for the set of domains D_1, with access control structure $\mathtt{AC}_1 = (N_1, V_1, contents_1, \mathtt{observe}_1, \mathtt{alter}_1)$ with respect to which M is a weak access control system. Let $r : (D_1, \rightarrowtail_1) \rightarrow (D_2, \rightarrowtail_2)$ be a refinement mapping. Define $r(\mathtt{AC}_1) = (N_2, V_2, \mathtt{contents}_2, \mathtt{observe}_2, \mathtt{alter}_2)$ to be the access control structure given by

1. $N_2 = N_1$ and $V_2 = V_1$ and $\mathtt{contents}_2 = \mathtt{contents}_1$,
2. $\mathtt{observe}_2(u) = \cup_{v \in r^{-1}(u)}\ \mathtt{observe}_1(v)$,
3. $\mathtt{alter}_2(u) = \cup_{v \in r^{-1}(u)}\ \mathtt{alter}_1(v)$.

Intuitively, $\mathtt{AC}_2$ has the same set of objects, with the same contents at each state of M as in $\mathtt{AC}_1$, and each domain of D_2 observes and alters the objects in all of its subdomains in D_1. The following result shows that weak access control structure is preserved under architectural refinement.

Theorem 7 *Let $r : (D_1, \rightarrowtail_1) \rightarrow (D_2, \rightarrowtail_2)$ be a refinement mapping.*

1. *If $(M, \mathtt{AC}_1)$ is a weak access control system then $(r(M), r(\mathtt{AC}_1))$ is a weak access control system.*

2. If $\mathtt{AC}_1$ *is consistent with* $\rightarrowtail_1$ *then* $r(\mathtt{AC}_1)$ *is consistent with* $\rightarrowtail_2$.

Proof: Write $\sim^1$ and $\sim^2$ for the relations of observable content equivalence on the states of M with respect to $\mathtt{AC}_1$ and $\mathtt{AC}_2 = r(\mathtt{AC}_1)$, respectively. Let O^2 be the observation function on $r(M)$. Since $\mathtt{contents}_1 = \mathtt{contents}_2$, we write just $\mathtt{contents}$ in either case. For part (1), we prove WAC1-WAC3.

WAC1: We need to show that $s \sim^2_u t$ implies $O^2_u(s) = O^2_u(t)$, for all $u \in D_2$. By definition, $s \sim^2_u t$ implies that $s \sim^1_v t$ for all $v \in r^{-1}(u)$. By WAC1 for M and $\mathtt{AC}_1$, we have $O^1_v(s) = O^1_v(t)$ for all $v \in r^{-1}(u)$. By construction of O^2 we obtain that $O^2_u(s) = O^2_u(t)$.

WAC3: Suppose that $n \notin \mathtt{alter}_2(\mathtt{dom}^2(a))$. Then for all $v \in r^{-1}(u)$, we have $n \notin \mathtt{alter}_1(v)$. In particular, since $r(\mathtt{dom}^1(a)) = \mathtt{dom}^2(a)$, we have $n \notin \mathtt{alter}_1(\mathtt{dom}^1(a))$. By WAC3 for $\mathtt{AC}_1$, we obtain that $\mathtt{contents}(s \cdot a, n) = \mathtt{contents}(s, n)$, as required.

WAC2: Let a be an action of M with $\mathtt{dom}^1(a) = v$ and $\mathtt{dom}^2(a) = u = r(v)$. Suppose that $n \in \mathtt{alter}_2(u)$ and $s \sim^2_u t$ and $\mathtt{contents}(s, n) = \mathtt{contents}(t, n)$ We need to show that $\mathtt{contents}(s \cdot a, n) = \mathtt{contents}(t \cdot a, n)$. There are two possibilities: $n \in \mathtt{alter}_1(v)$ or $n \notin \mathtt{alter}_1(v)$. In case $n \notin \mathtt{alter}_1(v)$, then we have by what we proved above for WAC3 that $\mathtt{contents}(s \cdot a, n) = \mathtt{contents}(s, n)$ and $\mathtt{contents}(t, n) = \mathtt{contents}(t \cdot a, n)$, from which it is immediate that $\mathtt{contents}(s \cdot a, n) = \mathtt{contents}(t \cdot a, n)$, as required. Alternately, suppose that $n \in \mathtt{alter}_1(v)$. By definition of $\mathtt{observe}^2$ and $\mathtt{contents}^2$ and the fact that $v \in r^{-1}(u)$, it follows that $s \sim^1_v t$. Thus, by WAC2 for $\mathtt{AC}_1$, we again obtain that $\mathtt{contents}(s \cdot a, n) = \mathtt{contents}(t \cdot a, n)$.

For part (2) suppose $\mathtt{AC}_1$ is consistent with $\rightarrowtail_1$. Let $\mathtt{alter}_2(u_1) \cap \mathtt{observe}_2(u_2) \neq \emptyset$, with this set containing, say, the object n. Then there exists $v_1 \in r^{-1}(u_1)$ and $v_2 \in r^{-1}(u_2)$ such that $n \in \mathtt{alter}_1(v_1)$ and $v \in \mathtt{observe}_1(v_2)$. Thus $\mathtt{alter}_1(v_1) \cap \mathtt{observe}_1(v_2) \neq \emptyset$, and by WAC3 on $\mathtt{AC}_1$, we obtain that $v_1 \rightarrowtail_1 v_2$. Since r is a refinement mapping, it follows that $u_1 = r(v_1) \rightarrowtail_2 r(v_2) = u_2$, as required. □

It is worth noting that we could also work with a more expressive notion of access control structure in which the functions $\mathtt{alter}$ and $\mathtt{observe}$ take as inputs the *actions* of a system M (rather than the domains.) Intuitively, this amounts to specifying constraints on the objects that each action is permitted to read and write. We will call an access control structure of this type an *action-based* access control structure, and refer to the former type as a *domain-based* access control structure.

Given $\mathtt{AC} = (N, V, \mathtt{contents}, \mathtt{observe}, \mathtt{alter})$, an action-based access control structure over actions A, and a domain mapping $\mathtt{dom} : A \to D$, we may construct $\mathtt{AC}/\mathtt{dom} = (N, V, \mathtt{contents}, \mathtt{observe}', \mathtt{alter}')$, a domain-based access control structure, by defining

$$\mathtt{observe}'(u) = \cup\{\mathtt{observe}(a) \mid a \in A,\ \mathtt{dom}(a) = u\}$$

and

$$\mathtt{alter}'(u) = \cup\{\mathtt{alter}(a) \mid a \in A,\ \mathtt{dom}(a) = u\}.$$

Given a system M, with domain u, let $\sim_u^{\mathtt{oc}}$ be the relation defined above, with respect to $\mathtt{AC/dom}$, and for an *action* a define the relation $\sim_a^{\mathtt{oc}}$ on the states of M by $s \sim_a^{\mathtt{oc}} t$ if $\mathtt{contents}(s,n) = \mathtt{contents}(t,n)$ for all $n \in \mathtt{observe}(a)$.

The semantic conditions defining an action-based access control system can now be stated as a variant of WAC1-WAC3.

> WAC1$_a$. For $u \in D$, if $s \sim_u^{\mathtt{oc}} t$ then $\mathtt{obs}_u(s) = \mathtt{obs}_u(t)$.
>
> WAC2$_a$. For all actions a, states s, t and names $n \in \mathtt{alter}(a)$, if $s \sim_a^{\mathtt{oc}} t$ and $\mathtt{contents}(s,n) = \mathtt{contents}(t,n)$ then $\mathtt{contents}(s{\cdot}a, n) = \mathtt{contents}(t{\cdot}a, n)$.
>
> WAC3$_a$. If $\mathtt{contents}(s \cdot a, n) \neq \mathtt{contents}(s,n)$ then $n \in \mathtt{alter}(a)$.

Intuitively, WAC1$_a$ says that the observations of a domain depend only on the contents of objects that actions in that domain may observe. The remaining conditions are similar to the domain-based versions, except that we work at the level of actions rather than domains.

Proposition 1 *If M satisfies WAC1$_a$-WAC3$_a$ with respect to the action-based access control structure* $\mathtt{AC}$ *and $\rightarrowtail$, then M satisfies WAC1-WAC3 with respect to* $\mathtt{AC/dom}$ *and $\rightarrowtail$.*

Example 5: To illustrate the interaction of architectural refinement and the implementation of architectures using access control structure, we reconsider the refinement of Example 2. As a further step of refinement towards the implementation of the system, we choose to implement architecture $\mathcal{B}$ using an action-based access control structure $(N, V, \mathtt{contents}, \mathtt{observe}, \mathtt{alter})$. The set N consists of the following objects:

1. local data $l_1, l_2, h_1, h_2, d, hdb$ for $L_1, L_2, H_1, H_2, D, HDB$, respectively,
2. high level files f_1, f_2
3. input buffers hin, din_h, din_l, lin for messages to H, D and L. (In the case of D, we have separate buffers din_h and din_l to receive communications from H and L respectively — this allows the sender to receive acknowledgements without creating a covert channel from H to L.)

The table in Figure 4 gives the actions associated to each domain, and the functions $\mathtt{observe}$ and $\mathtt{alter}$.

Finally, we define observations in the system using the pairs $(L_i, \{l_i\})$, $(D, \{d\})$, $(H_i, \{h_i\})$, $(HDB, \{hdb, f_1\})$. Here the first component of a pair gives a domain u, and the observation $O_u(s)$ at state s is defined to be the sequence of values $\mathtt{contents}(s,n)$ where n is an element of the second component. Note that we have not made f_2 observable - this might represent, e.g., that f_2 is used for internal data structures of the database and is not visible at the interface of the database.

Domain	Actions	`observe`	`alter`	Purpose
$L_i,\ i=1,2$	request(L_i)	l_i	din_l, l_i	request information from D
	send(L_i, H)	l_i	hin, l_i	send information to H
	get(L_i)	lin	lin, l_i	read L input buffer
	internal(L_i)	l_i	l_i	local computation
D	get$_h$(D)	din_h	din_h, d	read D input buffer from H
	get$_l$(D)	din_l	din_l, d	read D input buffer from L
	query(D)	d	d, hdb	send query to H database
	respond(D)	d	d, lin	send response to L request
	internal(D)	d	d	local computation
$H_i,\ i=1,2$	request(H_i)	h_i	h_i, hdb	send query/update to HDB
	internal(H_i)	h_i	h_i	local computation
HDB	get(HDB)	hin	hin, hdb	read H input buffer
	respond(HDB, H_i)	hdb, f_1, f_2	h_i, hdb	respond to H_i request
	respond(HDB, D)	hdb, f_1, f_2	d, hdb	respond to D request
	internal(HDB)	hdb, f_1, f_2	hdb, f_1, f_2	local computation

Figure 4: Actions of an access control system

It is straightforward to check that this action-based access control structure induces a domain-based access control structure that is consistent with the policy of $\mathcal{B}$. Thus, by Theorem 6, any system M for this access control structure that satisfies the conditions WAC1$_a$-WAC3$_a$ is TA-compliant with $\mathcal{B}$. Further, using either the preservation of TA-compliance under refinement (Theorem 4), or the preservation of access control structure under refinement (Theorem 7) and then Theorem 6, it also follows that $r(M)$ is TA-compliant with $\mathcal{A}$. Here $r(M)$ is the system where the observations are defined by the pairs $(L, \{l_1, l_2\})$, $(D, \{d\})$ and $(H, \{h_1, h_2, hdb, f_1\})$.

Note that this result applies to a *class* of systems, as we have not yet defined a unique system. To do so we would need to also give the values V, and define the effect that actions have on states. Once this is done, one way to ensure WAC2$_a$ and WAC3$_a$ might be by using static analysis to verify that the code implementing an action a reads only from `observe`(a) and writes only to `alter`(a). □

6 System Refinement

A plausible intuition concerning security is that reducing the amount of information that domains can observe will make the system *more* secure. Intuitively, this is because, with less information, domains are less likely to be able to learn secrets that they are not supposed to know. In this section we consider refinement from this perspective. Not all the definitions we consider in this paper, it turns out, support the intuition.

Suppose that $M = \langle S, s_0, A, \mathtt{dom}, \mathtt{step}, \mathtt{obs}\rangle$ and $M' = \langle S', s'_0, A, \mathtt{dom}, \mathtt{step}', \mathtt{obs}'\rangle$

are two systems with the same set of domains D, actions A and domain assignment $\mathtt{dom}$. Write $M \leq M'$ if for all sequences $\alpha, \beta \in A^*$, and all domains $u \in D$ we have $\mathtt{obs}'_u(s'_0 \cdot \alpha) = \mathtt{obs}'_u(s'_0 \cdot \beta)$ implies $\mathtt{obs}_u(s_0 \cdot \alpha) = \mathtt{obs}_u(s_0 \cdot \beta)$. That is, the observations in system M contain less information than those in system M'.

For reasons that will become clear below, it turns out to be useful to have a more general version of this notion. Let X be a function mapping domains $u \in D$ and sequences $\alpha \in A^*$ to some value. We write $M \leq^X M'$ if for all sequences $\alpha, \beta \in A^*$, and all domains $u \in D$, if $X_u(\alpha) = X_u(\beta)$ then $\mathtt{obs}'_u(s'_0 \cdot \alpha) = \mathtt{obs}'_u(s'_0 \cdot \beta)$ implies $\mathtt{obs}_u(s_0 \cdot \alpha) = \mathtt{obs}_u(s_0 \cdot \beta)$. Note that $M \leq M'$ implies $M \leq^X M'$ for all X, so this is a weaker notion.

Say that the system M is X-secure if for all $\alpha, \alpha' \in A^*$ and $u \in D$, if $X_u(\alpha) = X_u(\alpha')$ then $\mathtt{obs}_u(s_0 \cdot \alpha) = \mathtt{obs}_u(s_0 \cdot \alpha')$. Plainly, the notions of P-security, TO-security, ITO-security, TA-security and IP-security of a system M with respect to a policy $\rightarrowtail$ correspond to X-security for appropriate choices of the function X, viz., $\mathtt{purge}$, $\mathtt{to}$, $\mathtt{ito}$, $\mathtt{ta}$ and $\mathtt{ipurge}$ as defined with respect to M and $\rightarrowtail$.

The following result formalises the intuition that reducing the information in observations makes a system more secure.

Proposition 2 *If $M \leq^X M'$ and M' is X-secure then M is X-secure.*

Proof: Suppose $M \leq^X M'$ and M' is X-secure. Consider $\alpha, \alpha' \in A^*$ and $u \in D$ such that $X_u(\alpha) = X_u(\alpha')$. Then $\mathtt{obs}'_u(s'_0 \cdot \alpha) = \mathtt{obs}'_u(s'_0 \cdot \alpha')$ by X-security of M'. It follows using $M \leq^X M'$ that $\mathtt{obs}_u(s_0 \cdot \alpha) = \mathtt{obs}_u(s_0 \cdot \beta)$. This is exactly what we need to show that M is X-secure. □

Corollary 1 *For X any of P, TA, or IP, if $M \leq^X M'$ and M' is X-secure with respect to $\rightarrowtail$ then M is X-secure with respect to $\rightarrowtail$.*

We note that we need to take some more care with the notions of ITO-security and TO-security, since the functions $\mathtt{ito}$ and $\mathtt{to}$ depend on the observations in the system to which they are applied, whereas Proposition 2 concerns the *same* function X in the systems M and M'. Indeed, it is *not* the case that if $M \leq M'$ and M' is TO-secure (ITO-secure), then M is TO-secure (ITO-secure).

Example 6: For this, consider the system M' in Figure 5, where there are domains H, D, L with policy $H \rightarrowtail D \rightarrowtail L$. Intuitively, H is a high security domain, D is a downgrader and the downgrader action d releases the information that the H action h has been performed. This system is TO-secure (hence ITO-secure also). For, suppose $\mathtt{obs}_L(s_0 \cdot \alpha) = 1$ and $\mathtt{obs}_L(s_0 \cdot \beta) = 0$. Then β does not contain an h followed later by a d, so must be of the form $\beta = d^k h^l$ with $k, l \geq 0$. Thus, $\mathtt{to}_L(\beta) = \mathtt{to}_L(d^k)$, which contains no D view containing observation 1. On the other hand, α contains an h and later d, so $\mathtt{to}_L(\beta)$ contains a D view containing observation 1. Thus, $\mathtt{to}_L(\alpha) \neq \mathtt{to}_L(\beta)$, and there

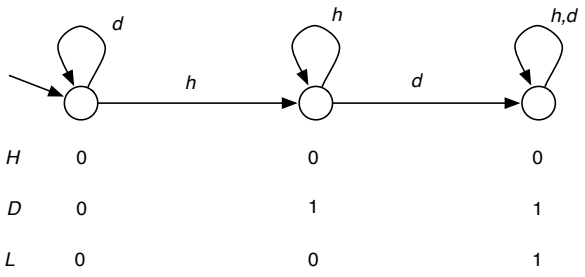

Figure 5: ITO-security and TO-security not preserved under system refinement

can be no violation of the condition for TO-security involving L. The argument for D is similar, and there is nothing to prove for L.

On the other hand, let M be the system obtained from M' by reducing the information observable to D, taking $\mathtt{obs}_D(s) = \bot$ for all states s. Then in M we have

$$\begin{array}{rcl} \mathtt{ito}_L(hd) & = & (\mathtt{to}_L(h), \mathtt{view}_D(hd), d) \\ & = & (\mathtt{ito}_L(\epsilon), \bot d \bot, d) \\ & = & (\mathtt{ito}_L(\epsilon), \mathtt{view}_D(d), d) \\ & = & \mathtt{ito}_L(d) \end{array}$$

but $\mathtt{obs}_L(hd) = 1$ and $\mathtt{obs}_L(d) = 1$. Hence M is not ITO-secure (and therefore also not TO-secure). □

Intuitively, TO-security and ITO-security do more than place upper bounds on domains' information: they also add a type of lower bound constraint: information observable to a recipient must also have been observed by its sender. Thus, a further constraint on the systems M and M' is required in order for these properties to be preserved by the systems refinement. The following result identifies a sufficient condition for preservation of these properties.

For a sequence of actions α, and a domain u define $\mathtt{res}_u(\alpha)$ to be $\alpha \restriction \{u\}$, i.e. the subsequence of actions of domain u.

Proposition 3 *Suppose that $M \leq^{\mathtt{res}} M'$ and for all sequences of actions α, α' and domains u, if $\mathtt{view}_u^M(\alpha) = \mathtt{view}_u^M(\beta)$ then $\mathtt{view}_u^{M'}(\alpha) = \mathtt{view}_u^{M'}(\beta)$. Then if M' is TO-secure (ITO-secure) with respect to $\rightarrowtail$ then M is TO-secure (ITO-secure) with respect to $\rightarrowtail$.*

Proof: An easy induction using the property on the views shows that $\mathtt{to}_u^M(\alpha) = \mathtt{to}_u^M(\beta)$ implies $\mathtt{to}_u^{M'}(\alpha) = \mathtt{to}_u^{M'}(\beta)$. This yields the result. For, we get as an immediate consequence of this and the TO-security of M' that $\mathtt{to}_u^M(\alpha) = \mathtt{to}_u^M(\beta)$ implies $\mathtt{obs}^{M'}(s_0 \cdot \alpha) = \mathtt{obs}^{M'}(s_0 \cdot \beta)$. Also, by an easy induction, $\mathtt{to}_u^M(\alpha) = \mathtt{to}_u^M(\beta)$ implies $\mathtt{res}_u(\alpha) = \mathtt{res}_u(\beta)$. Thus, using $M \leq^{\mathtt{res}} M'$, we conclude that $\mathtt{to}_u^M(\alpha) = \mathtt{to}_u^M(\beta)$ implies $\mathtt{obs}^M(s_0 \cdot \alpha) = \mathtt{obs}^M(s_0 \cdot \beta)$, as required for TO-security. The argument in the case of ITO-security is similar. □

We remark that under the condition $M \leq^{\mathtt{res}} M'$, we in fact have the determination of views in the other direction as well.

Proposition 4 *Suppose that $M \leq^{\mathtt{res}} M'$. Then for all sequences of actions α, β, if $\mathtt{view}_u^{M'}(\alpha) = \mathtt{view}_u^{M'}(\beta)$ then $\mathtt{view}_u^{M}(\alpha) = \mathtt{view}_u^{M}(\beta)$.*

Proof: Since $M \leq^{\mathtt{res}} M'$, there exists for each domain u a function $f_u : A_u^* \times O' \to O$ mapping sequences of actions of domain u and observations in M' to observations in M, such that for all sequences $\alpha \in A_u^*$, we have

$$f_u(\mathtt{res}_u(\alpha), O^{M'}(s'_0 \cdot \alpha)) = O^M(s_0 \cdot \alpha).$$

For each domain u, define the mapping F_u from views in M' to views in M inductively, by

1. $F_u(o) = f_u(\epsilon, o)$, when o is an observation,
2. $F_u(\sigma a) = F_u(\sigma)a$ when $a \in A$, and
3. $F_u(\sigma o) = F_u(\sigma) \circ f_u(\sigma \restriction A_u, o)$ when o is an observation and $\sigma \neq \epsilon$.

We claim that for all sequences $\alpha \in A^*$, we have $F_u(\mathtt{view}_u^{M'}(\alpha)) = \mathtt{view}_u^{M}(\alpha)$. The base case of $\alpha = \epsilon$ is immediate from the definitions. For the inductive case of a sequence αa where $a \in A$, consider three cases:

1. Case 1: $\mathtt{dom}(a) \neq u$ and $O_u^{M'}(s'_0 \cdot \alpha) = O_u^{M'}(s'_0 \cdot \alpha a)$. Note that in this case we have $\mathtt{res}_u(\alpha a) = \mathtt{res}_u(\alpha)$, hence $O_u^{M}(s'_0 \cdot \alpha) = O_u^{M}(s'_0 \cdot \alpha a)$ by the fact that $M \leq^{\mathtt{res}} M'$. Hence

$$\begin{array}{rcl} F_u(\mathtt{view}_u^{M'}(\alpha a)) & = & F_u(\mathtt{view}_u^{M'}(\alpha)) \\ & = & \mathtt{view}_u^{M}(\alpha) \quad \text{(by induction)} \\ & = & \mathtt{view}_u^{M}(\alpha a) \end{array}$$

2. Case 2: $\mathtt{dom}(a) \neq u$ and $O_u^{M'}(s'_0 \cdot \alpha) \neq O_u^{M'}(s'_0 \cdot \alpha a)$. Note that in this case we have $\mathtt{res}_u(\alpha a) = \mathtt{res}_u(\alpha) = \mathtt{view}_u^{M'}(\alpha) \restriction A_u$. Hence

$$\begin{array}{rcl} F_u(\mathtt{view}_u^{M'}(\alpha a)) & = & F_u(\mathtt{view}_u^{M'}(\alpha)\, O_u^{M'}(s'_0 \cdot \alpha a)) \\ & = & F_u(\mathtt{view}_u^{M'}(\alpha)) \circ f_u(\mathtt{view}_u^{M'}(\alpha) \restriction A_u,\, O_u^{M'}(s'_0 \cdot \alpha a)) \\ & = & F_u(\mathtt{view}_u^{M'}(\alpha)) \circ f_u(\mathtt{res}_u(\alpha a),\, O_u^{M'}(s'_0 \cdot \alpha a)) \\ & = & \mathtt{view}_u^{M}(\alpha) \circ O_u^{M}(s_0 \cdot \alpha a)) \\ & = & \mathtt{view}_u^{M}(\alpha a) \end{array}$$

3. Case 3: $\mathtt{dom}(a) = u$. Then

$$\begin{array}{rcl} F_u(\mathtt{view}_u^{M'}(\alpha a)) & = & F_u(\mathtt{view}_u^{M'}(\alpha)\, a\, O_u^{M'}(s'_0 \cdot \alpha a)) \\ & = & F_u(\mathtt{view}_u^{M'}(\alpha))\, a\, f_u(\mathtt{view}_u^{M'}(\alpha) a \restriction A_u,\, O_u^{M'}(s'_0 \cdot \alpha a)) \\ & = & F_u(\mathtt{view}_u^{M'}(\alpha))\, a\, f_u(\mathtt{res}_u(\alpha a),\, O_u^{M'}(s'_0 \cdot \alpha a)) \\ & = & \mathtt{view}_u^{M}(\alpha)\, a\, O_u^{M}(s_0 \cdot \alpha a)) \\ & = & \mathtt{view}_u^{M}(\alpha a) \end{array}$$

□

Thus, under the conditions of Proposition 3, we have $\mathtt{view}_u^{M'}(\alpha) = \mathtt{view}_u^{M'}(\beta)$ iff $\mathtt{view}_u^{M}(\alpha) = \mathtt{view}_u^{M}(\beta)$. This may seem to make Proposition 3 uninterestingly weak. Below, we show that Proposition 3 is not as weak as it may seem, by giving a non-trivial application of Proposition 3 in which information is spread out through the views in M and M' rather different ways.

7 Action Observed Systems

The definitions of the previous sections are concerned with a machine model in which observations are made on states. A variant model has been considered in the literature, in which observations are associated to actions instead [Rus92]. A comparison between the two models for the set of semantics of intransitive information flow policies discussed above was carried out in [Mey07]. In this section we consider how the results on refinement in state-observed systems carry over to this model.

An *action-observed* machine is a tuple $\langle S, s_0, A, \mathtt{step}, \mathtt{out}, \mathtt{dom}\rangle$, where all the components are as in the state observed system model, except that the observation function $\mathtt{obs}$ is replaced by a function $\mathtt{out} : S \times A \to O$. Intuitively, if s is a state and a is an action, then $\mathtt{out}(s, a)$ is the output, or return value, observed in domain $\mathtt{dom}(a)$ when action a is performed.

Each of the definitions of security for the state-observed system model has the form

> M is secure with respect to $\rightarrowtail$ if for all sequences α, α' and domains u, if $X_u(\alpha) = X_u(\alpha')$ then $\mathtt{obs}_u(s_0 \cdot \alpha) = \mathtt{obs}_u(s_0 \cdot \alpha')$.

where $X_u(\alpha)$ is a function of $\alpha, u, \rightarrowtail$ and M. We may obtain corresponding action-observed versions that have the form

> M is secure with respect to $\rightarrowtail$ if for all sequences α, α', domains u and actions a with $\mathtt{dom}(a) = u$, if $X_u(\alpha) = X_u(\alpha')$ then $\mathtt{out}(s_0 \cdot \alpha, a) = \mathtt{out}(s_0 \cdot \alpha', a)$.

In the cases of P-security, TA-security and IP-security, the corresponding functions $\mathtt{purge}$, $\mathtt{ta}$ and $\mathtt{ipurge}$ in fact depend only on u, α and $\rightarrowtail$, and we may use exactly the same functions as X to obtain the action-observed definitions of security.

In the case of TO-security and ITO-security, there is also a dependence on observations in the model, which become outputs in the action-observed case. Here we need to reformulate the definitions somewhat. This is done as follows. First, the notion of view is adapted to the action-observed system model by defining $\mathtt{view}_u^a : A^* \to (A \cup O)^*$ for $u \in D$ inductively by $\mathtt{view}_u^a(\epsilon) = \epsilon$, and

$$\mathtt{view}_u^a(\alpha a) = \begin{cases} \mathtt{view}_u^a(\alpha)\, a\, \mathtt{out}(s_0 \cdot \alpha, a) & \text{if } \mathtt{dom}(a) = u \\ \mathtt{view}_u^a(\alpha) & \text{otherwise.} \end{cases}$$

That is, the view of a domain is just the sequence of actions that the domain has performed, together with the outputs obtained from those actions.

We now define an action-observed variant $\mathtt{to}^a_u$ of $\mathtt{to}_u$, by $\mathtt{to}^a_u(\epsilon) = \epsilon$ and

$$\mathtt{to}^a_u(\alpha a) = \begin{cases} \mathtt{to}^a_u(\alpha) & \text{if } \mathtt{dom}(a) \not\rightarrowtail u, \\ (\mathtt{to}^a_u(\alpha), \mathtt{view}^a_{\mathtt{dom}(a)}(\alpha a), a) & \text{if } \mathtt{dom}(a) = u, \\ (\mathtt{to}^a_u(\alpha), \mathtt{view}^a_{\mathtt{dom}(a)}(\alpha), a) & \text{if } u \neq \mathtt{dom}(a) \rightarrowtail u. \end{cases}$$

Similarly, $\mathtt{ito}^a_u$ is defined by $\mathtt{ito}^a_u(\epsilon) = \epsilon$ and

$$\mathtt{ito}^a_u(\alpha a) = \begin{cases} \mathtt{ito}^a_u(\alpha) & \text{if } \mathtt{dom}(a) \not\rightarrowtail u, \\ (\mathtt{ito}^a_u(\alpha), \mathtt{view}^a_{\mathtt{dom}(a)}(\alpha a), a) & \text{otherwise.} \end{cases}$$

Taking these functions as X in the above pattern for action-observed security yields the definitions of TO-security and ITO-security in the action-observed case. (The reader may note some subtle differences between these definitions in the state- and action-observed cases. We refer to [Mey07] for an explanation of these differences.)

These definitions of security on action-observed systems may be shown to be related to the similarly named definitions on state-observed systems, by means of mapping from the action-observed to the state-observed domain. Given an action-observed machine $M = \langle S, s_0, A, \mathtt{step}, \mathtt{out}, \mathtt{dom}\rangle$ with domains D and outputs O, define the state observed machine $F_{as}(M) = \langle S', s'_0, A, \mathtt{step}', \mathtt{obs}, \mathtt{dom}\rangle$ by

1. $S' = S \times (D \rightarrow O \cup \{\bot\})$,
2. $s'_0 = (s_0, f_0)$, where $f_0(d) = \bot$ for all $d \in D$,
3. $\mathtt{step}'((s, f), a)) = (\mathit{step}(s, a), f[\mathtt{dom}(a) \rightarrow \mathtt{out}(s, a)])$,
4. $\mathtt{obs}_u((s, f)) = f(u)$.

Here, we write $f[u \mapsto x]$ for the function f' that is identical to f except that $f'(u) = x$. Intuitively, in a state (s, f), the value $f(d)$ for a domain d represents the observation most recently obtained in domain d, and is $\bot$ if there has been no observation in domain d.

The following result states relationships between the definitions of security on the two types of model under this mapping.

Theorem 8 *[Mey07] Let X be any of P, TO, ITO, TA, or IP. Then an action-observed machine M is X-secure with respect to a policy $\rightarrowtail$ (using the action-observed definitions) iff $F_{as}(M)$ is X-secure with respect to $\rightarrowtail$ (using the state-observed definitions).*

We will use this result to derive a result on the soundness of architectural refinement that similar to Theorem 4, but for action-observed systems.

First, we define the result of applying a refinement mapping to an action-observed system. Let $M = (S, S_0, A, \mathtt{step}, \mathtt{out}, \mathtt{dom})$ be an action-observed system with set of domains D_1. Let $r : D_1 \to D_2$. Then we define the system $r(M) = (S, S_0, A, \mathtt{step}, \mathtt{out}, \mathtt{dom}')$ simply by defining $\mathtt{dom}'(a) = r(\mathtt{dom}(a))$ for all $a \in A$; all other components are exactly as in M.

The following result characterizes how this operation relates to the mapping from action-observed to state-observed systems.

Lemma 3 *Let $r : D_1 \to D_2$, and let M be an action-observed system with set of domains D_1. Then $F_{as}(r(M)) \leq^{\mathtt{res}} r(F_{as}(M))$, and for all sequences $\alpha, \beta \in A^*$, and domains $u \in D_2$ we have that $\mathtt{view}_u^{F_{as}(r(M))}(\alpha) = \mathtt{view}_u^{F_{as}(r(M))}(\beta)$ implies $\mathtt{view}_u^{r(F_{as}(M))}(\alpha) = \mathtt{view}_u^{r(F_{as}(M))}(\beta)$.*

Proof: We first characterize the observations in the systems $F_{as}(r(M))$ and $r(F_{as}(M))$. Let $\mathtt{dom} : A \to D_1$ be the domain function of M.

Note that in $r(M)$, the output of an action a is observed by domain $r(\mathtt{dom}(a))$. Since the observation made by domain u in the state reached after a sequence α in $F_{as}(r(M))$ is the output obtained by the latest action of domain u, this observation is the output obtained from the latest action a in α with $\mathtt{dom}(a) \in r^{-1}(u)$.

On the other hand, in $F_{as}(M)$ the observation of domain v is the output obtained from the latest v action, so in $r(F_{as}(M))$, the observation of domain u in the state reached after a sequence α is the mapping taking $v \in r^{-1}(u)$ to the output of the latest action a in α with $\mathtt{dom}(a) = v$.

To see that $F_{as}(r(M)) \leq^{\mathtt{res}} r(F_{as}(M))$, suppose that $\mathtt{res}_u(\alpha) = \mathtt{res}_u(\beta)$ and $\mathtt{obs}_u^{r(F_{as}(M))}(s_0 \cdot \alpha) = \mathtt{obs}_u^{r(F_{as}(M))}(s_0 \cdot \beta)$. Then the most recent action of domain v with $r(v) = u$ is the same in α and β. Thus,

$$\begin{aligned} \mathtt{obs}_u^{F_{as}(r(M))}(s_0 \cdot \alpha) &= \mathtt{obs}_u^{r(F_{as}(M))}(s_0 \cdot \alpha)(v) \\ &= \mathtt{obs}_u^{r(F_{as}(M))}(s_0 \cdot \beta)(v) \\ &= \mathtt{obs}_u^{F_{as}(r(M))}(\beta), \end{aligned}$$

as required.

To see that $\mathtt{view}_u^{F_{as}(r(M))}(\alpha) = \mathtt{view}_u^{F_{as}(r(M))}(\beta)$ implies $\mathtt{view}_u^{r(F_{as}(M))}(\alpha) = \mathtt{view}_u^{r(F_{as}(M))}(\beta)$, we define for each $u \in D_2$ a mapping G_u taking views of $r(F_{as}(M))$ to views of $F_{as}(r(M))$, by the following induction. Note that, by construction, the views of $r(F_{as}(M))$ and $F_{as}(r(M))$ have the property that they alternate actions and observations, i.e., there are no adjacent observations. (This is because actions a with $r(\mathtt{dom}(a)) \neq u$ do not produce output in any domains observable to u, hence u observations are invariant under such actions, and repeated observations are removed by the absorptive concatenation operator.) The base case is given by $G_u(\perp) = \perp^{r^{-1}(u)}$, where we write $\perp^S$ for the function $\perp$ with domain S and constant value $\perp$. For σ a view, $a \in A$ and o an observation in M, the inductive case is given by $G_u(\sigma a o) = G_u(\sigma) a O'$, where $O' = O[\mathtt{dom}(a) \mapsto o]$ and O is the final observation in $G_u(\sigma)$, which is a mapping from $r^{-1}(u)$ to observations in M.

We claim that $G_u(\mathtt{view}_u^{F_{as}(r(M))}(\alpha)) = \mathtt{view}_u^{r(F_{as}(M))}(\alpha)$ for all $\alpha \in A^*$, from which it follows that $\mathtt{view}_u^{F_{as}(r(M))}(\alpha) = \mathtt{view}_u^{F_{as}(r(M))}(\beta)$ implies that $\mathtt{view}_u^{r(F_{as}(M))}(\alpha) = \mathtt{view}_u^{r(F_{as}(M))}(\beta)$. The proof of the claim is by induction. The base case is trivial. For the inductive step, we consider two cases. First, suppose that $\mathtt{dom}(a) \neq u$. Then

$$\begin{array}{rcll} G_u(\mathtt{view}_u^{F_{as}(r(M))}(\alpha a)) & = & G_u(\mathtt{view}_u^{F_{as}(r(M))}(\alpha)) & \\ & = & \mathtt{view}_u^{r(F_{as}(M))}(\alpha) & \text{(by induction)} \\ & = & \mathtt{view}_u^{r(F_{as}(M))}(\alpha a). & \end{array}$$

Alternately, if $\mathtt{dom}(a) = u$, then

$$G_u(\mathtt{view}_u^{F_{as}(r(M))}(\alpha a)) = G_u(\mathtt{view}_u^{F_{as}(r(M))}(\alpha) a o))$$

where $o = \mathtt{out}(s_0 \cdot \alpha, a)$ (in M). Let O be the final observation in $G_u(\mathtt{view}_u^{F_{as}(r(M))}(\alpha))$, and $O' = O[\mathtt{dom}(a) \mapsto o]$. Then, by induction,

$$\begin{array}{rcl} G_u(\mathtt{view}_u^{F_{as}(r(M))}(\alpha) a o) & = & G_u(\mathtt{view}_u^{F_{as}(r(M))}(\alpha)) a O' \\ & = & \mathtt{view}_u^{F_{as}(r(M))}(\alpha) a O'. \end{array}$$

In particular, since O is also the final observation in $\mathtt{view}_u^{F_{as}(r(M))}(\alpha)$, we have that $O = \mathtt{obs}_u^{F_{as}(r(M))}(s_0 \cdot \alpha)$, and, by construction, $O' = \mathtt{obs}_u^{F_{as}(r(M))}(s_0 \cdot \alpha a)$. We conclude that $G_u(\mathtt{view}_u^{F_{as}(r(M))}(\alpha a)) = \mathtt{view}_u^{r(F_{as}(M))}(\alpha a)$, as required. □

We now obtain the following:

Corollary 2 *Let M be an action-observed system with domains D_1, let $r : D_1 \to D_2$, and let $\rightarrowtail$ be a policy on D_2. For X any of P, TO, ITO, TA or IP, if $r(F_{as}(M))$ is X-secure with respect to $\rightarrowtail$ then $F_{as}(r(M))$ is X-secure with respect to $\rightarrowtail$.*

Proof: It is easily checked that for X equal to any of `purge`, `ta` or `ipurge`, if $X_u(\alpha) = X_u(\beta)$ then $\mathtt{res}_u(\alpha) = \mathtt{res}_u(\beta)$. Hence from $F_{as}(r(M)) \leq^{\mathtt{res}} r(F_{as}(M))$ we obtain that $F_{as}(r(M)) \leq^X r(F_{as}(M))$ for these values of X. Hence by Lemma 3 and Corollary 1, we obtain that if $r(F_{as}(M))$ is X-secure with respect to $\rightarrowtail$ then $F_{as}(r(M))$ is X-secure with respect to $\rightarrowtail$.

For the case of X equal to `to` or `ito`, we obtain this conclusion from Proposition 3 and Lemma 3. □

We may now conclude that each of our definitions of security is preserved under refinement of action-observed systems.

Corollary 3 *Let $\mathcal{A}_1 \leq_r \mathcal{A}_1$, and let X any of P, TO, ITO, TA or IP. If M is an action-observed system and M is X-compliant with $\mathcal{A}_1$ then $r(M)$ is X-compliant with $\mathcal{A}_2$.*

Proof: We have the following chain of implications:

1. M is X-secure with respect to $\rightarrowtail_1$
2. implies $F_{as}(M)$ is X-secure with respect to $\rightarrowtail_1$ (by Theorem 4)
3. implies $r(F_{as}(M))$ is X-secure with respect to $\rightarrowtail_2$ (by Theorem 8)
4. implies $F_{as}(r(M))$ is X-secure with respect to $\rightarrowtail_2$ (by Corollary 2)
5. implies $r(M)$ is X-secure with respect to $\rightarrowtail_2$ (by Theorem 4)

□

8 Related Work

To the best of our knowledge, our work is the first consideration of the relationship between architectural refinement and intransitive noninterference. However, both formal theories of architecture refinement and refinement of noninterference security properties have been presented in the past.

In general, work on architectural refinement [PR97, Bar05] is concerned with behavioural notions of refinement, and has not taken security into account. An interesting exception is a sequence of papers by Moriconi *et al.*, [MQR95, MQ94], who develop a very abstract formal account of architecture refinement in which architectural designs are represented as logical theories and refinement is treated as a mapping of the symbols of the abstract theory to those of the concrete theory that must satisfy the logical condition of being a *faithful interpretation*. In order to apply this account to a particular type of architectural design notation, it is necessary to concretize the abstract theory by giving both a syntax for the architectural elements in the notation, and to develop a logical theory that represents the semantics of this notation. (E.g. this is done in [MQ94] for dataflow and shared-memory architectural styles.) In [MQRG97], the framework is applied to establish security properties of a number of secure variants of the X/Open Distributed Transaction Processing architecture. The security policy considered here is the Bell La-Padula policy [BP76], which lacks the kind of information flow semantics that we have studied here, although it can be related to noninterference for transitive policies [Rus92]. It is not clear whether a concretization of the Moriconi *et al.* theory could be developed that would enable it to represent the content of our results, but this would be an interesting topic for further study. Zhou and Alves-Foss [ZAF06] have also proposed a number of architecture refinement patterns for Multi-Level Secure systems development, but do not provide any formal semantics for their work.

The other area of related work, dealing with preservation of noninterference properties under notions of refinement, has typically been concerned with just the (transitive) two-domain policy stating that High level information may not flow to Low, rather than with the more general intransitive policies that we have

considered in this paper. (All of the literature discussed below differs from our work in this regard.)

That we have obtained positive results concerning refinement of security properties may be surprising to those familiar with the literature on formal security properties, where it is folklore that such properties are *not* preserved under refinement [Jac89], a fact known as the "refinement paradox". However, our notions of refinement differ from the notions of refinement usually studied. Refinement is usually understood as a reduction in the set of possible behaviours of the system, which would be contrary to our assumptions that systems are *input-enabled* (actions always enabled) and *deterministic*.

It is possible to identify conditions under which reduction of the possible behaviours of a system preserves information flow security properties. Jacob [Jac89] presents a method in which an insecure system is first developed using a standard refinement methodology for functional properties, then made secure by a further deletion of behaviors in a fixpoint calculation. It is not guaranteed that this last step terminates, nor that it produces a useful system. Mantel [Man01] defines refinement operators that take as input a secure system, a set of transitions to be disabled, and a type of unwinding relation on the system that establishes the security property. The operators produce as output a refined system, as well as a new unwinding relation that establishes the security of the refined system. This is achieved by either disabling transitions other than those requested, or by maintaining some transitions whose disablement was requested. He considers a richer notion of information flow policy than we have treated, but with respect to a semantics that seems appropriate only for transitive policies. The practicality of these approaches has not been established.

A number of authors have also identified sufficient conditions under which data-refinement preserves transitive information flow policies [GCS91, O'H92]. Bossi *et al.* [BFPR03] develop conditions under which refinement of process algebra terms preserves bisimulation-based information flow security properties using a simulation-based notion of refinement. Roscoe [Ros95] defines *Low-determinism*, a very strong notion of security, which is always preserved under refinement, but at the cost of a significantly restricted range of applicability. Some recent works have also sought to overcome the refinement paradox by drawing a distinction between specification-level non-determinism and non-determinism that is inherent in a system, with the latter preserved under refinement [SS06, Jür05, Bib06].

We note that the notions of refinement we have considered in this paper are somewhat different from the standard notion of refinement considered in the literature, which reduces the set of possible behaviours of a system. Information flow properties are not generally preserved under such refinements [Jac89], although some authors have tried to obtain such results by modification of the refined system [Jac89, Man01]. Others have identified sufficient conditions for behavioral refinement to preserve information flow properties [GCS91, O'H92, BFPR03, Ros95]. Some recent works have also sought to overcome the refinement paradox by drawing a distinction between specification-level non-determinism and non-determinism that is inherent in a system, with

the latter preserved under refinement [SS06, Jür05, Bib06].

Another area in which refinement has been considered in the context of information flow security concerns refinement at the level of extended program notations [Mor09]. This approach aims to preserve the ignorance of a given agent during refinement, and has the advantage of providing an expressive framework for representing what an agent does not know. (We remark that [Mor09] contains a result related to the concerns of Section 6, showing that security is preserved by reducing the content of observations.) On the other hand, this line of work does not deal with the concerns of causal structure in a multi-agent setting that are our focus in the present paper.

9 Conclusion

This work is a contribution towards a formal design theory for information flow secure systems. Much remains to be done to realise such a theory. The notions of security studied here are based on an asynchronous modelling of systems - they do not take into account issues such as timing attacks and scheduling. Probabilistic reasoning, which is critical in practical security settings, is also ignored. Suitable variants of our definitions for the semantics of intransitive noninterference policies that take these concerns into account remain to be developed. We have considered only refinement of systems as a whole; it would be desirable to develop also an account of composition of architectural designs and policies, and to study how these interact with refinement, so that refinement can be carried out at the component level. Integrating our approach with approaches to refinement operating at lower levels of system description would also be desirable. We hope to address issues such as these in future work. Ultimately, one would like to have a theoretically sound and tool supported methodology that enables a system to be developed from the very abstract architectural level we have considered in this paper, all the way through to code running on particular hardware configurations.

References

[AFHOT06] J. Alves-Foss, W.S. Harrison, P. Oman, and C. Taylor. The MILS architecture for high-assurance embedded systems. *International Journal of Embedded Systems*, 2(3/4):239–47, Feb 2006.

[Bar05] M. A. Barbosa. A refinement calculus for software components and architectures. *ACM SIGSOFT Software Engineering Notes*, 30(5), September 2005.

[BFPR03] A. Bossi, R. Focardi, C. Piazza, and S. Rossi. Refinement operators and information flow security. In *Proc. Int. Conf. on Software Engineering and Formal Methods*, pages 44–53, 2003.

[Bib06] David Bibighaus. *Applying the doubly labeled transition system to the refinement paradox*. PhD thesis, Naval Postgraduate School, Monterey, 2006.

[BP76] D.E. Bell and L.J. La Padula. Secure computer system: unified exposition and multics interpretation. Technical Report ESD-TR-75-306, Mitre Corporation, Bedford, M.A., March 1976.

[GCS91] J. Graham-Cunning and J. Sanders. On the refinement of noninterference. In *Proc. IEEE Computer Security Foundations Workshop*, pages 35–42, 1991.

[GM82] J.A. Goguen and J. Meseguer. Security policies and security models. In *Proc. IEEE Symp. on Security and Privacy*, pages 11–20, Oakland, 1982.

[GM84] J.A. Goguen and J. Meseguer. Unwinding and inference control. In *IEEE Symp. on Security and Privacy*, 1984.

[HY87] J.T. Haigh and W.D. Young. Extending the noninterference version of MLS for SAT. *IEEE Trans. on Software Engineering*, SE-13(2):141–150, Feb 1987.

[Jac89] J. Jacob. On the derivation of secure components. In *Proc. IEEE Symp. on Security and Privacy*, pages 242–247, 1989.

[Jür05] J. Jürjens. *Secure Systems Development with UML*. Springer, 2005.

[Man01] H. Mantel. Preserving information flow properties under refinement. In *Proc. IEEE Symp. Security and Privacy*, pages 78–91, 2001.

[Mey07] R. van der Meyden. A comparison of semantic models of intransitive noninterference. submitted for publication, copy at `http://www.cse.unsw.edu.au/~meyden`, Dec 2007.

[Mey08] R. van der Meyden. What, indeed, is intransitive noninterference? (submitted for publication, copy at `http://www.cse.unsw.edu.au/~meyden` – an extended abstract of this paper appears in Proc. ESORICS 2007), Jan 2008.

[Mor09] Carroll Morgan. The shadow knows: Refinement and security in sequential programs. *Sci. Comput. Program.*, 74(8):629–653, 2009.

[MQ94] M. Moriconi and X. Qian. Correctness and composition of software architectures. In *Proc. 2nd ACM SIGSOFT Symposium on Foundations of Software Engineering*, pages 164–174, 1994.

[MQR95] M. Moriconi, X. Qian, and R.A. Riemenschneider. Correct architecture refinement. *IEEE Transactions on Software Engineering*, 21(4):356–372, April 1995.

[MQRG97] M. Moriconi, X. Qian, R. A. Riemenschneider, and L. Gong. Secure software architectures. In *Proc. IEEE Symp. on Security and Privacy*, pages 84–893, 1997.

[O'H92] C. O'Halloran. Refinement and confidentiality. In *Fifth Refinement Workshop*, pages 119–139. British Computer Society, 1992.

[PR97] J. Philipps and B. Rumpe. Refinement of information flow architectures. In *Proc. 1st IEEE Int. Conf. on Formal Engineering Methods*, pages 203 – 212, 1997.

[RG99] A. W. Roscoe and M. H. Goldsmith. What is intransitive noninterference? In *IEEE Computer Security Foundations Workshop*, pages 228–238, 1999.

[Ros95] A.W. Roscoe. CSP and determinism in security modelling. In *Proc. IEEE Symp. on Security and Privacy*, pages 114–221, 1995.

[RR83] J.M. Rushby and R. Randell. A distributed secure system. *IEEE Computer*, 16(7):55–67, 1983.

[Rus92] J. Rushby. Noninterference, transitivity, and channel-control security policies. Technical Report CSL-92-02, SRI International, Dec 1992.

[SS06] F. Seehusen and K. Stolen. Information flow property preserving transformation of UML interaction diagrams. In *Proc. ACM symposium on access control models and technologies*, pages 150 – 159, 2006.

[VBC+05] W.M. Vanfleet, R.W. Beckworth, B. Calloni, J.A. Luke, C. Taylor, and G. Uchenick. MILS:architecture for high assurance embedded computing. *Crosstalk: The Journal of Defence Engineering*, pages 12–16, Aug 2005.

[ZAF06] J. Zhou and J. Alves-Foss. Architecture-based refinements for secure computer system design. In *Proc. Policy, Security and Trust*, Nov 2006.